



SteelHead™ Deployment Guide

Including the SteelCentral™ Controller for SteelHead
Mobile

July 2018



© 2018 Riverbed Technology, Inc. All rights reserved.

Riverbed and any Riverbed product or service name or logo used herein are trademarks of Riverbed. All other trademarks used herein belong to their respective owners. The trademarks and logos displayed herein cannot be used without the prior written consent of Riverbed or their respective owners.

Akamai® and the Akamai wave logo are registered trademarks of Akamai Technologies, Inc. SureRoute is a service mark of Akamai. Apple and Mac are registered trademarks of Apple, Incorporated in the United States and in other countries. Cisco is a registered trademark of Cisco Systems, Inc. and its affiliates in the United States and in other countries. EMC, Symmetrix, and SRDF are registered trademarks of EMC Corporation and its affiliates in the United States and in other countries. IBM, iSeries, and AS/400 are registered trademarks of IBM Corporation and its affiliates in the United States and in other countries. Juniper Networks and Junos are registered trademarks of Juniper Networks, Incorporated in the United States and other countries. Linux is a trademark of Linus Torvalds in the United States and in other countries. Microsoft, Windows, Vista, Outlook, and Internet Explorer are trademarks or registered trademarks of Microsoft Corporation in the United States and in other countries. Oracle and JInitiator are trademarks or registered trademarks of Oracle Corporation in the United States and in other countries. UNIX is a registered trademark in the United States and in other countries, exclusively licensed through X/Open Company, Ltd. VMware, ESX, ESXi are trademarks or registered trademarks of VMware, Inc. in the United States and in other countries.

This product includes Windows Azure Linux Agent developed by the Microsoft Corporation (<http://www.microsoft.com/>). Copyright 2016 Microsoft Corporation.

This product includes software developed by the University of California, Berkeley (and its contributors), EMC, and Comtech AHA Corporation. This product is derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

The SteelHead Mobile Controller (virtual edition) includes VMware Tools. Portions Copyright © 1998-2016 VMware, Inc. All Rights Reserved.

NetApp Manageability Software Development Kit (NM SDK), including any third-party software available for review with such SDK which can be found at <http://communities.netapp.com/docs/DOC-1152>, and are included in a NOTICES file included within the downloaded files.

For a list of open source software (including libraries) used in the development of this software along with associated copyright and license agreements, see the Riverbed Support site at <https://support.riverbed.com>.

This documentation is furnished "AS IS" and is subject to change without notice and should not be construed as a commitment by Riverbed. This documentation may not be copied, modified or distributed without the express authorization of Riverbed and may be used only in connection with Riverbed products and services. Use, duplication, reproduction, release, modification, disclosure or transfer of this documentation is restricted in accordance with the Federal Acquisition Regulations as applied to civilian agencies and the Defense Federal Acquisition Regulation Supplement as applied to military agencies. This documentation qualifies as "commercial computer software documentation" and any use by the government shall be governed solely by these terms. All other use is prohibited. Riverbed assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation.



Riverbed Technology
680 Folsom Street
San Francisco, CA 94107
www.riverbed.com

Part Number
712-00003-25

Contents

Welcome	15
About this guide	15
Audience	16
Types of SteelHeads	16
Document conventions	17
Documentation and release notes	17
Contacting Riverbed	17
What's new	18
 1 - Optimization Techniques and Design Fundamentals	 19
How SteelHeads optimize data	19
Data streamlining	20
Transport streamlining	21
Application streamlining	26
Management streamlining	27
RiOS data store synchronization	27
RiOS data store synchronization requirements	28
RiOS data store error alarms	28
Choosing the right SteelHead model	28
Deployment modes for the SteelHead	31
Autodiscovery protocol	31
Original autodiscovery process	33
Configuring enhanced autodiscovery	35
Autodiscovery and firewall considerations	35
Removal of the Riverbed TCP option probe	35
Stateful firewall device in a multiple in-path environment	35
Multiple in-path discovery behavior	36
Controlling optimization	38
In-path rules	38
Default in-path rules	39
Peering rules	39
Kickoff and automatic kickoff features	40
Controlling optimization configuration examples	42

Configuring high-bandwidth, low-latency environment	42
Configuring pass-through transit traffic.....	44
Fixed-target in-path rules	48
Configuring a fixed-target in-path rule for an in-path deployment	49
Fixed-target in-path rule for an out-of-path deployment	50
Best practices for SteelHead deployments.....	51
2 - Network Integration Tools	53
Redundancy and clustering	53
Physical in-path deployments	53
Virtual in-path deployments.....	54
Out-of-path deployments.....	54
Fail-to-wire and fail-to-block	55
Overview of link state propagation.....	55
Connection forwarding	56
Configuring connection forwarding.....	57
Multiple-interface support within connection forwarding	58
Failure handling within connection forwarding.....	58
Connection-forwarding neighbor latency	59
Overview of simplified routing	60
3 - WAN Visibility Modes.....	63
Overview of WAN visibility	63
Correct addressing	64
Transparent addressing.....	65
Port transparency	66
Full address transparency.....	67
Full address transparency with forward reset	69
Implications of transparent addressing.....	71
Stateful systems	71
Network design issues	71
Integration into networks using NAT	75
Out-of-band connection.....	84
Overview of OOB connections and addressing modes.....	85
Configuring OOB connection destination transparency	85
Configuring OOB connection full transparency	86
Configuring WAN visibility modes	87
4 - Topology	91
Introduction to the topology concept.....	91
Defining a network	91
Defining a site	93

Configuring the local site.....	94
Configuring the default site	96
5 - Application Definitions.....	97
Applications	97
Defining an application	98
Application properties.....	100
Application Flow Engine	102
Overview of the Application Flow Engine	102
AFE and Microsoft Lync 2010 and 2013	104
Creating host labels	105
Creating port labels.....	108
Creating domain labels.....	109
6 - QoS configuration and integration.....	111
Overview of Riverbed QoS.....	112
QoS concepts	114
Overview of QoS concepts	114
QoS rules.....	116
QoS classes.....	117
QoS profiles.....	124
Configuring QoS.....	128
QoS configuration workflow	128
Enabling QoS.....	128
QoS default classes.....	129
Inbound QoS.....	130
Introduction to inbound QoS.....	131
Assigning an inbound QoS profile to a site	131
LAN bypass	132
QoS for IPv6.....	132
QoS in virtual in-path and out-of-path deployments	133
QoS in multiple SteelHead deployments.....	133
QoS and multiple WAN interfaces	134
Integrating SteelHeads into existing QoS architectures.....	134
WAN-side traffic characteristics and QoS	135
QoS integration techniques	135
QoS marking.....	137
QoS enforcement best practices	139
Upgrading to RiOS 9.0.....	139
Guidelines for the maximum number of QoS classes, sites, and rules.....	140

7 - QoS Configuration Examples	143
Configuring QoS using best practices	143
Example QoS scenario.....	143
Configuring QoS on the data center SteelHead	146
Configuring applications	147
Creating QoS profiles.....	149
Configuring topology	156
Enabling QoS on the SteelHead	160
Configuring QoS marking on SteelHeads	162
Configuring QoS and MX-TCP	165
8 - Path Selection	169
Overview of path selection	170
Path selection implementation.....	171
Path selection workflow	171
Example of a path selection implementation.....	176
Identifying traffic flow candidates.....	178
Configuring path selection	179
Valid path selection deployment design examples	180
Basic multiple route path deployment.....	181
Complex parallel path deployment	184
Complex single in-path interface deployment	186
Serial deployment	189
Firewall path traversal deployment	189
Path selection and virtual in-path deployment	192
Design validation	192
Design considerations	194
9 - Physical in-path deployments.....	197
Overview of in-path deployment.....	197
Logical in-path interface	198
In-path IP address selection	199
In-path default gateway and routing	199
Failure modes	200
Fail-to-wire mode.....	201
Fail-to-block mode.....	202
Configuring failure modes	202
Configuring link state propagation.....	203
EtherChannel	203
Cabling and duplex.....	205
Choosing the correct cables	205
Duplex configuration	206

Troubleshooting cable and duplex issues	207
Physical in-path deployment configuration examples	208
Configuring a basic physical in-path deployment	208
Configuring a physical in-path with dual links deployment.....	211
Configuring a serial cluster deployment with multiple links	211
In-path redundancy and clustering examples.....	213
Primary and backup deployments	213
Serial cluster deployments	215
Configuring simplified routing	219
Multiple WAN router deployments	221
Configuring multiple WAN router deployments without connection forwarding	223
Configuring multiple WAN router deployments with connection forwarding	228
802.1Q trunk deployments	236
Overview of VLAN trunk	237
Configuring a SteelHead on an 802.1Q trunk link.....	238
Capturing network traces using tcpdump.....	239
Layer-2 WAN deployments	239
Layer-2 WANs.....	239
Broadcast Layer-2 WANs.....	240
VLAN bridging deployments.....	241
Overview of VLAN bridging deployment	241
VLAN bridging considerations	242
VLAN bridging variations	242
10 - Virtual In-Path Deployments	245
Overview of virtual in-path deployment	245
Configuring an in-path, load-balanced, Layer-4 switch deployment	246
Configuring flow data exports in virtual in-path deployments	248
11 - WCCP Virtual In-Path Deployments	249
Overview of WCCP.....	249
WCCP fundamentals.....	250
Service groups	250
Assignment methods	251
Redirection and return methods.....	254
WCCP clustering and failover.....	255
Multiple in-path WCCP.....	256
Advantages and disadvantages of WCCP	256
Configuring WCCP.....	257
Basic steps for configuring WCCP.....	258
Configuring a simple WCCP deployment	259
Adding a SteelHead to an existing WCCP deployment.....	262
Configuring a WCCP high availability deployment	264

Configuring a basic WCCP router	272
Configuring additional WCCP features	273
Specifying the service group password	274
Configuring multicast groups	274
Configuring group lists to limit service group members	275
Configuring access control lists	276
Configuring load balancing in WCCP	279
Flow data in WCCP	282
Verifying and troubleshooting WCCP configurations	282
12 - Policy-Based Routing Virtual In-Path Deployments	287
Overview of PBR	287
PBR failover and Cisco Discovery Protocol	288
Alternate PBR failover mechanisms	289
Connecting the SteelHead in a PBR deployment	290
Configuring PBR	290
Overview of configuring PBR	290
Configuring a SteelHead to directly connect to the router	291
Configuring a SteelHead to connect to a Layer-2 switch	292
Configuring a SteelHead to connect to a Layer-3 switch	294
Configuring a SteelHead with object tracking	295
Configuring a SteelHead with multiple PBR interfaces	296
Configuring multiple SteelHeads to connect to multiple routers	297
Configuring PBR for load balancing WAN circuits	300
Configuring local PBR for ICMP redirection in a mixed MTU environment	304
Exporting flow data and virtual in-path deployments	306
13 - IPv6	307
Overview of IPv6	307
RiOS RFC compliance and feature compatibility	308
IPv6 addressing	310
Traffic interception	311
In-path rules	313
Deployment options	313
Configuring an in-path SteelHead IPv6 deployment	313
Configuring a SteelHead serial cluster IPv6 deployment	315
Configuring a connection forwarding and SteelHead IPv6 deployment	316
Configuring a virtual in-path SteelHead IPv6 deployment	317
Configuring a fixed-target rule SteelHead IPv6 deployment	319
Protocol support	320
Verification and troubleshooting	320

14 - Packet Mode Optimization	323
Overview of packet mode optimization	323
Comparison with TCP proxy mode optimization	323
Configuring packet mode optimization	324
Design considerations	328
Best practices for packet mode optimization	328
15 - Satellite Optimization	329
Overview of satellite networks	329
Impact of latency	330
Impact of loss.....	330
Satellite transport options	331
Overview of SCPS	331
SCPS benefits	332
Common uses for SCPS	332
SCPS and SteelHeads	333
TCP optimization for satellite environments	333
SCPS discovery	334
Transport optimization for satellite environments	335
Configuring automatic detect TCP optimization.....	338
Integrating the SteelHead with existing satellite modem TCP acceleration	339
Licensing SCPS on a SteelHead	340
Configuring satellite optimization features	340
Configuring transport optimization	340
Configuring rate pacing	344
Configuring single-ended connection rule table settings.....	345
Configuring single-ended rules	347
Verification and troubleshooting	350
Analyzing connection optimization information.....	350
Analyzing packets for discovery probe stripping.....	354
Understanding the health of the satellite signal.....	356
Potential under performance due to short bottleneck buffer.....	357
Potential performance impact of loss at the start of flow	358
Variance in SCPS performance	359
16 - VPN Routing and Forwarding	361
NSV with VRF Select	361
Virtual routing and forwarding	362
NSV with VRF Select	363
IOS requirements	364
Prerequisites for NSV	364
Example NSV network deployment.....	364
Configuring NSV	366
VRF-aware WCCP	372

VRF-aware WCCP design examples	373
VRF-aware WCCP best practices	384
17 - Out-of-Path Deployments	387
Overview of out-of-path deployment	387
Limitations of out-of-path deployments.....	388
Configuring out-of-path deployments	389
18 - Data Protection Deployments.....	391
Overview of data protection	391
Planning for a data protection deployment.....	392
LAN-side throughput and data reduction requirements	393
Predeployment questionnaire	394
Configuring SteelHeads for data protection.....	398
Adaptive data streamlining feature settings.....	399
CPU settings	400
Best practices for data streamlining and compression.....	401
MX-TCP settings	402
SteelHead WAN buffer settings	402
Router WAN buffer settings	403
Common data protection deployments.....	403
Remote office, branch office backups	403
Network attached storage replication	404
Storage area network replication	405
Designing for scalability and high availability	405
Overview of N+M architecture	405
Using MX-TCP in N+M deployments	405
Enhanced visibility and control for SnapMirror.....	407
Troubleshooting and fine-tuning.....	409
Third-party interoperability	410
19 - Storage Area Network Replication.....	411
Overview of SAN replication	411
Storage optimization modules.....	412
FCIP optimization module	412
SRDF optimization module.....	415
Best practices for SAN replication using TCP/IP	420
Best practices for SAN replication using Cisco MDS FCIP.....	421
FCIP profiles.....	421
FCIP tunnels	422
Configuring a Cisco MDS FCIP deployment.....	422
Best practices for RiOS 5.5.3 and later with Cisco MDS FCIP configuration	423

20 - Authentication, Security, Operations, and Monitoring	425
Overview of secure transport.....	425
Overview of authentication.....	426
Authentication features	427
Configuring SAML.....	428
Configuring a RADIUS server.....	429
Configuring a RADIUS server with FreeRADIUS	429
Configuring RADIUS authentication in the SteelHead	430
Configuring RADIUS CHAP authentication	431
Configuring a TACACS+ server	432
Configuring TACACS+ with Cisco Secure Access Control Servers	432
Configuring TACACS+ authentication in the SteelHead.....	432
Securing SteelHeads	433
Overview of securing SteelHeads	433
Best practices for securing access to SteelHeads.....	433
Best practices for enabling SteelHead security features.....	440
Best practices for policy controls	444
Best practices for security monitoring	444
Configuring SSL certificates for web user interface.....	445
Changing encryption for domain replication passwords	446
REST API access.....	446
Capacity planning.....	446
Model characteristics	447
Admission control	448
Overview of exporting flow data.....	450
SNMP monitoring	451
Configuring SNMPv3 authentication and privacy	457
21 - Troubleshooting SteelHead Deployment Problems	463
Common deployment issues.....	463
Duplex mismatches.....	463
Network asymmetry.....	466
Unknown (or unwanted) SteelHead appears on the current connections list	468
Outdated antivirus software.....	468
Packet ricochets	469
Router CPU spikes after WCCP configuration	469
Server Message Block signed sessions	471
Unavailable opportunistic locks	475
Underutilized fat pipes.....	476
MTU sizing.....	476
MTU issues.....	477
Determining MTU size in deployments.....	478
Connection-forwarding MTU considerations	479

22 - SteelHead and AppResponse Integration	481
Overview of SteelHead and AppResponse integration	481
AppResponse and SteelHead deployment scenarios	483
Data center deployment	483
Cloud deployment	486
23 - Deploying SteelHead-v in OpenStack	487
What you need	487
Deploying SteelHead-v appliances in OpenStack	487
Deleting an OpenStack instance.....	495
Using a Heat template to deploy SteelHead-v appliances.....	496
Heat template deployment guidelines	496
Heat template example	496
Creating an OpenStack stack from a Heat template	506
Deleting an OpenStack stack created by a Heat template	506
Creating OpenStack flavors: CLI examples.....	506
24 - SteelCentral Controller for SteelHead Mobile Deployments.....	511
Overview of SteelCentral Controller for SteelHead Mobile deployment.....	511
Basic setup for deploying Mobile Controller	511
Mobile Controller with VPN deployments	512
Mobile Controller with firewall deployments	513
Branch office and remote access deployments	514
Multiple Mobile Controller deployments.....	514
Overview of multiple Mobile Controller deployments.....	515
Mobile Controller Concurrent User Limits.....	517
Configuring multiple Mobile Controllers for redundancy.....	517
Preparing to join Mobile Controllers in a high-availability cluster	519
Sizing considerations in a high-availability cluster.....	520
Endpoint license pooling	521
Communication between HA cluster members	523
Ports used with Mobile Controllers and SteelHead Mobiles.....	524
Interaction between Mobile Controllers and SteelHead Mobile clients.....	524
Location awareness.....	525
Overview of location awareness	525
Branch warming	526
SSL with SteelCentral Controller for SteelHead Mobile	528
Traditional SSL optimization	528
Advanced high-security SSL optimization	529
Configuring SteelCentral Controller for SteelHead Mobile and SSL	530
Using SteelHead Mobile with SSL proxy devices.....	530
Supported TLS versions with SteelHead Mobile	531
Multiple Mobile Controllers and SSL	531
Mobile Controller best practices and other considerations	531

Deployment scenarios.....	532
Management best practices.....	533
Migration Mobile Controller hardware	534
Licensing best practices	535
Antivirus software	535
Signed SMB support	535
SSL client authentication support.....	536
SMC and Federal Information Processing Standard (FIPS)	536
Optimization before user log in.....	537

Welcome

About this guide

This guide describes why and how to configure the SteelHead in complex in-path and out-of-path deployments such as failover, multiple routing points, static clusters, connection forwarding, WCCP, Layer 4, PBR, and PFS. It also includes chapters on the SteelCentral Controller for SteelHead and SteelCentral Controller for SteelHead Mobile software.

This guide includes information relevant to the following products and product features:

- Riverbed Optimization System (RiOS)
- Riverbed SteelHead (SteelHead)
- Riverbed SteelHead CX (SteelHead CX)
- Riverbed SteelHead EX (SteelHead EX)
- Riverbed Virtual SteelHead (SteelHead-v)
- Riverbed Cloud SteelHead (SteelHead-c)
- Riverbed SteelHead SaaS (SteelHead SaaS)
- Riverbed Cloud Portal (Riverbed Cloud Portal)
- Riverbed SteelCentral Controller for SteelHead (Controller or SCC)
- Riverbed SteelCentral Controller for SteelHead Virtual Edition (SCC-VE)
- Riverbed SteelCentral Controller for SteelHead Mobile software (SteelCentral Controller for SteelHead Mobile)
- Riverbed SteelCentral Controller for SteelHead Mobile appliance (Mobile Controller)
- Riverbed Mobile Controller (virtual edition) (Mobile Controller-v)
- Riverbed SteelHead Mobile (SteelHead Mobile)
- Riverbed SteelHead Interceptor (Interceptor)
- Riverbed Virtual Services Platform (VSP)
- Riverbed SteelFusion Core (Core)
- Riverbed SteelFusion Edge (Edge)
- Riverbed SteelCentral NetShark (NetShark)

- Riverbed SteelCentral NetProfiler (NetProfiler)
- SteelCentral AppResponse (AppResponse)

Audience

This guide is written for storage and network administrators familiar with administering and managing WANs using common network protocols such as TCP, CIFS, HTTP, FTP, and NFS.

You must also be familiar with:

- the Management Console. For details, see the *SteelHead User Guide*.
- connecting to the RiOS CLI. For details, see the *Riverbed Command-Line Interface Reference Manual*.
- the installation and configuration process for the SteelHead. For details, see the *SteelHead Installation and Configuration Guide* and the *SteelHead (Virtual Edition) Installation Guide*.
- the SteelHead Interceptor. For details, see the *SteelHead Interceptor Deployment Guide* and the *SteelHead Interceptor User Guide*.
- the SCC. For details, see the *SteelCentral Services Controller for SteelApp Deployment Guide* and the *SteelCentral Controller for SteelHead User Guide*.
- the Mobile Controller. For details, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Types of SteelHeads

The SteelHead product line includes several types of devices. Most of the information in the *SteelHead Deployment Guide* applies to the following appliances:

- **SteelHead CX** - is a WAN optimization-only device. Desktop models have two in-path interfaces. For details, see the *SteelHead User Guide* for the SteelHead CX.
- **SteelHead EX** - includes WAN optimization and VSP. The Riverbed Granite product family, which provides branch storage services, is available with an additional license. For details, see the *SteelHead User Guide* for the SteelHead EX (xx60). There is also SteelHead EX functionality in the Edge.
- **SteelHead (virtual edition) (SteelHead-v)** - is a virtualized version of the SteelHead that runs on VMware ESX/ESXi and the Cisco Services-Ready Engine (SRE) platform. For details, see the *SteelHead (Virtual Edition) Installation Guide*.
- **SteelHead-c** and **SteelHead SaaS** - are the SteelHeads for public cloud computing environments. You deploy the SteelHead-c and SteelHead SaaS differently from the SteelHead and the SteelHead-v. For details, see the *SteelHead Cloud Services User Guide* and the *SteelHead SaaS User Guide*.
- **SteelHead Mobile** - optimizes network traffic from remote users who are accessing the enterprise network using any type of remote access (dial-up, broadband, wireless, and so on). For details, see the *SteelCentral Controller for SteelHead Mobile User Guide*.
- **SteelCentral Controller for SteelHead Mobile (Mobile Controller)** - provides management functionality for SteelHead Mobiles. For details, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

- **SteelCentral Controller for SteelHead (SCC)** - provides management functionality for various Riverbed products, including SteelHeads, Mobile Controllers, and SteelHead Interceptors. For details, see the *SteelCentral Controller for SteelHead User Guide*.

For more information about the SteelHead family, go to http://www.riverbed.com/us/products/steelhead_appliance/.

Document conventions

This guide uses the following standard set of typographical conventions.

Convention	Meaning
<i>italics</i>	Within text, new terms and emphasized words appear in <i>italic</i> typeface.
boldface	Within text, CLI commands, CLI parameters, and REST API properties appear in bold typeface.
Courier	Code examples appear in Courier font: <pre>amnesiac > enable amnesiac # configure terminal</pre>
< >	Values that you specify appear in angle brackets: interface <ip-address>
[]	Optional keywords or variables appear in brackets: ntp peer <ip-address> [version <number>]
{ }	Elements that are part of a required choice appear in braces: {<interface-name> ascii <string> hex <string>}
	The pipe symbol separates alternative, mutually exclusive elements of a choice. The pipe symbol is used in conjunction with braces or brackets; the braces or brackets group the choices and identify them as required or optional: { delete <filename> upload <filename>}

Documentation and release notes

The most current version of all Riverbed documentation can be found on the Riverbed Support site at <https://support.riverbed.com>.

See the Riverbed Knowledge Base for any known issues, how-to documents, system requirements, and common error messages. You can browse titles or search for keywords and strings. To access the Riverbed Knowledge Base, log in to the Riverbed Support site at <https://support.riverbed.com>.

Each software release includes release notes. The release notes list new features, known issues, and fixed problems. To obtain the most current version of the release notes, go to the Software and Documentation section of the Riverbed Support site at <https://support.riverbed.com>.

Examine the release notes before you begin the installation and configuration process.

Contacting Riverbed

This section describes how to contact departments within Riverbed.

- **Technical support** - Problems installing, using, or replacing Riverbed products? Contact Riverbed Support or your channel partner who provides support. To contact Riverbed Support, open a trouble ticket by calling 1-888-RVBD-TAC (1-888-782-3822) in the United States and Canada or +1 415-247-7381 outside the United States. You can also go to <https://support.riverbed.com>.
- **Professional services** - Need help with planning a migration or implementing a custom design solution? Contact Riverbed Professional Services. Email proserve@riverbed.com or go to <http://www.riverbed.com/services/index.html>.
- **Documentation** - Have suggestions about Riverbed's online documentation or printed materials? Send comments to techpubs@riverbed.com.

What's new

Since the last release of the *SteelHead Deployment Guide*, the following information has been added or updated. For a list of new features available in RiOS Version 9.8, see the *SteelHead Installation and Configuration Guide*.

- **Updated** – **Chapter 20, “Authentication, Security, Operations, and Monitoring.”** Added a section about using Security Authentication Markup Language (SAML) and a section on Changing encryption for domain replication passwords from Data Encryption Standard (DES) to Advanced Encryption Standard (AES).

Optimization Techniques and Design Fundamentals

This chapter describes how the SteelHead optimizes data, the factors you need to consider when designing your SteelHead deployment, and how and when to implement the most commonly used SteelHead features.

This chapter includes the following sections:

- [“How SteelHeads optimize data” on page 19](#)
- [“RiOS data store synchronization” on page 27](#)
- [“Choosing the right SteelHead model” on page 28](#)
- [“Deployment modes for the SteelHead” on page 31](#)
- [“Autodiscovery protocol” on page 31](#)
- [“Autodiscovery and firewall considerations” on page 35](#)
- [“Multiple in-path discovery behavior” on page 36](#)
- [“Controlling optimization” on page 38](#)
- [“Controlling optimization configuration examples” on page 42](#)
- [“Fixed-target in-path rules” on page 48](#)
- [“Best practices for SteelHead deployments” on page 51](#)

How SteelHeads optimize data

The SteelHead optimizes data in the following ways:

- [“Data streamlining” on page 20](#)
- [“Transport streamlining” on page 21](#)
- [“Application streamlining” on page 26](#)
- [“Management streamlining” on page 27](#)

The causes for slow throughput in WANs are well known: high delay (round-trip time or latency), limited bandwidth, and chatty application protocols. Large enterprises spend a significant portion of their information technology budgets on storage and networks. Much of the budgets are spent to compensate for slow throughput by deploying redundant servers and storage and the required backup equipment. SteelHeads enable you to consolidate and centralize key IT resources to save money, simplify key business processes, and improve productivity.

RiOS is the software that powers the SteelHead and SteelCentral Controller for SteelHead Mobile. With RiOS, you can solve a range of problems affecting WANs and application performance, including:

- insufficient WAN bandwidth.
- inefficient transport protocols in high-latency environments.
- inefficient application protocols in high-latency environments.

RiOS intercepts client-server connections without interfering with normal client-server interactions, file semantics, or protocols. All client requests are passed through to the server normally, although relevant traffic is optimized to improve performance.

Data streamlining

With data streamlining, SteelHeads and SteelCentral Controller for SteelHead Mobile can reduce WAN bandwidth utilization by 65 to 98% for TCP-based applications. This section includes the following topics:

- [“Scalable data referencing” on page 20](#)
- [“Bidirectional synchronized RiOS data store” on page 21](#)
- [“Unified RiOS data store” on page 21](#)

Scalable data referencing

In addition to traditional techniques like data compression, RiOS also uses a Riverbed proprietary algorithm called scalable data referencing (SDR). RiOS SDR breaks up TCP data streams into *unique data chunks* that are stored on the hard disks (*RiOS data store*) of the device running RiOS (a SteelHead or SteelCentral Controller for SteelHead Mobile host system). Each data chunk is assigned a unique integer label (*reference*) before it is sent to a peer RiOS device across the WAN. When the same byte sequence occurs in future transmissions from clients or servers, the reference is sent across the WAN instead of the raw data chunk. The peer RiOS device (a SteelHead or SteelCentral Controller for SteelHead Mobile host system) uses this reference to find the original data chunk on its RiOS data store and reconstruct the original TCP data stream.

Files and other data structures can be accelerated by data streamlining even when they are transferred using different applications. For example, a file that is initially transferred through CIFS is accelerated when it is transferred again through FTP.

Applications that encode data in a different format when they transmit over the WAN can also be accelerated by data streamlining. For example, Microsoft Exchange uses the MAPI protocol to encode file attachments prior to sending them to Microsoft Outlook clients. As a part of its MAPI-specific optimized connections, the RiOS decodes the data before applying SDR. This decoding enables the SteelHead to recognize byte sequences in file attachments in their native form when the file is subsequently transferred through FTP or copied to a CIFS file share.

Bidirectional synchronized RiOS data store

Data and references are maintained in persistent storage in the data store within each RiOS device and are stable across reboots and upgrades. To provide further longevity and safety, local SteelHead pairs optionally keep their data stores fully synchronized bidirectionally at all times. Bidirectional synchronization ensures that the failure of a single SteelHead does not force remote SteelHeads to send previously transmitted data chunks. This feature is especially useful when the local SteelHeads are deployed in a network cluster, such as a primary and backup deployment, a serial cluster, or a WCCP cluster.

For information about primary and backup deployments, see [“Redundancy and clustering” on page 53](#). For information about serial cluster deployments, see [“Serial cluster deployments” on page 215](#). For information about WCCP deployments, see [“WCCP Virtual In-Path Deployments” on page 249](#).

Unified RiOS data store

A key Riverbed innovation is the unified data store that data streamlining uses to reduce bandwidth usage. After a data pattern is stored on the disk of a SteelHead or Mobile Controller peer, it can be leveraged for transfers to any other SteelHead or Mobile Controller peer, across all accelerated applications. Data is not duplicated within the RiOS data store, even if it is used in different applications, in different data transfer directions, or with new peers. The unified data store ensures that RiOS uses its disk space as efficiently as possible, even with thousands of remote SteelHeads or Mobile Controller peers.

Transport streamlining

SteelHeads use a generic latency optimization technique called transport streamlining. This section includes the following topics:

- [“Overview of transport streamlining” on page 22](#)
- [“Connection pooling” on page 24](#)
- [“TCP automatic detection” on page 25](#)
- [“SteelCentral Controller for SteelHead Mobile TCP transport modes” on page 25](#)
- [“Tuning SteelHeads for high-latency links” on page 25](#)
- [“TCP algorithm selection” on page 25](#)
- [“WAN buffers” on page 26](#)

You can find additional information about the transport streaming modes in [“QoS configuration and integration” on page 111](#) and [“Satellite Optimization” on page 329](#).

Overview of transport streamlining

TCP connections suffer a lack of performance due to delay, loss, and other factors. There are many articles written about and other information available regarding how to choose the client and server TCP settings and TCP algorithms most appropriate for various environments. For example, without proper tuning, a TCP connection might never be able to fill the available bandwidth between two locations. You must consider the TCP window sizes used during the lifespan of a connection. If the TCP window size is not large enough, then the sender cannot consume the available bandwidth. You must also consider packet loss due to congestion or link quality.

In many cases, packet loss is an indication to a TCP congestion avoidance algorithm that there is congestion, and congestion is a signal to the sender to slow down. The sender then can choose at which rate to slow down. The sender can:

- undergo a multiplicative decrease: for example, send at one half the previous rate.
- use other calculations to determine a slightly lower rate, just before the point at which congestion occurred.

A SteelHead deployed on the network can automate much of the manual analysis, research, and tuning necessary to achieve optimal performance, while providing you with options to fine tune. Collectively, these settings in the SteelHead are referred to as transport streamlining. The objective of transport streamlining is to mitigate the effects of WANs between client and server. Transport streamlining uses a set of standards-based and proprietary techniques to optimize TCP traffic between SteelHeads. These techniques:

- ensure that efficient retransmission methods are used (such as TCP selective acknowledgments).
- negotiate optimal TCP window sizes to minimize the impact of latency on throughput.
- maximize throughput across a wide range of WAN links.

Additionally, a goal for selecting any TCP setting and congestion avoidance algorithm and using WAN optimization appliances is to find a balance between two extremes: acting fair and being cooperative by sharing available bandwidth with coexisting flows on one end of the spectrum, or acting aggressive by trying to achieve maximum throughput at the expense of other flows on the opposite end of the spectrum. Being on the former end indicates that throughput suffers, and being on the latter end indicates that your network is susceptible to congestion collapse.

By default, the SteelHeads use standard TCP (as defined in RFC 793) to communicate between peers. This type of TCP algorithm is a loss-based algorithm that relies on the TCP algorithm to calculate the effective throughput for any given connection based on packet loss. Additionally, SteelHead peers automatically implement extensions to RFC 793 to provide further enhancements for congestion control beyond that of a standard TCP connection.

Alternatively, you can configure SteelHeads to use a delay-based algorithm called *bandwidth estimation*. The purpose of bandwidth estimation is to calculate the rate, based on the delay of the link, to recover more gracefully in the presence of packet loss.

In higher-throughput environments you can enable high-speed TCP (HS-TCP), which is a high-speed loss-based algorithm (as defined in RFC 3649) on the SteelHeads to achieve high throughput for links with high bandwidth and high latency. This TCP algorithm shifts toward the more aggressive side of the spectrum. Furthermore, you can shift even further toward the aggressive side of the spectrum, sacrificing fairness, by selecting the maximum TCP (MX-TCP) feature for traffic that you want to transmit over the WAN at a rate defined by the QoS class.

Configuring MX-TCP through the QoS settings leverages QoS features to help protect other traffic and gives you the parameters, such as minimum and maximum percentages of the available bandwidth that TCP connections matching the class can consume. Although not appropriate for all environments, MX-TCP can maintain data transfer throughput in which adverse network conditions, such as abnormally high packet loss, impair performance. Data transfer is maintained without inserting error correction packets over the WAN through forward error correction (FEC). MX-TCP effectively handles packet loss without a decrease in throughput typically experienced with TCP.

The TCP algorithms that rely on loss or delay calculations to determine the throughput should have an appropriate-sized buffer. You can configure the buffer size and choose the TCP algorithm in the Transport Settings page. The default buffer is 262,140 bytes, which should cover any connection of 20 Mbps or less with a round-trip delay up to 100 ms. This connection speed and round-trip delay composes most branch office environments connecting to a data center or hub site.

The following list is a high-level summary of each SteelHead TCP congestion avoidance algorithm:

- **Standard TCP** - Standard TCP is a standards-based implementation of TCP and is the default setting in the SteelHead. Standard TCP is a WAN-friendly TCP stack and is not aggressive towards other traffic. Additionally, standard TCP benefits from the higher TCP WAN buffers, which are used by default for each connection between SteelHeads.
- **Bandwidth estimation** - Bandwidth estimation is the delay-based algorithm that incorporates many of the features of standard TCP and includes calculation of RTT and bytes acknowledged. This additional calculation avoids the multiplicative decrease in rate detected in other TCP algorithms in the presence of packet loss. Bandwidth estimation is also appropriate for environments in which there is variable bandwidth and delay.
- **HighSpeed TCP (HS-TCP)** - HS-TCP is efficient in long fat networks (LFNs) in which you have large WAN circuits (50 Mbps and above) over long distances. Typically, you use HS-TCP when you have a few long-lived replicated or backup flows. HS-TCP is designed for high-bandwidth and high-delay networks that have a low rate of packet loss due to corruption (bit errors). HS-TCP has a few advantages over standard TCP for LFNs. Standard TCP will *backoff* (slow down the transmission rate) in the presence of packet loss, causing connections to under use the bandwidth.

Also, standard TCP is not as aggressive during the TCP slow-start period to rapidly grow to the available bandwidth. HS-TCP uses a combination of calculations to rapidly fill the link and minimize backoff in the presence of loss. These techniques are documented in RFC 3649. HS-TCP is not beneficial for satellite links because the TCP congestion window recovery requires too many round trips or is too slow. HS-TCP requires that you adjust WAN buffers on the SteelHeads to be equal to 2 x BDP, where bandwidth-delay product (BDP) is the product of the WAN bandwidth and round-trip latency between locations. For more specific settings, see [“Storage Area Network Replication” on page 411](#).

- **SkipWare Space Communications Protocol Standards (SCPS) per connection** - SCPS per connection is for satellite links with few or no packet drops due to corruption. SCPS per connection requires a separate license.

For more details, see [“SCPS per connection” on page 336](#).

- **SCPS error tolerance** - SCPS error tolerance is for satellite links that have packet drops due to corruption. You must have a separate license to activate SCPS error tolerance.

For more details, see [“SCPS error tolerance” on page 336](#).

The following list is a high-level summary of additional modes that alter the SteelHead TCP congestion avoidance algorithm:

- **MX-TCP** - MX-TCP is ideal for dedicated links, or to compensate for poor link quality (propagation issues, noise, and so on) or packet drops due to network congestion. The objective of MX-TCP is to achieve maximum TCP throughput. MX-TCP alters TCP by disabling the congestion control algorithm and sending traffic up to a rate you configure, regardless of link conditions. Additionally, MX-TCP can share any excess bandwidth with other QoS classes through adaptive MX-TCP. MX-TCP requires knowledge of the amount of bandwidth available for a given QoS class because, provided that enough traffic matches the QoS class, connections using MX-TCP attempt to consume the bandwidth allotted without regard to any other traffic.

For more details, see [“MX-TCP” on page 123](#) and [“MX-TCP settings” on page 402](#).

- **Rate pacing** - Rate pacing is a combination of MX-TCP and a TCP congestion avoidance algorithm. You use rate pacing commonly in satellite environments, but you can use it in terrestrial connections as well. The combination of MX-TCP and a TCP congestion avoidance algorithm allows rate pacing to take the best from both features. Rate pacing leverages the rate configured for an MX-TCP QoS class to minimize buffer delays, but can adjust to the presence of loss due to network congestion.

For more details, see [“Configuring rate pacing” on page 344](#).

For additional information about transport streamlining mode options, see the following topics:

- [“Connection pooling” on page 24](#)
- [“TCP automatic detection” on page 25](#)

Connection pooling

Connection pooling adds a benefit to transport streamlining by minimizing the time for an optimized connection to set up.

Some application protocols, such as HTTP, use many rapidly created, short-lived TCP connections. To optimize these protocols, SteelHeads create pools of idle TCP connections. When a client tries to create a new connection to a previously visited server, the SteelHead uses a TCP connection from its pool of connections. Thus the client and the SteelHead do not have to wait for a three-way TCP handshake to finish across the WAN. This feature is called *connection pooling*. Connection pooling is available only for connections using the correct addressing WAN visibility mode.

Transport streamlining ensures that there is always a one-to-one ratio for active TCP connections between SteelHeads and the TCP connections to clients and servers. Regardless of the WAN visibility mode in use, SteelHeads do not tunnel or perform multiplexing and demultiplexing of data across connections.

For information about correct addressing modes, see [“WAN Visibility Modes” on page 63](#). For information about HTTP optimization, see the *SteelHead Deployment Guide - Protocols*.

TCP automatic detection

One best practice you can consider for nearly every deployment is the TCP automatic detection feature on the data center SteelHeads. This feature allows the data center SteelHead to reflect the TCP algorithm in use by the peer. The benefit is that you can select the appropriate TCP algorithm for the remote branch office, and the data center SteelHead uses that TCP algorithm for connections. If SteelHeads on both sides of an optimized connection use the automatic detection feature, then standard TCP is used.

SteelCentral Controller for SteelHead Mobile TCP transport modes

This section briefly describes specific transport streamlining modes that operate with SteelCentral Controller for SteelHead Mobile.

HS-TCP is not the best choice for interoperating in a SteelCentral Controller for SteelHead Mobile environment because it is designed for LFNs (high bandwidth and high delay). Essentially, the throughput is about equal to standard TCP.

MX-TCP is a sender-side modification (configured on the server side) and is used to send data at a specified rate. When SteelCentral Controller for SteelHead Mobile is functioning on the receiving side, it can be difficult to deploy MX-TCP on the server side. The issue is defining a sending rate in which it might not be practical to determine the bandwidth that a client can receive on their mobile device because it is unknown and variable.

Tuning SteelHeads for high-latency links

We recommend that you gather WAN delay (commonly expressed as RTT), packet-loss rates, and link bandwidth to better understand the WAN characteristics so that you can make adjustments to the default transport streamlining settings. Also, understanding the types of workloads (long-lived, high-throughput, client-to-server traffic, mobile, and so on) is valuable information for you to appropriately select the best transport streamlining settings.

Specific settings for high-speed data replication are covered in [“Storage Area Network Replication” on page 411](#). The settings described in this chapter approximate when you can adjust the transport streamlining settings to improve throughput.

TCP algorithm selection

The default SteelHead settings are appropriate in most deployment environments. Based on RTT, bandwidth, and loss, you can optionally choose different transport streamlining settings. A solid approach to selecting the TCP algorithm found on the Transport Settings page is to use the automatic detection feature (auto-detect) on the data center SteelHead. The benefit to automatic detection is that the data center SteelHead reflects the choice of TCP algorithm in use at the remote site. You select the TCP algorithm at the remote site based on WAN bandwidth, RTT, and loss.

A general guideline is that any connection more than 50 Mbps can benefit from using HS-TCP, unless the connection is over satellite (delay greater than 500 ms). You can use MX-TCP for high data rates if the end-to-end bandwidth is known and dedicated.

When you are factoring in loss at lower-speed circuits, consider using bandwidth estimation. When planning, consider when packet loss is greater than 0.1%. Typically, MPLS networks are below 0.1% packet loss, while other communication networks can be higher. For any satellite connection, the appropriate choices are SCPS (if licensed) or bandwidth estimation.

For specific implementation details, see [“Satellite Optimization” on page 329](#).

WAN buffers

After you select the TCP algorithm, another setting to consider is the WAN-send and WAN-receive buffers. You can use bandwidth and RTT to determine the BDP. BDP is a multiplication of bandwidth and RTT, and it is commonly divided by 8 and expressed in bytes. To get better performance, the SteelHead as a TCP proxy typically uses two times BDP as its WAN-send and WAN-receive buffer. For asymmetry, you can have the WAN-send buffer reflect the bandwidth and delay in the transmit direction, while the WAN-receive buffer reflects the bandwidth and delay in the receive direction. You do not have to adjust the buffer settings unless there is a are only a few connections and you want to consume most or all of the available WAN bandwidth.

Application streamlining

You can apply application-specific optimization for specific application protocols. For SteelHeads using RiOS 6.0 or later, application streamlining includes the following protocols:

- CIFS for Windows and Mac clients (Windows file sharing, backup and replication, and other Windows-based applications)
- MAPI, Outlook Anywhere (RPC over HTTP) and MAPI over HTTP, Microsoft Exchange 5.5, 2000, 2003, 2007, 2010, 2013, and 2016)
- NFSv3 for UNIX file sharing
- TDS for Microsoft SQL Server
- HTTP
- HTTPS and SSL
- IMAP-over-SSL
- Oracle 9i, which comes with Oracle Applications 11i
- Oracle10gR2, which comes with Oracle E-Business Suite R12
- Lotus Notes 6.0 or later
- Encrypted Lotus Notes
- ICA Client Drive Mapping
- Multi-stream ICA and multi-port ICA support

Protocol-specific optimization reduces the number of round trips over the WAN for common actions and help move through data obfuscation and encryption by:

- opening and editing documents on remote file servers (CIFS).
- sending and receiving attachments (MAPI and Lotus Notes).

- viewing remote intranet sites (HTTP).
- securely performing RiOS SDR for SSL-encrypted transmissions (HTTPS).

For more information about application streamlining, see the *SteelHead Deployment Guide - Protocols*.

Management streamlining

Developed by Riverbed, management streamlining simplifies the deployment and management of RiOS devices, including both hardware and software:

- **Autodiscovery Protocol** - Autodiscovery enables SteelHeads and SteelCentral Controller for SteelHead Mobile to automatically find remote SteelHeads and begin to optimize traffic. Autodiscovery avoids the requirement of having to define lengthy and complex network configurations on SteelHeads. The autodiscovery process enables administrators to:
 - control and secure connections.
 - specify which traffic is to be optimized.
 - specify peers for optimization.

For more information, see [“Autodiscovery protocol” on page 31](#).

- **SCC** - The SCC enables new, remote SteelHeads to be automatically configured and monitored. It also provides a single view of the overall optimization benefit and health of the SteelHead network.
- **Mobile Controller** - The Mobile Controller is the management appliance that you use to track the individual health and performance of each deployed software client and to manage enterprise client licensing. The Mobile Controller enables you to see who is connected, view their data reduction statistics, and perform support operations such as resetting connections, pulling logs, and automatically generating traces for troubleshooting. You can perform all of these management tasks without end-user input.

For more information, see [“SteelCentral Controller for SteelHead Mobile Deployments” on page 511](#).

RiOS data store synchronization

RiOS data store synchronization enables pairs of local SteelHeads to synchronize their data stores with each other, even while they are optimizing connections. RiOS data store synchronization is typically used to ensure that if a SteelHead fails, no loss of potential bandwidth savings occurs, because the data segments and references are on the other SteelHead. This section includes the following topics:

- [“RiOS data store synchronization requirements” on page 28](#)
- [“RiOS data store error alarms” on page 28](#)

You can use RiOS data store synchronization for physical in-path, virtual in-path, or out-of-path deployments. You enable synchronization on two SteelHeads: one as the synchronization primary, and the other as the synchronization backup.

The traffic for RiOS data store synchronization is transferred through either the SteelHead primary or auxiliary network interfaces, not the in-path interfaces.

RiOS data store synchronization is a bidirectional operation between two SteelHeads, regardless of which deployment model you use. The SteelHead *primary* and *backup* designations are relevant only in the initial configuration, when the primary SteelHead RiOS data store essentially overwrites the backup SteelHead RiOS data store.

RiOS data store synchronization requirements

The synchronization primary and its backup:

- must have the same hardware model.
- must be running the same version of RiOS.
- do not have to be in the same physical location. If they are in different physical locations, they must be connected through a fast, reliable LAN connection with minimal latency.

Note: Before you replace a synchronization primary for any reason, we recommend that you make the synchronization backup the new synchronization primary. Therefore, the new primary (the former backup) can warm the new (replacement) SteelHead, ensuring that the most data is optimized and none is lost.

For more details on data store synchronization, see <https://supportkb.riverbed.com/support/index?page=content&id=S12964>.

RiOS data store error alarms

You receive an email notification if an error occurs in the RiOS data store. The RiOS data store alarms are enabled by default.

If the alarm was caused by an unintended change to the configuration, you can change the configuration to match the old RiOS data store settings again and then restart the optimization service (without clearing the data store) to reset the alarm. In certain situations you might need to clear the RiOS data store. Typical configuration changes that require a clear data store are changes to the data store encryption type or enabling the extended peer table.

To clear the data store of data, restart the optimization service and click Clear the Data Store.

For more information about the RiOS data store error alarm, see the *SteelHead User Guide*.

Choosing the right SteelHead model

Generally, you select a SteelHead model based on the number of users, the bandwidth requirements, and the applications used at the deployment site. However:

- If you do not want to optimize applications that transfer large amounts of data (for example, WAN-based backup or restore operations, system image, or update distribution), choose your SteelHead model based on the amount of bandwidth and number of concurrent connections at your site.
- If you want to optimize applications that transfer large amounts of data, choose your SteelHead model based on the amount of bandwidth, the number of concurrent connections at your site, and the size of the RiOS data store.
- If you want to use SteelHead to enforce network Quality of Service (QoS), consider the SteelHead model QoS bandwidth in addition to the optimization criteria you need.

You can also consider high availability, redundancy, data protection, or other design-related requirements when you select a SteelHead. SteelHead models vary according to the following attributes:

- Optimized WAN bandwidth license limit
- Maximum number of concurrent TCP connections that can be optimized
- Maximum number of possible in-path interfaces
- Availability of RAID and solid-state drives (SSD) for the RiOS data store
- Availability of fiber interfaces
- Availability of redundant power supplies
- Availability of hardware-based compression card
- Availability of VSP
- Upgrade options through software licenses

All SteelHead models have the following specifications that are used to determine the amount of traffic that a single SteelHead can optimize:

- **Number of concurrent TCP connections** - Each SteelHead model can optimize a certain number of concurrent TCP connections.

The number of TCP concurrent connections that you need for optimization depends on the number of users at your site, the applications that you use, and whether you want to optimize all applications or just a few of them. When planning corporate enterprise deployments, we recommend that you use ratios of 5 to 15 concurrent connections per user if full optimization is desired, depending on the applications being used.

If the number of connections that you want to optimize exceed the limit of the SteelHead model, the excess connections are passed through unoptimized by the SteelHead.

The TCP protocol only supports approximately 64,000 ports per IP address for outbound or inbound connections to or from a unique IP-Port pair. If your design or environment requires you to support more than 64,000 concurrent connections to a single IP address for optimization, we recommend that you use multiple in-path interfaces or service port mapping.

For more information about service port mapping, go to <https://supportkb.riverbed.com/support/index?page=content&id=S16309>.

- **Number of encrypted TCP connections** - The SteelHead transparently optimizes encrypted connections by first decrypting it to optimize the payload, and then reencrypting it for secure transport over the WAN. Examples of encrypted applications and protocols that a SteelHead can optimize include HTTP Secure (or HTTPS/SSL) and Microsoft Outlook/Outlook Anywhere traffic (or encrypted-MAPI, RPC-over-HTTPS).

If you have predominately encrypted connections, contact your Riverbed account team for assistance with determining the appropriate model.

- **WAN bandwidth limit** - Each SteelHead model has a limit on the rate at which it sends optimized traffic toward the WAN. You might not need a SteelHead model that is rated for the same bandwidth available at the deployment site; however, we recommend that you make sure that the selected appliance is not a bottleneck for the outbound optimized traffic. The optimized WAN bandwidth limit applies only to optimized traffic.

When a SteelHead reaches its optimized WAN bandwidth limit, it begins shaping optimized traffic along this limit. New connections continue to be optimized as long as the concurrent TCP connection count is not exceeded. The optimized WAN bandwidth limit is not to be confused with the QoS WAN bandwidth limit that applies to both optimized and pass-through traffic.

- **RiOS data store size** - Each SteelHead model has a fixed amount of disk space available for RiOS SDR. Because SDR stores unique patterns of data, the amount of data store space needed by a deployed SteelHead differs from the amount needed by applications or file servers. For the best optimization possible, the RiOS data store must be large enough to hold all of the commonly accessed data at a site. Old data that is recorded in the RiOS data store might eventually be overwritten by new data, depending on traffic patterns.

At sites where applications transfer large amounts of data (for example, WAN-based backup or restore operations, system image, or update distribution), you must not select the SteelHead model based only on the amount of bandwidth and number of concurrent connections at the site, but also on the size of RiOS data store. Sites without these applications are typically sized by considering the bandwidth and number of concurrent connections.

If your requirements exceed the capacity of a single SteelHead, consider a SteelHead cluster. There are many ways to cluster SteelHeads and scale up the total optimized capacity and provide redundancy.

For more information about clustering, see [“In-path redundancy and clustering examples” on page 213](#).

You need to determine the SteelHead platform best suited for your deployment. SteelHeads are available in the following platforms:

- **Physical** - the SteelHead CX and SteelHead EX appliances
- **Virtual** - the SteelHead-v for VMware vSphere and Microsoft Hyper-V hypervisors
- **Cloud/SaaS** - the SteelHead-c for Amazon Web Services and Microsoft Azure
- **Mobile** - the Mobile Controller and the SteelHead Mobile client for Windows and Mac-based laptops

Note: The Hyper-V hypervisor platform does not support the Riverbed bypass NIC card.

The SteelHead CX model is a dedicated WAN optimization solution that supports a rich set of application protocols that includes but is not limited to Microsoft Exchange, CIFS, SMB, SharePoint, HTTP/HTTPS, Lotus Notes, FCIP, SRDF, and SnapMirror.

The SteelHead EX model combines WAN optimization, virtualization, and storage consolidation into a single appliance for the branch office. Virtualization is provided by VSP. VSP is a VMware-based hypervisor built into RiOS. VSP enables you to virtualize and localize multiple branch services such as print, DNS, and DHCP, into a single appliance for the branch office. Storage consolidation is provided by the SteelFusion Edge, a virtual edge service that enables end users in a branch office to access and write to centralized storage in the data center over the WAN at local speeds. SteelFusion Edge is available on the SteelHead EX as a license upgrade option.

A complete enterprise WAN optimization solution can require the deployment of multiple SteelHead platforms. For example, you have hosted some of your applications in the cloud and you do want to optimize the traffic for these applications. In this deployment you need a physical SteelHead in the data center (client side) and a SteelHead-c in the SaaS vendor cloud (server side).

You might also like to optimize traffic for your internally hosted business applications between their mobile users and data center. Your mobile users access their data center over the internet using a VPN. In this deployment you need a SteelHead Mobile client in the mobile user laptop (client-side) and a nonmobile SteelHead on the server side. In addition, you need to deploy a Mobile Controller in your data center to manage and license your SteelHead Mobile clients.

If you need help planning, designing, deploying, or operating your SteelHeads, Riverbed offers consulting services directly and through Riverbed authorized partners. For details, contact Riverbed Professional Services by email at proserve@riverbed.com or go to <http://www.riverbed.com/services/index.html>.

Deployment modes for the SteelHead

You can deploy SteelHeads into the network in many different ways. Deployment modes available for the SteelHeads include:

- **Physical In-Path** - In a physical in-path deployment, the SteelHead is physically in the direct path between clients and servers. In-path designs are the simplest to configure and manage, and they are the most common type of SteelHead deployment, even for large sites. Many variations of physical in-path deployments are possible to account for redundancy, clustering, and asymmetric traffic flows.
For details, see [“Physical in-path deployments” on page 197](#).
- **Virtual In-Path** - In a virtual in-path deployment, you can use a redirection mechanism (like WCCP, PBR, or Layer-4 switching) to place the SteelHead virtually in the path between clients and servers.
For details, see [“Virtual In-Path Deployments” on page 245](#).
- **Out-of-Path** - In an out-of-path deployment, the SteelHead is not in the direct path between the client and the server. In an out-of-path deployment, the SteelHead acts as a proxy. This type of deployment might be suitable for locations where physical in-path or virtual in-path configurations are not possible. However, out-of-path deployments have several drawbacks that you must be aware of.
For details, see [“Out-of-Path Deployments” on page 387](#).

Autodiscovery protocol

This section describes the SteelHead autodiscovery protocol. This section includes the following topics:

- [“Original autodiscovery process” on page 33](#)
- [“Configuring enhanced autodiscovery” on page 35](#)

Autodiscovery enables SteelHeads to automatically find remote SteelHeads and to optimize traffic with them. Autodiscovery relieves you of having to manually configure the SteelHeads with large amounts of network information.

The autodiscovery process enables you to:

- control and secure connections.
- specify which traffic is optimized.
- specify how remote peers are selected for optimization.

The types of autodiscovery are as follows:

- **Original Autodiscovery** - Automatically finds the first remote SteelHead along the connection path.
- **Enhanced Autodiscovery (available in RiOS 4.0.x or later)** - Automatically finds the last SteelHead along the connection path.

Most SteelHead deployments use autodiscovery. You can also manually configure SteelHead pairing using fixed-target in-path rules, but this approach requires ongoing configuration. Fixed-target rules also require tracking new subnets that are present in the network and for which SteelHeads are responsible for optimizing the traffic.

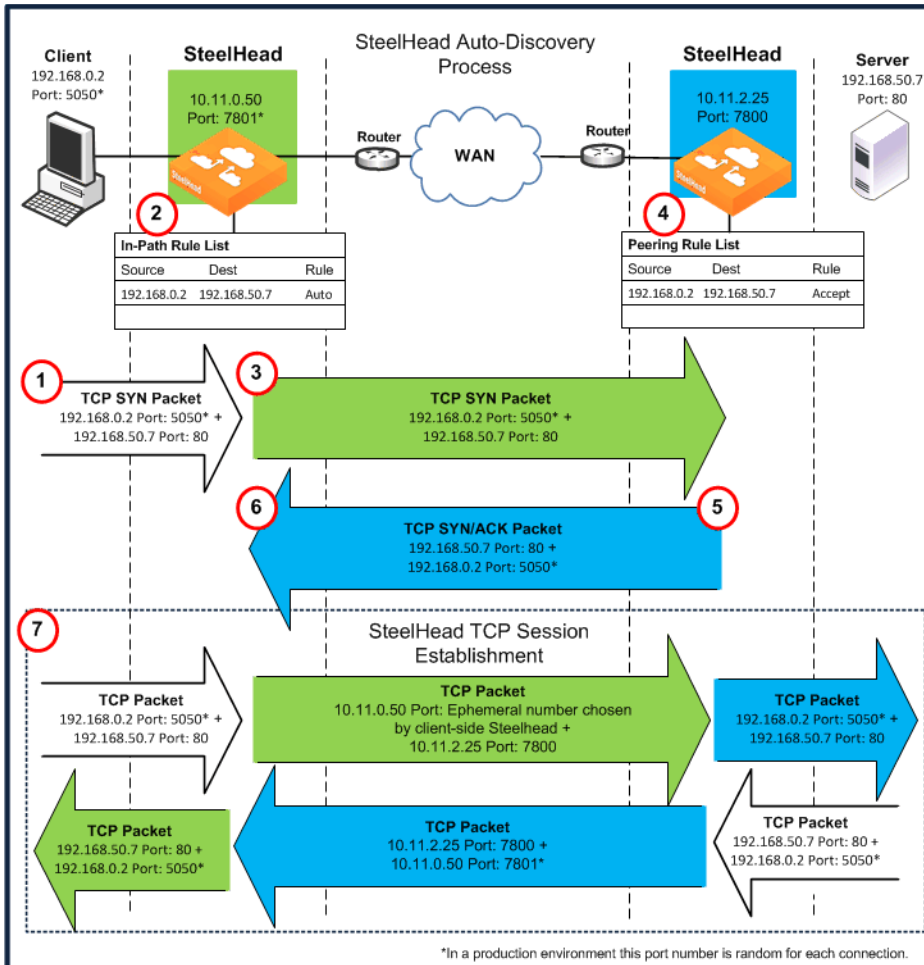
You can use autodiscovery if the SteelHeads are deployed physically in-path or virtually in-path (such as WCCP or PBR). The following section describes physically in-path SteelHeads, but the packet flow is identical to a virtual in-path deployment.

For information about fixed-target in-path rules, see [“Fixed-target in-path rules” on page 48](#).

Original autodiscovery process

The section describes how a client connects to a remote server when the SteelHeads have autodiscovery enabled. In this example, each SteelHead uses correct addressing and a single subnet.

Figure 1-1. Autodiscovery process



Note: This example does not show asymmetric routing detection or enhanced autodiscovery peering.

In the original autodiscovery process:

1. The client initiates the TCP connection by sending a TCP SYN packet.
2. The client-side SteelHead receives the packet on its LAN interface, examines the packet, discovers that it is a SYN, and continues processing the packet:
 - Using information from the SYN packet (for example, the source or destination address, or VLAN tag), the SteelHead performs an action based on a configured set of rules, called *in-path rules*. In this example, because the matching rule for the packet is set to auto, the SteelHead uses autodiscovery to find the remote SteelHead.
 - The SteelHead appends a TCP option to the packet TCP option field, which is called the *probe query option*. The probe query option contains the in-path IP address of the client-side SteelHead. Nothing else in the packet changes, only the option is added.

3. The SteelHead forwards the modified packet (denoted as SYN_probe_query) out of the WAN interface. Because neither the source nor destination fields are modified, the packet is routed in the same manner as if there was no SteelHead deployed.
4. The server-side SteelHead receives the SYN_probe_query packet on its WAN interface, examines the packet, discovers that it is a SYN packet, and searches for a TCP probe query. If found, the server-side SteelHead:
 - uses the packet fields and the IP address of the client-side SteelHead to determine what action to take, based on its peering rules. In this example, because the matching rule is set to accept (or auto, depending on the RiOS version), the server-side SteelHead communicates to the client-side SteelHead that it is the remote optimization peer for this TCP connection.
 - removes the probe_query option from the packet, and replaces it with a probe_response option (the probe_query and probe_response use the same TCP option number). The probe_response option contains the in-path IP address of the server-side SteelHead.
 - reverses all of the source and destination fields (TCP and IP) in the packet header. The packet sequence numbers and flags are modified to make the packet look like a normal SYN/ACK server response packet.

If no server-side SteelHeads are present, the server ignores the TCP probe that was added by the client-side SteelHead, responds with a regular SYN/ACK resulting in a pass-through connection, and sends the SYN/ACK.
5. The server-side SteelHead transmits the packet to the client-side SteelHead. Because the destination IP address of the packet is now the client IP address, the packet is routed through the WAN just as if the server was responding to the client.
6. The client-side SteelHead receives the packet on its WAN interface, examines it, and discovers that it is a SYN/ACK. The client-side SteelHead scans for and finds the **probe_response** field and reads the in-path IP address of the server-side SteelHead. Now the client-side SteelHead knows all the parameters of the packet TCP flow, including the:
 - IP addresses of the client and server.
 - TCP source and destination ports for this connection.
 - in-path IP address of the server-side SteelHead for this connection.
7. The SteelHeads establish three TCP connections:
 - The client-side SteelHead completes the TCP connection setup with the client as if it were the server.
 - The SteelHeads complete the TCP connection between each other.
 - The server-side SteelHead completes the TCP connection with the server as if it were the client.

After the three TCP connections are established, optimization begins. The data sent between the client and server for this specific connection is optimized and carried on its own individual TCP connection between the SteelHeads.

Configuring enhanced autodiscovery

In RiOS 4.0.x or later, enhanced autodiscovery is available. Enhanced autodiscovery automatically discovers the last SteelHead in the network path of the TCP connection. In contrast, the original autodiscovery protocol automatically discovers the first SteelHead in the path. The difference is only seen in environments where there are three or more SteelHeads in the network path for connections to be optimized.

Enhanced autodiscovery works with SteelHeads running the original autodiscovery protocol. Enhanced autodiscovery ensures that a SteelHead optimizes only TCP connections that are being initiated or terminated at its local site, and it ensures that a SteelHead does not optimize traffic that is transiting through its site.

For information about passing through transit traffic using enhanced autodiscovery and peering rules, see [“Configuring pass-through transit traffic” on page 44](#).

To enable enhanced autodiscovery

- Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
in-path peering auto
```

Autodiscovery and firewall considerations

This section describes factors to consider when using autodiscovery:

- [“Removal of the Riverbed TCP option probe” on page 35](#)
- [“Stateful firewall device in a multiple in-path environment” on page 35](#)

Removal of the Riverbed TCP option probe

The most common reason that autodiscovery fails is because a device (typically security related) strips out the TCP options from optimized packets. Autodiscovery relies on using TCP options to determine if a remote SteelHead exists. Common devices that remove TCP options are firewalls and satellite routers.

The Riverbed Support site has knowledge base articles that show example configurations to allow Riverbed autodiscovery options through different firewall types.

To solve the problem, you can configure the devices to ignore or prevent stripping out the TCP option. Alternatively, you can use fixed-target rules on the SteelHeads to bypass the autodiscovery process.

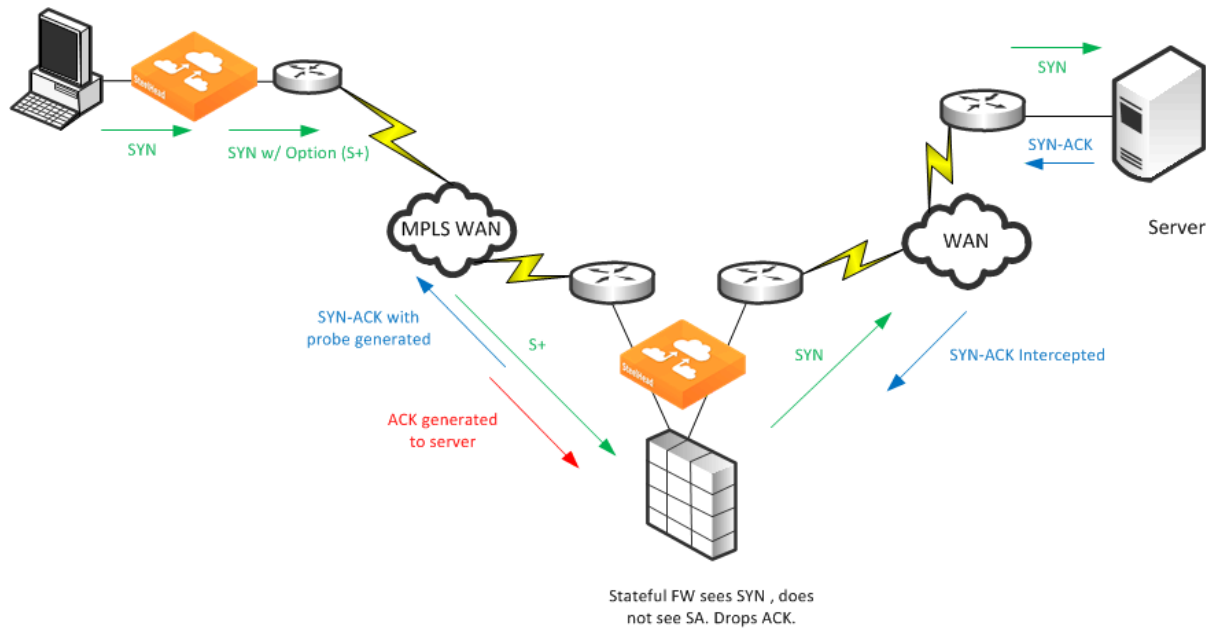
For details, see [“Configuring a fixed-target in-path rule for an in-path deployment” on page 49](#).

Stateful firewall device in a multiple in-path environment

Consider the following when you install a SteelHead in which a stateful firewall exists in a multiple in-path environment. A stateful firewall on the LAN side of a SteelHead might not detect the entire TCP handshake conversation, and this causes the firewall to drop packets.

Figure 1-2 shows an example when the SteelHead is at a transit site where traffic passes through multiple interfaces on the SteelHead, with a firewall located on the LAN. The firewall does not see the whole TCP handshake.

Figure 1-2. Multiple in-path interfaces with stateful firewall on the LAN



In this example, the stateful firewall detects the SYN packet to the server, but the SYN-ACK response from the server is intercepted at the SteelHead prior to reaching the firewall. When the server-side SteelHead originates the ACK to the server, the stateful firewall often denies the connection because it has not detected the proper SYN-ACK response.

Note: The SteelHead can appear to be bridging traffic between its in-path interfaces when it is not. The SteelHead always generates packets from the owner interface, but it intercepts packets on any in-path interface. This behavior is necessary in asymmetric routing environments.

When you have a stateful firewall device in a multiple in-path environment, the goal of the SteelHead is to provide optimization for local traffic only at the transit site. The remote site usually has its own local SteelHead if optimization is required. To prevent the transit SteelHead from participating in the autodiscovery process, you can use peering rules to accept only probed SYNs destined to devices at the transit site. Contact Riverbed Professional Services for further options and solutions.

For more information about peering rules, see [“Peering rules” on page 39](#).

Multiple in-path discovery behavior

This section describes how multiple in-path interfaces on a SteelHead interact to intercept and originate packets for optimized connections. The SteelHead can support up to ten in-path interfaces, depending on the model. The packet flow is the same no matter how many interfaces are used.

You can use multiple in-path interfaces on a SteelHead to allow different network paths in an asymmetric routing environment. For example, you can have two or more routers that have circuits to the same or different MPLS circuits, and you can route traffic over either path. Figure 1-3 shows multiple in-path interfaces.

Figure 1-3. Multiple in-path interface example

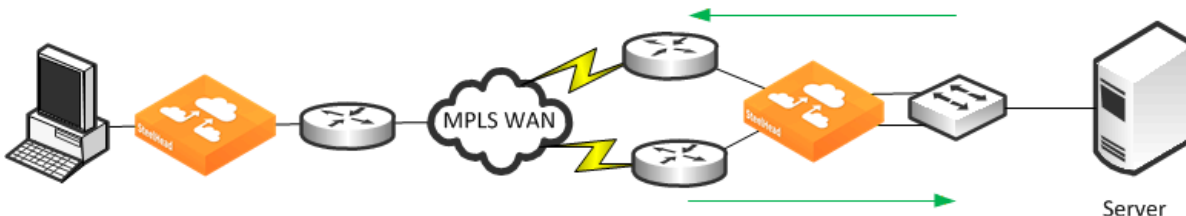


Figure 1-3 shows traffic flowing to the server through a different router than traffic returning from the server. In many cases, you can use the two circuits for load balancing, so inbound or outbound traffic can legitimately flow through either router. In Figure 1-3, the SteelHead on the right uses two in-path interfaces to allow for the possibility of asymmetric traffic.

The SteelHead intercepts packets for optimized connections on any in-path interface. The most common reason for enabling multiple in-path interfaces is to ensure that asymmetric routing does not prevent the SteelHead from detecting all packets for an optimized connection.

Packet origination from the SteelHead always comes from one in-path interface. Each of the optimizing SteelHeads has one in-path interface, which is referred to as the *owner interface*. The owner interface is bound to the optimized connection. On the client side, the first SteelHead's last in-path interface to detect the SYN packet from the client is the owner interface. The client-side SteelHead appends a TCP option to this SYN packet, known as a Riverbed probe.

If the SYN packet with a probe passes through the client-side SteelHead, the client-side SteelHead updates the IP address inside the probe option with the IP address of the in-path interface it is passing through. With original autodiscovery, the owner interface is the first in-path interface of the first server-side SteelHead. With enhanced autodiscovery, because the goal is to optimize to the last SteelHead in the path, the owner interface on the server side is the last SteelHead's first in-path interface to see the probed SYN. In most cases, the actual owner interface does not matter as much as which SteelHeads are performing the optimization.

Figure 1-4 shows a more detailed example of in-path interfaces. The server-side SteelHead is on the right.

Figure 1-4. Multiple in-path interface (more detailed example)

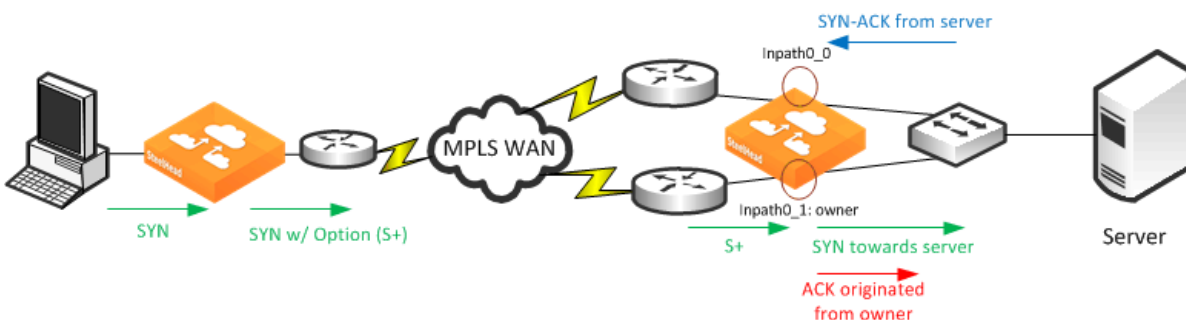


Figure 1-4 shows the client-side SteelHead owner interface is `inpath0_0`, because this interface detects the client SYN packet. The probed SYN packet reaches the server-side SteelHead on `inpath0_1`; therefore, `inpath0_1` on the server-side SteelHead is the owner. The SteelHead intercepts packets for the optimized connections it is aware of. Even though the server SYN-ACK response comes back into `inpath0_0`, the SteelHead intercepts these packets and prevents asymmetric routing from bypassing the SteelHead. Packets originating from the SteelHead are always sourced from the owner interface. The ACK towards the server originates from the server-side SteelHead's `inpath0_1` interface.

Controlling optimization

You can configure what traffic a SteelHead optimizes and what other actions it performs, using the following rules:

- **In-path rules** - In-path rules determine the action a SteelHead takes when a connection is initiated, usually by a client.
- **Peering rules** - Peering rules determine how a SteelHead reacts when it detects a probe query.

This section includes the following topics:

- [“In-path rules” on page 38](#)
- [“Default in-path rules” on page 39](#)
- [“Peering rules” on page 39](#)
- [“Kickoff and automatic kickoff features” on page 40](#)

In-path rules

In-path rules are used only when a connection is *initiated*. Because connections are typically initiated by clients, in-path rules are configured for the initiating, or client-side, SteelHead. In-path rules determine SteelHead behavior with SYN packets.

In-path rules are an ordered list of fields that a SteelHead attempts to match with SYN packet fields: for example, host label, port label, domain labels, source or destination subnet, IP address, VLAN, or TCP port. Each in-path rule has an *action* field. When a SteelHead finds a matching in-path rule for a SYN packet, the SteelHead treats the packet according to the action specified in the in-path rule.

RiOS 9.2 and later can use a preconfigured host, port, and domain label as a field identifier to intercept traffic. For more information on host and port labels for QoS and path selection, see [“Application Definitions” on page 97](#).

Note: Domain labels require both SteelHeads to be running RiOS 9.2 or later.

The in-path rule actions, each with different configuration possibilities, are as follows:

- **Auto Discover** - Uses the autodiscovery process to determine if a remote SteelHead is able to optimize the connection attempting to be created by this SYN packet.
- **Fixed-Target** - Omits the autodiscovery process and instead uses a specified remote SteelHead as an optimization peer. Fixed-target rules require the input of at least one remote target SteelHead; you can specify an optional backup SteelHead.

For information about fixed-target in-path rules, see [“Fixed-target in-path rules” on page 48](#).

- **Fixed-Target (Packet Mode Optimization)** - Skips the autodiscovery process and uses a specified remote SteelHead as an optimization peer to perform bandwidth optimization on TCPv4, TCPv6, UDPv4, or UDPv6 connections.
- **Pass Through** - Allows the SYN packet to pass through the SteelHead. No optimization is performed on the TCP connection initiated by this SYN packet.
- **Discard** - Drops the SYN packet silently.
- **Deny** - Drops the SYN packet and sends a message back to its source.

Only use in-path rules in the following scenarios:

- TCP SYN packet arrives on the LAN interface of physical in-path deployments.
- TCP SYN packet arrives on the WAN interface of virtual in-path deployments.

Again, both of these scenarios are associated with the first, or *initiating*, SYN packet of the connection. In-path rules are applicable to only the client-side SteelHead. In-path rules have no effect on connections that are already established, regardless of whether the connections are being optimized.

In-path rule configurations differ depending on the action. For example, both the fixed-target and the autodiscovery actions allow you to choose configurations such as what type of optimization is applied, what type of data reduction is used, and what type of latency optimization is applied.

For an example of how in-path rules are used, see [“Configuring high-bandwidth, low-latency environment” on page 42](#).

Default in-path rules

The SteelHead ships with three default in-path rules. Default rules pass through certain types of traffic unoptimized because these protocols (Telnet, SSH, HTTPS) are typically used when you deploy and configure your SteelHeads. The default in-path rules can be removed or overwritten by altering or adding other rules to the in-path rule list, or by changing the port groups that are used. The default rules allow the following traffic to pass through the SteelHead without attempting optimization:

- **Encrypted Traffic** - Includes HTTPS, SSH, and others.
- **Interactive Traffic** - Includes Telnet, ICA, and others.
- **Riverbed Protocols** - Includes the TCP ports used by Riverbed products (that is, the SteelHead, the SteelHead Interceptor, and the Mobile Controller).

Peering rules

Peering rules control SteelHead behavior when the appliance detects probe queries.

Peering rules (displayed using the **show in-path peering rules** command) are an ordered list of fields that a SteelHead uses to match with incoming SYN packet fields (for example, source or destination subnet, IP address, VLAN, or TCP port) and with the in-path IP address of the probing SteelHead. If more than one in-path interface exists on the probing SteelHead, apply one peering rule for each in-path interface. Peering rules are especially useful in complex networks.

Peering rule actions are as follows:

- **Pass** - The receiving SteelHead does not respond to the probing SteelHead, and it allows the SYN+probe packet to continue through the network.
- **Accept** - The receiving SteelHead responds to the probing SteelHead and becomes the remote-side SteelHead (that is, the peer SteelHead) for the optimized connection.
- **Auto** - If the receiving SteelHead is not using enhanced autodiscovery, this rule has the same effect as the Accept peering rule action. If enhanced autodiscovery is enabled, the SteelHead becomes the optimization peer only if it is the last SteelHead in the path to the server.

If a packet does not match any peering rule in the list, the default rule applies.

Kickoff and automatic kickoff features

The kickoff feature provides you with a simple way to ensure that unoptimized active TCP connections passing through the SteelHead can be reset. When a connection is reset, it tries to reestablish itself using the SYN, SYN-ACK, ACK handshake. The SteelHead uses its in-path rule table to determine if the connection should be optimized.

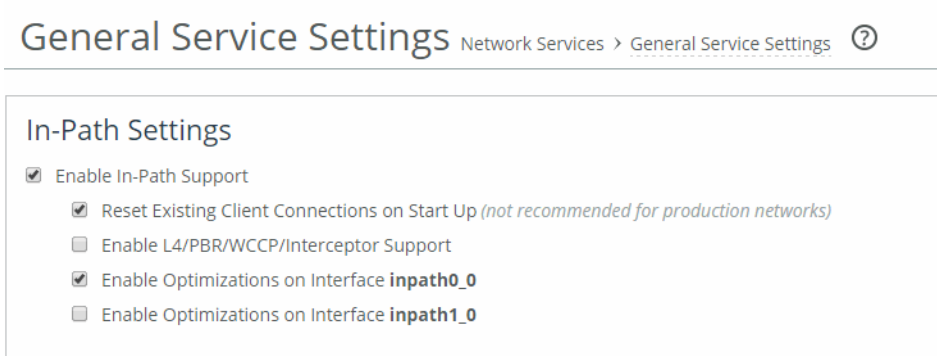
For more information about in-path rules, see [“In-path rules” on page 38](#).

Connections can pass through a SteelHead unoptimized when they are set up and active before the SteelHead optimization service is running. By default, the SteelHead does not reset legacy connections and reports them as preexisting.

The main difference between the auto kickoff feature and the kickoff feature is that kickoff has a global setting that can affect all existing connections passing through a SteelHead. The global setting sends a reset to all connections, regardless of whether they need one. The global setting is not recommended for production networks, but you can use it in lab test scenarios. By default, this setting is not enabled. You can enable this setting in the Management Console or with the **in-path kickoff** command.

The automatic kickoff feature is included in RiOS 6.1 or later.

Figure 1-5. Automatic kickoff feature global setting



Note: You can reset individual connections manually on the SteelHead. This reset forces an existing connection from optimized to pass-through, or vice versa, when you test a new rule in the SteelHead's in-path rule table. You can also use manual reset for diagnostic purposes. You can set this feature on the Current Connections page in the Management Console or with the **tcp connection send pass-reset** command.

If you configure the automatic kickoff feature, when a SteelHead comes out of bypass mode, it automatically resets (by sending RST to the client and server) only the preexisting TCP connections that match in-path rules for which automatic kickoff is enabled.

If automatic kickoff and the global kickoff feature are both enabled on the same SteelHead, the global kickoff setting takes precedence.

One of the reasons to use the automatic kickoff feature instead of the kickoff feature is that you can enable it as part of an in-path rule for optimization (Figure 1-6). Automatic kickoff is only available for autodiscovery and fixed-target rules. When you enable automatic kickoff as part of an in-path rule, and the rule matches packet flow for a preexisting connection, the individual connection is reset automatically. Connections that are preexisting and do not match an in-path rule with automatic kickoff enabled are unaffected. If you add a new rule with automatic kickoff, any preexisting connection that matches the new rule is not affected until the next service restart.

Figure 1-6. Enable automatic kickoff

In-Path Rules Network Services > In-Path Rules ?

▼ Add a New In-Path Rule ✕ Remove Selected Rules ⇅ Move Selected Rules...

Type: Auto Discover ▼

Source: { Subnet: All IP (IPv4 + IPv6) ▼

Destination: { Subnet: All IP (IPv4 + IPv6) ▼
Port: All Ports ▼
Domain Label: n/a ▼

VLAN Tag ID: all

Preoptimization Policy: None ▼

Latency Optimization Policy: Normal ▼

Data Reduction Policy: Normal ▼

Cloud Acceleration: Auto ▼ *Must be set to "Pass Through" if a Domain Label (see above) is selected*

Auto Kickoff: ☐

Neural Framing Mode: Always ▼

WAN Visibility Mode: Correct Addressing ▼

Position: End ▼

Description:

Enable Rule: ☒

Add

You use automatic kickoff primarily when you deploy SteelHeads in data protection environments. In data protection deployments, connections carrying the data replication traffic between the two storage arrays are often long-lived. This poses a problem if the connections are established as unoptimized or pass-through (for example, if the SteelHead is offline during connection setup), because the connections can remain unoptimized for a long time. Without the automatic kickoff on a SteelHead, you must manually intervene to reset the connections carrying data replication traffic on one of the storage arrays.

For more information about data protection deployment, see [“Data Protection Deployments” on page 391](#).

Although you can use automatic kickoff for any type of optimizable connection, the majority of connections for office applications—web, email, and so on—are comparatively short-lived and begin to be optimized after a brief period of time without any need for a reset.

When using the automatic kickoff feature, be aware of the following behaviors:

- Automatic kickoff does not have a timer.
A preexisting connection that remains inactive for a period of time is reset as soon as there is packet flow and it matches an in-path rule that has auto kickoff enabled. After the connection has been reset, an internal flag is set to prevent further kick offs for the connection unless the optimization service is restarted.
- Take note when you enable automatic kickoff to make sure you do not cause undesired behavior.
For example, in a design in which there is network asymmetry, if one or more SteelHead neighbors are configured and an in-path rule with automatic kickoff matches the connection, then the connection is kicked off even after detecting only one side of the handshake conversation.

For information about configuring the automatic kickoff feature, see the *Riverbed Command-Line Interface Reference Manual* and *SteelHead User Guide*.

Controlling optimization configuration examples

The following examples show common deployment configurations:

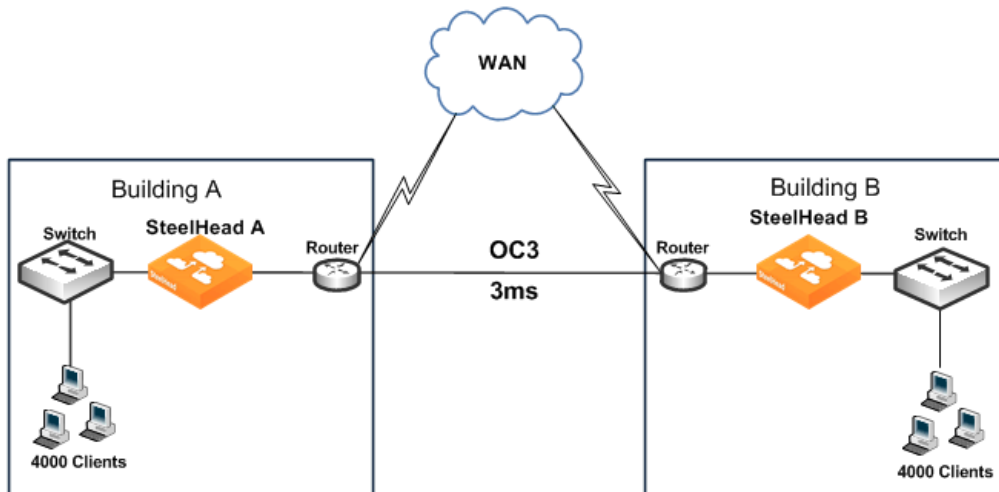
- [“Configuring high-bandwidth, low-latency environment” on page 42](#)
- [“Configuring pass-through transit traffic” on page 44](#)

Configuring high-bandwidth, low-latency environment

To show how in-path and peering rules might be used when designing SteelHead deployments, consider a network that has high bandwidth, low latency, and a large number of users.

Figure 1-7 shows this scenario occurring between two buildings at the same site. In this situation, you want to select SteelHead models to optimize traffic going to and from the WAN. However, you do not want to optimize traffic flowing between SteelHead A and SteelHead B.

Figure 1-7. High bandwidth utilization, low latency, and many connections between SteelHeads



You can achieve this result as follows:

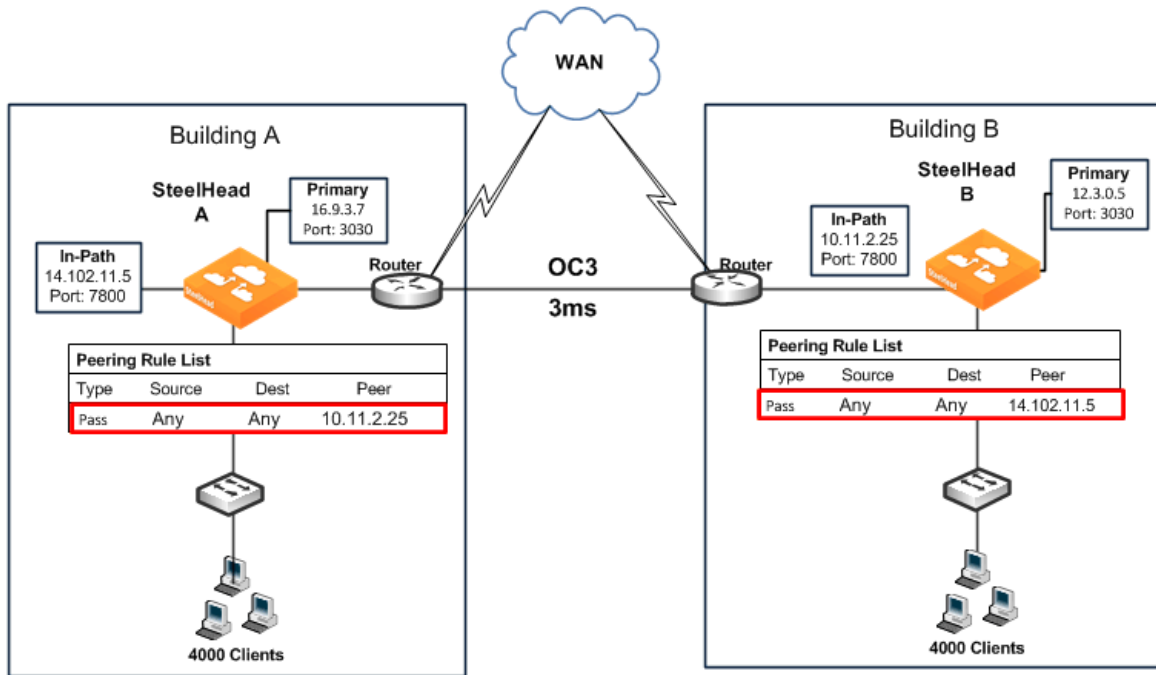
- **In-path Rules** - You can configure in-path rules on each of the SteelHeads (in Building A and Building B) so that the SteelHeads do not perform autodiscovery on any of the subnets in Building A and Building B. This option requires knowledge of all subnets within the two buildings, and it also requires that you update the list of subnets as the network is modified.
- **Peering Rules** - You can configure peering rules on SteelHead A and SteelHead B that pass through probe packets with in-path IP addresses of the other SteelHead (SteelHead A passes through probe packets with in-path IP addresses of SteelHead B, and vice versa). Using peering rules would require:
 - less initial configuration.
 - less ongoing maintenance, because you do not need to update the list of subnets in the list of peering rules for each of the SteelHeads.

Figure 1-8 shows how to use peering rules to prevent optimization from occurring between two SteelHeads and still allow optimization for traffic going to and from the WAN:

- SteelHead A has a Pass peering rule for all traffic coming from the SteelHead B in-path interface, so SteelHead A allows connections from SteelHead B to pass through it unoptimized.

- SteelHead B has a Pass peering rule for all traffic coming from the SteelHead A in-path interface, so SteelHead B allows connections from SteelHead A to pass through it unoptimized.

Figure 1-8. Peering rules for high utilization between SteelHeads



To configure a high-bandwidth, low-latency environment

1. On SteelHead A, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path peering rule pass peer 10.11.2.25 rulenum end
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path peering rule pass peer 14.102.11.5 rulenum end
```

If a packet does not apply to any of the configured peering rules, the auto-peering rule is used.

Configuring pass-through transit traffic

Transit traffic is data that is flowing through a SteelHead whose source or destination is not local to the SteelHead.

A SteelHead must optimize only traffic that is initiated or terminated at the site where it resides—any extra WAN hops between the SteelHead and the client or server greatly reduce the optimization benefits seen by those connections.

Figure 1-9 shows the SteelHead at the Chicago site detects transit traffic to and from San Francisco and New York (traffic that is not initiated or terminated in Chicago). You want the initiating SteelHead (San Francisco) and the terminating SteelHead (New York) to optimize the connection. You do not want the SteelHead in Chicago to optimize the connection.

Figure 1-9. Transit traffic

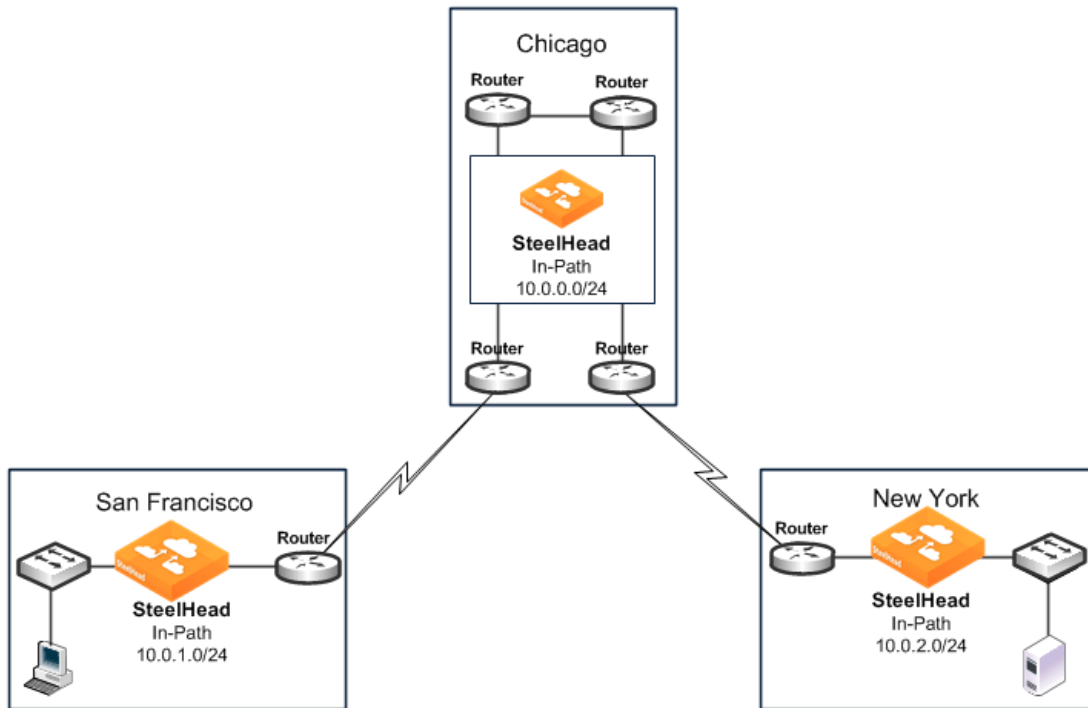


Figure 1-9 shows the possible solutions to resolve this transit traffic issue. These points do not include the configuration for features such as duplex, alarms, and DNS. Also assume that the default in-path rules are configured on all three SteelHeads. Because the default action for in-path rules and peering rules is to use autodiscovery, two in-path and two peering rules must be configured on the Chicago SteelHead.

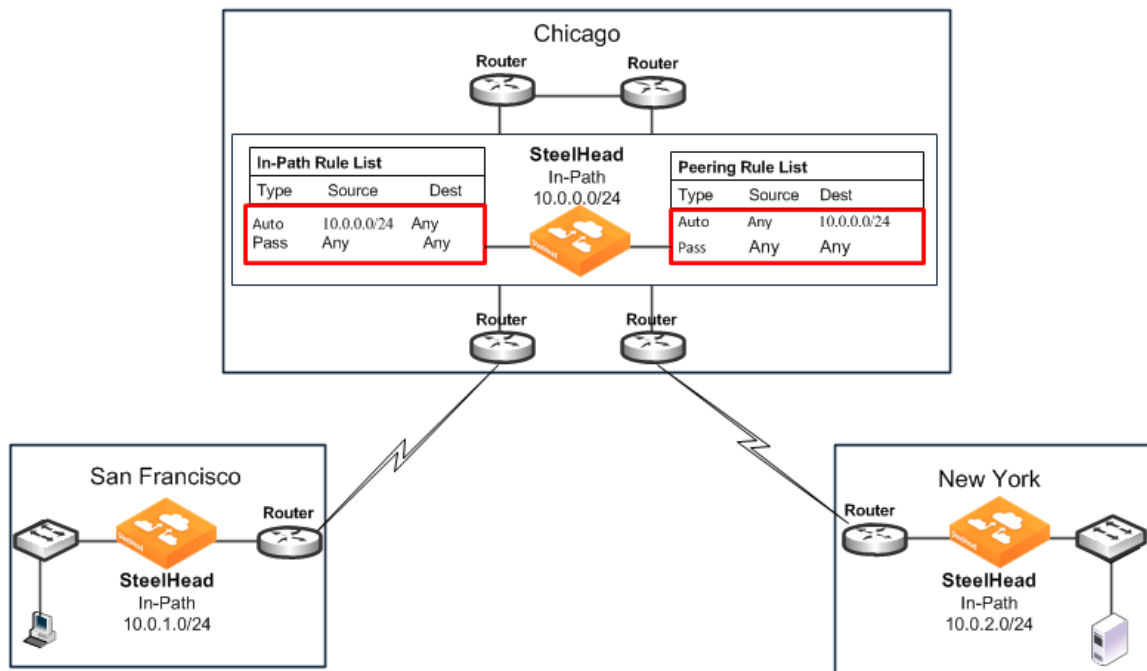
The following scenarios are possible solutions:

- **Configure manual peering and in-path rules** - Configure in-path and peering rules on the Chicago SteelHead to ignore transit traffic. You can configure peering rules for transit traffic by using the following commands:

```
enable
configure terminal
in-path rule auto srcaddr 10.0.0.0/24 rulenum end
in-path rule pass rulenum end
in-path peering rule auto dest 10.0.0.0/24 rulenum end
in-path peering rule pass rulenum end
```

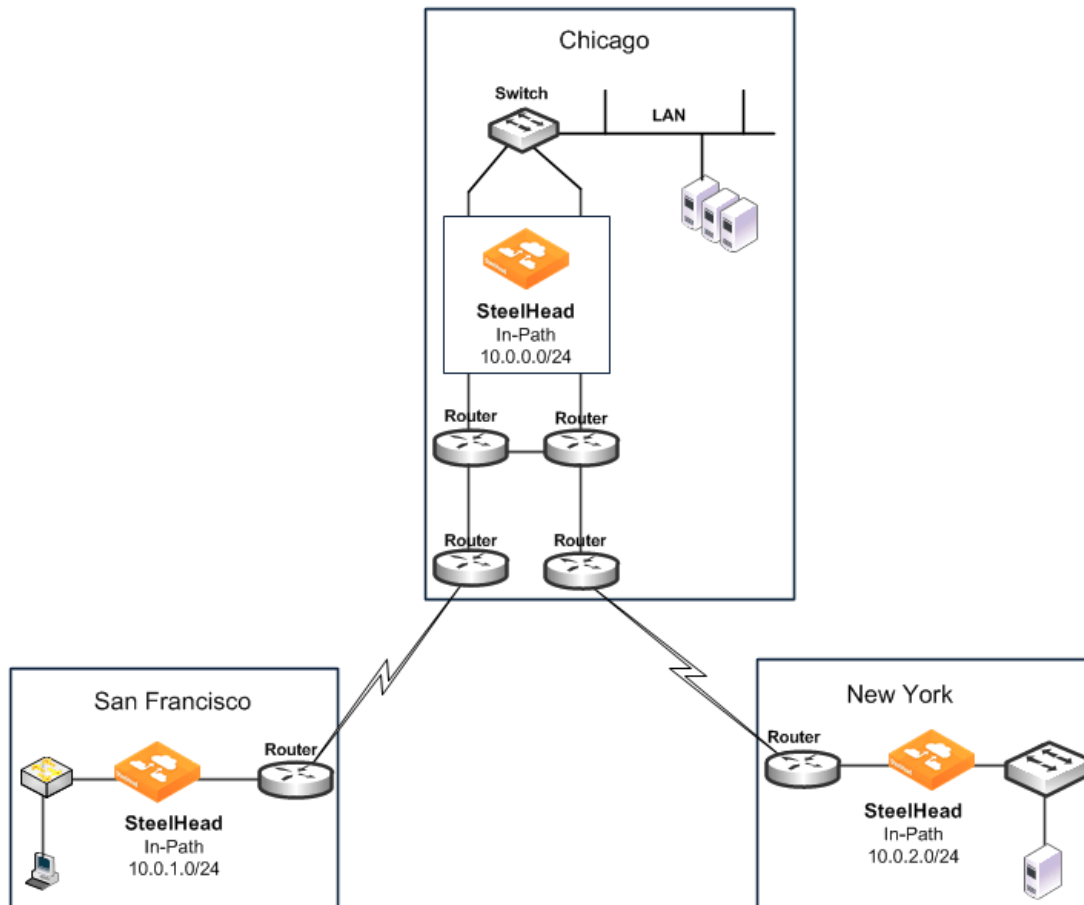
For information about peering rules, see [“Peering rules” on page 39](#).

Figure 1-10. Peering rules for transit traffic



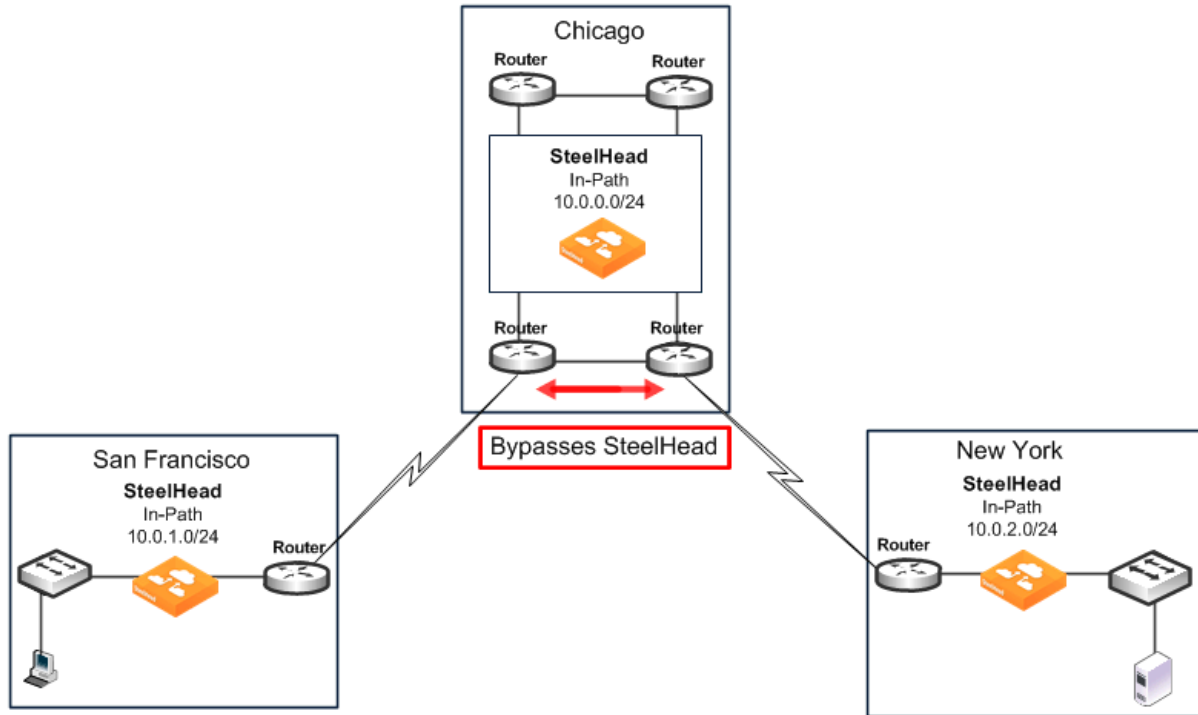
- **Adjust network infrastructure** - Relocate the Chicago SteelHead so that traffic initiated in San Francisco and destined for New York does not pass through the Chicago SteelHead. The Chicago SteelHead only detects traffic that is initiated or terminated at the Chicago site. **Figure 1-11** shows relocating the Chicago SteelHead to detect only traffic that is initiated or terminated at the Chicago site.

Figure 1-11. Resolving transit traffic by adjusting network infrastructure



- **Adjust traffic flow** - Configure the two routers at the Chicago site to bypass the Chicago SteelHead. [Figure 1-12](#) shows the flow of traffic (initiated or terminated in San Francisco or New York) when the routers at the Chicago site are configured to bypass the Chicago SteelHead.

Figure 1-12. Resolving transit traffic by adjusting traffic flow



- **Enable enhanced autodiscovery** - Enable enhanced autodiscovery on all of the SteelHeads (in San Francisco, Chicago, and New York). Enhanced autodiscovery enables SteelHeads to automatically find the first and the last SteelHead that a given packet must traverse and ensures that a packet does not become transit traffic. This feature is available in RiOS 4.0.x or later. For information about enhanced autodiscovery, see [“Configuring enhanced autodiscovery” on page 35](#).

Fixed-target in-path rules

A fixed-target in-path rule enables you to manually specify a remote SteelHead for optimization. As with all in-path rules, fixed-target in-path rules are executed only for SYN packets; therefore, they are configured on the initiating or client-side SteelHead. This section includes the following topics:

- [“Configuring a fixed-target in-path rule for an in-path deployment” on page 49](#)
- [“Fixed-target in-path rule for an out-of-path deployment” on page 50](#)

For information about in-path rules, see [“In-path rules” on page 38](#).

You can use fixed-target in-path rules in environments where the autodiscovery process cannot work.

A fixed-target rule requires the input of at least one target SteelHead; you can also specify an optional backup SteelHead.

Fixed-target in-path rules have several disadvantages compared to autodiscovery:

- You cannot easily determine which subnets to include in the fixed-target rule.
- Ongoing modifications to rules are needed as new subnets or SteelHeads are added to the network.
- Currently, you can specify two remote SteelHeads. All traffic is directed to the first SteelHead until it reaches capacity or until it stops responding to requests to connect. Traffic is then directed to the second SteelHead (until it reaches capacity or until it stops responding to requests to connect).

Note: Because of these disadvantages, fixed-target in-path rules are not as desirable as autodiscovery. In general, use fixed-target rules only when you cannot use autodiscovery.

LAN data flow has a significant difference depending on whether the fixed-target (or backup) IP address specified in the fixed-target in-path rule is for a SteelHead primary interface or its in-path interface.

Configuring a fixed-target in-path rule for an in-path deployment

In environments where the autodiscovery process does not work, use fixed-target in-path rules to target a remote in-path (physical or virtual) SteelHead. Configure the fixed-target in-path rule to target the remote SteelHead in-path interface.

Examples of environments where the autodiscovery process does not work are as follows:

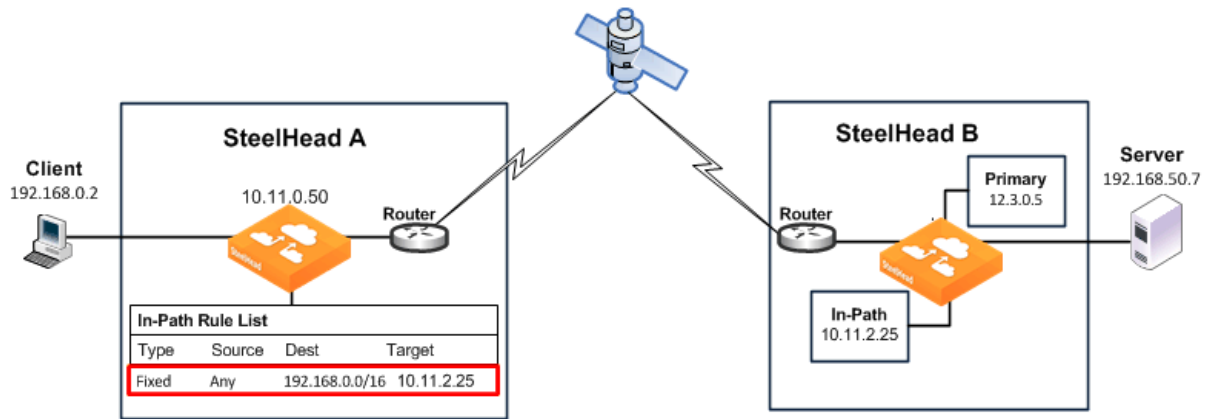
- Traffic traversing the WAN passes through a satellite or other device that strips off TCP options, including those used by autodiscovery.
- Traffic traversing the WAN goes through a device that proxies TCP connections and uses its own TCP connection to transport the traffic. For example, some satellite-based WANs use built-in TCP proxies in their satellite uplinks.

When the target IP address of a fixed-target in-path rule is a SteelHead in-path interface, the traffic between the server-side SteelHead and the server looks like client-to-server traffic; that is, the server detects connections coming from the client IP address. This process is the same as when autodiscovery is used.

Figure 1-13 shows how to use a fixed-target in-path rule to the SteelHead in-path interface. This example uses a fixed-target in-path rule to resolve an issue with a satellite. The satellite gear strips the TCP option from the packet, which means the SteelHead does not detect the TCP option, and the connection cannot be optimized.

To enable the SteelHead to detect the TCP option, you configure a fixed-target in-path rule on the initiating SteelHead (SteelHead A) that targets the terminating SteelHead (SteelHead B) in-path interface.

Figure 1-13. Fixed-target in-path rule to the SteelHead in-path interface



The fixed-target in-path rule specifies that only SYN packets destined for 192.168.0.0/16, SteelHead B subnets, are allowed through to the SteelHead B. All other packets are passed through the SteelHead.

You can configure in-path rules using the CLI.

To configure fixed-target in-path rule to an in-path address

- On SteelHead A, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path rule fixed-target target-addr 10.11.2.25 dstaddr 192.168.0.0/16 rulenum end
```

Fixed-target in-path rule for an out-of-path deployment

When you enable the remote SteelHead for out-of-path deployment, use fixed-target in-path rules to target the primary IP address. The most important caveat to this deployment method is that traffic to the remote server no longer uses the client IP address. Instead, the server detects connections coming to it from the out-of-path SteelHead primary IP address.

For deployment examples, see [“Out-of-Path Deployments” on page 387](#).

Best practices for SteelHead deployments

The following list represents best practices for deploying your SteelHeads. These best practices are not requirements, but we recommend that you follow these suggestions because they lead to designs that require the least amount of initial and ongoing configuration:

- **Use in-path designs** - Whenever possible, use a physical in-path deployment—the most common type of SteelHead deployment. Physical in-path deployments are easier to manage and configure than WCCP, PBR, and Layer-4 designs. In-path designs generally require no extra configuration on the connected routers or switches. If desired, you can limit traffic to be optimized on the SteelHead.

For details, see [“Physical in-path deployments” on page 197](#).

- **Use the correct cables** - To ensure that traffic flows not only when the SteelHead is optimizing traffic, but also when the SteelHead transitions to fail-to-wire mode, use the appropriate crossover or straight-through cable to connect the SteelHead to a router or switch. Verify the cable selection by removing power from the SteelHead and then test connectivity through it.

For details, see [“Choosing the correct cables” on page 205](#).

- **Set matching duplex speeds** - The most common cause of performance issues is duplex mismatch on the SteelHead WAN or LAN interfaces or on the interface of a device connected to the SteelHead. Most commonly, the issue is with the interface of a network device deployed prior to the SteelHead.

For information about duplex settings, see [“Cabling and duplex” on page 205](#). For information about troubleshooting duplex mismatch, see [“Physical in-path deployments” on page 197](#).

- **Minimize the effect of link state transition** - Use the Cisco **spanning-tree portfast** command on Cisco switches, or similar configuration options on your routers and switches, to minimize the amount of time an interface stops forwarding traffic when the SteelHead transitions to failure mode.

For details, see [“Fail-to-wire mode” on page 201](#).

- **Use serial rather than parallel designs** - Parallel designs are physical in-path designs in which a SteelHead has some, but not all, of the WAN links passing through it, and other SteelHeads have the remaining WAN links passing through them. Connection forwarding must be configured for parallel designs. In general, it is easier to use physical in-path designs where one SteelHead has all of the links to the WAN passing through it.

For information about serial designs, see [“Physical in-path deployments” on page 197](#). For information about connection forwarding, see [“Connection forwarding” on page 56](#).

- **Do not optimize transit traffic** - Ideally, SteelHeads optimize only traffic that is initiated or terminated at its local site. To avoid optimizing transit traffic, deploy the SteelHeads where the LAN connects to the WAN and not where LAN-to-LAN or WAN-to-WAN traffic can pass through (or be redirected to) the SteelHead.

For details, see [“Configuring pass-through transit traffic” on page 44](#).

- **Position your SteelHeads close to your network endpoints** - For optimal performance, minimize latency between SteelHeads and their respective clients and servers. By deploying SteelHeads as close as possible to your network endpoints (that is, place client-side SteelHeads as close to your clients as possible, and place server-side SteelHeads as close to your servers as possible).

- **Use WAN visibility modes that interoperate with monitoring, QoS, and security infrastructure** - RiOS currently supports four different WAN visibility modes, including granular control for their usage. This support ensures that the most appropriate mode is used.
For details, see [“WAN Visibility Modes” on page 63](#).
- **Use RiOS data store synchronization** - Regardless of the deployment type or clustering used at a site, RiOS data store synchronization can allow significant bandwidth optimization, even after a SteelHead or hard drive failure.
For details, see [“RIOS data store synchronization” on page 27](#).
- **Use connection forwarding and allow-failure in a WCCP cluster** - In a WCCP cluster, use connection forwarding and the **allow-failure** command between SteelHeads. For details, see [“Connection forwarding” on page 56](#).
- **Avoid using fixed-target in-path rules** - Use the autodiscovery feature whenever possible, thus avoiding the need to define fixed-target, in-path rules.
For information about autodiscovery, see [“Autodiscovery protocol” on page 31](#). For information about fixed-target in-path rules, see [“Fixed-target in-path rules” on page 48](#).
- **Understand in-path rules versus peering rules** - Use in-path rules to modify SteelHead behavior when a connection is *initiated*. Use peering rules to modify SteelHead behavior when it detects autodiscovery tagged packets.
For details, see [“In-path rules” on page 38](#) and [“Peering rules” on page 39](#).
- **Use Riverbed Professional Services or an authorized Riverbed Partner** - Training (both standard and custom) and consultation are available for small to large, and extra-large, deployments.
For details, contact Riverbed Professional Services by email at proserve@riverbed.com or go to <http://www.riverbed.com/services/index.html>.
- **Manually configure the primary interface in out-of-path deployments** - You must manually configure the IP address for any optimization interface, including the primary interface, in an out-of-path deployment. DHCP is not supported.

Network Integration Tools

This chapter describes SteelHead tools that you can integrate with your network. This chapter includes the following sections:

- [“Redundancy and clustering” on page 53](#)
- [“Fail-to-wire and fail-to-block” on page 55](#)
- [“Overview of link state propagation” on page 55](#)
- [“Connection forwarding” on page 56](#)
- [“Overview of simplified routing” on page 60](#)

Redundancy and clustering

This section describes redundant deployment of SteelHeads in your network. Redundant deployment ensures that optimization continues in case of a SteelHead failure. Redundancy and clustering options are available for each type of deployment. This section includes the following topics:

- [“Physical in-path deployments” on page 53](#)
- [“Virtual in-path deployments” on page 54](#)
- [“Out-of-path deployments” on page 54](#)

Physical in-path deployments

The following redundancy options for physical in-path deployments are available:

- **Primary and backup in-path deployment** - In a primary and backup deployment, two SteelHeads are placed in a physical in-path mode. One of the SteelHeads is configured as a primary, and the other is configured as the backup. The primary SteelHead (we recommend the SteelHead closest to the LAN) optimizes traffic, and the backup SteelHead constantly checks to make sure the primary SteelHead is functioning. If the backup SteelHead cannot reach the primary, or if the primary reaches maximum-connection-count capacity (called *admission control*), the backup SteelHead begins optimizing new connections until the primary returns to an operational state. After the primary has recovered, the backup SteelHead stops optimizing new connections and allows the

primary to resume optimizing new connections. However, the backup SteelHead continues to optimize connections that were made while the primary was down. While the backup SteelHead should not intercept or optimize new connections in normal operation, we recommend that you configure peering rules on the SteelHeads to prevent them from choosing each other as optimization peers even in the event of a check failure. In a serial deployment, we recommend planning and using a primary and backup configuration over a serial cluster configuration.

For details, see [“Primary and backup deployments” on page 213](#). For more information about admission control, see [“Admission control” on page 448](#).

- **Serial cluster in-path deployment** - In a serial cluster deployment, two or more SteelHeads are placed in a physical in-path mode, and the SteelHeads concurrently optimize connections. Because the SteelHead closest to the LAN detects the combined LAN bandwidth of all of the SteelHeads in the series, serial clustering is supported on only the higher-end SteelHead models. Serial clustering requires configuring peering rules on the SteelHeads to prevent them from choosing each other as optimization peers.

In general, we recommend primary and backup deployments due to simplicity in troubleshooting and sizing planning. Serial clusters can be appropriate in specific environments in which different subsets of traffic are partitioned to each SteelHead.

Deployments that use connection forwarding with multiple SteelHeads, each covering different links to the WAN, do not necessarily provide redundancy.

For information about serial clustering, see [“Serial cluster deployments” on page 215](#). For information about connection forwarding and multiple SteelHead deployment, see [“Connection forwarding” on page 56](#) and [“Configuring multiple WAN router deployments with connection forwarding” on page 228](#).

Virtual in-path deployments

For virtual in-path deployments, the clustering and redundancy options vary depending on which redirection method is being used. WCCP, the most common virtual in-path deployment method, allows options like N+1 redundancy and 1+1 redundancy.

For information about virtual in-path deployments, see [“Virtual In-Path Deployments” on page 245](#).

Out-of-path deployments

For an out-of-path deployment, you can configure two SteelHeads (a primary and a backup), with fixed-target rules that specify traffic for optimization. If the primary SteelHead becomes unreachable, new connections are optimized by the backup SteelHead. If the backup SteelHead is down, no optimization occurs, and traffic is passed through the network unoptimized.

The primary SteelHead uses an out-of-band (OOB) connection. The OOB connection is a single, unique TCP connection that communicates internal information only; it does not contain optimized data. If the primary SteelHead becomes unavailable, it loses this OOB connection and the OOB connection times out in approximately 40 to 45 seconds. After the OOB connection times out, the client-side SteelHead declares the primary SteelHead unavailable and connects to the backup SteelHead.

During the 40- to 45-second delay before the client-side SteelHead declares a peer unavailable, it passes through any incoming new connections; they are not black holed.

Although the client-side SteelHead is using the backup SteelHead for optimization, it attempts to connect to the primary SteelHead every 30 seconds. If the connection succeeds, the client-side SteelHead reconnects to the primary SteelHead for any new connections. Existing connections remain on the backup SteelHead for their duration. Immediately after a recovery from a primary failure, connections are optimized by both the primary SteelHead and the backup.


If both the primary and backup SteelHeads become unreachable, the client-side SteelHead tries to connect to both appliances every 30 seconds. Any new connections are passed through the network unoptimized.

For information about out-of-path deployments, see [“Out-of-Path Deployments” on page 387](#).

Fail-to-wire and fail-to-block

In physical in-path deployments, the SteelHead LAN and WAN ports that traffic flows through are internally connected by circuitry that can take special action in the event of a disk failure, a software crash, a runaway software process, or even loss of power to the SteelHead.

All SteelHead models and in-path network interface cards support fail-to-wire mode, where, in the event of a failure or loss of power, the LAN and WAN ports become internally connected as if they were the ends of a crossover cable, thereby providing uninterrupted transmission of data over the WAN. The default failure mode is fail-to-wire mode.

 SteelHead-v supports fail-to-wire or fail-to-block only when deployed with a Riverbed NIC. For more details, see the *SteelHead (Virtual Edition) Installation Guide*.


Certain in-path network interface cards also support a fail-to-block mode, where in the event of a failure or loss of power, the SteelHead LAN and WAN interfaces completely lose link status. When fail-to-block is enabled, a failed SteelHead blocks traffic along its path, forcing traffic to be rerouted onto other paths (where the remaining SteelHeads are deployed).


For information about fail-to-block mode, see [“Fail-to-block mode” on page 202](#). For information about SteelHead LAN and WAN ports and physical in-path deployments, see [“Logical in-path interface” on page 198](#). For information about physical in-path deployments, see [“Physical in-path deployments” on page 197](#).

Overview of link state propagation

In physical in-path deployments, link state propagation (LSP) can shorten the recovery time of a link failure. Link state propagation communicates link status between the devices connected to the SteelHead. When this feature is enabled, the link state of each SteelHead LAN-WAN pair is monitored. If either physical port loses link status, the other corresponding physical port brings its link down. Allowing link failure to quickly propagate through a chain of devices, LSP is useful in environments where link status is used for fast failure detection.

In RiOS 6.0 or later, link state propagation is enabled by default.

 SteelHead-c models do not support LSP.

 SteelHead-v running RiOS 8.0.3 with ESXi 5.0 and later using a Riverbed NIC card support LSP.

These SteelHead-v configurations do not support LSP:

- SteelHead-v models running ESX/ESXi 4.0 or 4.1
- SteelHead-v models running Microsoft Hyper-V
- SteelHead-v models running RiOS 8.0.2 and earlier

For information about physical in-path deployments, see [“Physical in-path deployments” on page 197](#). For more information about LSP, see [“Configuring link state propagation” on page 203](#).

Connection forwarding

For a SteelHead to optimize a TCP connection, it must detect all of the packets for that connection. When you use connection forwarding, multiple SteelHeads work together and share information about which connections are being optimized by each. With connection forwarding, the LAN interface forwards and receives connection-forwarding packets. This section includes the following topics:

- [“Configuring connection forwarding” on page 57](#)
- [“Multiple-interface support within connection forwarding” on page 58](#)
- [“Failure handling within connection forwarding” on page 58](#)
- [“Connection-forwarding neighbor latency” on page 59](#)

SteelHeads that are configured to use connection forwarding with each other are known as *connection-forwarding neighbors*. If a SteelHead detects a packet belonging to a connection that is optimized by a different SteelHead, it forwards it to the correct SteelHead. When a neighbor SteelHead reaches its optimization capacity, that SteelHead stops optimizing new connections but continues to forward packets for TCP connections being optimized by its neighbors.

You can use connection forwarding in both physical in-path deployments and virtual in-path deployments. In physical in-path deployments, connection forwarding is used between SteelHeads that are deployed on separate parallel paths to the WAN. In virtual in-path deployments, connection forwarding is used when the redirection mechanism does not guarantee that packets for a TCP connection are always sent to the same SteelHead. This includes the WCCP protocol, a commonly used virtual in-path deployment method.

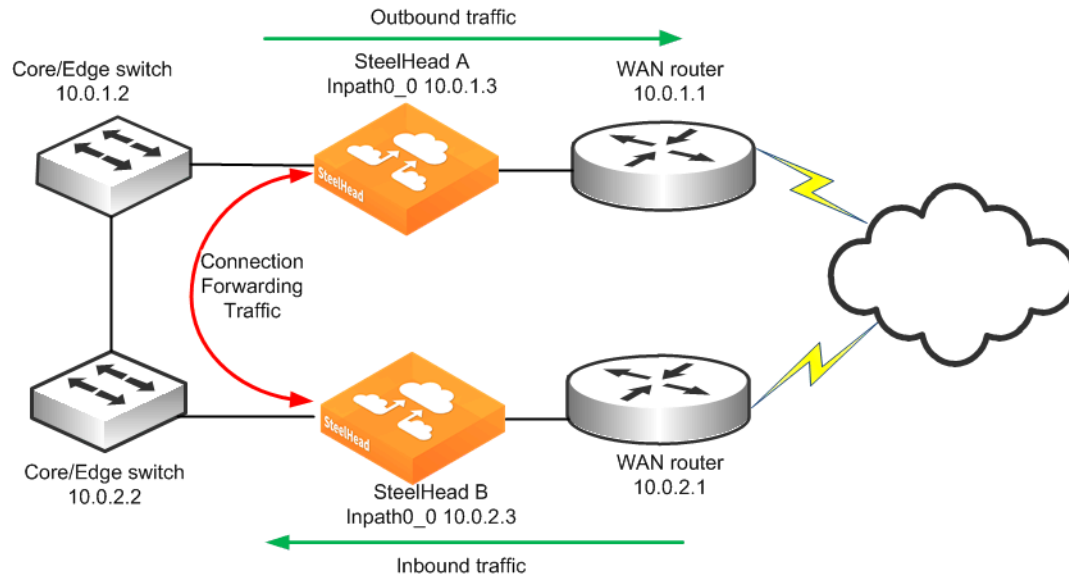
Important: We recommend that you configure support for multiple interfaces with connection forwarding, even if multiple interfaces are not used in your deployment. Enabling multiple-interface support can prevent memory allocation issues for SteelHead appliances that have been active for long periods of time. See [“Multiple-interface support within connection forwarding” on page 58](#) for more information.

Typically, it is easier to design physical in-path deployments that do not require connection forwarding. For example, if you have multiple paths to the WAN, you can use a SteelHead model that supports multiple in-path interfaces, instead of using multiple SteelHeads with single in-path interfaces. In general, serial deployments are preferred over parallel deployments.

For information about deployment best practices, see [“Best practices for SteelHead deployments” on page 51](#).

Figure 2-1 shows a site with multiple paths to the WAN. SteelHead A and SteelHead B can be configured as connection-forwarding neighbors. This configuration ensures that if a routing or switching change causes TCP connection packets to change paths, either SteelHead A or SteelHead B can forward the packets back to the correct SteelHead.

Figure 2-1. Connection forwarding SteelHeads



For information about connection forwarding and MTU sizing, see [“Connection-forwarding MTU considerations” on page 479](#).

Configuring connection forwarding

The following example is based on the assumption that the SteelHeads have already been configured properly for in-path interception.

To configure connection forwarding

1. On SteelHead A, connect to the CLI and enter the following commands:

```
enable
configure terminal
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadB main-ip 10.0.2.3
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
enable
configure terminal
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadA main-ip 10.0.1.3
```

When SteelHead A begins optimizing a new TCP connection, it communicates this activity to SteelHead B, provides the IP addresses and TCP port numbers for the new TCP connection, and defines a dynamic TCP port on which to forward packets.

If SteelHead B detects a packet that matches the connection, it takes the packet, alters its destination IP address to be the in-path IP address of SteelHead A, alters its destination TCP port to be the specific dynamic port that SteelHead A specified for the connection, and transmits the packet using its routing table.

In most environments, we recommend that you configure connection-forwarding SteelHeads to send traffic to each other through the LAN side of the network. Generally, the LAN-side network equipment is connected through low-latency network equipment with more than sufficient connectivity, and the WAN-side equipment might not be directly connected. To make sure that the connection-forwarding neighbor SteelHead sends traffic to each of their in-path IP addresses through the LAN, install a static route for the addresses whose next hop is the LAN gateway device.

For information about connection forwarding in multiple WAN routers, see [“Configuring basic connection forwarding” on page 228](#).

Multiple-interface support within connection forwarding

By default, SteelHeads communicate with neighbor appliances over a single in-path interface, on whatever is the lowest-numbered, enabled interface. If reachability is lost across the single interface, then the connection-forwarding capabilities are degraded or broken.

The **steelhead communication multi-interface enable** command allows all SteelHead neighbor in-path interface IP addresses to be visible to each peer. This visibility ensures neighbor communication if an interface fails. This command provides a level of interface redundancy; however, you can also think of the multiple-interface option as an improved version of the connection-forwarding protocol. We recommend that you enable multiple-interface support, regardless of the number of interfaces.

Connection-forwarding SteelHeads with multiple-interface support attempt to establish communication from every enabled in-path interface to every neighbor appliance in-path interface. Depending on traffic flow, you can forward optimized traffic between SteelHeads through any active in-path interfaces. Therefore, in typical environments, we recommend that all enabled and connected in-path interfaces on the SteelHeads be reachable by their connection-forwarding neighbors. Please consult Riverbed Professional Services or your account team for environments in which reachability between neighbor in-path interfaces is limited.

Important: SteelHeads that have multiple-interface support enabled cannot be used with SteelHeads that have multiple-interface support disabled. We recommend enabling multiple-interface support on all SteelHeads.

Failure handling within connection forwarding

By default, if a SteelHead loses connectivity to a connection-forwarding neighbor, the SteelHead stops attempting to optimize new connections. This behavior can be changed with the **steelhead communication allow-failure** command. If the **allow-failure** command is enabled, a SteelHead continues to optimize new connections, regardless of the state of its neighbors.

For virtual in-path deployments with multiple SteelHeads, including WCCP clusters, you must always use connection forwarding and the **allow-failure** command. Certain events, such as network failures and router or SteelHead cluster changes, can cause routers to change the destination SteelHead for TCP connection packets. When the destination changes, SteelHeads must be able to redirect traffic to each other to ensure that optimization continues.

For parallel physical in-path deployments, where multiple paths to the WAN are covered by different SteelHeads, connection forwarding is needed because packets for a TCP connection might be routed asymmetrically; that is, the packets for a connection might sometimes go through one path, and other times go through another path. The SteelHeads on these paths must use connection forwarding to ensure that the traffic for a TCP connection is always sent to the SteelHead that is performing optimization for that connection.

If the **allow-failure** command is used in a parallel physical in-path deployment, SteelHeads optimize only those connections that are routed through the paths with operating SteelHeads. TCP connections that are routed across paths without SteelHeads (or with a failed SteelHead) are detected by the asymmetric routing detection feature.

For physical in-path deployments, the **allow-failure** command is commonly used with the fail-to-block feature (on supported hardware). When fail-to-block is enabled, a failed SteelHead blocks traffic along its path, forcing traffic to be rerouted onto other paths (where the remaining SteelHeads are deployed).

For an example configuration, see [“Configuring connection forwarding with allow-failure and fail-to-block” on page 229](#).

You can configure your SteelHeads to automatically detect and report asymmetry within TCP connections as seen by the SteelHead. Asymmetric route auto-detection does not solve asymmetry; it simply detects and reports it and passes the asymmetric traffic unoptimized. For information about enabling asymmetric route auto-detection, see the *SteelHead User Guide*.

Connection-forwarding neighbor latency

In general, we recommend that the maximum round-trip latency between connection forwarding SteelHeads is less than one millisecond.

You can deploy SteelHeads so that moderate latency exists between connection forwarding SteelHeads. Latency has an impact on the optimized traffic because each optimized connection requires communication between all connection forwarding SteelHead neighbors in order to share state about recognizing flows for redirection.

The longest round-trip latency between any two connection forwarding SteelHeads should be less than one-fifth of the round-trip latency to the closest optimized remote site. This precaution ensures that connection forwarding communication does not cause the connection setup time to be greater for optimized connections compared to unoptimized connections for the closest remote site. Deployments with round-trip latencies higher than 10 milliseconds between connection forwarding SteelHeads should only be implemented after a technical consultation with Riverbed.

For more details, see the *SteelHead User Guide*.

Overview of simplified routing

Simplified routing avoids situations when a packet traverses a SteelHead more than once—this behavior is called *packet ricochet*. In environments where the SteelHead is installed in a subnet different than the clients and servers, simplified routing prevents packet ricochet for optimized traffic from the SteelHead.

Figure 2-2 shows an example of packet ricochet when the SteelHead default gateway is configured for the WAN router, the host sits on a different network than the SteelHead, and simplified routing is not enabled.

Figure 2-2. Packet ricochet when the SteelHead default gateway is on the WAN

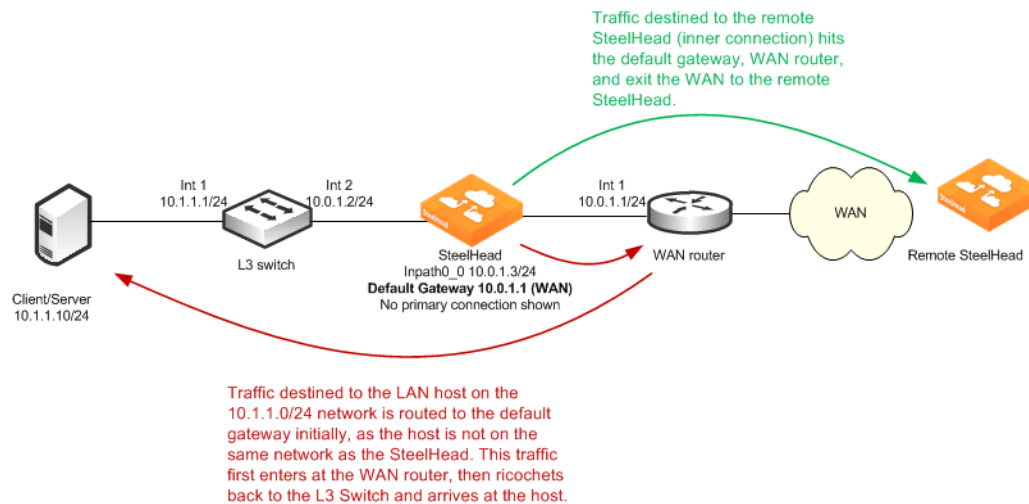
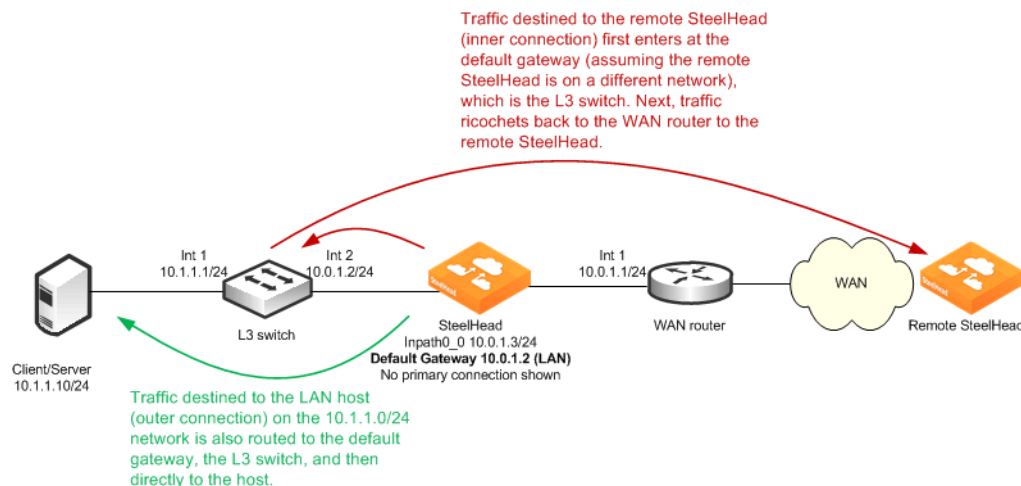


Figure 2-3 shows a similar packet ricochet scenario, but with the default gateway of the SteelHead pointed to the LAN L3 switch.

Figure 2-3. Packet ricochet when the SteelHead default gateway is on the LAN



In both [Figure 2-2](#) and [Figure 2-3](#), packets for some traffic take a suboptimal first hop from the SteelHead. While the detrimental effects of an extra hop are typically minor, packet ricochet causes problems in the following environments:

- Some environments that include firewalls or routers with ACLs might not permit traffic to ricochet or traverse back out the same interface as it came in.
- Some monitoring tools that rely on NetFlow or SNMP data count the ricocheted traffic as additional traffic.
- Packet ricochet causes the adjacent network devices to perform unnecessary work.

The packet ricochet scenarios only occur in physically in-path environments where the SteelHead is installed in a subnet different than the clients or servers. In these environments, you can avoid packet ricochet by either configuring static routes or by using simplified routing.

For example, [Figure 2-2](#) shows you can configure a static route for the host network, 10.1.1.0/24 to point directly to the 10.0.1.2 L3 switch, preventing this traffic from using the default gateway. However, the static route method often becomes administratively burdensome, especially in larger or complex LAN environments.

Simplified routing resolves packet ricochet, without using static routes or routing protocols, by building an IP to next-hop MAC address mapping learned from received packets. The SteelHead learns the correct MAC address by examining the packet's destination or source IP and MAC address.

Using [Figure 2-2](#) as an example, assume simplified routing is enabled. If an autodiscovery packet arrives from the WAN to the 10.1.1.10 host, the SteelHead detects the packet with the destination IP of 10.1.1.10 along with the destination MAC of the L3 switch, and it records the IP with associated MAC in its simplified routing table—also referred to as the *macmap table*. Whenever the SteelHead generates traffic destined to the 10.1.1.10 host, it uses the associated MAC of the L3 switch instead of the default gateway. This behavior avoids the packet ricochet.

Only use simplified routing for optimized traffic generated by the SteelHead, not pass-through traffic. For pass-through traffic, the SteelHead sends the packets out the opposite WAN or LAN interface as it came in. You can also use simplified routing when the destination IP is on a different subnet than the SteelHead in-path IP. If the destination IP resides on the same network, the SteelHead uses ARP for the correct MAC address. When the destination IP resides on a different network, then a simplified routing entry (if recorded) takes precedence over the default gateway, or by default, any configured static routes. To override the default behavior and have the static routes take precedence over simplified routing, use the **in-path simplified mac-def-gw-only** command.

Simplified routing plays an important role in maintaining VLAN ID when transmitting across the WAN when the SteelHead is deployed on an 802.1Q trunk and using the full address transparency WAN visibility mode.

For more information about simplified routing in physical in-path deployments, see [“Configuring simplified routing” on page 219](#).

WAN Visibility Modes

This chapter describes SteelHead WAN visibility modes and how to configure them. This chapter includes the following sections:

- [“Overview of WAN visibility” on page 63](#)
- [“Correct addressing” on page 64](#)
- [“Transparent addressing” on page 65](#)
- [“Implications of transparent addressing” on page 71](#)
- [“Out-of-band connection” on page 84](#)
- [“Configuring WAN visibility modes” on page 87](#)

For information about the factors you must consider before you design and deploy the SteelHead in a network environment, see [“Choosing the right SteelHead model” on page 28](#).

Overview of WAN visibility

Each LAN-side TCP connection that is optimized by a SteelHead is carried on a unique WAN-side connection. By configuring a WAN visibility mode for some or all optimized connections, you can control which IP addresses and TCP ports are used on these WAN-side TCP connections.

RiOS 6.0 or later offers the following options for configuring WAN visibility modes:

- **Correct addressing** - WAN-side connections use SteelHead IP addresses and SteelHead server ports.
- **Transparent addressing** - The following are transparent addressing options:
 - **Port transparency** - WAN-side connections use SteelHead IP addresses but use TCP server ports that mirror the LAN-side connection.
 - **Full transparency** - WAN-side connections mirror all IP addresses and TCP ports used on the LAN-side connection.
 - **Full transparency with forward reset** - The same as full transparency, with an additional packet during autodiscovery to aid with integration of stateful network devices on the WAN.

The most suitable WAN visibility mode depends primarily on your existing network configuration. For example, if you manage IP address-based or TCP port-based QoS policies for optimized traffic on your WAN or WAN routers, you might use full address transparency or port transparency. However, if you need your optimized traffic to pass through a content-scanning firewall that creates alarms when application ports are used on optimized traffic payload, you might use correct addressing instead. You can configure WAN visibility modes on the client-side SteelHead (where the connection is initiated).

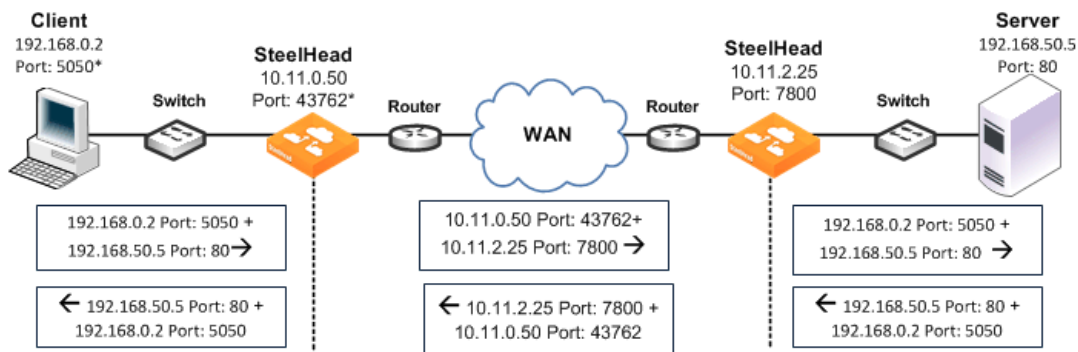
There can be different types of addressing modes on the same SteelHead. Choose the most appropriate addressing mode for your configuration, based on IP addresses, subnets, TCP ports, and VLAN.

Correct addressing

Correct addressing uses SteelHead IP addresses and port numbers in the TCP/IP packet header fields for optimized traffic in both directions across the WAN. By default, SteelHeads use correct addressing.

Figure 3-1 shows TCP/IP packet headers when correct addressing is used. The IP addresses and port numbers of your SteelHeads are visible across your WAN. Refer to Figure 3-1 to compare it to port transparency (Figure 3-2) and full address transparency (Figure 3-3) packet headers.

Figure 3-1. Correct addressing



Correct addressing uses the following values in your TCP/IP packet headers in both directions:

- **Client to client-side SteelHead** - Client IP address and port + server IP address and port.
- **Client-side SteelHead to server-side SteelHead** - Client-side SteelHead IP address and port + server-side SteelHead IP address and port.
- **Server-side SteelHead to server** - Client IP address and port + server IP address and port.

Correct addressing avoids networking risks that are inherent to enabling transparent addressing.

For information about configuring correct addressing, see [“Configuring WAN visibility modes” on page 87](#). For information about avoiding network risks, see [“Implications of transparent addressing” on page 71](#).

Correct addressing enables you to use the connection pooling optimization feature. Connection pooling works only for connections optimized using correct addressing. Connection pooling enables SteelHeads to create several TCP connections between each other before they are needed. When transparent addressing is enabled, SteelHeads cannot create the TCP connections in advance because they cannot detect what types of client and server IP addresses and ports are needed.

For information about connection pooling, see [“Connection pooling” on page 24](#).

Transparent addressing

This section describes port transparency and full address transparency. This section includes the following topics:

- [“Port transparency” on page 66](#)
- [“Full address transparency” on page 67](#)
- [“Full address transparency with forward reset” on page 69](#)

Transparent addressing reuses client and server addressing for optimized traffic across the WAN. Traffic is optimized, although addressing appears to be unchanged. Both optimized and pass-through traffic present identical addressing information to the router and network monitoring devices.

In RiOS 5.0.x or later, transparent addressing can be used in conjunction with many deployment configurations and features, including, but not limited to:

- physical in-path deployments (serial clusters, primary/backup, and deployments using connection forwarding).
- virtual in-path deployments (WCCP, PBR, Layer-4 switching, and SteelHead Interceptor deployments).
- autodiscovery, including enhanced autodiscovery.
- asymmetric route detection.
- QoS marking and classification.
- flow data export.

Transparent addressing does not support the following deployment configurations:

- Server-side out-of-path SteelHead configurations
- Fixed-target rules
- Connection pooling

You configure transparent addressing on the client-side SteelHead (where the connection is initiated). Both the server-side and the client-side SteelHeads must support transparent addressing (RiOS 5.0.x or later) for transparent addressing to work. You can configure a SteelHead for transparent addressing even if its peer does not support it. The connection is optimized, but it is not transparent.

When you use full or port transparency, SteelHeads add a TCP option field to the packet headers of optimized traffic. This TCP option field is sent between the SteelHeads. For transparency to work, this option must not be stripped off by intermediate network devices.

A given pair of SteelHeads can also have multiple types of transparent addressing enabled for different connections. For example, a pair of SteelHeads can use correct addressing for connections to a destination subnet and use full address transparency or port transparency for connections to another destination subnet. A pair of SteelHeads can also use correct addressing for connections to a destination port and use full address transparency or port transparency for connections to another destination subnet.

If both port transparency and full address transparency are acceptable solutions, port transparency is preferable. Port transparency avoids potential networking risks that are inherent in enabling full address transparency.

For details, see [“Implications of transparent addressing” on page 71](#).

Port transparency

Port transparency preserves your server port numbers in the TCP/IP header fields for optimized traffic in both directions across the WAN. Traffic is optimized even though the server port number in the TCP/IP header field appears to be unchanged. Routers and network monitoring devices deployed in the WAN segment between the communicating SteelHeads can view these preserved fields.

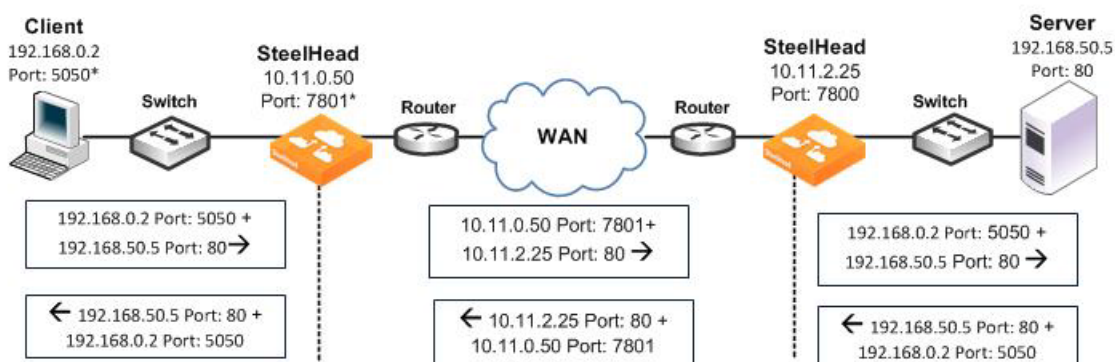
Port transparency does not require dedicated port configurations on your SteelHeads.

Port transparency provides server only port visibility. Port transparency does not provide client and server IP address visibility, nor does it provide client port visibility.

[Figure 3-2](#) shows TCP/IP packet headers when port transparency is enabled. Server port numbers are visible across your WAN.

To compare port transparency packet headers to correct addressing packet headers, see [Figure 3-1](#).

Figure 3-2. Port transparency



*In a production environment this port number is random for each connection.

Port transparency uses the following values in your TCP/IP packet headers in both directions:

- **Client to client-side SteelHead** - Client IP address and port + server IP address and port.
- **Client-side SteelHead to server-side SteelHead** - Client-side SteelHead IP address and port + server-side SteelHead IP address with server port.
- **Server-side SteelHead to server** - Client IP address and port + server IP address and port.

Use port transparency if you want to manage and enforce QoS policies that are based on destination ports. If your WAN router is following traffic classification rules that are written in terms of TCP destination port numbers, port transparency enables your routers to use existing rules to classify the traffic without any changes.

Port transparency enables network analyzers deployed within the WAN (between the SteelHeads) to monitor network activity, and to capture statistics for reporting, by inspecting traffic according to its original TCP destination port number.

For information about configuring port transparency, see [“Configuring WAN visibility modes” on page 87](#).

Full address transparency

This section describes full address transparency. This section includes the following topics:

- [“Overview of Full address transparency” on page 67](#)
- [“Configuring VLANs and full address transparency” on page 69](#)
- [“Full address transparency with forward reset” on page 69](#)

Overview of Full address transparency

Full address transparency preserves your client and server IP addresses and port numbers in the TCP/IP header fields for optimized traffic in both directions across the WAN. VLAN tags can also be preserved. Traffic is optimized even though these TCP/IP header fields appear to be unchanged. Routers and network monitoring devices deployed in the WAN segment between the communicating SteelHeads can view these preserved fields.

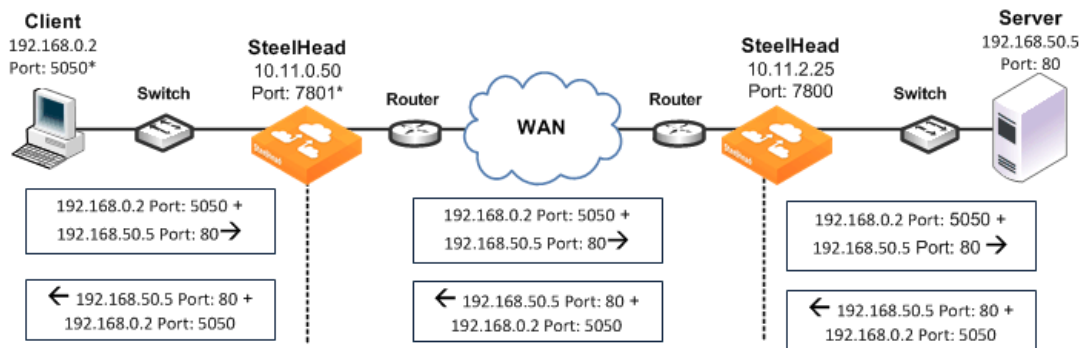
[Figure 3-3](#) shows how TCP/IP packet headers might be addressed when full address transparency is enabled. In this example, SteelHead IP addresses and port numbers are no longer visible on the optimized connections. Client and server IP addresses and port numbers are now visible in both directions across the WAN.

When you enable full address transparency, you have several addressing options for the out-of-band (OOB) connection. The type of addressing you configure for your OOB connection ultimately determines whether the SteelHead in-path IP addresses are used in the TCP/IP packet headers.

For information about OOB, see [“Out-of-band connection” on page 84](#).

To compare full address transparency packet headers with correct addressing packet headers, see [Figure 3-1](#).

Figure 3-3. Full address transparency



*In a production environment this port number is random for each connection.

In this example, full address transparency uses the following values in the TCP/IP packet headers in both directions:

- **Client to client-side SteelHead** - Client IP address and port + server IP address and port.
- **Client-side SteelHead to server-side SteelHead** - Client IP address and port + server IP address and port.
- **Server-side SteelHead to server** - Client IP address and port + server IP address and port.

For information about configuring full address transparency, see [“Configuring WAN visibility modes” on page 87](#).

If both port transparency and full address transparency are acceptable solutions, port transparency is preferable. Port transparency mitigates potential networking risks that are inherent in enabling full address transparency.

For details, see [“Implications of transparent addressing” on page 71](#).

However, if you must use your client or server IP addresses across your WAN, full address transparency is your only configuration option. Full address transparency enables network monitoring applications deployed within the WAN (between the SteelHeads) to measure traffic sent to the WAN by the end-host. Network routers can also perform load balancing and policy-based routing. Full address transparency also enables you to manage and enforce QoS policies based on port numbers or IP addresses.

Important: When full address transparency is enabled, router QoS policies cannot distinguish between optimized and unoptimized traffic, even though an optimized packet might represent much more data.

Full address transparency also enables the use of Network Address Translation (NAT). With correct addressing, SteelHeads use their own IP addresses in the packet header, which NAT does not recognize. When full address transparency is enabled, the original client and server IP addresses are used, and the connections are recognizable to NAT. However, the type of addressing you configure for your out-of-band (OOB) connection ultimately determines whether the SteelHead in-path IP addresses are used in the TCP/IP packet headers.

Full address transparency also supports several transparency options for the OOB connection.

For information about OOB, see “[Out-of-band connection](#)” on page 84.

Important: Some firewalls, QoS devices, and other stateful devices might require additional configuration to successfully allow full transparency connections to optimize. Search the Riverbed Knowledge Base for information about any particular device.

Configuring VLANs and full address transparency

Full address transparency supports transparent VLANs. You can configure full address transparency so that optimized traffic remains on the original VLANs. Because you can keep traffic on the original VLANs, full address transparency enables you to perform VLAN-based QoS on the WAN side of the SteelHead.

Note: You must first configure WAN visibility full address transparency for VLAN transparency to function correctly.

To configure full address transparency for a VLAN

- On the SteelHead, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path peering auto
in-path simplified routing all
in-path vlan-conn-based
in-path mac-match-vlan
no in-path probe-caching enable
in-path probe-ftp-data
in-path probe-mapi-data
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

If packets on your network use two different VLANs in the forward and reverse directions, see the Riverbed Knowledge Base article, *Understanding VLANs and Transparency*, located at <https://supportkb.riverbed.com/support/index?page=content&id=S15176>.

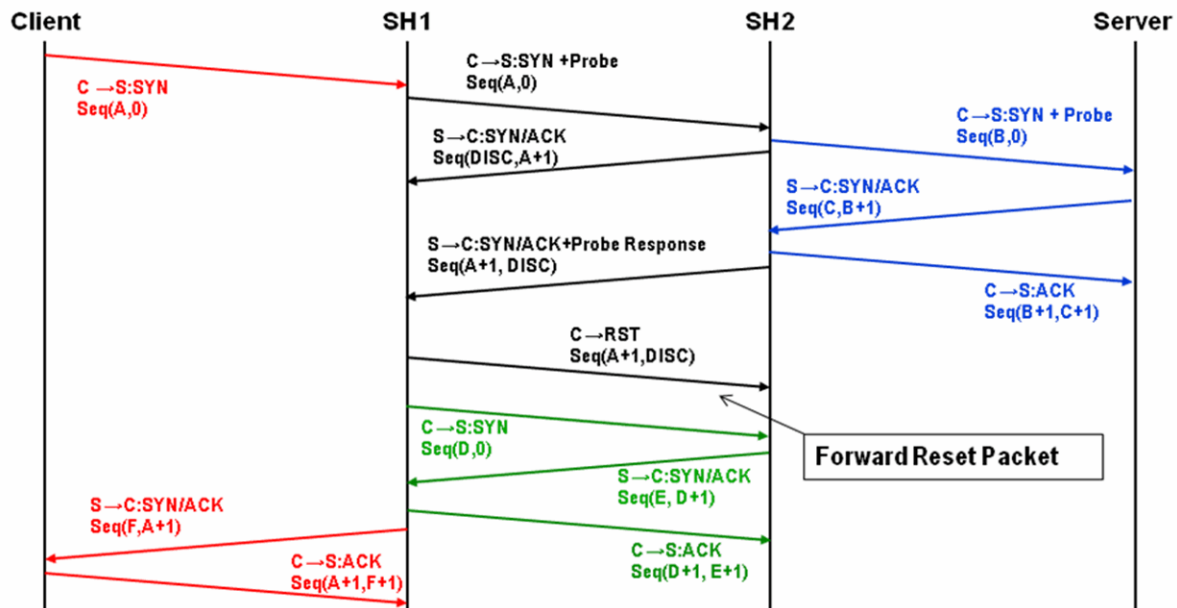
Full address transparency with forward reset

The *Full Address Transparency with Forward Reset* mode is similar to the *Full Address Transparency WAN visibility* mode. Like Full Address Transparency, use this mode to preserve the client and server IP addresses and TCP ports used for the WAN-side TCP connections between SteelHeads. The difference between the two modes happens during the autodiscovery phase, during which TCP reset packets are transmitted on the WAN. These packets help network devices like stateful firewalls separate TCP state between the SteelHeads discovery phase and the data transmission phase.

Except for the TCP reset packets during the discovery phase, there are no other differences between Full Address Transparency and Full Address Transparency with Forward Reset, including the addressing of the WAN-side TCP connection, considerations for 802.1Q VLAN tracking, and OOB transparency.

Figure 3-4 shows the autodiscovery packet flow when using the Full Transparency with Forward Reset mode and enhanced autodiscovery. The packet marked *Forward Reset* is the only difference between this mode and the Full Transparency mode.

Figure 3-4. Full transparency with forward reset



In Full Address Transparency with Forward Reset mode, TCP reset packets are transmitted by the initiating SteelHead immediately after the remote SteelHead is discovered. The reset packets traverse the WAN and are absorbed by the remote SteelHead. The packets aid any TCP-aware device on the WAN to understand that the sequence numbers used during the autodiscovery phase is different from the sequence numbers used during the data transmission phase. Example devices include:

- firewalls or other network security devices on the WAN that statefully track TCP sessions, and devices that might block WAN-side SteelHead connections from being created due to seeing different sequence numbers in use.
- QoS devices that alter TCP headers to affect congestion. Examples include Blue Coat Packetshaper appliances using rate policies, and the Allot Netenforcer.

Important: Some firewalls, QoS devices, and other stateful devices might require additional configuration to successfully optimize connections using the full transparency with forward reset connections to operate. Search the Riverbed Knowledge Base for information about any particular device.

Implications of transparent addressing

This section describes some of the common problems that are inherent to transparent addressing:

- [“Stateful systems” on page 71](#)
- [“Network design issues” on page 71](#)
- [“Integration into networks using NAT” on page 75](#)

The problems described in this section occur with all proxy-based solutions.

Stateful systems

Transparent addressing can have an impact on systems that monitor or alter the state of TCP connections on the WAN for reporting, security, or congestion control. For example, some stateful firewalls might detect the difference in sequence numbers between the autodiscovery phase and the data transmission phase of fully transparent connections, and then react by raising alarms or disallowing connections between the SteelHeads. Using the Full Transparency with Forward Reset mode might alleviate this issue, but might also cause monitoring systems to record more TCP connection activity across the WAN than is actually present.

Transparent addressing also does not work with intrusion detection and prevention systems that perform stateful packet inspection. SteelHeads use a proprietary Riverbed application protocol to communicate. When intrusion detection and prevention systems perform stateful packet inspections, they expect an application protocol based on the port numbers of the original client and server connection. When these systems discover the Riverbed proprietary application protocol, they perceive this as a mismatch and either log the packet, drop it, trigger an alarm, or perform all these actions.

You can avoid these problems with stateful systems, which are inherent to transparent addressing, by using correct addressing.

Network design issues

This section describes some of the common networking problems that are inherent to transparent addressing. This section includes the following topics:

- [“Network asymmetry” on page 71](#)
- [“Misrouting optimized traffic” on page 72](#)
- [“Firewalls located between SteelHeads” on page 74](#)

Network asymmetry

Enabling full address transparency increases the likelihood of problems inherent to asymmetric routing.

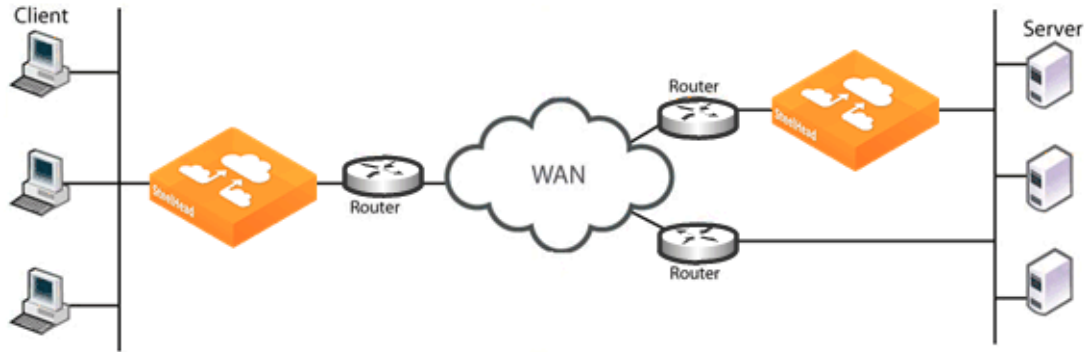
For a connection to be optimized, its packets to and from its LAN hosts must pass through either:

- one or more in-path interfaces on the same SteelHead, or
- one or more in-path interfaces on SteelHeads that are configured as connection-forwarding neighbors.

When full address transparency is used, WAN-side routers detect the client or server addresses in the optimized connections packets and use those addresses to make routing decisions. If the router has a route to the client or server that does not pass through a SteelHead, and it transmits the optimized packets on that route, the optimized and LAN-side connections might fail.

Figure 3-5 shows a network in which a link to the server location does not have a SteelHead installed. Depending on the exact routing configuration in the **Figure 3-5**, it is possible that correct addressing can work, but full transparency does not, because the optimized traffic from the client side might be sent through the link that does not have the SteelHead.

Figure 3-5. Server-side asymmetric network



To ensure that all required traffic is optimized and accelerated, you must install a SteelHead on every possible path that a packet traverses. Connection forwarding must also be configured and enabled for each SteelHead.

If there is a path that does not have a SteelHead, it is possible that some traffic will not be optimized. You can avoid this type of asymmetric routing problem, which is inherent to transparent addressing, by using correct addressing.

For information about connection forwarding, see [“Connection forwarding” on page 56](#). For information about how to eliminate asymmetric routing problems, see [“Troubleshooting SteelHead Deployment Problems” on page 463](#).

Note: With RiOS 3.0.x or later, you can configure your SteelHeads to automatically detect and report asymmetric routes within your network. For details, see the *SteelHead User Guide*.

Misrouting optimized traffic

Enabling transparent addressing introduces the likelihood of misrouting optimized traffic in the event of a SteelHead failure.

SteelHeads use a proprietary Riverbed protocol to communicate. Normally, a functioning server-side SteelHead receives a packet from the WAN and converts the packet to its native format before forwarding it to the server.

In an environment in which transparent addressing is used, if the server-side SteelHead is not functioning, or if a packet is routed along an alternative network path, the packet might go from the client-side SteelHead directly to the server. Because the server-side SteelHead does not have an opportunity to convert the packet to its native format, the server cannot recognize it, and the connection fails.

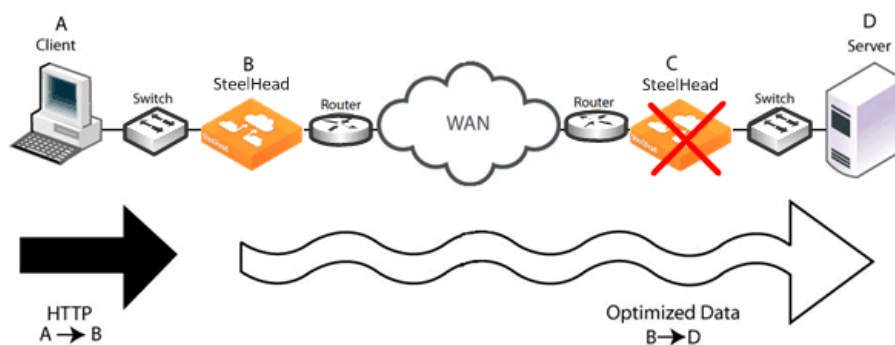
In most cases, the server is able to detect whether a packet contains invalid payload information or, in this case, has an unrecognizable format, and rejects the packet. Assuming that the server does detect that the format is unrecognizable, the server rejects the packet and resets the TCP connection. If the client TCP connection is reset, the client can reconnect to the server without any SteelHead involvement.

This type of traffic misrouting can occur in both directions across the WAN. If the client-side SteelHead experiences a failure, or if an alternate network path exists from the server to the client, traffic might go from the server-side SteelHead directly to the client.

Important: Before enabling and using full address transparency, carefully consider the risks and exposures in the event that a server accepts and routes a packet that has an unrecognizable format.

Figure 3-6 shows traffic being misrouted when the server-side SteelHead fails on a network using transparent addressing.

Figure 3-6. Transparent addressing and misrouting optimized traffic



The failure scenario follows this process:

1. Client A sends HTTP data to the server.
2. SteelHead B receives the HTTP data optimizes it. SteelHead B eventually transmits packets carrying the optimized data toward SteelHead C, but due to the transparent addressing mode, the packets are addressed to Server D.
3. SteelHead C suffers a failure and is in fail-to-wire mode, so all packets are traversing it, including the packets from SteelHead B.
4. Server D receives the packets from SteelHead B, but does not recognize the packet format, so the connection fails or suffers an application-dependent error.

You can avoid this type of misrouting problem, which is inherent to transparent addressing, by using correct addressing.

If correct addressing is configured for this scenario, the client-side SteelHead detects that the server-side SteelHead has failed. The client-side SteelHead *automatically* resets the client connection, allowing the client to connect directly to the server without SteelHead involvement.

Firewalls located between SteelHeads

There are addressing issues that you must be aware of if your firewall inspects traffic between two SteelHeads (Figure 3-7).

Figure 3-7. Firewalls and transparent addressing



The following table summarizes deployment configurations that might affect addressing when a firewall inspects traffic between two SteelHeads. Firewall behavior differs, depending on the type of addressing being used. A Yes value indicates that your firewall will perform as expected.

Firewall configuration	Full address transparency	Port transparency	Correct addressing
Firewall rules based on a server port	Yes	Yes Note: This configuration does not support active FTP.	Yes, if the following conditions are true: <ul style="list-style-type: none"> ■ The firewall checks on the session establishment. ■ The firewall is enabled. ■ The firewall allows port 7800 traffic.
Firewall rules based on IP addresses	Yes	Yes, if the following conditions are true: <ul style="list-style-type: none"> ■ IP-based rules are based only on server addresses. ■ Probe caching is disabled. 	Yes, if the following conditions are true: <ul style="list-style-type: none"> ■ The firewall checks on the session establishment. ■ The firewall is enabled. ■ Probe caching is disabled. For information about disabling probe caching, see the <i>Riverbed Command-Line Interface Reference Manual</i>.

For information about stateful firewalls and intrusion detection and prevention systems, see “[Stateful systems](#)” on page 71.

Integration into networks using NAT

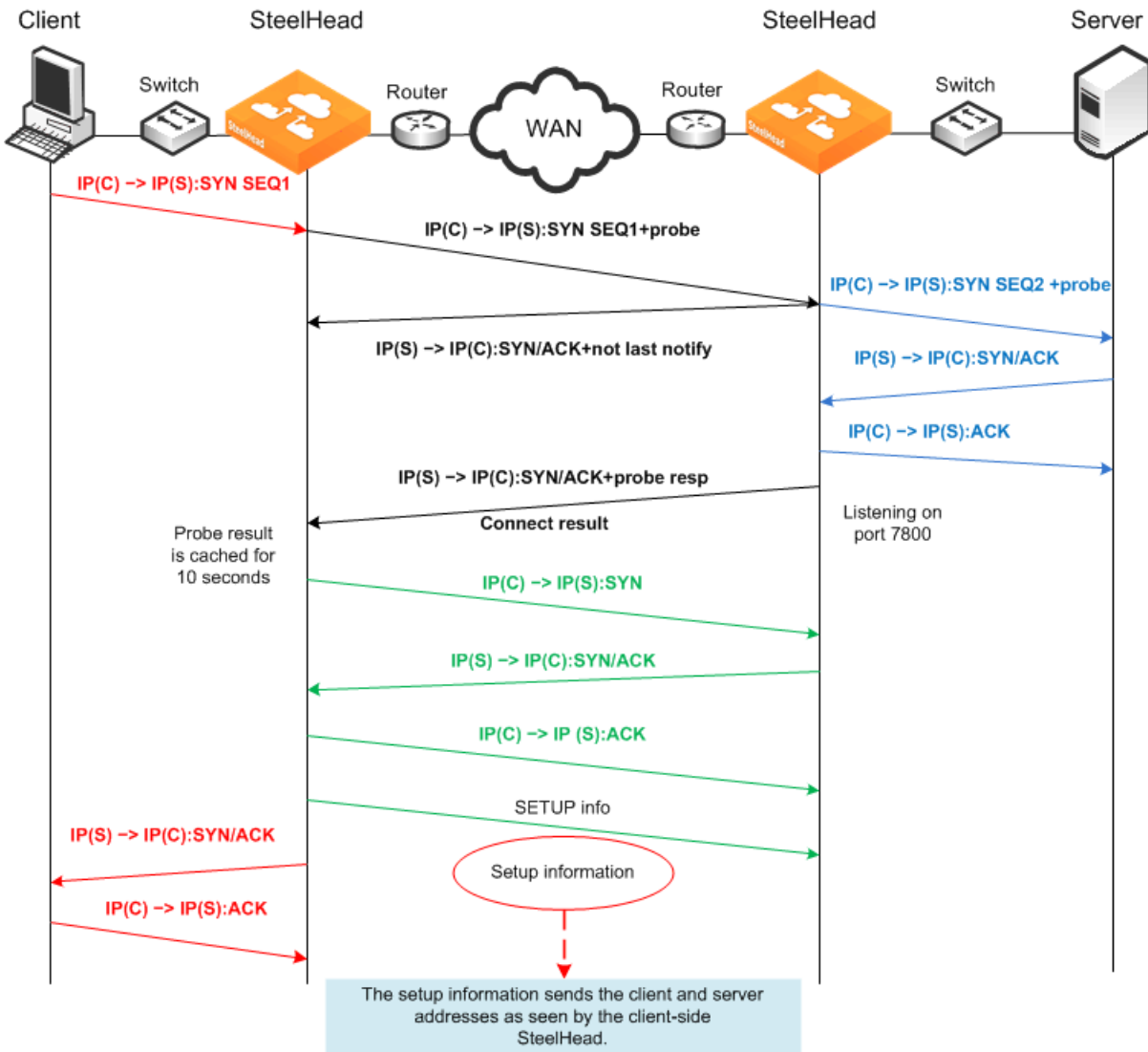
Network address translation (NAT) affects the addresses the SteelHead uses in different ways, depending on which addressing mode is in use. This section provides several NAT deployment scenarios using various addressing modes:

- [“NAT deployment using enhanced autodiscovery and full transparency” on page 76](#)
- [“NAT deployment using fixed-target rules” on page 76](#)
- [“NAT deployment using correct and port transparency addressing modes” on page 79](#)
- [“Client-side source NAT using enhanced autodiscovery and full transparency” on page 81](#)
- [“Failed client-side source NAT deployment using enhanced autodiscovery and correct addressing” on page 82](#)
- [“Dual NAT deployment using enhanced autodiscovery and full transparency” on page 83](#)
- [“Failed dual NAT deployment using enhanced autodiscovery and correct addressing” on page 84](#)

NAT deployment using enhanced autodiscovery and full transparency

Figure 3-8 shows full transparency addressing mode. The client and server IP addresses are preserved, and the presence of the full transparency TCP is a signal to the server-side SteelHead that it can use the addresses arriving from the WAN. This behavior ensures that any translation is preserved for optimized traffic.

Figure 3-8. Enhanced autodiscovery and full transparency mode

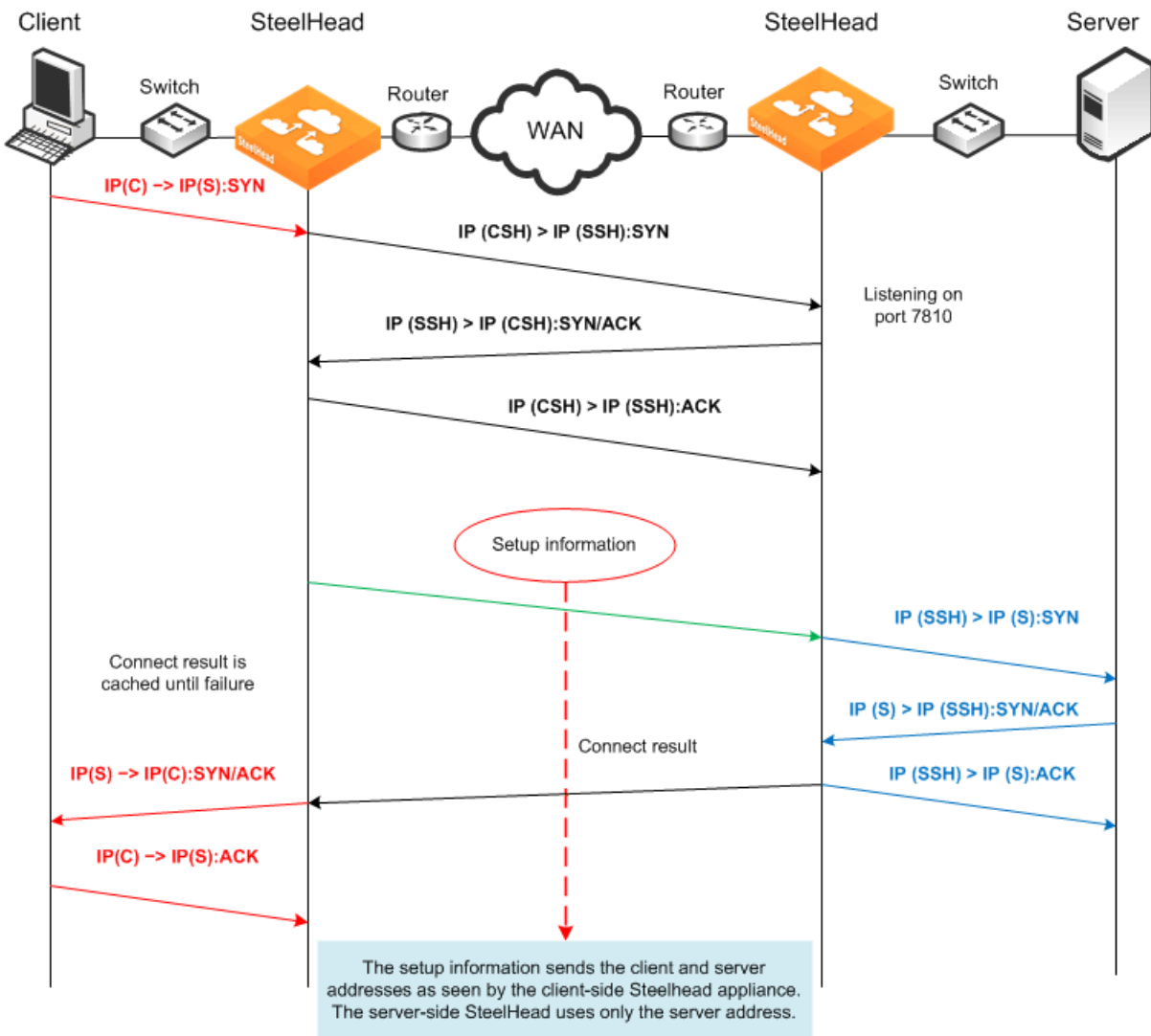


NAT deployment using fixed-target rules

When you use a fixed-target rule to the primary IP address of a remote SteelHead, the remote SteelHead makes a connection to the destination IP address detected on the initiating SteelHead. Figure 3-9 shows that the source address is the primary IP address of the remote SteelHead.

When you use a fixed-target rule to the in-path IP address of a remote SteelHead, the remote SteelHead makes a connection to the destination IP address detected on the initiating SteelHead. The remote SteelHead also uses the same source address detected on the initiating SteelHead.

Figure 3-9. Fixed-target rule to primary IP address



In Figure 3-9, the out-of-path packet flow on incoming connection requests is very similar to an established in-path partnership. However, there is an important distinction in that the IP address of the server-side SteelHead replaces the IP address of the client in communication between the server-side SteelHead and the destination server. The traffic travels the following route:

1. The packet is created coming from the initiator client (C) IP address to the destination server (S) IP address.
2. The client-side SteelHead sends a packet to port 7810 on the server-side SteelHead, requesting to open a session.
3. The server-side SteelHead acknowledges the connection request.

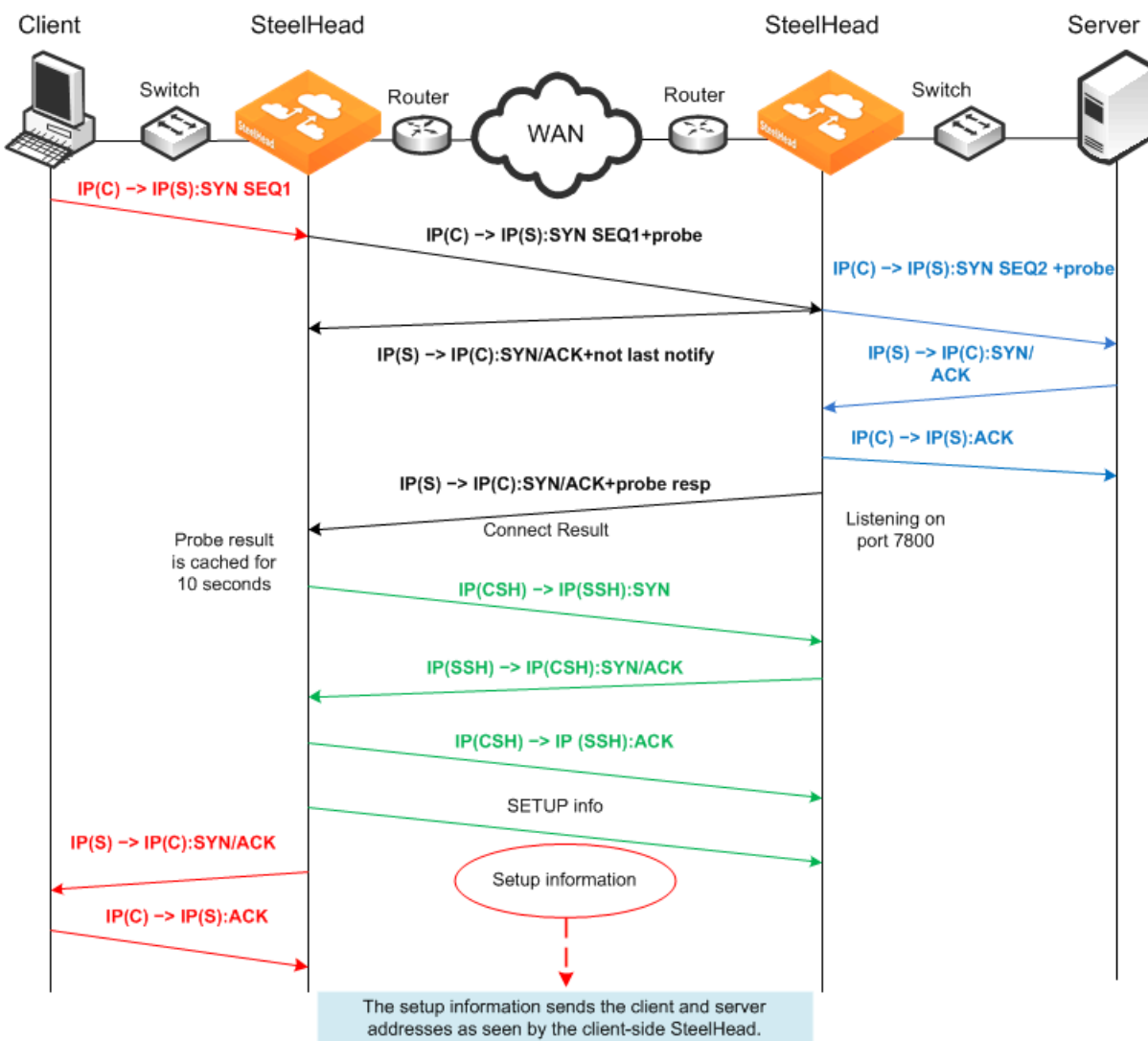
4. The client-side SteelHead acknowledges the connection.
5. The client-side SteelHead sends session setup information to the server-side SteelHead.
6. The server-side SteelHead forwards the original connection request to the destination server, replacing the client IP address with the server-side SteelHead IP address.
7. The destination server acknowledges the connection request.
8. The server-side SteelHead sends a packet acknowledgment to the destination server.
9. The server-side SteelHead sends the connection acknowledgment to the client-side SteelHead.
10. The client-side SteelHead sends the acknowledgment packet to the requesting client.
11. The client sends an acknowledgment to the destination server.
12. The client-side SteelHead discards the client acknowledgment.

Some applications and protocols require that the server initiate a new session or that they see the IP address of the requesting client. These applications and protocols do not function in this configuration. Consider using an in-path deployment, a WCCP deployment, or use rules on the SteelHead to pass through this traffic.

NAT deployment using correct and port transparency addressing modes

In both correct and port transparency addressing modes, whatever IP addresses are detected on the initiating SteelHead (typically, the client-side SteelHead) are used by the corresponding SteelHead on the remote side as [Figure 3-10](#) shows. This deployment can bypass any NAT that occurs in the WAN between the SteelHeads. To ensure that NAT is still used for the optimized traffic, you must configure the full transparency addressing mode for this traffic.

Figure 3-10. Enhanced autodiscovery in correct and port transparency modes



In this example, the TCP connection request travels the following route:

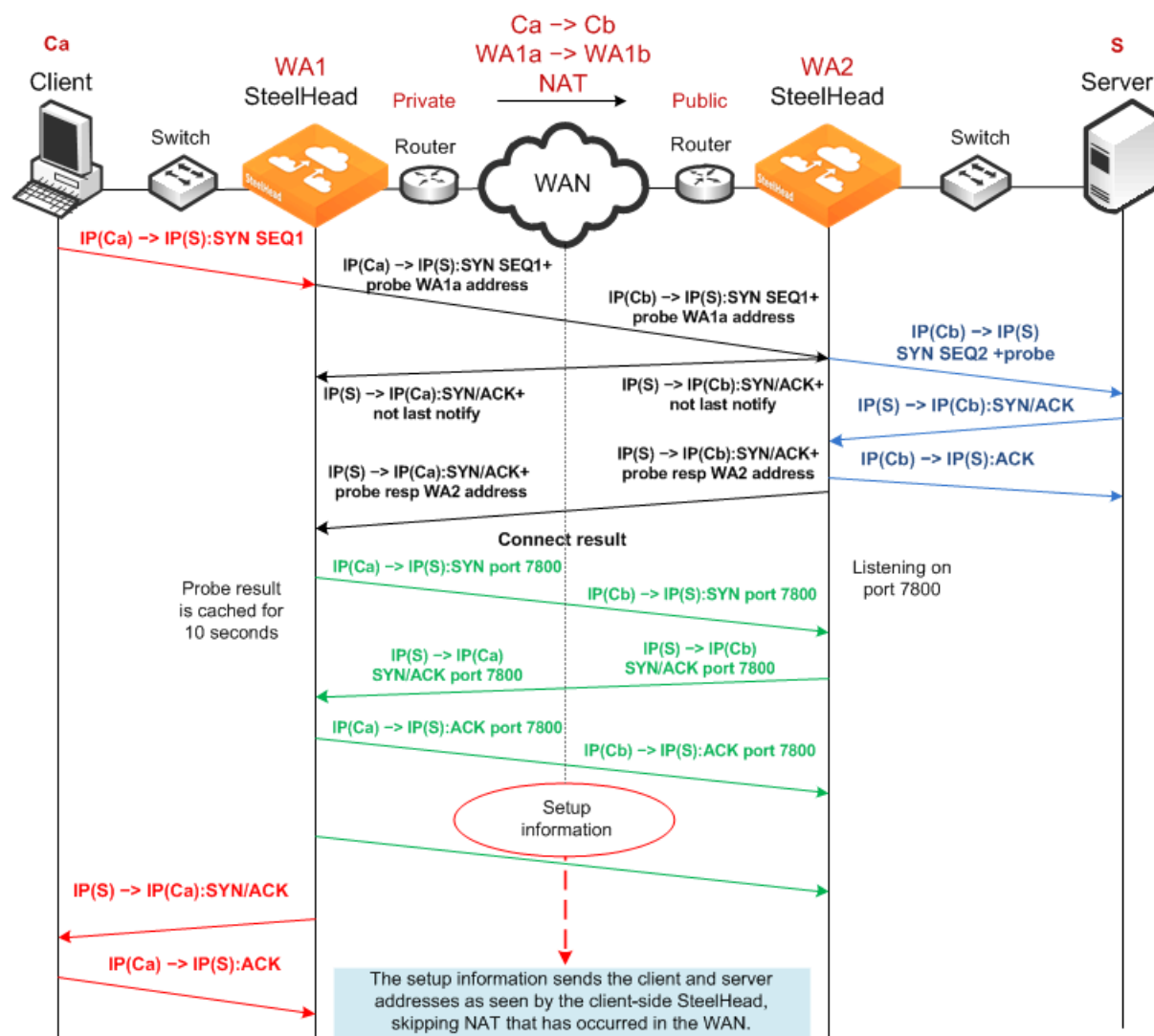
1. The packet is created coming from the initiator client (C) IP address to the destination server (S) IP address.
2. The client-side SteelHead adds a probe to the TCP connection request.
3. The server-side SteelHead sends *not last notify* immediately after receiving the probe.

4. The server-side SteelHead forwards the original connection request to the destination server using client IP:port as source address, with a sequence number 2.
5. The destination server acknowledges the client connection request.
6. The server-side SteelHead intercepts the return packet.
7. The server-side SteelHead sends a packet acknowledgment to the destination server on behalf of the client.
8. The server-side SteelHead acknowledges the client-side SteelHead, and it sends a notification with a probe response indicating that it is the last SteelHead before the server.
9. The client-side SteelHead sends a request to open port 7800 on the server-side SteelHead.
10. The server-side SteelHead acknowledges the connection request.
11. The client-side SteelHead acknowledges the connection.
12. The client-side SteelHead sends session setup information to the server-side SteelHead.
13. The server-side SteelHead connection sends an acknowledgment to the client-side SteelHead.
14. The client-side SteelHead sends the connection acknowledgment to the requesting client.
15. The client sends an acknowledgment to the destination server.
16. The client-side SteelHead discards the client acknowledgment.

Client-side source NAT using enhanced autodiscovery and full transparency

Figure 3-11 shows a client-side source NAT deployment using enhanced autodiscovery and full address transparency. In this configuration, the presence of the full transparency TCP option with autodiscovery probe is a signal to the server-side SteelHead that it can use the addresses arriving from the WAN. Because the server-side addresses are reachable from the client side, when the client-side SteelHead makes its out-of-band connection to the server-side SteelHead, the address it uses is valid and is properly translated across the WAN.

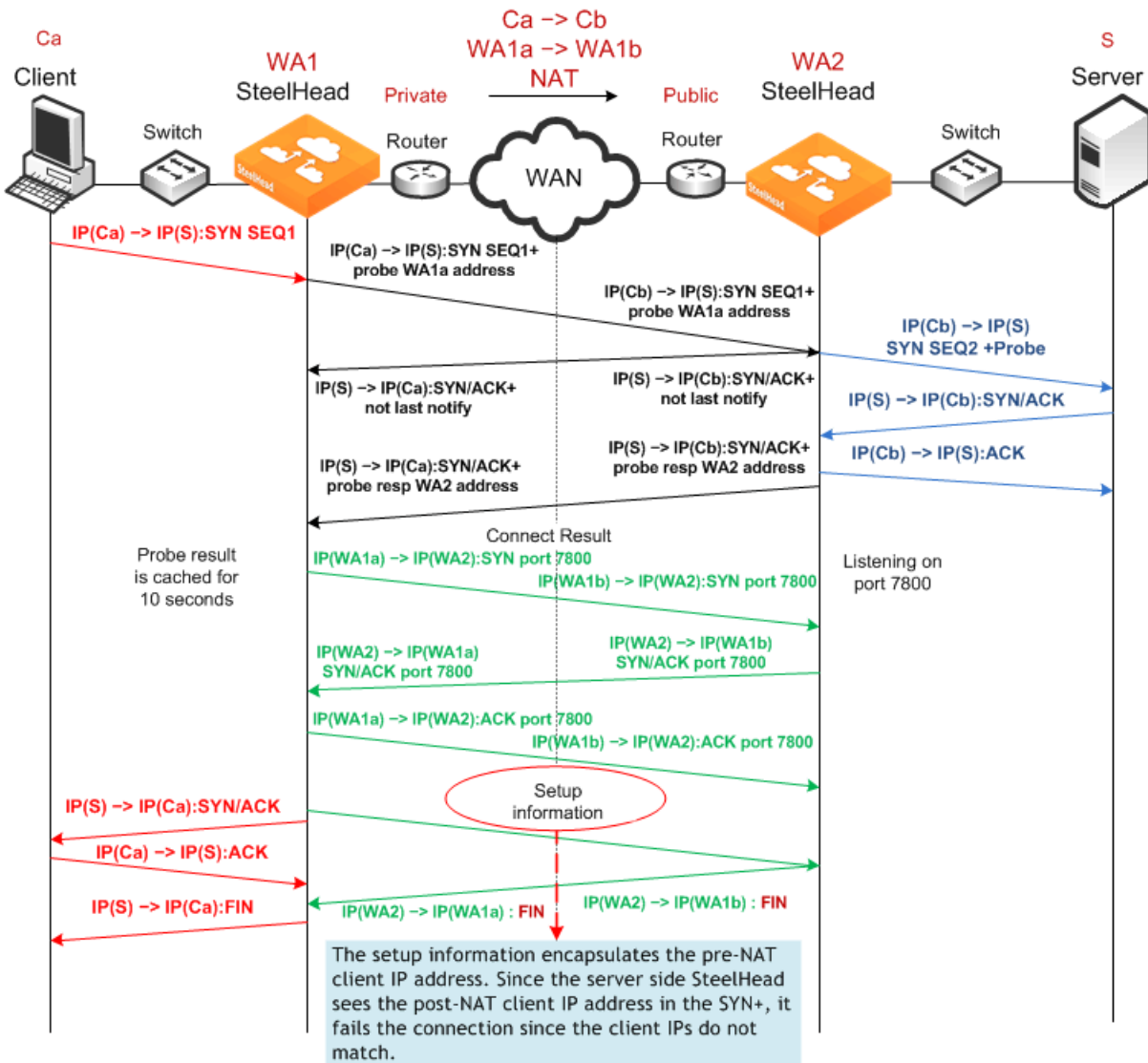
Figure 3-11. Full transparency, enhanced autodiscovery, and client-side source NAT



Failed client-side source NAT deployment using enhanced autodiscovery and correct addressing

Figure 3-12 shows a deployment in which NAT is occurring at the client location. In this example, enhanced autodiscovery with correct addressing does not work because the server-side SteelHead cannot match the client connection that it originally sees in the SYN+ with the pre-NAT (original) client IP address that is sent as part of the setup information.

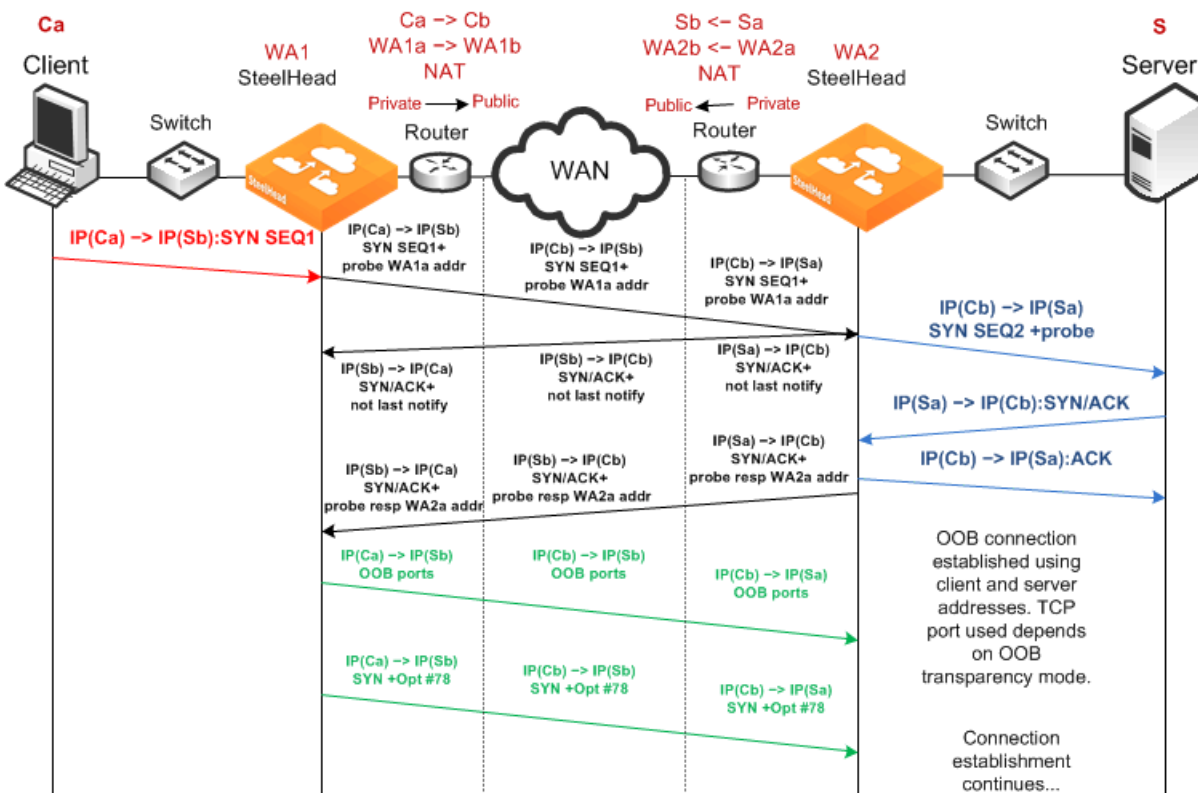
Figure 3-12. Enhanced autodiscovery, correct addressing, and client-side source NAT



Dual NAT deployment using enhanced autodiscovery and full transparency

Figure 3-13 shows NAT used at the client and the server. In this network, full transparency and some form of OOB transparency is required for successful connection establishment and optimization.

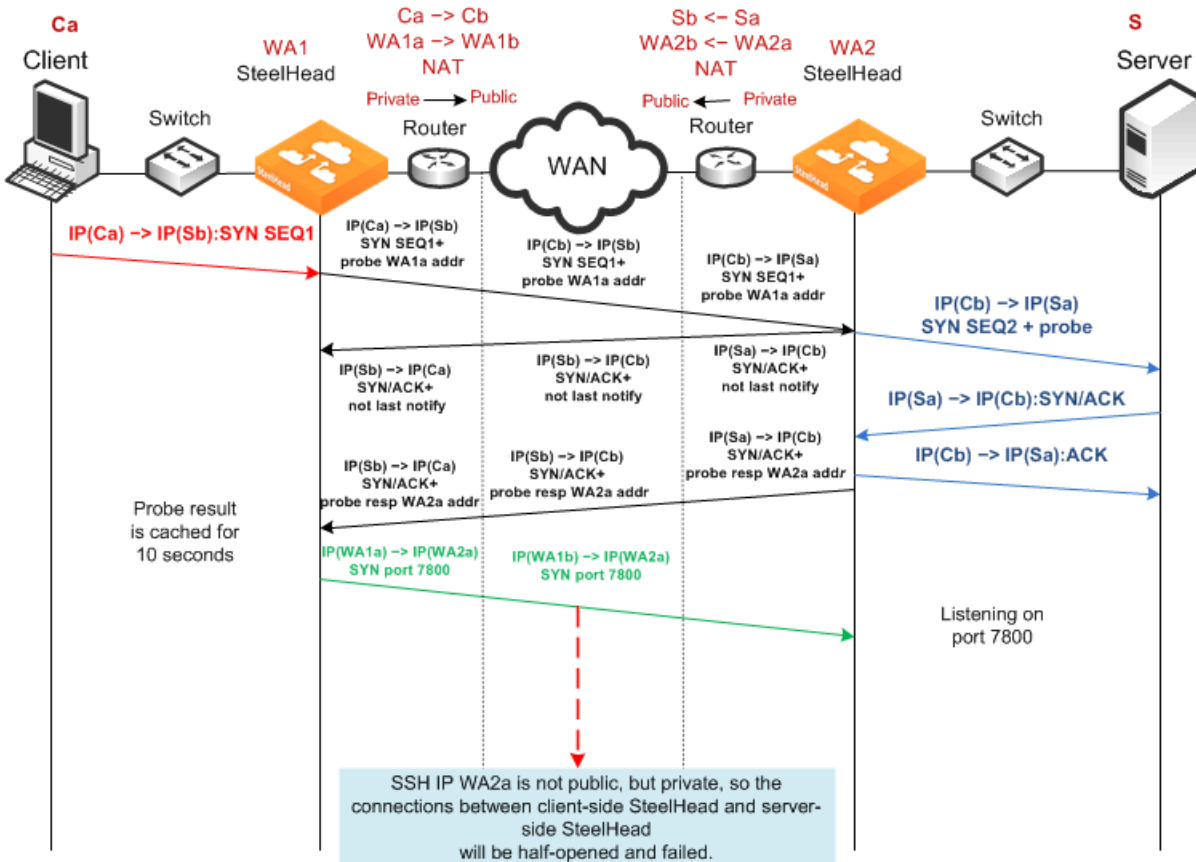
Figure 3-13. Enhanced autodiscovery, full transparency, OOB Transparency, and dual NAT



Failed dual NAT deployment using enhanced autodiscovery and correct addressing

Figure 3-14 shows a deployment in which NAT is occurring at both the client and server locations. In this example, enhanced autodiscovery with correct addressing is unlikely to work, because the probe response from the server-side SteelHead included the WA2a internal IP address for the server-side SteelHead.

Figure 3-14. Enhanced autodiscovery, correct addressing, dual NAT, resulting in half-opened connection



Out-of-band connection

This section describes transparency options for the out-of-band (OOB) connection. This section includes the following topics:

- "Overview of OOB connections and addressing modes" on page 85
- "Configuring OOB connection destination transparency" on page 85
- "Configuring OOB connection full transparency" on page 86

Overview of OOB connections and addressing modes

A SteelHead OOB connection is a TCP connection that SteelHeads establish with each other when they begin optimizing traffic to exchange capabilities and feature information and to detect failures. A SteelHead creates an OOB connection for each pair of local and remote in-path interfaces that are used when optimizing connections. OOB connections are created by the SteelHead closest to the initiating side of the optimized connection.

The addresses and ports used by OOB connections depend on the addressing mode used for the first optimized connection between SteelHeads. If the addressing mode for the first connection is correct addressing or port transparency, the OOB connection uses correct addressing. If the first connection is full transparency, the default behavior is to make the OOB connection use correct addressing, but you can alter this behavior such that the connection uses a form of network transparency.

In some environments, it might be necessary to make OOB connections use some form of network transparency: for example, the network is unable to route between the in-path IP addresses or VLANs of SteelHeads that are optimizing traffic. Two options for OOB transparency exist:

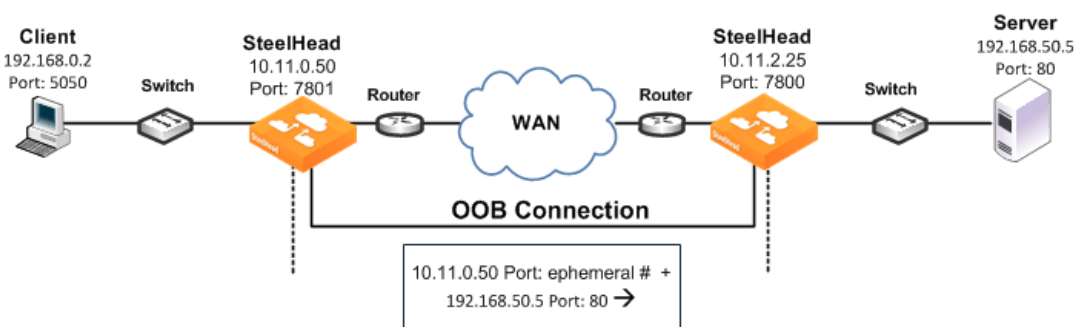
- Destination transparency
- Full transparency

The two options differ in which source IP address and TCP port are used. For details see [“Configuring OOB connection destination transparency” on page 85](#) and [“Configuring OOB connection full transparency” on page 86](#).

Configuring OOB connection destination transparency

Figure 3-15 shows TCP/IP packet headers when OOB connection destination transparency is enabled.

Figure 3-15. OOB connection destination transparency



OOB connection destination transparency uses the following values in the TCP/IP packet headers in both directions across the WAN:

- Client-side SteelHead IP address and an ephemeral port number chosen by the client-side SteelHead + server IP address and port number.
- SteelHeads use the server IP address and port number from the first optimized connection.

Use OOB connection destination transparency if the client-side SteelHead cannot establish the OOB connection to the server-side SteelHead.

Note: You must first configure WAN visibility full address transparency for OOB connection destination transparency to function correctly.

To enable OOB connection destination transparency

- Connect to the CLI on the client-side SteelHead and enter the following commands:

```
enable
configure terminal
in-path peering oobtransparency mode destination
write memory
```

Note: The changes take effect immediately. You must save your changes or they are lost upon reboot.

To disable OOB connection destination transparency

- Connect to the Riverbed CLI on the client-side SteelHead and enter the following commands:

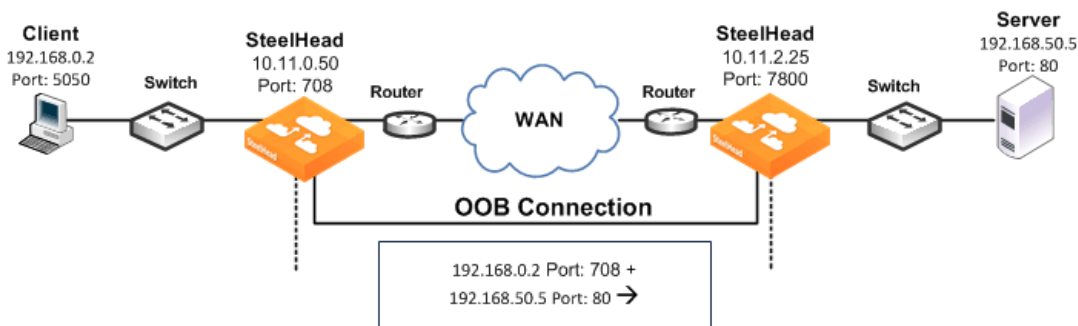
```
enable
configure terminal
in-path peering oobtransparency mode none
write memory
```

Note: The changes take effect immediately. You must save your changes or they are lost upon reboot.

Configuring OOB connection full transparency

Figure 3-16 shows TCP/IP packet headers when OOB connection full transparency is enabled.

Figure 3-16. OOB connection full transparency



OOB connection full transparency uses the following values in the TCP/IP packet headers in both directions across the WAN:

- Client IP address and client-side SteelHead predetermined port number 708 + server IP address and port number.

SteelHeads use the client IP address and the server IP address and port number from the first optimized connection.

If the client is already using port 708 to connect to the destination server, enter the **in-path peering oobtransparency port <port-number>** command to change the client-side SteelHead predetermined port number.

OoB connection full transparency supports SteelHeads deployed on trunks. Because you can configure full address transparency so that optimized traffic remains on the original VLAN, you no longer need for a SteelHead VLAN.

Use OoB connection full transparency if your network is unable to route between SteelHead in-path IP addresses or in-path VLANs or you do not want to see SteelHead IP addresses used for the OoB connection.

You must first configure WAN visibility full address transparency for OoB connection full transparency to function correctly. For details, see [“Full address transparency” on page 67](#).

To enable OoB connection full transparency

- Connect to the CLI on the client-side SteelHead and enter the following commands:

```
enable
configure terminal
in-path peering oobtransparency mode full
write memory
```

The changes take effect immediately. You must save your changes or they are lost upon reboot.

To disable OoB connection full transparency

- Connect to the CLI on the client-side SteelHead and enter the following commands:

```
enable
configure terminal
in-path peering oobtransparency mode none
write memory
```

The changes take effect immediately. You must save your changes or they are lost upon reboot.

Configuring WAN visibility modes

This section describes how to configure WAN visibility modes using an example deployment and the RiOS CLI.

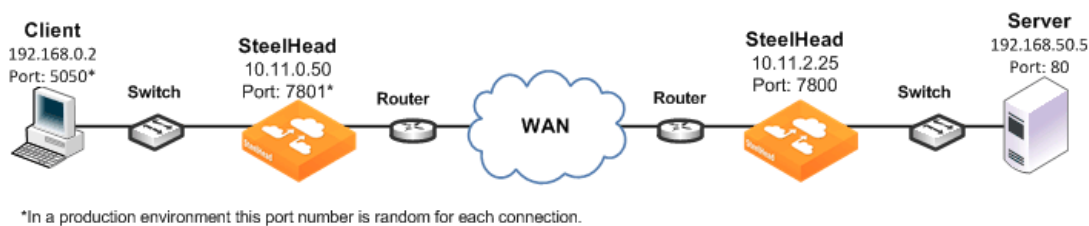
You configure WAN visibility modes by creating an in-path rule on the client-side SteelHead (where the connection is initiated). By default, the rule is placed before the default in-path rule and after the secure, interactive, and RBT-protocol rules.

For transparent addressing to function correctly, both of the SteelHeads must have RiOS 5.0.x or later installed. If one SteelHead does not support transparent addressing (that is, it has RiOS 4.1 or earlier installed), the SteelHead attempting to optimize a connection in one of the transparent addressing modes automatically reverts to correct addressing mode, and optimization continues.

By default, SteelHeads use correct addressing (for all RiOS versions).

Figure 3-17 shows the IP addresses and ports used in the example deployments.

Figure 3-17. Example deployment for configuring WAN visibility modes



The following table summarizes the port transparency commands.

Action	CLI command
Enable port transparency for a specific server.	in-path rule auto-discover wan-visibility port dstaddr 192.168.50.1/32 dstport 80
Enable full address transparency for a specific group of servers, and port transparency for servers not in the group.	in-path rule auto-discover wan-visibility full dstaddr 192.168.0.0/24 in-path rule auto-discover wan-visibility port
Important: In this example, the first in-path rule must precede the second in-path rule in the rule list. To specify the placement of a rule in the list, use the rulenum keyword and value. For details, see the <i>Riverbed Command-Line Interface Reference Manual</i> .	
Disable port transparency.	Delete the in-path rule that enables it. For information about deleting in-path rules, see the <i>Riverbed Command-Line Interface Reference Manual</i> .

The following table summarizes the full address transparency commands.

Action	CLI command
Enable full address transparency globally.	in-path rule auto-discover wan-visibility full
Enable full address transparency for servers in a specific IP address range.	in-path rule auto-discover wan-visibility full dstaddr 192.168.0.0/16
Enable full address transparency for a specific server.	in-path rule auto-discover wan-visibility full dstaddr 192.168.50.1/32

Action	CLI command
Enable full address transparency for a specific group of servers, and enable port transparency for servers not in the group.	in-path rule auto-discover wan-visibility full dstaddr 192.168.0.0/24 in-path rule auto-discover wan-visibility port Important: In this example, the first in-path rule must precede the second in-path rule in the rule list. To specify the placement of a rule in the list, use the rulenum keyword and value. For details, see the <i>Riverbed Command-Line Interface Reference Manual</i> .
Disable full address transparency	Delete the in-path rule that enables it. For information about deleting in-path rules, see the <i>Riverbed Command-Line Interface Reference Manual</i> .

Topology

This chapter describes how to configure the topology for a SteelHead with RiOS 9.0 and later. This chapter includes the following sections:

- [“Introduction to the topology concept” on page 91](#)
- [“Defining a network” on page 91](#)
- [“Defining a site” on page 93](#)

This chapter requires you be familiar with [“QoS configuration and integration” on page 111](#), [“Path Selection” on page 169](#), and the secure transport information in [“Overview of secure transport” on page 425](#) and the *SteelCentral Controller for SteelHead Deployment Guide* Version 9.0 or later.

For more information about configuring topology for QoS, see [“Configuring topology” on page 156](#).

Introduction to the topology concept

A topology consists of networks and sites. Path selection, QoS, and secure transport can use a common topology that you configure once, and then the topology simplifies configuration by providing a building block that you can reuse.

The information contained in the topology definition is used by RiOS to perform certain functions like path monitoring or bandwidth calculation for QoS without requiring further input. The topology provides the SteelHead with:

- which networks it is connected to.
- which of its interfaces are used to connect to these networks.
- how much bandwidth is available.
- which networks to use to connect to a SteelHead at a remote sites.
- how much bandwidth is available to send traffic to a remote site.

Defining a network

Within a topology, *network* is a label for a connection to an available WAN. In other words, it is the WAN cloud that sites use to communicate with each other. The network describes the type of transport available for traffic. Name your networks with descriptions such as Primary MPLS or Internet.

The default network is My WAN. My WAN is a private network and associated with the in-path interfaces and the primary interface of the local site. We recommend that you change the name to a more descriptive one. Because an uplink must connect to a network, we recommend that you add a management network for managing the SteelHead and connect the primary interface to the My WAN network.

If you use the primary interface to join a Microsoft Windows domain, you must create a network accordingly.

The configuration parameters of a network are as follows:

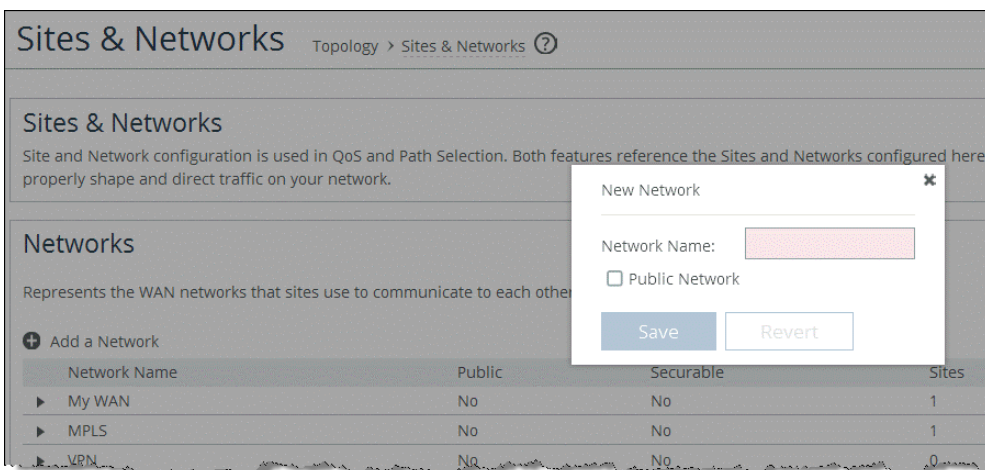
- **Name** - The name of your network.
- **Public network** - Tells the SteelHead if the network is a public or a private network. If you define your network as a public network by selecting the check box, the SteelHead assumes that the traffic sent to this network traverses a device that performs NAT. The public network option only takes effect if you also configure secure transport, which is available when configuring the SteelHead using the SCC.

For more information on secure transport, see [“Overview of secure transport” on page 425](#) and the *SteelCentral Controller for SteelHead Deployment Guide*.

You can only set the Secure option when you configure the SteelHead using the SCC.

To configure a network, choose Networking > Topology: Sites & Networks, and select Add a Network (Figure 4-1).

Figure 4-1. Add a network



Defining a site

Within a topology the site defines the configuration parameters, which are necessary to connect the site to a network. You can also view a site as a remote office, branch office, or data center. You must define a site to use the QoS, path selection, and secure transport features. Two sites are configured by default, which have special characteristics:

- [“Configuring the local site” on page 94](#)
- [“Configuring the default site” on page 96](#)

Note: You configure Secure Transport using the SCC 9.0 or later.

The configuration parameters of a site are as follows:

- **Site name** - A descriptive name for the site.
- **Subnets** - IP addresses of the subnets existing within the site.
- **SteelHead peers** - The IP addresses of the remote SteelHeads that are reachable from the SteelHead you are configuring. The IP addresses you enter in this field are probed to determine if a path to this IP address exists and if the address is reachable. If you want to configure GRE encapsulation or secure transport, this IP address must be the in-path interface IP address of a SteelHead. If not, this IP address can be a server or routers interface.
- **QoS Profiles Inbound and Outbound** - A set of classes and rules to be used for QoS.

For more information about QoS profiles, [“QoS profiles” on page 124](#).

- **Uplinks** - Connects the site to a network. A site can have a single or multiple uplinks to the same network and can connect to multiple networks. You can use multiple uplinks to the same network for redundancy. You must specify, per uplink, the bandwidth available for uploading and downloading data. The values of the configured bandwidth are used by RiOS to calculate the bandwidth available for traffic for inbound and outbound QoS configurations. In combination with the bandwidth configuration of the local sites uplink, the SteelHead can calculate the oversubscription factor in case the sum of the bandwidths of the remote sites to a network is greater than the bandwidth of the local site to the same network.

RiOS 9.0 and later provide a simplified configuration. The oversubscription factor is automatically computed and applied to sites when they are initially configured or when a new site is added.

Consider the following:

- The bandwidth configured for the uplinks in the sites is also used to calculate the bandwidth for class tree for QoS.
- Uplinks used by the local and the default site are special. See [“Configuring the local site” on page 94](#) and [“Configuring the default site” on page 96](#) for more information.
- If you do not specify a bandwidth for an uplink, the link speed of the in-path interface is used by default.

To configure a site, choose Networking > Topology: Sites & Networks, and select Add a Site (Figure 4-4).

Figure 4-2. Add a site

Create a New Site

Basic Information

Site name

Network Information

Subnets
Subnets define how the SteelHead identifies this site. Separate with comma ","

SteelHead Peers
Peers are used for path monitoring and GRE Tunneling. Separate with comma ","


QoS Profiles

Inbound QoS Profile

Outbound QoS Profile

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

 Add New Uplink

Configuring the local site

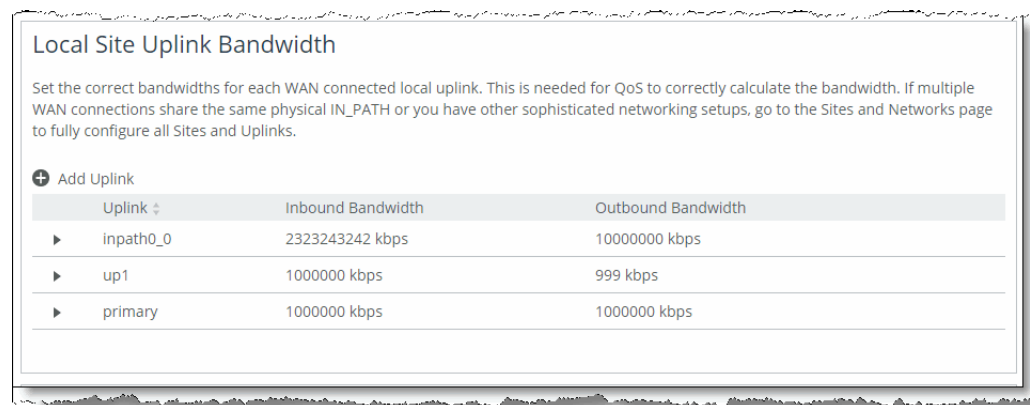
The local site is the physical location of the SteelHead you are connected to and actively configuring. Local sites have special characteristics. In contrast to a configuration of a SteelHead using the SCC, configuring a SteelHead using the SteelHead Management Console is configuring a single appliance. This means that the SteelHead to configure is the point of view to the topology. For example, you connect to the data center and configure on the data center SteelHead all the remote sites. Then you connect to a remote site, which now becomes the local site, and the data center becomes a remote site. Next, you need to configure gateways for your new local site, which you did not have to do when it was a remote site.

Note: Keep in mind that the point of view to the topology changes when connecting to the next SteelHead to configure.

To configure the bandwidth of the uplink for QoS for the local site, you can either use the Local Site Uplink Bandwidth pane in the Networking > Network Services: Quality of Service page, or you can edit the local site on the Networking > Topology: Sites & Networks page. The bandwidth configuration of the uplinks on either page will be mirrored on the other. Note that you can configure the gateway IP address only on the Networking > Topology: Sites & Networks page.

To configure the bandwidth of the local uplink for QoS, on the Networking > Network Services: Quality of Service page, go to the Local Site Uplink Bandwidth pane on the page and expand the uplink you want to configure.

Figure 4-3. Local Site Uplink Bandwidth pane on the Quality of Service page



Then enter the values for Bandwidth Down (inbound bandwidth) and Bandwidth Up (outbound bandwidth). Then click Save.

To configure the bandwidth of the local uplink for QoS, on the Networking > Network Services: Sites & Networks page go to Networking - Sites & Networks and edit the Local (Local) site. In the local site, you configure the uplink bandwidth for uploading and downloading data and with the IP address of the gateway to the network the uplink connects to. If you do not have the gateway configured, the default gateway of the in-path interface is used.

In some deployment scenarios, the default gateway of the in-path interface points to the LAN side of the SteelHead. When you configure path selection, we recommend that the gateway to a network points to the WAN side of the SteelHead to avoid packet ricochet.

You also configure the probe settings for path selection in the uplink of the local site.

To configure uplinks for the local site, choose **Networking > Topology: Sites & Networks**, select **Edit a Site** from an existing local site, and scroll down to **Uplinks** (Figure 4-4).

Figure 4-4. Edit uplinks on a local site

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

+ Add New Uplink

inpath0_0 ⓘ

Uplink Name:

Network:

Gateway IP:

Inpath Interface:

☐ GRE Tunneling ⓘ

Bandwidth Up: kbps

Bandwidth Down: kbps

▼ Probe Settings

Outbound DSCP:

Timeout: second(s)

Threshold:

primary ⓘ

Uplink Name:

Network:

Gateway IP:

Inpath Interface:

☐ GRE Tunneling ⓘ

Bandwidth Up: kbps

Bandwidth Down: kbps

▶ Probe Settings

Save Revert

The primary interface of the local site also connects to the My WAN network by default. If you are using the primary interface for the SteelHead management, or to integrate with a Windows domain, you might need to configure a separate network to connect the primary interface.

Configuring the default site

The default site catches traffic that is not destined to a configured site, such as traffic to the internet or traffic that does not match any of the subnets configured within any of the site definitions.

The default site does not have an uplink configured by default. Without a configured uplink, the interface physical bandwidth is used for traffic the default site catches.

If you are using QoS on the SteelHead, we strongly recommend that you configure an uplink for the default site and assign an appropriate bandwidth to it.

To configure uplinks for the default site, choose **Networking > Topology: Sites & Networks**, select **Edit a Site**, and select **Add New Uplink**.

Application Definitions

Application definitions is a grouping of related configuration pages in RiOS 9.0. This chapter describes how to define applications and application groups. Additionally it describes how to create host and port labels. To see the application definitions menu items, select Networking > App Definitions.

Important: If you are using a release previous to RiOS 9.0, the features described in the chapter are not applicable. For information about applications before RiOS 9.0, see earlier versions of the *SteelHead Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

This chapter includes the following sections:

- “Applications” on page 97
- “Application Flow Engine” on page 102
- “Creating host labels” on page 105
- “Creating port labels” on page 108
- “Creating domain labels” on page 109

This chapter requires you be familiar with “QoS configuration and integration” on page 111 and “Path Selection” on page 169.

For more information about configuring applications for QoS, see “Configuring applications” on page 147.

Applications

This section describes how to define applications and work with application groups. Additionally, it describes the concept of application properties. This section includes the following topics:

- “Defining an application” on page 98
- “Application properties” on page 100

To simplify the SteelHead configuration, the definition of an application is a separate task in RiOS 9.0 and later. A separate application definition allows for the configuration of multiple rules, using the same application without having to repeat the application definition for each rule.

Note: In RiOS versions prior to 9.0, the configuration of an application was tightly coupled with QoS rules.

Application definitions also enable you to group applications, so that you can configure and reuse a single rule for multiple applications. Using an application group in a rule can reduce the number of rules significantly.

For more information about QoS rules, see [“QoS rules” on page 116](#). For more information about grouping applications, see [“Application properties” on page 100](#).

Defining an application

Defining an application means that you group together a set of criteria to match certain traffic. After you define the criteria, you can use an application to configure QoS and path selection rules.

Note: Custom-defined applications are not displayed in the Application Statistics or Application Visibility reports.

To define an application

1. Choose Networking > App Definitions: Applications.

The custom applications group is empty until you add application groups. For more information about application groups, see [“Application properties” on page 100](#).

2. Select Custom Applications from the drop-down menu and select Add ([Figure 5-1](#)).

Figure 5-1. Add an application

New Application

Name:

Description:

Traffic Characteristics:

Local Subnet: Port:

Remote Subnet: Port:

Transport Layer Protocol:

Application Layer Protocol:

VLAN Tag ID:

Outbound DSCP:

Traffic Type:

Application Properties:

Application Group:

Category:

Business Criticality:

3. Specify a name and a description for the new application.

Figure 5-1 shows the example new application, MyApp.

4. Specify its traffic characteristics (**Figure 5-2**).

You can use host labels instead of local and remote subnets, and you can use port labels instead of TCP/UDP port numbers.

For more information about traffic characteristics, see the *SteelHead User Guide*.

In addition to criteria matching on the IP-header based characteristics or the VLAN ID, you can use the Riverbed Application Flow Engine (AFE) to let RiOS automatically detect the application by typing the first letters of the application into the Application Layer Protocol field. For example, if you want to create specific criteria to identify Facebook traffic, type in the first three or four letters and a drop down menu opens, which lets you choose from Facebook applications.

Figure 5-2. Facebook application

Traffic Characteristics:

Local Subnet: 0.0.0.0/0 Port:

Remote Subnet: 0.0.0.0/0 Port:

Transport Layer Protocol: Any ▼

Application Layer Protocol: Face ▼

VLAN Tag ID:

Outbound DSCP:

Traffic Type:

Application Properties:

Application Group: Custom

Category: Collapsed

Business Criticality: Low

Facebook applications in dropdown:

- Facebook
Facebook is a social networking service.
- FaceTime
Video conferencing between supported Apple mobile devices and Macintosh computers
- Facebook-Event
Creating/editing Facebook events
- Facebook-Messages
- Facebook E-mail and Instant Messaging
- Facebook-Post
Interactions with Facebook walls
- Facebook-Search
Searching within Facebook's website
- Facebook-Video-Chat
Facebook Video Chat
- Facebook-Video

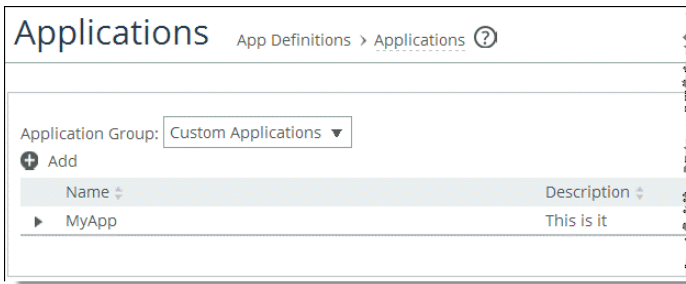
Buttons: Save, Reveal

You can also create a QoS or path selection rule using the AFE directly in the rule. For more information about AFE, see [“Overview of the Application Flow Engine” on page 102](#).

5. Click **Save** to save the application to the list.

Figure 5-3 shows MyApp in the custom application list.

Figure 5-3. New application



Application properties

Application properties provide a way to group applications based on the following criteria:

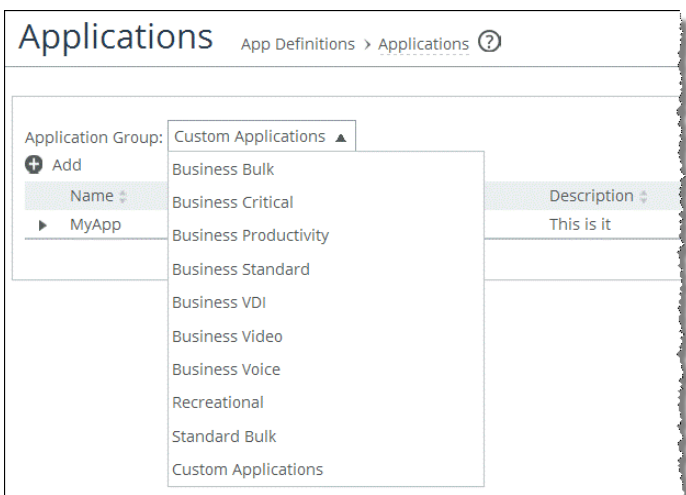
- Application group
- Category
- Business criticality

The grouping of applications enables you to create path selection or QoS rules based on your users intent instead of single applications. For example, to configure all business critical applications for a QoS class, you can configure a single rule in a QoS profile based on the application group *Business Critical*. Without the application group you need to configure as many rules as there are business critical applications.

Note: Currently, the application properties *Category* and *Business Criticality* are not supported to configure path selection or QoS rules, nor can you create additional application groups.

Select the drop-down menu next to the application group on the Networking > App Definitions: Applications page to display all available groups (Figure 5-4).

Figure 5-4. Application groups



Select an application group in the drop-down menu to view the application list associated with that group.

You can also add a new application to a group by selecting the group from the Application Group drop-down menu in the Application Properties section of the configuration page (Figure 5-5).

Figure 5-5. Assigning the new application a group

The screenshot shows the 'New Application' configuration page. The 'Name' field contains 'NewApp' and the 'Description' field contains 'Brandnew'. The 'Traffic Characteristics' section includes fields for 'Local Subnet' (0.0.0.0/0), 'Remote Subnet' (0.0.0.0/0), 'Transport Layer Protocol' (Any), 'Application Layer Protocol' (Any), 'VLAN Tag ID' (All), 'Outbound DSCP' (Any), and 'Traffic Type' (Any). The 'Application Properties' section shows the 'Application Group' dropdown menu open, displaying a list of categories: Business Bulk, Business Critical, Business Productivity, Business Standard, Business VDI, Business Video, Business Voice, Recreational, Standard Bulk, and Custom Applications. A 'Save' button is visible at the bottom left of the form.

You can also change the membership of an application in an application group by editing the application.

For example, the application Facebook is preconfigured in the Recreational application group, but your company is using Facebook as a business productivity tool.

To move the Facebook application into the Business Productivity group, choose Networking > App Definitions > Applications and select the Recreational application group. Next, select Facebook and change the application group (Figure 5-6).

Figure 5-6. Change application group membership

Facebook is a social networking service.

Name: Facebook

Description: Facebook is a social networking service.

Traffic Characteristics:

Local Subnet: 0.0.0.0/0 Port:

Remote Subnet: 0.0.0.0/0 Port:

Transport Layer Protocol: Any

Application Layer Protocol: Facebook

VLAN Tag ID: All

Outbound DSCP: Any

Traffic Type: Any

Application Properties:

Application Group: Business Productivity

Category: Social Networking

Business Criticality: Low Criticality

Apply Revert

Application Flow Engine

This section includes the following topics:

- [“Overview of the Application Flow Engine” on page 102](#)
- [“AFE and Microsoft Lync 2010 and 2013” on page 104](#)

Overview of the Application Flow Engine

The Riverbed Application Flow Engine (AFE), QoS can identify applications accurately using a variety of technologies. It is a powerful engine that you can use to automatically detect applications on the network. You can also use AFE in QoS and path selection rules, in application definitions, and the application visibility reporting.

Examples of technologies the AFE uses are:

- Application signature/Pattern matching
 - Match well-known byte patterns in a flow and across multiple packets
 - Match the URL within HTTP
- Protocol dissection/Protocol awareness
 - Layer-7 fluency to following the conversation

- Behavioral classification/Heuristic classification
 - Analyses packet size, packet inter-arrival time, packet rate, data rate, and calculates entropy (randomness) to detect an application (for example, Skype)
- Dynamic decode
 - Undoing obfuscation techniques (for example, base64)

If the AFE fails to identify the application based on the above technologies, it falls back to a TCP/UDP port-based classification.

To view a completed global application list, see the *SteelHead User Guide*.

In addition to the AFE supporting many well-known applications, you can add rules to identify custom applications. For example, you can identify a new HTTP application based on specific domain name or relative path (Figure 5-7).

Figure 5-7. Rules to Identify applications

The screenshot shows the 'New Application' configuration window. The 'Name' field is 'HttpApp'. The 'Description' field is 'URL based'. Under the 'Traffic Characteristics' section, the following fields are visible:

- Local Subnet: 0.0.0.0/0
- Remote Subnet: 0.0.0.0/0
- Transport Layer Protocol: Any
- Application Layer Protocol: HTTP
- Domain Name: intranet.net
- Relative Path: contactlist/customers
- VLAN Tag ID: All
- Outbound DSCP: Any
- Traffic Type: Any

With RiOS 8.6 or later, you can use the AFE to classify unoptimized SSL traffic based on the TLS/SSL server common name in the server certificate. To do this, add an application, type SSL in the Application Layer Protocol text box, and type the common name of the server certificate (for example, `www.yoursite.com/*`) in the Common Name field. To make the configuration easier, you can use wildcards in the name ([Figure 5-8](#)).

Figure 5-8. SSL common name matching configuration

The screenshot shows the 'New Application' configuration window. The 'Name' field is 'SSLApp' and the 'Description' is 'Common Name based'. Under 'Traffic Characteristics', the 'Local Subnet' and 'Remote Subnet' are both '0.0.0.0/0'. The 'Transport Layer Protocol' is set to 'Any'. The 'Application Layer Protocol' is 'SSL'. The 'Common Name' is 'www.yoursite.com/*'. The 'VLAN Tag ID' is 'All'. The 'Outbound DSCP' is 'Any'. The 'Traffic Type' is 'Any'. The 'Application Properties' section is partially visible at the bottom.

You cannot classify SSL optimized traffic using the common name control. For optimized or decrypted SSL traffic, the AFE uses the same techniques as nonencrypted traffic to classify the traffic.

For information about defining an application using the AFE, see [“Applications” on page 97](#). For more information about SSL, see the *SteelHead Deployment Guide - Protocols*.

AFE and Microsoft Lync 2010 and 2013

RiOS 9.1 and later enhance support for Microsoft Lync. Lync is a multiple-feature communication suite that carries traffic over an extensive selection of protocols. The AFE classification of Lync traffic covers the majority of traffic generated between Lync clients and Lync servers.

The following table summarizes the types of traffic Lync generates and the classification the AFE provides for them:

Workload	Classified as
Client login	LYNC, LYNCCTRL
Chat message	LYNC, LYNCCTRL
File transfer	LYNC, LYNCSHRE
Group voice chat	LYNC, LYNCMDIA
Video call	LYNC, LYNCMDIA
Application screen sharing	LYNC, LYNCSHRE
Desktop sharing	LYNC, LYNCSHRE
Voice call	LYNC, LYNCMDIA
Presentation sharing	SSL, SIP
White-board session	SSL, SIP

Note: A Lync server uses the default SIP port of TCP 5061. You can use this information to build a custom rule to classify Lync SIP traffic.

Creating host labels

Use host labels to group multiple subnets or hostnames into one label. You create host labels on the Networking > App Definitions > Host Labels page (Figure 5-9).

Figure 5-9. Host Labels page

Host Labels App Definitions > Host Labels ?

Summary of Hostname Resolution

- 0 Unique Hostnames
- 0 Checking DNS
- 0 Unresolvable

Hostnames are automatically resolved once every day.

[Resolve Hostnames](#)

☐ Show resolved IPs for the hostnames in the table below

[Add a New Host Label](#) [Remove Selected](#)

Name:

Hostnames/Subnets:

[Add New Host Label](#)

Label	Hostnames
-------	-----------

The host label option covers a group of server IP addresses, subnets, and fully qualified domain name (FQDN) entries. FQDNs require that you configure a DNS server on the SteelHead so that it can resolve the host label of the FQDN.

After you create the host label, you can use the host label:

- to define an application instead of using an IP address in the local and remote subnet fields of the New Application box (Figure 5-10).
- in the in-path rule list (Figure 5-11).

An application or an in-path rule defined using a host label can match traffic coming from multiple hostnames or IP addresses with a single rule definition.

For information about how to define an application, see [“Defining an application” on page 98](#).

Figure 5-10. Adding the host label to an application

New Application

Name: MyExchange

Description: Uses a host label containing my Exchange servers

Traffic Characteristics:

Local Subnet: ExchangeServers Port:

Remote Subnet: 0.0.0.0/0 Port:

Transport Layer Protocol: Any

Application Layer Protocol: Any

VLAN Tag ID: All

Outbound DSCP: Any

Traffic Type: Any

Application Properties:

Application Group: Custom Applications

Category: Collaboration

Business Criticality: Lowest Criticality

Save Revert

Figure 5-11. Adding the host label to the in-path rule

In-Path Rules Network Services > In-Path Rules ?

▼ Add a New In-Path Rule ✕ Remove Selected Rules ⇅ Move Selected Rules...

Type: Auto Discover

Source: Subnet: All IP (IPv4 + IPv6)

Destination: Subnet: Host Label ExchangeServers

Port: All Ports

Domain Label: n/a

VLAN Tag ID: all

Use the following guidelines when creating a host label:

- If hostnames in a host label are not resolved by DNS, traffic does not match.
- You cannot use host labels to define sites.
- Host labels only support IPv4.
- All configured hostnames are automatically resolved by DNS every 24 hours by default. You can change the interval with the following CLI command:

```
host label refresh intvl <minutes>
```

- Click **Resolve Hostnames** to immediately resolve hostnames through DNS.
- Any changes in IP addresses that a hostname resolves are relayed to the rule using the host label.
- You can configure a maximum of 100 unique hostnames (across all host labels combined).
- There is a maximum of 64 subnets and hostnames per host label.
- There is no limit on the number of host labels that you can configure.
- Host labels are not supported on the source subnet field of an in-path rule.
- We recommend that you use caution when using a host label in an in-path rule when you have cloud acceleration configured if the IP addresses or FQDNs in the host labels coincide with the cloud acceleration IP addresses.

The following table shows host label usage supportability with various deployment options:

SteelHead deployment options	Host label in-path rule destination
Client-side and server-side inline	Yes. Host-label rule only configured on the client-side SteelHead.
Client-side virtual in-path	Yes
Server-side out-of-path	Yes, when using a fixed-target rule on the client-side SteelHead with a host label.
Connection forwarding	Client-side: Yes Server-side: N/A
Agent-intercept mode (with discovery agent)	Client-side: Yes Server-side: N/A
SteelHead Mobile	No

For more information about host labels, see the *SteelHead User Guide*.

Creating port labels

A port label is a name given to a set of TCP/UDP port numbers. You use port labels when you configure in-path rules, peering rules, or applications.

RiOS uses predefined port labels and in-path rules to pass-through certain traffic types, like encrypted or already optimized traffic.

Note: If you are optimizing SSL encrypted HTTP traffic make sure to remove port 443 from the *Secure* port label.

To edit or create a port label, choose Networking > App Definitions: Port Labels (Figure 5-12).

Figure 5-12. Port Labels page

Label	Ports
Interactive	7, 23, 37, 107, 179, 513-514, 1494, 1718-1720, 2000-2003, 2427, 2598, 2727, 3389, 5060, 5631, 5900-5903
RBT-Proto	7744, 7800-7801, 7810, 7820, 7850, 7860, 7870
Secure	22, 49, 88, 261, 322, 443, 448, 465, 563, 585, 614, 636, 684, 695, 902, 989-990, 992-995, 1701, 1723, 2252, 3471, 3496, 3509, 3529, 3539, 3660-3661, 3713, 3747, 3864, 3885, 3896-3897, 3995, 4031, 5007, 5061, 57
SteelFusion	7950-7954, 7970

For more information about port labels, see the *SteelHead User Guide*.

Creating domain labels

A domain label is an augmentation for further refining an in-path rule by using the domain name field. For example, rather than specifying multiple IP addresses or FQDNs in the destination field, you can use *.ABC.com to intercept only destination hosts matching these criteria. A domain label field can also lessen the number of hosts to optimize, or exclude from optimization, in case the field does not match the domain label and the other source or destination fields.

Note: We recommend that you read the knowledge base article at <https://supportkb.riverbed.com/support/index?page=content&id=S28159> before implementing domain labels.

Note: Domain labels are not supported on SteelHead SaaS.

The domain label field does not replace the destination IP address parameter. The in-path rule needs to match the destination IP address or host label field, and next match the domain label entry. If the connection matches the source and destination list, the client-side SteelHead begins to process subsequent data packets looking to match the domain in the host field if HTTP or the SNI field in HTTPS.

With a domain label you can selectively perform:

- dual-ended optimization of individual services on an on-premises server running multiple applications.
- web-proxy optimization of individual domain of HTTP/HTTPS web proxy traffic in single-ended deployment.

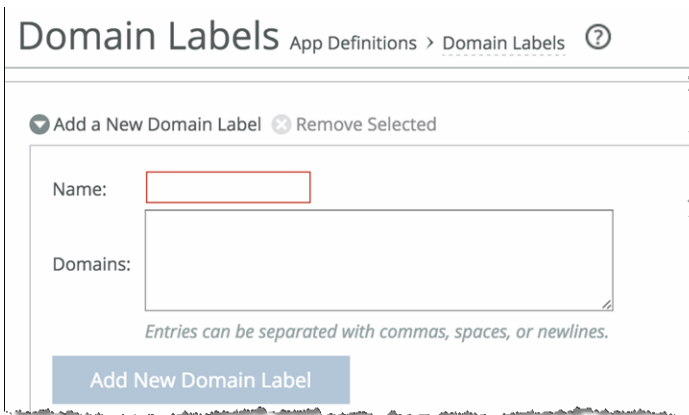
Use the following guidelines when creating a domain label:

- We recommend that you configure a domain label as the last rule in the listing.
- For optimized traffic, both the client-side and server-side SteelHeads must be running RiOS 9.2 or later.
- Domain labels are not compatible with packet mode optimization.

- Domain label names are alphanumeric.
- Domain label names cannot be more than 64 characters long.
- You can create a maximum of 63 domain labels.
- The default destination ports are 80 and 443 unless manually specified in the in-path rule.
- Domain labels are compatible with IPv4 only.
- Place an in-path rule using a domain label and destination address set to All at the end of the rule list.
- Domain labels are not supported on the source subnet field.
- The web proxy feature is compatible with domain label rule lists. For more information about web proxy, see *SteelCentral Controller for SteelHead Deployment Guide*.
- Domain labels allow you to specify an internet domain with wildcards to define a wider group, such as *.ABC.com.
- By default, domain labels support only HTTP-based and HTTPS-based traffic (ports 80 and 443). For HTTP, the domain label feature looks at the Host field in the GET request to find a match with the configured domains. For HTTPS, the domain label looks at the SNI field of the Client Hello.
- SteelHead SaaS optimization is bypassed in the presence of a rule defined with a domain label and destination set to All.
- SteelHead SaaS optimization and domain labels in the same rule is not supported.

To edit or create a domain label, choose Networking > App Definitions: Domain Labels ([Figure 5-13](#)).

Figure 5-13. Domain Labels page



The screenshot shows the 'Domain Labels' page under 'App Definitions > Domain Labels'. At the top, there are two buttons: 'Add a New Domain Label' (with a plus icon) and 'Remove Selected' (with a minus icon). Below these is a form with two main input fields: 'Name:' followed by a single-line text box, and 'Domains:' followed by a larger multi-line text box. Below the 'Domains' text box, a note states: 'Entries can be separated with commas, spaces, or newlines.' At the bottom of the form is a blue button labeled 'Add New Domain Label'.

Note: Domain labels require both SteelHeads to be running RiOS 9.2 and later.

QoS configuration and integration

This chapter provides an overview of the principles of Riverbed Quality of Service (QoS). Additionally, it shows how to configure Riverbed QoS and how to integrate SteelHeads into existing QoS architectures.

This chapter includes the following sections:

- “Overview of Riverbed QoS” on page 112
- “QoS concepts” on page 114
- “Configuring QoS” on page 128
- “Inbound QoS” on page 130
- “LAN bypass” on page 132
- “QoS for IPv6” on page 132
- “QoS in virtual in-path and out-of-path deployments” on page 133
- “QoS in multiple SteelHead deployments” on page 133
- “QoS and multiple WAN interfaces” on page 134
- “Integrating SteelHeads into existing QoS architectures” on page 134
- “QoS enforcement best practices” on page 139
- “Upgrading to RiOS 9.0” on page 139
- “Guidelines for the maximum number of QoS classes, sites, and rules” on page 140

Important: A QoS configuration generally requires the configuration of many SteelHeads. To avoid repetitive configuration steps on single SteelHeads, we strongly recommend that you use the SCC 9.0 or later to configure QoS on your SteelHeads. SCC enables you to configure one time and to send the configuration out to multiple SteelHeads instead of connecting to a SteelHead, performing the configuration, and repeating the same configuration for all SteelHeads in the network. For details, see the *SteelCentral Controller for SteelHead User Guide*.

Important: If you are using a release previous to RiOS 9.0, the features described in the chapter are not applicable. For information about applications before RiOS 9.0, see earlier versions of the *SteelHead Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

For more information about Riverbed QoS, see the *SteelHead User Guide*. For configuration examples, see [“QoS Configuration Examples” on page 143](#).

This chapter requires you be familiar with [“Topology” on page 91](#) and [“Application Definitions” on page 97](#).

Overview of Riverbed QoS

Riverbed QoS is complementary to RiOS WAN optimization. Whereas Scalable Data Referencing (SDR), transport streamlining, and application streamlining techniques perform best with bandwidth-hungry or thick applications (such as email, file sharing, and backup), QoS improves performance in latency-sensitive or thin applications (such as VoIP and interactive applications). QoS depends on accurate classification of traffic, bandwidth reservation, and proper traffic priorities.

You can also look at the perspective as WAN optimization speeds up some traffic by reducing its bandwidth needs and accelerating it and QoS slows down some traffic to guarantee latency and bandwidth to other traffic. A combination of both techniques is ideal.

Because the SteelHead acts as a TCP proxy, the appliance already works with traffic flows. Riverbed QoS is an extensive flow-based QoS system, which can queue traffic on a per-flow basis and uses standard TCP mechanics for traffic shaping to avoid packet loss on a congested link. SteelHeads support inbound and outbound QoS.

The major functionalities of QoS are as follows:

- **Classification** - Identifies and groups traffic. Riverbed QoS identifies and groups traffic using the TCP/UDP-header information, VLAN ID, or AFE. Identified traffic is grouped into classes or QoS marked (differentiated services code point [DSCP] or Type of Service [ToS])—or both.
For more information about AFE, see [“Application Flow Engine” on page 102](#).
- **Policing** - Defines the action against the classified traffic. Riverbed QoS can define a minimum and maximum bandwidth per class, the priority of a class relative to other classes, and a weight for the usage of excess bandwidth (unused bandwidth, which is allocated to other classes).
- **Enforcement** - Determines how the action takes place. Enforcement is performed using the SteelHead QoS scheduler, which is based on the Hierarchical Fair Service Curve (HFSC) algorithm.

You can perform policing and enforcement on a downstream networking device when classification and QoS marking are performed on the SteelHead. This is useful if the SteelHead must integrate with an existing QoS implementation.

Riverbed QoS is flow based. For TCP, the SteelHead must detect the three-way handshake or it cannot classify a traffic flow. If a traffic flow is not classified, it falls into the default class.

After a traffic flow is classified, it is registered and cannot be reclassified. If a flow changes to a different application and is not reset by the application, the classification stays the same as before the change.

SteelHeads set up 1024-packet-deep packet buffers per configured class, regardless of the packet size. You can adjust the depths of the packet buffers using the CLI.

For information about adjusting packet buffers, see the *SteelHead User Guide*.

Riverbed QoS takes effect as soon as traffic congestion occurs on a link. Congestion occurs when multiple flows are sending data at the same time and the packets of the flows are not forwarded immediately, forming a queue.

There are two types of congestion: long term, which lasts a second or longer, and short term, which occurs for less than a second. Both types of congestion signal that one or more applications are slowed down because of other traffic.

You can best manage long-term congestion by shaping traffic: reserving bandwidth for the more important traffic. This is the most well-known implementation for QoS.

Short-term congestion is what can cause applications to hang for a second or reduce the quality of a VoIP conversation. You can manage short-term congestion by prioritizing traffic packets that are latency-sensitive to move ahead in the queue.

A well-designed QoS environment uses both prioritization and traffic shaping to guarantee bandwidth and latency for applications.

Many QoS implementations use some form of packet fair queueing (PFQ), such as weighted fair queueing (WFQ) or class-based weighted fair queueing (CBWFQ). As long as high-bandwidth traffic requires a high priority (or vice versa), PFQ systems perform adequately. However, problems arise for PFQ systems when the traffic mix includes high-priority, low-bandwidth traffic (such as VoIP), or high-bandwidth traffic that does not require a high priority (such as e-mail), particularly when both of these traffic types occur together.

Additional features such as low-latency queueing (LLQ) attempt to address these concerns by introducing a separate system of strict priority queueing that is used for high-priority traffic. However, LLQ is not a principled way of handling bandwidth and latency trade-offs. LLQ is a separate queueing mechanism meant as a work around for PFQ limitations.

The Riverbed QoS system is based on a patented version of HFSC. HFSC allows bandwidth allocation for multiple applications of varying latency sensitivity. HFSC explicitly considers delay and bandwidth at the same time. Latency is described in six priority levels (real-time, interactive, business critical, normal, low, and best-effort) that you assign to classes.

If you assign a priority to a class, the class can tolerate X delay, in which X is the priority setting. At the same time, bandwidth guarantees are respected. This enables Riverbed to deliver low latency to traffic without wasting bandwidth and deliver high bandwidth to delay-insensitive traffic without disrupting delay-sensitive traffic.

The Riverbed QoS system achieves the benefits of LLQ without the complexity and potential configuration errors of separate, parallel queueing mechanisms.

For example, you can enforce a mix of high-priority, low-bandwidth traffic patterns (SSH, Telnet, Citrix, RDP, CRM systems, and so on) with lower-priority, high-bandwidth traffic (FTP, backup, replication, and so on). This enables you to protect delay-sensitive traffic such as VoIP, alongside other delay-sensitive traffic such as video conferencing, RDP, and Citrix. You can do this without having to reserve large amounts of bandwidth for the traffic classes.

Additionally HFSC provides a framework for the following:

- **Link sharing** - Specifies how excess bandwidth is allocated among sibling classes. By default, all link shares are equal. QoS classes with a larger link-share weight are allocated more of the excess bandwidth than QoS classes with a lower link share weight.
- **Class hierarchy** - A class hierarchy lets a user create QoS classes as children of QoS classes other than the root class. This allows creating a class tree with overall parameters for a certain traffic types and specific parameters for subtypes of that traffic.

For more information about QoS classes, see [“QoS classes” on page 117](#).

You can apply Riverbed QoS to both pass-through and optimized traffic, which do not require the optimization service. QoS classification occurs during connection setup for optimized traffic—before optimization and compression. QoS shaping and enforcement occurs after optimization and compression. Pass-through traffic has the QoS shaping and enforcement applied appropriately. However, with the introduction of the SteelHead CX and SteelHead EX, there are platform-specific limits defined for the following QoS settings for outbound QoS:

- Maximum configurable root bandwidth
- Maximum number of classes
- Maximum number of rules
- Maximum number of sites

There are no platform-specific limits for inbound QoS.

For more information about limits, see [“Guidelines for the maximum number of QoS classes, sites, and rules” on page 140](#).

You can perform differentiated services code point (DSCP) marking and QoS enforcement on the same traffic. First mark the traffic, and then perform QoS qualification and management on the post-marked traffic.

For information about marking traffic, [“QoS marking” on page 137](#).

QoS concepts

This section describes the concepts of Riverbed QoS. It includes the following topics:

- [“Overview of QoS concepts” on page 114](#)
- [“QoS rules” on page 116](#)
- [“QoS classes” on page 117](#)
- [“QoS profiles” on page 124](#)

Overview of QoS concepts

In RiOS 9.0, Riverbed introduces a new process to configure QoS on SteelHeads. This new process greatly simplifies the configuration and administration efforts, compared to earlier RiOS versions.

A QoS configuration generally requires the configuration of many SteelHeads. To avoid repetitive configuration steps on single SteelHeads, we strongly recommend that you use the SCC 9.0 or later to configure QoS on your SteelHeads. SCC enables you to configure one time and to send the configuration out to multiple SteelHeads instead of connecting to a SteelHead, performing the configuration, and repeating the same configuration for all SteelHeads in the network.

For more information about using the SCC to configure SteelHeads, see the *SteelCentral Controller for SteelHead Deployment Guide* 9.0 or later.

QoS configuration consists of three building blocks, of which some are shared with the RiOS features path selection and secure transport.

- **Topology** - The topology combines a set of parameters that enable a SteelHead to build its view onto the network. The topology consists of network and site definitions, including information about how a site connects to a network. With the concept of a topology, a SteelHead can automatically build paths to remote sites and calculate the bandwidth that is available on these paths.

For more information about topology, see [“Topology” on page 91](#).

- **Applications** - An application is a set of criteria to classify traffic. The definition of an application enables the SteelHead to ensure that the traffic belonging to this application is treated according how you have configured it. Technically this means that the SteelHead can allocate the necessary bandwidth and priority for an application to ensure its optimal transport through the network. You can define an application on manually configured criteria or by using the AFE, which can recognize over 1200 applications.

For more information about applications and the AFE, see [“Application Definitions” on page 97](#).

- **QoS profile** - A profile combines a set of QoS classes and rules, which reflect your intent how to treat applications going to (outbound) or coming from (inbound) a site. To ensure an application is optimally working, you must classify it into a QoS class, which holds the parameters for bandwidth allocation and prioritization.

The classification of the application into a QoS class is done with QoS rules. A rule can be based on a single application or application groups.

You can assign a profile to one or multiple sites.

For more information about QoS classes, rules, and profiles, see [“QoS rules” on page 116](#), [“QoS classes” on page 117](#), and [“QoS profiles” on page 124](#).

The information contained in the building blocks topology, application and profile enables the SteelHead to efficiently apply QoS.

For more information on secure transport, see [“Overview of secure transport” on page 425](#) and the *SteelCentral Controller for SteelHead Deployment Guide*.

Application configuration and detection enables easy traffic classification. In topology, the configuration of networks, sites, and uplinks enables RiOS to build the QoS class tree at the site level and to calculate the available bandwidth to each configured site. This calculation also takes care of a possible oversubscription of the local links and enables you to add or delete sites without having to recalculate the bandwidths to the sites. The profiles apply the site specific QoS classes and rules to the sites to allocate bandwidth and priority to applications.

QoS rules

In RiOS 9.0 and later, QoS rules assign traffic to a particular QoS class. Prior to RiOS 9.0 QoS rules defined an application; however, applications are now defined separately. For more information, see [“Application Definitions” on page 97](#).

Note: You must define an application before you use it in a QoS rule.

A QoS rule is part of a profile and assigns an application or application group to a QoS class. If a QoS rule is based on an application group, it counts as a single rule. Grouping applications intelligently can significantly reduce the number of rules per sites.

Including the QoS rule in the profile prevents the repetitive configuration of QoS rules, because you can assign a QoS profile to multiple sites.

The order in which the rules appear in the QoS rules table is important. Rules from this list are applied from top to bottom. As soon as a rule is matched, the list is exited. Make sure that you place the more granular QoS rules at the top of the QoS rules list.

The SteelHead comes with default QoS classes, which are displayed on the Networking: Network Services: Quality of Service page, and include a section called Default QoS Rules. The section includes a default rule (Any), which sends all traffic to the LowPriority class. Starting with RiOS 9.5, there is no default profile; you need to create a profile and then add rules to classify traffic into the classes.

For more information about QoS profiles, see [“QoS profiles” on page 124](#).

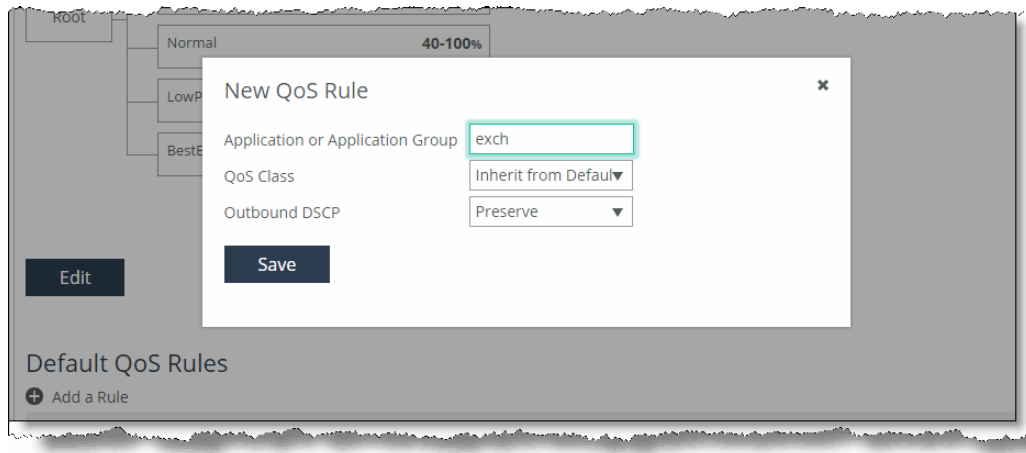
Riverbed QoS supports shaping and marking for multicast and broadcast traffic. To classify this type of traffic, you must configure a custom application based on IP header, because the AFE does not support multicast and broadcast traffic.

To configure a QoS rule

1. Choose Networking > Network Services: Quality of Service.
2. Click **Edit** for the QoS profile for which you want to configure a rule, or add a rule in the Default QoS Rules pane.
3. Select Add a Rule.

- Specify the first three or four letters of the application or application group you want to create a rule for and select it from the drop down menu of the available application and application groups (Figure 6-1).

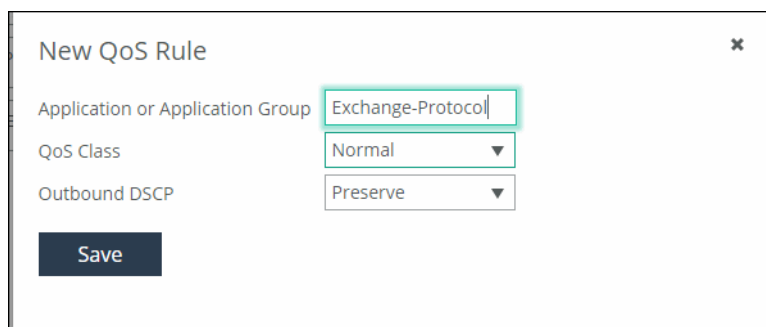
Figure 6-1. New QoS rule



- Assign a QoS class to the rule (Figure 6-2).

You can mark traffic for this application with a DSCP or preserve the existing one and click **Save**. The newly created QoS rule displays in the QoS rules table of the QoS profile.

Figure 6-2. New QoS rule priority and outbound DSCP setting



QoS classes

This section describes QoS classes and includes the following topics:

- “Class hierarchy” on page 118
- “Per-class parameters” on page 121
- “QoS class latency priorities” on page 122
- “QoS queue types” on page 123
- “QoS queue depth” on page 123
- “MX-TCP” on page 123

A QoS class represents an aggregation of traffic that is treated the same way by the QoS scheduler. The QoS class specifies the constraints and parameters, such as minimum bandwidth guarantee and latency priority, to ensure the optimal performance of applications. Additionally, the queue type specifies whether packets belonging to the same TCP/UDP session are treated as a flow or as individual packets.

In RiOS 9.0 and later, QoS classes are part of a QoS profile.

For more information about QoS profiles, see [“QoS profiles” on page 124](#).

To configure a QoS class

1. Chose Networking > Network Services: Quality of Service.
2. Create a new profile or edit the QoS profile for which you want to configure QoS classes.
3. In the QoS classes section, click **Edit**.

RiOS 9.0 does not create a default class in the profiles. Make sure you create a default class and point the default rule to it.

In RiOS versions prior to 9.0, you were required to choose between different QoS modes. RiOS 9.0 automatically sets up the QoS class hierarchy based on the information contained in the topology and profile configuration.

Class hierarchy

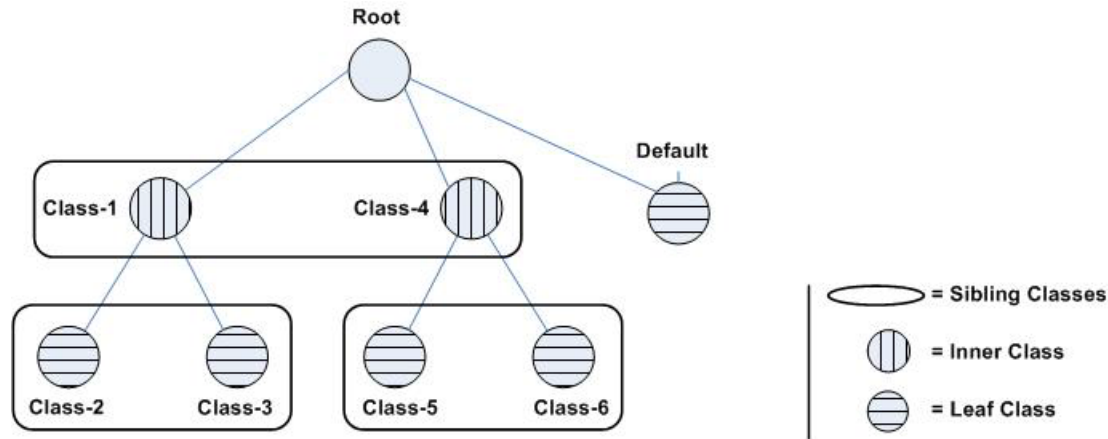
A class hierarchy allows creating QoS classes as children of QoS classes other than the root class. This allows you to create overall parameters for a certain traffic type and specify parameters for subtypes of that traffic.

In a class hierarchy, the following relationships exist between QoS classes:

- **Sibling classes** - Classes that share the same parent class.
- **Leaf classes** - Classes at the bottom of the class hierarchy.
- **Inner classes** - Classes that are neither the root class nor leaf classes.

QoS rules can only specify leaf classes as targets for traffic. **Figure 6-3** shows a class hierarchy's structure and the relationships between the QoS classes.

Figure 6-3. Hierarchical mode class structure



QoS controls the traffic of hierarchical QoS classes in the following manner:

- QoS rules assign active traffic to leaf classes.
- The QoS scheduler:
 - applies active leaf class parameters to the traffic.
 - applies parameters to inner classes that have active leaf class children.
 - continues this process up the class hierarchy.
 - constrains the total output bandwidth to the WAN rate specified on the root class.

The following examples show how class hierarchy controls traffic.

Figure 6-4 shows six QoS classes. The root and default QoS classes are built-in and are always present. This example shows the QoS class hierarchy structure.

Figure 6-4. Example of QoS class hierarchy

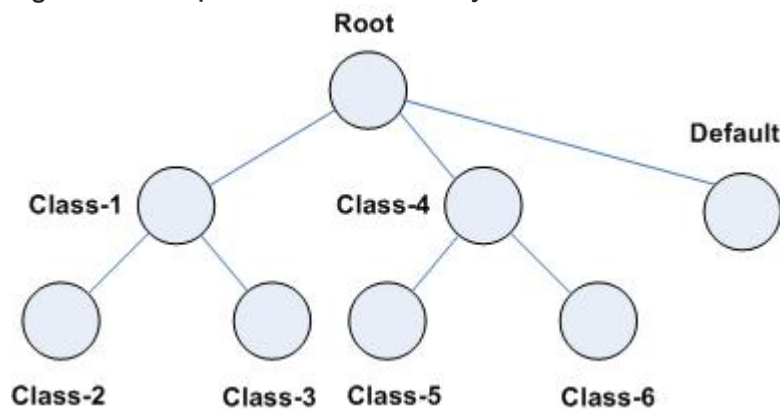
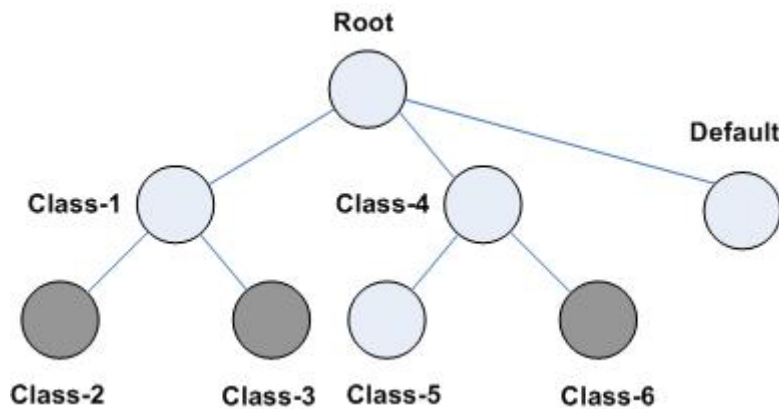


Figure 6-5 shows there is active traffic beyond the overall WAN bandwidth rate. This example shows a scenario in which the QoS rules place active traffic into three QoS classes: Classes 2, 3, and 6.

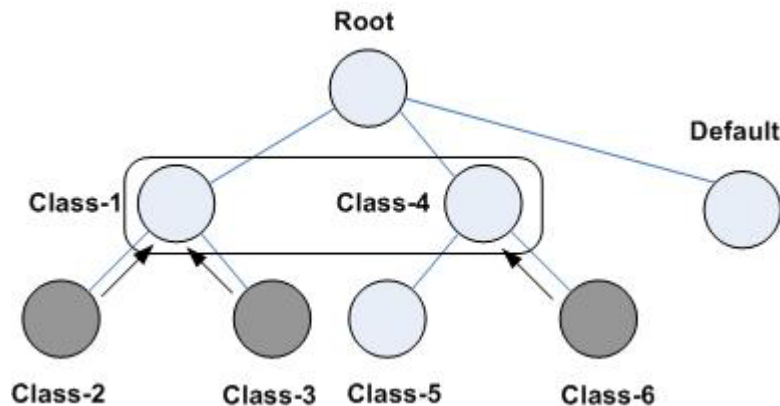
Figure 6-5. QoS classes 2, 3, and 6 have active traffic



Riverbed QoS rules place active traffic into QoS classes in the following manner.

- The QoS scheduler:
 - applies the constraints for the lower leaf classes.
 - applies bandwidth constraints to all leaf classes. The QoS scheduler awards minimum guarantee percentages among siblings, after which the QoS scheduler awards excess bandwidth, after which the QoS scheduler applies upper limits to the leaf class traffic.
 - applies latency priority to the leaf classes. For example, if class 2 is configured with a higher latency priority than class 3, the QoS scheduler gives traffic in class 2 the chance to be transmitted before class 3. Bandwidth guarantees still apply for the classes.
 - applies the constraints of the parent classes. The QoS scheduler treats the traffic of the children as one traffic class. The QoS scheduler uses class 1 and class 4 parameters to determine how to treat the traffic. Figure 6-6 shows the following points:
 - Traffic from class 2 and class 3 is logically combined and treated as if it were class 1 traffic.
 - Because class 4 only has active traffic from class 6, the QoS scheduler treats the traffic as if it were class 4 traffic.

Figure 6-6. How the QoS scheduler applies constraints of parent class to child classes

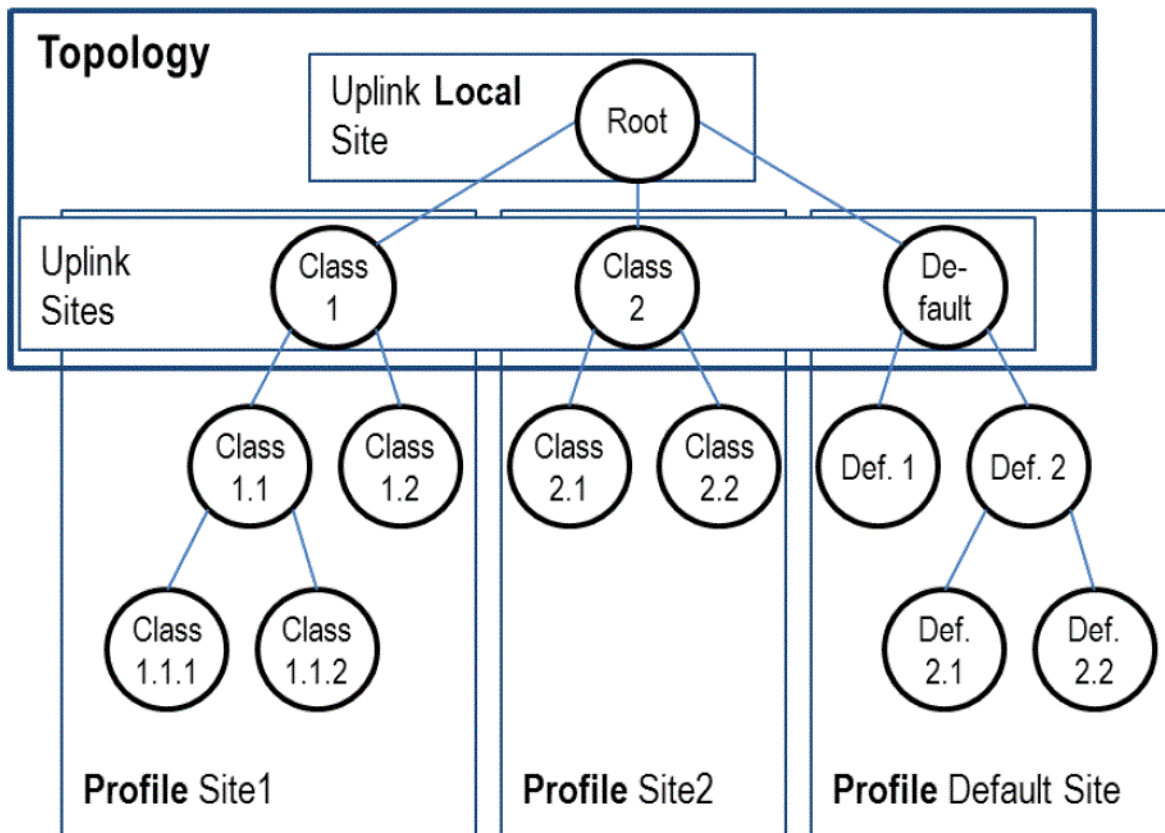


In RiOS 9.0 and later, you configure sites in the topology and traffic classes in profiles. With this information, RiOS 9.0 and later automatically set up the QoS class hierarchy.

The bandwidth of the uplink of the local site is assigned to the root class. The sites form the next layer of the class hierarchy, and the bandwidth of the site uplinks are assigned to the classes. The next layer of the hierarchy is then obtained from the profile, which is assigned to the class.

Figure 6-7 shows the configuration in the topology and the profile is translated into a class hierarchy.

Figure 6-7. Example of topology and profile translating into class hierarchy



For more information about topology, see [“Topology” on page 91](#). For more information about profiles, see [“QoS profiles” on page 124](#).

Per-class parameters

The QoS scheduler uses the per-class configured parameters to determine how to treat traffic belonging to the QoS class. The per-class parameters are as follows:

- **Latency priority** - There are six QoS class latency priorities. For details, see [“QoS class latency priorities” on page 122](#).
- **Queue types** - For details, see [“QoS queue types” on page 123](#).

- **Guaranteed minimum bandwidth** - When there is bandwidth contention, this bandwidth parameter specifies the minimum amount of bandwidth as a percentage of the parent class bandwidth. The QoS class might receive more bandwidth if there is unused bandwidth remaining. The ratio, of how much of the unused bandwidth is allocated to a class in relation to other classes, is determined by the link-share weight. The total minimum guaranteed bandwidth of all QoS classes must be less than or equal to 100% of the parent class. You can adjust the value as low as 0%.
- **Link share weight** - Allocates excess bandwidth based on the minimum bandwidth guarantee for each class. This parameter is not exposed in the SteelHead Management Console and implicitly configured with the minimum bandwidth guarantee for a class.
- **Maximum bandwidth** - Specifies the maximum allowed bandwidth a QoS class receives as a percentage of the parent class guaranteed bandwidth. The upper bandwidth limit is applied even if there is excess bandwidth available. The upper bandwidth limit must be greater than or equal to the minimum bandwidth guarantee for the class. The smallest value you can assign is 0.01%.
- **Connection limit** - Specifies the maximum number of optimized connections for the QoS class. When the limit is reached, all new connections are passed through unoptimized. In hierarchical mode, a parent class connection limit does not affect its child. Each child-class optimized connection is limited by the connection limit specified for its class. For example, if B is a child of A and the connection limit for A is set to 5, although the connection limit for A is set to 5, the connection limit for B is 10. Connection limit is supported only in in-path configurations. Connection limit is not supported in out- of-path or virtual-in-path configurations.

RiOS does not support a connection limit assigned to any QoS class that is associated with a QoS rule with an AFE component. An AFE component consists of a Layer-7 protocol specification. RiOS cannot honor the class connection limit because the QoS scheduler might subsequently reclassify the traffic flow after applying a more precise match using AFE identification.

In RiOS 9.0 and later, this parameter is available through the CLI only. For information about how to configure the connection limit, please refer to the *Riverbed Command-Line Interface Reference Manual*.

QoS class latency priorities

Latency priorities indicate how delay-sensitive a traffic class is. A latency priority does not control how bandwidth is used or shared among different QoS classes. You can assign a QoS class latency priority when you create a QoS class or modify it later.

Riverbed QoS has six QoS class latency priorities. The following table summarizes the QoS class latency priorities in descending order.

Latency priority	Example
Real time	VoIP, video conferencing
Interactive	Citrix, RDP, Telnet, and SSH
Business critical	Thick client applications, ERPs, CRMs
Normal priority	Internet browsing, file sharing, email

Latency priority	Example
Low priority	FTP, backup, replication, and other high-throughput data transfers; recreational applications such as audio file sharing
Best effort	Lowest priority

Typically, applications such as VoIP and video conferencing are given real-time latency priority, although applications that are especially delay-insensitive, such as backup and replication, are given low latency priority.

Important: The latency priority describes only the delay sensitivity of a class, not how much bandwidth it is allocated, nor how important the traffic is compared to other classes. Therefore, it is common to configure low latency priority for high-throughput, delay-insensitive applications such as ftp, backup, and replication.

QoS queue types

Each QoS class has a configured queue type parameter. The following types of parameters are available:

- **Stochastic Fairness Queueing (SFQ)** - Determines SteelHead behavior when the number of packets in a QoS class outbound queue exceeds the configured queue length. When SFQ is used, packets are dropped from within the queue, among the present traffic flows. SFQ ensures that each flow within the QoS class receives a fair share of output bandwidth relative to each other, preventing bursty flows from starving other flows within the QoS class. SFQ is the default queue parameter.
- **First-in, First-Out (FIFO)** - Determines SteelHead behavior when the number of packets in a QoS class outbound queue exceeds the configured queue length. When FIFO is used, packets received after this limit is reached are dropped, hence the first packets received are the first packets transmitted.
- **MX-TCP** - Not a queueing algorithm but rather a TCP transport type. For details, see [“MX-TCP” on page 123](#).

QoS queue depth

RiOS assigns a queue to each configured QoS class, which has the size of 100 packets. Therefore the queue size in bytes depends on the packet size. You can modify the queue size using the CLI.

For more information about how to configure the queue size of a QoS class, see the *Riverbed Command-Line Interface Reference Manual*.

MX-TCP

MX-TCP is a QoS class queue parameter, but with very different use cases than the other queue parameters. MX-TCP also has secondary effects that you must understand before you configure it.

When optimized traffic is mapped into a QoS class with the MX-TCP queueing parameter, the TCP congestion control mechanism for that traffic is altered on the SteelHead. The normal TCP behavior of reducing the outbound sending rate when detecting congestion or packet loss is disabled, and the outbound rate is made to match the minimum guaranteed bandwidth configured on the QoS class.

You can use MX-TCP to achieve high throughput rates even when the physical medium carrying the traffic has high loss rates. For example, a common usage of MX-TCP is for ensuring high throughput on satellite connections where no lower-layer loss recovery technique is in use.

Another usage of MX-TCP is to achieve high throughput over high-bandwidth, high-latency links, especially when intermediate routers do not have properly tuned interface buffers. Improperly tuned router buffers cause TCP to perceive congestion in the network, resulting in unnecessarily dropped packets, even when the network can support high throughput rates.

You must ensure the following guidelines when you enable MX-TCP:

- The QoS rule for MX-TCP is at the top of QoS rules list
- Use MX-TCP for optimized traffic only

As with any other QoS class, you can configure MX-TCP to scale from the specified minimum bandwidth up to a specific maximum. This allows MX-TCP to use available bandwidth during nonpeak hours. During traffic congestion, MX-TCP scales back to the configured minimum bandwidth.

For information about configuring MX-TCP before RiOS 8.5, see earlier versions of the *SteelHead Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

You can configure a specific rule for an MX-TCP class for packet-mode UDP traffic. An MX-TCP rule for packet-mode UDP traffic is useful for UDP bulk transfer and data-replication applications (for example, Aspera and Veritas Volume Replicator).

Packet-mode traffic matching any non-MX-TCP class is classified into the default class because QoS does not support packet-mode optimization.

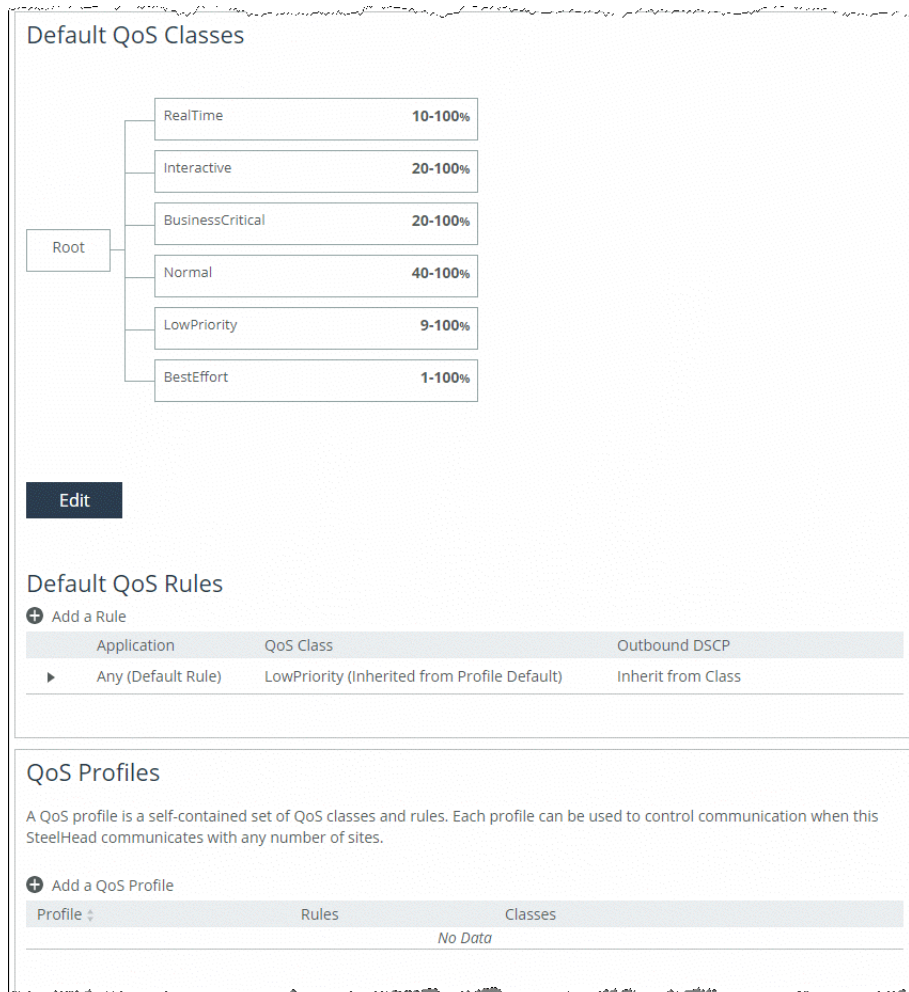
For more information about MX-TCP as a transport streaming lining mode, see “[Transport streamlining](#)” on [page 21](#). For an example of how to configure QoS and MX-TCP, see “[Configuring QoS and MX-TCP](#)” on [page 165](#).

QoS profiles

The QoS profile combines QoS classes and rules into a building block for QoS.

The SteelHead comes with a set of default QoS classes, which are shown on the Networking - Quality of Service configuration page. These classes map into the default QoS rules on the same page. You create a profile, then configure QoS rules to classify traffic into the QoS classes for the profile. The default classes are automatically assigned to the DefaultSite. This automatic assignment can be changed if needed on the Networking > Topology: Sites & Networks page. On Steelheads running RiOS versions earlier than 9.5, the default classes map into a default profile as well as the default QoS rules on the same page.

Figure 6-8. Default QoS Classes, Default QoS Rules, and QoS Profiles panes



You can create a hierarchical QoS class structure within a profile to segregate traffic based on flow source or destination, and you can apply different shaping rules and priorities to each leaf-class.

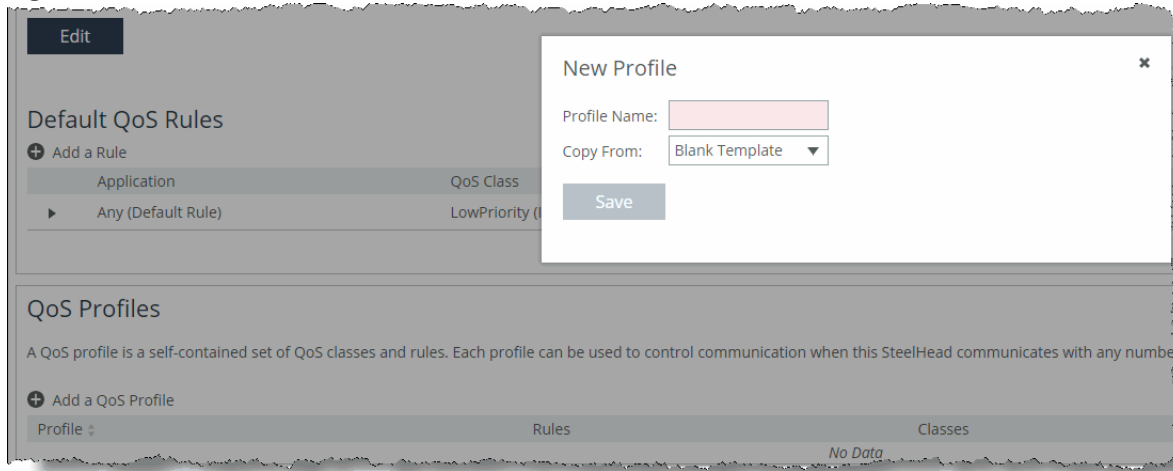
The SteelHead Management Consoles GUI supports the configurations of three levels of hierarchy. If more levels of hierarchy are needed, you can configure them using the CLI.

A profile can be used for inbound and outbound QoS.

To configure a QoS profile

1. Choose Networking > Network Service: Quality of Service.
2. Select Add a QoS Profile (Figure 6-9).

Figure 6-9. Add a new QoS profile



You can either create a new QoS profile based on a blank template or create a new QoS profile based on an existing one. Creating a new QoS profile based on an existing one simplifies the configuration because you can adjust parameters of the profile instead of starting with nothing.

3. Specify a name for the QoS profile and click **Save**.

The newly created QoS profile is displayed in the list.

4. Click **Edit** to configure the parameters ([Figure 6-10](#)).

Figure 6-10. QoS Profile Details page

QoS Profile Details Network Services > Quality of Service > QoS Profile Details ?

Profile Name

To manage which sites are assigned to this profile, visit the [Sites & Networks page](#).

Profile Name

QoS Classes

QoS Rules

Application	QoS Class
Any (Default Rule)	Root (Inherited from Profile Default)

A blank QoS profile comes with a root class and the default rule. To configure QoS classes, see [“QoS classes” on page 117](#). To configure QoS rules, see [“QoS rules” on page 116](#).

After you create the QoS profile, you can use the configured set of QoS classes and rules for multiple sites.

To assign a QoS profile to a site

1. Choose Networking > Topology: Sites & Networks.
2. Click **Edit Site** in the Sites section.
3. In the QoS Profiles section of the configuration page, open the Outbound QoS Profile drop-down menu and select the profile for this site.
4. Click **Save**.

You can assign the same QoS profile for inbound and outbound QoS. However, usually inbound QoS and outbound QoS have different functions, so it is likely that you need to configure a separate QoS profile for inbound QoS. For more information about inbound QoS, see [“Inbound QoS” on page 130](#).

For more information about configuration profiles, see [“Creating QoS profiles” on page 149](#) and [“MX-TCP” on page 123](#).

Configuring QoS

This section shows how to configure Riverbed QoS. It includes the following topics:

- [“QoS configuration workflow” on page 128](#)
- [“Enabling QoS” on page 128](#)
- [“QoS default classes” on page 129](#)

This section requires you be familiar with [“QoS concepts” on page 114](#), [“Topology” on page 91](#), and [“Application Definitions” on page 97](#).

QoS configuration workflow

In RiOS 9.0 and later, you configure QoS modularly. Some configuration tasks needed for QoS are also needed for the path selection and secure transport feature. You configure modules independently from each other, and RiOS combines the information from the modules into a QoS configuration. This greatly simplifies work you need to do to configure QoS in previous RiOS versions.

To configure QoS you must complete the following tasks:

1. **Configure the applications** - Configure and define the applications, which are to be handled with QoS. For details, see [“Application Definitions” on page 97](#).
2. **Configure the profiles** - Configure the classes and rules to prioritize and shape the applications. For details, see [“QoS profiles” on page 124](#).
3. **Configure the topology** - Set up the local and destination sites to do QoS for. Configure the bandwidths from and to the sites and assign a profile to the sites. For details, see [“Topology” on page 91](#).
4. Enable QoS on the SteelHead.

A QoS configuration generally requires the configuration of many SteelHeads. To avoid repetitive configuration steps on single SteelHeads, we strongly recommend that you use the SCC 9.0 or later to configure QoS on your SteelHeads. SCC enables you to configure one time and to send the configuration out to multiple SteelHeads instead of connecting to a SteelHead, performing the configuration and repeating the same configuration for all SteelHeads in the network.

Enabling QoS

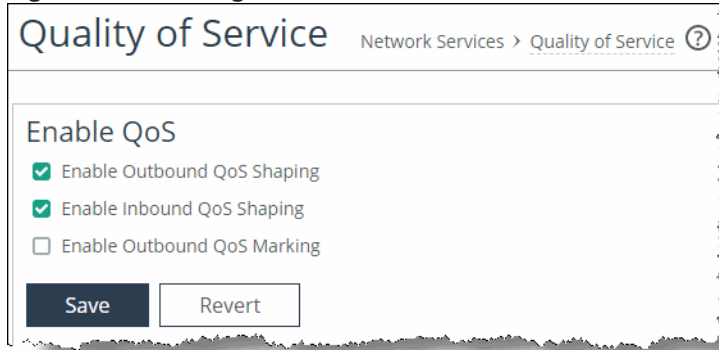
This section describes how to enable QoS.

To enable QoS on a SteelHead

1. Choose Network > Network Services: Quality of Service.
2. Select the check box next to the QoS feature you want to enable.

3. Click **Save** (Figure 6-11).

Figure 6-11. Enabling QoS



You can enable outbound QoS marking without having to enable inbound or outbound QoS shaping.

After you enable QoS globally, you must check to see if the in-path interface of the SteelHead is also enabled for QoS.

4. In the Manage QoS Per Interface section of the configuration page, enable QoS on the in-path interface.

Inbound and Outbound QoS shaping is enabled on the wan0_0 interface by default.

5. Click **Save**.

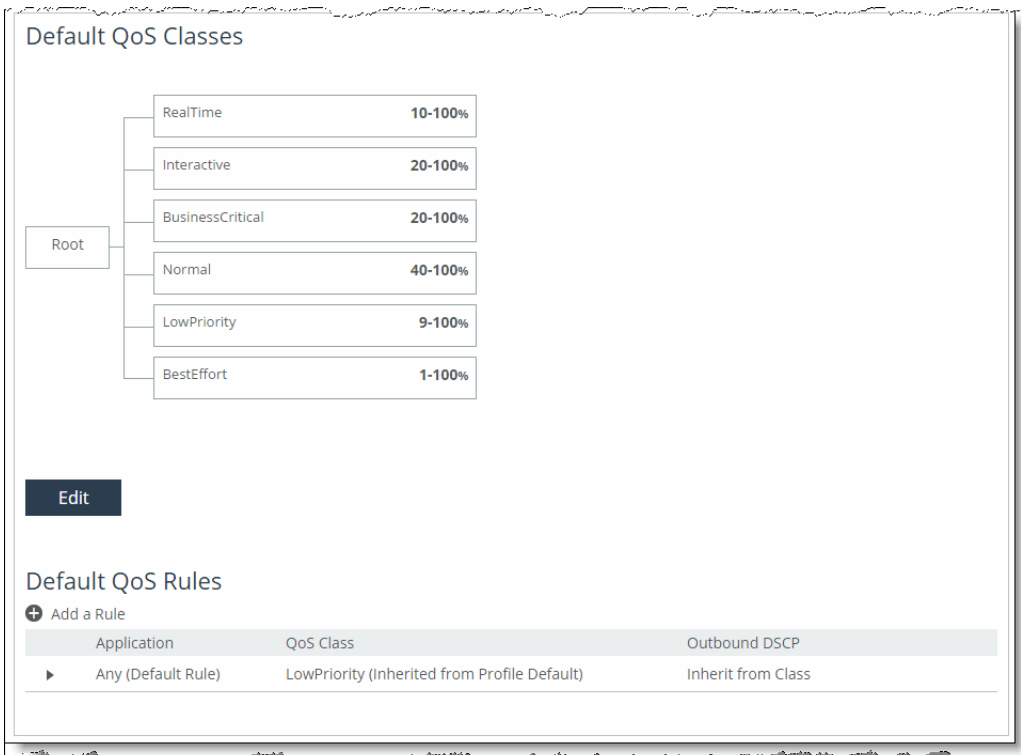
Note: Enabling QoS marking is a global configuration setting. You cannot enable it per in-path interface.

Important: RiOS QoS needs to be part of the TCP three-way handshake to be able to classify traffic. If you enable QoS on a SteelHead, all existing TCP flows are classified into the default site. We recommend that you enable QoS during nonproduction hours in a network.

QoS default classes

This section describes the default QoS classes.

The QoS default configuration is a set of classes that is based on Riverbed best practices. It is designed to provide an easy start into a complete QoS configuration and applied to the DefaultSite. Adding QoS rules to classify traffic into the default QoS classes will result in a working QoS configuration for inbound and outbound traffic on the SteelHead.

Figure 6-12. Default QoS Classes and Default QoS Rules panes

For more information about QoS classes, QoS rules, and QoS profiles, see [“QoS concepts” on page 114](#).

Inbound QoS

This section explains how Riverbed inbound QoS works and how you configure it. It includes the following topics:

- [“Introduction to inbound QoS” on page 131](#)
- [“Assigning an inbound QoS profile to a site” on page 131](#)

This section requires that you be familiar with [“QoS concepts” on page 114](#).

Introduction to inbound QoS

Inbound QoS enables you to allocate bandwidth and prioritize traffic flowing from the WAN into the LAN network behind the SteelHead. This configuration provides the benefits of QoS for environments that cannot meet their QoS requirements with only outbound QoS.

Reasons to configure inbound QoS include the following:

- Many business applications, such as VoIP and desktop video conferencing, now run over any-to-any mesh topologies.

The traffic generated by these applications typically travels directly from one branch office to another; for example, if a user in Branch Office A calls a user in Branch Office B, the VoIP call is routed directly, without having to traverse the data center. This traffic might compete with other traffic that is coming from the data center or from other sites, but it bypasses any QoS that is deployed at those sites. As a result, there is no network location from which you can use outbound QoS to control all incoming traffic going to Branch Office B. The only place where you can control all incoming traffic is at the branch itself. You can use inbound QoS at Branch Office B to guarantee bandwidth for critical applications and slow down traffic from other, less critical applications.

- Software as a Service (SaaS) applications and public cloud services accessed over the internet.

These applications compete with recreational internet traffic for bandwidth at the branch office. When users watch online videos or browse social networking sites, business applications can struggle to get the resources they need. With inbound QoS, you can ensure that business applications have enough room to get through.

Inbound QoS is used the same way as outbound QoS—to prioritize traffic from sites and types of traffic using rules and classes. You define the applications on the local SteelHead and then create a QoS profile corresponding to the shaping and prioritization policies.

For information about how to configure inbound QoS, see the *SteelHead User Guide*.

Inbound QoS applies the HFSC shaping policies to the ingress traffic. This behavior addresses environments in which bandwidth constraints exist at the downstream location. When this occurs, the downstream SteelHead (where inbound QoS is enabled) dynamically communicates the bandwidth constraints to the client transmitting the traffic. The client slows down the throughput and the traffic adheres to the configured inbound QoS rule. Inbound QoS, just like outbound QoS, is not a dual ended SteelHead solution. A single SteelHead can control inbound WAN traffic on its own.

For information about the HFSC queueing technology, see [“Overview of Riverbed QoS” on page 112](#) and the *SteelHead User Guide*.

Unlike earlier RiOS versions, RiOS 9.0 allows for applying inbound QoS per site (hierarchical inbound).

Assigning an inbound QoS profile to a site

You configure an inbound QoS profile the same way as an outbound QoS profile. Assigning an inbound QoS profile to a site requires you to think backwards. For example, you want to restrict certain traffic coming into the data center. When you configure the SteelHead that is located in the data center for inbound QoS, you need to assign a QoS profile to that site. Do not assign a QoS profile for inbound QoS to the data center SteelHead.

To assign a QoS profiles for inbound QoS to a site

1. Choose Networking > Topology: Sites & Networks.
2. Click **Edit Site** in the Sites section.
3. In the QoS Profiles section of the configuration page, open the Inbound QoS Profile drop-down menu and select the profile for this site (**Figure 6-13**).

Figure 6-13. Edit an Existing Site page

Edit an Existing Site

Basic Information

Site name

Network Information

Subnets Subnets define how the SteelHead identifies this site. Separate with comma " , "

SteelHead Peers Peers are used for path monitoring and GRE Tunneling. Separate with comma " , "

QoS Profiles

Inbound QoS Profile

Outbound QoS Profile

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

+ Add New Uplink

X

LAN bypass

Virtual in-path network topologies in which the LAN-bound traffic traverses the WAN interface might require that you configure the SteelHead to bypass LAN-bound traffic so that it is not subject to the maximum root bandwidth limit. Some deployment examples are WCCP or a WAN-side default gateway. The LAN bypass feature enables you to exempt certain subnets from QoS enforcement. You can configure LAN bypass on both inbound and outbound QoS.

For information about LAN bypass, see the *SteelHead User Guide*.

QoS for IPv6

IPv6 traffic is not currently supported for QoS shaping or AFE-based classification. If you enable QoS shaping for a specific interface, all IPv6 packets for that interface are classified to the default class.

You can mark IPv6 traffic with an IP ToS value. You can also configure the SteelHead to reflect an existing traffic class from the LAN side to the WAN side of the SteelHead.

For more information about IPv6, see [“IPv6” on page 307](#).

QoS in virtual in-path and out-of-path deployments

You can use QoS enforcement, on both inbound and outbound traffic, in virtual in-path deployments (for example, WCCP and PBR) and out-of-path deployments. In both of these types of deployments, you connect the SteelHead to the network through a single interface: the WAN interface for WCCP deployments, and the primary interface for out-of-path deployments. You enable QoS for these types of deployments:

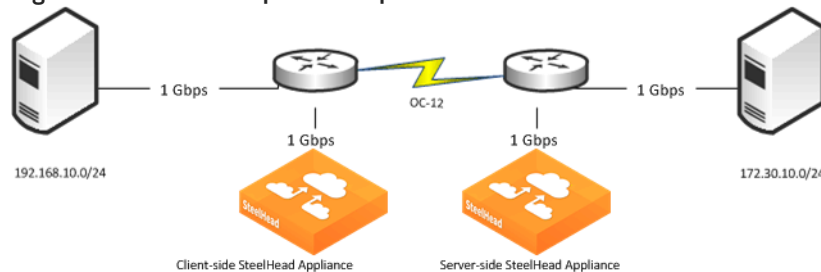
- Configure subnet side rules appropriately with LAN IP subnets on the LAN side. The subnet side rules define which part of the interface traffic belongs to the LAN side.

For information about subnet side rules, see the *SteelHead User Guide*.

- Set the WAN throughput for the network interfaces to the total speed of the LAN and WAN interfaces or to the speed of the local link, whichever number is lower.
- Configure QoS as if you have deployed the SteelHead in a physical in-path mode.

[Figure 6-14](#) shows two SteelHeads are deployed in a virtual in-path deployment. The client side IP subnets (192.168.10.0/24 for the client side SteelHead and 172.30.10.0/24 on the server-side SteelHead) need to be configured as LAN-side subnets in the Subnet side rules table.

Figure 6-14. Virtual in-path example



QoS in multiple SteelHead deployments

You can use QoS when multiple SteelHeads are optimizing traffic for the same WAN link. In these cases, you must configure the QoS settings so that the SteelHeads share the available WAN bandwidth. For example, if traffic is to be load balanced evenly across two SteelHeads, then the maximum WAN bandwidth configured for each SteelHead is one-half of the total available WAN bandwidth. This scenario is often found in multiple SteelHead data protection deployments.

For information about data protection deployments, see [“Designing for scalability and high availability” on page 405](#).

QoS and multiple WAN interfaces

You can enable QoS on WAN interfaces using different bandwidths. For example, you can configure the uplink for the inpath0_0 interface for 10 Mbps to connect to a 10-Mbps terrestrial link and configure the uplink for inpath0_1 for 1 Mbps to connect to a 1-Mbps satellite backup link.

Keep in mind that the outbound WAN capacity limit is based per SteelHead, regardless of the number of interfaces. For example, with a SteelHead EX1160, the QoS bandwidth is limited to 100 Mbps of traffic regardless of the number of interfaces installed or enabled for QoS.

The bandwidth and latency allocation are applied across all the interfaces. Continuing with the previous example, if you allocate 10% of the bandwidth for CIFS traffic, then that would be 10% of the 10-Mbps link and 10% of the 1-Mbps link.

Integrating SteelHeads into existing QoS architectures

This section describes the integration of SteelHeads into existing QoS architectures. This section includes the following topics:

- [“WAN-side traffic characteristics and QoS” on page 135](#)
- [“QoS integration techniques” on page 135](#)
- [“QoS marking” on page 137](#)

When you integrate SteelHeads into your QoS architecture, you can:

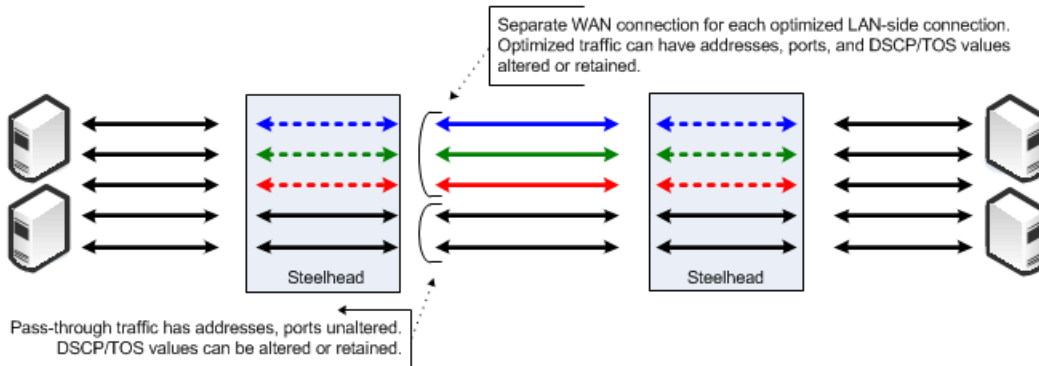
- retain the original DSCP or IP precedence values.
- choose the DSCP or IP precedence values.
- retain the original destination TCP port.
- choose the destination TCP port.
- retain all of the original IP addresses and TCP ports.

You do not have to use all of the SteelHead functions on your optimized connections. You can selectively apply functions to different optimized traffic, based on attributes such as IP addresses, TCP ports, DSCP, VLAN tags, and payload.

WAN-side traffic characteristics and QoS

When you integrate SteelHeads into an existing QoS architecture, it is helpful to understand how optimized and pass-through traffic appear to the WAN or any WAN-side infrastructure. [Figure 6-15](#) shows how traffic appears on the WAN when SteelHeads are present.

Figure 6-15. How traffic appears to the WAN when SteelHeads are present



When SteelHeads are present in a network:

- the optimized data for each LAN-side connection is carried on a unique WAN-side TCP connection.
- the IP addresses, TCP ports, and DSCP or IP precedence values of the WAN connections are determined by the SteelHead WAN visibility mode, and the QoS marking settings configured for the connection.
- the amount of bandwidth and delay assigned to traffic when you enable Riverbed QoS enforcement is determined by the Riverbed QoS enforcement configuration. This configuration applies to both pass-through and optimized traffic. However, this configuration is separate from configuring SteelHead WAN visibility modes.

For information about WAN visibility modes, see [“Overview of WAN visibility” on page 63](#).

QoS integration techniques

This section provides examples of different QoS integration techniques, depending on the environment, as described in the following topics:

- [“QoS policy differentiating voice versus nonvoice traffic” on page 136](#)
- [“SteelHead honoring premarked LAN-side traffic” on page 136](#)
- [“SteelHead remarking traffic from the LAN side” on page 136](#)
- [“SteelHead enforcing the QoS policy” on page 136](#)

These examples assume that the post-integration goal is to treat optimized and nonoptimized traffic in the same manner with respect to QoS policies; you might want to allocate different network resources to optimized traffic.

For information about QoS marking, see [“QoS marking” on page 137](#).

In networks in which both classification or marking and enforcement are performed on traffic after it passes through the SteelHead, you have the following configuration options:

- In a network in which classification and enforcement is based only on TCP ports, you can use port mapping or the port transparency WAN visibility mode.

For information about port transparency, see [“Port transparency” on page 66](#).

- In a network where classification and enforcement is based on IP addresses, you can use the full address transparency WAN visibility mode.

For information about full address transparency, see [“Full address transparency” on page 67](#).

QoS policy differentiating voice versus nonvoice traffic

In some networks, QoS policies do not differentiate traffic that is optimized by the SteelHead. For example, because VoIP traffic is passed through the SteelHead, a QoS policy that gives priority to only VoIP traffic, without differentiating between non-VoIP traffic, is unaffected by the introduction of SteelHeads. In these networks, you do not need to make QoS configuration changes to maintain the existing policy, because the existing configuration treats all non-VoIP traffic identically, regardless of whether it is optimized by the SteelHead.

SteelHead honoring premarked LAN-side traffic

Another example of a network that might not require QoS configuration changes to integrate SteelHeads is where traffic is marked with DSCP or ToS values before reaching the SteelHead, and enforcement is made after reaching the SteelHeads based only on DSCP or ToS. The default SteelHead settings reflect the DSCP or ToS values from the LAN side to the WAN side of an optimized connection.

For example, you configure QoS by marking the DSCP values at the source or on LAN-side switches, and enforcement is performed on WAN routers, the WAN routers detect the same DSCP values for all classes of traffic, optimized or not.

SteelHead remarking traffic from the LAN side

You can mark or remark a packet with a different DSCP or ToS value as it enters the SteelHead. This remarking process is sometimes necessary because an end host can set an inappropriate DSCP value for traffic that might otherwise receive a lower priority. In this scenario, the QoS enforcement remains the responsibility of the WAN router.

SteelHead enforcing the QoS policy

Instead of enforcing the QoS policy on the WAN router, you can configure the SteelHead to enforce the QoS policy instead. However, we recommend that you also maintain a QoS policy on the WAN router to ensure that the traffic quality is guaranteed in the event that the SteelHead becomes unavailable.

QoS marking

This section describes how to use SteelHead QoS marking when integrating SteelHeads into an existing QoS architecture. This section includes the following topics:

- [“QoS marking for SteelHead control traffic” on page 137](#)
- [“QoS marking default setting” on page 138](#)
- [“QoS marking design considerations” on page 139](#)

SteelHeads can retain or alter the DSCP or IP ToS value of both pass-through traffic and optimized traffic. The DSCP or IP ToS values can be set in the QoS profile per QoS class and per QoS rule or per application.

Note: You can enable QoS marking without enabling QoS shaping.

QoS reporting does not show any output if QoS marking is enabled without enabling QoS shaping. To get reporting for marked traffic, you must enable QoS shaping.

For example QoS marking configurations, see [“Configuring QoS marking on SteelHeads” on page 162](#).

Note: In RiOS 7.0, the DSCP or IP ToS value definition changed significantly from earlier versions of RiOS. If you are running an earlier version of the RiOS, see an earlier version of the *SteelHead Deployment Guide* for instructions on how to configure the DSCP or IP ToS value.

QoS marking for SteelHead control traffic

By default, the setup of out-of-band control connections (OOB splice) are not marked with a DSCP value. If you are integrating a SteelHead into a network that provides multiple classes of service, SteelHead control traffic is classified into the default class. The default class is usually treated with the lowest priority. This traffic classification can lead to packet loss and high latency for SteelHead control traffic if the link is congested, which impacts optimization performance.

For optimized connections (inner channel), the client-side SteelHead sets a configured DSCP value from the very start of the connection, according to the first fully matching QoS rule. Riverbed Application Flow Engine (AFE) cannot classify traffic on the first packets (TCP handshake), so only QoS rules based on the IP protocol header (IP address, TCP port number, UDP port number, and so on) match on the first packet (TCP SYN). On the server-side SteelHead, the DSCP marking value seen on the TCP SYN packet is reflected onto the TCP SYN-ACK packet, so the packets of the TCP session setup (TCP handshake) are correctly marked with the configured DSCP.

For more information about AFE, see [“Application Flow Engine” on page 102](#).

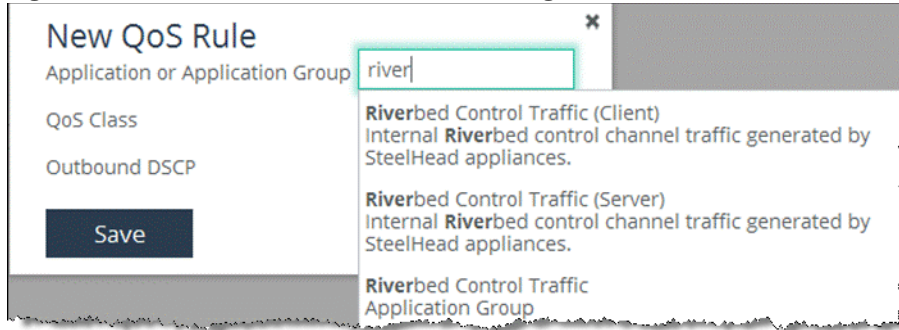
In versions of RiOS prior to 9.1, the Global DSCP feature was used to set a DSCP value to the first packets of a session (TCP handshake), which is no longer needed. For more information about Global DSCP, refer to an earlier version of the *SteelHead Deployment Guide*.

If your existing network provides multiple classes of service based on DSCP values, and you are integrating a SteelHead into your environment, you can use the out-of-band (OOB) DSCP marking feature to prevent dropped packets and other undesired effects for SteelHeads control traffic.

The OOB DSCP marking feature enables you to explicitly set a DSCP value for the control channel between SteelHeads (OOB splice).

To configure OOB DSCP marking, create a new rule in the QoS profile in use, which is based on the Riverbed control traffic application group, and configure the desired DSCP value.

Figure 6-16. Riverbed control traffic application group



If you want to set a DSCP value for only one direction, choose either the Riverbed Control Traffic (Client) or Riverbed Control Traffic (Server) application.

For more information about how to configure OOB DSCP marking, see the *SteelHead User Guide*.

QoS marking default setting

By default, SteelHeads reflects the DSCP or IP ToS value found on pass-through traffic and optimized connections. The default value for DSCP or IP ToS are set to:

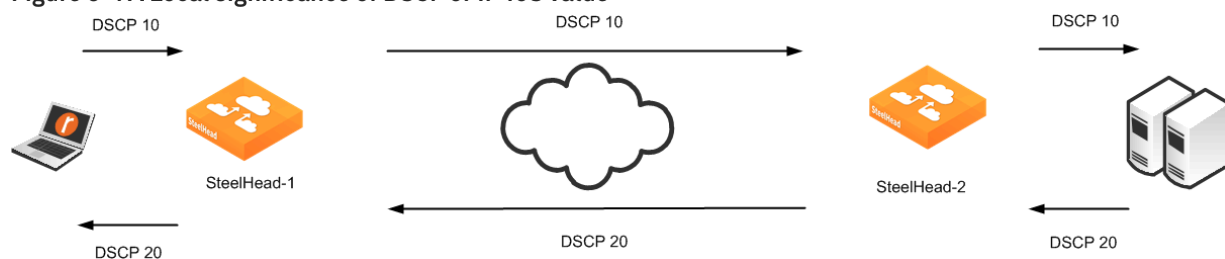
- preserve in the QoS class.
- preserve in the QoS rules.
- Any in the application rules configuration (this means do not change anything).

By default, the DSCP or IP ToS value on pass-through or optimized traffic is unchanged when it passes through the SteelHead.

If a DSCP or IP ToS value is set for certain traffic, this value is only significant for the SteelHead on which it is configured. Another SteelHead on the same network may set the DSCP or IP ToS value differently.

Figure 6-17 shows the local significance of DSCP or IP ToS values.

Figure 6-17. Local significance of DSCP or IP ToS value



QoS marking design considerations

Consider the following when using QoS marking:

- You can use a DSCP or IP ToS value to classify traffic or even to define an application. You can define an application based on an existing DSCP or IP ToS value instead of specifying individual header or application flow rules. This value simplifies the configuration, because traffic is marked before it is classified.
- You cannot use QoS marking for traffic exiting the LAN interface. The LAN interface reflects the QoS marking value it receives from the WAN interface, for both optimized and unoptimized traffic.
- QoS marking behavior is the same for in-path, virtual in-path, or out-of-path deployments. The one exception is that in a virtual in-path deployment you can set a DSCP or IP ToS traffic for traffic destined to the LAN, because in a virtual in-path deployment only the WAN interface of the SteelHead is activated.

QoS enforcement best practices

We recommend the following actions to ensure optimal performance with the least amount of initial and ongoing configuration:

- Configure QoS while the QoS functionality is disabled and only enable it after you are ready for the changes to take effect.
- Configure an uplink for the default class and assign a reasonable bandwidth to it.

The built-in default class initially does not have uplinks configured. Because of this, the physical bandwidth of the in-path interface is used. This use can lead to a high and undesired oversubscription of the uplink of the local site and thus impact the minimum bandwidth guarantee of the remote sites.

A typical indication that you must adjust the default class uplinks bandwidth is when traffic that is specified in the sites or QoS rules appears to be slow during times of congestion, while other traffic not specified in sites or QoS rules (typical examples include web browsing and routing updates) works fine.

- When configuring a profile, create a default class and point the default rule (Any) to the default class.
- Be aware that flows can be incorrectly classified if there are asymmetric routes in the network in which you have enabled the QoS features.

Upgrading to RiOS 9.0

RiOS 9.0 provides a mechanism to convert the QoS configuration of earlier RiOS versions into the format of a RiOS 9.0 configuration. The conversion process is transparent to and happens during the first boot-up of RiOS 9.0 on a SteelHead.

The QoS concepts and resulting configuration in RiOS 9.0 are very different from those of earlier RiOS versions. In some cases it can be easier to create a new QoS configuration in RiOS 9.0, than to migrate an existing QoS configuration. You must handle any migration of an advanced QoS configuration from earlier RiOS versions with care.

We strongly recommend that you closely analyze a migrated QoS configuration after you upgrade a SteelHead to RiOS 9.0.

For more information about migrations, see *RiOS 9.0 QoS Migration* at <https://splash.riverbed.com/docs/DOC-5443>.

Guidelines for the maximum number of QoS classes, sites, and rules

The number of QoS classes, sites, and rules you can create on a SteelHead depends on the appliance model number, the traffic flow, and other RiOS features you enable.

Important: If you are using a release previous to RiOS 8.5.1, some of the features described in the chapter might not be applicable. For information about QoS before RiOS 8.5.1, see earlier versions of the *SteelHead Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

We recommend that you do not exceed the guidelines in this section.

In RiOS 9.0 and later, the maximum number of rules is 2000. The number of rules used is determined by multiplying the number of rules with the number of profiles. For example, if you have 15 rules and 1 profile configured for 100 sites, the total number of rules is 15. If you have 20 rules in 2 profiles configured and apply each profile to 100 sites, the total number of rules is 40. If you are using an application group in a rule, it counts as 1 rule regardless of the number of applications in the group.

RiOS 7.0.3 and later enforce a maximum configurable root bandwidth per appliance model. Bandwidth enforcement issues a warning when you configure the sum of the bandwidth interfaces as a value greater than the model-specific QoS limits. The limits were introduced in RiOS 6.5.4 and 7.0.1. The warning appears when you save a configured bandwidth limit that exceeds the supported limit. If you receive the warning, adjust the sum of the configured QoS interface values to be lower or equal to the model-specific bandwidth limit and save the configuration.

We strongly recommend that you configure the bandwidth at or below the appliance limit, because problems arise when you exceed it. Upgrading a SteelHead to RiOS 7.0.3 does not automatically fail when the configuration has a greater QoS bandwidth limit than the appliance supports. However, after the upgrade, RiOS begins enforcing the bandwidth limits and disallows any QoS configuration changes. This limitation could result in having to reconfigure all QoS policies to accommodate the bandwidth limit.

In RiOS 8.5.1 and later, QoS limits are no longer enforced, with the exception of the QoS bandwidth. To restrict the possibility of exhausting the system resources, use these safeguard limits:

- Maximum number of total rules: 2000
- Maximum number of sites in basic outbound QoS mode: 100
- Maximum number of sites in advanced outbound QoS mode: 200

In RiOS 6.5 and later, QoS performance is based on the number of connections per second. The guidelines are as follows:

- Desktop models: 200 new connections per second
- 1U models: 500 new connections per second
- 3U models: 1,000 new connections per second

Important: Exceeding the recommended limits can lead to severe delays when you change your Riverbed QoS configuration or boot the SteelHead.

For more information about the bandwidth limits and the Riverbed recommended optimal performance guidelines for the SteelHead models, see <https://www.riverbed.com/document/fpo/Products/SteelHead/Spec%20Sheet%20-%20Steelhead%20Family.pdf>.

QoS Configuration Examples

This chapter provides examples of QoS configurations. This chapter includes the following sections:

- [“Configuring QoS using best practices” on page 143](#)
- [“Configuring QoS marking on SteelHeads” on page 162](#)
- [“Configuring QoS and MX-TCP” on page 165](#)

For general QoS information, see [“QoS configuration and integration” on page 111](#).

For a QoS and Citrix configuration example, see the *SteelHead Deployment Guide - Protocols*. For a QoS and SSL common name matching example, see [“Overview of the Application Flow Engine” on page 102](#).

Configuring QoS using best practices

This section describes an example network and the basic steps for configuring Riverbed QoS using the given specifications. This section includes the following topics:

- [“Example QoS scenario” on page 143](#)
- [“Configuring QoS on the data center SteelHead” on page 146](#)
- [“Configuring applications” on page 147](#)
- [“Creating QoS profiles” on page 149](#)
- [“Configuring topology” on page 156](#)
- [“Enabling QoS on the SteelHead” on page 160](#)

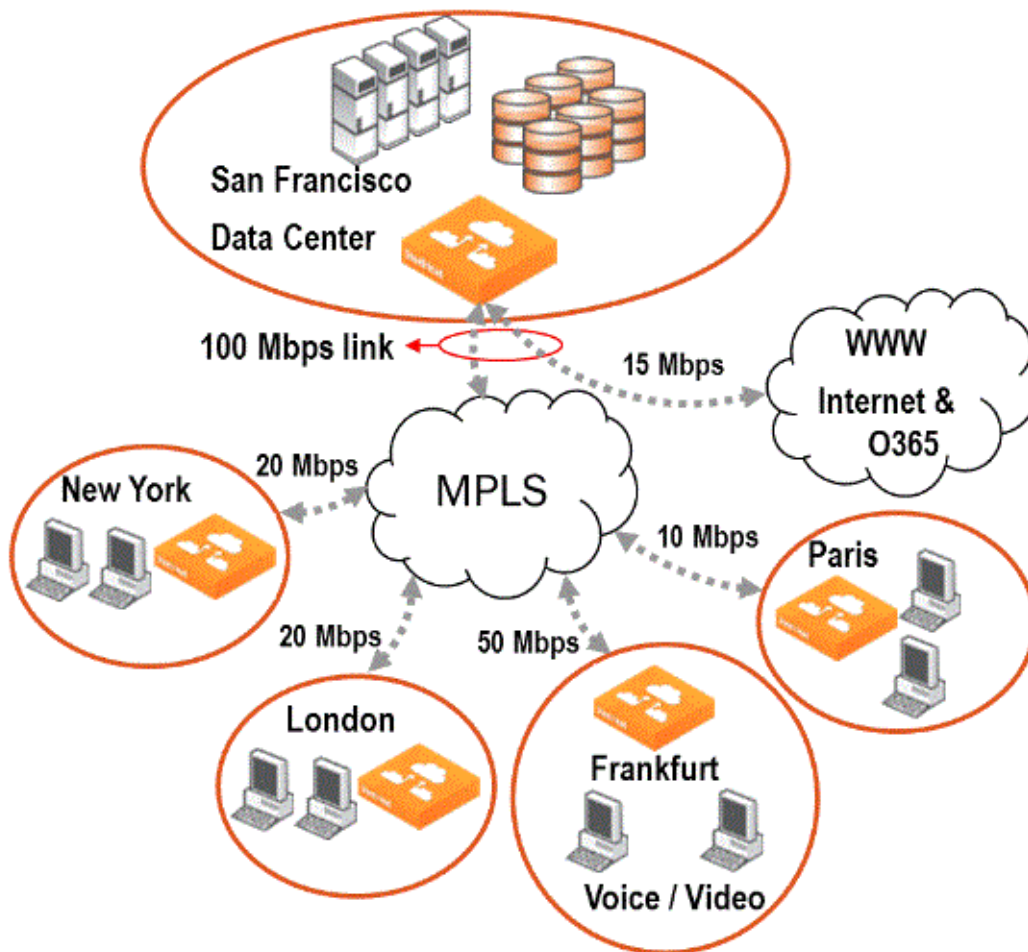
For more information on best practices, see [“QoS enforcement best practices” on page 139](#).

Example QoS scenario

This scenario is the basis for the configuration described in [“Configuring QoS on the data center SteelHead” on page 146](#).

Figure 7-1 shows a simple hub and spoke setup. The San Francisco data center provides the services for the remote sites and it has a connection to the internet for browsing and MS Office 365 applications.

Figure 7-1. SteelHead configuration example



The overall goal for implementing QoS is to protect VoIP traffic from all other traffic and to protect MS Office 365 traffic from internet browsing in the data center.

The data center:

- hosts telephony services using the RTP protocol for voice.
- hosts all other services that do not need special shaping or prioritization.
- uses a SteelHead that:
 - is deployed physically in-path.
 - has an uplink on its in-path0_0 interface with 100 Mbps of bandwidth to the MPLS network.
 - serves the four remote branch offices—New York, London, Frankfurt, and Paris—which all connect to the MPLS network.
 - has an uplink on in-path0_1 to the internet with a bandwidth of 15 Mbps for browsing and access to MS Office 365.

- has the following QoS goals for outbound traffic:
 - for the New York, London, and Paris sites, VoIP (using RTP) traffic is prioritized and guaranteed 20% of the sites bandwidth.
 - for the Frankfurt site, 30% of the sites bandwidth is guaranteed for a video conferencing system. Of these 30%, one-third of the bandwidth is guaranteed for voice and two-thirds of the bandwidth for video. The video conferencing system uses the RTP-Voice and RTP-Video protocol.
- has the following goal for inbound traffic:
 - guarantee the same bandwidth for incoming VoIP calls (using RTP) from the remote sites as for outgoing VoIP calls.
 - protect MS Office 365 traffic from ordinary internet browsing traffic in the data center.

Branch offices have:

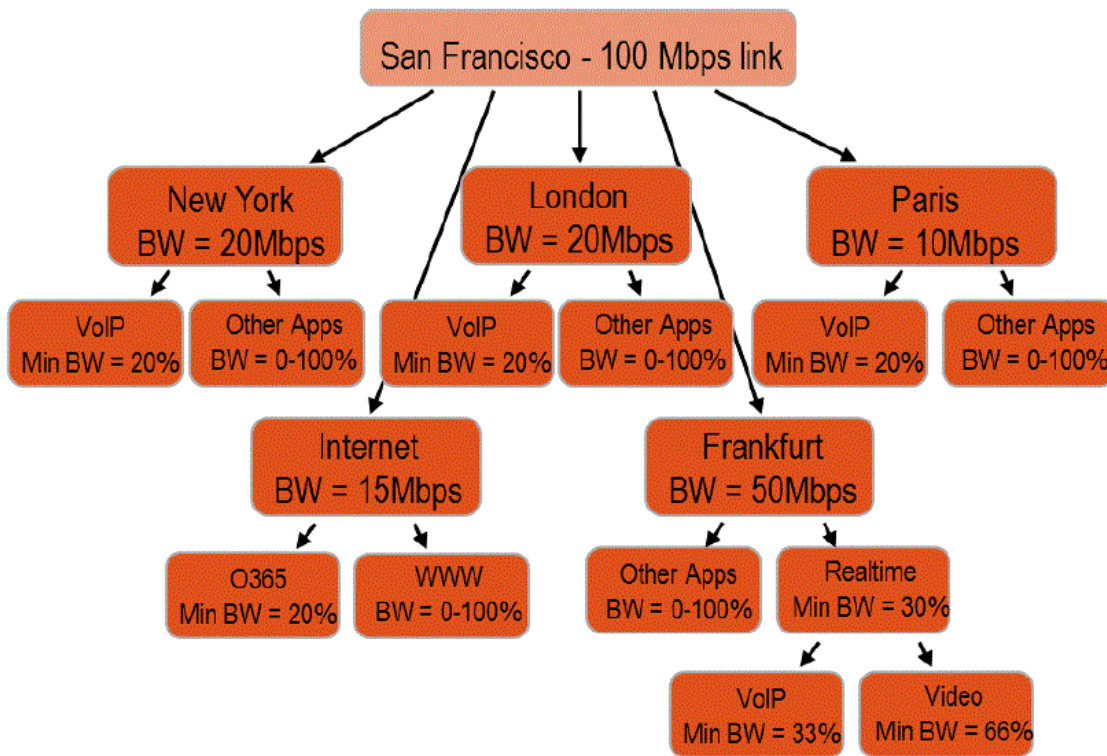
- a 20-Mbps link to the MPLS network for New York and London each.
- a 10-Mbps link to the MPLS network for Paris
- a 50-Mbps link to the MPLS network for Frankfurt.

Frankfurt has a video conferencing system installed and wants to guarantee 15Mbps of bandwidth for it. Of the 15 Mbps, 5 are guaranteed for voice and 10 for video.

- SteelHeads that are deployed physically in-path.

Figure 7-2 shows a graphical representation of the goal for implementing QoS results with the following site and profile structure.

Figure 7-2. Graphical representation of the goal for QoS implementation



From the point of view of the San Francisco data center, you can use this site and profile structure for outbound and inbound QoS.

Configuring QoS on the data center SteelHead

This section describes the overall workflow to configure QoS on the data center SteelHead. This example does not require QoS configuration of the SteelHeads in the branch offices.

The workflow is as follows:

1. Configure applications, if necessary.
2. Configure QoS profiles for the sites for inbound and outbound traffic.
3. Configure the topology and assign the QoS profiles to the sites.
4. Enable QoS.

Configuring applications

Applications must be known to the SteelHead so that you can configure QoS rules. You must configure or verify application definitions as the first step when configuring QoS. Configuring applications simplifies the workflow, so you do not have to jump between configuration pages.

For more information about applications, see [“Application Definitions” on page 97](#).

The example QoS scenario has the following important applications:

- RTP-Voice
- RTP-Video
- Office 365

To be able to protect these applications from other traffic, you need to ensure that the SteelHead can recognize them so you can use them to set up a QoS rule in the QoS profile later.

To verify that an application is known to the SteelHead

1. Choose Networking > App Definitions: Applications.
2. Select Add.
3. In the New Application screen, specify the first letters of the application name you want to verify into the Application Layer Protocol field.

The applications RTP-Voice and RTP-Video are already known to the SteelHead (Figure 7-3).

Figure 7-3. New Application page

The screenshot shows the 'New Application' dialog box in the SteelHead configuration interface. The dialog is titled 'New Application' and has a close button (X). It contains the following fields and options:

- Name:** A text input field.
- Description:** A text input field.
- Traffic Characteristics:**
 - Local Subnet:** 0.0.0.0/0
 - Remote Subnet:** 0.0.0.0/0
 - Transport Layer Protocol:** Any
 - Application Layer Protocol:** RTP (selected, with a dropdown menu open showing options: RTP, RTP-Audio, RTP-Video, SRTP, SRTP-Audio, SRTP-Video)
 - VLAN Tag ID:**
 - Outbound DSCP:**
- Application Properties:**
 - Application Group:** Custom Applications
 - Category:** Collaboration
 - Business Criticality:** Low

The background shows the 'Applications' page with a table of application definitions. The table has columns for Name, Application Group, and Category. The 'Add' button is visible.

4. Check for Office 365.

Office 365 is known to the SteelHead (Figure 7-4).

Figure 7-4. MS-Office-365 in AFE

The screenshot shows the 'MS-Office-365' entry in the AFE (Application Feature Engine) table. The entry has the following values:

- Transport Layer Protocol:** Any
- Application Layer Protocol:** MS-Of
- VLAN Tag ID:** MS-Office-365
- Outbound DSCP:** Traffic generated by office 365 applications and web services.
- Traffic Type:** Any

So for the example QoS scenario, you do not need to configure a new application, but you can rely on the AFE. When you configure a real-life QoS environment, you will most likely make use of the Application Groups.

You can use this process to create new applications, if the application you want is not in the AFE.

To learn about Application Groups and creating new applications, see [“Applications” on page 97](#).

Creating QoS profiles

The QoS profile is the building block that contains the QoS classes and rules for traffic going to a site. You can assign a single QoS profile to many sites and you can use it to configure inbound and outbound QoS.

For more information about profiles, see [“QoS profiles” on page 124](#).

For the example QoS scenario, you must create three QoS profiles. One profile for the sites New York, London and Paris, one for Frankfurt, and one for incoming MS Office 365 traffic that must be protected from incoming internet browsing traffic.

To create a QoS profile for New York, London, and Paris

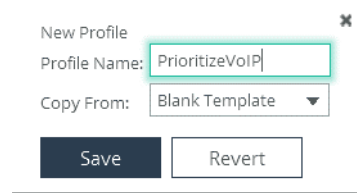
1. Choose Networking > Network Services: Quality of Service.

2. Select Add a QoS Profile.

The New Profile box opens and prompts for a profile name.

3. If you already had created a profile earlier, you are able to use this profile as a template for a new profile. For this example, choose Blank Template, enter a name (PrioritizeVoIP) and click **Save** ([Figure 7-5](#)).

Figure 7-5. Add a Profile Name



The screenshot shows a 'New Profile' dialog box. It has a title bar with a close button. Inside, there's a 'Profile Name' label followed by a text input field containing 'PrioritizeVoIP'. Below that is a 'Copy From' label followed by a dropdown menu showing 'Blank Template'. At the bottom, there are two buttons: 'Save' and 'Revert'.

The new profile appears in the QoS profiles table.

4. Click **Edit**.

The empty profile opens.

You must configure the class. An empty profile always starts with the *Root* class, which represents the bandwidth of the uplink of a site.

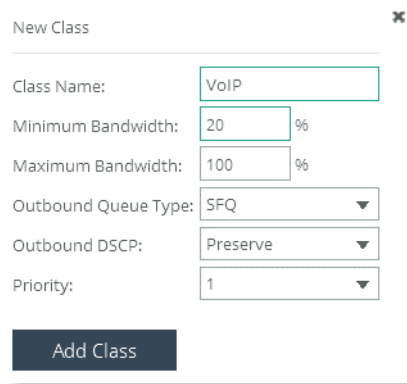
Sites and uplinks are configured later in the process and is described in [“Configuring topology” on page 156](#).

5. In the QoS classes section of the page, click **Edit**, and then select Add Class.

6. Configure a class for the VoIP traffic. Give the class a name, set the minimum bandwidth to 20%, and choose priority 1 for real-time traffic.

7. Click **Add Class** (Figure 7-6).

Figure 7-6. Add a VoIP class



The screenshot shows a 'New Class' dialog box with the following fields and values:

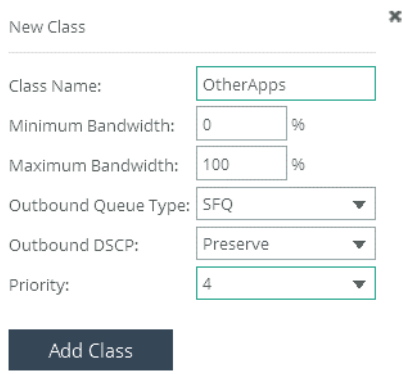
Field	Value
Class Name:	VoIP
Minimum Bandwidth:	20 %
Maximum Bandwidth:	100 %
Outbound Queue Type:	SFQ
Outbound DSCP:	Preserve
Priority:	1

An 'Add Class' button is located at the bottom left of the dialog box.

8. Create the class for other traffic. Click Add Class, give the class a name, and set the priority to 4, which is Normal priority.

9. Click **Add Class** (Figure 7-7).

Figure 7-7. Add OtherApps class



The screenshot shows a 'New Class' dialog box with the following fields and values:

Field	Value
Class Name:	OtherApps
Minimum Bandwidth:	0 %
Maximum Bandwidth:	100 %
Outbound Queue Type:	SFQ
Outbound DSCP:	Preserve
Priority:	4

An 'Add Class' button is located at the bottom left of the dialog box.

The configured classes are shown on the QoS Classes page ([Figure 7-8](#)).

Figure 7-8. QoS configured classes

The screenshot shows the 'QoS Classes' configuration page. On the left, a tree view shows 'Root' expanded. Two class configuration boxes are visible:

- Class Name: VoIP**
 - Min. % Max. %
 - Outbound Queue Type:
 - Outbound DSCP:
 - Priority:
- Class Name: OtherApps**
 - Min. % Max. %
 - Outbound Queue Type:
 - Outbound DSCP:
 - Priority:

Each class box has an 'add class' button to its right. At the bottom of the page are 'Save' and 'Revert' buttons.

10. Click **Save.**

Next, configure the QoS rules to direct the traffic into the classes.

11. In the QoS Rules section, select Add a Rule.

12. RTP-Audio from the Application or Application Group drop-down menu.

13. Select VoIP from the QoS Class drop-down menu ([Figure 7-9](#)).

Figure 7-9. Add new RTP-Audio rule

The screenshot shows the 'New QoS Rule' dialog box with the following fields:

- Application or Application Group:
- QoS Class:
- Outbound DSCP:

At the bottom are 'Save' and 'Revert' buttons.

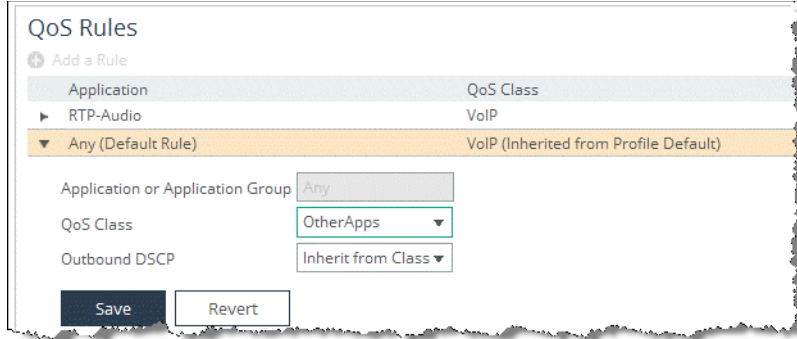
14. Click **Save.**

The new QoS rule is shown in the QoS Rules table.

Now you need to edit the Any or Default Rule to point it to the OtherApp class you created.

- Expand the Any rule and change the QoS class by selecting OtherApps from the drop-down menu (Figure 7-10).

Figure 7-10. Changing the QoS class



- Click **Save**.

The QoS Profile for New York, London, and Paris is now ready to use.

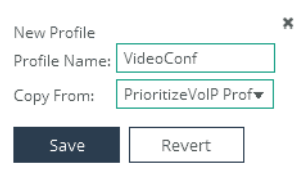
The Frankfurt site requires an additional level of hierarchy to accommodate for the video conferencing traffic and also has different bandwidth requirements. However, in this example, you can use the existing PrioritizeVoIP profile as a template and modified accordingly.

To create a QoS profile for Frankfurt

- To edit the existing PrioritizeVoIP profile, choose Networking > Network Services: Quality of Service.
- In the QoS Profiles section of the page, select Add a QoS Profile.
- Enter a name and select to copy from the PrioritizeVoIP Profile from the drop-down menu (Figure 7-11).

The VideoConf profile appears in the QoS Profiles table.

Figure 7-11. Creating a profile from an existing profile



- Click **Save**.

The VideoConf profile appears in the QoS Profiles table.

- Click **Edit**.

- Edit the QoS classes of the profile. Change the:

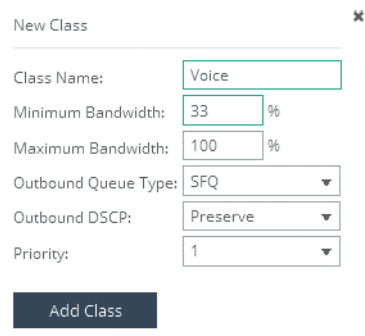
- name of the VoIP class to VideoConference.

- minimum bandwidth to 30% according to the example QoS scenario.
7. Before you can add a level of hierarchy to this class, edit all rules that point to it to point to the default class or delete the rule. A parent class cannot have a QoS rule assigned to it.

Next, add a class for the voice part of the video conferencing system.

8. Click **Add Class**, which is connected to the VideoConference class.
9. Enter a name (Voice) for the class, set the minimum bandwidth to 33% according to the example QoS scenario, and select priority 1, which is real-time (**Figure 7-12**).

Figure 7-12. Add Voice class



The screenshot shows a 'New Class' dialog box with the following fields and values:

Field	Value
Class Name:	Voice
Minimum Bandwidth:	33 %
Maximum Bandwidth:	100 %
Outbound Queue Type:	SFQ
Outbound DSCP:	Preserve
Priority:	1

An 'Add Class' button is located at the bottom left of the dialog box.

10. Click **Add Class**.
11. Repeat the above steps to configure a class for the video part of the video conferencing system.
Choose Video for the name of the class and configure the Priority to 2, which is interactive. You do not have to set a minimum bandwidth, because you already guaranteed bandwidth to the voice part.

Your class configuration now looks like (Figure 7-13).

Figure 7-13. Finished QoS class configuration



12. Click **Save**.

13. Configure the QoS rules to direct the traffic into the classes.

- Configure one rule for RTP-Audio to point to the Voice class.
- Configure one rule for RTP-Video to point to the Video class.

The QoS Rules table now looks like (Figure 7-14).

Figure 7-14. Finished QoS rules table

Add a Rule		
Application	QoS Class	Outbound DSCP
▶ RTP-Video	Video	Preserve
▶ RTP-Audio	Voice	Preserve
▶ Any (Default Rule)	OtherApps	Inherit from Class

The QoS profile for Frankfurt is now ready to use.

Setting up a QoS profile is the same for inbound and outbound QoS. Internet traffic usually generates more incoming than outgoing traffic. That is why in this example QoS scenario, the incoming MS Office 365 traffic must be protected from incoming internet browsing traffic.

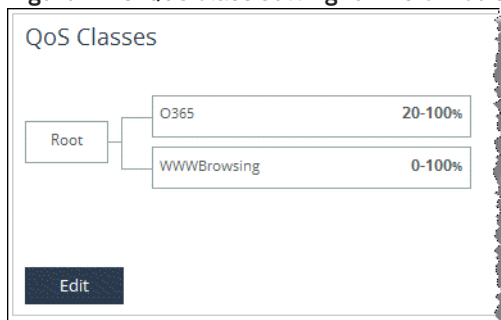
Note: This procedure assumes you have read the beginning of this section and know the intermediary steps.

To create a QoS profile and rules to protect MS Office 365

1. Create a class for the MS Office 365 traffic.
2. Set the minimum bandwidth to 20% and configure the priority to 3, which is business critical.
3. Create a class for all other internet traffic and set its priority to 5, which is low priority.

The QoS classes in the MS Office 365 profile look like (Figure 7-15).

Figure 7-15. QoS class setting for MS Office 365



4. To configure the QoS rules, select Add a Rule and select MS-Office-365.
5. Select O365 as QoS class.
6. Click **Save**.
7. Point the default rule to the WWWBrowsing class.

The QoS rules table looks like (Figure 7-16).

Figure 7-16. Configured QoS rules table

The screenshot shows a configuration window titled "QoS Rules". It contains a table with two columns: "Application" and "QoS Class". The first row shows "MS-Office-365" mapped to "O365". The second row shows "Any (Default Rule)" mapped to "WWWBrowsing". Above the table is a button labeled "Add a Rule".

Application	QoS Class
MS-Office-365	O365
Any (Default Rule)	WWWBrowsing

The QoS profiles needed for the example QoS scenario are now configured.

Configuring topology

The topology provides the SteelHead with a view onto the network it is connected to. The topology consists of the network, the sites, and the uplinks to the network for the sites. Additionally, the QoS profiles are linked to the sites.

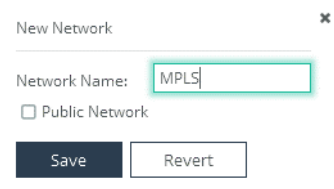
For more information about topology, see [“Topology” on page 91](#).

To configure the topology, choose Networking > Topology: Sites & networks. The My WAN network is configured by default.

To configure networks

1. Select Add a Network.
2. According to the example, specify MPLS as the name in the New Network box ([Figure 7-17](#)).
The Public Network check box is used with the secure transport feature for SCC. For more details, see the *SteelCentral Controller for SteelHead Deployment Guide*.

Figure 7-17. Network Name page



The image shows a 'New Network' dialog box with a close button (X) in the top right corner. It contains a 'Network Name' label followed by a text input field containing the text 'MPLS'. Below this is a checkbox labeled 'Public Network' which is currently unchecked. At the bottom of the dialog are two buttons: 'Save' and 'Revert'.

3. Click **Save**.

To configure the Local site

1. Click **Edit** for the Local (Local) site.
The local site is the physical location of the SteelHead you are connected to.
2. According to the example, rename the site San Francisco.
3. Specify the local subnets into the Subnets field.
4. According to this example (because the uplink connects to the MPLS network), select MPLS as uplink inpath0_0 from the name drop-down menu.
5. Enter the up and down bandwidth of the link to the MPLS network ([Figure 7-18](#)).
The configuration options for SteelHead Peers, Gateway IP, GRE Tunneling, and Probe are only used for the path selection feature and not needed for QoS.

You can leave the default values in the primary uplink because you do not need to configure it for QoS. Because the local site is the site in which the SteelHead you are configuring is physically located, you cannot assign any QoS profiles to the local site.

Figure 7-18. MPLS network

Edit an Existing Site

Basic Information

Site name:

Network Information

Subnets:
Subnets define how the SteelHead identifies this site. Separate with comma ","

SteelHead Peers:
Peers are used for path monitoring and GRE Tunneling. Separate with comma ","

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

[Add New Uplink](#)

inpath0_0

Uplink Name:

Network:

Gateway IP:

Inpath Interface: inpath0_0

☐ GRE Tunneling

Bandwidth Up: kbps

Bandwidth Down: kbps

[Probe Settings](#)

primary

Uplink Name:

Network:

Gateway IP:

Inpath Interface: primary

☐ GRE Tunneling

Bandwidth Up: kbps

Bandwidth Down: kbps

[Probe Settings](#)

6. Click **Save**.

To configure the remote sites

1. Select **Add a Site**.
2. According to the example, specify **New York**.
3. Specify the local subnets.
4. Assign the QoS profiles by selecting **PrioritizeVoIP** as inbound as well as outbound QoS profile.
5. Select **Add New Uplink**.
6. Select **MPLS** from the Network drop-down box and enter the up and down bandwidth.

According to the example QoS scenario, the bandwidth for the New York site is 20 Mbps for both.

Your site configuration looks like **Figure 7-19**.

Figure 7-19. Site Configuration page

Create a New Site

Basic Information

Site name

Network Information

Subnets Subnets define how the SteelHead identifies this site. Separate with comma.

SteelHead Peers Peers are used for path monitoring and GRE Tunneling. Separate with comma.

QoS Profiles

Inbound QoS Profile

Outbound QoS Profile

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

+ Add New Uplink

New Uplink

Uplink Name:

Network:

Bandwidth Up: kbps

Bandwidth Down: kbps

7. Click **Save**.

8. Repeat the same process for the London and Paris sites.

According to the example QoS scenario, the sites New York, London and Paris, can all use the PrioritizeVoIP QoS profile. Remember to configure the correct up and down bandwidth for the Paris site.

9. Configure the Frankfurt site.

Use the same procedure as above. The differences are the bandwidth and the QoS profile. Select 50 Mbps as up and down bandwidth and assign the VideoConf profile as inbound and outbound QoS profile.

10. Configure the DefaultSite.

According to the example QoS scenario, you want to configure MS Office 365 traffic to be protected from internet browsing. Internet traffic in general is not bound to a specific site, which is why you need to use the default site.

- Select the MSOffice365 profile as inbound QoS profile.

- Select Add New Uplink to the default site.
- Specify a name.
- Connect the uplink to the MPLS network.
- Configure the up and down bandwidth to 15 Mbps.

Your default site configuration looks like **Figure 7-20**.

Figure 7-20. Default site configuration

Edit an Existing Site

Basic Information

Site name

Network Information

Subnets
Subnets define how the SteelHead identifies this site.
Separate with comma ","

SteelHead Peers
Peers are used for path monitoring and GRE Tunneling.
Separate with comma ","

QoS Profiles

Inbound QoS Profile

Outbound QoS Profile

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

+ Add New Uplink

MPLS

Uplink Name:

Network:

Bandwidth Up: kbps

Bandwidth Down: kbps

Save Revert

11. Click Save.

Your final Sites & Networks page looks like [Figure 7-21](#).

Figure 7-21. Final Sites & Networks page

Sites & Networks
[Topology](#) > [Sites & Networks](#)
?

Sites & Networks

Site and Network configuration is used in QoS and Path Selection. Both features reference the Sites and Networks configured here in order to properly shape and direct traffic on your network.

Networks

Represents the WAN networks that sites use to communicate to each other such as MPLS, VSAT or Internet.

[+ Add a Network](#)

Network Name	Public	Securable	Sites
▶ My WAN	No	No	1
▶ MPLS	No	No	6

Sites

A collection of resources which that share one or more common WAN links, usually in one physical location.

[+ Add a Site](#)

Site Name	Uplinks	Outbound QoS	Inbound QoS	
San Francisco (Local)	2	N/A	N/A	Edit Site
DefaultSite	1	Default	Default	Edit Site
New York	1	PrioritizeVoIP	PrioritizeVoIP	Edit Site
London	1	PrioritizeVoIP	PrioritizeVoIP	Edit Site
Paris	1	PrioritizeVoIP	PrioritizeVoIP	Edit Site
Frankfurt	1	VideoConf	VideoConf	Edit Site

Enabling QoS on the SteelHead

We recommend as a best practice to first configure QoS on a SteelHead and then enable QoS as the final step. This order prevents unexpected network behavior while configuring QoS. However, to correctly classify traffic, the SteelHead must detect the three-way TCP handshake of the session carrying that traffic. Therefore, when you enable QoS, existing TCP/UDP session is not classified correctly and is classified to the default class.

You must make sure to enable QoS at a time when network usage is low, or during a maintenance window.

To enable QoS

1. Choose Networking > Network Services: Quality of Service.
2. Select Enable Outbound QoS Shaping and select Enable Inbound QoS Shaping (Figure 7-22).

Figure 7-22. Enable QoS shaping

Quality of Service Network Services > Quality of Service ?

Enable QoS

- ☒ Enable Outbound QoS Shaping
- ☒ Enable Inbound QoS Shaping
- ☐ Enable Outbound QoS Marking

Save **Revert**

Manage QoS Per Interface

Both inbound and outbound QoS can be enabled on a per-interface level.

Interface	Outbound QoS	Inbound QoS
primary	Disabled	Not Available
wan0_0	Enabled	Enabled

3. Click **Save**.

In the Manage QoS Per Interface section of the Quality of Service page, wan0_0 is enabled for inbound and outbound QoS by default.

In summary, the Networking > Network Services: Quality of Service pages shows you a summary of what you have been configuring to set up QoS on the SteelHead. You can use this page to quickly check what is configured for QoS and if it is configured correctly (Figure 7-23).

Figure 7-23. Quality of Service Page as a summary

QoS Profiles

A QoS profile is a self-contained set of QoS classes and rules. Each profile can be used to control communication when this SteelHead communicates with any number of sites.

+ Add a QoS Profile

Profile	Rules	Classes	
Default	1	7	Edit
MSOffice365	2	3	Edit
PrioritizeVoIP	3	5	Edit
VideoConf	2	3	Edit

QoS Remote Site Information

Configured bandwidths are indicated with an asterisk. All other bandwidths are estimated.

Site Name	Outbound QoS	Inbound QoS	Outbound BW (Kbps)	Inbound BW (Kbps)
DefaultSite	Default	MSOffice365	MPLS: 15000 primary: 15000	MPLS: 15000 primary: --
New York	PrioritizeVoIP	PrioritizeVoIP	MPLS: 20000 primary: 20000	MPLS: 20000 primary: --
London	PrioritizeVoIP	PrioritizeVoIP	MPLS: 20000 primary: 20000	MPLS: 20000 primary: --
Paris	PrioritizeVoIP	PrioritizeVoIP	MPLS: 20000 primary: 20000	MPLS: 20000 primary: --
Frankfurt	VideoConf	VideoConf	MPLS: 50000 primary: 50000	MPLS: 50000 primary: --

Configuring QoS marking on SteelHeads

You can mark incoming traffic to a LAN port of a SteelHead with a DSCP or an IP ToS value (for information about default behavior, see [“QoS marking default setting” on page 138](#)).

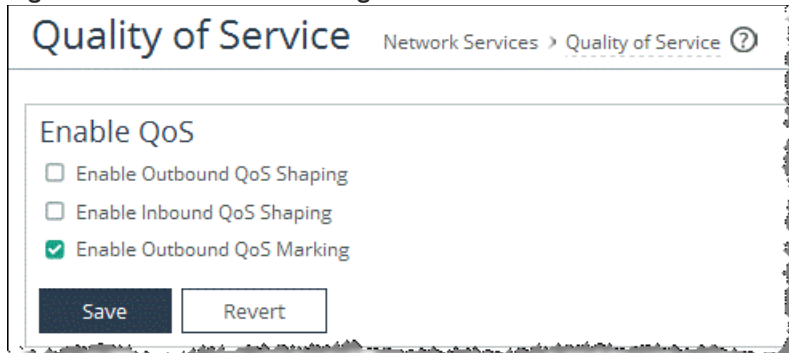
For more information about QoS marking settings, see the *SteelHead User Guide*.

Note: Prior to RiOS 7.0, the DSCP or IP TOS value on a server-side SteelHead was determined by the DSCP or IP ToS value of the client-side SteelHead. If you are running an earlier version of RiOS, see an earlier version of this guide for instructions on how to configure the DSCP or IP ToS value.

To enable QoS marking on a SteelHead

1. Choose Networking > Network Services: Quality of Service.
2. Select Enable Outbound QoS Marking ([Figure 7-24](#)).

Figure 7-24. Enable QoS marking



After you define the application, you can mark it with a DSCP or IP TOS value in a QoS profile. The application can either inherit the DSCP or IP TOS value from a QoS class or you can create a QoS rule to mark it with a DSCP or IP TOS value. For more information about how to define an application, see [“Defining an application” on page 98](#).

With RiOS 9.0 and later, marking traffic with a DSCP or IP TOS value is no longer part of the application definition. If you are using a release previous to RiOS 9.0, see earlier versions of the *SteelHead Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

To set DSCP or IP ToS value per class in a QoS profile

1. Choose Networking > Network Services: Quality of Service.
2. Edit an existing QoS profile or select Add a Profile to create a new QoS profile.

3. Create or edit the class for which you want to set a DSCP or IP ToS value (**Figure 7-25**).

Figure 7-25. Setting DSCP or IP ToS value per class in a QoS Profile

The screenshot shows the 'QoS Classes' configuration window. It features a tree view on the left with a 'Root' node. Two classes are listed: 'VoIP' and 'OtherApps'. Each class has a corresponding configuration panel on the right. The 'VoIP' class configuration includes: Class Name (VoIP), Min (20), Max (100), Outbound Queue Type (SFQ), Outbound DSCP (Preserve), and Priority (1). The 'OtherApps' class configuration includes: Class Name (OtherApps), Min (0), Max (100), Outbound Queue Type (SFQ), Outbound DSCP (Preserve), and Priority (1). A dropdown menu for the 'Priority' field is open, showing a list of values from 0 to 12 (AF12). At the bottom of the window, there are 'Save' and 'Revert' buttons, and a 'QoS Rules' section with an 'Add a Rule' button.

To set DSCP or IP ToS value per application in a QoS profile

1. Choose Networking > Network Services: Quality of Service.
2. Edit an existing QoS profile or select Add a Profile to create a new QoS profile.
3. Edit an existing QoS rule or select Add a Rule to create a new QoS rule.
4. Select the application or application group to mark with a DSCP or IP ToS value.

5. Select the DSCP or IP TOS value from the drop down list (Figure 7-26).

Figure 7-26. Setting a DSCP or IP ToS value

The screenshot shows the 'QoS Rules' configuration window. At the top, there is a table with two columns: 'Application' and 'QoS Class'. The first row is highlighted in orange and shows 'RTP' under 'Application' and 'Realtime' under 'QoS Class'. Below the table, there are three input fields: 'Application or Application Group' with 'RTP' entered, 'QoS Class' with 'Realtime' selected from a dropdown, and 'Outbound DSCP' with '34 (AF41)' selected from a dropdown. To the right of the 'Outbound DSCP' field, a list of DSCP values is displayed, including 'Inherit from Class', 'Preserve', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10 (AF11)', '11', and '12 (AF12)'. Below the input fields are 'Save' and 'Revert' buttons. At the bottom left, there is a '► Any (Default Rule)' button.

The DSCP or IP ToS value that you define in a QoS rule of a QoS profile takes precedence over the values defined in the QoS class of the same QoS profile.

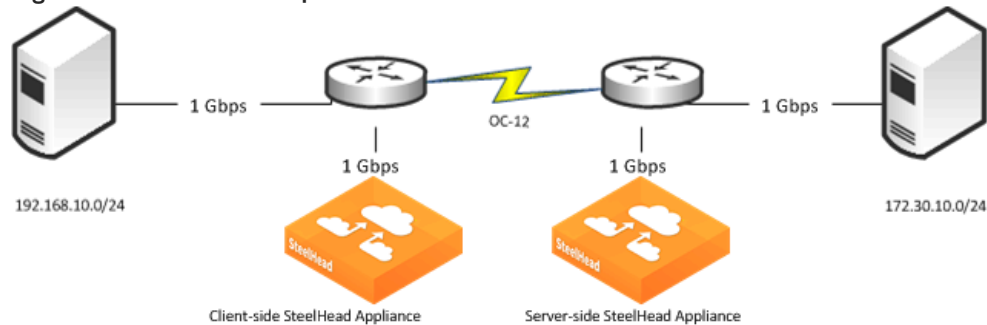
We recommend the following settings:

- For QoS marking only, set the DSCP or IP ToS value in the QoS rule of a QoS profile.
- For QoS marking and traffic shaping, set the DSCP or IP ToS value in the QoS class of a QoS profile.

Configuring QoS and MX-TCP

This section describes how to configure Riverbed QoS and MX-TCP on the SteelHeads. [Figure 7-27](#) shows an example in which the client expects to have a certain application—running over port 10566—to be classified into an MX-TCP QoS class.

Figure 7-27. MX-TCP example



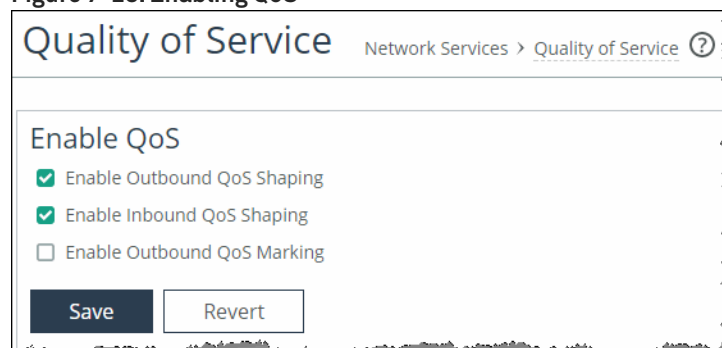
For additional information about MX-TCP, see [“MX-TCP” on page 123](#) and [“MX-TCP settings” on page 402](#).

To configure QoS and MX-TCP on the client-side SteelHead

1. Enable QoS on the SteelHead by completing the following steps:

- Choose Network > Network Services: Quality of Service.
- Select the Enable Outbound QoS Shaping and Enable Inbound QoS Shaping check boxes.
Inbound and Outbound QoS shaping is enabled by default on the wan0_0 interface. Some newer SteelHead models have different interface configurations and might require that you enable QoS per interface. See [“To enable QoS” on page 161](#).
- Optionally, select the Enable Outbound QoS Marking check box.
- Click **Save** ([Figure 7-28](#)).

Figure 7-28. Enabling QoS



- On the client-side and server-side SteelHead, choose Networking > App Definitions: Applications and create a custom application that is based on optimized traffic only, with header based rules, and without selecting any applications in the Application Layer Protocol field (Figure 7-29).

Figure 7-29. Custom MX-TCP application

New Application

Name: CustomMXTCP

Description: optimized traffic

Traffic Characteristics:

Local Subnet: 192.168.10.10/32 Port:

Remote Subnet: 172.30.16.10/32 Port: 10566

Transport Layer Protocol: Any

Application Layer Protocol: Any

VLAN Tag ID: Any

DSCP: Any

Traffic Type: Optimized

Application Properties:

Application Group: Custom Applications

Category: Collaboration

Business Criticality: Medium Criticality

Save

- Click **Save**.
- Choose Networking > Network Services: Quality of Service.
- Select Add a Profile.
- Specify a name for the profile (Figure 7-30).

Figure 7-30. MX-TCP QoS profile

New Profile

Profile Name: MXTCP

Copy From: Blank Template

Save

- Click **Save**.

8. Click **Edit** on the profile you created.
9. Under QoS Classes, click Edit, then add a Default class for any traffic not matching any rules.
10. Add a class for the MX-TCP traffic (Figure 7-31).

Make sure you select MX-TCP as the Outbound Queue Type. If the SteelHead has a bandwidth limit, then you must configure the minimum bandwidth within that limit. The MX-TCP class can use up to 99% of the bandwidth of the WAN class.

Figure 7-31. Configuring a default class

The screenshot shows a configuration window with a tree view on the left containing a 'Root' node. Two class configuration panels are visible:

- Class Name: default**
 - Min. **5** % Max. 100 %
 - Outbound Queue Type: SFQ
 - Outbound DSCP: Preserve
 - Priority: 1
- Class Name: MXTCP**
 - Min. **95** % Max. 100 %
 - Outbound Queue Type: MX-TCP
 - Outbound DSCP: Preserve
 - Priority: 1

At the bottom, there is an 'add class' button with a plus icon.

11. Click **Save**.
12. Select Add a Rule, and select the custom application you already created and assign it to the MX-TCP class (Figure 7-32).

Figure 7-32. MX-TCP New QoS Rule page

The 'New QoS Rule' dialog box contains the following fields:

- Application or Application Group: CustomMXTCP
- QoS Class: MXTCP
- Outbound DSCP: Preserve

A 'Save' button is located at the bottom left of the dialog.

13. Click **Save**.

14. Assign this Profile to a site by completing the following steps:

- Choose Networking > Topology: Sites & Networks.
- Click **Edit Site** in the Sites section.
- In the QoS Profiles section of the configuration page, open the Outbound QoS Profile drop-down menu and select the MX-TCP profile you created in **Step 5** of this procedure.
- Click **Save**.

Path Selection

This chapter describes path selection. Path selection is available in RiOS 8.5 and later. Path selection enables the SteelHead to redirect traffic to a predefined available WAN path for a given application in real-time, based on path availability. This chapter includes the following sections:

- “Overview of path selection” on page 170
- “Path selection implementation” on page 171
- “Configuring path selection” on page 179
- “Valid path selection deployment design examples” on page 180
- “Path selection and virtual in-path deployment” on page 192
- “Design validation” on page 192
- “Design considerations” on page 194

Note: To avoid repetitive configuration steps on single SteelHeads, we strongly recommend that you use the SCC 9.0 or later to configure path selection on your SteelHeads. SCC enables you to configure one time and to send the configuration out to multiple SteelHeads instead of connecting to a SteelHead, performing the configuration, and repeating the same configuration for all SteelHeads in the network.

Important: After upgrading from a 9.0.x or 9.1.x version to a later version of RiOS, the first policy push from SCC can cause pre-existing path selected connections to be blocked and/or QoS shaped connections to be misclassified. For more information, go to <https://supportkb.riverbed.com/support/index?page=content&id=S28250> (Riverbed Knowledge Base access required).

For more information about path selection, including configuration, see the *SteelHead User Guide* and the *Riverbed Command-Line Interface Reference Manual*.

This chapter requires you be familiar with “Topology” on page 91 and “Application Definitions” on page 97.

Note: If you are using a release previous to RiOS 9.0, the features described in this chapter are not applicable. For information about applications before RiOS 9.0, see earlier versions of the *SteelHead Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

Overview of path selection

Path selection is a RiOS technology commonly known as *intelligent dynamic WAN selection*. You can use path selection to define a specific WAN gateway for certain traffic flows, overriding the original destined WAN gateway.

WAN egress control is a transparent operation to the client, server, and any networking devices such as routers or switches. When you configure path selection, the SteelHead can alter the next hop gateway transparently for the client traffic. This granular path manipulation enables you to better use and more accurately control traffic flow across multiple WAN circuits.

You must know the following nomenclature prior to reading the information in this chapter:

- **Topology** - The topology combines a set of parameters that enables a SteelHead to build its view onto the network. The topology consists of network and site definitions, including information about how a site connects to a network. With the concept of a topology, a SteelHead automatically builds paths to remote sites.

For more information about topology IP segment parameters, see [“Topology” on page 91](#).

- **Uplinks** - An uplink is a logical medium that connects the site to a WAN network. A site can have a single or multiple uplinks to the same network and can connect to multiple networks. You can use multiple uplinks to the same network for redundancy. Uplinks serve as a logical connection that path selection uses to steer traffic. Local site uplinks are of upmost relevance for path selection.

For more information on uplinks, see [“Defining a site” on page 93](#).

- **Destination site** - A destination site provides the IP segment parameter for a path selection rule destined towards a preconfigured remote site destination. The *Any* parameter selection is a blend of all configured custom sites, and the *DefaultSite* selection indicates any destination excluding preconfigured custom sites.

For details, see [“Configuring the local site” on page 94](#) and [“Configuring the default site” on page 96](#).

- **Applications** - An application is a set of criteria to classify traffic. The definition of an application enables the SteelHead to ensure that the traffic belonging to this application is treated according how you have configured it. Technically an application definition means that the SteelHead can allocate the necessary bandwidth and priority for an application to ensure its optimal transport through the network. You can define an application on manually configured criteria or by using the AFE, which can recognize more than 1200 applications.

For more information about applications and the AFE, see [“Application Definitions” on page 97](#).

- **Relay** - Relay is traffic specified by the rule to be subject to path selection internal switching and follows the normal client default gateway (the original path) as intended by the end client or routed LAN.

Path selection implementation

This section includes the following topics:

- “Path selection workflow” on page 171
- “Example of a path selection implementation” on page 176
- “Identifying traffic flow candidates” on page 178

Path selection workflow

Path selection configuration is highly dependent on the network, site, and uplink configurations, defined in “Topology” on page 91. You must complete topology configuration according to your physical network design.

Note: Path selection configuration in RiOS 9.0 and later differs considerably from previous RiOS versions. As such, configuration migration from previous versions is neither compatible nor supported.

To avoid repetitive configuration steps on single SteelHeads, we strongly recommend that you use the SCC 9.0 or later to configure path selection on your SteelHeads. SCC enables you to configure one time and to send the configuration out to multiple SteelHeads instead of connecting to a SteelHead, performing the configuration, and repeating the same configuration for all SteelHeads in the network.

To configure path selection, you must complete the following tasks:

1. Configure the multiple different WAN networks in the environment on the Networking > Topology > Sites & Networks page. Even though this configuration is not required, we recommend that you complete this step to simplify the configuration.

The network topology configuration is more of a concept to bind different SteelHeads over a common logical connection. Path selection is typically deployed with WAN designs composed of two or more different circuits; each one of those circuits provides a distinct path to be used by path selection (example shown in Figure 8-1). There are three distinct circuits: MPLS, VPN, and internet.

Figure 8-1. Example network topology

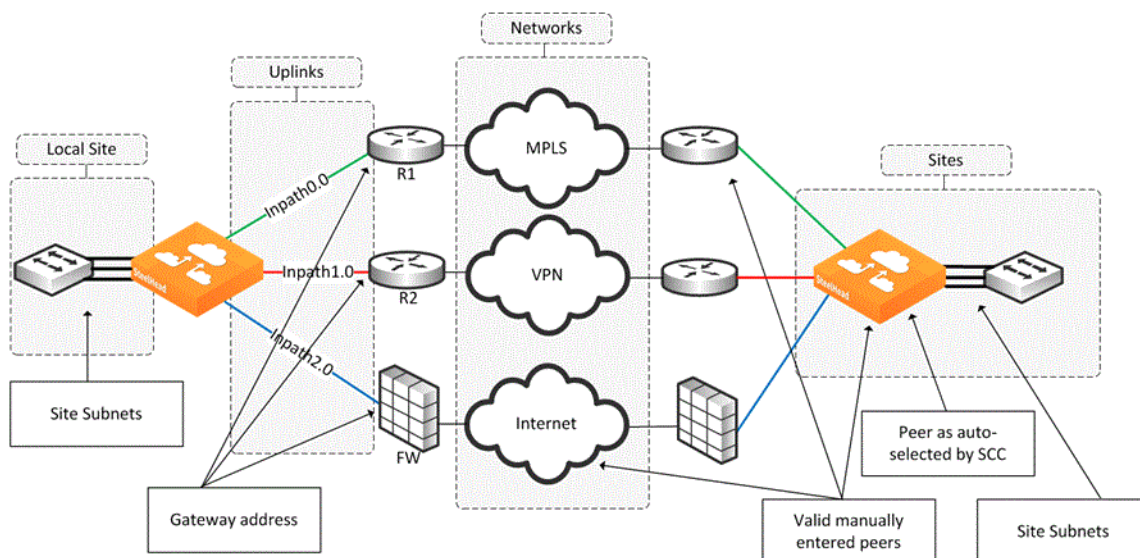


Figure 8-2 shows the SteelHead configuration for the Figure 8-1, in which each WAN path is labeled in the configuration.

Figure 8-2. WAN networks

Networks			
Represents the WAN networks that sites use to communicate to each other such as MPLS, VSAT or Internet.			
+ Add a Network			
Network Name	Public	Securable	Sites
▶ MPLS	No	No	2
▶ VPN	No	No	1
▶ Internet	Yes	No	1

2. Configure the sites on the Networking > Topology > Sites & Networks page (Figure 8-3).

Figure 8-3. Site configuration

Create a New Site

Basic Information

Site name

Network Information

Subnets

SteelHead Peers

Subnets define how the SteelHead identifies this site. Separate with comma ","

Peers are used for path monitoring and GRE Tunneling. Separate with comma ","

The Sites configuration is integral for the path selection feature. You must create any remote destination site you want to build a path to. Path selection takes into account the remote destination IP subnets that you configure as a parameter for the site. The IP subnet property is how a SteelHead is able to direct traffic towards a specific destination site because it can identify the destination IP address in the packet header.

A site configuration contains the SteelHead peers field property. SteelHead peers are distinct IP addresses you choose to poll, in order, to verify path availability. We strongly recommend that you use the remote SteelHead in-path IP address as a peer address when possible; for example, a remote peer SteelHead in-path IP is required for the firewall traversal GRE feature. You can enter additional addresses that are probed for path availability status. Each entered IP address is attempted as a separate independent path.

As part of the site configuration, a default site is preconfigured by default. You need the default site because it serves to catch traffic that is not part of any preconfigured sites: for example, internet-bound traffic. Depending on this traffic flow pattern, you must enter a value in the SteelHead Peer field. We generally recommend that you edit the existing default site (Figure 8-4) to use:

- the data center SteelHead IP address for internet-bound traffic that is backhauled through the data center.
- the local router gateway when the internet-bound traffic is exiting directly out of the branch.

Figure 8-4. Existing default site

Edit an Existing Site

Basic Information

Site name

Network Information

Subnets <small>Subnets define how the SteelHead identifies this site. Separate with comma ','</small>	<input type="text" value="0.0.0.0/0"/>	SteelHead Peers <small>Peers are used for path monitoring and GRE tunneling. Separate with comma ','</small>	<input type="text"/>
--	--	---	----------------------

3. Edit the existing local site to configure it for your design.

A local site is always created by default and cannot be deleted. However, you can rename the local site to reflect your network design.

The uplinks configuration is integral to configuring the local site. Uplinks dictate the egress path out of the SteelHead and hence are critical to the path selection configuration. You can rename uplinks to a more meaningful description, and you can tie them to a network that you have already defined.

The uplink name is recalled in the path selection page as a selection to direct traffic. Uplinks, which are configured by default, are hard tied to each in-path interface available to the SteelHead. Local site uplinks require that you configure a gateway IP address. By default, the gateway IP address is identical to the already configured in-path gateway IP address, but it is configurable if you want to change it. We recommend that you point the uplink gateway to the WAN-facing IP address in case the in-path gateway is configured towards the LAN.

The gateway IP address is a WAN-side IP address of the next hop device you want to direct the traffic to. You do not need to configure remote site uplinks for path selection. The GRE tunneling option is enabled for certain designs that require tunneling.

For more information about GRE tunneling, see [“Firewall path traversal deployment” on page 189](#).

In RiOS 9.0 and later, the SteelHead automatically probes through each uplink you configure at the local site. This probe is the mechanism by which the SteelHead automatically configures the path that is available. The SteelHead probes from each uplink towards each configured remote site that you configure in the Peer SteelHead IP address field parameter.

RiOS 9.2 introduces the optional values to the probe settings:

- **Probe backoff** - Sets an upper limit to the frequency of the probing in the absence of traffic on the network. This limit is beneficial in a hub-and-spoke networks in which spoke-to-spoke communication is not frequent; therefore, unnecessary probing is greatly reduced. On the SteelHead, you can change the value using the following CLI command:

```
topology network <name> probe_backoff <seconds>
```

where <name> is the network you want to administer, and <seconds> is the numeric value in seconds.

- **Probe bandwidth** - Sets an upper bandwidth limit to the probes, which are being generated by a specific uplink. This option helps to control the amount of traffic used by probing. This feature is beneficial on small links on which you want to exercise more control on the link. On the SteelHead, you can change the value using the following CLI command:

```
topology site {<site-name> | local | default-site} uplink <uplink-name> network <name>
bandwidth_up <kbps> bandwidth_down <kbps> [gateway <ip-address>] [probing_bw <kbps>]
```

where <site-name> is the site name, <uplink-name> is a specific uplink, and <kbps> is the bandwidth limit amount of the probing you configure in kilobits.

For more information, see the *SteelCentral Controller for SteelHead Deployment Guide*.

Figure 8-5. Example uplink configuration

Uplinks

An uplink represents a single connection this site should be an uplink to represent each WAN.

+ Add New Uplink

inpath0_0 ⓘ

Uplink Name:

Network:

Gateway IP:

Inpath Interface: inpath0_0

☐ GRE Tunneling ⓘ

Bandwidth Up: kbps

Bandwidth Down: kbps

▼ Probe Settings

DSCP:

Timeout: second(s)

Threshold:

4. Configure path selection as described in [“Configuring path selection” on page 179](#).

Path selection is a global function that influences all traffic traversing the SteelHead. You cannot configure path selection to only intercept traffic on certain LAN interfaces. Path selection is unlike QoS traffic enforcement in which you can select the desired interfaces to enforce traffic shaping. In RiOS 9.0 and later, path selection introduces the concept of site identification. For example, if you want to identify a certain application that is destined to a certain site, you can elect to take an action on the exit paths.

For example, [Figure 8-6](#) shows an applications named RDP. Depending on the original destination, the traffic can follow two different uplinks. If you want to send the RDP traffic towards remotebranch1, then the traffic is steered towards the VPN path. On the other hand, if you want the traffic to travel to RemoteBranch2, then the traffic is steered to the PTP uplink. If the PTP path is not available, then traffic is configured to be dropped.

Figure 8-6. Path Selection Rules page

Application	Destination Site	Actions	DSCP
▶ RDP	RemoteBranch1	1 st VPN_uplink 2 nd -- 3 rd -- 4 th Relay	Preserve -- --
▶ RDP	RemoteBranch2	1 st PTP_uplink 2 nd -- 3 rd -- 4 th Drop	Preserve -- --

RiOS 9.0 and later include the following destination concepts:

- **Default Site** - The Default Site contains the IP subnet property of the default site configured in the Topology section. For destination identification, the IP subnet property matches the 0.0.0.0/0 setting. You select the default-site as the destination for connections typically oriented towards unknown areas, such as internet-bound destinations.
- **Any** - The Any setting combines identifications of all known configured sites, including the Default-Site. Rather than configuring a separate identical path selection rule for every known site, choose the Any setting to match the destination address of every configured site. This setting ensures that the application configured, and any matching configured site or the default-site are steered onto the selected uplink. The Any destination concept is important to understand. The concept serves as a mean to reduce the configuration steps required, yet provide a common application steering design.

These settings are available when you add a rule on the Networking > Network Services: Path Selection page.

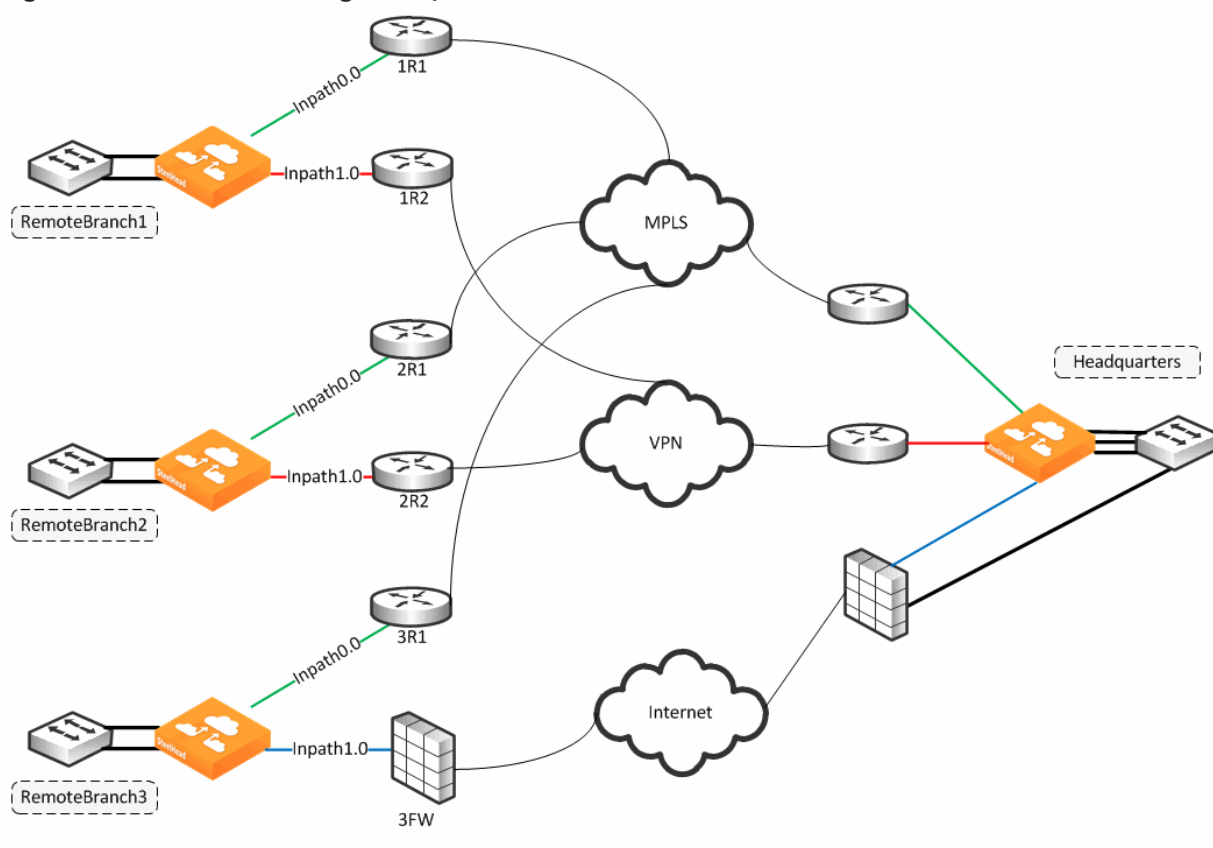
Note the order of the path selection rules in relation to the applications they refer to and how the site definitions come into play. As RiOS 9.0 introduces application groupings, certain configurations can consist of a path selection rule that you configure for a specific single application followed by another application group rule, which includes the previous application.

In the case of overlapping application, the order of the rule is the deciding factor as far as which rule is enacted for the path selection logic. In relation to the sites concept, RiOS identifies and selects the rule with the longest site prefix match first. Therefore, a rule specific for a site takes effect before the Any rule.

Example of a path selection implementation

Figure 8-7 shows an example of a WAN design in which path selection is implemented. The example shows multiple dual-homed sites with an MPLS WAN link provided by a carrier, labeled MPLS, and the second link is private VPN circuit. A third site, Remote Branch3, has a single connection back to the MPLS cloud and a secondary link through an internet-based firewall. All traffic, including public internet, is backhauled through the main headquarters site and egresses directly through the firewall connection.

Figure 8-7. Path selection design example



Each path is probed for availability based on the probe-setting schedule with default configuration of 2 seconds. The probe transmits an ICMP request from the configured in-path interface toward the probe destination IP, and after receipt of an ICMP response, the path is declared available for use. A path is determined down after the count of consecutive probe failures surpasses the configured probe threshold. The default threshold is three probe packets.

The SteelHead is looking for an ICMP response from the probe destination to determine path availability. Even if the ICMP response traverses unintended devices or WANs, the path is available as long as the configured in-path interface receives the ICMP response. This behavior can result in false positive path availability. The example shown in Figure 8-7 assumes that the MPLS path is configured with inpath0_0, along with a probe destination of 2R1. Even if the MPLS network fails, the path remains up as long as 2R1 continues to send ICMP responses to the SteelHead inpath0_0. Likewise, assume that the MPLS path, inpath0_0, is configured to send probes to 3R1 as the probe destination.

Given the bad scenario, in which the MPLS fails, 1R1 forwards the ICMP request to 2R1, across the VPN, through 3FW, and on to 3R1. 3R1 can respond, sending ICMP responses down and back over the VPN, reaching the SteelHead inpath0_0. In either case, the MPLS path availability remains connected, though the likely intention is that the path shows it is not connected when the MPLS WAN is down.

We recommend that you locate an address on the remote side of the path, and make sure devices in the path treat the probe as expected during a failure. This connection is best verified by conducting traceroute operations to verify the path flow traversal during outages. If the MPLS path has inpath0_0 configured with a probe destination of R31 and a next-hop gateway of 1R1, then configure 1R1 in which traffic to 3R1 can only go over the MPLS network. If the MPLS network fails, then configure 1R1, or another device, to drop the ICMP request probe from inpath0_0 to 3R1. An appropriate probe destination for a path can be a remote router loopback address or one of the remote SteelHead in-path interfaces.

In RiOS 8.6 and later, you can configure the SteelHead on a per-uplink basis to perform firewall traversal using GRE encapsulation to a remote SteelHead peer. The SteelHeads use the configured destination IP for the probe as the other endpoint when you set the tunnel mode setting in a path to GRE. Remote side GRE tunneling will require to be configured as well, with similar Path Selection rules, if you intend to maintain traffic symmetry.

Each path has its own independent IP address to probe, yet this address can be the identical one for each path. Therefore, each path can poll on the same probe destination. The ICMP request has to use whichever physical interface is selected for the path. In [Figure 8-7](#), the MPLS path egresses its packets using the inpath0_0 interface, hence all traffic uses the corresponding WAN0_0 interface. Meanwhile, the VPN path egresses its packing using in-path0_1, hence all traffic uses the corresponding WAN0_1 interface.

The next-hop gateway serves the following purposes for the path selection design because the gateway provides the new routing path for packets to travel through to their destination:

- Replaces the destination MAC address of packets with the MAC address of the alternate gateway. The gateway MAC address is learned by the SteelHead in-path interface. As part of steering packets, the destination MAC address of the packets is altered to match learned MAC address of the configured new next-hop gateway.

Path selection requires a Layer-2 connection between the SteelHead and the gateway; the connection between the SteelHead and the next-hop gateway cannot be a routed link. This action is referred to as *Layer-2 redirect by next hop MAC*.

- Switches the outbound interface from the original in-path interface to the desired primary path. The path selection solution is implemented completely transparently, regardless of existing routing metrics. The primary path selection path gateway accepts the steered packets and proceeds to forward them onto the corresponding WAN.
- The SteelHead takes no action in having to reconfigure the Layer-3 routing parameters of the routers in the network.

The SteelHead never takes an action to inject any routes or alter the routing instances. The traffic source whose packets are sent to the primary path selection gateway have no visibility into the changes the SteelHead applies. Therefore, the client (or server) continues to send any and all packets to the gateway address they are configured with. This action is referred to as *Layer-2 redirect by interface*.

WAN interface selection is based on identified traffic type and availability of the end-to-end path, depending on how you configure your SteelHead. Path selection remains functional even if you pause the optimization service or if the optimization service becomes unavailable. If the SteelHead fails completely, then path selection is no longer applicable and traffic proceeds as normal, following its default gateway.

Identifying traffic flow candidates

The critical step for path selection is to identify traffic flow and to associate these traffic flow candidates with a different, configured uplink. In this step, the AFE interacts with the path selection feature. Use the following methods to identify traffic that can benefit from path selection:

- The AFE can help you to identify the traffic and steer the traffic along a configured path. For more information about AFE, see [“Application Flow Engine” on page 102](#).

Some limitations exist when you use AFE in conjunction with path selection. AFE, or any deep packet inspection technique, requires examining several of the beginning packets of a connection before it can identify the traffic. That means that after the beginning packets have been identified, they can be sent on a path different than the one you chose. This midstream switching has implications in various environments involving firewalls and dual internet egress environments. For more details, see [“Firewall path traversal deployment” on page 189](#) and [“Design considerations” on page 194](#).

- You can also use IP header information as an alternate method for identifying traffic. IP header information identification consists of any of the following combinations:
 - Source IP
 - Destination IP
 - Source port address
 - Destination port address
 - DSCP mark
 - VLAN tag
 - Optimized/Unoptimized traffic
 - Layer-4 protocols (TCP, UDP, GRE, and so on)

For each path selection rule you configure, you can add a maximum of three different uplinks. The uplinks you choose cascade from one to the next, based on availability. RiOS 9.0 and later include the concept of application groups and enables you to reference multiple application types using a single path selection rule designating this application group.

In the brief duration when classification of traffic is not yet completed, traffic is treated according to a matching header-based rule or the site default rule. This period of time could have implications during path failure when such a rule does not specify the uplink preference and sent down the original path. In this case, new connections, or connections on which application identification is yet to complete, might never become established. We recommend that you specify an uplink preference for all path selection rules, both application and header-based rules, as well as specify the uplink preference for the site default rule.

You have the option to drop the traffic if no alternate path is available. Dropping traffic is useful when you prefer not to use bandwidth on an available path in case of failure on the primary selected path. If you choose not to override the original intended route, then traffic is relayed normally. The traffic continues to flow normally along the original intended path, following the default gateway.

Traffic identification and path steering is independent of optimized versus pass-through traffic. Path selection takes action on the configured traffic, no matter the optimization status of the traffic.

Path selection configuration is also independent of any QoS settings; this means that you can apply path selection rules with and without enabling QoS marking or shaping or both. Path selection uses its own independent rule sets apart from QoS; therefore, it does not increase the rule count number against the model limit as specified for QoS.

Remember that return traffic in path selection is not influenced or manipulated in any way to take the steered path from the sending SteelHead. You must install and configure a remote SteelHead with the appropriate path selection configuration, and steer the return traffic on the same path.

Configuring path selection

This section describes the basic steps for configuring path selection using the SteelHead Management Console. This section also includes a configuration example. For more information about the Management Console, see the *SteelHead User Guide*.

You can also use the Riverbed CLI to configure path selection. For more information about path selection commands, see the *Riverbed Command-Line Interface Reference Manual*.

We strongly recommend that you use the SCC 9.0 or later to configure path selection on your SteelHeads. SCC enables you to configure one time and to send the configuration out to multiple SteelHeads instead of connecting to a SteelHead, performing the configuration, and repeating the same configuration for all SteelHeads in the network. For details, see the *SteelCentral Controller for SteelHead User Guide*.

To perform the basic steps to configure path selection

1. Configure the topology as described in **“Topology” on page 91**.
 - Configure the uplinks for the local site and define the proper gateway IP address and peer IP address.
 - Configure remote sites and define the associated subnets.

You do not need to configure uplinks for the remote and default site.

2. Choose Network > Network Services: Path Selection and select Enable Path Selection.
3. Click **Save**.

4. Select Add a Rule (Figure 8-8).

Figure 8-8. Add a New Path page

New Path Selection Rule

Application/Application Group
Any

Destination Site
Any

Uplinks

Uplink 1: --	Outbound DSCP: Preserve
Uplink 2: --	Outbound DSCP: Preserve
Uplink 3: --	Outbound DSCP: Preserve

If all of the above uplinks are down: Relay

Save Revert

5. Specify the name of the application or application group name.
6. Select the destination.
7. Specify a preconfigured uplink to carry the traffic.
8. Change the DSCP mark per uplink path (optional).
9. Click **Save**.

You do not need to restart the SteelHead to enable path selection. At this point, path selection is enabled and you have configured the different available paths.

Valid path selection deployment design examples

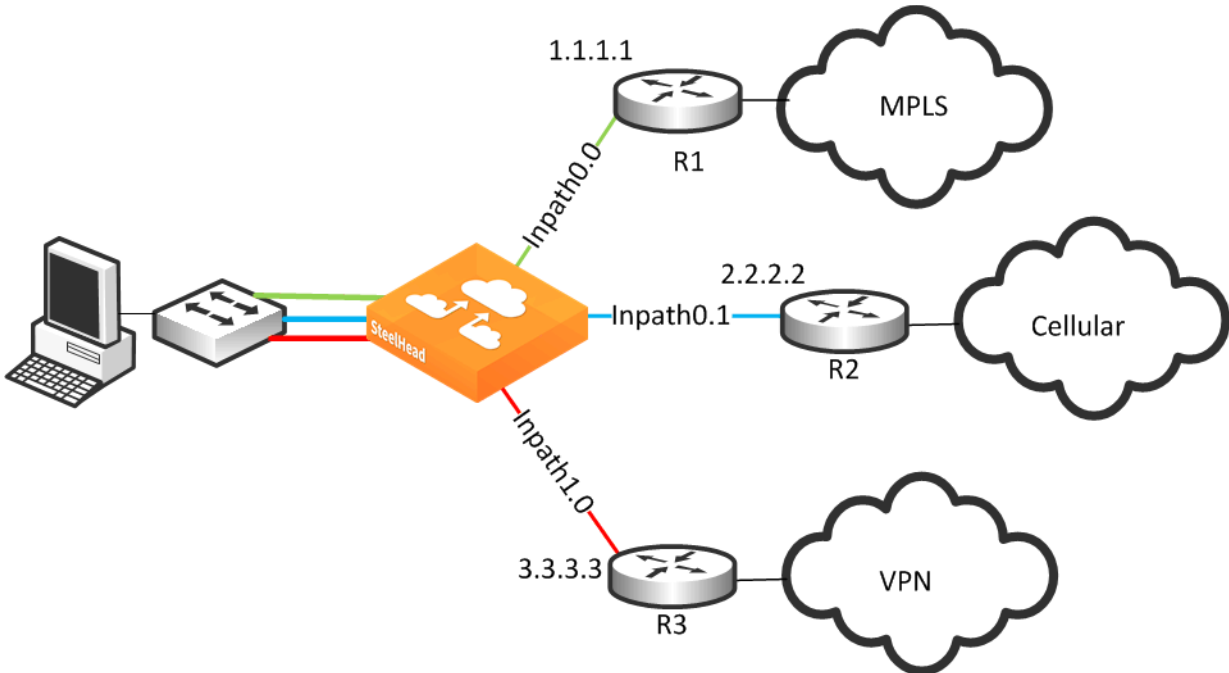
This section shows valid path selection deployment examples. The examples in this section show only one side of the WAN. You must assume that the remote side also has similar path selection capabilities and configurations for symmetric for return traffic. This section includes the following topics:

- “Basic multiple route path deployment” on page 181
- “Complex parallel path deployment” on page 184
- “Complex single in-path interface deployment” on page 186
- “Serial deployment” on page 189
- “Firewall path traversal deployment” on page 189

Basic multiple route path deployment

Figure 8-9 shows a SteelHead connected to three separate routers on three distinct in-path connections. Inpath0_0 is connected to Router 1, which is serving an MPLS connection. Inpath0_1 is connected to Router 2, which is serving a cellular-based WAN connection. Inpath1_0 is connected to Router 3, which is serving a VPN-based connection.

Figure 8-9. Basic multiple route path deployment



To configure path selection on the SteelHead as shown in Figure 8-9

1. From the Management Console, choose Networking > Topology: Sites & Networks.
2. Select Add a Network and create a separate network for each of the three different carriers, labeling each path with the proper name as shown in Figure 8-10.

Figure 8-10. Three path selection networks

Networks			
Represents the WAN networks that sites use to communicate to each other such as MPLS, VSAT or Internet.			
+ Add a Network			
Network Name	Public	Securable	Sites
▶ MPLS	No	No	1
▶ VPN	No	No	1
▶ Cellular	Yes	No	0

3. Scroll down and select Add a Site (Figure 8-11). The Subnets field is composed of the local subnets situated on the LAN side of the SteelHead. The SteelHead Peers field is a collection of remote IP addresses you want to probe to validate the path status.

Figure 8-11. Add a new site

Create a New Site

Basic Information

Site name

Network Information

Subnets <small>Subnets define how the SteelHead connects to the remote site. Separate with comma</small>	<input type="text" value="192.168.10.0/24, 192.168.11.0/24, 172.16.32.0/24"/>	SteelHead Peers <small>Peers are used for path selection. Separate with comma</small>	<input type="text" value="172.16.32.2, 192.168.1.2"/>
---	---	--	---

4. Repeat Step 3 for all remote sites in the path selection design.

To configure remote sites, you are required to enter the remote site name, IP segments residing at that remote site, and peer IP addresses to probe to determine path availability.

- Click **Edit Site** for the Local site, change the site name, and properly assign the Network label according to the proper in-path interface as shown in **Figure 8-12**.

Figure 8-12. Editing the Local site

Uplinks

An uplink represents a single connection this site has to a WAN. If this site has connections to multiple WANs, there should be an uplink to represent each WAN.

+ Add New Uplink

MPLS1

Uplink Name:

Network:

Gateway IP:

Inpath Interface:

☐ GRE Tunneling

Bandwidth Up: kbps

Bandwidth Down: kbps

► Probe Settings

MPLS

Uplink Name:

Network:

Gateway IP:

Inpath Interface:

☐ GRE Tunneling

Bandwidth Up: kbps

Bandwidth Down: kbps

► Probe Settings

MPLS2

Uplink Name:

Network:

Gateway IP:

Inpath Interface:

☐ GRE Tunneling

Bandwidth Up: kbps

Bandwidth Down: kbps

► Probe Settings

- Click **Save**.
- Choose Networking > Network Services: Path Selection.
- Select Enable Path Selection and click **Save**.
- Scroll down and select Add a Rule.
- Configure the application with the desired path order.

Select the specific application or entire Application Group. Next, select the destination site, and choose the uplink interface in order of preference.

Figure 8-13 shows an application group of type *General Internet* is selected to be steered through the VPN path, followed by MPLS if VPN is not available.

Figure 8-13. Configuring path selection for application general internet

New Path Selection Rule

Application/Application Group
General Internet

Destination Site
RemoteDataCenter

Uplinks

Uplink 1: VPN_uplink DSCP: Preserve

Uplink 2: MPLS_uplink DSCP: Preserve

Uplink 3: -- DSCP: Preserve

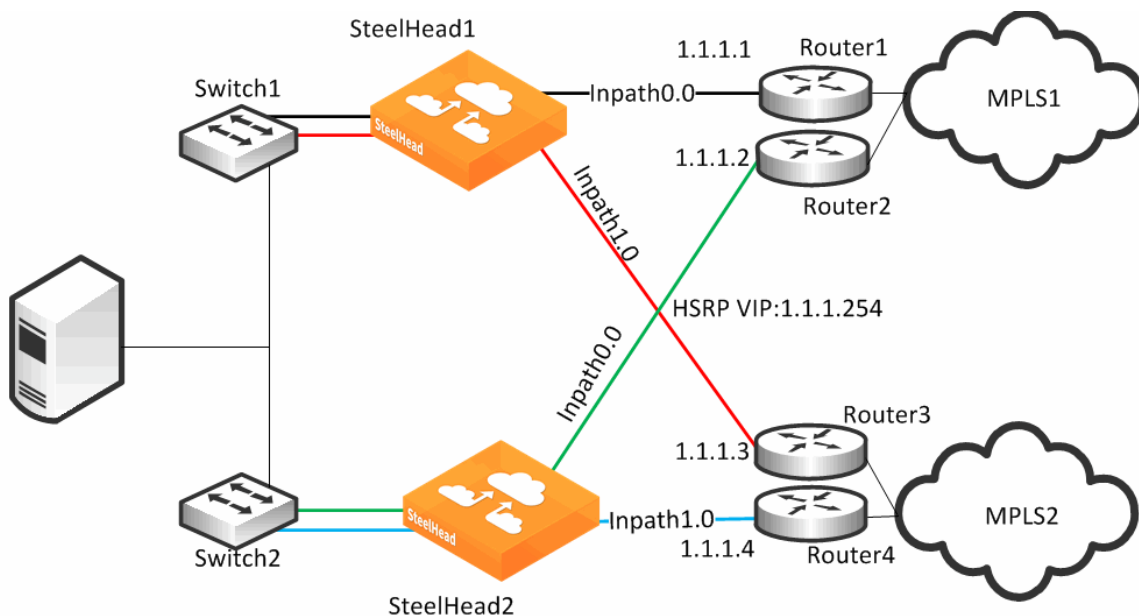
If all of the above uplinks are down: Relay

Save Revert

Complex parallel path deployment

Figure 8-14 shows a dual parallel SteelHead deployment on the WAN side with a four-way HSRP design. On the WAN side, Router 1 and 2 connect to the MPLS1 provider, and Router 3 and 4 connect to the MPLS2 provider. On the LAN side, each switch has a connection to both providers through a separate router.

Figure 8-14. Complex parallel path deployment



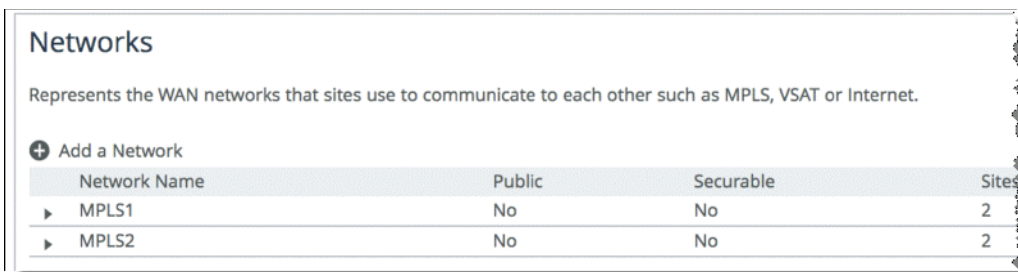
While each of the links in [Figure 8-14](#) can also be individual Layer-3 links, in this example there are two networks with HSRP configured on each network. When you define uplinks, use the real IP addresses of the routers as the gateway, not the virtual IP address. If you use a virtual IP address, you can cause the gateway to reside on the LAN side of the SteelHead in-path interface, resulting in unintended traffic flow. If your design is configured with a single HSRP group covering both routers, you must use the real IP address of the router as the gateway, not the virtual IP address.

Each SteelHead is connected to each router with the MPLS provider; therefore, both SteelHeads can make uniform path selection decisions, with traffic moving toward the same router and provider.

We recommend that you configure both SteelHeads with equivalent paths, uniform path selection rules and logic. [Figure 8-14](#) shows SteelHead configured with two networks: MPLS1 and MPLS2.

[Figure 8-15](#) shows the network topology configuration of SteelHead1 from [Figure 8-14](#).

Figure 8-15. SteelHead1 network topology configuration



Networks			
Represents the WAN networks that sites use to communicate to each other such as MPLS, VSAT or Internet.			
+ Add a Network			
Network Name	Public	Securable	Sites
▶ MPLS1	No	No	2
▶ MPLS2	No	No	2

Figure 8-16 shows SteelHead1 uplink configuration.

Figure 8-16. SteelHead1 uplink configuration

inpath0_0

Uplink Name: MPLS1

Network: MPLS1

Gateway IP: 1.1.1.1

Inpath Interface: inpath0_0

☐ GRE Tunneling

Bandwidth Up: 10000000 kbps

Bandwidth Down: 10000000 kbps

► Probe Settings

New Uplink

Uplink Name: MPLS2

Network: MPLS2

Gateway IP: 1.1.1.3

Inpath Interface: inpath0_1

☐ GRE Tunneling

Bandwidth Up: 1000000 kbps

Bandwidth Down: 1000000 kbps

► Probe Settings

SteelHead2 is configured with the equivalent paths, but with the respective gateway of Router2 IP address 1.1.1.2 and Router4 IP address 1.1.1.4.

If your design includes a single HSRP group covering both routers, configure SteelHead2 so that it is identical to that shown in Figure 8-16, in which the path gateway references the real IP interface and not the virtual IP.

Complex single in-path interface deployment

Figure 8-17 shows the SteelHead connected through a single in-path interface connection, but the WAN side is composed of multiple WAN routers, each to their own separate provider. The LAN side of the routers all share the same IP segment. Because they share the same IP segment, achieving path selection configuration in this setup is valid, because the SteelHead can be configured with different gateway addresses traversing the same in-path interface. You must configure a router gateway redundancy mechanism, such as HSRP or VRRP. This configuration needs a redundancy mechanism, because the SteelHead does not act as the default gateway for the clients.

The redundancy mechanism is completed using multiple uplinks for a single in-path.

Figure 8-17. Complex single in-path interface deployment

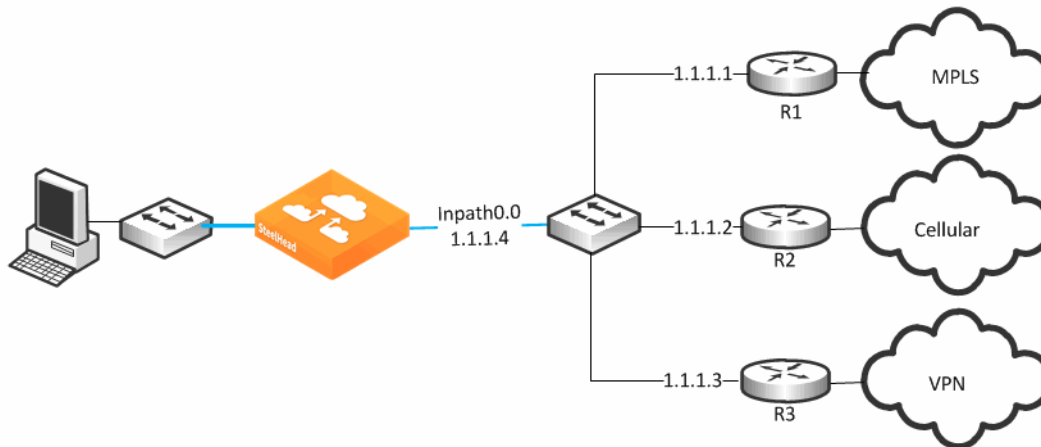


Figure 8-18 shows the network topology configuration for Figure 8-17.

Figure 8-18. SteelHead1 network topology configuration

Networks		
Represents the WAN networks that sites use to communicate to each other such as MPLS, VSAT or Internet.		
+ Add a Network		
Network Name	Public	Securable
▶ VPN	No	No
▶ MPLS	No	No
▶ Cellular	No	No

Figure 8-19 represents the uplink configuration for the deployment shown in Figure 8-17. The Local Site uplink configuration reflects additional uplinks each associated with a separate network in which each uplink shares the same inpath0_0 as the egress interface, but the gateway IP differs for each network desired.

Figure 8-19. Uplink configuration

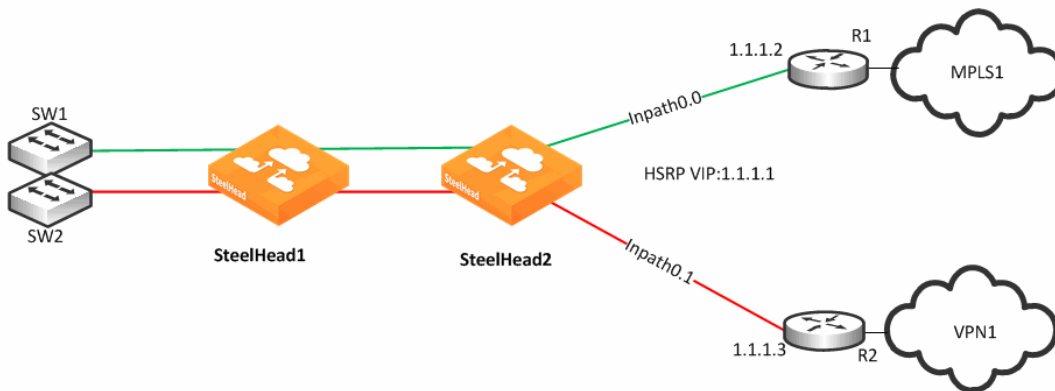
The screenshot displays three configuration windows for uplinks. The first window, titled 'inpath0_0', shows settings for a VPN uplink: Uplink Name is 'VPN', Network is 'VPN', Gateway IP is '1.1.1.3', Inpath Interface is 'inpath0_0', GRE Tunneling is unchecked, Bandwidth Up is '10000000' kbps, and Bandwidth Down is '10000000' kbps. Below these are 'Probe Settings' and a right-pointing arrow. The second window, titled 'New Uplink', shows settings for a Cellular uplink: Uplink Name is 'Cellular', Network is 'Cellular', Gateway IP is '1.1.1.2', Inpath Interface is 'inpath0_0', GRE Tunneling is unchecked, Bandwidth Up is '1000000' kbps, and Bandwidth Down is '1000000' kbps. It also includes 'Probe Settings' and a right-pointing arrow. The third window, also titled 'New Uplink', shows settings for an MPLS uplink: Uplink Name is 'MPLS', Network is 'MPLS', Gateway IP is '1.1.1.1', Inpath Interface is 'inpath0_0', GRE Tunneling is unchecked, Bandwidth Up is '1000000' kbps, and Bandwidth Down is '1000000' kbps. It includes 'Probe Settings' and a right-pointing arrow.

Note: A complex single in-path interface deployment is valid for path selection when all the routers are on the same subnet as the SteelHead single in-path interface. If the in-path interface is on an 802.1Q trunk, you cannot use path selection to direct traffic to different routers on different VLANs. The switch shown in Figure 8-17 is a Layer-2 switch; therefore, path selection can make the decision to send traffic to the appropriate router MAC address.

Serial deployment

Figure 8-20 shows a dual serial SteelHead deployment. SteelHead 1 is the client SteelHead, and SteelHead 2 is referred to as the *middle file engine* (MFE). On the WAN, Router 1 is connected to the MPLS provider, and Router 2 connects to the customer internal network using a VPN connection. In this example, we recommend that you use the real IP address of the router as the path gateway instead of the virtual IP provided by HSRP and VRRP.

Figure 8-20. Serial deployment



You can use path selection in a serial deployment if:

- SteelHeads have identical path selection configuration.
- correct addressing is in use. You must configure SteelHead2 to relay the inner channel of SteelHead1.
- you are using Full Transparency. You must use the **path-selection settings bypass non-local-trpy enable** command on SteelHead2.

Firewall path traversal deployment

This section describes how to configure firewall path traversal deployment. This section contains the following topics:

- [“MTU and MSS adjustment when using firewall path traversal” on page 190](#)
- [“Firewall path traversal deployment example” on page 191](#)

Stateful firewall devices typically provide security services including:

- tracking the TCP connection state.
- blocking a sequence of packets.

Stateful security devices add a level of complexity to path selection environments when the SteelHead attempts to make any path changes to a connection midstream. The most common examples of midstream switching are:

- failure of a higher-priority path, failing to firewall path.
- recovery of path, resuming traffic to a firewall path.

- using AFE for identification because the first packets of a connection are not recognized yet and can traverse a default path.

When a path changes midstream, the stateful firewall device is likely to see only some or none of the packets necessary to keep state and sequence numbers. When receiving packets are perceived to be out of order or belonging to a connection with inaccurate state information, stateful firewalls generally drop these packets.

Beginning with RiOS 8.6, we recommend that you use the firewall path traversal capability to leverage GRE tunneling over paths traversing a stateful firewall. When you use standard GRE between SteelHeads, connections can be switched midstream because the firewall only detects the encapsulated packets. You can configure GRE tunneling per uplink, and when enabled, the SteelHead attempts to encapsulate packets to the remote SteelHead at the configured remote peer IP address. The peer IP address needs to be an in-path IP address of a remote SteelHead. You can use multiple uplinks using GRE tunneling between the same SteelHeads, and the original packet or QoS-configured DSCP values are reflected in the GRE packets.

Note: There is a loss of visibility on the firewall when you use GRE encapsulation. You also might need additional configuration on the firewall to allow the GRE packets between SteelHeads.

MTU and MSS adjustment when using firewall path traversal

When you use GRE tunneling, consider that there is an additional 24-byte overhead added to packets. This overhead can cause fragmentation of large packets, because the extra added bytes cause the packet to exceed the maximum transmission unit (MTU) configured in the network. Fragmentation has the negative effect of sending inefficiently sized packets and dropping packets that might have the *do not fragment* option set.

You can prevent fragmentation by adjusting the maximum TCP payload, or MSS value, to account for the overhead added by GRE. When you configure a path with the tunnel mode set to GRE, the SteelHead measures to reduce potential fragmentation for TCP traffic.

This automatically applied MSS value ensures that in most environments TCP packets are not fragmented, even with the additional GRE overhead. In an optimized case, the client and server connections with the SteelHead are not impacted by the MSS adjustment procedure. For pass-through TCP traffic, the SteelHead adjust the MSS value to make room for a GRE header. To turn off this automatic MSS adjustment, use the **no path-selection settings tunnel adjust-mss enable** command.

For more information about MTU, see [“MTU sizing” on page 476](#).

Firewall path traversal deployment example

Figure 8-21 shows a dual-parallel installation of SteelHeads in a dual-homed WAN scenario. In this example, a firewall is installed on the edge of the internet path. The SteelHead has visibility into both MPLS and VPN paths.

Figure 8-21. Firewall path deployment

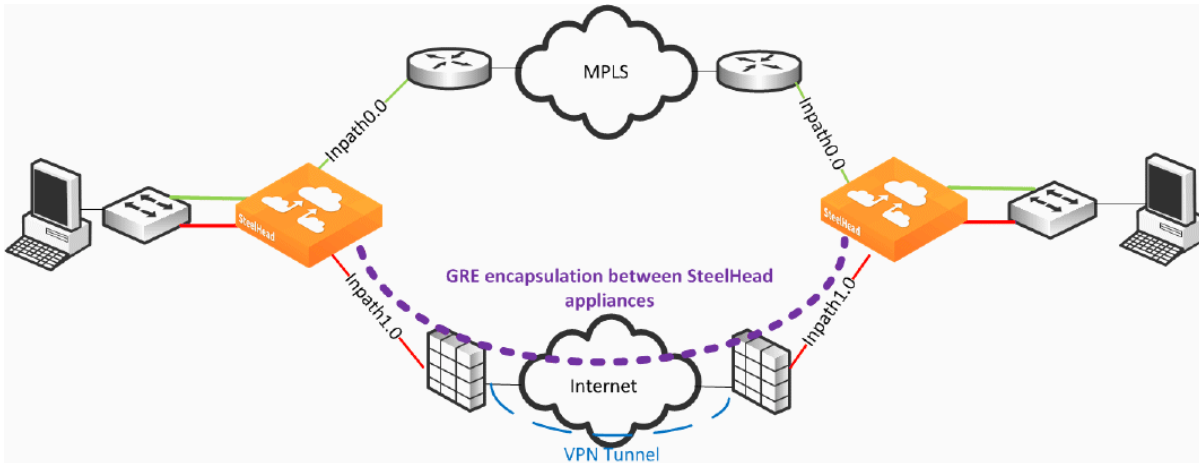


Figure 8-21 defines two uplinks: the green uplink over MPLS and the red uplink traversing stateful firewall devices. The firewall path is configured for GRE tunneling between the SteelHeads. When you configure a path for GRE encapsulation, it affects only the decision of the local SteelHead in that the opposing SteelHead must also have similar path configuration for return traffic to be encapsulated.

The SteelHeads are not providing the VPN functionality over the internet but sending traffic over the VPN tunnel provided by the existing firewalls in the path. Remember that this SteelHead configuration for path traversal over firewalls uses standard GRE encapsulation, which is not a secure method of traversing the internet.

In addition, the firewalls provide other necessary functions, such as NAT, and we do not recommend that you use the SteelHead GRE capability instead of direct VPN functionality.

GRE tunneling configuration is enabled during the uplink setup (Figure 8-22). You must denote the remote SteelHead in-path IP address in the peer setup field in order to terminate the GRE tunnel.

Figure 8-22. GRE tunneling

inpath1_0 ⓘ

Uplink Name: VPN

Network: Internet ▼

Gateway IP: 2.2.2.2

Inpath Interface: inpath1_0

☒ GRE Tunneling ⓘ

Bandwidth Up: 1500 kbps

Bandwidth Down: 1500 kbps

► Probe Settings

Path selection and virtual in-path deployment

We recommend that you not use virtual in-path deployments for path selection, but always use physical in-path deployments. Virtual in-path deployments often have caveats that limit path selection effectiveness, including but not limited to the following:

- Typically, only traffic that is optimized is redirected to the SteelHead; therefore, the SteelHead is limited to identifying and acting on only that subset of total traffic. Although you can configure devices to redirect all traffic, this configuration is often undesirable due to adding increased load and complexity.
- Additional routing devices often exist after the SteelHead makes the path selection decision.

For example, consider an environment with dual Layer-3 switches and dual routers connected to different service providers. Policy-based routing (PBR) is configured on the Layer-3 switches, and the SteelHeads make the path selection decision about which Layer-3 switch to send traffic to. The Layer-3 switches then make an independent routing decision to send traffic to a router, and therefore provider, rendering the SteelHead path selection decision meaningless.

Considering additional routing devices is important in physical in-path deployments, but holds additional weight in virtual in-path deployments because of the added restriction of only certain devices being capable of redirection. For example, many firewall devices have limited functionality when supporting virtual in-path mechanisms.

- Path selection next-hop functionality is not supported with WCCP. The SteelHead cannot choose to redirect packets using a different in-path interface or to send traffic to a configured gateway. Only limited functionality is available, enabling the SteelHead to mark packets with DCSP with different criteria depending on path availability.

Design validation

In RiOS 9.0 and later, you can use CLI commands and SteelHead Management Console report pages to verify path selection operations and to validate your path selection configuration.

For details, see the *Riverbed Command-Line Interface Reference Manual* and the *SteelHead User Guide*.

You can use the **show** commands to verify path selection settings and configuration:

```
CFE # show path-selection ?
channels          Display channel states
interface         Name of the interface
rules            Display configured path-selection rules
settings         Path Selection settings
status           Display feature status
CFE # show topology uplinks path-selection stats
```

Uplink	Bytes	Probe Requests	Probe Response	Relay Mismatch	Probe Requests Ricochet Dropped
VPN_uplink	0	0		0	0
MPLS_uplink	364042549	637013		0	0
PTP_uplink	0	0		0	0


```
show topology uplink <uplinkname> site <Site name> path-selection state
```

```
CFE # sh topology uplink MPLS_uplink site Default-Site path-selection state
Uplink:      MPLS_uplink
Network:     MPLS
Site:        Default-Site
VLAN:        None
Source Mac:  00:50:56:b8:1f:eb
Next Hop Mac: 00:01:e8:8b:d1:7a

Peer IP:     10.33.249.65(*)
Status:      Reachable
Probe Sequence: 61960
Encap port:  0
```

You can validate your design from the SteelHead Management Console with these reports:

- **Reports > Networking: Current Connections** - shows details per connection (Figure 8-23).
- **Networking > Networking Services: Path Selection (Uplink Status)** - shows you details per path (Figure).

Figure 8-23. Current Connections report

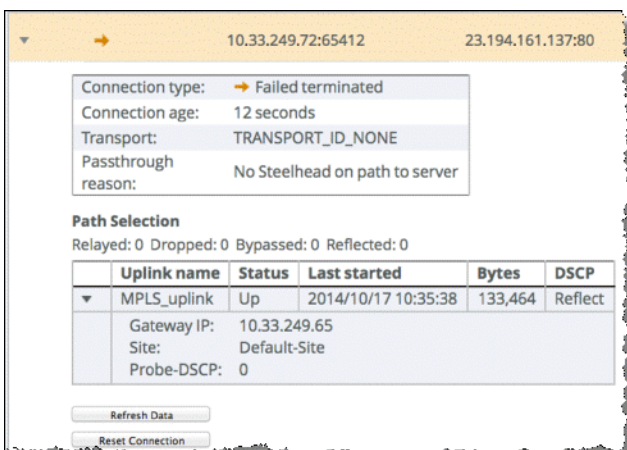


Figure 8-24. Path Selection report

Uplink Status			
Uplink Name	Uplink Status	Bytes Sent	
▼ MPLS_uplink	Healthy	56877158	
Remote Site	Peer Availability	Bytes Sent	
Default-Site	Never Reached: 10.33.248.225	0	
RemoteBranch1	Never Reached: 172.16.32.2, 192.168.1.2	0	
RemoteBranch2	Never Reached: 1.1.1.1	0	
RemoteDataCenter	Never Reached: 2.2.2.2	0	
▶ VPN_uplink	Healthy	0	
▶ PTP_uplink	Critical	0	

Design considerations

Consider the following guidelines when you use path selection:

- Path selection does not require dual-ended SteelHead deployments, but we strongly recommend that you maintain return-traffic symmetry.
- You cannot use AFE for internet-bound applications to select paths with different internet egress points. Using AFE implies that packets prior to identification traverse one path, but that after identification, the connection can switch midstream to a different path. If these two paths use different egress points to the internet, the packets on each path use different NAT public internet IP addresses and appear as two different sources to the internet server. Multiple internet egress can exist in these scenarios:
 - **Direct-to-internet at the branch office and internet at the data center** - You cannot use AFE to decide that some internet applications exit directly from the branch office and others from the data center.

For example, the default path that directly reaches the internet is at the branch, but you configure AFE for Facebook traffic to traverse a path to the data center. The beginning packets of the connection exit from the branch with a externally translated address from the branch internet provider. After identified, path selection switches midstream to the path to the data center, where the traffic is translated to a different internet address.
 - **Dual data centers, each with internet egress** - You cannot use AFE to determine what path internet-bound applications traverse.
- Be mindful of WAN-side routing, because it always takes precedence over path selection. Routers on the WAN side of a SteelHead can always override and reroute traffic according to their configuration. Be aware of upstream router configuration, so you that avoid unintended traffic redirection. Placing the SteelHead closer to the edge of a WAN helps to avoid this scenario. Some examples of this scenario include but are not limited to:
 - **WAN-aggregation layer** - All routers consolidate into a pair of Layer-3 switches. Path selection must occur on the WAN side of this layer. This configuration is more common in data centers.
 - **WAN-side router with multiple circuits** - The router decides on which circuit to send traffic. You cannot use path selection effectively in this scenario.
- **Transit networks** - Transit site traffic is defined as traffic that is not sourced or destined locally. In a topology where some of the sites do not have SteelHeads, behavior can occur where path selection rules are applied asymmetrically, which can lead to asymmetrical GRE-encapsulated traffic. This behavior can cause issues with firewalls such as dropped connections.

To push general path selection rules but selectively turn off path selection for transit site traffic, enter this command:

```
path-selection-transit-bypass enable
```

When this command is enabled and transit traffic is bypassed, no path selection matching of rules is applied to transit traffic, which results in traffic being relayed with no failover. Path selection rules are applied to local site traffic even if this command is enabled.

For details, see the *Riverbed Command-Line Interface Reference Manual*.

- Path selection is not effective in any environment in which independent routing decisions are made at the site after the SteelHead path selection decision has already occurred.

- Path selection in virtual in-path environments have additional considerations. For details, see [“Path selection and virtual in-path deployment” on page 192](#).
- Subnet side rules exclude subnets from changing paths.
- The SteelHead does not apply path selection configuration for traffic destined onto the same IP segment as the in-path interface. This is useful for routing updates if you have deployed the SteelHead in the direct path of that traffic.
- The SteelHead never takes on the role of the router or of a default gateway. Because the path selection solution is transparent, you do not have to make network design changes to accommodate path selection design.
- The primary and auxiliary interfaces of a SteelHead do not support path selection.
- Path selection is compatible with all virtual and physical SteelHead models running RiOS 8.5 or later.
- You must disable RSP to enable path selection. Current virtualization capabilities, including VSP on SteelHead EX 2.0 and later, are compatible with path selection.
- A SteelHead with path selection enabled has no enforcement on the return path.

If you want to influence the return path of traffic and override the original traffic path, you must deploy a SteelHead near the return traffic WAN junction point. Traffic returning on a different path is commonly known as *asymmetric routing*. Typically, networks are not designed in this way; however, if this traffic pattern exists, it might not be completely detrimental, because the SteelHead can rely on existing features and complete the optimization.

For more information about asymmetric routing, see the *SteelHead User Guide*.

- A single SteelHead can maintain optimization even if traffic is received on a different in-path interface from the original sending in-path interface. Because the SteelHead shares internal flow table with itself, it can complete the optimization process with no asymmetric alarm generation.

The following are not supported by path selection:

- Packet-mode optimization
- IPv6 optimization
- WCCP designs with Layer-2 redirection
- Designs requiring specific LAN-side redirection
- Layer-2 WAN
- Single-ended SCPS connections
- Maintaining VLAN transparency

For example, network designs in which the in-path interface is sitting on a VLAN trunk connection and you want to switch a flow onto another VLAN, results in discarded packets, because the VLAN ID field is not reflected upon steering.

Path Selection using GRE encapsulation has the following additional restrictions:

- Virtual in-path deployment is not supported.
- Inbound QoS is not applicable to inner or encapsulated incoming traffic.

- Simplified routing does not learn from tunneled packets. If the default gateway is pointed to the WAN, make sure that you have configured the proper static routes for networks that reside in the LAN.
- Flow export or reports reliant on flow show GRE traffic on the WAN interface. Visibility tools that coalesce or stitch LAN and WAN flows together can be affected adversely.
- The downstream SteelHeads in serial deployments cannot intercept and take over new TCP connections when an upstream SteelHead sends GRE traffic. Even in the event of admission control, a SteelHead continues to perform path selection and tunneling, preventing proper connection spillover to a downstream SteelHead.

Physical in-path deployments

This chapter describes a physical in-path SteelHead deployment. This chapter includes the following sections:

- [“Overview of in-path deployment” on page 197](#)
- [“Logical in-path interface” on page 198](#)
- [“Failure modes” on page 200](#)
- [“Configuring link state propagation” on page 203](#)
- [“EtherChannel” on page 203](#)
- [“Cabling and duplex” on page 205](#)
- [“Physical in-path deployment configuration examples” on page 208](#)
- [“In-path redundancy and clustering examples” on page 213](#)
- [“Configuring simplified routing” on page 219](#)
- [“Multiple WAN router deployments” on page 221](#)
- [“802.1Q trunk deployments” on page 236](#)
- [“Layer-2 WAN deployments” on page 239](#)
- [“VLAN bridging deployments” on page 241](#)

This chapter requires that you be familiar with:

- Hot Standby Router Protocol (HSRP)
- Virtual Router Redundancy Protocol (VRRP)
- Gateway Load Balancing Protocol (GLBP)

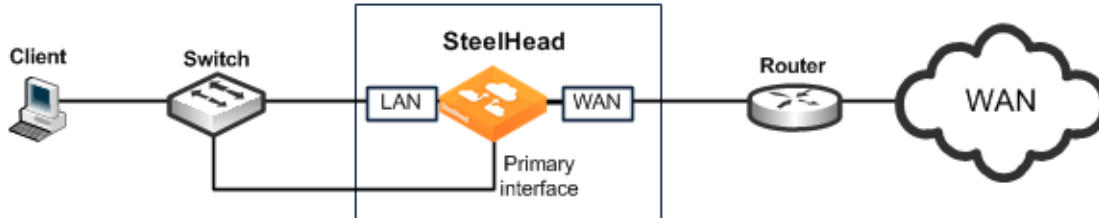
Overview of in-path deployment

In a physical in-path SteelHead deployment, a SteelHead LAN interface connects to a LAN-side device (typically a switch), and a corresponding SteelHead WAN interface connects to a WAN connecting device (typically a router). This allows the SteelHead to detect all traffic flowing to and from the WAN and to perform optimization.

Depending on the SteelHead model and its hardware configuration, you can use multiple pairs of WAN and LAN interfaces simultaneously, and you can connect them to multiple switches and routers.

Figure 9-1 shows the simplest type of physical in-path SteelHead deployment.

Figure 9-1. Single subnet, physical in-path deployment



Most SteelHead deployments are physical in-path deployments. Physical in-path configurations are the easiest to deploy and do not require ongoing maintenance as other configurations do (such as virtual in-path configurations: WCCP, PBR, and Layer-4 redirection).

For networks that contain firewalls or tunnels (VPN, GRE, IPSec transport mode) between SteelHeads and require manual tuning of the MTU values, see [“MTU sizing” on page 476](#).

You must manually configure the IP address for any optimization interface, including the primary interface, in an out-of-path deployment. DHCP is not supported.

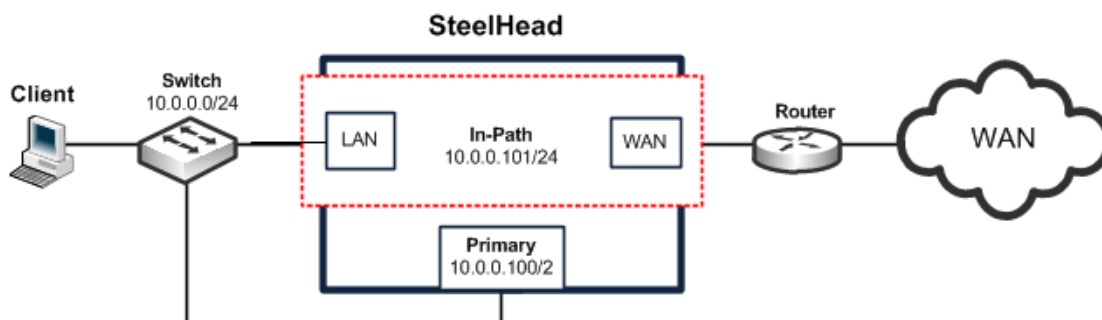
Logical in-path interface

All SteelHeads ship with at least one pair of ports that are used for in-path deployments. This pair of ports forms the logical in-path interface. The logical in-path interface acts as an independent, two-port bridge, with its own IP address. This section includes the following topics:

- [“In-path IP address selection” on page 199](#)
- [“In-path default gateway and routing” on page 199](#)

Figure 9-2 shows the SteelHead logical in-path interface and how it is physically connected to network devices in a single subnet, in-path deployment.

Figure 9-2. Logical in-path interface in a single subnet in-path deployment



The simplest in-path SteelHead has two IP addresses:

- **Primary** - Used for system management, RiOS data store synchronization, and SNMP.
- **Inpath0_0** - Used for optimized data transmission.

Several types of network interface cards (bypass cards) are available for SteelHeads. The desktop SteelHeads have network bypass functionality built in. With 1U and 3U systems, you can choose the type of bypass card. SteelHeads can have both copper and fiber Ethernet bypass cards.

For information about bypass cards, see the *Network and Storage Card Installation Guide* on the Riverbed Support site.

In-path IP address selection

An IP address is required for each SteelHead in-path interface. When using correct addressing or port transparency, the IP address must be reachable by remote SteelHeads for optimization to occur.

In some environments, the link between the switch and the router might reside in a subnet that has no available IP address. You can use the following solutions to accommodate the IP address requirement:

- Create a secondary interface, with a new subnet and IP address on the router or switch, and pull the SteelHead in-path interface IP address from the new subnet.
- Create a new 802.1Q VLAN interface and subnet on the router and switch link, and pull the SteelHead in-path interface IP address from the new subnet. This solution also requires entering the appropriate in-path VLAN tag on the SteelHead.

Note: You must manually configure the in-path IP address for in-path deployments. DHCP is not supported.

With RiOS 5.0.x or later, you can deploy SteelHeads so that the in-path interface IP address is not actually used. This deployment option can be useful for integrating with certain network configurations, such as NAT. However, an IP address must be configured for each enabled in-path interface.

For information about correct addressing, port transparency, and full transparency, see **“WAN Visibility Modes” on page 63**. For more information about deploying a SteelHead into an existing network, see the Riverbed Knowledge Base article *SteelHead Deployment onto an Existing /30 Network* at <https://supportkb.riverbed.com/support/index?page=content&id=S14964>.

In-path default gateway and routing

Almost all in-path deployments require the configuration of a default gateway for the in-path interfaces. A physical in-path SteelHead might need to transmit packets from its in-path interface to any:

- local hosts, for the LAN side of any optimized connections.
- remote SteelHeads, for the WAN side of any optimized connections.
- remote hosts, when transmitting packets during autodiscovery.
- local SteelHead and SteelHead Interceptors, when communicating with connection-forwarding neighbors.

You must configure an in-path gateway if any of these devices is on a different subnet from the in-path interface.

In small branches, where a SteelHead is physically placed between an access switch and a router or firewall, and all hosts are on the same subnet, then the in-path default gateway must use the same IP address that the local hosts use—that of the router or firewall. With this configuration, the SteelHead uses the gateway as the Layer-2 next hop when transmitting to remote hosts or SteelHeads, and the SteelHead uses MAC address discovery through ARP when transmitting packets to the local hosts.

In larger branches, where the SteelHead is deployed between two Layer-3 devices (for example, between a Layer-3 switch and a WAN-side router), then the SteelHead can be configured with a specific in-path gateway, static routes, and simplified routing to ensure that it always transmits packets to the optimal next hop. Although it is impossible to generalize for all environments, a typical configuration for locations that minimize packet ricochet and ensure the best performance use the:

- WAN-side Layer-3 device as the in-path default gateway.
- simplified routing destination-only option.
- enhanced autodiscovery feature.

Some environments require different settings or additional configuration. For more information, go to the Riverbed Support site at <https://support.riverbed.com>.

Failure modes

This section describes the SteelHead failure modes. This section includes the following topics:

- [“Fail-to-wire mode” on page 201](#)
- [“Fail-to-block mode” on page 202](#)
- [“Configuring failure modes” on page 202](#)

All SteelHead models and in-path network interface cards support fail-to-wire mode. In the event of a disk failure, a software crash, a runaway software process, or even loss of power to the SteelHead, the LAN and WAN ports that form the logical in-path interface become internally connected as if they were the ends of a crossover cable, thereby providing uninterrupted transmission of data over the WAN.

Certain in-path network interface cards also support a fail-to-block mode, where in the case of a failure or loss of power, the SteelHead LAN and WAN interfaces completely lose link status, blocking traffic and forcing it to be rerouted onto paths where the remaining SteelHeads are deployed. The default failure mode is fail-to-wire mode.

For a list of in-path network interface cards or bypass cards that support fail-to-block mode, see [“Fail-to-block mode” on page 202](#).

If a SteelHead transitions to fail-to-wire or fail-to-block mode, you are notified in the following ways:

- The Intercept/Bypass status light is on.
- Critical appears in the Management Console status bar.
- SNMP traps are sent (if you have set this option).
- The event is logged to system logs (syslog) (if you have set this option).
- Email notifications are sent (if you have set this option).

Fail-to-wire mode

Fail-to-wire mode allows the SteelHead WAN and LAN ports to serve in the same way as an Ethernet crossover cable. In fail-to-wire mode, SteelHeads cannot view or optimize traffic. Instead, all traffic is passed through the SteelHead unoptimized.

All SteelHead in-path interfaces support fail-to-wire mode. Fail-to-wire mode is the default setting for SteelHeads.

When a SteelHead transitions from normal operation to fail-to-wire mode, SteelHead circuitry physically moves to electronically connect the SteelHead LAN and WAN ports to each other, and physically disconnects these two ports from the rest of the SteelHead. During the transition to fail-to-wire mode, device linked to the SteelHead momentarily disconnect and then immediately connect again. After the transition, traffic resumes flowing as quickly as the connected devices are able to process it. For example, spanning-tree configuration and routing-protocol configuration influence how quickly traffic resumes flowing. Traffic that was passed-through is uninterrupted. Traffic that was optimized might be interrupted, depending on the behavior of the application-layer protocols. When connections are restored, the traffic resumes flowing, although without optimization.

After the SteelHead returns to normal operation, it transitions the SteelHead LAN and WAN ports out of fail-to-wire mode. The devices connected to the SteelHead perceive this mode as another link state transition. After they are back online, new connections that are made are optimized. However, connections made during the failure are not optimized.

To force all connections to be optimized, you can enable the kickoff feature. This feature resets established connections to force them to go through the connection creation process again. For this reason, before enabling the kickoff feature in production deployments, you must understand and accept that all TCP connections are reset. Generally, connections are short lived and kickoff is not necessary.

For information about enabling the kickoff feature, see [“Kickoff and automatic kickoff features” on page 40](#) and the *SteelHead User Guide*.

When a SteelHead transitions to fail-to-wire mode, the transition can have an effect on devices connected to the SteelHead. For example, one common implication pertains to the spanning-tree protocol. In many physical in-path deployments, the SteelHead LAN port is connected to an Ethernet switch, and the SteelHead WAN port is connected to a router.

When a SteelHead transitions from bridging mode to failure mode, a switch might force the port that is connected to the SteelHead to go through the 30- to 45-second, nonforwarding states of spanning tree which can result in packet delay or packet loss.

You can resolve this issue by making configuration modifications on your switch. Depending on your switch vendor, there are many different methods to alleviate this issue, ranging from skipping the nonforwarding states (for example, running the **spanning-tree portfast** command on Cisco switches), to using newer 802.1d STP protocols that converge faster on link transitions.

RiOS 5.0.x or later has this mode transition issue only when the SteelHead experiences a power loss. RiOS 4.1 and earlier has this transition state issue when the SteelHead experiences a power loss, software failure, or when the optimization service is restarted.

Fail-to-block mode

Some network interfaces support fail-to-block mode. In fail-to-block mode, if the SteelHead has an internal software failure or power loss, the SteelHead LAN and WAN interfaces power down and stop bridging traffic. Fail-to-block mode is useful only if the network has a routing or switching infrastructure that can automatically divert traffic from the link after the failed SteelHead blocks it. You can use fail-to-block mode with connection forwarding, the **allow-failure** command, and an additional SteelHead on another path to the WAN to achieve redundancy.

For more information about connection forwarding and fail-to-block, see [“Configuring connection forwarding with allow-failure and fail-to-block” on page 229](#).

Check the *Network and Storage Card Installation Guide* on the Riverbed Support site for a current list of SteelHead in-path interfaces that support fail-to-block mode.

Configuring failure modes

This section shows common failure mode configurations using the CLI.

To enable fail-to-block mode

- Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
no interface inpath0_0 fail-to-bypass enable
write memory
```

Note: The changes take effect immediately. You must save your changes or they are lost upon reboot.

To change from fail-to-block mode back to fail-to-wire mode

- Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 fail-to-bypass enable
write memory
```

Note: The changes take effect immediately. You must save your changes or they are lost upon reboot.

To check failure mode status

- Connect to the SteelHead CLI and enter the following commands:

```
enable
show interface inpath0_0
```

Configuring link state propagation

In physically in-path deployments, link state propagation (LSP) helps communicate link status between the devices connected to the SteelHead. When this feature is enabled, the link state of each SteelHead LAN and WAN pair is monitored. If either physical port loses link status, the link of the corresponding physical port is also brought down. Link state propagation allows link failure to quickly propagate through the SteelHead, and it is useful in environments where link status is used as a fast-fail trigger.

For example, in a physical in-path deployment where the SteelHead is connected to a router on its WAN port and a switch on its LAN port, if the cable to the router is disconnected, the SteelHead deactivates the link on its LAN port. This deactivation causes the switch interface that is connected to the SteelHead to also lose the link. The reverse is also true: if the cable to the switch is disconnected, the router interface that is connected to the SteelHead loses the link.

You can use LSP in a SteelHead serial cluster. In a serial cluster deployment, link state propagation can be useful to quickly propagate failure if the cables between SteelHeads are disconnected. For example, in a two-appliance SteelHead serial cluster, if you disconnect the cable between the SteelHeads, then both the WAN-side router and the LAN-side switch lose the link.

Link state propagation is supported on either all or none of the interfaces of a SteelHead; it cannot be used to selectively activate an in-path interface.

To enable link state propagation on a SteelHead


Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
in-path lsp enable
write memory
```

Note: The changes take effect immediately. You must save your changes or they are lost upon reboot.

In RiOS 6.0 or later, link state propagation is enabled by default.

 SteelHead-c models do not support LSP.

 SteelHead-v running RiOS 8.0.3 with ESXi 5.0 and later using a Riverbed NIC card support LSP.

These SteelHead-v configurations do not support LSP:

- SteelHead-v models running ESX/ESXi 4.0 or 4.1
- SteelHead-v models running Microsoft Hyper-V
- SteelHead-v models running RiOS 8.0.2 and earlier

EtherChannel

In RiOS 9.1 and later, you can deploy SteelHeads across Ethernet links that are logically aggregated using EtherChannel or IEEE 802.3ad. These links use protocols such as Cisco Port Aggregation Protocol (PAgP) or the IEEE Link Aggregation Control Protocol (LACP) to negotiate how multiple physical links are bundled into a single logical link. The SteelHead passes through negotiation protocols without participating in them.

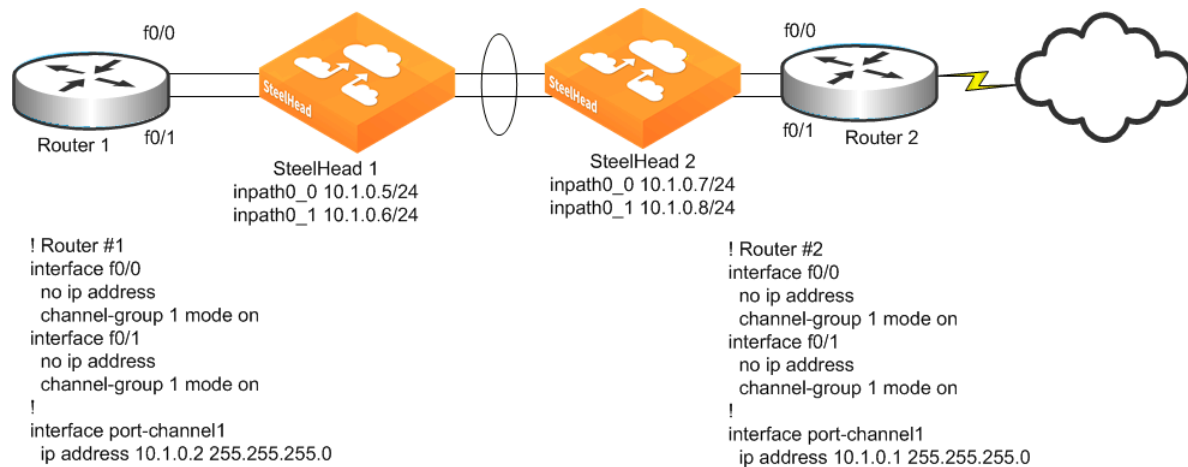
When deploying SteelHeads on aggregated links, you must:

- configure the in-path interfaces to have a unique IP address.
- configure the in-path interfaces within the channel (CLI only) using the **in-path bundle <bundle-name> interfaces <in-path-interface-name-1>,<in-path-interface-name-2>,<additional-interfaces>** command. Each bundle can have as many interfaces as you want, and you can configure multiple bundles per SteelHead.
- enable LSP.

The SteelHead must be physically in-path to use EtherChannel—you cannot use EtherChannel in a logical in-path configuration. You can configure SteelHeads with connection forwarding to neighbor appliances, which might or might not be deployed on port aggregated interfaces.

Figure 9-3 shows two SteelHeads with two links each deployed in series on a two-link EtherChannel. Each SteelHead in-path interface has its own unique IP address in the EtherChannel link's subnet. Router 1 and Router 2 each have a single IP address for the two physical links, but each SteelHead has two IP addresses, one for each in-path interface.

Figure 9-3. Two SteelHeads in a serial deployment on a 2-port EtherChannel link



The following example shows the CLI configuration for the in-path and ARP filter configuration for SteelHead 1 as shown in Figure 9-3:

```

in-path interface inpath0_0 enable
interface inpath0_0 ip address 10.1.0.5 /24
ip in-path-gateway inpath0_0 10.1.0.1
in-path interface inpath0_1 enable
interface inpath0_1 ip address 10.1.0.6 /24
ip in-path-gateway inpath0_1 10.1.0.1
in-path bundle lagg1 interfaces inpath0_0,inpath0_1
  
```

The SteelHead supports the port channel or channels through all of its available interfaces, including the four 10-Gbps in-path interfaces, or up to ten 1-Gbps interfaces. All links within the port channel must pass through the same SteelHead.

To ensure the SteelHead bundles interfaces appropriately in an EtherChannel, the following parameters must be consistent on all in-path interfaces:

- Speed/Duplex
- MTU
- VLAN ID
- IP address in the same subnet
- Default gateway
- User defined routes

Note: You can deploy SteelHeads with virtual port channel technologies such as Cisco Catalyst 6500s in a VSS configuration. The restriction is that all links within the port channel must traverse the same SteelHead.

Cabling and duplex

Using the appropriate cables and interface settings for in-path deployments are vital to performance and resiliency. The physical cabling and interface settings that connect the SteelHead to the LAN and WAN equipment (typically a switch and router) at the site must be correct. Duplex mismatches between the SteelHead and equipment connected to it, either during normal operations or during failures when the SteelHead is in fail-to-wire mode, have a significant impact on the performance of all traffic passing through the SteelHead. This section includes the following topics:

- [“Choosing the correct cables” on page 205](#)
- [“Duplex configuration” on page 206](#)
- [“Troubleshooting cable and duplex issues” on page 207](#)

A duplex mismatch or incorrect interface settings might cause optimized performance to be less than nonoptimized performance or prevent traffic from flowing—even if the SteelHead is configured properly.

Choosing the correct cables

The LAN and WAN ports on the SteelHead bypass cards act like host interfaces during normal operation. During fail-to-wire mode, the LAN and WAN ports act as the ends of a crossover cable. Using the correct cable to connect these ports to other network equipment ensures proper operation during fail-to-wire mode and normal operating conditions.

We recommend that you do not rely on automatic MDI/MDI-X to automatically sense the cable type. The installation might work when the SteelHead is optimizing traffic, but it might not if the in-path bypass card transitions to fail-to-wire mode.

One way to help ensure that you use the correct cables during an installation is to connect the LAN and WAN interfaces of the SteelHead while the SteelHead is powered off. Connecting the interfaces proves that the devices on either side of the SteelHead can communicate correctly without any errors or other problems.

In the most common in-path configuration, a SteelHead's LAN port is connected to a switch and the SteelHead's WAN port is connected to a router. In this configuration, a straight-through Ethernet cable can connect the SteelHead's LAN to the switch, and a crossover cable must be used to connect the SteelHead's WAN port to the router.

The following table summarizes the correct cable usage in the SteelHead.

Devices	Cable
SteelHead or Interceptor to SteelHead or Interceptor	Crossover
SteelHead or Interceptor to router	Crossover
SteelHead or Interceptor to switch	Straight-through
SteelHead or Interceptor to host	Crossover

Duplex configuration

Depending on which SteelHead bypass card you use, you must choose between manually setting the speed and duplex for its LAN and WAN interfaces or allowing the interfaces to automatically negotiate. Choosing the correct setting ensures that packets can pass through the interfaces, both during normal operating mode and during fail-to-wire mode, without any errors or drops due to a mismatch between the SteelHead and its connected network equipment.

The correct duplex settings to use depend on the capabilities of all of the interfaces in the *chain of in-path interfaces*: the connected LAN device (typically a switch), the LAN and WAN ports on the SteelHead bypass cards in use by one or more in-path SteelHeads, and the connected device (typically a router). A typical in-path deployment has a SteelHead LAN port connecting to a switch port that is 10/100/1000 Mbps capable, but the SteelHead WAN port connects to a router interface that is only capable of 10/100. In this deployment, manually set both the SteelHead LAN and WAN ports to use 100 Mbps, full duplex. These settings ensure correct operation during normal operation and fail-to-wire mode.

We make the following recommendations:

- If all interfaces in the in-path chain are capable of 1 Gbps or higher speeds, use automatic negotiation on all interfaces in the in-path chain.
- Configure for automatic negotiation on all Ethernet ports running at 100 Mbps unless you know for sure of a specific automatic negotiation issue between the SteelHead port and the peer device.
- Never use half duplex—either set manually to full duplex or use automatic negotiation.

If you deviate from these recommendations, you must perform tests to verify that traffic flows when the SteelHead is optimizing traffic and entered fail-to-wire mode.

For example, an interface mismatch can happen if the LAN interface is connected to a 1-Gbps device, the WAN interface is connected to a 100-Mbps device, and WAN bandwidth is close or equal to 100 Mbps. To avoid any potential bottleneck that prevents the SteelHead LAN interface from sending or receiving data at a rate greater than 100 Mbps, it is prudent for you to use automatic negotiation on both LAN and WAN interfaces. If you use automatic negotiation, the SteelHead can perform at its best both on the LAN side and the WAN side.

Duplex misconfiguration is not limited to the chain of in-path interfaces, especially at remote locations, where WAN limitations restrict the potential performance of applications. There might be long-standing, unrealized duplex-related errors in the existing LAN infrastructure or even on host interfaces. You might only discover these long-standing issues when a SteelHead is deployed at the site and attention is concentrated on achieving high performance. Due to the infrastructure duplex problems, the SteelHead performance gains are not as significant as expected. Any such infrastructure issues limit the optimization possible by deploying a SteelHead and must be resolved to realize the full benefits of a SteelHead deployment.

The following signs indicate a duplex misconfiguration:

- You cannot connect to an attached device.
- You can connect to a device when you choose automatic negotiation, but you cannot connect to that same device when you manually set the speed or duplex.
- You detect performance issues across the network.

Troubleshooting cable and duplex issues

This section shows common cable and duplex troubleshooting procedures.

To verify if slow performance on the network is due to a duplex problem on the chain of in-path interfaces

1. From the Management Console, choose Reports > Networking: Interface Counters.
2. Look for positive values for the following fields:
 - Discards
 - Errors
 - Overruns
 - Frame
 - Carrier counts
 - Collisions

These values are zero on a healthy network, unless you use half duplex. We recommend that you do not use half duplex.

To verify if slow performance on the network is due to a duplex problem within the LAN infrastructure, and not on an interface in the in-path chain

1. From the Management Console, choose Reports > Networking: Current Connections.
2. Look for any optimized connection, and click the magnifying glass icon next to the connection to see its details.
3. Look for zeros in the following fields:
 - Retransmitted
 - Fast Retransmitted
 - Time-outs

If the values are greater than zero, some type of LAN-side packet loss was experienced for that connection. This packet loss might be because of a duplex misconfiguration somewhere between the local host and the SteelHead's LAN interface.

Note: Speed and duplex issues might be present at other points in the network path besides the interfaces directly connected to the SteelHead. There might be long-standing interface errors within the LAN, whose symptoms might have been incorrectly blamed on WAN performance.

Physical in-path deployment configuration examples

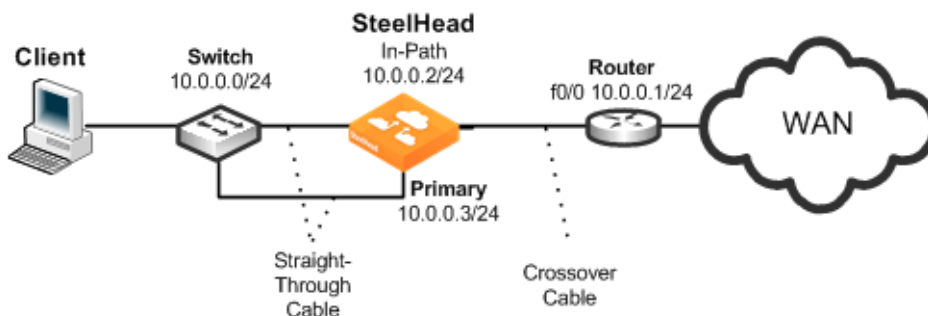
This section describes common deployment options. This section includes the following topics:

- [“Configuring a basic physical in-path deployment” on page 208](#)
- [“Configuring a physical in-path with dual links deployment” on page 211](#)
- [“Configuring a serial cluster deployment with multiple links” on page 211](#)

Configuring a basic physical in-path deployment

The simplest physical in-path SteelHead deployment is also the most commonly deployed.

Figure 9-4. Simple, physical in-path deployment



Basic steps to perform before you deploy a physical in-path SteelHead

1. Determine the speed for the:

- switch interface.
- router interface.
- SteelHead primary interface.
- SteelHead WAN interface.
- SteelHead LAN interface.

We recommend the following speeds:

- **Fast Ethernet interfaces** - 100 Mb full duplex
- **Gigabit interfaces** - 1000 Mb full duplex

2. Determine the IP addresses for the SteelHead. A SteelHead that is deployed in a physical in-path mode requires two IP addresses, one each for the:

- SteelHead in-path interface.
- SteelHead primary interface (used for managing the SteelHead).

In addition to using the primary interface for management purposes, you can also use the auxiliary (AUX) interface or the in-path management interface to manage the SteelHead. The AUX interface is another physical interface on the SteelHead, and the in-path management interface is a virtual interface that is associated to the SteelHead in-path interfaces.

When you configure the AUX interface, you cannot have it in the same subnet as the primary interface.

Each in-path management interface has one in-path interface. For example, inpath0_0 has a corresponding mgmt0_0 interface, inpath0_1 has a corresponding mgmt0_1 interface, and so on. Any connections destined to the in-path management interface are not optimized, and these connections do not appear in the Current Connections report.

The following characteristics apply to the in-path management interface:

- Must be in its own subnet
- Cannot share the same subnet with any other interfaces on the SteelHead (this includes other in-path interfaces)
- Is accessible from either the LAN side or WAN side
- Uses the main routing table and is always up
- Supports 802.1Q and processes only packets destined to its VLAN ID (if you configure one)

3. Manually configure the speed for the:

- switch interface.
- router interface.
- SteelHead primary interface.
- SteelHead WAN interface.
- SteelHead LAN interface.

Important: We strongly recommend that you manually configure interface speed, unless you want to use 1 Gbps, which must be set using automatic negotiation. For details, go to <https://supportkb.riverbed.com/support/index?page=content&id=S14623>.

4. Configure the appropriate default gateway for the primary and in-path interfaces:

- **Primary port gateway IP** - Specify the primary gateway IP address.
- **In-path gateway IP** - Specify the IP address for the in-path gateway. If you have a router (or a Layer-3 switch) on the LAN side of your network, specify this device as the in-path gateway. Make sure that you have simplified routing enabled.

For more information, see [“In-path default gateway and routing” on page 199](#) and [“Configuring a physical in-path with dual links deployment” on page 211](#).

Using [Figure 9-4](#) as your environment, the following task includes the minimum steps required to configure the simplest physical in-path SteelHead deployment.

The example requires that you have configured your cabling and duplex according to the recommendations described in [“Cabling and duplex” on page 205](#).

To configure the SteelHead for basic physical in-path deployment

- On the SteelHead, connect to the CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 ip address 10.0.0.2 /24
ip in-path-gateway inpath0_0 10.0.0.1
interface primary ip address 10.0.0.3 /24
ip default-gateway 10.0.0.1
in-path enable
```

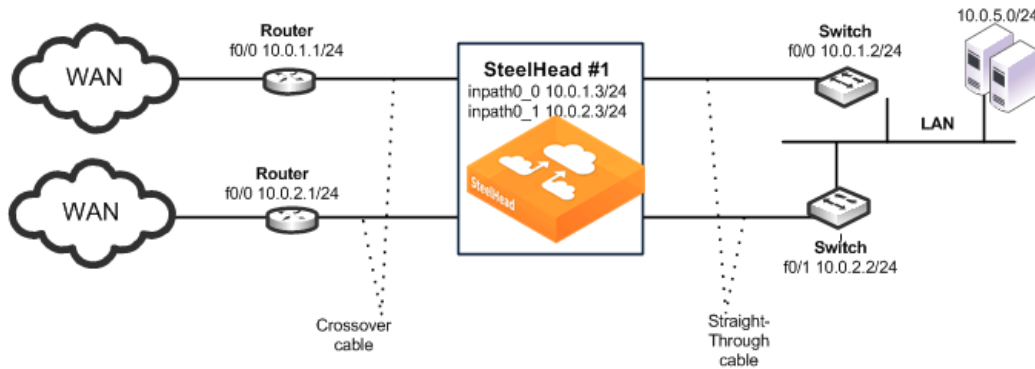
Configuring a physical in-path with dual links deployment

This example requires that you have configured your cabling and duplex according to the recommendations described in [“Cabling and duplex” on page 205](#).

Figure 9-5 shows a physical in-path with dual links SteelHead deployment.

Note: Simplified routing removes any packet ricochet that occurs when the SteelHead sends traffic to the 10.0.5.0/24 LAN.

Figure 9-5. Physical in-path with dual links deployment



The following SteelHead CLI commands are the minimum commands required to configure the physical in-path SteelHead with dual links. These commands do not include the configuration of features such as duplex, alarms, SNMP, and DNS.

To configure a SteelHead physically in-path with dual links

1. On SteelHead 1, connect to the CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 ip address 10.0.1.3 /24
ip in-path-gateway inpath0_0 10.0.1.2
interface inpath0_1 ip address 10.0.2.3 /24
ip in-path-gateway inpath0_1 10.0.2.2
in-path enable
in-path peering auto
in-path simplified routing all
write memory
restart
```

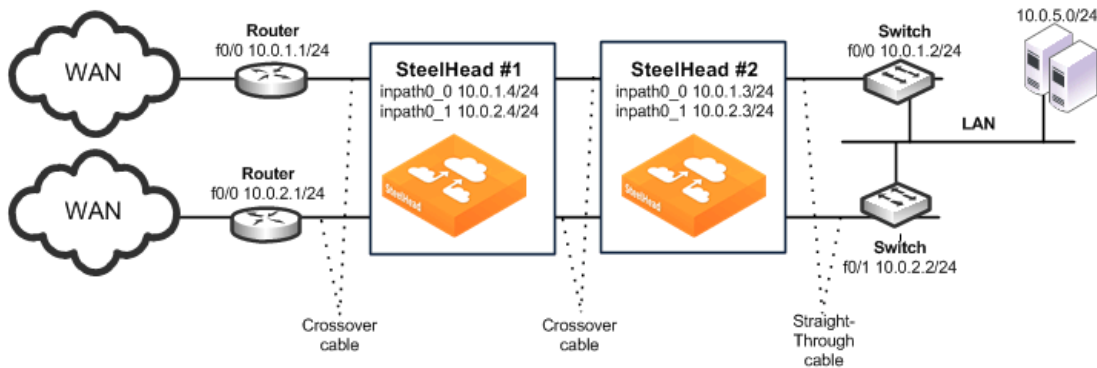
Configuring a serial cluster deployment with multiple links

This example requires that you have configured your cabling and duplex according to the recommendations described in [“Cabling and duplex” on page 205](#).

Figure 9-6 shows a serial cluster deployment with multiple WAN links. Each of the links are on different subnets, but they might also be in the same subnet.

Note: Link state propagation is enabled between the SteelHeads. For details, see the *SteelHead User Guide*.

Figure 9-6. Physical in-path, multiple link serial cluster deployment



The following SteelHead CLI commands are the minimum commands required to configure a serially clustered SteelHead deployment with multiple WAN links. These commands do not include the configuration of features such as duplex, alarms, and DNS.

To configure serially clustered SteelHeads with multiple WAN links

1. On SteelHead 1, connect to the CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 ip address 10.0.1.4 /24
ip in-path-gateway inpath0_0 10.0.1.2
interface inpath0_1 ip address 10.0.2.4 /24
ip in-path-gateway inpath0_1 10.0.2.2
in-path enable
in-path peering auto
in-path simplified routing dest-only
in-path peering rule pass peer 10.0.1.3 rulenum end
in-path peering rule pass peer 10.0.2.3 rulenum end
write memory
restart
```

2. On SteelHead 2, connect to the CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 ip address 10.0.1.3 /24
ip in-path-gateway inpath0_0 10.0.1.2
interface inpath0_1 ip address 10.0.2.3 /24
ip in-path-gateway inpath0_1 10.0.2.2
in-path enable
in-path simplified routing dest-only
in-path peering auto
in-path peering rule pass peer 10.0.1.4 rulenum end
in-path peering rule pass peer 10.0.2.4 rulenum end
write memory
restart
```

In-path redundancy and clustering examples

You can use the following techniques to configure multiple SteelHeads in physical in-path deployments. These deployments achieve redundancy and clustering for optimization. This section covers the following scenarios:

- “Primary and backup deployments” on page 213
- “Serial cluster deployments” on page 215

You can use the techniques in each scenario to provide optimization across several physical links. You can use these techniques in conjunction with connection forwarding when all of the physical links to and from the WAN are unable to pass through a single SteelHead.

For information about connection forwarding, see “Connection forwarding” on page 56.

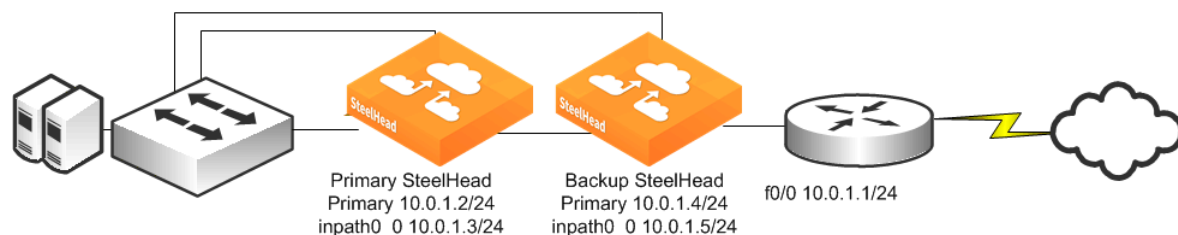
Primary and backup deployments

In a primary and backup deployment, two equivalent model SteelHeads are placed physically in-path. This section includes the following topics:

- “Configuring a primary and backup deployment” on page 214
- “Adjusting the timers for faster primary and backup failover” on page 215

The SteelHead closest to the LAN is configured as a primary, and the other SteelHead is configured as the backup. The primary SteelHead optimizes traffic and the backup SteelHead checks to make sure the primary SteelHead is functioning and not in *admission control*. Admission control means the SteelHead has stopped trying to optimize new connections, due to reaching its TCP connection limit or due to some abnormal condition. If the backup SteelHead cannot reach the primary or if the primary has entered admission control, the backup SteelHead begins optimizing new connections until the primary recovers. After the primary has recovered, the backup SteelHead stops optimizing new connections, but continues to optimize any existing connections that were made while the primary was down. The recovered primary optimizes any newly formed connections.

Figure 9-7. Primary and backup deployment



If you use the primary and backup deployment method with SteelHeads that have multiple active in-path links, peering rules must also be configured. Add peering rules to both SteelHeads for each in-path interface; these peering rules must have an action *pass* for a peer IP address of each of the in-path IP addresses. This setting ensures that during any window of time during which both SteelHeads are active (for example, during a primary recovery), the SteelHeads do not try to optimize connections between themselves.

Typically, RiOS data store synchronization is used in primary and backup deployments. RiOS data store synchronization ensures that any data written to one SteelHead eventually is pushed to the other SteelHead. Although both the primary and backup deployment option and the RiOS data store synchronization feature use the terms *primary* and *backup*, the uses are different and separate. You can typically configure one SteelHead to be a primary for both, but it is not a requirement.

For information about data synchronization, see “[RiOS data store synchronization](#)” on page 27.

Consider using a primary and backup deployment instead of a serial cluster when all of the following statements are true:

- Only two SteelHeads are placed physically in-path.
- The capacity of a single SteelHead is sufficient for the site.
- Only a single in-path interface is active on both SteelHeads.

Some environments might require additional considerations. For more information, go to the Riverbed Support site at <https://support.riverbed.com>.

Configuring a primary and backup deployment

This section describes how to configure the primary and backup deployment shown in [Figure 9-7](#).

To configure the primary and backup SteelHeads

1. Connect to the primary SteelHead CLI and enter the following commands:

```
#-- Primary SteelHead.
interface primary ip address 10.0.1.2/24
ip default gateway 10.0.1.1
interface inpath0_0 ip address 10.0.1.3/24
ip in-path-gateway inpath0_0 10.0.1.1
#-- Failover should point to the inpath0_0 address.
failover steelhead addr 10.0.1.5
failover master
failover enable
in-path enable
#-- Although not required, RiOS data store synchronization is usually enabled
#-- in primary/backup deployments.
datastore sync master
#-- RiOS data store should point to peer primary or aux interface address.
datastore sync peer-ip 10.0.1.4
datastore sync enable
write memory
restart
```

2. Connect to the backup SteelHead CLI and enter the following commands:

```
#-- Backup SteelHead.
interface primary ip address 10.0.1.4/24
ip default gateway 10.0.1.1
interface inpath0_0 ip address 10.0.1.5/24
ip in-path-gateway inpath0_0 10.0.1.1
#-- Failover should point to the inpath0_0 address.
failover steelhead addr 10.0.1.3
no failover master
failover enable
in-path enable
#-- Although not required, RiOS data store synchronization is usually enabled in
#-- primary/backup deployments.
```

```
no datastore sync master
#-- RiOS data store should point to peer's primary or aux interface address.
datastore sync peer-ip 10.0.1.2
datastore sync enable
write memory
restart
```

Note: For more information about configuring primary and backup deployment, see the Failover Support commands in the *Riverbed Command-Line Interface Reference Manual* and the “Enabling Failover” section in the *SteelHead User Guide*.

Adjusting the timers for faster primary and backup failover

In a steady, normal operating state, the backup SteelHead periodically sends keep-alive messages to the primary SteelHead on TCP port 7820. If the primary SteelHead does not respond to the keep-alive message within 5 seconds, the backup SteelHead drops the connection and attempts to reconnect to the primary SteelHead. The backup SteelHead attempts to reconnect a maximum of five times, and each time it waits for 2 seconds before aborting the connection.

If all connection attempts fail, the backup SteelHead transitions into an active state and starts optimizing the connections. If you use the default value failover settings, it can take as long as 15 seconds before the backup SteelHead starts optimizing connections.

You can adjust several failover settings to shorten the failover time:

- **Read timeout (in milliseconds)** - Governs how many milliseconds the backup SteelHead waits for the primary SteelHead to respond to its keep-alive messages. Use the **failover read timeout** command to adjust this setting. The default value is 5000 ms.
- **Connection attempts** - The number of times the backup SteelHead attempts to reconnect to the primary SteelHead after read time-out has expired. Use the **failover connection attempts** command to adjust this setting. The default value is 5.
- **Connection timeout (in milliseconds)** - The number of milliseconds the backup SteelHead waits before aborting the reconnection attempt to the primary SteelHead. Use the **failover connection timeout** command to adjust this setting. The default value is 2000 ms.

To reduce the failover time to 5 seconds, you can adjust the timers to the following settings:

- **Read timeout:** 1000 ms
- **Connection attempts:** 4
- **Connection timeout:** 1000 ms

Serial cluster deployments

You can provide increased optimization by deploying two or more SteelHeads back-to-back in an in-path configuration to create a serial cluster. This section includes the following topics:

- [“Serial cluster rules” on page 216](#)
- [“Configuring a basic serial cluster deployment” on page 217](#)
- [“Configuring faster peer failure detection” on page 219](#)

SteelHeads in a serial cluster process the peering rules you specify in a spillover fashion. When the maximum number of TCP connections for a SteelHead is reached, that appliance stops intercepting new connections. This behavior allows the next SteelHead in the cluster the opportunity to intercept the new connection, if it has not reached its maximum number of connections.

The in-path peering rules and in-path rules tell the SteelHead in a cluster not to intercept connections between themselves. You configure peering rules that define what to do when a SteelHead receives an autodiscovery probe from another SteelHead. You can deploy serial clusters on the client or server side of the network.

Important: For environments in which you want to optimize MAPI or FTP traffic, which require all connections from a client to be optimized by one SteelHead, we strongly recommend using the primary and backup redundancy configuration instead of a serial cluster deployment. For larger environments that require multiple appliance scalability and high availability, we recommend using the SteelHead Interceptor to build multiple appliance clusters. For details, see the *SteelHead Interceptor User Guide*.

Before you configure a serial cluster deployment, consider the following factors:

- The total optimized WAN capacity of the cluster can reach the sum of the optimized WAN capacity of the individual SteelHeads, assuming that both SteelHeads are optimizing connections. Typically, both SteelHeads optimize connections if connections originate from both WAN and LAN or if one of the SteelHeads reaches its capacity limit and passes through subsequent connections (which are then optimized by the other SteelHead).
- If the active SteelHead in the cluster enters a degraded state because the CPU load is too high, it continues to accept new connections.

For more information about working with serial clusters, see the Riverbed Knowledge Base article *Working with Serial Clustering* at <https://supportkb.riverbed.com/support/index?page=content&id=s15555>.

Serial cluster rules

The in-path peering rules and in-path pass-through rules tell the SteelHeads in a serial cluster not to intercept connections between each other. The peering rules define what happens when a SteelHead receives an autodiscovery probe from another SteelHead in the same cluster.

You can deploy serial clusters on the client or server side of the network.

Figure 9-8. Serial cluster deployment

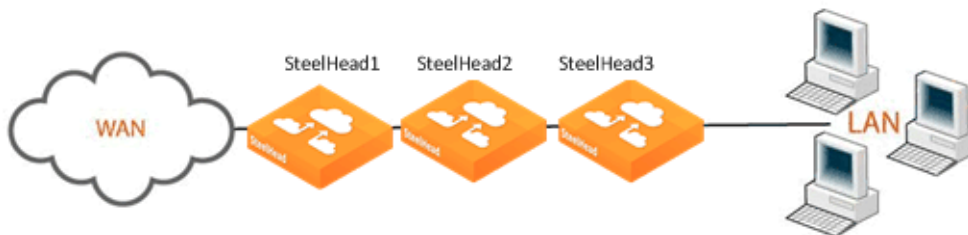


In this example, SteelHead1, SteelHead2, and SteelHead3 are configured with in-path peering rules so they do not answer probe requests from one another, and with in-path rules so they do not accept their own WAN connections. Similarly, SteelHead4, SteelHead5, and SteelHead6 are configured so that they do not answer probes from one another and do not intercept inner connections from one another. The SteelHeads are configured to find an available peer SteelHead on the other side of the WAN.

Configuring a basic serial cluster deployment

Figure 9-9 shows an example serial cluster deployment of three in-path SteelHeads in a data center.

Figure 9-9. Serial cluster in a data center



This example uses the following parameters:

- SteelHead1 in-path IP address is 10.0.1.1
- SteelHead2 in-path IP address is 10.0.1.2
- SteelHead3 in-path IP address is 10.0.1.3

In this example, you configure each SteelHead with in-path peering rules to prevent peering with another SteelHead in the cluster, and with in-path rules to not optimize connections originating from other SteelHeads in the same cluster.

To configure a basic serial cluster deployment with three SteelHeads

1. On SteelHead1, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path peering rule pass peer 10.0.1.2 rulenum 1
in-path peering rule pass peer 10.0.1.3 rulenum 1
in-path rule pass-through srcaddr 10.0.1.2/32 rulenum 1
in-path rule pass-through srcaddr 10.0.1.3/32 rulenum 1
write memory
show in-path peering rules
```

Rule	Type	Source Network	Dest Network	Port	Peer Addr
1	pass	*	*	*	10.0.1.3
2	pass	*	*	*	10.0.1.2
def auto	*	*	*	*	*

```
show in-path rules
```

Rule	Type	Source Addr	Dest Addr	Port	Target Addr	Port
1	pass	10.0.1.3/24	*	*	--	--
2	pass	10.0.1.2/24	*	*	--	--
def auto	*	*	*	*	--	--

The changes take effect immediately. You must save your changes or they are lost upon reboot.

2. On SteelHead2, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path peering rule pass peer 10.0.1.1 rulenum 1
in-path peering rule pass peer 10.0.1.3 rulenum 1
in-path rule pass-through srcaddr 10.0.1.1/32 rulenum 1
in-path rule pass-through srcaddr 10.0.1.3/32 rulenum 1
write memory
```

```
show in-path peering rules
```

Rule	Type	Source Network	Dest Network	Port	Peer Addr
1	pass	*	*	*	10.0.1.3
2	pass	*	*	*	10.0.1.1
def auto	*	*	*	*	*

```
show in-path rules
```

Rule	Type	Source Addr	Dest Addr	Port	Target Addr	Port
1	pass	10.0.1.3/24	*	*	--	--
2	pass	10.0.1.1/24	*	*	--	--
def auto	*	*	*	*	--	--

The changes take effect immediately. You must save your changes or they are lost upon reboot.

3. On SteelHead3, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path peering rule pass peer 10.0.1.1 rulenum 1
in-path peering rule pass peer 10.0.1.2 rulenum 1
in-path rule pass-through srcaddr 10.0.1.1/32 rulenum 1
in-path rule pass-through srcaddr 10.0.1.2/32 rulenum 1
write memory
```

```
show in-path peering rules
```

Rule	Type	Source Network	Dest Network	Port	Peer Addr
1	pass	*	*	*	10.0.1.2
2	pass	*	*	*	10.0.1.1
def auto	*	*	*	*	*

```
show in-path rules
```

Rule	Type	Source Addr	Dest Addr	Port	Target Addr	Port
1	pass	10.0.1.2/24	*	*	--	--
2	pass	10.0.1.1/24	*	*	--	--
def auto	*	*	*	*	--	--

The changes take effect immediately. You must save your changes or they are lost upon reboot.

Port 7800 is included in the default pass-through rule that specifies the RBT-Proto port label. This configuration ensures that SteelHeads by default pass through and do not intercept SYN packets arriving on port 7800. These additional in-path pass-through rules are necessary only if the SteelHeads have been configured to use service ports other than 7800 for the SteelHead-to-SteelHead connections.

Configuring faster peer failure detection

A SteelHead uses the out-of-band (OOB) connection to inform a peer SteelHead of its capabilities. The OOB connection is also used to detect failures. By default, a SteelHead sends a keep-alive message every 20 seconds, and it declares a peer down after sending two keep-alive messages (40 seconds) and no response is received. If you want faster peer failure detection, use the following commands to adjust the interval and the number of keep-alive messages sent:

```
protocol connection wan keep-alive oob def-count (default of 2; minimum value of 2)
protocol connection wan keep-alive oob def-intvl (default of 20; minimum value of 5)
```

Losing the OOB connection does not affect the optimized sessions, because the optimized sessions have a one-to-one mapping between the outer channel (the LAN-side TCP connection between the client and server and the SteelHead) and the inner channel (the WAN-side TCP connection between the SteelHeads). The disadvantage to this approach is that the application does not notice when the peer is unavailable and the application might appear as if it is not working to the end user.

To address this you can disconnect the inner and outer channels when the SteelHead loses its OOB connection by using the **protocol connection lan on-oob-timeout drop all enable** command.

For SteelHeads with multiple in-path interfaces, the **protocol connection lan on-oob-timeout drop all enable** command disconnects all the optimized sessions, even if there are other OOB connections originating from other in-path interfaces. To configure the SteelHead to drop only the connections related to a specific in-path interface, use the **protocol connection lan on-oob-timeout drop same-inpath enable** command.

For more information about OOB, see [“Out-of-band connection” on page 84](#).

Configuring simplified routing

Simplified routing is only effective in topologies where the in-path SteelHead resides on a different network than the end hosts. We recommend that you use simplified routing in deployments where a Layer-3 switch separates the end hosts from the SteelHead.

For an overview of simplified routing and packet ricochet, see [“Overview of simplified routing” on page 60](#).

You can enable simplified routing with the **in-path simplified routing <option>** command or in the Management Console. Use the options in the following table to determine if simplified routing is enabled and what packet elements the SteelHead can use (that is, what is *learned*).

To configure simplified routing from the CLI

- Connect to the CLI and enter the following command:

```
in-path simplified routing <option>
```

The following table summarizes the simplified routing keywords that are available for this command.

Parameter	Definition
none	Does not collect mappings. This setting disables simplified routing learning. The none keyword is required for virtual in-path deployments.
dest-only	Destination only. Collects mappings from destination IP, destination MAC, and VLAN tag (when deployed on 802.1q trunk). We recommend that you use the dest-only keyword for most deployments with multiple in-paths or connection forwarding SteelHeads. Destination only is enabled by default on appliances manufactured with RiOS 6.0 or later. SteelHeads do not usually learn incorrect mappings unless the network devices themselves are routing incorrectly.
dest-source	Destination and source. Collects mappings from destination and source IP, destination and source MAC, and VLAN tag (when deployed on 802.1q trunk).
all	Collects mappings for destination and source IP, destination and source MAC, VLAN tag, and SteelHead inner connection traffic and autodiscovery options. This keyword has the advantage of learning simplified entries faster than the destination only. We recommend that you use the all keyword in topologies when you deploy the SteelHead a 802.1q trunk.

You can view the simplified routing table that includes the in-path interface learned information, the entry, IP address, MAC address, VLAN tag ID, and the times the entry was used.

To view the simplified routing table on the SteelHead

- Connect to the CLI and enter the following command to see the following output:

```
show in-path macmap-tables
relay      ip_addr      mac_addr      vlan  ref_count
inpath0_0  10.18.4.9      00:0d:66:95:e8:00  0      6
```

You can use the output of this command if an entry points to an incorrect MAC address, such as a firewall. We recommend that you collaborate with Riverbed Support for troubleshooting—the understanding of simplified routing learning in complex topologies can require detailed traffic flow analysis.

Simplified routing has the following constraints:

- The **none** keyword must be used when you configure simplified routing in a virtual in-path (WCCP/PBR/SteelHead Interceptor) environment.
- The default route must exist on the SteelHead.

In the following deployment examples, the recommended simplified routing settings are specified in the CLI configurations.

Multiple WAN router deployments

Typically, multiple WAN routers are used at locations where redundancy or high availability is important. With multiple routers, the loss of a single WAN link or a single WAN router does not prevent hosts at the locations from reaching WAN resources. SteelHeads can be deployed and configured to maintain the high availability for network access. Additionally, multiple SteelHeads can be deployed and configured so that a SteelHead failure allows new connections to be optimized.

This section describes the following tasks:

- [“Configuring multiple WAN router deployments without connection forwarding” on page 223](#)
- [“Configuring multiple WAN router deployments with connection forwarding” on page 228](#)

If one or more SteelHeads are deployed to cover all the links between the LAN switches and the WAN connecting routers, connection forwarding is not required. These deployments are referred to as *serial* deployments, and they can use multiple SteelHeads (in a [“Primary and backup deployments” on page 213](#) or [“Serial cluster deployments” on page 215](#)) to achieve optimization high availability.

If it is impossible or impractical to have all the WAN links covered by a single SteelHead, multiple SteelHeads are used. They must have connection forwarding configured. These deployments are known as *parallel* deployments. High availability for optimization is achieved by using the connection forwarding fail-to-block configuration, the primary and backup, or serial clustering on each of the parallel links to the WAN.

For more information about connection forwarding, see [“Connection forwarding” on page 56](#). For more information about fail-to-block mode, see [“Fail-to-block mode” on page 202](#).

We recommend that you use designs that do not require connection forwarding (that is, serial designs) whenever possible. Serial designs require less configuration, and are easier to troubleshoot, than parallel designs. If you need a parallel design, a deployment using the SteelHead Interceptor might have several advantages, including policy-based load balancing and failover handling.

Using the WAN-side or LAN-side HSRP IP address improves the likelihood that optimized connections survive a network outage. For this outcome, you must understand how the SteelHead learns and reacts to changes in the network. Keep in mind that there are many different network design possibilities, and it is not possible to explain all the caveats here.

By default, the SteelHead uses simplified routing, which learns the association between IP addresses and MAC addresses, and Address Resolution Protocol (ARP), which learns associations between IP addresses and MAC addresses on the same subnet as the in-path interface and MAC addresses.

The SteelHead also builds a table of MAC addresses to the LAN or WAN interface, based on the Ethernet frames that cross through the SteelHead in-path interface. Using these tables, the SteelHead learns which destination MAC address to use for packets the SteelHead originates and the corresponding interface on which to transmit the packet. For example, if local hosts are on the same subnet as the SteelHead in-path interfaces and the WAN routers are using HSRP, the transmitting in-path interface learns all local IP addresses on the same subnet to MAC address, through ARP, through the LAN interface. The SteelHead learns remote IP address-to-MAC address relationships from simplified routing = destination-only through the WAN interface. If the SteelHead has not learned the IP address-to-MAC address relationship through simplified routing or ARP, it follows its default gateway.

When the network experiences an outage, the SteelHead in-path interface that transmits for the connection does not react to the change in the network until there has been a change in the flow of traffic through the transmitting in-path interface. After the flow of traffic changes, the SteelHead in-path interface can learn that the destination is available through the opposite interface (WAN now goes to LAN). For example, using the default simplified routing setting, a SteelHead learns that a remote IP address is associated with the MAC address of the primary WAN router that owns the HSRP virtual MAC. When the primary router WAN circuit fails, the primary router can decrement its HSRP priority and the standby router can preempt the primary router to assume control of the HSRP virtual MAC. When this happens, the transmitting interface learns the HSRP MAC through the LAN interface and can continue transmitting traffic across the WAN for optimized connections.

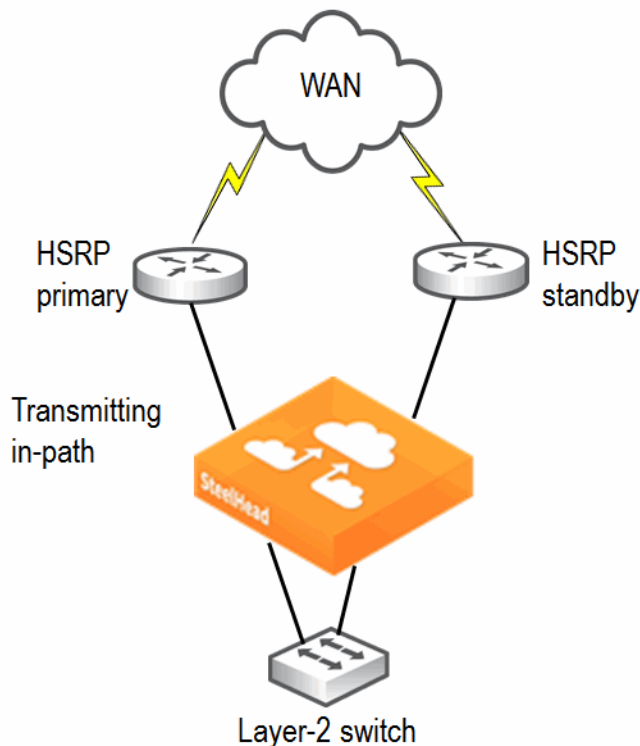
On the other hand, if the transmitting interface learned the remote IP-to-MAC relationship as the physical interface of the primary router then it needs to detect a packet ricochet to learn that the path across the WAN is actually through the LAN interface. Also, you might need to consider any TCP connections that the SteelHead originates, such as the OOB splice. You can use the **protocol connection lan on-oob-timeout drop** command.

In certain scenarios, existing optimized connections do not survive: for example, if there is a failure between the SteelHead and a directly connected device, such as the cable is damaged or a device lost power. When a directly connected device fails, the first-hop redundancy protocol such as HSRP detects the failure and a standby device can assume the primary role. However, some features on the SteelHead, such as link state propagation, can also detect the failure and stop connectivity in the associated interface (if the failure affects LAN0_0 then WAN0_0 also stops connectivity). The result is that all paths from the in-path interface that transmit for the optimized connection are in a down state and the optimized connections do not continue. Most applications restart new connections. Link state propagation provides feedback to other devices of the failure, and network protocols can more quickly detect the failure.

The choice of a default gateway is very important in locations with multiple WAN routers. In addition to choosing a default gateway (and simplified routing) that minimizes packet ricochet, HSRP or similar protocols can be used to ensure the loss of a single WAN router does not prevent the SteelHead from transmitting packets over the WAN. Most WAN devices that support HSRP or similar protocols have a *link tracking* option that allows them to relinquish the HSRP virtual IP address if a WAN link fails; this option should be used when possible.

Note: In a high-availability environment, there are often multiple gateways or next hops to choose from. To minimize the disruption to any existing optimized connections when a network device fails, it is important that the correct settings are configured on the SteelHeads.

Figure 9-10. HSRP diagram



Configuring multiple WAN router deployments without connection forwarding

This section describes best practices for serial SteelHead deployments at locations with multiple routers. Each of the following scenarios can be modified to use multiple SteelHeads, either in primary and backup or serial cluster configurations. This section discusses the following scenarios:

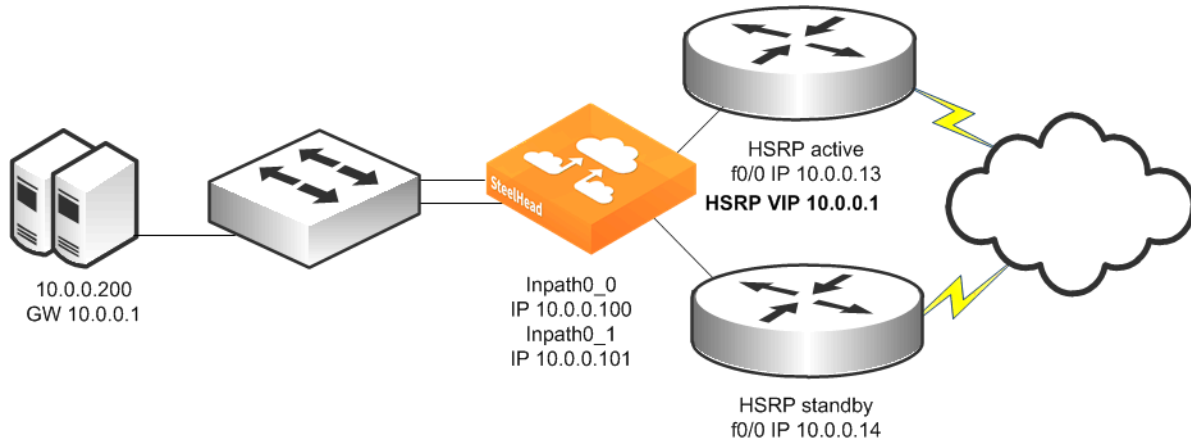
- [“Configuring a single SteelHead and single Layer-2 switch deployment” on page 224](#)
- [“Configuring a single SteelHead and dual Layer-2 switches deployment” on page 224](#)
- [“Configuring a single SteelHead and single Layer-3 switch deployment” on page 225](#)
- [“Configuring single SteelHead and dual Layer-3 switches deployment” on page 226](#)

Configuring a single SteelHead and single Layer-2 switch deployment

Figure 9-11 shows a topology consisting of two routers, a single Layer-2 switch, and one SteelHead with a 4-port card. The client and the SteelHead are in the same subnet. The client uses the HSRP virtual IP as its default gateway (10.0.0.1).

In this environment, the in-path gateway for both the inpath0_0 and inpath0_1 interfaces must point to the HSRP virtual IP (10.0.0.1). You do not need to enable simplified routing as the client is on the same subnet as the SteelHead.

Figure 9-11. Single SteelHead, single Layer-2 switch, dual router deployment



To configure a SteelHead, single Layer-2 switch, and dual routers

- Connect to the CLI and enter the following commands:

```

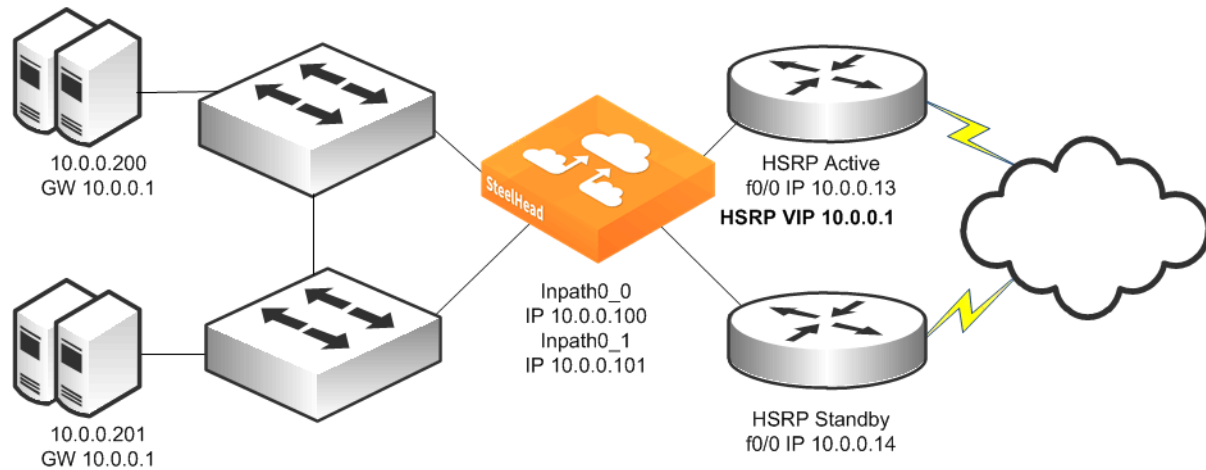
#--- Enable and configure the in-path interfaces.
in-path enable
in-path interface inpath0_1 enable
interface inpath0_0 ip address 10.0.0.100 /24
interface inpath0_1 ip address 10.0.0.101 /24
#--- Set the default gateway for the in-path interfaces to be the WAN
#--- side HSRP VIP.
ip in-path-gateway inpath0_0 "10.0.0.1"
ip in-path-gateway inpath0_1 "10.0.0.1"
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing is not required but can be enabled. (Simplified Routing
#--- destination only is on by default with new RiOS 6.x installs).
in-path simplified routing dest-only
  
```

Configuring a single SteelHead and dual Layer-2 switches deployment

Figure 9-12 shows a topology in which there are two routers, two Layer-2 switches, and one SteelHead with a 4-port card. The client and the SteelHead are in the same subnet. The client uses the HSRP virtual IP as its default gateway (10.0.0.1).

In this environment, the in-path gateway for both the inpath0_0 and inpath0_1 interfaces must point to the HSRP virtual IP (10.0.0.1). You do not need to enable simplified routing because the clients are on the same subnet as the SteelHead.

Figure 9-12. Single SteelHead, dual Layer-2 switches, dual router deployment



To configure a SteelHead, dual Layer-2 switches, and dual routers

- Connect to the CLI and enter the following commands:

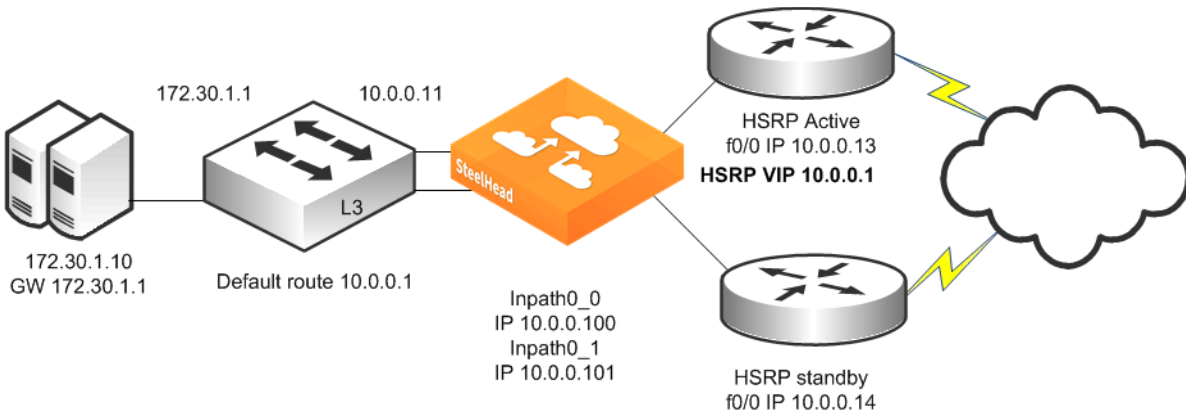
```
#--- Enable and configure the in-path interfaces.
in-path enable
in-path interface inpath0_1 enable
interface inpath0_0 ip address 10.0.0.100 /24
interface inpath0_1 ip address 10.0.0.101 /24
#--- Set the default gateway for the in-path interfaces to be the WAN
#--- side HSRP VIP.
ip in-path-gateway inpath0_0 "10.0.0.1"
ip in-path-gateway inpath0_1 "10.0.0.1"
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing is not required but can be enabled. (Simplified Routing
#--- destination only is on by default with new RiOS 6.x installs).
in-path simplified routing dest-only
```

Configuring a single SteelHead and single Layer-3 switch deployment

Figure 9-13 shows a topology in which there are two routers, a single Layer-3 switch, and a single SteelHead with a 4-port card. The client and the SteelHead are in different subnets. The client is using the Layer-3 switch as its default gateway. The Layer-3 switch does not have any routing protocols configured and relies on the default route to reach other subnets. The default route uses the HSRP IP address as the next hop.

In this environment, the in-path gateway on the inpath0_0 and inpath0_1 interface must use the Layer-3 switch as its default gateway (10.0.0.11) while configuring simplified routing to populate its table based on destination MAC address (**in-path simplified routing dest-only** command).

Figure 9-13. Single SteelHead, single Layer-3 switch, static routing, dual router deployment



To configure a SteelHead, single Layer-3 switch, static routing, and dual routers

- Connect to the CLI and enter the following commands:

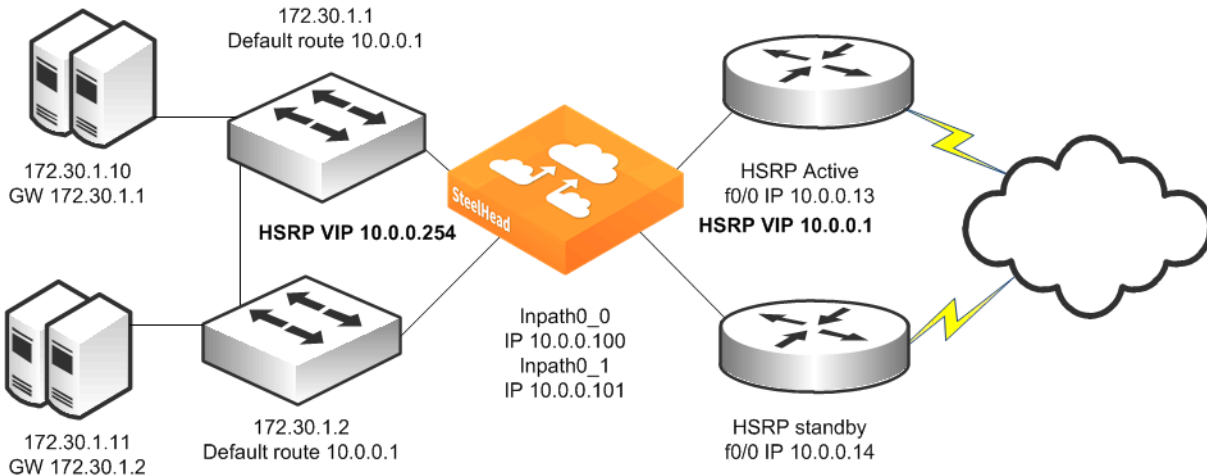
```
#--- Enable and configure the in-path interfaces.
in-path enable
in-path interface inpath0_1 enable
interface inpath0_0 ip address 10.0.0.100 /24
interface inpath0_1 ip address 10.0.0.101 /24
#--- Set the default gateway for the in-path interfaces to be the LAN
#--- side Layer-3 Switch IP.
ip in-path-gateway inpath0_0 "10.0.0.11"
ip in-path-gateway inpath0_1 "10.0.0.11"
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
```

Configuring single SteelHead and dual Layer-3 switches deployment

Figure 9-14 shows a topology in which there are two routers, two Layer-3 switches, and a single SteelHead with a 4-port card. The clients and the SteelHead are in different subnets. The clients are using the Layer-3 switches as their default gateways. The Layer-3 switches do not have any routing protocols configured and relies on the default route to reach other subnets. The default route uses the HSRP IP address as the next hop.

In this environment, the in-path gateway on the inpath0_0 and inpath0_1 interface must use the HSRP address of the Layer-3 switches as its default gateway (10.0.0.254) while configuring simplified routing to populate its table based on destination MAC address (**in-path simplified routing dest-only** command).

Figure 9-14. Single SteelHead, dual Layer-3 switches, dual HSRP, static routing, dual router deployment



To configure a SteelHead, dual Layer-3 switches, dual HSRP, static routing, and dual routers

Connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
in-path interface inpath0_1 enable
interface inpath0_0 ip address 10.0.0.100 /24
interface inpath0_1 ip address 10.0.0.101 /24
#--- Set the default gateway for the in-path interfaces to be the LAN
#--- side Layer-3 switch HSRP VIP.
ip in-path-gateway inpath0_0 "10.0.0.254"
ip in-path-gateway inpath0_1 "10.0.0.254"
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
```

Configuring multiple WAN router deployments with connection forwarding

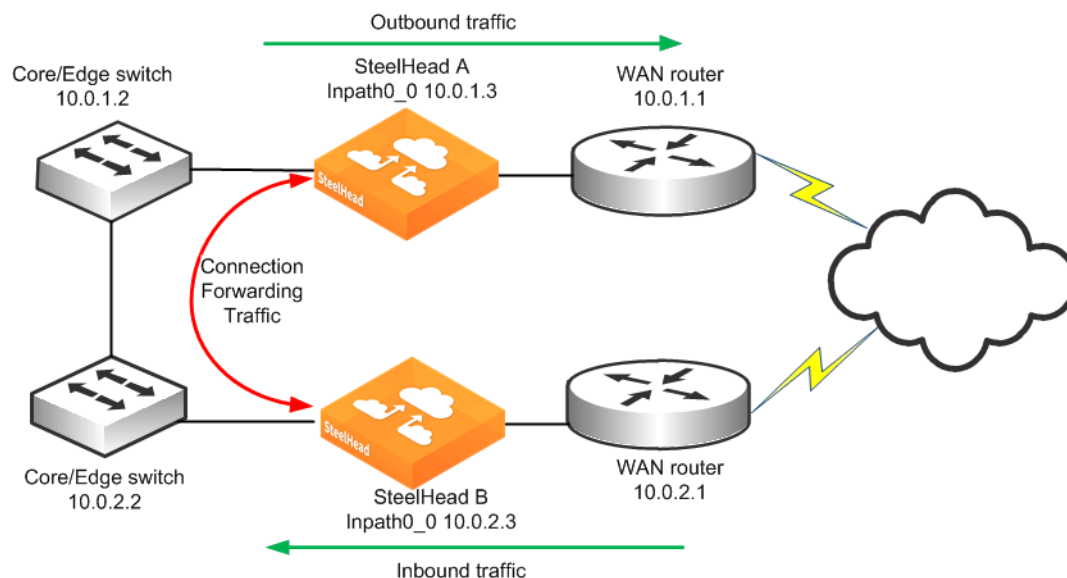
This section describes best practices for parallel SteelHead deployments at locations with multiple routers. Each of the scenarios that follow can be modified to use additional SteelHeads for each path to the WAN, using either the primary and backup or serial cluster configurations. If you are using multiple SteelHeads on each path, every SteelHead at the location must be configured as a connection-forwarding neighbor for every other SteelHead at the location. This section covers the following scenarios:

- “Configuring basic connection forwarding” on page 228
- “Configuring connection forwarding with allow-failure and fail-to-block” on page 229
- “Configuring a dual SteelHead and dual Layer-2 switches deployment” on page 231
- “Configuring a dual SteelHead and dual Layer-3 switches deployment” on page 232
- “Configuring a dual SteelHead with multiple in-path deployment” on page 234

Configuring basic connection forwarding

This example requires you to have configured your cabling and duplex according to the recommendations described in “Cabling and duplex” on page 205.

Figure 9-15. Physical in-path deployment with connection forwarding



This example makes the following assumptions:

- Connection forwarding is enabled by configuring the in-path0_0 IP address of the two SteelHeads as neighbors.
- When one of the SteelHeads fails, the neighbor SteelHead stops attempting to optimize new connections until the down SteelHead recovers or is replaced.
- Simplified routing removes any packet ricochet that might occur when the SteelHead sends traffic to remote SteelHeads.

To configure connection-forwarding multiple WAN routers

1. On SteelHead A, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.1.3 /24
#--- Set the default gateway for the in-path interface to be the LAN
#--- side Layer-3 switch.
ip in-path-gateway inpath0_0 10.0.1.2
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.2.3.
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadB main-ip 10.0.2.3
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.2.3 /24
#--- Set the default gateway for the in-path interface to be the LAN
#--- side Layer-3 switch.
ip in-path-gateway inpath0_0 10.0.2.2
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.1.3.
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadA main-ip 10.0.1.3
```

Note: These steps do not include the configuration of features such as duplex, alarms, and DNS.

For information about connection forwarding, see [“Connection forwarding” on page 56](#).

Configuring connection forwarding with allow-failure and fail-to-block

This example requires you to have configured your cabling and duplex according to the recommendations described in [“Cabling and duplex” on page 205](#).

The following example represents the minimum steps required to configure a SteelHead deployment in which connection forwarding is configured and both the **fail-to-block** and **allow-failure** commands are enabled. This example does not include configuration instruction for features such as the management interface, DNS, and SNMP.

This example makes the following assumptions:

- Connection forwarding is enabled by configuring the in-path0_0 IP address of the two SteelHeads as neighbors.
- Fail-to-block option is enabled. (This option is not supported with all in-path hardware and SteelHead models.)
- The **allow-failure** command is enabled. This specifies that a SteelHead B continues to optimize new connections, if SteelHead A down.
- Simplified routing removes any packet ricochet that might occur when the SteelHead sends traffic to remote SteelHeads.

To configure connection forwarding with multiple WAN routers, allow-failure, and fail-to-block

1. On SteelHead A, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.1.3 /24
#--- Set the default gateway for the in-path interface to be the LAN
#--- side Layer-3 switch.
ip in-path-gateway inpath0_0 10.0.1.2
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.2.3
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadB main-ip 10.0.2.3
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0.
no interface inpath0_0 fail-to-bypass enable
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.2.3 /24
#--- Set the default gateway for the in-path interface to be the LAN
#--- side Layer-3 switch.
ip in-path-gateway inpath0_0 10.0.2.2
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default starting with RiOS 6.x
in-path simplified routing dest-only
```

```
#--- Enable Connection Forwarding to neighbor 10.0.1.3
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadA main-ip 10.0.1.3
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0.
no interface inpath0_0 fail-to-bypass enable
```

Note: These steps do not include the configuration of features such as duplex, alarms, and DNS.

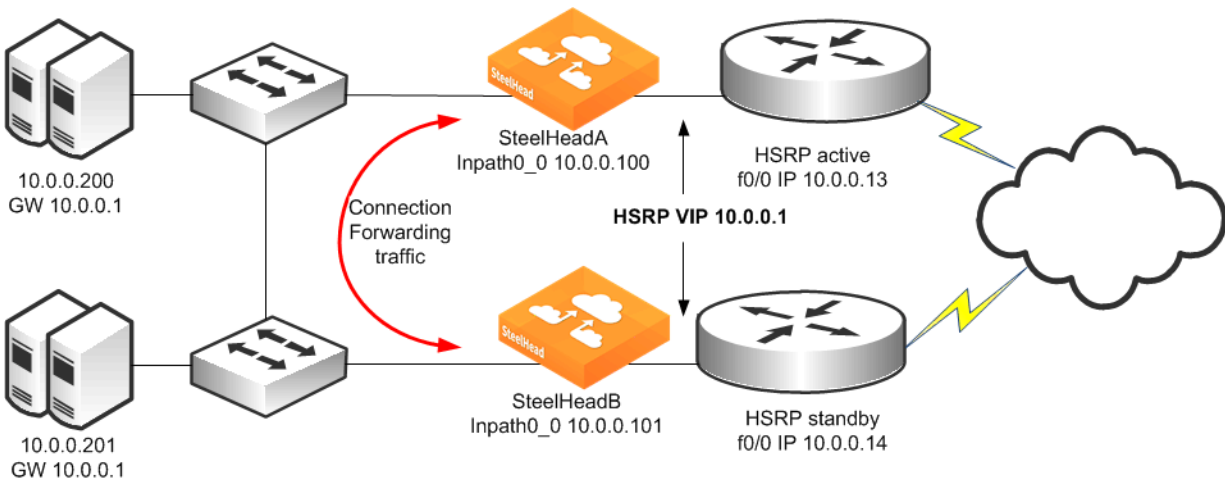
For information about connection forwarding, see [“Connection forwarding” on page 56](#).

Configuring a dual SteelHead and dual Layer-2 switches deployment

Figure 9-16 shows a topology in which there are two routers, two Layer-2 switches, and two SteelHeads at the remote location. The client and the SteelHeads are all in the same subnet. The client uses the HSRP virtual IP as its default gateway (10.0.0.1).

In this environment, the in-path gateway on both SteelHeads must point to the HSRP virtual IP (10.0.0.1). You do not need to enable simplified routing because the client is on the same subnet as the SteelHead. You must configure connection forwarding between the two SteelHeads. The connection forwarding path must use the LAN interface of the SteelHeads.

Figure 9-16. Dual SteelHeads, dual Layer-2 switches, dual router deployment



To configure dual SteelHeads, dual Layer-2 switches, and dual routers

1. On SteelHead A, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.0.100 /24
#--- Set the default gateway for the in-path interfaces to be the WAN
#--- side HSRP VIP.
ip in-path-gateway inpath0_0 10.0.0.1
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
```

```

in-path peering auto
#--- Simplified Routing is not required but can be enabled. (Simplified Routing
#--- destination only is on by default with new RiOS 6.x installs).
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.0.101
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadB main-ip 10.0.0.101
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0.
no interface inpath0_0 fail-to-bypass enable

```

2. On SteelHead B, connect to the CLI and enter the following commands:

```

#--- Enable and configure the in-path interfaces
in-path enable
interface inpath0_0 ip address 10.0.0.101 /24
#--- Set the default gateway for the in-path interfaces to be the WAN
#--- side HSRP VIP.
ip in-path-gateway inpath0_0 10.0.0.1
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing is not required but can be enabled. (Simplified Routing
#--- destination only is on by default with new RiOS 6.x installs).
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.0.100
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadA main-ip 10.0.0.100
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0.
no interface inpath0_0 fail-to-bypass enable

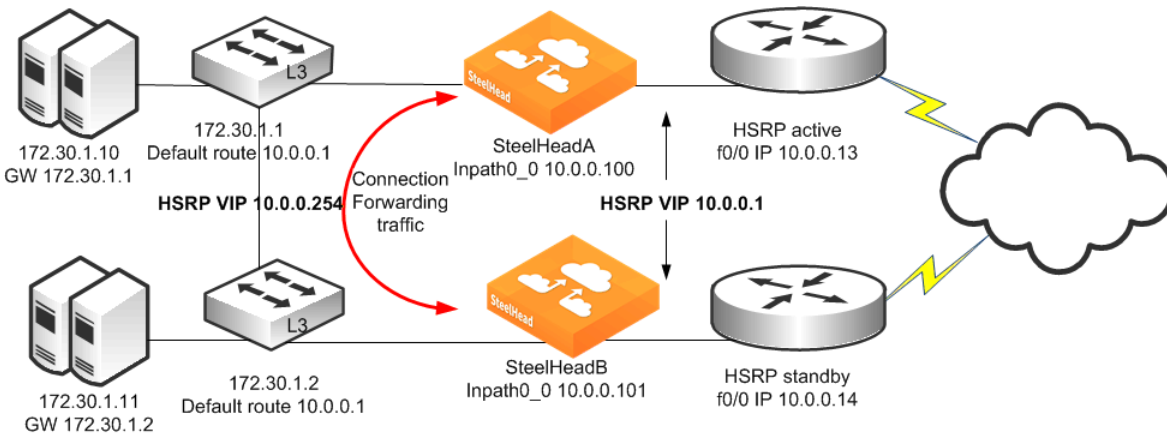
```

Configuring a dual SteelHead and dual Layer-3 switches deployment

Figure 9-17 shows a topology in which there are two routers, two Layer-3 switches, and two SteelHeads at the remote location. The clients and the SteelHeads are in different subnets. The clients use the Layer-3 switch as the default gateway. The Layer-3 switch does not have any routing protocols configured and relies on the default route to reach other subnets. The default route uses the HSRP IP address as the next hop.

In this environment, the in-path gateway on both SteelHeads must use the HSRP address of the Layer-3 switches as its default gateway (10.0.0.254) while configuring simplified routing to populate its table based on destination MAC address (**in-path simplified routing dest-only** command). You must configure connection forwarding between the two SteelHeads. The connection forwarding path must use the LAN interface of the SteelHeads.

Figure 9-17. Dual SteelHeads, dual Layer-3 switches, static routing, dual router deployment



To configure dual SteelHeads, dual Layer-3 switches, static routing, and dual routers

1. On SteelHead A, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.0.100 /24
#--- Set the default gateway for the in-path interfaces to be the LAN
#--- side Layer-3 switch HSRP VIP.
ip in-path-gateway inpath0_0 10.0.0.254
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.0.101
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadB main-ip 10.0.0.101
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0.
no interface inpath0_0 fail-to-bypass enable
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.0.0.101 /24
#--- Set the default gateway for the in-path interfaces to be the LAN
#--- side Layer-3 switch HSRP VIP.
ip in-path-gateway inpath0_0 10.0.0.254
```

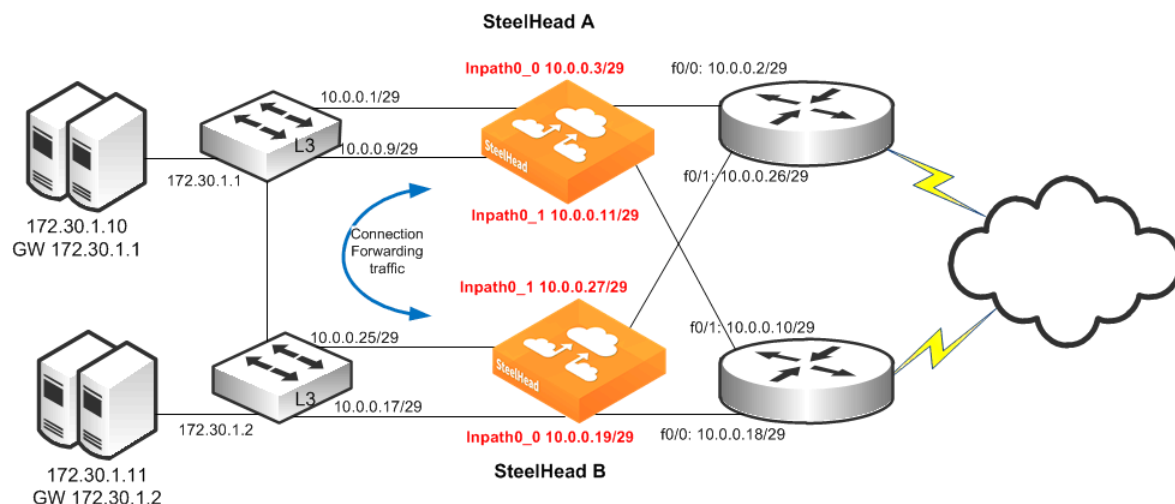
```
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.0.0.100
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadA main-ip 10.0.0.100
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0.
no interface inpath0_0 fail-to-bypass enable
```

Configuring a dual SteelHead with multiple in-path deployment

Figure 9-18 shows a topology similar to Figure 9-17, but each router has two links going back to the Layer-3 switches totaling four links. Each link between the router and switch are independent /29 bit networks. You can configure dynamic routing protocols so that traffic can flow inbound or outbound. You can use this network design to prevent the loss of a single Layer-3 switch from cutting connectivity to the attached router. To ensure traffic is routed to the partner SteelHead during a failure, you can use the two in-path interfaces and deploy the SteelHeads in parallel, with connection forwarding, and fail-to-block.

The clients and the SteelHeads are in different subnets and simplified routing is enabled (**in-path simplified routing dest-only**). To ensure connection forwarding traffic is sent to the LAN side, we recommend that you configure each in-path interface default gateway to point to the LAN side, the Layer-3 switch.

Figure 9-18. Dual SteelHeads with multiple in-path deployment



To configure dual SteelHeads with multiple in-path interfaces

1. On SteelHead A, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
in-path interface inpath0_0 enable
interface inpath0_0 ip address 10.0.0.3 /29
in-path interface inpath0_1 enable
interface inpath0_1 ip address 10.0.0.11 /29
#--- Set the default gateway for the in-path interfaces to be the LAN-side Layer-3 Switch.
ip in-path-gateway inpath0_0 10.0.0.1
ip in-path gateway inpath0_1 10.0.0.9
#--- Enable enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- simplified routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to SteelHead neighbor, specifying both neighbors
#--- in-paths.
#--- Enable multi-interface support.
#--- Allow-failure allows the SteelHead to continue optimizing traffic even
#--- if neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadB main-ip 10.0.0.19
steelhead name SteelHeadB additional-ip 10.0.0.27
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0 and inpath0_1.
no interface inpath0_0 fail-to-bypass enable
no interface inpath0_1 fail-to-bypass enable
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
in-path interface inpath0_0 enable
interface inpath0_0 ip address 10.0.0.19 /29
in-path interface inpath0_1 enable
interface inpath0_1 ip address 10.0.0.27 /29
#--- Set the default gateway for the in-path interfaces to be the LAN-side L3 switch.
ip in-path-gateway inpath0_0 10.0.0.17
ip in-path gateway inpath0_1 10.0.0.25
#--- Enable Enhanced autodiscovery, enabled by default on new factory installs of
#--- RiOS 6.x. If you have upgraded a SteelHead to that level or more recently, you
#--- will need to apply this command.
in-path peering auto
#--- Simplified Routing destination only should be used and
#--- is on by default with new RiOS 6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to SteelHead neighbor, specifying both neighbors
#--- in-paths.
#--- Enable multi-interface support.
#--- Allow-failure allows the SteelHead to continue optimizing traffic even if
#--- neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHeadA main-ip 10.0.0.3
steelhead name SteelHeadA additional-ip 10.0.0.11
steelhead communication allow-failure
#--- Enable fail-to-block on inpath0_0 and inpath0_1.
no interface inpath0_0 fail-to-bypass enable
```

```
no interface inpath0_1 fail-to-bypass enable
```

You can configure and deploy routers in an endless number of variations, for example:

- The links between Layer-3 switches and routers can be Layer-2 instead of separate Layer-3 networks.
- Layer-3 switches that are WAN distribution switches connected to a pair of core switches, then connected to other distribution switches before connecting to the end hosts.
- Instead of two routers, you can use four routers connecting to various WAN providers for a total of eight connections to the Layer-3 switches.
- You can use static routes, or any number of routing protocols, to balance traffic across multiple paths.

Regardless of the variations, the same deployment logic applies. The SteelHeads must cover all the desired paths of traffic to be optimized. You can use up to 10 supported in-path interfaces on the larger SteelHeads. If you have parallel SteelHeads, follow the same simplified routing and connection forwarding guidelines in this section. For information about serial clusters, see [“Serial cluster deployments” on page 215](#).

802.1Q trunk deployments

This section describes the use of virtual LANs (VLANs) and 802.1Q, which allows multiple logical networks to span a single physical link. IEEE 802.1Q is a networking standard that allows multiple bridged networks to transparently share the same physical network. IEEE 802.1Q is also referred to as *VLAN Tagging* and *dot1q*.

This section includes the following topics:

- [“Overview of VLAN trunk” on page 237](#)
- [“Configuring a SteelHead on an 802.1Q trunk link” on page 238](#)
- [“Capturing network traces using tcpdump” on page 239](#)

The SteelHead does not support overlapping IP address spaces, even if the overlapping IPs are kept separate through VLAN tags.

For information about alternative configurations, see [“VPN Routing and Forwarding” on page 361](#).

Overview of VLAN trunk

A SteelHead can be deployed physically in-path on an 802.1Q trunk link, and it can optimize connections where packets have been tagged with an 802.1Q header. As in other physical in-path deployments, the SteelHeads in-path interface must be configured with an IP address and a default gateway. If the SteelHeads in-path IP address is in a subnet whose traffic is normally tagged when present on the in-path link, the SteelHead's in-path interface must be configured with the VLAN for that subnet. This configuration allows the SteelHead to appropriately tag packets transmitted from the in-path interface that use the in-path IP address as the source address. The SteelHead can optimize traffic on the VLANs different from the VLAN containing the in-path IP address.

SteelHeads can be deployed across multiple 802.1Q trunk links. Each in-path interface requires:

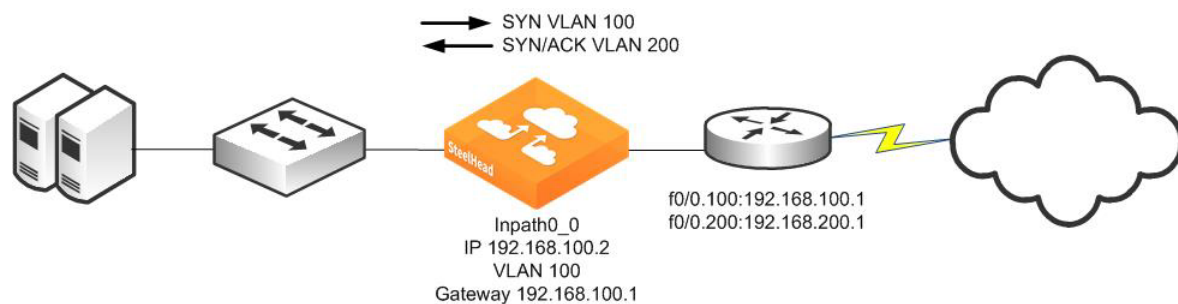
- an IP address.
- a default gateway.
- a VLAN ID (if required for the in-path IP address subnet).

SteelHeads configured as connection-forwarding neighbors can be deployed on 802.1Q trunk links.

SteelHeads maintain the VLAN ID when transmitting packets for the LAN side of optimized connections. If correct addressing or port transparency is used, the SteelHead uses the configured in-path VLAN ID when transmitting packets towards the WAN. When using the full address transparency WAN visibility mode (for details, see [“Full address transparency” on page 67](#)), SteelHeads maintain the VLAN ID (along with IP address and TCP ports) when transmitting packets on the WAN side of optimized connections. The SteelHead can maintain the VLAN IDs even if there is a difference between the packets going to the WAN and those returning from the WAN. You do not have to configure the VLAN ID on the in-path interface to match any of those seen for optimized connections.

For example, assume that SteelHead is configured for full address transparency. The SYN packet for a TCP connection traveling toward the WAN has a VLAN ID of 100, and the SYN/ACK packet returning from the WAN has a VLAN of 200. If a SteelHead optimizes this connection, then the SteelHead transmits packets toward the LAN using VLAN ID 200. The SteelHead monitors the MAC addresses associated with each VLAN.

Figure 9-19. Example of basic VLAN trunk deployment



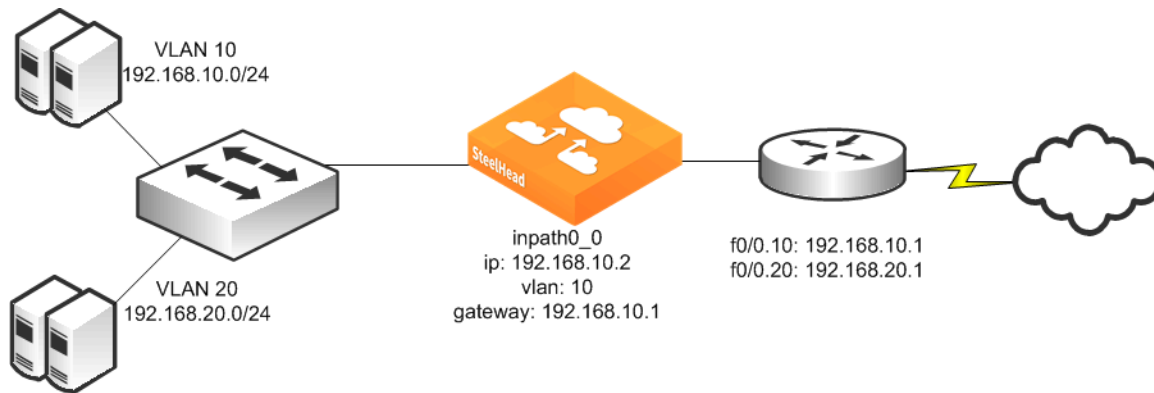
Maintaining the VLAN IDs in this manner requires using the **vlan-conn-based** and **mac-match-vlan** commands, which are enabled by default in RiOS 6.0 or later, and the other commands listed in the following configuration example. For details, see [“Configuring a SteelHead on an 802.1Q trunk link” on page 238](#).

In-path rules can use VLAN tags as matching parameters, which allows you to control optimization based on VLAN tags, along with other information such as IP addresses, subnets, or TCP ports.

Configuring a SteelHead on an 802.1Q trunk link

Figure 9-20 shows a SteelHead deployed physically in-path on an 802.1Q trunk link. Two VLANs are present on the in-path link: VLAN 10, which contains subnet 192.168.10.0/24, and VLAN 20, which contains subnet 192.168.20.0/24. The SteelHead is given an in-path IP address in the 192.168.10.0/24 subnet. The SteelHead is configured to use VLAN 10 for its in-path interface and to use the routers subinterface IP address as its default gateway. Even though the SteelHead has an in-path IP address in subnet 192.168.10.0/24 and VLAN 10, it can optimize traffic in VLAN 20.

Figure 9-20. SteelHead deployed physically in-path on an 802.1q trunk link



To deploy a SteelHead physically in-path on an 802.1Q trunk link

1. On the SteelHead, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 192.168.10.2 /24
#--- Set the default gateway for the inpath0_0 interface to be the WAN
#--- side router VLAN 10 interface.
ip in-path-gateway inpath0_0 192.168.10.1
#--- Assign VLAN 10 to the inpath0_0 interface.
in-path interface inpath0_0 vlan 10
#--- Enable Simplified Routing All. (Simplified Routing destination only is on
#--- by default with new RiOS 6.x installs).
in-path simplified routing all
#--- New factory installs of RiOS 6.0.1 and later will have all of the following
#--- configuration parameters set already. If you have upgraded a SteelHead to that
#--- level or more recent, you will need to apply some or all of the following.
#--- Enable Enhanced autodiscovery.
in-path peering auto
#--- Ensure the SteelHead performs autodiscovery probing for the
#--- FTP and MAPI data channels.
in-path probe-ftp-data
in-path probe-mapi-data
#--- Allow static routes to take precedence over Simplified Routing.
in-path simplified mac-def-gw-only
#--- Ensure SteelHead transmits to LAN hosts with the same vlan tags as received.
in-path mac-match-vlan
in-path vlan-conn-based
#--- Ensure autodiscovery probing happens whenever possible, to learn vlan to IP mappings.
no in-path peer-probe-cach
no in-path probe-caching enable
#--- Ensure interoperability with remote SteelHeads whose in-path address is
#--- in the same subnet or vlan
in-path mac-except-locl
```

Capturing network traces using tcpdump

Use caution when using network traces on trunk links. If you configure a TCP dump filter that restricts the captured packets to a specified set (for example, based on IP addresses or ports), by default tcpdump does not capture packets with an 802.1Q tag.

To capture packets with an 802.1Q tag, you must prefix the filter string with the keyword **vlan**. Enter the **tcpdump -i wanX_Y vlan and host** command.

For details, see the *Riverbed Command-Line Interface Reference Manual*.

Layer-2 WAN deployments

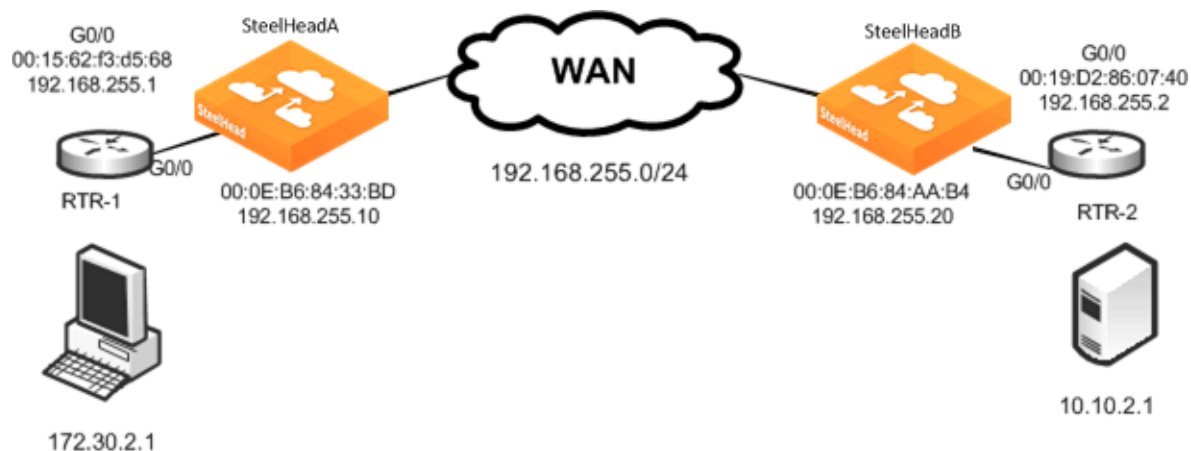
The following sections describe the types of Layer-2 WANs:

- [“Layer-2 WANs” on page 239](#)
- [“Broadcast Layer-2 WANs” on page 240](#)

Layer-2 WANs

On a Layer-2 WAN, we recommend setting the LAN router closest to the SteelHead as its in-path default gateway. For example, in [Figure 9-21](#), SteelHeadA must use 192.168.255.1 as its in-path default gateway and SteelHeadB must use 192.168.255.2 as its in-path default gateway.

Figure 9-21. SteelHeads deployed in a Layer-2 WAN



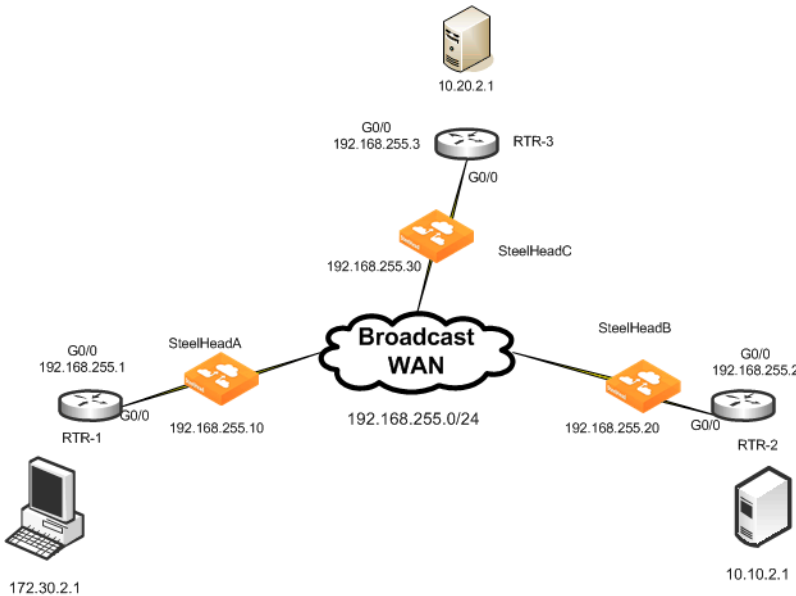
If the links to the WAN carry 802.1Q tagged packets, and if the WAN service provider routes packets based on 802.1Q tags, it is necessary to configure the SteelHeads for 802.1Q deployment and to use the full address transparency mode.

For information about full address transparency mode, see [“Full address transparency” on page 67](#).

Broadcast Layer-2 WANs

A Layer-2 broadcast network over the WAN is very similar to a Layer-2 WAN. However, a Layer-2 broadcast WAN behaves like a hub, whereby packets from each site are replicated over the WAN to all the other sites. For example, in [Figure 9-22](#), Router 2 can detect traffic between 172.30.2.1 and 10.10.2.1.

Figure 9-22. A Layer-2 broadcast WAN deployment



When deploying SteelHeads in a Layer-2 broadcast WAN, you must ensure that the correct SteelHead is handling the traffic. Furthermore, a Layer-2 broadcast WAN is not compatible with simplified routing; therefore, you must use static routes to avoid packet ricochet.

In [Figure 9-22](#), when 172.30.2.1 sends a SYN packet toward 10.10.2.1, SteelHeadA adds its probe option to the SYN packet (SYN+). The correct SteelHead that responds to this packet is SteelHeadB, because it is closest to the 10.10.2.1 server. However, because of the Layer-2 broadcast WAN, SteelHeadC would also detect the SYN+ packet and respond to SteelHeadA. This behavior creates two probe responses (SYN/ACK+) from two separate SteelHeads.

If the latency between SteelHeadA and SteelHeadC is lower than that of SteelHeadA and SteelHeadB, then SteelHeadA receives the probe response from SteelHeadC first. When the probe response SteelHead B arrives at SteelHeadA, SteelHeadA ignores the probe response, because it has already received a probe response from SteelHeadC.

This situation is undesirable, because SteelHeadA must be peering with SteelHeadB and not SteelHeadC when trying to reach the 10.10.2.1 server. To avoid this situation, enter the **in-path broadcast support enable** command.

When broadcast support is enabled, the SteelHead checks its routing table to see whether it uses its LAN or WAN interface to reach the destination IP. If the destination IP is reachable through the LAN, the appliance sends a probe response to the sender. Otherwise, it simply ignores the probe request.

Alternatively, you can use a fixed-target rule to define the SteelHead peers. However, fixed-target rules might not be scalable for larger deployments.

VLAN bridging deployments

This section describes the use of virtual LAN (VLAN) bridging. This section includes the following topics:

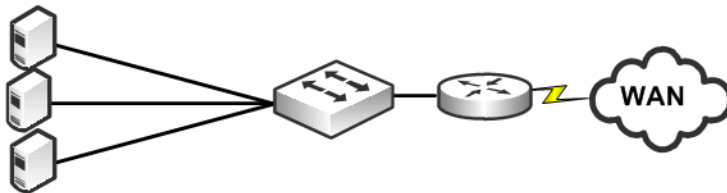
- “Overview of VLAN bridging deployment” on page 241
- “VLAN bridging considerations” on page 242
- “VLAN bridging variations” on page 242

Overview of VLAN bridging deployment

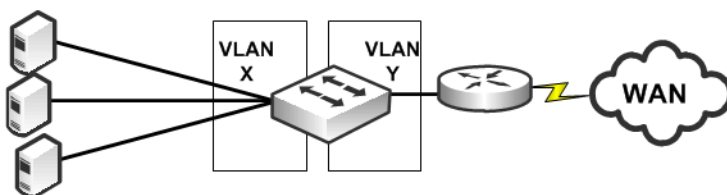
The term *VLAN bridging* refers to a network design in which both the LAN and WAN ports of a SteelHead's in-path interface are connected to a single switch or router. The switch or router is then configured so that traffic to be optimized must pass through the SteelHead—by forcing the traffic's Layer-2 path to or from the WAN to pass through the in-path interface. VLAN bridging is useful in network environments in which it is difficult to install a SteelHead physically in-path. For example, if fiber interfaces are needed for a physical in-path installation, but only copper interfaces are available on the SteelHead, you can use VLAN bridging as a simpler alternative to WCCP or PBR.

Figure 9-23 shows the principles of VLAN bridging. An existing switch or router is divided into two separate VLANs, and the SteelHead's LAN and WAN interfaces are used as the Layer-2 bridge that connects the VLANs.

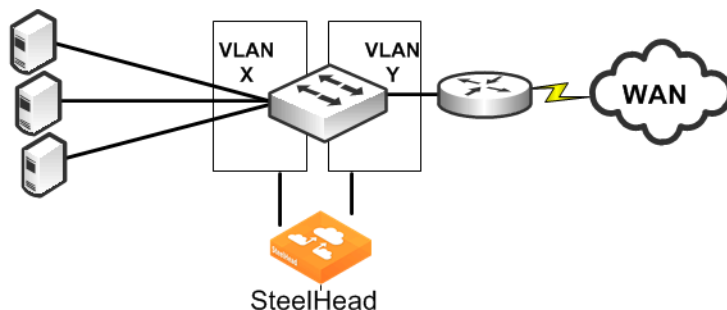
Figure 9-23. VLAN bridging principles



Switch divided into 2 VLANs



VLANs Bridged



VLAN bridging considerations

A VLAN bridging deployment has the same features and functionality as a physically in-path SteelHead, with the exception of 802.1Q VLAN trunking.

Consider the following guidelines when you use a VLAN bridging deployment:

- You can use an 802.1Q trunk with VLAN bridging between multiple VLANs on the same in-path interface, but doing this requires switch-specific features. For information about multiple VLANs on the same in-path interface, see [“Multiple VLAN bridging with VLAN mapping” on page 243](#).
- Use the same cables for the WAN and LAN interfaces—the same as you use for physical in-path deployments. For information about cables, see [“Cabling and duplex” on page 205](#).
- The switch detects the same MAC addresses in two different VLANs. Because most switches have separate MAC address tables per VLAN (independent VLAN learning, or IVL), some older switches can have only one MAC table for all VLANs (shared VLAN learning). Use only switches that have IVL with VLAN bridging.
- Verify that the switch allows access to its management IP address from multiple VLANs. Avoid using a switch whose management IP is only reachable from the default VLAN, because doing this prevents managing the switch. Some switches assign their management IP address to the default VLAN and cannot be altered—for example, the Cisco 2950 switch.

VLAN bridging variations

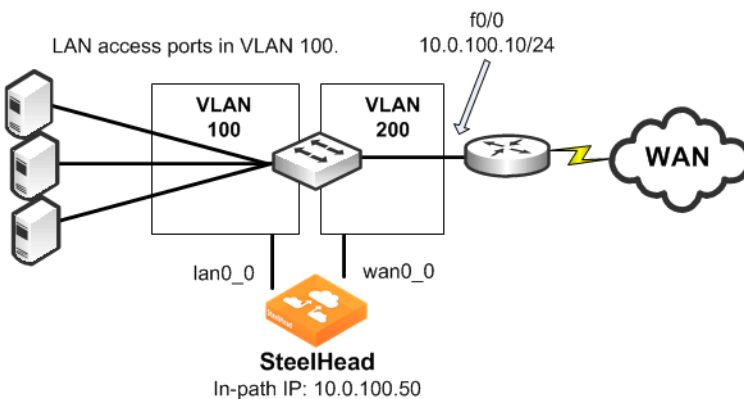
The variations of VLAN bridging are as follows:

- [“Layer-2 VLAN bridging” on page 242](#)
- [“Layer-3 VLAN bridging” on page 243](#)
- [“Multiple VLAN bridging with VLAN mapping” on page 243](#)

Layer-2 VLAN bridging

In a Layer-2 VLAN bridging deployment, the SteelHead is connected by VLANs on the Layer-2 switch. All traffic is bridged through the SteelHeads as it passes to and from the WAN routers. [Figure 9-24](#) shows a Layer-2 VLAN bridging deployment.

Figure 9-24. Layer-2 VLAN bridging



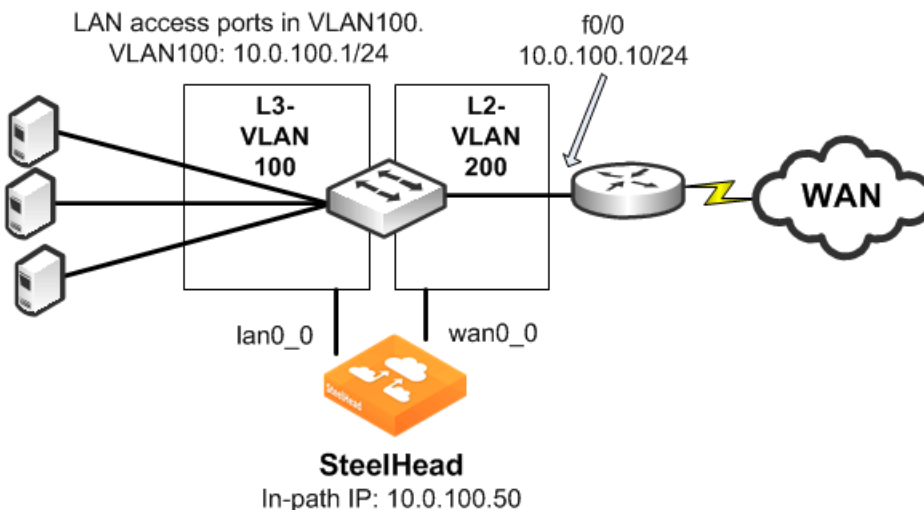
Note the following:

- VLAN 100 and VLAN 200 are Layer-2 VLANs.
- The default gateway of the hosts on the LAN must point to the router interface IP address.
- VLAN 100 contains the switch ports of the hosts and the switch port connected to the lan0_0 interface of the SteelHead.
- VLAN 200 contains the switch ports, the router, and the wan0_0 interface of the SteelHead.
- The default gateway of the SteelHead is the IP address of the WAN router.

Layer-3 VLAN bridging

In a Layer-3 VLAN bridging deployment, the SteelHead is connected across Layer-3 and Layer-2 VLANs on a Layer-2/Layer-3 switch. All traffic is switched through the SteelHead as it passes to and from the WAN router. [Figure 9-25](#) shows a Layer-3 VLAN bridging deployment.

Figure 9-25. Layer-3 VLAN bridging



Note the following:

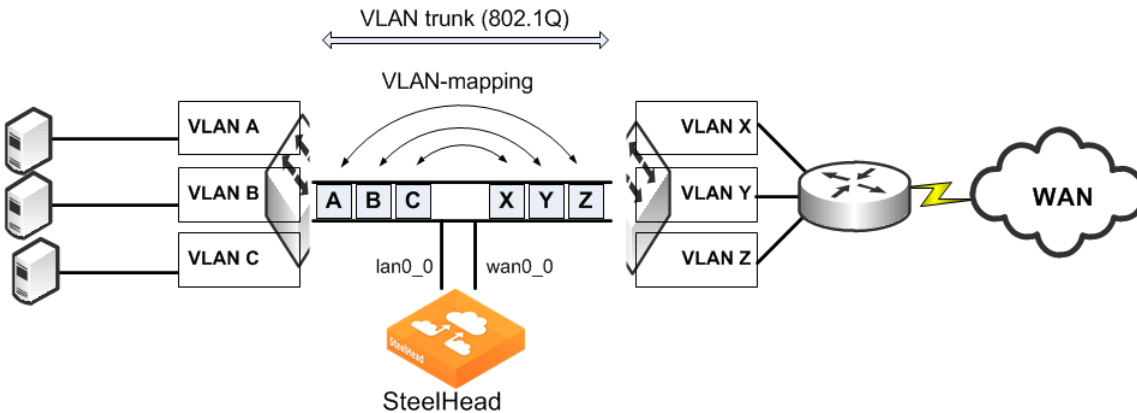
- Hosts on the VLAN 100 must point to VLAN 100 IP address as the default gateway.
- VLAN 100 contains the switch ports of the hosts, and the switch port connected to the lan0_0 interface of the SteelHead.
- VLAN 200 contains the switch ports, the router, and the wan0_0 interface that the SteelHead is connected to.
- The default gateway of the SteelHead is the IP address of the WAN router.

Multiple VLAN bridging with VLAN mapping

A limitation in Layer-2 VLAN bridging and Layer-3 VLAN bridging is that a single in-path interface can only bridge two different VLANs. If the SteelHead-connected switch ports are configured to be 802.1Q trunks (so that many VLANs can be sent or received), the switch does not bridge traffic for the VLANs across the ports.

To connect to multiple VLANs, you need a switch that supports *VLAN mapping* (also referred to as *VLAN translation* or *VLAN normalization*, depending on the switch vendor). VLAN mapping allows a trunk interface to change the 802.1Q tag. You must configure the switch with the mapping of one VLAN tag (used on the LAN side of the SteelHead) to another VLAN tag (used on the WAN side of the SteelHead) for packets to be sent or received. **Figure 9-26** shows multiple VLAN bridging with VLAN mapping deployment.

Figure 9-26. Multiple VLAN bridging with VLAN mapping



The following functionality makes it possible to optimize multiple VLANs with VLAN bridging:

- The VLAN mapping function on a switch changes the VLAN tags.
- VLAN mapping takes 802.1Q tagged traffic from an incoming trunk switch-port and maps it to a different local VLAN.

For information about configuring your specific hardware, refer to the documentation provided with your switch.

Virtual In-Path Deployments

This chapter describes virtual in-path deployments and summarizes the basic steps for configuring an in-path, load-balanced, Layer-4 switch deployment.

This chapter includes the following sections:

- [“Overview of virtual in-path deployment” on page 245](#)
- [“Configuring an in-path, load-balanced, Layer-4 switch deployment” on page 246](#)
- [“Configuring flow data exports in virtual in-path deployments” on page 248](#)

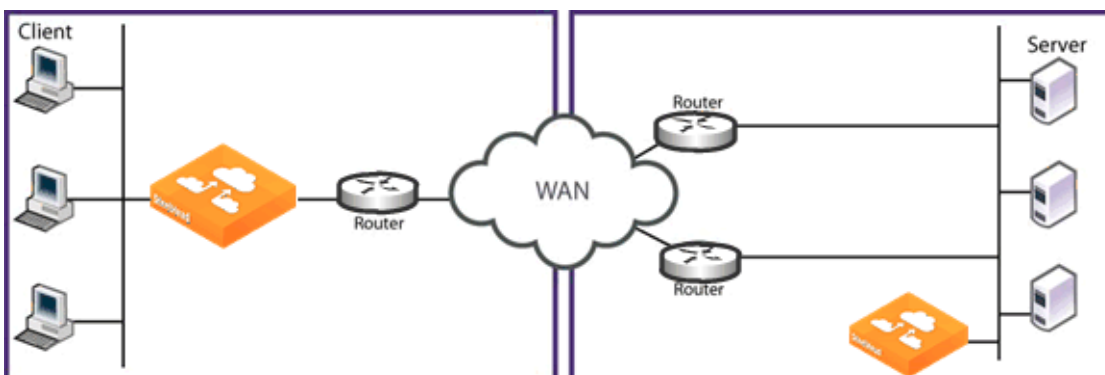
This chapter provides the basic steps for configuring one type of virtual in-path deployment. It does not provide detailed procedures for all virtual in-path deployments. Use this chapter as a general guide to virtual in-path deployments.

For information about the factors you must consider before you design and deploy the SteelHead in a network environment, see [“Choosing the right SteelHead model” on page 28](#).

Overview of virtual in-path deployment

In a virtual in-path deployment, the SteelHead is virtually in the path between clients and servers. Traffic moves in and out of the same WAN interface, and the LAN interface is not used. This deployment differs from a physical in-path deployment in that a packet redirection mechanism directs packets to SteelHeads that are not in the physical path of the client or server.

Figure 10-1. Virtual in-path deployment on the server side of the network



Redirection mechanisms include:

- **Layer-4 switch** - You enable Layer-4 switch (or server load balancer) support when you have multiple SteelHeads in your network to manage large bandwidth requirements.

For details, see [“Configuring an in-path, load-balanced, Layer-4 switch deployment” on page 246](#).

- **Hybrid** - In a hybrid deployment, the SteelHead is deployed either in a physical or virtual in-path mode, and it has out-of-path mode enabled. A hybrid deployment is useful in which the SteelHead must be referenced from remote sites as an out-of-path device (for example, to bypass intermediary SteelHeads).

For details, see [“Out-of-Path Deployments” on page 387](#).

- **PBR** - PBR enables you to redirect traffic to a SteelHead that is configured as a virtual in-path device. PBR allows you to define policies that override routing behavior. For example, instead of routing a packet based on routing table information, it is routed based on the policy applied to the router. You define policies to redirect traffic to the SteelHead and policies to avoid loop-back.

For details, see [“Policy-Based Routing Virtual In-Path Deployments” on page 287](#).

- **WCCP** - WCCP was originally implemented on Cisco routers, multilayer switches, and web caches to redirect HTTP requests to local web caches (Version 1). Version 2, which is supported on SteelHeads, can redirect any type of connection from multiple routers to multiple web caches. For example, if you have multiple routers or if there is no in-path place for the SteelHead, you can place the SteelHead in a virtual in-path mode through the router so that they work together.

For details, see [“WCCP Virtual In-Path Deployments” on page 249](#).

- **Interceptor appliance** - The SteelHead Interceptor is a load balancer specifically used to distribute optimized traffic to a local cluster of SteelHeads. The SteelHead Interceptor is SteelHead aware, so it offers several benefits over other clustering techniques like WCCP and PBR. The SteelHead Interceptor is dedicated to redirecting packets for optimized connections to SteelHeads, but it does not perform optimization itself. As a result, you can use the SteelHead Interceptor in extremely demanding network environments with extremely high throughput requirements. For information about the SteelHead Interceptor, see the *SteelHead Interceptor Deployment Guide* and the *SteelHead Interceptor User Guide*.

For networks that contain firewalls or tunnels (VPN, GRE, IPSec transport mode) between SteelHeads and require manual tuning of the MTU values, see [“MTU sizing” on page 476](#).

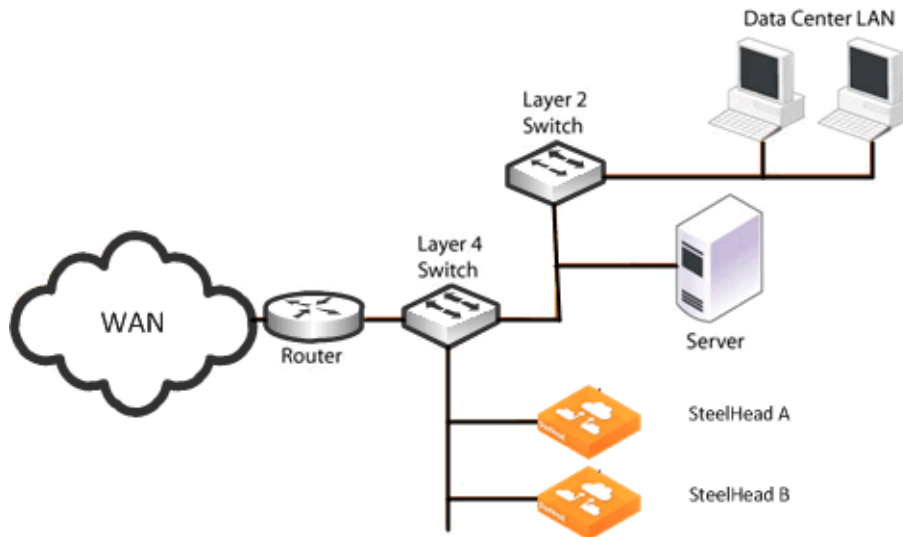
Configuring an in-path, load-balanced, Layer-4 switch deployment

An in-path, load-balanced, Layer-4 switch deployment serves high-traffic environments or environments with large numbers of active TCP connections. It handles failures, scales easily, and supports all protocols.

When you configure the SteelHead using a Layer-4 switch, you define the SteelHeads as a pool in which the Layer-4 switch redirects client and server traffic. Only one WAN interface on the SteelHead is connected to the Layer-4 switch, and the SteelHead is configured to send and receive data through that interface.

Figure 10-2 shows the server side of the network where load balancing is required.

Figure 10-2. In-path, load-balanced, Layer-4 switch deployment



To configure the client-side SteelHead for in-path load-balanced, Layer-4 switch

- Configure the client-side SteelHead as an in-path device. For details, see the *SteelHead Installation and Configuration Guide*.

To configure the server-side SteelHead for in-path load-balanced, Layer-4 switch

1. Install and power on the SteelHead.
2. Connect to the SteelHead. Make sure you properly connect to the Layer-2 switch, for example:
 - On SteelHeadA, plug the straight-through cable into the primary port of the SteelHead and connect it to the LAN-side switch.
 - On SteelHeadB, plug the straight-through cable into the primary port of the SteelHead and connect it to the LAN-side switch.
3. Configure the SteelHead in an in-path configuration.
4. Connect the Layer-4 switch to the SteelHead:
 - On SteelHeadA, plug the straight-through cable into the WAN port of the SteelHead and the Layer-4 switch.
 - On SteelHeadB, plug the straight-through cable into the WAN port of the SteelHead and the Layer-4 switch.
5. Connect to the Management Console.
6. Choose Optimization > Network Services: General Service Settings page and enable Layer-4 switch support. For example, select Enable In-Path Support and select Enable L4/PBR/WCCP Support.

7. Apply and save the new configuration in the Management Console.
8. Configure your Layer-4 switch as instructed by your switch documentation.
9. Choose Administration > Maintenance: Services and restart the optimization service.
10. View performance reports and system logs.
11. Perform the above steps for each SteelHead in the cluster.

Configuring flow data exports in virtual in-path deployments

The SteelHead supports the export of data flows to any compatible flow data collector. During data flow export, the flow data fields provide information such as the interface index that corresponds to the input and output traffic. An administrator can use the interface index to determine how much traffic is flowing from the LAN to the WAN and from the WAN to the LAN.

In virtual in-path deployments, such as the server side of the network, traffic moves in and out of the same WAN interface; the LAN interface is not used. As a result, when the SteelHead exports data to a flow data collector, all traffic has the WAN interface index. Though it is technically correct for all traffic to have the WAN interface index because the input and output interfaces are the same, this setting makes it impossible for an administrator to use the interface index to distinguish between LAN-to-WAN and WAN-to-LAN traffic.

In RiOS 6.0 or later, the fake index feature is enabled by default if you enable the *CascadeFlow* export option. Prior to RiOS 6.0, or if you are using standard NetFlow, you can work around this issue by turning on the SteelHead fake index feature, which inserts the correct interface index before exporting data to a flow data collector. The fake index feature works only for optimized traffic, not unoptimized or passed-through traffic.

For information about how to configure a fake index in the CLI in release prior to RiOS 6.0, see the appropriate version of the *Riverbed Command-Line Interface Reference Manual*.

Note: Subnet side rules are necessary for correct unoptimized or passed-through traffic reporting. For details, see the *SteelHead User Guide*.

For information about exporting flow data, see [“Overview of exporting flow data” on page 450](#).

WCCP Virtual In-Path Deployments

This chapter describes how to configure WCCP to redirect traffic to one or more SteelHeads. This chapter includes the following sections:

- “Overview of WCCP” on page 249
- “WCCP fundamentals” on page 250
- “Advantages and disadvantages of WCCP” on page 256
- “Configuring WCCP” on page 257
- “Configuring additional WCCP features” on page 273
- “Flow data in WCCP” on page 282
- “Verifying and troubleshooting WCCP configurations” on page 282

This chapter provides basic information about WCCP network deployments and examples for configuring WCCP deployments. Use this chapter as a general guide to WCCP deployments.

For information about the factors you must consider before you design and deploy the SteelHead in a network environment, see “Choosing the right SteelHead model” on page 28.

For information about WCCP, see the Cisco documentation website at <http://www.cisco.com/en/US/docs/ios-xml/ios/ipapp/configuration/12-4/iap-wccp.html>.

Overview of WCCP

This section provides an overview of WCCP. WCCP Version 1 was originally implemented on Cisco routers, multilayer switches, and web caches to redirect HTTP requests to local web caches.

WCCP Version 2, which SteelHeads support, can redirect any type of connection from multiple routers to multiple WCCP clients (also referred to as *caches* or *engines*). SteelHeads deployed with WCCP can interoperate with remote SteelHeads deployed in any way, such as WCCP, PBR, in-path, and out-of-path.

WCCP requires either a Cisco router or a Cisco switch.

The most important factors in a successful WCCP implementation are the Cisco hardware platform and the IOS revision you use. There are many possible combinations of Cisco hardware and IOS revisions, and each combination has different capabilities.

Cisco platforms and IOS do not support all assignment methods, redirection methods, use of ACLs to control traffic, and interface interception directions. You can expect the Cisco minimum recommended IOS to change as WCCP becomes more widely used and new IOS technical issues are discovered.

Cisco recommends the following minimum IOS releases for specific hardware platforms.

Cisco hardware	Cisco IOS
ASR 1000	2.2XE
ISR and 7200 Routers	12.1(14), 12.2(26), 12.3(13), 12.4(10), 12.1(3)T, 12.2(14)T, 12.3(14)T5, 12.4(15)T8, 12.4(24)T
ISR G2	15.0(1)M1
Catalyst 6500 with Sup720 or Sup32	12.2(18)SXF16, 12.2(33)SXI
Catalyst 6500 with Sup2	12.2(18)SXF16
Catalyst 7600	12.2(18)SXF16, 12.2(33)SXI
Catalyst 4500	12.2(46)SG
Catalyst 3750	12.2(46)SE
Nexus 7000	4.2.4

Regardless of how you configure a SteelHead, if the Cisco IOS version on the router or switch is below the current Cisco minimum recommendations, it might be impossible to have a functioning WCCP implementation or the implementation might not have optimal performance.

WCCP fundamentals

This section describes some of the fundamental concepts for configuring WCCP. This section includes the following topics:

- [“Service groups” on page 250](#)
- [“Assignment methods” on page 251](#)
- [“Redirection and return methods” on page 254](#)
- [“WCCP clustering and failover” on page 255](#)
- [“Multiple in-path WCCP” on page 256](#)

Service groups

A central concept of WCCP is the service group. The service group logically consists of up to 32 WCCP routers and 32 WCCP clients that work together to redirect and optimize traffic. WCCP routers are Cisco routers or switches capable of forwarding traffic meeting defined criteria. WCCP clients are the devices receiving this traffic. RiOS 6.1 or later includes multiple in-path WCCP (for details, see [“Multiple in-path WCCP” on page 256](#)). The same WCCP routers and clients can participate in one or more service groups.

Service groups are differentiated by a service group number. The service group number is local to the site where WCCP is used. The service group number is not transmitted across the WAN.

When a router participates in a WCCP service group, it is configured to monitor traffic passing through a user-defined set of interfaces. When a router receives traffic of interest, it redirects the IP packets to be transmitted to a designated interface in another system in the WCCP service group.

Note: We recommend that you use WCCP service groups 61 and 62.

Routers redirect traffic to the SteelHead interfaces in their WCCP service group. The assignment method and the load-balancing configuration determine which SteelHead interface the router redirects traffic to.

Assignment methods

This section describes WCCP assignment methods. This section includes the following topics:

- [“Hash assignment” on page 251](#)
- [“Mask assignment” on page 252](#)
- [“Choosing an assignment method” on page 253](#)

Routers participating in WCCP support two assignment methods. The assignment method affects how a router redirects traffic when multiple target systems are specified in a service group. Assignment methods are important when two or more SteelHeads are deployed at the same site for high availability or load balancing. The assignment methods are as follows:

- **Hash assignment** - Uses the software to calculate part of the load distribution, placing a significant load on the switch CPU.
- **Mask assignment** - Processes traffic entirely in hardware, so that the impact on the switch CPU is minimal. Mask assignment was specifically designed for hardware-based switches and routers.

Note: Do not confuse assignment methods with forwarding methods. Assignment methods determine how packets are distributed across multiple SteelHeads (through mask or hash), and forwarding methods determine how intercepted packets are forwarded from the router or switch to the SteelHead (through GRE or Layer 2).

Hash assignment

The hash assignment method redirects traffic based on a hashing scheme and the *weight* of the SteelHead interfaces. A hashing scheme is a combination of the source IP address, destination IP address, source port, or destination port. The hash assignment method is commutative: a packet with a source IP address X and a destination IP address Y hashes to the same value as a packet with a source IP address Y and a destination IP address X.

The weight of a SteelHead is determined by the number of connections the SteelHead supports. The default weight is based on the SteelHead model number. The more connections a SteelHead model supports, the heavier the weight of that model. You can modify the weight for each in-path interface to manually tune the proportion of traffic a SteelHead interface receives.

The hash assignment method supports failover and load balancing. In a failover configuration, you configure one or more SteelHead interfaces to be used only if no other SteelHead interfaces within the WCCP service group are operating. To configure a SteelHead interface for failover, set the SteelHead interface weight to 0 on the desired service group.

If a SteelHead interface has a weight of 0 and another SteelHead interface in the same WCCP service group has a nonzero weight, the SteelHead interface with the 0 weight does not receive redirected traffic. If all of the operating SteelHeads interfaces have a weight of 0, traffic is redirected equally among them.

Mask assignment

The mask assignment method redirects traffic based on administrator-specified bits pulled, or masked, from the IP address and TCP port fields. Unlike the hash assignment method, these bits are not hashed. Instead, the Cisco switch concentrates the bits to construct an index into the load-balancing table.

You must carefully choose these bits. Mask assignment uses up to 7 bits, which allows for a maximum of 128 buckets ($2^7=128$) for load balancing across SteelHead interfaces in the same service group. RiOS 6.1 or later supports load balancing and redundancy per interface by configuring each SteelHead interface with the appropriate service groups and router bindings (for details, see [“Multiple in-path WCCP” on page 256](#)).

The mask assignment method processes the first packet for a connection in the router hardware. To force mask assignment, use the `assign-scheme` option for the **wccp service-group** command:

```
wccp interface inpath0_0 service-group 61 routers 10.0.0.1 assign-scheme mask
```

Some Cisco platforms, such as the Catalyst 4500 and the Catalyst 3750, only support the mask assignment method.

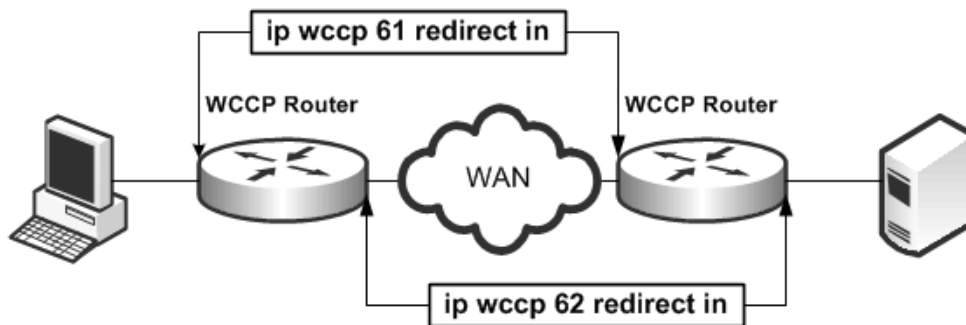
When you use the mask assignment method, you configure failover in the same manner as you do with the hash assignment method.

The mask assignment method requires that, for every connection, packets are redirected to the same SteelHead or interface in both directions (client-to-server and server-to-client). To achieve redirection, you configure the following settings:

- Because only one set of masks can be used per service group, we recommend that you use two different service groups for inbound traffic from the client (group 61) and inbound traffic from the server (group 62).
- Configure the Cisco switch to redirect packets to a WCCP service group in the client-to-server direction and to redirect packets to another WCCP group in the server-to-client direction. In most cases, service group 61 is placed on the inbound interface closest to the client and service group 62 is placed on the inbound interface closest to the server. Generally when you use a mask, service group 61 is configured based on the source IP, and service group 62 is configured for the destination IP to maintain consistent assignment in either direction.

Figure 11-1 shows the reversed mask redirection technique.

Figure 11-1. Mask assignment method packet redirection



For information about mask assignment method parameters, see the *Riverbed Command-Line Interface Reference Manual*.

Choosing an assignment method

Unless otherwise specified in the SteelHead interface WCCP service group setting, and if the router supports it, choose the hash assignment method. The hash assignment method generally achieves better load distribution, and mask assignment provides more user controlled configuration, including the ability to distribute traffic based on subnet mask values.

The following scenarios are instances when the mask assignment method is preferable:

- Certain lower-end Cisco switches (3750, 4000, 4500 series, among others) do not support hash assignment.
- The hash assignment method uses a NetFlow table entry on the switch for every connection. The NetFlow table entry can support up to 256K connections, depending on the hardware. However, when the switch runs out of NetFlow table entries, every WCCP-redirection packet is process switched, which has a crippling effect on the switch CPU, and very large WCCP deployments are constrained to the mask assignment load distribution method.
- The hash assignment method processes the first packet of every new redirected connection using the switch CPU. The switch CPU installs the NetFlow table entry that is used to hardware-switch subsequent packets for a given connection. This process limits the number of connection setups a switch can perform per unit of time. Thus, in WCCP deployments where the connection setup rate is very high, the mask assignment method is the only option.
- In multiple SteelHead environments, it is often desirable to send all users in subnet range to the same SteelHead. Mask assignment provides a basic ability to leverage a branch subnet and SteelHead to the same SteelHead in a WCCP cluster.

Redirection and return methods

This section describes the WCCP redirection and return methods. It includes the follow topics:

- “WCCP Return Router Determination” on page 255
- “Best practices for determining a redirection and return method” on page 255

WCCP supports two methods for transmitting packets between a router or switch and SteelHead interfaces: the GRE encapsulation method and the Layer-2 method. SteelHeads support both the Layer-2 and GRE encapsulation methods, in both directions, to and from the router or switch.

The Layer-2 method is generally preferred from a performance standpoint because it requires fewer resources from the router or switch than the GRE encapsulation does. The Layer-2 method modifies only the destination Ethernet address. However, not all combinations of Cisco hardware and IOS revisions support the Layer-2 method. Also, the Layer-2 method requires the absence of Layer-3 hops between the router or switch and the SteelHead.

The GRE encapsulation method appends a GRE header to a packet before it is forwarded. This method can cause fragmentation and imposes a performance penalty on the router and switch, especially during the GRE packet deencapsulation process. This performance penalty can be too great for production deployments.

If your deployment requires the use of GRE return, the SteelHead automatically changes the maximum segment size (MSS) for connections to 1432 bytes. You can overwrite this option with the **no wccp adjust-mss enable** command, although we do not recommend it.

You can avoid using the GRE encapsulation method for the traffic return path from the SteelHead by using the SteelHead **wccp override-return route-no-gre** or **wccp override-return sticky-no-gre** command. The **wccp override-return route-no-gre** command enables the SteelHead to return traffic without GRE encapsulation to a SteelHead in-path gateway, determined by the in-path routing table. The **wccp override-return sticky-no-gre** command enables the SteelHead to return traffic without GRE encapsulation to the router that forwarded the traffic. Both commands accomplish a similar goal of forcing the SteelHead to return packets through Layer 2, but the **wccp override-return sticky-no-gre** command is generally used because returning packets to the originating router is often most often preferred.

Use the **wccp override-return route-no-gre** or **wccp override-return sticky-no-gre** command only if the SteelHead is no more than a Layer-2 hop away from the potential next-hop routers and if the unencapsulated traffic does not pass through an interface that redirects the packet back to the SteelHead (that is, there is no WCCP redirection loop). For information about the **wccp override-return route-no-gre** or **wccp override-return sticky-no-gre** commands, see the *Riverbed Command-Line Interface Reference Manual*.

The following table summarizes Cisco hardware platform support for redirection and return methods.

Cisco hardware	Redirection and return method
Nexus 7000	Layer 2
ASR 1000	GRE or Layer 2
ISR and 7200 routers	GRE or Layer 2 (Layer 2 requires 12.4(20)T or later)
Catalyst 6500 with Sup720 or Sup32	GRE or Layer 2

Cisco hardware	Redirection and return method
Catalyst 6500 with Sup2	GRE or Layer 2
Catalyst 4500	Layer 2
Catalyst 3750	Layer 2

WCCP Return Router Determination

When a SteelHead in a WCCP cluster transmits packets for optimized or pass-through connections, how it decides to address those packets depends on the RiOS version, WCCP configuration, and its in-path routing table. RiOS 6.0 or later includes more techniques to statefully track the originating router, both for Layer-2 and GRE methods.

Best practices for determining a redirection and return method

We recommend the following best practices for determining your redirection and return method:

- Design your WCCP deployment so that your SteelHeads are no more than a Layer-2 hop away from the router or switch performing WCCP redirection.
- Do not configure a specific redirection or assignment method on your SteelHead. Allow the SteelHead to negotiate these settings with the router unless one of the reasons for using mask assignment applies to your deployment.

For information about mask assignment, see [“Choosing an assignment method” on page 253](#).

- Use the `wccp override-return route-no-gre` or `wccp override-return sticky-no-gre` commands only if the following conditions are both met:
 - The SteelHead is no more than a Layer-2 hop away from the router or switch.
 - Unencapsulated traffic going to the next-hop router or switch does not pass through an interface that redirects the packet back to the SteelHead (that is, there is no WCCP redirection loop). If this condition is not met, traffic redirected by the SteelHead is continually returned to the same SteelHead.

WCCP clustering and failover

WCCP clustering refers to two or more SteelHeads participating in the same service group. A single service group can support 32 WCCP clients, counting a single SteelHead interface as a client. However, a single SteelHead with multiple interfaces is typically not considered a cluster, even though each interface is considered a client.

Load balancing and redundancy is provided across all participating SteelHeads in a WCCP cluster by default. You can adjust the weight per each in-path interface to modify behavior for load balancing and redundancy. For example, you can configure a second in-path interface connected to a redundant device, with a smaller weight to be given a smaller proportion of traffic.

Certain timers in the WCCPv2 implementation directly affect failover time and are not adjustable. Every 10 seconds, the SteelHead and router exchange WCCP *Here I Am* and *I See You* messages (*Hello* messages). The router declares a client failed after missing three messages. If a SteelHead or SteelHead interface fails, the router always waits between 20 to 30 seconds before declaring the SteelHead down. If there are buckets (traffic distribution) assigned to the failed client, the router forwards traffic to the failed client during this interval, potentially black holing that traffic. Traffic to other WCCP clients is redirected and optimized as normal. After the WCCP client is declared failed, buckets are recalculated. The buckets belonging to the failed device are distributed among the remaining WCCP clients. The hello and failure time intervals cannot be adjusted in the current WCCPv2 implementation.

In high-availability environments, optimization redundancy refers to the ability to quickly fail over to another appliance to continue application acceleration with minimal impact to users. However, no matter what deployment method is chosen, optimized connections to one SteelHead are never statefully failed to a different SteelHead. SteelHeads optimizing the connection act as TCP proxies and, therefore, are fully aware of the connection state. Although connection forwarding allows a SteelHead to inform neighbors what connections are current, it does not exchange the full state of each connection, because doing this requires extensive and continuous updates with all neighbors. Without knowing the full state of a connection, a new device cannot safely resume the optimization of an existing connection. The majority of applications resend a SYN to establish a new connection with a redundant peer, transparent to the user. However, not all applications behave in this manner. Be aware that loss of a SteelHead requires the client to reestablish a new TCP connection, which is usually transparent to the user.

In RiOS 6.5 or later, you must enable connection forwarding in a WCCP cluster. With connection forwarding enabled, the WCCP load-balancing algorithm accounts for the total number of in-path interfaces of all neighbors in the service group when balancing the load across the interfaces. If you do not enable connection forwarding, the SteelHead with the lowest IP address assigns all traffic flows to itself.

Multiple in-path WCCP

RiOS 6.1 or later provides additional WCCP configuration, allowing each individual SteelHead in-path interface to be configured as a WCCP client. Each configured in-path interface participates in WCCP service groups as an individual WCCP client, providing flexibility to determine load-balancing proportions and redundancy.

Prior to RiOS 6.1, WCCP was configured with the **wccp service group** command. This command now includes the interface as a mandatory parameter, changing the command syntax to **wccp interface <interface> service group**.

For details, see the *Riverbed Command-Line Interface Reference Manual*.

Advantages and disadvantages of WCCP

Physical in-path deployments require less initial and ongoing configuration and maintenance than out-of-path or virtual in-path deployments. Physical in-path SteelHeads are placed at the points in your network where data already flows. Thus, with in-path deployments you do not need to alter your existing network infrastructure.

For information about physical in-path deployments, see [“Physical in-path deployments” on page 197](#).

Virtual in-path techniques, such as WCCP, require more time to configure because the network infrastructure must be configured to redirect traffic to the SteelHeads.

WCCP also has the following advantages:

- **No rewiring required** - You do not need to move any wires during installation. At large sites with multiple active links, you can adjust wiring by moving individual links, one at a time, through the SteelHeads.
- **An option when no other is available** - At sites where a physical in-path deployment is not possible, WCCP might achieve the integration you need. For example, if your site has a WAN link terminating directly into a large access switch, there is no place to install a physical in-path SteelHead.

WCCP has the following disadvantages:

- **Network design changes required** - WCCP deployments with multiple routers can require significant network changes (for example, spanning VLANs and GRE tunnels).
- **Hardware and IOS upgrades required** - To avoid hardware limitations and IOS issues, you must keep the Cisco platform and IOS revisions at the current minimum recommended levels. Otherwise, it might be *impossible* to create a stable deployment, regardless of how you configure the SteelHead. For future IOS feature planning you must consider compatibility with WCCP.
- **Additional evaluation overhead** - It can take more time to evaluate the integration of the SteelHeads. This overhead is in addition to evaluating SteelHead performance gains. You might need Riverbed Professional Services to test and perform network infrastructure upgrades before any optimization can be performed, especially when WCCP is deployed at numerous sites.
- **Additional configuration management** - You must create access control lists and manage them on an ongoing basis. At small sites, it might be feasible to redirect all traffic to the SteelHeads. However, at larger sites, access control lists might be required to ensure that traffic that cannot be optimized (for example, LAN-to-LAN traffic) is not sent to the SteelHeads.
- **GRE encapsulation** - If your network design does not support the presence of the SteelHeads and the Cisco router or switch interface in a common subnet, you must use GRE encapsulation for forwarding packets. SteelHeads can accommodate the subsequent extra performance utilization, but your existing router or switch might experience large resource utilization.

Configuring WCCP

This section describes how to configure WCCP and provides example deployments. This section includes the following topics:

- [“Basic steps for configuring WCCP” on page 258](#)
- [“Configuring a simple WCCP deployment” on page 259](#)
- [“Adding a SteelHead to an existing WCCP deployment” on page 262](#)
- [“Configuring a WCCP high availability deployment” on page 264](#)
- [“Configuring a basic WCCP router” on page 272](#)

Basic steps for configuring WCCP

This section describes the basic steps to set up WCCP.

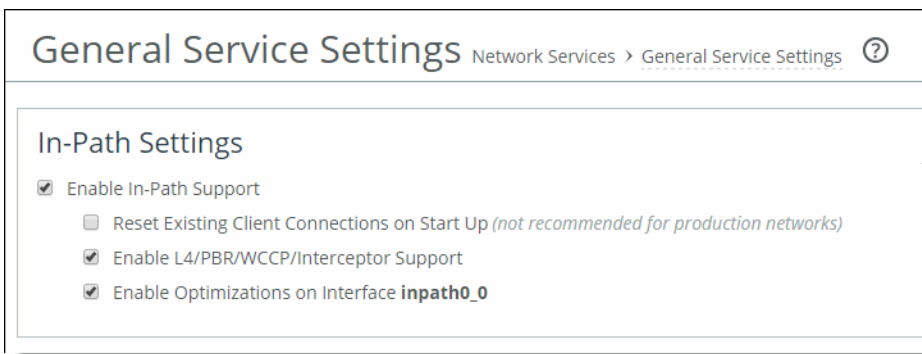
Note: Before you perform these steps on the SteelHead, make sure that simplified routing is set to None and connection forwarding is enabled. Out-of-path deployments are not compatible with simplified routing.

To perform the basic steps to configure WCCP

1. Configure the SteelHead as an in-path device and enable in-path support.

You can use the **in-path enable** and **in-path oop enable** commands, or you can use In-Path Settings page shown in [Figure 11-2](#).

Figure 11-2. In-path settings



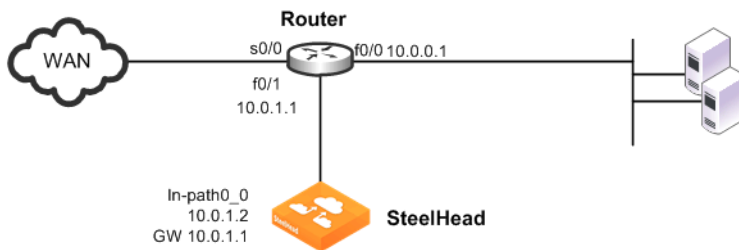
For details, see [“Physical in-path deployments” on page 197](#) and the *SteelHead Installation and Configuration Guide*.

2. Enable WCCP on the router by creating a service group on the router.
3. Set the router to use WCCP to redirect traffic to the WCCP SteelHead.
4. Attach the desired SteelHead in-path interface WAN interface to the network. The WAN interface must be able to communicate with the switch or router on which WCCP is configured and where WCCP redirection takes place.
5. Add the service group on the WCCP SteelHead interface.
6. Enable WCCP on the WCCP SteelHead.

Configuring a simple WCCP deployment

Figure 11-3 shows a WCCP deployment that is simple to deploy and administer and achieves high performance. This example includes a single router and a single SteelHead.

Figure 11-3. A single SteelHead and a single router



In this example:

- The router and the SteelHead use WCCP service groups 61 and 62. In this example, as long as the SteelHead interface is a member of all of the service groups, and the service groups include all of the interfaces on all of the paths to and from the WAN, it does not matter whether a single service group or multiple service groups are configured.
- The SteelHead wan0_0 interface is directly attached to the router, using a crossover cable.
- The SteelHead virtual inpath0_0 interface uses the IP information that is visible to the router and the remote SteelHeads for data transfer.
- The SteelHead does *not* have an encapsulation scheme in the WCCP service group configuration. Therefore, the SteelHead informs the router that it supports both the GRE and the Layer-2 redirection methods. The method negotiated and used depends on the methods that the router supports.
- You can force the SteelHead to perform Layer-2 return of packets, regardless of the negotiated method of return, by using either of the following commands: **wccp override-return route-no-gre** or **wccp override-return sticky-no-gre**. Enabling one of these commands potentially decreases the resource utilization on the router, especially with Layer-3 switches that must perform de-encapsulation of GRE packets in software.

For information about the **wccp override-return route-no-gre** command, see “Redirection and return methods” on page 254 and the following Riverbed Knowledge Base article, *What WCCP Redirect and Return Method Should I Use?*, located at <https://supportkb.riverbed.com/support/index?page=content&id=s15432>.

- The router uses the **ip wccp redirect exclude** command on the router interface connected to the SteelHead wan0_0 interface. This command configures the router to never redirect packets arriving on this interface, even if they are later sent out of an interface with an **ip wccp redirect out** command. Unless you configure the router with the **ip wccp redirect out** command on an interface, then you do not need to configure the **ip wccp redirect exclude** command. Almost all Cisco WCCP capable Layer-3 switches prefer the **ip wccp redirect in** command, so using the **ip wccp redirect exclude** command serves no purpose and, furthermore, can add overhead to the switch CPU.

Although the primary interface is not included in this example, we recommend that you connect the primary interface for management purposes. For information about configuring the primary interface, see the *SteelHead User Guide*.

To configure WCCP on the SteelHead

- On the SteelHead, connect to the CLI and enter the following commands:

```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead.
interface primary ip address 10.0.0.2 /24
ip default-gateway 10.0.0.1
interface inpath0_0 ip address 10.0.1.2 /24
ip in-path-gateway inpath0_0 10.0.1.1
in-path enable
#--- Enables virtual In-path support for WCCP / PBR / or Layer-4 switch.
in-path oop enable
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is Layer-2 adjacent, use the interface IP of the router.
wccp enable
wccp interface inpath0_0 service-group 61 routers 10.0.1.1
wccp interface inpath0_0 service-group 62 routers 10.0.1.1
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

In the following example, only traffic to or from IP addresses 192.168.1.1 is sent to the SteelHead.

To configure WCCP on the Cisco router

- On the router, at the system prompt, enter the following commands:

```
enable
configure terminal
!--- Create the access control lists that determine what traffic to redirect
!--- to the SteelHeads. Creating two separate ACLs is optional.
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the client subnets to
!--- the server subnets.
ip access-list extended wccp_acl_61
deny tcp 10.0.1.0 0.0.0.255 any
deny tcp any 10.0.1.0 0.0.0.255
permit tcp <client-subnets> <server-subnets>
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the server subnets to
!--- the client subnets
ip access-list extended wccp_acl_62
deny tcp 10.0.1.0 0.0.0.255 any
deny tcp any 10.0.1.0 0.0.0.255
permit tcp <server-subnets> <client-subnets>
!--- Enable WCCPv2 and service groups 61 & 62; define the redirect
!--- lists for each service group
ip wccp version 2
ip wccp 61 redirect-list wccp_acl_61
ip wccp 62 redirect-list wccp_acl_62
!--- Add WCCP service group 62 to the server-facing interfaces
interface f0/0
ip wccp 62 redirect in
!--- Add WCCP service group 61 to the client-facing interfaces
interface s0/0
ip wccp 61 redirect in
!--- As a best practice use "redirect exclude in" on the interfaces or VLANs
!--- that are connected to the SteelHeads. If you are using
!--- redirect out on any interface this command is REQUIRED.
interface f0/1
ip wccp redirect exclude in
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

For information about how to verify the WCCP configuration, [“Verifying and troubleshooting WCCP configurations” on page 282](#).

Configuring WCCP using the mask assignment method is very similar to the standard WCCP configuration. The following example uses the mask of 0x3 that creates four buckets.

To configure WCCP on the SteelHead using the mask assignment method

- On the SteelHead, connect to the CLI and enter the following commands:

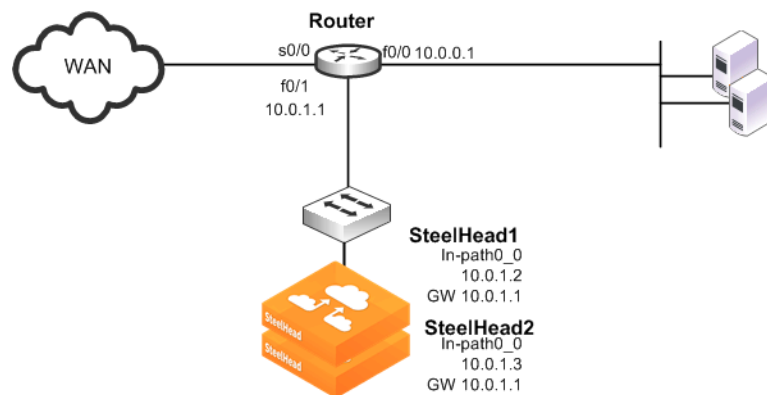
```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead.
interface primary ip address 10.0.0.2 /24
ip default-gateway 10.0.0.1
interface inpath0_0 ip address 10.0.1.2 /24
ip in-path-gateway inpath0_0 10.0.1.1
in-path enable
#--- Enables virtual In-path support for WCCP / PBR / or L4 switch.
in-path oop enable
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is L2 adjacent, use the interface IP of the router.
wccp enable
wccp interface inpath0_0 service-group 61 routers 10.0.1.1 assign-scheme mask src-ip-mask 0x3
wccp interface inpath0_0 service-group 62 routers 10.0.1.1 assign-scheme mask dst-ip-mask 0x3
write memory
restart
```

Adding a SteelHead to an existing WCCP deployment

You can have a maximum of 32 SteelHeads in your WCCP deployment. When you add new SteelHeads to an existing deployment, the buckets used by the router for load distribution are recalculated. New connections that were previously directed to one SteelHead might be redirected, resulting initially in cold performance after you restart service.

Note: Adding a configuration to the existing SteelHeads requires a service restart during a performance window.

Figure 11-4. Adding a SteelHead to an existing WCCP deployment



To add an additional SteelHead to an existing WCCP deployment

1. On SteelHead1, connect to the CLI and enter the following commands:

```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead.
interface inpath0_0 ip address 10.0.1.2 /24
ip in-path-gateway inpath0_0 10.0.1.1
in-path enable
#--- Enable virtual In-path support for WCCP / PBR / or Layer-4 switch.
in-path oop enable
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is Layer-2 adjacent, use the interface IP of the router.
wccp enable
wccp interface inpath0_0 service-group 61 routers 10.0.1.1
wccp interface inpath0_0 service-group 62 routers 10.0.1.1
#--- If the router negotiates GRE return, use route-no-gre to return
#--- the packets to the MAC of the next hop in the routing table instead
#--- of using GRE return. Alternately "wccp override-return sticky-no-gre"
#--- will return packets to the MAC address of the router that forwarded
#--- the packet to the SteelHead.
wccp override-return route-no-gre
#--- Enable Connection Forwarding to neighbor 10.0.1.3
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHead2 main-ip 10.0.1.3
steelhead communication allow-failure
steelhead communication advertiseresync
#--- save the config
write memory
#--- Restart the optimization service.
restart
```

2. On SteelHead2, connect to the CLI and enter the following commands:

```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead.
interface inpath0_0 ip address 10.0.1.3 /24.
ip in-path-gateway inpath0_0 10.0.1.1
in-path enable
#--- Enables virtual In-path support for WCCP / PBR / or Layer-4 switch
in-path oop enable
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is Layer-2 adjacent, use the interface IP of the router
wccp enable
wccp interface inpath0_0 service-group 61 routers 10.0.1.1
wccp interface inpath0_0 service-group 62 routers 10.0.1.1
#--- Enables Connection Forwarding to neighbor 10.0.1.2
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelHead1 main-ip 10.0.1.2
steelhead communication allow-failure
steelhead communication advertiseresync
#--- save the config
write memory
#--- Restart the optimization service.
restart
```

Configuring a WCCP high availability deployment

This section describes configuring a WCCP high availability deployment. This section includes the following topics:

- [“Single SteelHead with interface high availability” on page 265](#)
- [“Dual WCCP SteelHeads and interfaces with high availability” on page 267](#)

RiOS 6.1 or later supports redundancy across multiple interfaces. Previously, high availability was only available at the appliance level. The following examples show appliances running 6.1 or later with multiple WCCP interfaces. The same configuration concepts apply to versions before 6.1, except that each appliance can have only one WCCP interface configured.

If you use RiOS versions before 6.1, you cannot achieve the high availability shown in [“Single SteelHead with interface high availability” on page 265](#). In [“Dual WCCP SteelHeads and interfaces with high availability” on page 267](#), you can provide appliance redundancy, but each SteelHead does not have interface redundancy.

The examples in [“Single SteelHead with interface high availability” on page 265](#) and [“Dual WCCP SteelHeads and interfaces with high availability” on page 267](#) show the configuration of SteelHeads for interface high availability. These examples focus solely on setting up multiple SteelHead interfaces to communicate with multiple routers and, therefore omit any best practice recommendations on redirection and assignment method configurations.

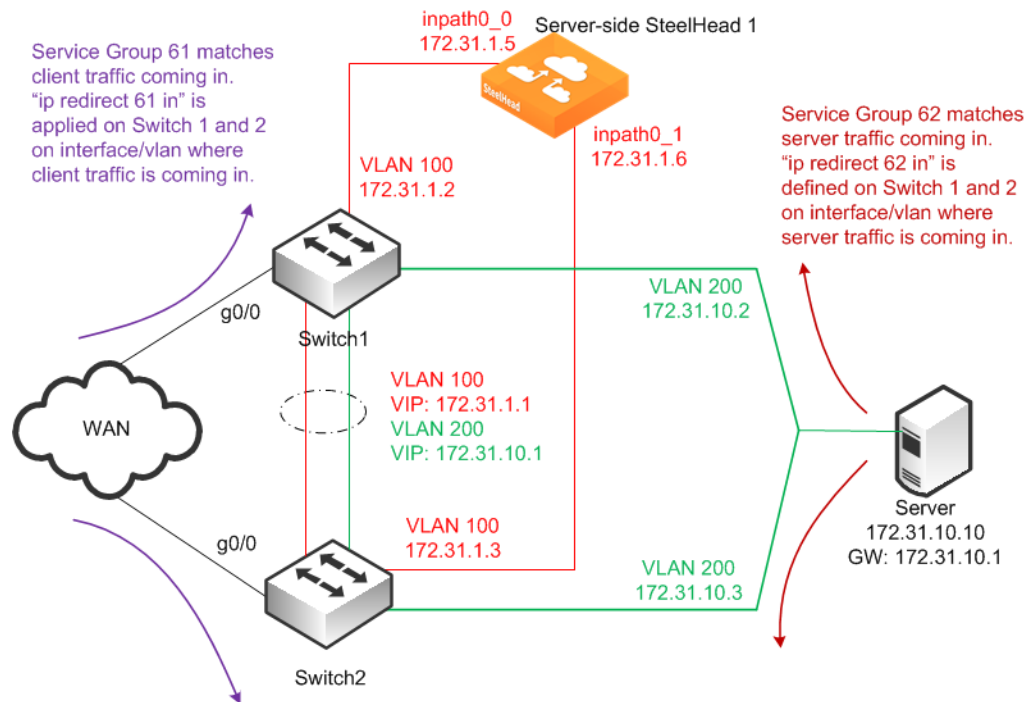
You must be familiar with [“Assignment methods” on page 251](#) and [“Redirection and return methods” on page 254](#).

Single SteelHead with interface high availability

Figure 11-5 shows a high availability WCCP deployment in which a single SteelHead with two in-path interfaces and two routers are used in a WCCP configuration. This configuration ensures that traffic continues to optimize in the event of a SteelHead interface, router, or link failure. This example does not provide SteelHead high availability.

Note: This deployment requires multiple in-path WCCP in RiOS 6.1 or later.

Figure 11-5. WCCP with interface high availability



In this example:

- The WCCP service groups are composed of two routers (Layer-3 switches) redirecting traffic and two SteelHead interfaces acting as the cache engines. The SteelHead is connected to both routers: wan0_0 goes to Switch1, and wan0_1 goes to Switch2.
- If a single SteelHead interface fails, all traffic is forwarded to the remaining SteelHead interface.

To configure WCCP with a single SteelHead with interface high availability

1. On the SteelHead 1, connect to the CLI and enter the following commands:

```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead. Primary address is used for
#--- management as well as for RiOS data store sync. The primary interface is not shown
#--- in the diagram as this can be attached to any accessible network.
interface primary ip address 10.10.1.10 /24
ip default-gateway 10.10.1.2
interface inpath0_0 ip address 172.31.1.5 /24
ip in-path-gateway inpath0_0 172.31.1.1
interface inpath0_1 ip address 172.31.1.6 /24
ip in-path-gateway inpath0_1 172.31.1.1
in-path enable
```

```
#--- Enables virtual In-path support for WCCP/PBR/ or Layer-4 switch.
in-path oop enable
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is Layer-2 adjacent, use the interface IP of the router.
#--- If the SteelHead is not Layer-2 adjacent, use the RID (highest loopback) address.
wccp enable
wccp interface inpath0_0 service-group 61 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_0 service-group 62 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_1 service-group 61 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_1 service-group 62 routers 172.31.1.2 172.31.1.3
#--- The above omits configurations related to selecting redirection or assignment methods.
#--- It is recommended to read, understand, and select the methods most appropriate for the
#--- environment. For example, the majority of L3 switches prefer L2 redirection and mask
#--- assignment. When using mask assignment, follow the best practices to ensure consistent
#--- assignment in either direction, typically by using source IP mask in one service group,
#--- and destination IP mask in the other.
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. To configure WCCP on Cisco router 1 (Switch1), at the system prompt, enter the following commands:

```
enable
configure terminal
!--- Create the access control lists that determine what traffic to redirect
!--- to the SteelHeads. Creating two separate ACLs is optional.
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the client subnets to
!--- the server subnets.
ip access-list extended wccp_acl_61
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <client-subnets> <server-subnets>
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the server subnets to
!--- the client subnets.
ip access-list extended wccp_acl_62
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <server-subnets> <client-subnets>
!--- Enable WCCPv2 and service groups 61 & 62; define the redirect
!--- lists for each service group.
ip wccp version 2
ip wccp 61 redirect-list wccp_acl_61
ip wccp 62 redirect-list wccp_acl_62
interface vlan 100
!--- Add WCCP service group 61 to the client-facing interfaces; in this example
!--- clients traffic arrives via gigabit interface 0/0.
interface g0
ip wccp 61 redirect in
!--- Add WCCP service group 62 to the server-facing interfaces; in this example
!--- servers are coming in via the LAN on VLAN 200.
interface vlan 200
ip wccp 62 redirect in
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

3. To configure WCCP on Cisco router 2 (Switch2), at the system prompt, enter the following commands:

```
enable
configure terminal
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the client subnets to
!--- the server subnets.
ip access-list extended wccp_acl_61
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <client-subnets> <server-subnets>
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the server subnets to
!--- the client subnets.
ip access-list extended wccp_acl_62
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <server-subnets> <client-subnets>
!--- Enable WCCPv2 and service groups 61 & 62; define the redirect
!--- lists for each service group.
ip wccp version 2
ip wccp 61 redirect-list wccp_acl_61
ip wccp 62 redirect-list wccp_acl_62
interface vlan 100
!--- Add WCCP service group 61 to the client-facing interfaces; in this example
!--- client traffic arrives via gigabit interface 0/0.
interface g0/0
ip wccp 61 redirect in
!--- Add WCCP service group 62 to the server-facing interfaces; in this example
!--- servers are coming in via the LAN on VLAN 200.
interface vlan 200
ip wccp 62 redirect in
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

For information about verifying the WCCP configuration, see [“Verifying and troubleshooting WCCP configurations” on page 282](#).

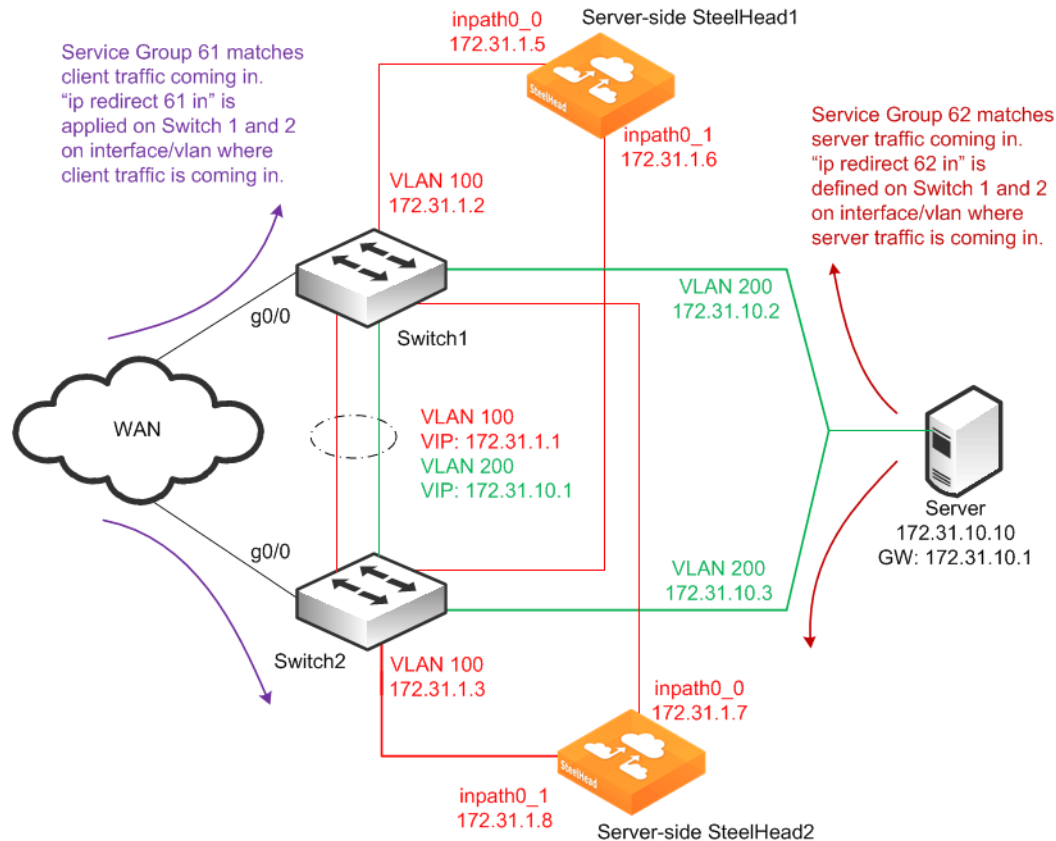
Dual WCCP SteelHeads and interfaces with high availability

Figure 11-6 shows a high availability WCCP deployment where two SteelHeads with two in-path interfaces and two routers are used in a WCCP configuration. This configuration ensures that traffic continues to be optimized in the event of a SteelHead interface or router failure.

This deployment requires multiple in-path WCCP in RiOS 6.1 or later.

RiOS data store synchronization is commonly used in high availability designs. You can configure RiOS data store synchronization between any two local SteelHeads, regardless of how they are deployed: physical in-path, virtual in-path, or out-of-path. For information about data store synchronization, see [“RiOS data store synchronization” on page 27](#).

Figure 11-6. High availability WCCP with RiOS data store synchronization



In this example:

- The SteelHeads are connected to both routers (Layer-3 switches). For each SteelHead, wan0_0 is connected to Switch 1, and wan0_1 is connected to Switch 2.
- The WCCP service groups are composed of two routers redirecting traffic and four SteelHead interfaces acting as the cache engines.
- If a single SteelHead interface fails, all traffic is forwarded to the remaining SteelHead interfaces, including the second interface on the same SteelHead.
- If a single SteelHead fails, all traffic is forwarded to the other SteelHead's two in-path interfaces.
- Because the two SteelHeads synchronize their RiOS data stores, the remaining SteelHead provides the same level of acceleration as the failed SteelHead.

If you are using a cluster of WCCP-attached SteelHeads, all remote client-side SteelHeads must have probe caching disabled using the **no in-path probe-caching enable** command.

To configure dual WCCP SteelHeads with interfaces with high availability

1. To configure WCCP on SteelHead1, connect to the CLI and enter the following commands:

```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead.
#--- Primary address is used for management as well as for RiOS data store sync.
#--- The primary interface is not shown in the diagram
#--- as this can be attached to any accessible network.
interface primary ip address 10.10.1.10 /24
ip default-gateway 10.10.1.2
interface inpath0_0 ip address 172.31.1.5 /24
ip in-path-gateway inpath0_0 172.31.1.1
interface inpath0_1 ip address 172.31.1.6 /24
ip in-path-gateway inpath0_1 172.31.1.1
in-path enable
#--- Enables virtual In-path support for WCCP/PBR/ or Layer-4 switch.
in-path oop enable
#--- Enables Connection Forwarding to neighbor 172.31.1.7
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SH2 main-ip 172.31.1.7
steelhead communication allow-failure
steelhead communication advertiseresync
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is Layer-2 adjacent, use the interface IP of the router.
#--- If the SteelHead is not Layer-2 adjacent, use the RID (highest loopback) address.
wccp enable
wccp interface inpath0_0 service-group 61 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_0 service-group 62 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_1 service-group 61 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_1 service-group 62 routers 172.31.1.2 172.31.1.3
#--- The above omits configurations related to selecting redirection or assignment methods.
#--- It is recommended to read, understand, and select the methods most appropriate for the
#--- environment. For example, the majority of L3 switches prefer L2 redirection and mask
#--- assignment. When using mask assignment, follow the best practices to ensure consistent
#--- assignment in either direction, typically by using source IP mask in one service group,
#--- and destination IP mask in the other.
#--- Enable RiOS data store synchronization and set this SteelHead as the primary.
datastore sync master
datastore sync peer-ip 10.10.1.13
datastore sync enable
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. To configure WCCP on SteelHead2, connect to the CLI and enter the following commands:

```
enable
configure terminal
#--- Configure the basic IP addressing of the SteelHead.
#--- Primary address is used for management as well as for RiOS data store sync.
#--- The primary interface is not shown in the diagram as this
#--- can be attached to any accessible network.
interface primary ip address 10.1.1.13 /24
ip default-gateway 10.10.1.3
interface inpath0_0 ip address 172.31.1.7 /24
ip in-path-gateway inpath0_0 172.31.1.1
interface inpath0_1 ip address 172.31.1.8 /24
ip in-path-gateway inpath0_1 172.31.1.1
in-path enable
#--- Enables virtual In-path support for WCCP / PBR / or Layer-4 switch.
in-path oop enable
#--- Enables Connection Forwarding to neighbor 172.31.1.5.
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
#--- allow-failure allows the SteelHead to continue optimizing
#--- traffic even if the neighbor is down.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SH1 main-ip 172.31.1.5
steelhead communication allow-failure
steelhead communication advertiseresync
#--- Enable WCCP and create Service Groups 61 & 62; assign
#--- router IP addresses for each service group.
#--- If the SteelHead is Layer-2 adjacent, use the interface IP of the router
#--- If the SteelHead is not Layer-2 adjacent, use the RID (highest loopback) address.
wccp enable
wccp interface inpath0_0 service-group 61 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_0 service-group 62 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_1 service-group 61 routers 172.31.1.2 172.31.1.3
wccp interface inpath0_1 service-group 62 routers 172.31.1.2 172.31.1.3
#--- The above omits configurations related to selecting redirection or assignment methods.
#--- It is recommended to read, understand, and select the methods most appropriate for the
#--- environment. For example, the majority of L3 switches prefer L2 redirection and mask
#--- assignment. When using mask assignment, follow the best practices to ensure consistent
#--- assignment in either direction, typically by using source IP mask in one service group,
#--- and destination IP mask in the other.
#--- Enables RiOS data store synchronization and sets this SteelHead as the backup.
no datastore sync master
datastore sync peer-ip 10.10.1.10
datastore sync enable
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

3. To configure WCCP on Cisco router 1 (Switch1), at the system prompt, enter the following commands:

```
enable
configure terminal
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the client subnets to
!--- the server subnets.
ip access-list extended wccp_acl_61
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <client-subnets> <server-subnets>
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the server subnets to
!--- the client subnets.
ip access-list extended wccp_acl_62
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <server-subnets> <client-subnets>
!--- Enable WCCPv2 and service groups 61 & 62; define the redirect
!--- lists for each service group.
ip wccp version 2
ip wccp 61 redirect-list wccp_acl_61
ip wccp 62 redirect-list wccp_acl_62
interface vlan 100
!--- Add WCCP service group 61 to the client-facing interfaces; in this example
!--- client traffic arrives via gigabit interface 0/0.
interface g0/0
    ip wccp 61 redirect in
!--- Add WCCP service group 62 to the server-facing interfaces; in this example
!--- servers are coming in via the LAN on VLAN 200
interface vlan 200
    ip wccp 62 redirect in
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

4. To configure WCCP on Cisco router 2 (Switch 2), at the system prompt, enter the following commands:

```
enable
configure terminal
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the client subnets to
!--- the server subnets.
ip access-list extended wccp_acl_61
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <client-subnets> <server-subnets>
!--- Deny all traffic sourced from or destined to the SteelHead
!--- in-path IP addresses and allow traffic from the server subnets to
!--- the client subnets.
ip access-list extended wccp_acl_62
deny tcp 172.31.1.0.0.0.255 any
deny tcp any 172.31.1.0 0.0.0.255
permit tcp <server-subnets> <client-subnets>
ip wccp version 2
ip wccp 61 redirect-list wccp_acl_61
ip wccp 62 redirect-list wccp_acl_62
interface vlan 100
!--- Add WCCP service group 61 to the client-facing interfaces; in this example
!--- client traffic arrives via gigabit interface 0/0.
interface g0/0
    ip wccp 61 redirect in
!--- Add WCCP service group 62 to the server-facing interfaces; in this example
!--- servers are coming in via the LAN on VLAN 200.
interface vlan 200
    ip wccp 62 redirect in
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

For information about how to verify the WCCP configuration, [“Verifying and troubleshooting WCCP configurations” on page 282](#).

Configuring a basic WCCP router

This section summarizes some of the basic WCCP router configuration commands. For complete information about WCCP router configuration commands, refer to your router documentation.

To enable WCCP and define a service group on the router

- On the router, at the system prompt, enter the following commands:

```
enable
configure terminal
ip wccp <service-number>
end
write memory
```

Note: The service group you specify on the router must also be set on the WCCP SteelHead.

Note: The WCCP protocol allows you to add up to 32 SteelHeads and 32 routers to a service group.

To specify inbound traffic redirection for each router interface

- On the router, at the system prompt, enter the following commands:

```
enable
configure terminal
!--- Add WCCP service group 61 to the client-facing interfaces.
interface FastEthernet 0/0
ip wccp 61 redirect in
!--- Add WCCP service group 62 to the server-facing interfaces.
interface serial 0
ip wccp 62 redirect in
end
write memory
```

To retain information you previously specified with **ip wccp**, you must enter a new **ip wccp** command that includes the information you previously specified and whatever you want to configure.

For example, you can configure your router using the following set of commands:

```
enable
configure terminal
ip wccp 61 redirect-list 100
end
write memory
```

If you want to specify a password on the router later, the **ip wccp 61 password <password>** command overwrites the previous redirect list configuration.

To retain the previous redirect list configuration and set a password, you must use the following command:

```
ip wccp 61 redirect-list 100 password <password>
```

For example:

```
enable
configure terminal
ip wccp 61 redirect-list 100 password XXXYYZZ
end
write memory
```

Configuring additional WCCP features

This section describes additional WCCP features and how to configure them. This section includes the following topics:

- [“Specifying the service group password” on page 274](#)
- [“Configuring multicast groups” on page 274](#)
- [“Configuring group lists to limit service group members” on page 275](#)
- [“Configuring access control lists” on page 276](#)
- [“Configuring load balancing in WCCP” on page 279](#)

Specifying the service group password

You can configure password authentication of WCCP protocol messages between the router and the SteelHead interface:

- The router service group must match the service group password configured on the WCCP SteelHead interface.
- The same password must be configured on the router and the WCCP SteelHead interface.
- Passwords must be no more than eight characters.

Note: The following router commands are not required for the example network configurations in this chapter. Use caution when you enter the `ip wccp [SG]` router command because each `ip wccp [SG]` router command overwrites the previous `ip wccp [SG]` router command. You cannot use an `ip wccp [SG]` router command to augment `ip wccp [SG]` router commands you previously issued. For details, see [“Configuring a basic WCCP router” on page 272](#).

To specify the service group password on the WCCP router

- On the router, at the system prompt, enter the following commands:

```
enable
configure terminal
ip wccp <service-group> password <password>
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

To set the service group password on the WCCP SteelHead interface

- Connect to the Riverbed CLI on the WCCP SteelHead and enter the following commands:

```
enable
configure terminal
wccp interface <interface> service-group <service-id> routers <ip-address> password <password>
write memory
restart
```

For example, to set the password on inpath0_0, where the router service group is 61 and the router IP address is 10.1.0.1, enter the following command:

```
wccp inpath0_0 service-group 61 routers 10.1.0.1 password XXXYYYYZ
```

Note: You must set the same password on the SteelHead interface and the Cisco router.

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

Configuring multicast groups

If you add multiple routers and SteelHead interfaces to a service group, you can configure them to exchange WCCP protocol messages through a multicast group.

Configuring a multicast group is advantageous because if a new router is added, it does not need to be explicitly added on each SteelHead interface.

Note: The following router commands are not required for the example network configurations in this chapter. Use caution when you enter the `ip wccp [SG]` router command because each `ip wccp [SG]` router command overwrites the previous `ip wccp [SG]` router command. You cannot use an `ip wccp [SG]` router command to augment `ip wccp [SG]` router commands you previously issued. For details, see [“Configuring a basic WCCP router” on page 272](#).

To configure multicast groups on the WCCP router

- On the router, at the system prompt, enter the following commands:

```
enable
configure terminal
ip wccp 61 group-address 239.0.0.1
interface fastEthernet 0/0
ip wccp 61 redirect in
ip wccp 61 group-listen
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

Note: Multicast addresses must be between 224.0.0.0 and 239.255.255.255.

To configure multicast groups on the WCCP SteelHead interface

- Connect to the Riverbed CLI on the WCCP SteelHead and enter the following commands:

```
enable
configure terminal
wccp enable
wccp mcast-ttl 10
wccp interface inpath0_0 service-group 61 routers 239.0.0.1
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

Note: You must set the same password on the SteelHead interface and the Cisco router.

Configuring group lists to limit service group members

You can configure a group list on your router to limit service group members (for instance, SteelHead interfaces) by IP address.

For example, if you want to allow only SteelHead interfaces with IP addresses 10.1.1.23 and 10.1.1.24 to join the router service group, you create a group list on the router.

Note: The following WCCP router commands are not required for the example network configurations in this chapter. Use caution when you enter the `ip wccp [SG]` router command because each `ip wccp [SG]` router command overwrites the previous `ip wccp [SG]` router command. You cannot use an `ip wccp [SG]` router command to augment `ip wccp [SG]` router commands you previously issued. For details, see [“Configuring a basic WCCP router” on page 272](#).

To configure a WCCP router group list

- On the WCCP router, at the system prompt, enter the following commands:

```
enable
configure terminal
access-list 1 permit 10.1.1.23
access-list 1 permit 10.1.1.24
ip wccp 61 group-list 1
interface fastEthernet 0/0
ip wccp 61 redirect in
end
write memory
```

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

Configuring access control lists

This section describes how to configure access control lists (ACLs). This section includes the following topics:

- [“Using access control lists for specific traffic redirection” on page 276](#)
- [“Cisco access control list command parameters” on page 277](#)
- [“Using access control lists with WCCP” on page 278](#)

When you configure ACLs, consider the following guidelines:

- ACLs are processed in order, from top to bottom. As soon as a particular packet matches a statement, it is processed according to that statement and the packet is not evaluated against subsequent statements. The order of your access control list statements is very important.
- If port information is not explicitly defined, all ports are assumed.
- By default, all lists include an implied **deny all** Cisco command at the end, which ensures that traffic that is not explicitly included is denied. You cannot change or delete this implied entry.

Using access control lists for specific traffic redirection

If redirection is based on traffic characteristics other than ports, you can use ACLs on the router to define which traffic is redirected.

If you only want the traffic for IP address **10.2.0.0/16** to be redirected to the WCCP SteelHead, configure the router according to the following example.

Note: The following router commands are not required for the example network configurations in this chapter. Use caution when you enter the **ip wccp [SG]** router command because each **ip wccp [SG]** router command overwrites the previous **ip wccp [SG]** router command. You cannot use an **ip wccp [SG]** router command to augment **ip wccp [SG]** router commands you previously issued. For details, see [“Configuring a basic WCCP router” on page 272](#).

To configure specific traffic redirection on the router

- On the router, at the system prompt, enter the following commands:

```
enable
configure terminal
access-list 101 permit tcp any 10.2.0.0 0.0.255.255
access-list 101 permit tcp 10.2.0.0 0.0.255.255 any
ip wccp 61 redirect-list 101
interface fastEthernet 0/0
ip wccp 61 redirect in
end
interface serial0
ip wccp 61 redirect in
end
write memory
```

Note: If you have defined fixed-target rules, redirect traffic in one direction as shown this example.

Note: Enter configuration commands, one per line. End each command with Ctrl+Z.

Cisco access control list command parameters

This section describes the Cisco **access-list** router command for using ACLs to configure WCCP redirect lists. For information about ACL commands, refer to your router documentation.

The **access-list** router command has the following syntax:

access-list <access-list-number> [permit | deny] tcp <source-IP/mask> <source-port> <destination-IP/mask> <destination-port>

access-list-number	Number from 1 to 199 that identifies the access control list. Standard access control lists are numbered 1 to 99 ; extended access control lists are numbered 100 to 199 . A standard access control list matches traffic based on source IP address. An extended access control list matches traffic based on source or destination IP address. We recommend that you use extended IP access control lists.
permit	Allows traffic redirection.
deny	Stops traffic redirection.
tcp	Specifies the traffic to redirect. WCCP only redirects TCP traffic. Use this option only when configuring a redirect list for WCCP.
source-IP/mask	Source IP address and mask. To set the mask, specify 0 or 1 , where 0 = match and 1 = does not matter. For example: <ul style="list-style-type: none"> ■ any - Matches any IP address. ■ 10.1.1.0 0.0.0.255 - Matches any host on the 10.1.1.0 network. ■ 10.1.1.1 0.0.0.0 - Matches host 10.1.1.1 exactly. ■ 10.1.1.1 - Matches host 10.1.1.1 exactly. This option is identical to specifying 10.1.1.1 0.0.0.0.

source-port	<p>Source port number or corresponding keyword. For example:</p> <ul style="list-style-type: none"> ▪ eq 80 or eq www - Identical options that match port 80. ▪ gt 1024 - Matches any port greater than 1024. ▪ lt 1024 - Matches any port less than 1024. ▪ neq 80 - Matches any port except port 80. ▪ range 80 90 - Matches any port between and including 80 through 90. <p>Cisco routers support many keywords. For details, refer to your router documentation.</p>
destination-IP/mask	<p>Destination IP address and mask. To set the mask, specify 0 or 1, where 0 = match and 1 = does not matter. For example:</p> <ul style="list-style-type: none"> ▪ any - Matches any IP address. ▪ 10.1.1.0 0.0.0.255 - Matches any host on the 10.1.1.0 network. ▪ 10.1.1.1 0.0.0.0 - Matches host 10.1.1.1 exactly. ▪ 10.1.1.1 - Matches host 10.1.1.1 exactly. This option is identical to specifying 10.1.1.1 0.0.0.0.
destination-port	<p>Destination port number or corresponding keyword. For example:</p> <ul style="list-style-type: none"> ▪ eq 80 or eq www - Identical options that match port 80. ▪ gt 1024 - Matches any port greater than 1024. ▪ lt 1024 - Matches any port less than 1024. ▪ neq 80 - Matches any port except port 80. ▪ range 80 90 - Matches any port between and including 80 through 90. <p>Cisco routers support many keywords. For details, refer to your router documentation.</p>

Using access control lists with WCCP

To avoid requiring the router to do extra work, we recommend that you create an ACL that routes only traffic that you intend to optimize to the SteelHead.

Suppose your network is structured so that all internet traffic passes through the WCCP-configured router, and all intranet traffic is confined to 10.0.0.0/8. Because it is unlikely that remote internet hosts have a SteelHead, do not redirect internet traffic to the SteelHead. The following is an example ACL that achieves this goal.

Note: The following router commands are not required for the example network configurations in this chapter. Use caution when you enter the **ip wccp [SG]** router command because each **ip wccp [SG]** router command overwrites the previous **ip wccp [SG]** router command. You cannot use an **ip wccp [SG]** router command to augment **ip wccp [SG]** router commands you previously issued. For details, see [“Configuring a basic WCCP router” on page 272](#).

To configure an ACL to route only intranet traffic to a WCCP-enabled SteelHead interface

- On the WCCP router, at the system prompt, enter the following commands:

```
enable
configure terminal
access-list 101 deny ip host <wccp-steelhead-ip> any
access-list 101 deny ip any host <wccp-steelhead-ip>
access-list 101 permit tcp 10.0.0.0 0.255.255.255 any
access-list 101 permit tcp any 10.0.0.0 0.255.255.255
access-list 101 deny ip any any
```

```

!
ip wccp 61 redirect-list 101
!
end
write memory

```

Repeat these commands for each WCCP SteelHead in the service group.

Note: Enter configuration commands, one per line. Press Ctrl+Z to end the configuration.

Configuring load balancing in WCCP

You can perform load balancing using WCCP. WCCP supports load balancing using either the hash assignment method or the mask assignment method. This section includes the following topics:

- [“Configuring load balancing using the hash assignment method” on page 279](#)
- [“Configuring load balancing using the mask assignment method” on page 279](#)
- [“Using the weight parameter” on page 281](#)

Configuring load balancing using the hash assignment method

With the hash assignment method, traffic is redirected based on a hashing scheme and the weight of the SteelHead interfaces. You can hash on a combination of the source IP address, destination IP address, source port, or destination port. The default weight is based on the SteelHead model (for example, for the Model 5000, the weight is **5000**). You can modify the weight on an interface per service group.

To change the hashing scheme and assign a weight on a WCCP SteelHead interface

1. Connect to the Riverbed CLI on the WCCP SteelHead interface and enter the following command:

```
wccp interface inpath0_0 service-group 61 routers 10.1.0.1 flags dst-ip-hash,src-ip-hash
```

2. To change the weight on the WCCP SteelHead interface, at the system prompt, enter the following command:

```
wccp interface inpath0_0 service-group 61 routers 10.1.0.1 weight 20
```

Configuring load balancing using the mask assignment method

Mask assignment uses 7 bits, which allows for a maximum of 128 buckets ($2^7 = 128$) for load balancing across SteelHead interfaces in the same service group. When deciding the number of bits to use, always keep in mind the number of SteelHead interfaces in the service group. Ensure that you create enough buckets for all the SteelHead interfaces in the service group. For example, with three SteelHeads with two in-path interfaces each in a service group, use at least 3 bits for mask assignment to create 8 buckets ($2^3 = 8$). Having more buckets than SteelHead interfaces is not a problem; in fact, it might be necessary to do so to distribute the load correctly. However, if there are more SteelHead interfaces than available buckets, some SteelHead interfaces remain idle.

Mask assignments have two subcategories:

- **Address mask** - A 4-byte value, each byte of which corresponds to each octet of the IP address.
- **Port mask** - A 2-byte value used to match the port number.

You can combine address masks with port masks as long as the total number of bits used for the mask assignment value does not exceed 7 bits.

Note: The algorithm used for determining bucket allocation and assignment is vendor specific; there is no common standard in the industry. The following explanation is specific to SteelHeads. Other vendors who support load distribution with mask assignment might use a different algorithm to distribute the loads amongst their own devices.

The default mask on the SteelHead is 0x1741. Change this default to suit your network. At a minimum, the number of bits you use in the mask must provide enough buckets to load balance the traffic among the SteelHeads in the cluster. In addition, make sure there are enough buckets created to fairly load balance the traffic.

When determining bucket allocations, mask assignment uses the WCCP *weight* parameter. The higher the weight, the more buckets are allocated to that SteelHead interface. However, even if all the SteelHead interfaces in the service group share the same weight, the distribution among the SteelHead interfaces might not be perfectly equal if the number of buckets is not divisible by the number of SteelHead interfaces in the service group.

A mask of 0x1 creates two buckets ($2^1=2$), which is appropriate for a deployment consisting of a single SteelHead or a cluster of two. A mask of 0x3 creates four buckets ($2^2=4$), but is most likely not appropriate for a three SteelHead deployment because it cannot lead to a fair distribution of the traffic.

When the number of buckets is not divisible by the number of SteelHead interfaces in the service group, the remaining buckets are assigned to the SteelHead interface with the highest effective weight. If all weights are equal, then the interface with the lowest IP address receives the remaining buckets. In other words, the remainder from the following operation is assigned to the SteelHead interface with the highest effective weight:

<number-of-buckets> modulo <number-of-steelhead-interfaces>

Effective weight with multiple in-path WCCP means each of the configured SteelHead interface weights are divided by the number of that SteelHead interfaces participating that service group. For example, a SteelHead has two interfaces participating in service group 61. They have a weight of 100 configured. Their effective weight is $100/2$, or 50, per interface. A SteelHead with a single interface participating in a service group with a weight of 100 configured would simply have an effective weight of 100.

Example: Bucket allocation for 8 buckets and 3 SteelHead interfaces of equal weight

When there are eight buckets and three SteelHead interfaces of effective equal weight (SteelHead1 inpath0_0 (weight 200): 1.1.1.1, SteelHead1 inpath0_1 (weight 200): 2.2.2.2, and SteelHead2 inpath0_0 (weight 100): 3.3.3.3), the *initial* bucket allocation is:

- Interface 1.1.1.1 - two buckets
- Interface 2.2.2.2 - two buckets
- Interface 3.3.3.3 - two buckets

Using the expression $8 \bmod 3 = 2$, the remaining two buckets are assigned to Interface 1.1.1.1. The final allocation is:

- Interface 1.1.1.1 - four buckets (50%)
- Interface 2.2.2.2 - two buckets (25%)

- Interface 3.3.3.3 - two buckets (25%)

Example: Bucket allocation for 16 buckets and 3 SteelHead interfaces of equal weight

The same operation applies to 16 buckets and 3 SteelHead interfaces of equal effective weight.

Using the expression $16 \bmod 3 = 1$, the final allocation is:

- Interface 1.1.1.1 - six buckets (37.5%)
- Interface 2.2.2.2 - five buckets (31.25%)
- Interface 3.3.3.3 - five buckets (31.25%)

The example shows that the number of bits used for the mask and the number of SteelHead interfaces in the service group affect the accuracy of the load distribution.

Using the weight parameter

To assign weight in the mask assignment method, you use the weight parameter in the same way as the hash assignment method: for example,

```
wccp interface inpath0_0 service-group 61 routers 10.1.0.1 weight 20
```

You can also assign weight to each SteelHead interfaces so that the larger model SteelHeads are assigned more buckets. However, doing this is generally unnecessary because the SteelHead models have appropriately larger default weight values relative to their higher capacities. WCCP uses the following formula to assign buckets to each SteelHead interface:

Bucket allocation = (bucket/size/sum of effective weight) * configured weight of the SteelHead interface

Example: Bucket allocation for 8 buckets and 2 SteelHeads with single interfaces with different weights

In this example, SteelHeadA has a single in-path interface, inpath0_0, with a weight of 25 and SteelHeadB has a single in-path interface, inpath0_0, with a weight of 50.

8 buckets

Total SteelHead interface weight: $25 + 50 = 75$

SteelHeadA inpath0_0 weight is 25

$(8/75) * 25 = 2.7$ buckets

SteelHeadB inpath0_0 weight is 50

$(8/75) * 50 = 5.3$ buckets

Because the number of allocated buckets must be integers, the number of buckets is rounded to the nearest integer. In this case, WCCP allocates three buckets to SteelHead A inpath0_0 and five buckets to SteelHeadB inpath0_0.

Example: Bucket allocation for 16 buckets and 3 different SteelHead models, each with two

in-path interfaces

In this example, SteelHeadA has two interfaces, inpath0_0 with IP address of 10.1.1.1 and inpath0_1 with IP address 10.1.1.2, each configured with weight 25. SteelHeadB has two interfaces, inpath0_0 with IP address of 10.1.1.3 and inpath0_1 with 10.1.1.4, each configured with weight 50. SteelHeadC has inpath0_0 with IP address 10.1.1.5 and inpath0_1 with IP address 10.1.1.6, each configured with weight 75. The mask for the service group is 0xE, creating 16 buckets.

The total weight equals the effective weight of all SteelHead interfaces. The effective weight of a SteelHead interface is equal to its configured weight divided by the number of that SteelHead interfaces participating in the service group.

Total weight: $A\text{-in0_0}/2 + A\text{-in0_1}/2 + B\text{-in0_0}/2 + B\text{-in0_1} + C\text{-in0_0} + C\text{-in0_1}$

Total weight: $(25/2) + (25/2) + (50/2) + (50/2) + (75/2) + (75/2) = 150$

Allocation:

SteelHeadA inpath0_0 (10.1.1.1): $(12.5/150) * 16 \text{ buckets} = 1.33 = 1 \text{ bucket}$

SteelHeadA inpath0_1 (10.1.1.2): $(12.5/150) * 16 \text{ buckets} = 1.33 = 1 \text{ bucket}$

SteelHeadB inpath0_0 (10.1.1.3): $(25/150) * 16 \text{ buckets} = 2.66 = 2 \text{ buckets}$

SteelHeadB inpath0_1 (10.1.1.4): $(25/150) * 16 \text{ buckets} = 2.66 = 2 \text{ buckets}$

SteelHeadC inpath0_0 (10.1.1.5): $(37.5/150) * 16 \text{ buckets} = 4 = 4 \text{ buckets}$

SteelHeadC inpath0_1 (10.1.1.6): $(37.5/150) * 16 \text{ buckets} = 4 = 4 \text{ buckets}$

Flow data in WCCP

In virtual in-path deployments such as WCCP, traffic moves in and out of the same WAN interface. The LAN interface is not used. When the SteelHead exports data to a data flow collector, all traffic has the WAN interface index. Although it is technically correct for all traffic to have the WAN interface index because the input and output interfaces are the same, it is impossible to use the interface index to distinguish between LAN-to-WAN and WAN-to-LAN traffic.

You can configure the fake index feature on your SteelHead to insert the correct interface index before exporting data to a data flow collector.

For information about configuring the fake index feature, see [“Configuring flow data exports in virtual in-path deployments” on page 248](#).

Verifying and troubleshooting WCCP configurations

This section describes the basic commands for verifying WCCP configuration on the router and the WCCP SteelHead.

To verify the router configuration

- On the router, at the system prompt, enter the following commands:

```
enable
show ip wccp
```

```
show ip wccp 61 detail
show ip wccp 61 view
```

To verify the WCCP configuration on an interface

- On the router, at the system prompt, enter the following commands:

```
enable
show ip interface
```

Look for WCCP status messages near the end of the output.

To verify the access control list configuration

- On the router, at the system prompt, enter the following commands:

```
enable
show access-lists <access-list-number>
```

To trace WCCP packets and events on the router

- On the router, at the system prompt, enter the following commands:

```
enable
debug ip wccp events
WCCP events debugging is on
debug ip wccp packets
WCCP packet info debugging is on
term mon
```

To verify the WCCP SteelHead configuration

- Connect to the Riverbed CLI on the WCCP SteelHead and enter the following command:

```
show wccp interface <interface> service-group 61 detail
WCCP Support Enabled:          yes
WCCP Multicast TTL:            1
WCCP Return via Gateway Override: no

Router IP Address:              89.1.1.1
Identity:                      1.1.1.1
State:                         Connected
Redirect Negotiated:           12
Return Negotiated:             12
Assignment Negotiated:         mask
i-see-you Message Count:       20
Last i-see-you Message:        2008/07/06 22:05:16 (1 second(s) ago)
Removal Query Message Count:   0
Last Removal Query Message:    N/A (0 second(s) ago)
here-i-am Message Count:       20
Last here-i-am Message:        2008/07/06 22:05:16 (1 second(s) ago)
Redirect Assign Message Count:  1
Last Redirect Assign Message:   2008/07/06 22:02:21 (176 second(s) ago)

Web Cache Client Id: 89.1.1.2
Weight:                     25
Distribution:                 1 (25.00%)

Mask      SrcAddr      DstAddr      SrcPort      DstPort
----      -
0000:    0x02000000    0x00000000    0x0000      0x0001
```

```

Value  SrcAddr  DstAddr  SrcPort  DstPort  Cache-IP
-----
0000:  0x00000000  0x00000000  0x0000  0x0000  89.1.1.2

Web Cache Client Id: 89.1.1.6
Weight:             25
Distribution:        2 (50.00%)

Mask    SrcAddr  DstAddr  SrcPort  DstPort
-----
0000:  0x02000000  0x00000000  0x0000  0x0001

Value  SrcAddr  DstAddr  SrcPort  DstPort  Cache-IP
-----
0002:  0x00000000  0x00000000  0x0000  0x0001  89.1.1.6
0003:  0x02000000  0x00000000  0x0000  0x0001  89.1.1.6

Web Cache Client Id: 89.1.1.5
Weight:             25
Distribution:        1 (25.00%)

Mask    SrcAddr  DstAddr  SrcPort  DstPort
-----
0000:  0x02000000  0x00000000  0x0000  0x0001

Value  SrcAddr  DstAddr  SrcPort  DstPort  Cache-IP
-----
0001:  0x02000000  0x00000000  0x0000  0x0000  89.1.1.5

```

To verify the WCCP bucket allocation

- Connect to the Riverbed CLI on the WCCP SteelHead and enter the **show wccp interface <interface> service-group 61 detail** command.

This command is available only in RiOS 6.1 or later.

The example shows WCCP bucket allocation details.

- WCCP used 2 bits for the mask, 1 in the Source Address and 1 in the Destination Port:

```

Mask    SrcAddr  DstAddr  SrcPort  DstPort
-----
0000:  0x02000000  0x00000000  0x0000  0x0001

```

- WCCP created four buckets ($2^2 = 4$) and allocated them to the SteelHead interfaces as follows:

89.1.1.2 was allocated one bucket:

```

Value  SrcAddr  DstAddr  SrcPort  DstPort  Cache-IP
-----
0000:  0x00000000  0x00000000  0x0000  0x0000  89.1.1.2

```

89.1.1.5 was allocated one bucket:

```

Value  SrcAddr  DstAddr  SrcPort  DstPort  Cache-IP
-----
0001:  0x02000000  0x00000000  0x0000  0x0000  89.1.1.5

```

89.1.1.6 was allocated two buckets because it has the highest IP address of the attached SteelHead interfaces:

Value	SrcAddr	DstAddr	SrcPort	DstPort	Cache-IP
-----	-----	-----	-----	-----	-----
0002:	0x00000000	0x00000000	0x0000	0x0001	89.1.1.6
0003:	0x02000000	0x00000000	0x0000	0x0001	89.1.1.6

The following table lists some of the configurations that the **show wccp service-group <num> details** CLI command displays.

Configuration	Example	
Redirection Method	Redirect Negotiated:	12
Return Method	Return Negotiated:	12
Assignment Method	Assignment Negotiated:	mask
GRE Encapsulation	WCCP Return via Gateway Override: no	
WCCP Control Messages	i-see-you Message Count:	20

For information about troubleshooting WCCP and other deployments, see [“Troubleshooting SteelHead Deployment Problems” on page 463](#).

Policy-Based Routing Virtual In-Path Deployments

This chapter describes the basic steps for policy-based routing (PBR) network deployments and how to configure PBR to redirect traffic to a SteelHead or group of SteelHeads.

This chapter includes the following sections:

- [“Overview of PBR” on page 287](#)
- [“Connecting the SteelHead in a PBR deployment” on page 290](#)
- [“Configuring PBR” on page 290](#)
- [“Exporting flow data and virtual in-path deployments” on page 306](#)

For information about the factors you must consider before you design and deploy the SteelHead in a network environment, see [“Choosing the right SteelHead model” on page 28](#).

Overview of PBR

PBR is a packet redirection mechanism that allows you to define policies to route packets instead of relying on routing protocols. PBR redirects packets to SteelHeads that are in a virtual in-path deployment. This section includes the following topics:

- [“PBR failover and Cisco Discovery Protocol” on page 288](#)
- [“Alternate PBR failover mechanisms” on page 289](#)

You define PBR policies on your router for switching packets. PBR policies can be based on identifiers available in access lists, such as the source IP address, destination IP address, protocol, source port, or destination port.

When a PBR-enabled router interface receives a packet that matches a defined policy, PBR switches the packet according to the rule defined for the policy. If a packet does not match a defined policy, the packet is routed by the IP address specified in the routing table entry that most closely matches the destination address.

Important: To avoid an infinite loop, PBR must be enabled on the router interfaces where client traffic arrives and disabled on the router interface that is connected to the SteelHead.

PBR is enabled as a global configuration and applied on an interface basis. Each virtual in-path interface can be used simultaneously for receiving traffic redirected through PBR; physically, the WAN port is cabled and used to receive the redirected traffic.

The SteelHead that intercepts traffic redirected by PBR is configured with both in-path and virtual in-path support enabled.

PBR failover and Cisco Discovery Protocol

A major issue with PBR is that it can cause a traffic black hole; that is, it drops all packets to a destination if the device it is redirecting to fails. You can avoid the traffic black holes by enabling PBR to track whether or not the PBR next-hop IP address is available. You configure the PBR-enabled router to use the Cisco Discovery Protocol (CDP). CDP is a protocol used by Cisco routers and switches to obtain information such as neighbor IP addresses, models, and IOS versions. The protocol runs at the OSI Layer 2 using the 802.3 Ethernet frame. You also enable CDP on the SteelHead.

CDP must be enabled on the SteelHead that is used in the PBR deployment. You enable CDP using the **in-path cdp enable** command. For details, see the *Riverbed Command-Line Interface Reference Manual*.

CDP enables SteelHeads to provide automatic failover for PBR deployments. You configure the SteelHead to send out CDP frames. The PBR-enabled router uses these frames to determine whether the SteelHead is operational. If the SteelHead is not operational, the PBR-enabled router stops receiving the CDP frames, and PBR stops switching traffic to the SteelHead.

The SteelHead must be physically connected to the PBR-enabled router for CDP to send frames. If a switch or other Layer-2 device is located between the PBR-enabled router and the SteelHead, CDP frames cannot reach the router. If the CDP frames do not reach the router, the router assumes that the SteelHead is not operational.

CDP is not supported as a failover mechanism on all Cisco platforms. For information about whether your Cisco device supports this feature, refer to your router documentation.

To enable CDP on the SteelHead

- Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
in-path cdp enable
write memory
restart
```

Note: You must save your changes and restart the SteelHead for your changes to take effect.

To enable CDP failover on the router

1. On the PBR router, at the system prompt, use the **set ip next-hop verify-availability** command. For details, refer to your router documentation.

ICMP and HTTP GET can both also be used to track whether or not the PBR next-hop IP address is available.

When you configure the **set ip next-hop verify-availability** Cisco router command, PBR sends a packet in the following manner:

2. PBR checks the CDP neighbor table to verify that the PBR next-hop IP address is available.
3. If the PBR next-hop IP address is available, PBR sends an ARP request for the address, obtains an answer for it, and redirects traffic to the PBR next-hop IP address (the SteelHead).
4. PBR continues sending traffic to the next-hop IP address as long as the ARP requests obtain answers for the next-hop IP address.
5. If the ARP request fails to obtain an answer, PBR checks the CDP table. If there is no entry in the CDP table, PBR stops using the route map to send traffic. This verification provides a failover mechanism.

A Cisco 6500 router and switch combination that is configured in hybrid mode does not support PBR with CDP. A hybrid setup requires that you use a native setup for PBR with CDP to work. This configuration fails because all routing is performed on the Multilayer Switch Feature Card (MSFC). The MSFC is treated as an independent system in a hybrid setup. Therefore, when you run the **show cdp neighbors** Cisco command on the MSFC, it displays the supervisor card as its only neighbor. PBR does not detect the devices that are connected to the switch ports. As a result, PBR does not redirect any traffic for route maps that use the **set ip next-hop verify-availability** Cisco command. For details, refer to your router documentation.

Alternate PBR failover mechanisms

Several other PBR failover methods exist:

- Object tracking
- Dedicated Layer-3 subnet
- Scriptable programming language

Object tracking is a Cisco IOS feature. The Cisco router generates synthetic traffic through a variety of methods (HTTP GET, ping, TCP connect, and so on) to determine if the SteelHead interface is available. If the SteelHead interface is declared unavailable by object tracking, then the router moves to the next SteelHead, or routes the packet normally.

For more information about object tracking, see [“Configuring a SteelHead with object tracking” on page 295](#).

You can deploy the SteelHead on a dedicated Layer-3 subnet. A dedicated Layer-3 subnet provides a simple approach to failover without incurring additional CPU load on the Layer-3 device. The SteelHead must be the only device in the subnet and connected to the Layer-3 device. If the SteelHead becomes unavailable, the dedicated Layer-3 subnet disappears from the routing table and the packet is routed normally. When all the interfaces for a VLAN or subnet do not connect to a Layer-3 switch, the corresponding route is withdrawn from the routing table and the policy statement is bypassed.

If you use a Layer-3 switch, there cannot be another interface in the WAN optimization VLAN, including 802.1Q trunks.

Some Layer-3 devices include a scriptable programming language: for example, Cisco Embedded Event Manager. You can use a scriptable programming language to actively detect an event and initiate a series of CLI commands to disable PBR. If an event occurs indicating the SteelHead has failed, the PBR configuration is automatically removed. If the event reverses, the PBR configuration is automatically reapplied.

Connecting the SteelHead in a PBR deployment

You can use several types of Ethernet cables to attach to the SteelHead in PBR deployments:

- A straight-through cable to the primary interface. You use this connection to manage the SteelHead, reaching it through HTTPS or SSH.
- A straight-through cable to the WAN0_0 interface if you are connecting to a switch.
- A crossover cable to the WAN0_0 interface if you are connecting to a router. You assign an IP address to the in-path interface; this is the IP address that you redirect traffic to (the target of the router PBR rule).

Configuring PBR

This section describes how to configure PBR and provides example deployments. This section includes the following topics:

- [“Overview of configuring PBR” on page 290](#)
- [“Configuring a SteelHead to directly connect to the router” on page 291](#)
- [“Configuring a SteelHead to connect to a Layer-2 switch” on page 292](#)
- [“Configuring a SteelHead to connect to a Layer-3 switch” on page 294](#)
- [“Configuring a SteelHead with object tracking” on page 295](#)
- [“Configuring a SteelHead with multiple PBR interfaces” on page 296](#)
- [“Configuring multiple SteelHeads to connect to multiple routers” on page 297](#)
- [“Configuring PBR for load balancing WAN circuits” on page 300](#)
- [“Configuring local PBR for ICMP redirection in a mixed MTU environment” on page 304](#)

Overview of configuring PBR

You can use access lists to specify which traffic is redirected to the SteelHead. Traffic that is not specified in the access list is switched normally. If you do not have an access list, or if your access list is not correctly configured in the route map, traffic is not redirected.

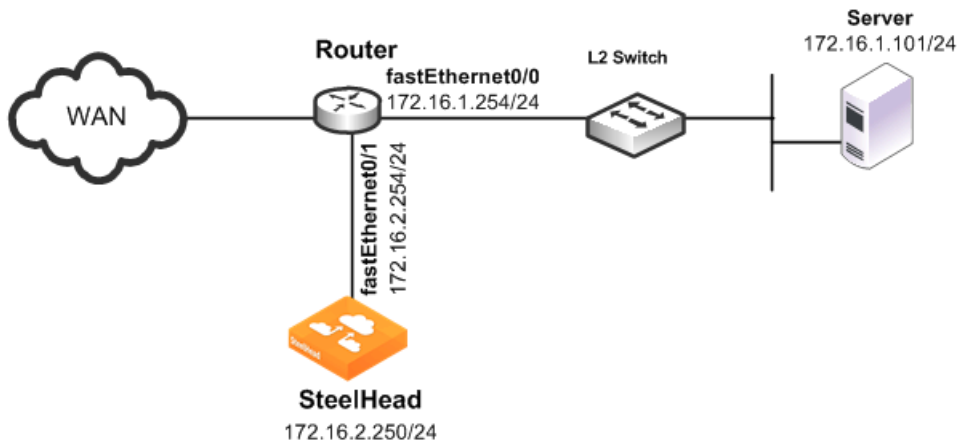
For information about access lists, see [“Configuring access control lists” on page 276](#).

Important: We recommend that you define a policy based on the source or destination IP address rather than on the TCP source or destination ports, because certain protocols use dynamic ports instead of fixed ones.

Configuring a SteelHead to directly connect to the router

Figure 12-1 shows a SteelHead deployment in which the SteelHead is configured with PBR and is directly connected to the router.

Figure 12-1. SteelHead directly connected to the router



In this example:

- The router fastEthernet0/0 interface is attached to the Layer-2 switch.
- The router fastEthernet0/1 interface is attached to the SteelHead.
- A single SteelHead is configured. You can add more SteelHeads using the same method as for the first SteelHead.

Although the primary interface is not included in this example, we recommend, as a best practice, that you connect the primary interface for management purposes.

You must configure subnet side rules when you use VSP in a virtual in-path deployment. To configure subnet side rules, choose **Networking > Networking: Subnet Side Rules**. VSP is enabled by default in the SteelHead EX series.

For information about configuring the primary interface, see the *SteelHead User Guide*.

To configure a SteelHead with PBR connected directly to the router

1. Connect to the SteelHead CLI and enter the following commands:

```

enable
configure terminal
in-path enable
in-path oop enable
interface in-path0_0 ip address 172.16.2.250/24
ip in-path-gateway inpath0_0 172.16.2.254
write memory
restart
  
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. On the PBR router, at the system prompt, enter the following commands:

```

enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.2.250
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa0/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory

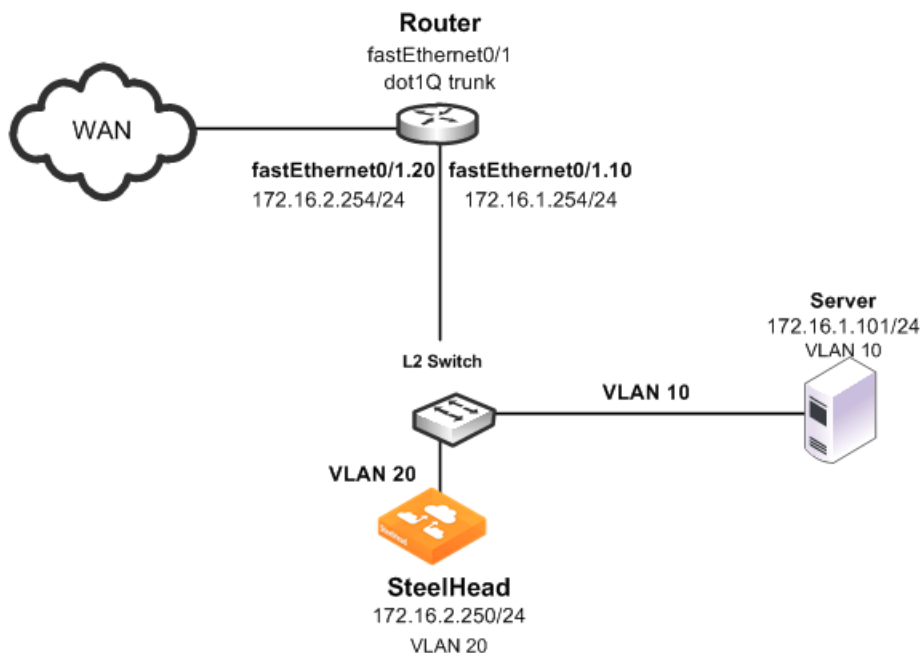
```

Enter one configuration command per line. Press Ctrl+Z to end the configuration.

Configuring a SteelHead to connect to a Layer-2 switch

Figure 12-2 shows a SteelHead deployment in which the SteelHead is configured with PBR and is directly connected to the router through a switch. This deployment also has a trunk between the switch and the router.

Figure 12-2. SteelHead connected to a Layer-2 switch with a VLAN



In this example:

- The switch logically separates the server and the SteelHead by placing:
 - the server on VLAN 10.
 - the SteelHead on VLAN 20.

- The router fastEthernet0/1 interface is attached to the Layer-2 switch.
- The router performs inter-VLAN routing; that is, the router switches packets from one VLAN to the other.
- The link between the router and the switch is configured as a **dot1Q** trunk to transport traffic from multiple VLANs.

Although the primary interface is not included in this example, we recommend that you connect the primary interface for management purposes.

For information about configuring the primary interface, see the *SteelHead User Guide*.

To configure a SteelHead with PBR to a connected Layer-2 switch with a VLAN to the router

1. Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
in-path enable
in-path oop enable
interface in-path0_0 ip address 172.16.2.250/24
ip in-path-gateway inpath0_0 172.16.2.254
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. On the PBR router, at the system prompt, enter the following commands:

```
enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.2.250
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa0/1.10
encapsulation dot1Q 10
ip address 172.16.1.254 255.255.255.0
interface fa0/1.20
encapsulation dot1Q 20
ip address 172.16.2.254 255.255.255.0
exit
interface fa0/1.10
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory
```

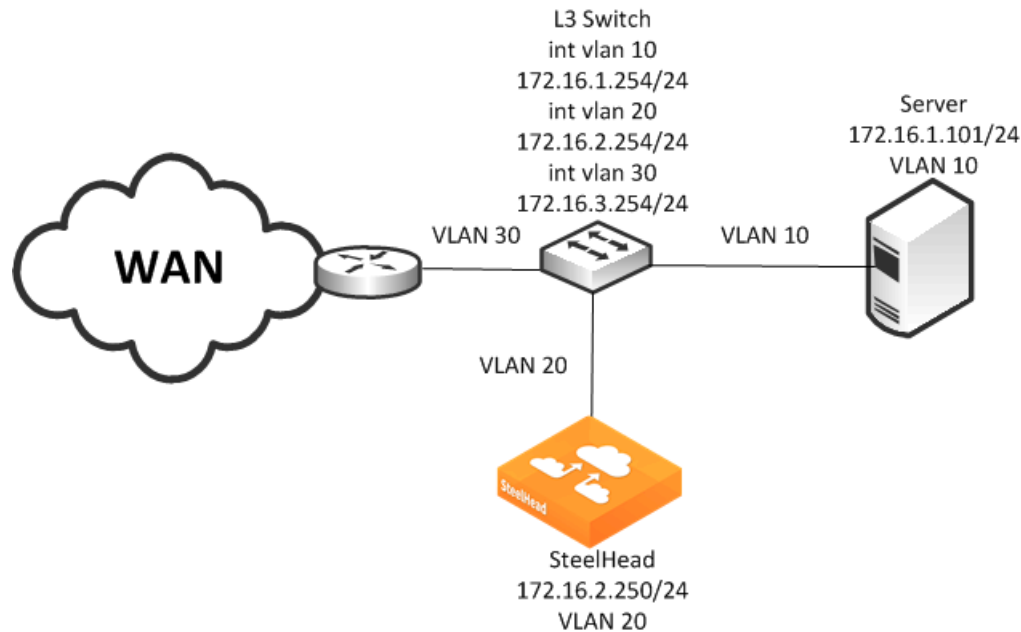
Tip: Enter one configuration command per line. Press Ctrl+Z to end the configuration.

Note: In this example, assume that both the SteelHead and the server are connected to the correct VLAN. Also assume that these VLAN connections are established through the switch port configuration on the Layer-2 switch.

Configuring a SteelHead to connect to a Layer-3 switch

Figure 12-3 shows a SteelHead deployment in which the SteelHead is configured with PBR and is directly connected to a Layer-3 switch.

Figure 12-3. SteelHead connected to a Layer-3 switch



In this example:

- The Layer-3 switch fastEthernet0/0 interface is attached to the server and is on VLAN 10.
- The Layer-3 switch fastEthernet0/1 interface is attached to the SteelHead and is on VLAN 20.
- A single SteelHead is configured. You can add more appliances using the same method as used for the first SteelHead.

Note: Although the primary interface is not included in this example, we recommend that you connect the primary interface for management purposes. For information about configuring the primary interface, see the *SteelHead User Guide*.

To configure a SteelHead with PBR connected directly to a Layer-3 switch

1. Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
in-path enable
in-path oop enable
in-path cdp enable
interface in-path0_0 ip address 172.16.2.250/24
ip in-path-gateway inpath0_0 172.16.2.254
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. On the Layer-3 switch, at the system prompt, enter the following commands:

```
enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.2.250
set ip next-hop verify-availability
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface vlan 10
ip address 172.16.1.254 255.255.255.0
ip policy route-map riverbed
interface vlan 20
ip address 172.16.2.254 255.255.255.0
interface vlan 30
ip policy route-map riverbed
exit
exit
write memory
```

Tip: Enter one configuration commands per line. Press Ctrl+Z to end the configuration.

Configuring a SteelHead with object tracking

In an object tracking deployment, the SteelHead is connected to the router, and the router tracks whether the SteelHead is reachable using the Object Tracking feature of Cisco IOS. Object Tracking enables you to use methods such as HTTP GET and ping, to determine whether the PBR next-hop IP address is available.

Object Tracking is not available on all Cisco devices. For information about whether your Cisco device supports this feature, refer to your router documentation.

To configure the SteelHead with object tracking

For diagram details, see [Figure 12-1](#).

1. Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
interface in-path0_0 ip address 172.16.2.2
50/24
ip in-path-gateway inpath0_0 172.16.2.254
in-path enable
in-path oop enable

no interface inpath0_0 fail-to-bypass enable
write memory
```

2. On the PBR router, at the system prompt, enter the following commands:

```
enable
configure terminal
ip sla 1
icmp-echo 172.16.2.250
ip sla schedule 1 life forever start-time now
track 101 rtr 1 reachability
route-map riverbed
match ip address 101
set ip next-hop verify-availability 172.16.2.250 10 track 101
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa0/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory
```

Configuring a SteelHead with multiple PBR interfaces

In a deployment that uses multiple PBR interfaces, the SteelHead is connected to two routers, each of which is configured to redirect traffic to a separate interface on the SteelHead. Each router is configured similarly to the single router deployment, except that you specify a next-hop IP address that corresponds to the interface to which the SteelHead connects.

To configure the SteelHead with multiple PBR interfaces

For diagram details, see [Figure 12-1](#).

1. Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
in-path enable
Interface inpath0_1 enable
in-path oop enable
interface in-path0_0 ip address 172.16.2.250/24
ip in-path-gateway inpath0_0 172.16.2.254
interface in-path0_1 ip address 172.16.3.250/24
ip in-path-gateway inpath0_1 172.16.3.254
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. On the first PBR router, at the system prompt, enter the following commands:

```
enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.2.250
exit
ip access-list extended 101
```



```
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa0/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory
```

3. On the second PBR router, at the system prompt, enter the following commands:

```
enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.3.250
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa0/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
```

Configuring multiple SteelHeads to connect to multiple routers

In a PBR environment, you can deploy multiple SteelHeads for optimization redundancy. [Figure 12-4](#) shows a SteelHead deployment in which two routers are redirecting packets to two SteelHeads. The SteelHeads are directly connected through multiple interfaces. This deployment provides a high-availability environment—a full-mesh topology between the routers and SteelHeads.

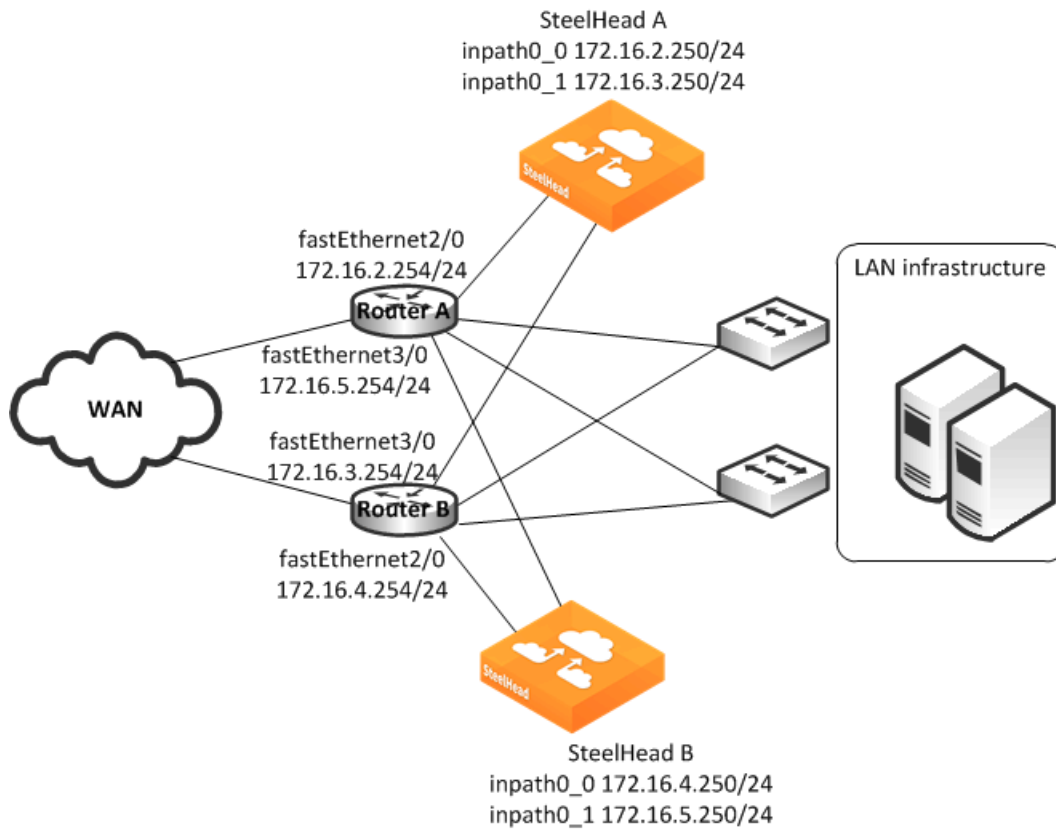
When you deploy multiple SteelHeads, the routers can redirect packets within a TCP connection to different SteelHeads, depending on the router policy. For example, Router A redirects packets to SteelHead A. Following network convergence due to WAN outage packets for this TCP connection, the packets arrive on Router B. Router B has a policy configured to redirect packets to SteelHead B. In this type of environment, we recommend that you use connection forwarding to ensure packets within a flow are forwarded to the owning SteelHead for optimization.

For more information about connection forwarding, see [“Connection forwarding” on page 56](#).

Data store synchronization is not a requirement for SteelHeads in a PBR environment, but it is useful if your design goal is for warm performance after a SteelHead fails. To use data store synchronization, you must meet certain criteria.

For more information about the data store, see [“RiOS data store synchronization” on page 27](#).

Figure 12-4. Multiple SteelHeads directly connected to dual routers



The example in [Figure 12-4](#) shows the following configuration:

- Each router fastEthernet2/0 interface is attached to the SteelHead wan0_0 interface using the inpath0_0 IP address.
- Each router fastEthernet3/0 interface is attached to the SteelHead wan0_1 interface using the inpath0_1 IP address.
- Each router withdraws the PBR next-hop statement because the SteelHeads are directly connected (in case a SteelHead interface or appliance fail). You can use other methods for failover, such as CDP, object tracking, or embedded event manager.
- Connection forwarding is enabled to ensure the owning SteelHead receives all packets for the TCP connection.

You can redirect packets to SteelHead interfaces in any order. [Figure 12-4](#) shows Router A redirecting packets first to SteelHead A inpath0_0 and then to SteelHead B inpath0_1; and Router B redirecting packets first to SteelHead B inpath0_0 and then SteelHead A inpath0_1. Router A and Router B can redirect to SteelHead A inpath0_0 and inpath0_1 respectively.

This example shows two SteelHeads. You can add more SteelHeads using a similar method as for the first SteelHead.

To configure dual SteelHeads with PBR connected to dual routers**1. Connect to SteelHead A CLI and enter the following commands:**

```
enable
configure terminal
in-path enable
in-path oop enable
interface in-path0_0 ip address 172.16.2.250/24
ip in-path-gateway inpath0_0 172.16.2.254
interface in-path0_1 ip address 172.16.3.250/24
ip in-path-gateway inpath0_1 172.16.3.254
in-path neighbor enable
in-path neighbor multi-interface enable
in-path name SteelHeadB main-ip 172.16.4.250
in-path name SteelHeadB additional-ip 172.16.5.250
in-path neighbor allow-failure
no interface inpath0_0 fail-to-bypass enable
no interface inpath0_1 fail-to-bypass enable
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. Connect to SteelHead B CLI and enter the following commands:

```
enable
configure terminal
in-path enable
in-path oop enable
interface in-path0_0 ip address 172.16.4.250/24
ip in-path-gateway inpath0_0 172.16.4.254
interface in-path0_1 ip address 172.16.5.250/24
ip in-path-gateway inpath0_1 172.16.5.254
in-path neighbor enable
in-path neighbor multi-interface enable
in-path name SteelHeadA main-ip 172.16.2.250
in-path name SteelHeadA additional-ip 172.16.3.250
in-path neighbor allow-failure
no interface inpath0_0 fail-to-bypass enable
no interface inpath0_1 fail-to-bypass enable
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

3. On Router A, at the system prompt, enter the following commands:

```
enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.2.250 172.16.5.250
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa2/0
ip address 172.16.2.254 255.255.255.0
interface fa3/0
ip address 172.16.5.254 255.255.255.0
```

```

exit
interface fa1/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory

```

4. On Router B, at the system prompt, enter the following commands:

```

enable
configure terminal
route-map riverbed
match ip address 101
set ip next-hop 172.16.4.250 172.16.3.250
exit
ip access-list extended 101
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
interface fa2/0
ip address 172.16.3.254 255.255.255.0
interface fa3/0
ip address 172.16.4.254 255.255.255.0
exit
interface fa1/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory

```

Configuring PBR for load balancing WAN circuits

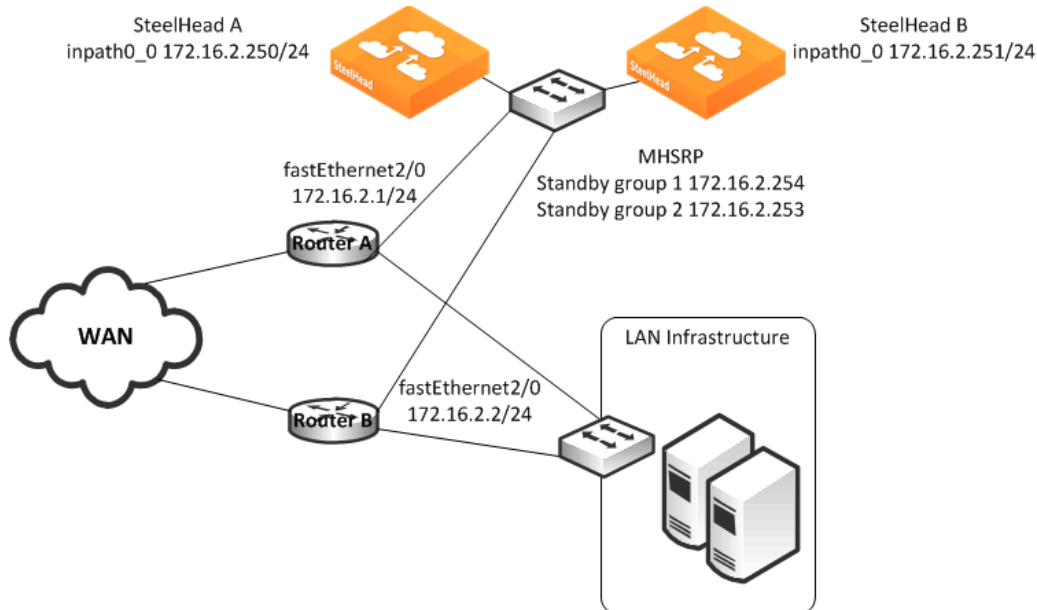
In a network with multiple entry and exit points, you can configure the router and SteelHead to support two goals:

- Support peer affinity so one SteelHead more efficiently optimizes the traffic for a remote peer
- Maintain outbound load balancing by allowing the SteelHeads to use separate WAN circuits to reach the remote sites

Figure 12-5 shows all SteelHeads in one subnet. The routers are configured with access control lists (ACLs) to direct traffic from remote peer A to SteelHead A and traffic for remote peer B to SteelHead B. The failover method is object tracking. An optional all-inclusive rule is added to simplify the addition of a new remote site. To support outbound load balancing, each SteelHead uses a different default gateway.

Optimized traffic from SteelHead A reaches the WAN through Router A, and SteelHead B reaches the WAN through router B. Multiple HSRP (MHSRP) allows the routers to present two default gateways for the WAN optimization subnet, at the same time having redundancy in case a WAN circuit fails. Instead of MHSRP, you could use Gateway Load Balancing Protocol (GLBP) or static routes on the SteelHeads. If you use one of these two methods, the SteelHeads are unaware of a WAN circuit outage.

Figure 12-5. PBR load balancing



In this example:

- Each router fastEthernet2/0 interface and SteelHead wan0_0 interface are attached to same VLAN on the switch.
- Each router uses object tracking to bypass the PBR next-hop statement because the SteelHeads are not directly connected to the router. Embedded event manager is another option.
CDP frames from the SteelHead are consumed by the switch and do not reach the routers. This renders CDP ineffective for failover.
- Packets are redirected to a SteelHead based on remote IP address subnet for peer affinity.
- Multiple HSRP groups are used on the router fastEthernet2/0 interfaces to achieve outbound load balancing on the WAN. (GLBP, or configuring static routes on the SteelHead in-path interface can be used instead of multiple HSRP.)
- Router WAN interfaces are tracked when using HSRP to ensure the router has a path to the remote site.
- Multi-interface is enabled in case another SteelHead interface is connected to another WAN optimization subnet.

Two SteelHeads are configured in this example. You can add more SteelHeads using the same method as used for one of the SteelHeads ensuring policy route statements, any ACLs, and the default gateway align with load balancing and redundancy goals.

To configure PBR appliances in a load-balanced environment

1. Connect to SteelHead A CLI and enter the following commands:

```
enable
configure terminal
in-path enable
in-path oop enable
interface in-path0_0 ip address 172.16.2.250/24
ip in-path-gateway inpath0_0 172.16.2.254
in-path neighbor enable
in-path neighbor multi-interface enable
in-path name SteelHeadB main-ip 172.16.2.251
in-path neighbor allow-failure
no interface inpath0_0 fail-to-bypass enable
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

2. Connect to SteelHead B CLI and enter the following commands:

```
enable
configure terminal
in-path enable
in-path oop enable
interface in-path0_0 ip address 172.16.2.251/24
ip in-path-gateway inpath0_0 172.16.2.253
in-path neighbor enable
in-path neighbor multi-interface enable
in-path name SteelHeadA main-ip 172.16.2.250
in-path neighbor allow-failure
no interface inpath0_0 fail-to-bypass enable
write memory
restart
```

You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

3. On PBR router A, at the system prompt, enter the following commands:

```
enable
configure terminal
ip access-list extended Remotes-To-A
permit tcp address 10.1.100.0 0.0.0.255 any
permit tcp address any 10.1.100.0 0.0.0.255
exit
ip access-list extended Remotes-To-B
permit tcp address 10.1.200.0 0.0.0.255 any
permit tcp address any 10.1.200.0 0.0.0.255
exit
ip access-list extended Catch-All
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
ip sla 1
icmp-echo 172.16.2.250 source-interface fastEthernet2/0
exit
ip sla schedule 1 life forever start-time now
ip sla 2
icmp-echo 172.16.2.251 source-interface fastEthernet2/0
exit
ip sla schedule 2 life forever start-time now
```

```

track 1 ip sla 1 reachability
track 2 ip sla 2 reachability
track 100 interface S0/0 line-protocol
route-map riverbed permit 10
match ip address Remotes-To-A
set ip next-hop verify-availability 172.16.2.250 1 track 1
set ip next-hop verify-availability 172.16.2.251 2 track 2
route-map riverbed permit 20
match ip address Remotes-To-B
set ip next-hop verify-availability 172.16.2.251 1 track 2
set ip next-hop verify-availability 172.16.2.250 2 track 1
route-map riverbed permit 30
match ip address Catch-All
set ip next-hop verify-availability 172.16.2.250 1 track 1
set ip next-hop verify-availability 172.16.2.251 2 track 2
exit
interface fa2/0
ip address 172.16.2.1 255.255.255.0
standby 1 ip 172.16.2.254
standby 1 priority 110
standby 1 preempt
standby 1 track 100 20
standby 2 ip 172.16.2.253
standby 2 preempt
exit
interface fa1/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory

```

4. On PBR Router B, at the system prompt, enter the following commands:

```

enable
configure terminal
ip access-list extended Remotes-To-A
permit tcp address 10.1.100.0 0.0.0.255 any
permit tcp address any 10.1.100.0 0.0.0.255
exit
ip access-list extended Remotes-To-B
permit tcp address 10.1.200.0 0.0.0.255 any
permit tcp address any 10.1.200.0 0.0.0.255
exit
ip access-list extended Catch-All
permit tcp any 172.16.1.101 0.0.0.0
permit tcp 172.16.1.101 0.0.0.0 any
exit
ip sla 1
icmp-echo 172.16.2.250 source-interface fastEthernet2/0
exit
ip sla schedule 1 life forever start-time now
ip sla 2
icmp-echo 172.16.2.251 source-interface fastEthernet2/0
exit
ip sla schedule 2 life forever start-time now
track 1 ip sla 1 reachability
track 2 ip sla 2 reachability
track 100 interface S0/0 line-protocol
route-map riverbed permit 10
match ip address Remotes-To-A
set ip next-hop verify-availability 172.16.2.250 1 track 1
set ip next-hop verify-availability 172.16.2.251 2 track 2

```

```
route-map riverbed permit 20
match ip address Remotes-To-B
set ip next-hop verify-availability 172.16.2.251 1 track 2
set ip next-hop verify-availability 172.16.2.250 2 track 1
route-map riverbed permit 30
match ip address Catch-All
set ip next-hop verify-availability 172.16.2.250 1 track 1
set ip next-hop verify-availability 172.16.2.251 2 track 2
exit
interface fa2/0
ip address 172.16.2.2 255.255.255.0
standby 1 ip 172.16.2.254
standby 1 preempt
standby 2 ip 172.16.2.253
standby 2 priority 110
standby 2 preempt
standby 2 track 100 20
exit
interface fa1/0
ip policy route-map riverbed
interface S0/0
ip policy route-map riverbed
exit
exit
write memory
```

Configuring local PBR for ICMP redirection in a mixed MTU environment

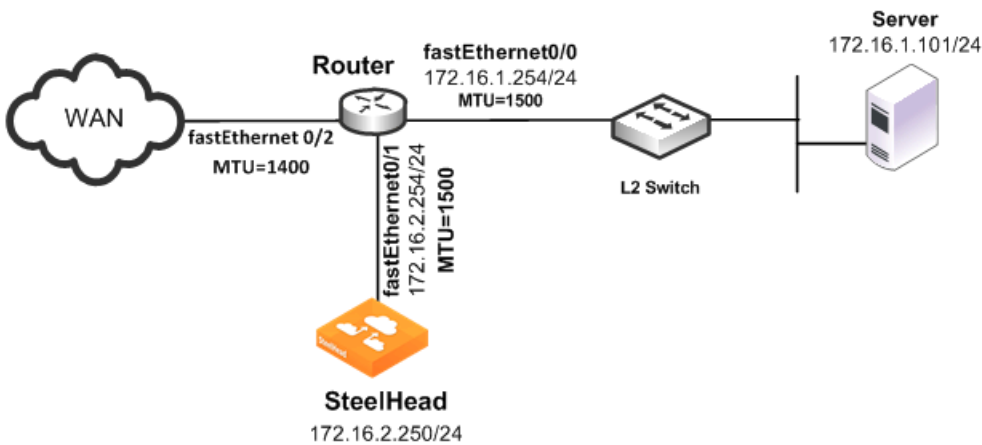
You can add a Local PBR configuration in addition to the regular PBR configuration discussed in previous sections. You use a Local PBR configuration in environments where ICMP messages generated by the router need special routing configurations: for example, mixed-size maximum transmission unit (MTU) environment with SteelHead full-transparency in-path rules.

In networks that have a mix of MTU interface configurations, path MTU (PMTU) discovery determines the MTU size on the network path between two IP hosts, to avoid IP fragmentation. Network routers send *ICMP Fragment needed but do not fragment bit set* messages and packets back to the sending host; the host then decreases the segment size and retransmits the segment.

You must forward ICMP packets to the correct host (the client, server, or SteelHead). In some network scenarios, you need a Local PBR configuration to ensure ICMP messages, generated by a router, are redirected out of the same interface of the inbound IP datagram that triggers the ICMP message. Cisco Local PBR is a local policy route map configuration that can achieve this.

Figure 12-6 shows an example scenario for which you can configure Local PBR.

Figure 12-6. Local PBR configuration



In this example, the:

- router fastEthernet0/0 interface is attached to the layer 2 switch. The MTU is configured for 1500 bytes.
- router fastEthernet0/1 interface is attached to the SteelHead. The MTU is configured for 1500 bytes.
- router fastEthernet 0/2 interface is attached to the WAN. The MTU is configured for 1400 bytes.
- SteelHead is configured to optimize with full transparency.

Without a Local PBR configuration on the router, optimized connections on the SteelHead can attempt to send IP datagrams of 1500 bytes across the WAN. Due to PMTU discovery, the router generates an *ICMP fragment needed but do not fragment bit set* message.

Because the SteelHead is configured with full transparency, the router forwards the ICMP message back to the server instead of the SteelHead. This results in the server reducing its segment size for the connection, when in fact, the SteelHead must be the one to reduce its segment size. This confusion results in transfer failure.

You can use the following Local PBR configuration on the router to ensure that packets received on fastethernet0/1, and trigger any ICMP message on the router, are forwarded to the SteelHead.

To configure Local PBR on the router

- On the PBR router, at the system prompt, enter the following commands:

```

enable
configure terminal
ip local policy route-map local_pbr_1
access-list 101 permit icmp host 172.16.2.254 any
route-map local_pbr_1 permit 10
match ip address 101
set ip next-hop 172.16.2.250
exit
exit
write memory
  
```

Exporting flow data and virtual in-path deployments

In virtual in-path deployments, such as PBR, traffic moves in and out of the same WAN0_0 interface. The LAN interface is not used. When the SteelHead exports data to a flow data collector, all traffic has the wan0_0 interface index, making it impossible for an administrator to use the interface index to distinguish between LAN-to-WAN and WAN-to-LAN traffic.

You can configure the fake index feature on your SteelHead to insert the correct interface index before exporting data to a flow data collector.

For details, see [“Configuring flow data exports in virtual in-path deployments” on page 248](#).

IPv6

Internet Protocol version 6 (IPv6) has become an important factor to consider as the pool of IPv4 addresses becomes exhausted. You must also plan to meet the organizational mandates for compliance with IPv6. This chapter describes how to configure SteelHeads running RiOS 8.5 or later for optimization of TCP over IPv6 using autodiscovery and fixed-target rules. The benefit of autodiscovery and fixed-target rules is that you can provide both application streamlining for select protocols and transport streamlining, in addition to data streamlining. Packet mode optimization is still an alternative for data streamlining of IPv6 traffic. This chapter includes the following sections:

- [“Overview of IPv6” on page 307](#)
- [“In-path rules” on page 313](#)
- [“Deployment options” on page 313](#)
- [“Protocol support” on page 320](#)
- [“Verification and troubleshooting” on page 320](#)

Overview of IPv6

This section provides an overview of various aspects of IPv6. This section includes the following topics:

- [“RiOS RFC compliance and feature compatibility” on page 308](#)
- [“IPv6 addressing” on page 310](#)
- [“Traffic interception” on page 311](#)

IPv6 is the next revision of the Internet Protocol. IPv6 has become a critical enabler as more devices are being connected to the internet. The main purpose of IPv6 is to help service providers and companies manage the exhaustion of IPv4 addresses without relying on other techniques such as network address translation (NAT), which hide and manipulate the source IP address when connecting across organizational boundaries. IPv6 was designed to overcome the limitation in IPv4 by using a 128-bit address instead of a 32-bit address, thereby supporting up to 3.4×10^{38} addresses; IPv4 supports up to 4.3 billion only.

The use of private IPv4 address space has created challenges for applications, security, and performance because the source and destination addresses can change through the connection. RiOS extended support for IPv6 traffic with packet mode optimization, RiOS 8.5 further enhanced its IPv6 capabilities by supporting autodiscovery and fixed target rules and RiOS 9.5 supports autodiscovery with IPv6 for the optimized connection over the WAN. By using autodiscovery or fixed target rules, RiOS can apply transport and application streamlining techniques (similarly as it does for TCP connections over IPv4) to improve the user experience as the transition to IPv6 continues.

RiOS RFC compliance and feature compatibility

The following RiOS features are compatible with IPv6.

RiOS IPv6 support includes	RiOS version	Notes
Full and port transparency support	9.7 and later	
Enhanced autodiscovery of SteelHeads	9.5 and later for IPv6-only (single-stack) networks 8.5 and later for IPv4 only or dual-stack IPv4 and IPv6 networks	Starting with RiOS 9.5, enhanced autodiscovery is supported for SteelHeads in networks that run IPv6 only (IPv6 single-stack). SteelHeads running RiOS 8.5 to 9.2 require IPv4 for the TCP inner connections between the peer SteelHeads.
IPv6 support for the SteelHead communication channel with the SteelCentral Controller for SteelHead, appliance manageability (for example, NTP servers, logging, hosts, DNS, Web/FTP proxy, email, and management interfaces) policy pages, and Interceptor Cluster pages (for example, in-path rules and load balancing).	9.5 and later	
Encrypted Outlook Anywhere latency optimization.	8.6 and later	
MAPI, eMAPI latency optimization.	8.6 and later	Authentication is over IPv4.
Authentication over IPv6.	8.6 and later	
Latency optimization of signed-SMB, CIFS/SMB1, SMB2, and SMB3 using IPv6 endpoint addressing.	8.5.2 and later	The authentication stack continues to require IPv4 endpoint addressing.
Conformance with Request for Comments (RFCs) 1981, 2460, 2464, 2710, 3590, 4007, 4291, 4443, 4861, 4862, 4943, 5095, and 5156.	8.5 and later	
TCP IPv6 traffic interception between source and destination, bandwidth optimization.	8.5 and later	

RiOS IPv6 support includes	RiOS version	Notes
Ability to automatically discover fixed-target and pass-through in-path rules, along with ability to deny and reject IPv6 TCP traffic as configured in the in-path rules.	8.5 and later	RiOS doesn't support the neural framing modes Always, TCP Hints, and Dynamic. RiOS doesn't support the Oracle forms and Oracle forms over SSL pre-optimization policies.
HTTP and HTTPS latency optimization for IPv6 TCP traffic.	8.5 and later	
Ability to configure serial clusters.	8.5 and later	
Interception of IPv6 traffic for in-path, virtual in-path, and server-side out-of-path configurations.	8.5 and later	WCCPv6 support is not available. Virtual in-path support is PBR. Interceptor deployments are supported in RiOS 9.5 and Interceptor 6.0.
Intercepting and passing through IPv4 and/or IPv6 traffic, depending on the in-path rules.	8.5 and later	
Ability to detect asymmetric routes for IPv6 TCP traffic; enables connection forwarding of IPv6 TCP traffic in asymmetric conditions.	8.5 and later	The connection-forwarding control channel between the neighbors is strictly IPv4. You must configure IPv4 addresses on the SteelHead appliances' in-path interfaces when using a connection-forwarding control channel.
Ability to configure IPv4 and IPv6 addresses on every in-path interface and intercepting and optimizing IPv4 and IPv6 traffic.	8.5 and later	
Ability to configure one IPv6 address configuration for every in-path interface. RiOS intercepts and optimizes traffic matching the scope of the IPv6 address configured on the in-path interface. Not applicable for a link-local address configured on the in-path interface.	8.5 and later	RiOS passes through IPv6 TCP traffic not matching the scope of the IPv6 address configured on the in-path interface.
Ability to configure IPv6 addresses on any in-path interface.	8.5 and later	RiOS 8.5 - RiOS 9.2: IPv6 TCP inner connections only in fixed target cases.
Enhanced autodiscovery of SteelHead appliances for IPv6 TCP traffic.	8.5 and later	RiOS 8.5 - RiOS 9.2: TCP inner connections between the peer SteelHead appliances is IPv4 only. RiOS 9.5 allows for IPv6 TCP inner connections between peers.
Simplified routing for IPv6 TCP traffic.	8.5 and later	

RiOS IPv6 support includes	RiOS version	Notes
Connection forwarding for IPv6 traffic in multi-interface mode.	8.5 and later	The control connection between neighbors is still IPv4 only. When multiple interface support in the Networking > Network Integration: Connection Forwarding page is not enabled, IPv6 traffic is passed through.
Ability to configure peering rules for IPv6 traffic.	8.5	The peer client-side SteelHead IP address is IPv4 only.
Ability to configure IPv6 addresses in Single Ended Interception (SEI) rules under Optimization > Network Services: Transport Settings.	8.5 and later	
Global and automatic kickoff for pass-through TCP IPv6 traffic.	8.5 and later	
Ability to configure asymmetric VLANs for IPv6 TCP traffic.	8.5 and later	

IPv6 addressing

IPv6 addresses share many characteristics with their IPv4 counterparts. Each version separates the address into a network identifier and host identifier. Depending on the type of address, IPv6 specifies certain bit ranges for specific purposes.

In many ways, this is similar to how IPv4 uses address classes. It is important to note that IPv6 addresses use hexadecimal digits instead of a dotted-decimal format. Also, an interface on a device has multiple IPv6 addresses (instead of a single IPv4 address) that it uses to communicate. For example, each interface always has a link-local address and can have one or multiple global unicast addresses.

There are several types of IPv6 addresses. We recommend that you verify with the IPv6 address registry, because new IPv6 transition mechanisms or address blocks are reserved. The types of addresses are as follows:

- Global unicast--2000::/3
 - 2002::/16 reserved for 6to4
 - 2001:0000::/32 reserved for TEREDO
- ::ffff:0:0/96 reserved for IPv4-mapped address
- Unique local--fc00::/7
- Link local--fe80::/10
- Multicast--ff00::/8
- Loopback--::1/128 reserved for loopback address
- Unspecified--::/128 reserved for unspecified address

Source: <http://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xml> (October 2015)

Note: The types of IPv6 addresses are subject to change.

RiOS supports a single-system, generated link-local address and a user-assigned IPv6 address for each interface (primary, auxiliary, and in-path). The link-local address is automatically assigned as the network identifier using FE80 in the first 64 bits and a modified EUI-64 identifier for the host bits following RFC 4291. This address is automatically assigned by the system and cannot be changed.

The link-local address is used as part of IPv6 neighbor discovery, which is analogous to address resolution protocol for IPv4. You must manually configure the user-assigned IPv6 address—it cannot be derived from stateless automatic configuration or dynamic host communication protocol for IPv6. There are two address types, depending on whether you need to natively communicate outside of an organizational boundary—in much the same way a public or private IPv4 address is selected. For SteelHeads, choose one of the following IPv6 addresses:

- An *aggregatable global unicast* address for the SteelHead to communicate with devices not directly connected to the interface.
- A *unique local unicast* address, which is the updated address range to replace site local addresses.

The intent of a unique local unicast address is for a private IPv6 address range.

The IPv6 gateway is also user assigned, and you can link it to the local address of the router or the address on the same subnet as the manually assigned address. RiOS does not support receiving router advertisements as a means to discover the IPv6 gateway. Configure the link-local address for a virtual router in circumstances in which you use a first hop redundancy protocol: for example, HSRPv6.

Traffic interception

RiOS 8.5 introduces the ability to intercept TCP-over-IPv6 traffic and perform optimization by performing transport streamlining on the connection instead of performing data reduction on a packet-by-packet basis. The SteelHeads perform their roles in the connection by establishing three separate connections:

- Outer channel from the client to the client-side SteelHead
- Outer channel from the server to the server-side SteelHead
- Inner channel between SteelHeads

The key difference between packet mode optimization and IPv6-based optimization is that IPv6 supports autodiscovery and fixed-target rules. Packet mode performs data reduction on each packet, and TCP-over-IPv6 supports application, data, and transport streamlining by inserting the SteelHeads into the connection.

Support for WAN visibility modes with autodiscovery of TCP-over-IPv6 traffic is limited to correct addressing. Identical to IPv4 autodiscovery, IPv6 autodiscovery uses the same TCP options present in the TCP header of the connection setup packets to discover a remote SteelHead on the path between the end systems. However, beginning in RiOS 9.5, the SteelHead appliances can be configured to use autodiscovery to learn the IPv6 address of the remote peer. The SteelHead appliances insert their IPv6 address in an IPv6 header option; the specific header option is the destination option.

Figure 13-1. IPv6 autodiscovery header example

```

Destination Option
Next header: TCP (6)
Length: 5 (48 bytes)
IPv6 Option (Unknown 16)
Type: Unknown (16)
Length: 43
Unknown Option Payload: 4c260f002000000000000000000000000000000000000101020000000...
0000 00 50 b6 0e 49 8c 00 0e b6 b3 81 58 86 dd 60 00 .P..I... ..X...`
0010 00 00 00 50 3c 80 20 00 00 00 00 00 00 00 00 ...P<.. ....
0020 00 00 00 00 11 11 20 00 00 00 00 00 00 00 00 .....
0030 00 00 00 00 10 11 06 05 10 2b 4c 26 0f 00 20 00 .....+L&...
0040 00 00 00 00 00 00 00 00 00 00 00 00 10 10 20 00 .....
0050 00 00 00 00 00 00 00 00 00 00 00 00 11 10 1e 78 .....x
0060 4c 05 3f 00 3d 00 01 bd c4 c9 01 31 7d 28 c9 81 L.?.=...}(.
0070 89 85 80 12 20 00 56 11 00 00 02 04 05 a0 01 03 ....V.....
0080 03 02 01 01 04 02 .....

```

More information on other IPv6 header options is available at the following link:

<http://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml#extension-header>

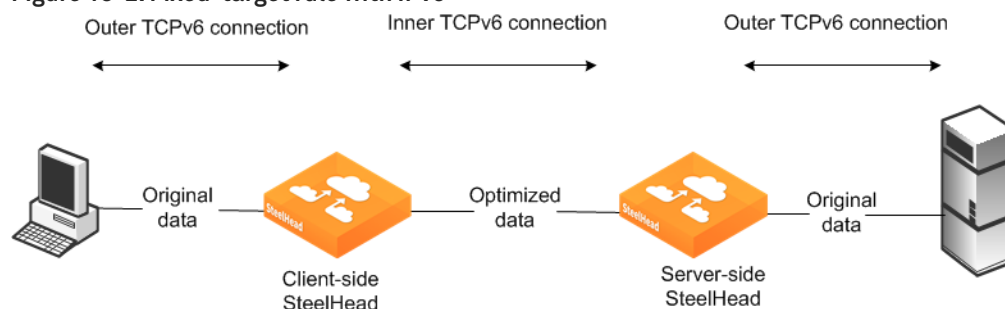
IPv6 autodiscovery is disabled by default and must be enabled on the peering rules page.

For more information about packet mode optimization, see [“Packet Mode Optimization” on page 323](#). For more information about WAN visibility modes and correct addressing, see [“WAN Visibility Modes” on page 63](#).

RiOS 9.5 and later support autodiscovery for IPv6 single-stack networks. For RiOS 8.5 to 9.2, you can use a fixed-target rule for single-stack networks that enables you to optimize traffic end-to-end using IPv6 addresses. The inner channel between SteelHeads forms a TCPv6 connection using the discovered IPv6 address of the peer in the IPv6 destination option or the manually assigned IPv6 address in the fixed target rule. Starting with RiOS 9.7, port and full transparency modes are supported for IPv6, and inner channels that use IPv6 can be addressed in a similar way as inner channels that use IPv4.

The latter method is similar to an IPv4 fixed-target rule, and you configure it the same way. Although you can use IPv6 addresses end-to-end in the connection, the SteelHead in-path interface continues to require an IPv4 address for the optimization service to start.

Figure 13-2. Fixed-target rule with IPv6



In-path rules

The default rule was previously *all-IPv4* but is now *all-IP*. All-IP includes all IPv4 and all IPv6 traffic. The default rule for TCP traffic, either IPv4 or IPv6, attempts autodiscovery with correct addressing as the WAN visibility mode. IPv6 autodiscovery for single-stack networks only uses correct addressing. Starting with RiOS 9.7, port and full transparency modes are supported for IPv6, and inner channels that use IPv6 can be addressed in a similar way as inner channels that use IPv4.

A fixed-target rule with an IPv6 target appliance address requires the source and destination address type to be an IPv6 address. You must change the use of all-IP to all-IPv6. If you do not change to all-IPv6, use specific source addresses or destination addresses or both.

Deployment options

You can configure SteelHeads for in-path or virtual in-path deployment for TCP-over-IPv6 traffic. Riverbed also supports server-side out-of-path deployments. This section includes the following topics:

- [“Configuring an in-path SteelHead IPv6 deployment” on page 313](#)
- [“Configuring a SteelHead serial cluster IPv6 deployment” on page 315](#)
- [“Configuring a connection forwarding and SteelHead IPv6 deployment” on page 316](#)
- [“Configuring a virtual in-path SteelHead IPv6 deployment” on page 317](#)
- [“Configuring a fixed-target rule SteelHead IPv6 deployment” on page 319](#)

Considerations for the deployment type are the same as the considerations for optimizing of IPv4 connections. Network integration features such as fail-to-wire, link state propagation, parallel deployments, firewalls, and so on continue to be relevant for optimization of TCP-over-IPv6 traffic. IPv4 connections can coexist with TCP-over-IPv6 traffic.

The following list can help you simplify the choice of deployment options:

- If you have a single-stack (RiOS version 9.5 or later) or dual-stack network, use enhanced autodiscovery.
- A parallel SteelHead deployment requires IPv4 between the local SteelHead in-path interfaces, for connection forwarding.
- Virtual in-path deployments support policy-based routing. WCCP deployments are not supported with TCP over IPv6.

Configuring an in-path SteelHead IPv6 deployment

The in-path deployment for optimization of TCP over IPv6 traffic is similar to an in-path deployment for IPv4 connections. You deploy the SteelHead physically in-path using an IPv6 address on the SteelHead in-path interface. Optionally, for IPv6 management, you can configure an IPv6 address on the primary interface.

In addition to using the primary interface, you can also use the auxiliary (AUX) interface for management. The AUX interface must be on a different subnet than the primary interface. Likewise, the management in-path interface can be used and it also must be on a different subnet.

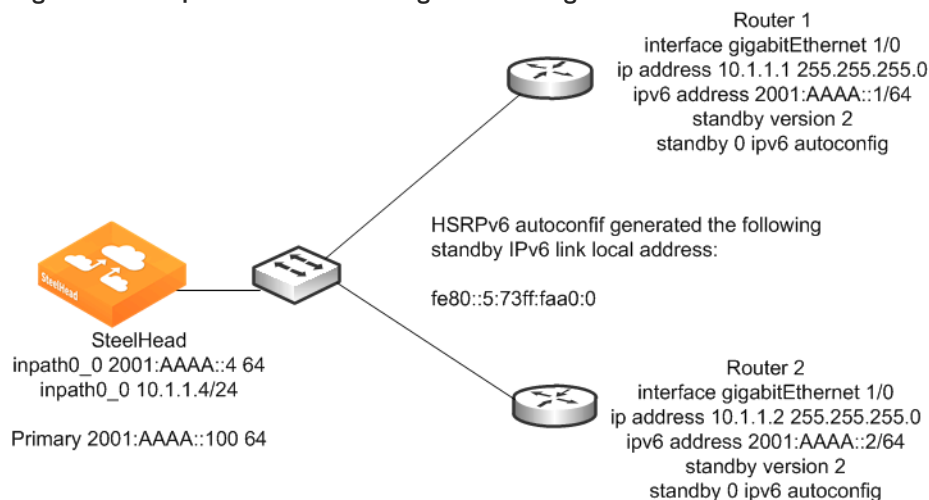
The in-path interface has its own IPv6 routing table. You can add destinations to the table and select an appropriate next hop. You can also add simplified routing to reduce the burden of administering IPv6 routes.

For information about simplified routing, see [“Overview of simplified routing” on page 60](#).

Figure 13-3 shows an example deployment with IPv6 addresses from the globally assigned address range manually assigned to the primary and inpath0_0 interfaces. The primary interface is assigned 2001:aaaa::100 with a prefix length of 64 bits. The inpath0_0 interface is assigned 2001:aaaa::10 with a prefix length of 64 bits. The primary interface has the global unicast address of Router 1 configured as its IPv6 gateway, but you could use a link-local address.

In this example, the link-local address of the HSRPv6 virtual gateway is used as the default route for the in-path interface and the global unicast address is used for the primary interface. Normally, the same default route is used. The in-path interface is configured to use the HSRPv6 virtual gateway link local address for its IPv6 gateway. This link-local address is always on the same network as the SteelHead in-path interface automatically assigned link-local address according to the IPv6 standards.

Figure 13-3. In-path SteelHead configuration using IPv6



To configure an in-path SteelHead using IPv6

- Connect to the SteelHead CLI and enter the following commands:

```
enable
configure terminal
interface primary ipv6 address 2001:aaaa::100 64
ipv6 default-gateway "2001:aaaa::1"
interface inpath0_0 ipv6 address 2001:aaaa::4 64
ipv6 in-path-gateway inpath0_0 fe80::5:73ff:fea0:0
in-path enable
in-path peering-ipv6 enable
```

In RiOS 8.5 to 9.2, using autodiscovery requires an IPv4 address interface:

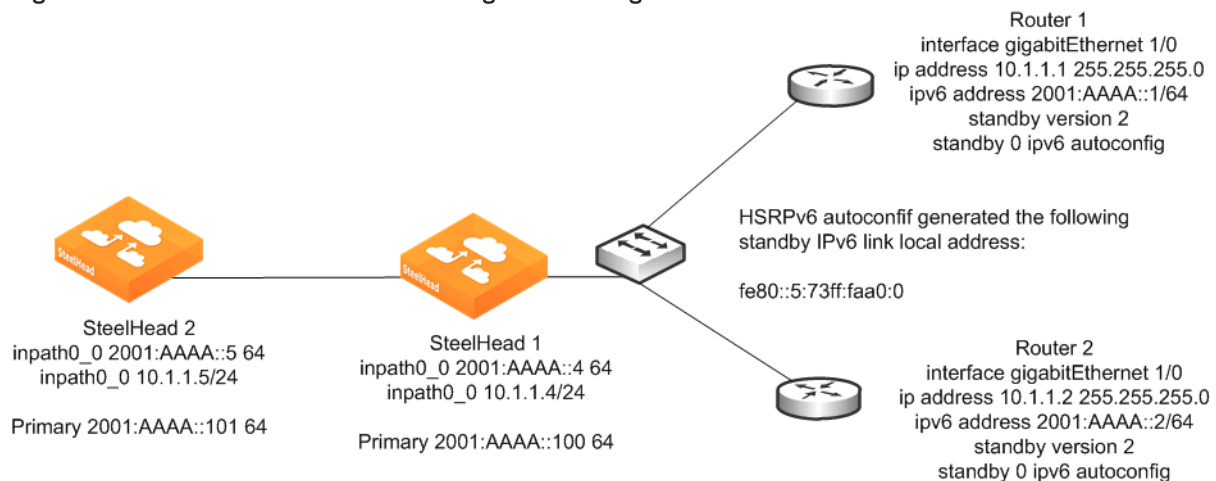
```
inpath0_0 ip address 10.1.1.4 /24
ip in-path-gateway inpath0_0 10.1.1.1
```

Configuring a SteelHead serial cluster IPv6 deployment

Figure 13-4 shows an example deployment with IPv6 addresses in a SteelHead serial cluster deployment. You can deploy SteelHeads in a serial cluster and optimize TCP-over-IPv6 traffic. In dual-stack networks, set the peering rule to match the local peer IPv4 address, because SteelHeads insert the IPv4 address in the autodiscovery probe to identify themselves in the autodiscovery process. In single-stack networks, set the peering rule to match the local peer IPv6 address. Setting the peering rule for serial cluster deployments correctly is important because we do not recommend that you optimize connections between locally connected SteelHeads.

For information about serial cluster deployments, see [“In-path redundancy and clustering examples” on page 213](#).

Figure 13-4. Serial cluster SteelHead configuration using IPv6



To configure a SteelHead serial cluster using IPv6

1. On SteelHead A, connect to the CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 ip address 10.1.1.4 /24
ip in-path-gateway inpath0_0 10.1.1.1
interface inpath0_0 ipv6 address 2001:aaaa::4 64
ipv6 in-path-gateway inpath0_0 fe80::5:73ff:fea0:0
in-path enable
in-path peering auto
in-path peering-ipv6 enable
in-path simplified routing dest-only
in-path peering rule pass peer 10.1.1.5 rulenum end
in-path peering rule pass peer 2001:aaaa::5 rulenum end
write memory
restart
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
enable
configure terminal
interface inpath0_0 ip address 10.1.1.5 /24
ip in-path-gateway inpath0_0 10.1.1.1
interface inpath0_0 ipv6 address 2001:aaaa::5 64
ipv6 in-path-gateway inpath0_0 fe80::5:73ff:fea0:0
in-path enable
in-path simplified routing dest-only
in-path peering auto
in-path peering-ipv6 enable
in-path peering rule pass peer 10.1.1.4 rulenum end
in-path peering rule pass peer 2001:aaaa::4 rulenum end
write memory
restart
```

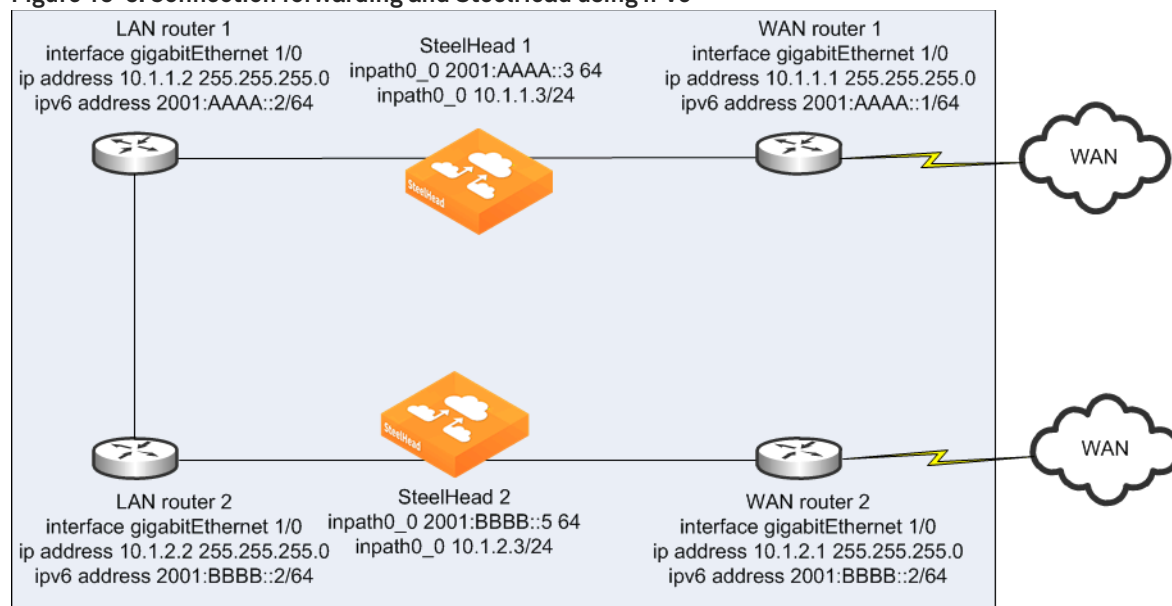
Configuring a connection forwarding and SteelHead IPv6 deployment

Figure 13-5 shows an example deployment of connection forwarding for TCP-over-IPv6 traffic. You can deploy SteelHeads in a parallel topology using connection forwarding to optimize TCP-over-IPv6 traffic. You must use RiOS 8.5 or later and configure your SteelHead to communicate using multi-interface. In addition, each in-path interface requires an IPv4 and an IPv6 address.

Connection forwarding uses the IPv4 addresses (TCP port 7850 connection) and redirects the connection setup packets through GRE encapsulation between the IPv4 addresses. When the connection is established and optimized, asymmetric traffic is redirected through NAT, destined to the peer in-path interface IPv6 address. You can optimize TCP-over-IPv4 and TCP-over-IPv6 traffic concurrently and the NAT entries are resynchronized between peers, even if a peer has its optimization service restarted or is disconnected.

For information about connection forwarding, see [“Connection forwarding” on page 56](#).

Figure 13-5. Connection forwarding and SteelHead using IPv6



To configure connection forwarding and SteelHeads using IPv6

1. On SteelHead A, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.1.1.3 /24
interface inpath0_0 ipv6 address 2001:aaaa::3 64
#--- Set the default gateway for the in-path interface to be the LAN-side Layer-3 switch.
ip in-path-gateway inpath0_0 10.1.1.1
ipv6 in-path-gateway inpath0_0 2001:aaaa::1
#--- Add routes to reach connection forwarding peer over the LAN router connection.
ip in-path route inpath0_0 10.1.2.0 255.255.255.0 10.1.1.2
ipv6 in-path route inpath0_0 2001:bbbb::/64 2001:aaaa::2
#--- Enable enhanced auto discovery.
in-path peering auto
#-- Enable IPv6 enhanced auto-discovery
in-path peering-ipv6 enable
#--- Simplified Routing destination only should be used and is on by default
#--- with new RiOS v6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.1.2.3.
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelheadB main-ip 10.1.2.3
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
#--- Enable and configure the in-path interfaces.
in-path enable
interface inpath0_0 ip address 10.1.2.3 /24
interface inpath0_0 ipv6 address 2001:bbbb::3 64
#--- Set the default gateway for the in-path interface to be the LAN-side Layer-3 switch.
ip in-path-gateway inpath0_0 10.1.2.2
ipv6 in-path-gateway inpath0_0 2001:bbbb::1
#--- Add routes to reach connection forwarding peer over the LAN router connection.
ip in-path route inpath0_0 10.1.1.0 255.255.255.0 10.1.2.2
ipv6 in-path route inpath0_0 2001:aaaa::/64 2001:bbbb::2
#--- Enable enhanced auto discovery
in-path peering auto
#-- Enable IPv6 enhanced auto-discovery
in-path peering-ipv6 enable
#--- Simplified Routing destination only should be used and is on by default
#--- with new RiOS v6.x installs.
in-path simplified routing dest-only
#--- Enable Connection Forwarding to neighbor 10.1.1.3.
#--- Enable multiple-interface forwarding - required for connection forwarding
#--- even if multiple interfaces are not used.
steelhead communication enable
steelhead communication multi-interface enable
steelhead name SteelheadA main-ip 10.1.1.3
```

Configuring a virtual in-path SteelHead IPv6 deployment

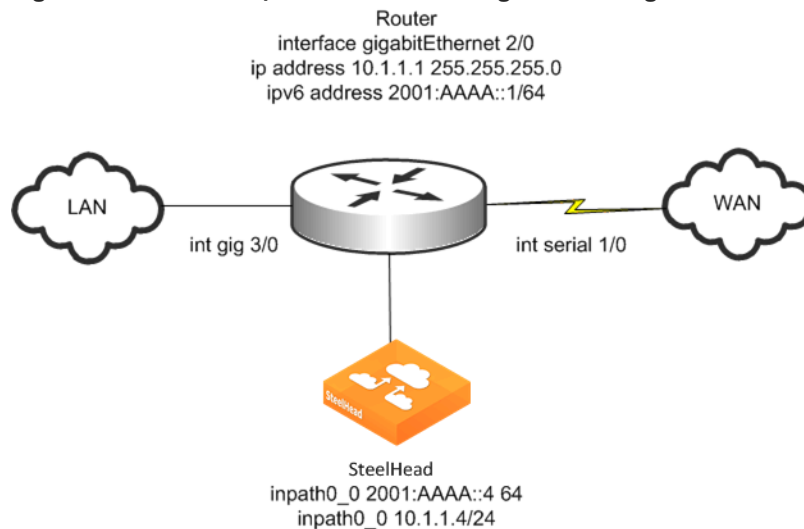
Figure 13-6 shows an example deployment of a virtual in-path SteelHead using IPv6. Virtual in-path support for TCP-over-IPv6 traffic is limited to policy-based routing (PBR) in RiOS 8.5 and later. You configure a SteelHead virtually in-path similarly to an IPv4 PBR deployment.

WCCPv6 is not supported. The Layer-3 device redirecting traffic must support PBR for IPv6 traffic. Cisco provides a list of software in which the PBR for IPv6 feature was introduced (12.2(30)S, 12.2(33)SX14, 12.3(7)T, 12.4, and 12.4(2)T).

Reference: <http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipv6/configuration/12-2sx/ipv6-12-2sx-book.pdf>

For information about PBR, see [“Policy-Based Routing Virtual In-Path Deployments” on page 287](#).

Figure 13-6. Virtual in-path SteelHead configuration using IPv6



To configure a virtual in-path SteelHead using IPv6

1. Connect to the SteelHead CLI and enter the following commands:

```

interface inpath0_0 ipv6 address 2001:aaaa::4 64
ipv6 in-path-gateway inpath0_0 2001:aaaa::1
in-path enable
in-path simplified routing "none"
in-path oop enable
#-- Enable IPv6 enhanced auto-discovery
in-path peering-ipv6 enable

```

You must configure an IPv4 address for autodiscovery of TCP-over-IPv6 traffic if you are using connection forwarding.

2. Configure the Layer-3 device (use Cisco IOS syntax):

```

ipv6 access-list OPTIMIZEv6
 permit tcp any host 2001:DDDD::100
 permit tcp host 2001:DDDD::100 any
route-map OPTIMIZEv6 permit 10
 match ipv6 address OPTIMIZEv6
 set ipv6 next-hop 2001:aaaa::4
interface serial1/0
 ipv6 policy route-map OPTIMIZEv6
interface gigabitEthernet3/0
 ipv6 policy route-map OPTIMIZEv6

```

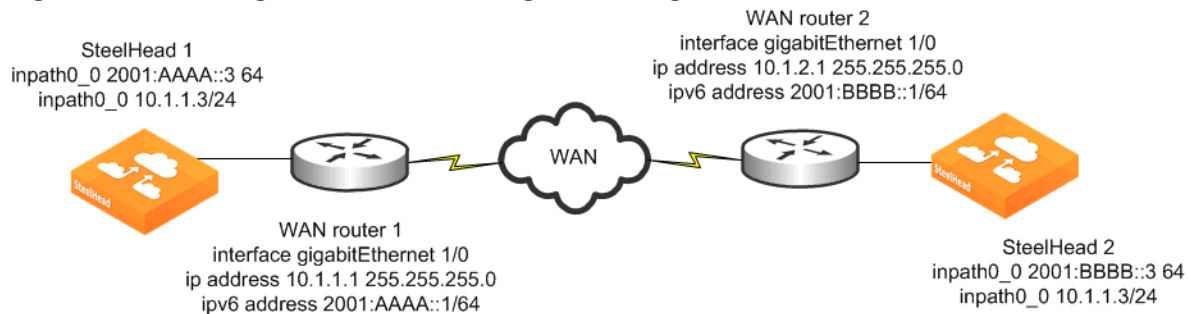
Configuring a fixed-target rule SteelHead IPv6 deployment

As of RiOS 9.5, fixed-target rules are not required for single-stack networks when optimizing TCP-over-IPv6 connections. But fixed-target rules are still helpful in some scenarios such as testing or server-side out-of-path configurations. [Figure 13-7](#) shows an example deployment of a SteelHead using a fixed-target rule in an IPv6 environment. For SteelHeads running RiOS 8.5 to 9.2, you can use fixed-target rules for end-to-end optimization using an IPv6 address when you have SteelHeads in an all-IPv6 network or when peer SteelHeads communicate using their in-path interface IPv6 addresses. RiOS 9.5 and later support IPv6 inner paths for enhanced autodiscovery.

The fixed-target rule operates similarly to IPv4 traffic, by replacing the IPv4 header and sending a directed connection setup packet to the target appliance IPv6 address. A fixed-target rule also works for server-side out-of-path (SSOOP) deployments. In a SSOOP deployment, the outer channel between the server-side SteelHead and server uses the primary interface IPv6 address in the same way it operates for SSOOP on IPv4 traffic.

For information about SSOOP, see [“Out-of-Path Deployments” on page 387](#).

Figure 13-7. Fixed-target rule SteelHead configuration using IPv6



To configure a SteelHead using a fixed-target rule and IPv6

- Connect to the SteelHead CLI and enter the following commands:

```

enable
configure terminal
interface inpath0_0 ip address 10.1.1.3 /24
ip in-path-gateway inpath0_0 10.1.1.1
interface inpath0_0 ipv6 address 2001:aaaa::3 64
ipv6 in-path-gateway inpath0_0 2001:aaaa::1
in-path enable
in-path simplified routing dest-only
in-path peering auto

in-path rule fixed-target target-addr 2001:bbbb::3 target-port 7810 backup-addr :: backup-port
7810 dstaddr all-ipv6 dstport "80" srcaddr all-ipv6 preoptimization "none" optimization
"normal" latency-opt "normal" neural-mode "always" wan-visibility "correct" vlan -1 description
"" auto-kickoff disable rule-enable true rulenum start
  
```

Protocol support

Optimization for TCP-over-IPv6 traffic supports data reduction for any connection. In addition to data reduction, the following protocols have application streamlining support:

- FTP
- SSL
- HTTP
- MAPI
- Encrypted MAPI
- Outlook Anywhere (RPC over HTTP)
- MAPI over HTTP
- SMB

For more information about protocol support, see the *SteelHead Deployment Guide - Protocols* and the *Riverbed Command-Line Interface Reference Manual*.

Verification and troubleshooting

There are three ways to verify and troubleshoot your IPv6 deployments:

- Utility ping
- Traceroute
- Connection report

Make sure that the SteelHead can reach the end system (client or server). You can use a utility ping at the SteelHead CLI and in the SteelHead Management Console. Using the ping6 utility, you must specify the interface IPv6 address instead of the interface name. The following example shows how to send an ICMPv6 echo request using the IPv6 address of 2001:aaaa::10 to reach 2001:aaaa::2:

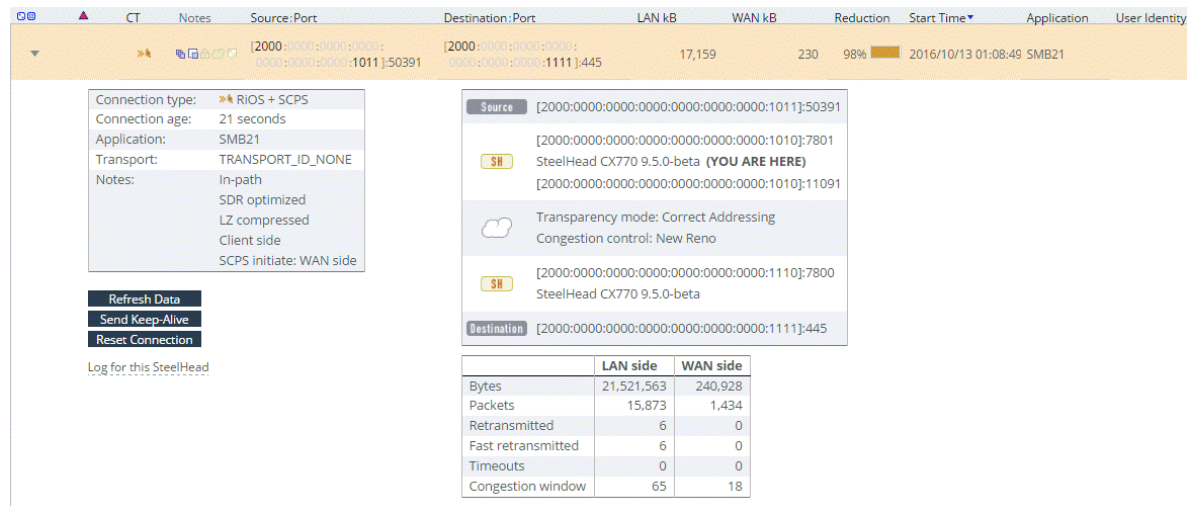
```
VSH # ping6 -I 2001:aaaa::10 2001:aaaa::2
PING 2001:aaaa::2 (2001:aaaa::2) from 2001:aaaa::10 : 56 data bytes
64 bytes from 2001:aaaa::2: icmp_seq=1 ttl=64 time=22.0 ms
64 bytes from 2001:aaaa::2: icmp_seq=2 ttl=64 time=31.1 ms
64 bytes from 2001:aaaa::2: icmp_seq=3 ttl=64 time=20.3 ms
```

Traceroute for IPv6 has also been included for verifying the path from the SteelHead to an IPv6 device or verifying the path between SteelHeads using a fixed-target rule:

```
VSH # traceroute6 -I 2001:aaaa::10 2001:bbbb::2
traceroute to 2001:aaaa::10 (2001:aaaa::10), 30 hops max, 2001 byte packets
 1  2001:aaaa::10 (2001:aaaa::10)  0.328 ms  309.442 ms  309.450 ms
```


Figure 13-8 shows a connection report in which the status of an IPv6 connection is optimized by the SteelHead.

Figure 13-8. Connection report



Packet Mode Optimization

This chapter describes packet mode optimization. Packet mode optimization enables the SteelHead to perform packet-by-packet traffic optimization on various types of IPv4 and IPv6 traffic. This chapter includes the following sections:

- [“Overview of packet mode optimization” on page 323](#)
- [“Comparison with TCP proxy mode optimization” on page 323](#)
- [“Configuring packet mode optimization” on page 324](#)
- [“Design considerations” on page 328](#)
- [“Best practices for packet mode optimization” on page 328](#)

This chapter requires that you be familiar with [“Optimization Techniques and Design Fundamentals” on page 19](#), including data streamlining and fixed-target in-path rules.

For more information about IPv6, see [“IPv6” on page 307](#).

Overview of packet mode optimization

In RiOS 7.0 or later, the packet mode optimization feature can optimize TCP IPv6 and UDP IPv4 traffic. When you use packet mode optimization, the SteelHead applies the same SDR and LZ data streamlining techniques to UDP or TCP IPv6 packets. Packet mode optimization achieves data reduction rates similar to TCP IPv4 traffic.

RiOS 8.5 and later expand support of packet mode optimization to include TCP IPv4 and UDP IPv6 traffic. In addition, connection or flow reporting for packet mode optimization is greatly enhanced. To optimize TCP IPv4 or UDP IPv6, the client-side and server-side SteelHeads must run RiOS 8.5.

Comparison with TCP proxy mode optimization

The bulk of most network traffic is TCP IPv4 traffic, which SteelHeads typically optimize using a TCP proxy architecture. TCP proxy mode optimization separates a TCP connection into three individual connections:

- Client to client-side SteelHead
- SteelHead to SteelHead
- Server-side SteelHead to server

The advantage of TCP proxy mode is that it increases performance. Because the SteelHead acts as a local proxy to the host, the SteelHead is able to perform transport streamlining and application streamlining, which results in increased user performance.

However, sometimes you might want to use the SteelHead optimization to reduce the amount of traffic traversing the WAN. Packet mode optimization provides a simple approach in which the SteelHead looks at a packet, or small group of packets, and performs SDR and LZ on the data payload for data reduction. The host and SteelHead do not create an individual TCP handshake, and the SteelHead reduces payload for packets as the traffic flows through.

The advantage of packet mode optimization is that it is a universal method that applies data streamlining to diverse protocols. The disadvantage is the lack of performance benefits from transport streamlining or application streamlining, because the SteelHead does not proxy or perform intelligent application prediction.

Packet mode optimization is unidirectional—from sender to receiver—which is an effect of not acting as a proxy. For example, take two sites with SteelHeads, Site A and Site B. A device at Site A sends SNMP traps over UDP to a server at Site B, and only SteelHead A is configured to optimize UDP traffic to SteelHead B. If the server at Site B, for some reason, responds over UDP to the device at Site A, this traffic is not optimized unless you have configured SteelHead B to optimize UDP traffic to SteelHead A.

Configuring packet mode optimization

This section describes how to configure packet mode optimization for UDP IPv4 and TCP IPv6 traffic.

Packet mode optimization supports only correct addressing. Packet mode optimization does not support autodiscovery and requires that you configure fixed-target, packet-mode, in-path rules.

For more design details, see [“Design considerations” on page 328](#).

Note: The following example shows UDP traffic.

To configure packet mode optimization for IPv4 traffic

1. From the SteelHead Management Console, choose Optimization > Network Services: General Service Settings and select Enable Packet Mode Optimization; otherwise, use the **packet-mode enable** command.

2. Configure a fixed-target (packet mode optimization) in-path rule on the initiating SteelHead.

To optimize traffic in both directions, you must configure a similar in-path rule on the peer SteelHead.

Figure 14-1 shows an example that creates an in-path rule with the following key settings.

Field	Option	Description
Type	Fixed Target (Packet Mode Optimization)	Specifies the rule type for performing packet mode optimization.
Source Subnet	All-IPv4	Selects all IPv4 addresses. You can choose specific IP addresses or ranges.

Field	Option	Description
Destination Subnet	All-IPv4	Selects all IPv4 addresses. You can choose specific IP addresses or ranges.
Protocol	UDP	Selects optimization of TCP, UDP, or any type of traffic.
Target Appliance IP Address	10.32.9.211	Specifies the remote SteelHead to optimize to. You must specify a target with a fixed-target rule. You can also specify a backup appliance. A backup SteelHead is not specified in this example.
Data Reduction Policy	Normal	Specifies SDR, LZ, or both for data reduction. Normal usage includes both.
Position	End	Determines which position the rule is in the rule list. You can decide this based on your environment.

Figure 14-1. Fixed-target (packet mode optimization) rule for UDP IPv4 traffic

The screenshot shows the 'In-Path Rules' configuration page. At the top, there's a breadcrumb 'Network Services > In-Path Rules' and a help icon. Below this, there are three buttons: 'Add a New In-Path Rule' (selected), 'Remove Selected Rules', and 'Move Selected Rules...'. The main configuration area is titled 'Type: Fixed-Target (Packet Mode Optimization)'. It contains several fields: 'Source Subnet' (All-IPv4), 'Destination Subnet' (All-IPv4), 'VLAN Tag ID' (all), 'Protocol' (UDP), 'Target Appliance IP Address' (10.32.9.211), 'Backup Appliance IP Address' (empty), 'Data Reduction Policy' (Normal), 'Position' (End), 'Description' (Packet Mode - UDP IPv4 Traffic), and 'Enable Rule' (checked). There are also 'Port or Port Label' fields for both source and destination, both set to 'all'. At the bottom, there is an 'Add' button.

3. Click **Add**.

To optimize IPv6 traffic, you must enable packet mode optimization and create a fixed-target packet-mode rule similar to IPv4 traffic, but you must enable IPv6 (IPv6 is enabled by default) and configure an IPv6 setting on the SteelHead in-path interfaces.

Note: This example shows TCP traffic.

To configure packet mode optimization for IPv6 traffic

1. From the SteelHead Management Console, choose Optimization > Network Services: General Service Settings, otherwise use the **packet-mode enable** command.
2. Configure a fixed-target (packet mode optimization) in-path rule on the initiating SteelHead.
To optimize TCP traffic in both directions, you must configure a similar in-path rule on the peer SteelHead.

Figure 14-2 shows an example that creates a rule with the following key settings.

Field	Option	Description
Type	Fixed-target (packet mode optimization)	Specifies the rule type for performing packet mode optimization.
Source Subnet	All-IPv6	Selects all IPv6 addresses. You can choose specific IP addresses or ranges.
Destination Subnet	All-IPv6	Selects all IPv6 addresses. You can choose specific IP addresses or ranges.
Protocol	TCP	Selects optimization of TCP, UDP, or any type of traffic.
Target Appliance IP Address	10.32.9.211	Specifies the remote SteelHead to optimize to. You must specify a target with a fixed-target rule. You can also specify a backup appliance. A backup SteelHead is not specified in this example.
Data Reduction Policy	Normal	Specifies SDR, LZ, or both for data reduction. Normal usage includes both.
Position	End	Determines which position the rule is in the rule list. You can decide this based on your environment.

Figure 14-2. Fixed-target (packet mode optimization) rule for TCP IPv6 traffic

The screenshot shows the 'In-Path Rules' configuration page. At the top, there's a breadcrumb 'Network Services > In-Path Rules' and a help icon. Below the header, there are three action buttons: 'Add a New In-Path Rule', 'Remove Selected Rules', and 'Move Selected Rules...'. The main form is titled 'Fixed-Target (Packet Mode Optimization)' and contains the following fields:

- Type:** Fixed-Target (Packet Mode Optimization) (dropdown)
- Source Subnet:** All-IPv6 (text input)
- Destination Subnet:** All-IPv6 (text input)
- Port or Port Label:** all (text input, next to both Source and Destination Subnet)
- VLAN Tag ID:** all (text input)
- Protocol:** TCP (dropdown)
- Target Appliance IP Address:** 10.32.9.211 (text input)
- Port:** 7800 (text input, next to Target Appliance IP Address)
- Backup Appliance IP Address:** (empty text input)
- Port:** 7810 (text input, next to Backup Appliance IP Address)
- Data Reduction Policy:** Normal (dropdown)
- Position:** End (dropdown)
- Description:** Packet Mode - TCP IPv6 Traffic (text input)
- Enable Rule:** ☒ (checkbox)

At the bottom left of the form is an 'Add' button.

3. Click **Add**.
4. Connect to the Riverbed CLI. You can also use the SteelHead Management Console.
5. Enable IPv6.
IPv6 is enabled by default, unless you are using a RiOS version earlier than 8.0.
6. Configure an IPv6 address on a SteelHead in-path interface: **interface inpathX_Y ipv6 address <address> <mask-length>**.
7. Add IPv6 routes: **ipv6 in-path route inpathX_Y <prefix> <prefix-length> <nexthop-v6-address>**.
For example, `ipv6 in-path route inpath0_0 0::0/0 ba5e:ba11:bab3:f005:ba11:bab3:dead:bea7` sets a default IPv6 gateway.

Design considerations

Packet mode optimization does not support several network integration features available with TCP proxy optimization. The following limitations apply to packet-mode optimized traffic.

Packet mode optimization in RiOS 8.5 and later supports:

- only fixed-target (packet mode optimization). autodiscovery is not supported.
- only correct addressing. Full transparency and port transparency are not supported. Because full transparency is not supported, VLAN transparency is also not supported.
- physically and virtually in-path deployments. The exception is WCCP for IPv6 traffic, because the SteelHead WCCPv2 implementation currently does not support IPv6.
- data reduction for all IPv4 and IPv6 traffic.
- connection reporting of packet-mode optimized traffic flows.

Packet mode optimization does not support:

- RSP data flow rules.
- NetFlow export of the packet-mode optimized traffic.
- connection forwarding. This limitation does not imply that parallel SteelHead deployments do not work. As each flow is optimized unidirectionally, asymmetry does not have the same relevance on packet-mode optimized traffic.
- SteelHead Interceptor deployments.
- server-side out-of-path configuration.
- QoS shaping, enforcement, or marking. All packet-mode optimized traffic matches the default QoS default rule and class. The exception is UDP IPv4 traffic, which you can place into an MX-TCP class.

For more information about MX-TCP class, see [“MX-TCP” on page 123](#).

Best practices for packet mode optimization

We recommend that you use the following best practice guidelines when using packet mode optimization:

- **Target specific applications and servers for packet mode optimization** - Similar to the considerations when performing optimization on TCP IPv4 traffic, certain traffic types do not lend themselves well to data reduction. For example, encrypted or compressed traffic does not receive significant data reduction. We recommend that you use rules targeting specific networks or ports instead of using broad All-IP targets. Traffic types such as TFTP or UDP-based replication traffic are example target applications.
- **Do not optimize voice or video bearer traffic** - While VoIP is one of the more pervasive applications over UDP, VoIP and video RTP traffic are already compressed using specialized codecs. If you attempt to perform further data reduction with SDR or LZ, it is ineffective and can add latency resulting in jitter.
- **Use TCP proxy mode optimization** - For application latency improvements for TCP IPv4 or IPv6 traffic, you can use the typical TCP proxy mode optimization. RiOS 8.5 and later include TCP proxy mode optimization for IPv6 traffic. For information about IPv6 traffic, see [“IPv6” on page 307](#).

Satellite Optimization

This chapter describes how to configure transport optimization for satellite networks. When you tune transport settings for satellite networks, you can achieve improved performance and interoperability with other TCP performance-enhancing proxies (TCP-PEP). Tuning transport settings for satellite networks falls into two primary categories:

- **Space communications protocol specification (SCPS) discovery** - Provides interoperations with third-party TCP-PEP devices.
- **Transport settings** - Provides flexibility for TCP optimization algorithms.

This chapter includes the following sections:

- [“Overview of satellite networks” on page 329](#)
- [“Overview of SCPS” on page 331](#)
- [“TCP optimization for satellite environments” on page 333](#)
- [“Licensing SCPS on a SteelHead” on page 340](#)
- [“Configuring satellite optimization features” on page 340](#)
- [“Verification and troubleshooting” on page 350](#)

Note: If you are using a release prior to RiOS 7.0, SCPS was available through an RSP package and was not native to RiOS. The RSP SCPS package interoperates with SCPS native to RiOS 7.0. RSP SCPS package licenses are not valid for use as native RiOS SCPS licenses. Contact Riverbed Support or your sales team for assistance in converting RSP SCPS package licenses to native RiOS SCPS licenses.

Note: In RiOS 7.0.1 or later, RSP is replaced with VSP. VSP comes preinstalled in the SteelHead EX. For more information about VSP, see the *SteelHead User Guide* for the SteelHead EX. Your existing RSP packages work on VSP.

Overview of satellite networks

This section introduces satellite networks. This section includes the following topics:

- [“Impact of latency” on page 330](#)
- [“Impact of loss” on page 330](#)
- [“Satellite transport options” on page 331](#)

You can use satellite networks for WAN connectivity for people in remote locations or when users are in a temporary or mobile environment (for example, a ship or an oil rig). Satellite networks have several characteristics that differ from terrestrial networks. The most prevalent differences include the following:

- High latency
- Dynamic bandwidth
- Asymmetric bandwidth capability
- Loss due to signal to noise issues

These characteristics can create challenges for traditional TCP algorithms for various reasons. This chapter helps you to understand and address these characteristics.

Impact of latency

When it comes to TCP, latency is the enemy. The higher the latency, the longer it takes ordinary TCP to ramp up and use the available capacity. In satellite networks, latency is typically 10 to 100 times higher than in terrestrial links. Latency in a single-hop satellite network usually varies from 550 ms to 800 ms. Dual-hop satellite network latency commonly ranges from 1100 ms to 1600 ms. With Inmarsat BGAN networks (broadband satellite communications), it is not unusual to observe peak latencies of 2000 ms and higher when congestion is present.

Even though the SteelHead has TCP optimization enabled by default, tuning transport settings for a target satellite environment results in even better performance.

Impact of loss

Satellite communications are known for having less than perfect packet delivery. Loss can wreak havoc on TCP throughput. If the loss is high enough, it can cause sessions to time-out. Loss can happen for several reasons on satellite networks. You must determine the primary cause of loss in your network so that you can tune your TCP optimization accordingly.

The two primary reasons for loss in satellite networks are congestion and poor signal quality. Do not ignore the possibility of a simple bad cable or speed and duplex mismatch when troubleshooting. Congestion-based loss is normal and expected when a link or path is saturated. Congestion-based loss triggers a TCP stacks congestion avoidance algorithm so that individual TCP sessions can adapt intelligently. Poor signal quality increases the bit error rate (BER), which is not due to congestion, but rather because of satellite communications. BER is independent of load on the circuit.

Many different events can trigger increased BER. A few of the most common are obstructions (buildings, bridges, and so on), a misaligned satellite dish, and weather. Weather events are temporary and go away in time. With obstructions or poor angles, the loss can be more persistent. Most satellite modems have forward error correction (FEC) built-in for free, and can overcome an increased BER in many situations.

In some cases, the BER loss is too severe for the satellite modem to overcome with FEC. In these situations, you might detect consistent TCP loss. Ideally, you should solve the root cause of the increased BER, but as a network administrator, this solution might be out of your control. In this situation, you can leverage a TCP Stack optimized for a high error environment.

Satellite transport options

You can choose from many satellite transport options. Their performance characteristics vary widely. A reasonable estimation is the more terminals sharing a segment, the higher the possibility for variable latency and loss. This section provides a brief summary of popular satellite solutions, along with general assumptions of loss and latency expectations. You should verify these details for your network because they might be different than those in the following table.

Satellite transport option	Variable or fixed throughput	Latency characteristics	Loss characteristics
Inmarsat	Fixed	Predictable	
Inmarsat BGAN	Highly variable—shared broadband	Highly variable	Highly variable
Single channel per carrier (SCPC)	Fixed	Predictable	
TDMA	Variable	Variable due to wait time for frequency slot	
FDMA	Variable	Variable due to wait time for frequency slot	
DVB-RCS/MF-TDMA	Variable	Variable	Variable: certain codec modulations are adaptive and adjust to deal intelligently with poor signal quality.

Overview of SCPS

This section describes SCPS. It does not cover the implementation of SCPS on the SteelHead. This section includes the following topics:

- [“SCPS benefits” on page 332](#)
- [“Common uses for SCPS” on page 332](#)
- [“SCPS and SteelHeads” on page 333](#)

For information about how to implement SCPS on the SteelHead, see [“TCP optimization for satellite environments” on page 333](#).

SCPS is a group of several protocol specifications developed by the consultative committee for space data systems (CCSDS) to address the limitations of communications in space. In the WAN optimization market, SCPS refers to the transport protocol specification, otherwise known as SCPS-TP. SCPS-TP is the most widely supported SCPS protocol. In the WAN optimization market, SCPS-TP is commonly called SCPS. The use of SCPS in SteelHeads is specifically referencing SCPS-TP. Definitions for all SCPS protocols are as follows:

- **SCPS-FP (file transfer protocol)** - A set of extensions to FTP to make it more bit efficient and to add advanced features such as record update within a file and integrity checking on file transfers. This protocol is optional.

- **SCPS-TP (transport protocol)** - A set of transmission control protocol (TCP) options, such as selective negative acknowledgment (SNACK), selective acknowledgment (SACK), modified slow start algorithms, modified congestion avoidance algorithms, and Windows scaling. Additionally, SCPS-TP includes sender-side modifications to enhance the TCP performance in the stressed environments, such as long delays, high bit error rates, and significant asymmetries. For flow negotiation, SCPS uses TCP options that are registered with the Internet Assigned Numbers Authority (IANA). As a result, the SCPS-TP stack is compatible with other recognized TCP implementations.
- **SCPS-SP (security protocol)** - Based on security protocol 3 (SP3) and network layer security protocol (NLSP), with reduced protocol overhead of header. SCPS-SP also provides integrity, confidentiality, authentication, and access control for the data transmitted over the network. SCPS-SP is an optional protocol and is comparable to Internet Protocol Security (IPSec).
- **SCPS-NP (network protocol)** - A bit-efficient network protocol that is analogous to IP. However, it is not compatible with IP. The protocol is designed for use in space systems. The protocol supports static or dynamic routing, precedence, and multiple routing options. This protocol is optional.

SCPS benefits

SCPS provides improved TCP performance in high-latency and high-loss environments, such as satellite networks. As a specification, SCPS enables third-party WAN optimization solutions that support SCPS, to provide TCP acceleration in a heterogeneous WAN optimization environment.

SCPS is designed for use in dual-ended proxy (symmetric) scenarios. However, some SCPS devices also provide single-ended optimization which enables a device to provide sender-side benefits even when there is no optimization appliance at the far end. However, not all SCPS solutions support single-ended proxy implementations.

Common uses for SCPS

The three primary markets for SCPS are as follows:

- Space agency space networks
- Commercial satellite networks
- Private government satellite networks

SCPS is a common request in large multiagency government satellite architectures in which a lowest common denominator approach to TCP acceleration is desirable. This enables the hub organization to provide basic TCP optimization to multiple agencies connecting into their teleports, without requiring a specific vendor solution.

SCPS and SteelHeads

SCPS is available in RiOS 7.0 or later. An SCPS license enables SteelHeads running RiOS 7.0 to negotiate SCPS optimization with another SteelHead or a third-party WAN optimizer or TCP-PEP. The SCPS license also enables the SCPS per connection and SCPS error-tolerant transport setting options. These two TCP stacks are tuned specifically for satellite networks. SteelHeads running RiOS 8.5 or later include a rate pacing mechanism to further tune the TCP. The mechanism can also negotiate compression with a third-party WAN optimizer or TCP-PEP.

You can configure transport optimization used by the SteelHead separately from SCPS negotiation. This separation provides extensive flexibility for a broad range of environments.

A SteelHead with SCPS, optimizing traffic to another SteelHead, uses the configured TCP stack and still performs standard RiOS optimization functions such as data reduction and application streamlining. The same SteelHead can negotiate SCPS with third-party TCP-PEP devices. This provides organizations with an approach for supporting third-party SCPS interoperability, while at the same time maximizing performance and productivity within their architectures with RiOS optimization.

For more information about using SCPS in SteelHeads, see [“TCP optimization for satellite environments” on page 333](#) and [“Licensing SCPS on a SteelHead” on page 340](#).

Note: In RiOS 7.0 or later, you must license SCPS and restart the optimization service to enable SCPS negotiation service. This is irrelevant of which transport optimization method you select (for example, standard TCP, high-speed TCP, BW estimation, per connection, or error tolerant).

TCP optimization for satellite environments

This section introduces TCP optimization for satellite environments. This section includes the following topics:

- [“SCPS discovery” on page 334](#)
- [“Transport optimization for satellite environments” on page 335](#)
- [“Configuring automatic detect TCP optimization” on page 338](#)
- [“Integrating the SteelHead with existing satellite modem TCP acceleration” on page 339](#)

TCP optimization enhancements in RiOS 7.0 or later provide a robust and easy-to-use set of TCP optimization options for satellite networks. These mechanisms are specifically tuned for the performance challenges of satellite environments. The capabilities in RiOS 7.0 or later fall into two primary categories:

- SCPS discovery mechanisms
- Transport optimization mechanisms

To use SCPS discovery or the SCPS transport options, you must install an SCPS license on the SteelHead. All non-SCPS options described in this chapter are included with the basic RiOS licenses, which include bandwidth estimation transport optimization and error recovery mode.

In RiOS 7.0 or later, the SCPS discovery advertises support for SCPS negotiation. SCPS discovery does not determine the TCP stack used during optimization. The transport optimization setting you configure on the SteelHead determines the TCP stack used during optimization. This is different from the RiOS 6.5 implementation of SCPS in RSP, which always uses SCPS transport optimization. The new approach in RiOS 7.0 provides more flexibility for complex dynamic environments and is also easier to use.

The separation of SCPS discovery and the transport optimization setting is a subtle point, but it is important to understand when you implement SteelHeads within a satellite environment with SCPS requirements. The remaining subsections provide more information about SCPS discovery and transport optimization settings.

SCPS discovery

This section describes SCPS discovery, also known as SCPS negotiation. SCPS discovery uses TCP options for discovery. SCPS uses a 4-byte TCP option with the decimal number 20 (hexadecimal 0x14). SCPS header also has extended capabilities that use a 10-byte TCP option with the decimal number 20 (hexadecimal 0x14). When you use the extended SCPS header, it immediately follows the mandatory 4-byte SCPS header. SteelHeads licensed for SCPS support dual-ended proxy connections to SteelHeads and dual-ended proxy connections to third-party SCPS TCP-PEP devices. Keep in mind that the congestion-avoidance algorithm determines how TCP is optimized; SCPS discovery is only used for negotiating SCPS interoperability.

Client-side SteelHeads initially mark the SYN packet with the RiOS discovery TCP option (decimal 76 or 78). If no RiOS discovery response is detected in the initial SYN/ACK, then an RST packet is sent and a new SYN is sent with an SCPS TCP option. The new SYN uses the same client and server ports as the original SYN but has a different TCP sequence number. If an SCPS TCP option is returned in the SYN/ACK from the server-side peer, SCPS optimization is established.

If no SCPS TCP option is returned from the server side, a client-side SteelHead using SCPS does not optimize the flow and passes the traffic through. If you do not want SCPS optimization for specific traffic, you can use the single-ended connection rule table to exclude it. The following table summarizes the discovery process used in the various device scenarios.

Device location/type	Server - SteelHead without SCPS	Server - SteelHead with SCPS	Server - SCPS TCP-PEP	Server - no device
Client - SteelHead without SCPS	RiOS	RiOS	No acceleration	No acceleration
Client - SteelHead with SCPS	RiOS	RiOS	SCPS	No acceleration
Client - SCPS TCP-PEP	No acceleration	SCPS	SCPS	No acceleration

Transport optimization for satellite environments

RiOS 8.5 and later have a broad set of transport options that you can use to adapt the TCP optimization for specific segments of your organization and respective performance characteristics. This section specifically addresses transport optimization in satellite networks. This section includes the following topics:

- [“Bandwidth estimation” on page 335](#)
- [“Configuring error recovery” on page 336](#)
- [“SCPS per connection” on page 336](#)
- [“SCPS error tolerance” on page 336](#)
- [“Rate pacing” on page 337](#)
- [“SCPS single-ended rules” on page 337](#)
- [“SCPS compression” on page 338](#)

RiOS 8.5 and later support four specific transport settings for satellite networks. These settings include three TCP stacks and an error recovery mechanism. The transport settings are included in the following:

- Bandwidth estimation and error recovery are available in the base license of RiOS 6.5 or later.
- SCPS per connection and SCPS error tolerance requires you to install an SCPS license in your SteelHead.
- Rate pacing requires you to configure MX-TCP through advanced QoS.

For information about MX-TCP and QoS, see [“MX-TCP” on page 123](#).

- SCPS compression.

SteelHead transport settings are communicated among peers through the out-of-band connection between SteelHead peers.

Note: All RiOS TCP stacks support selective acknowledgments (SACK) for efficient retransmission of packets.

Bandwidth estimation

The bandwidth estimation transport setting uses an intelligent bandwidth estimation algorithm along with a modified slow-start algorithm to optimize performance in long lossy networks. These networks typically include satellite and other wireless environments, such as cellular networks, longer microwave, or Wi-Max networks. Bandwidth estimation is a sender-side modification of TCP and is compatible with the other TCP stacks in RiOS. The intelligent bandwidth estimation is based on analysis of both ACKs and latency measurements. The modified slow-start mechanism enables a flow to ramp up faster in high latency environments than traditional TCP. The intelligent bandwidth estimation algorithm allows it to learn effective rates for use during modified slow start and also to differentiate BER loss from congestion-derived loss and deal with them accordingly. Bandwidth estimation has good fairness and friendliness qualities toward other traffic along the path.

Configuring error recovery

To handle satellite transmission errors and intermittent connectivity, RiOS 6.5 or later includes the lossy link-error recovery mechanism. In a lossy environment, you can enable error recovery to modify the congestion avoidance algorithm of the underlying TCP stack. This modification causes the underlying RiOS TCP stack congestion avoidance algorithm to be less responsive to retransmissions: for example, duplicate ACKs. This can be quite effective in situations with BER loss. However, you might not want it in situations where loss is congestion-based. By making TCP less responsive to loss, you might cause congestion to worsen. Due to this trade-off, use caution when you enable error recovery, particularly in situations with coexisting traffic along a path. We recommend that you do not enable error recovery on terrestrial channels.

You can enable error recovery only from the CLI. Use the **tcp err-recovery loss-recovery mode always** command on the client-side (remote) SteelHead.

To configure satellite transmission error recovery

- Connect to the client-side (remote) SteelHead and enter the following command:

```
tcp err-recovery loss-recovery mode always
restart
```

This enables lossy link error recovery on all traffic sent by this remote SteelHead. This configuration is communicated to a peer SteelHead so that the peer SteelHead can use error recovery when it sends traffic to the remote SteelHead. By default, error recovery is disabled.

To disable lossy link error recovery, use **tcp err-recovery loss-recovery mode disable**.

SCPS per connection

The SCPS per connection transport setting uses a modified slow-start algorithm and a modified congestion-avoidance approach. This approach enables SCPS per connection to ramp up flows faster in high-latency environments, and handle lossy scenarios, while remaining reasonably fair and friendly to other traffic. SCPS per connection does a very good job of efficiently filling up satellite links of all sizes. SCPS per connection is a high performance option for satellite networks.

SCPS error tolerance

The SCPS error tolerance transport setting uses a modified slow-start algorithm and a modified congestion avoidance approach. SCPS error tolerance requires many more retransmitted packets to trigger the congestion avoidance algorithm than the SCPS per connection. The assumption is that the environment in which you use SCPS error tolerance has a high BER and most retransmissions are due to poor signal quality instead of congestion. This setting enables SCPS error tolerance to efficiently maximize performance in high loss environments, without incurring the additional per-packet overhead of a FEC algorithm at the transport layer. SCPS error tolerance is a high performance option for lossy satellite networks.

Do not use SCPS error tolerance in clean networks, because it can be quite aggressive and compete unfairly with other traffic.

Rate pacing

The rate pacing setting uses a modified slow start algorithm to intelligently switch to congestion avoidance without incurring the penalty of the first loss to exit TCP slow start. Additionally, the SCPS rate pacing mechanism maintains a steady data rate in congestion avoidance while efficiently adapting to congestion in a shared network.

Rate pacing is a marked improvement over using SCPS per connection, SCPS error tolerant, or MX-TCP. SCPS per connection and error tolerant switch from slow start to congestion avoidance on the first loss event. MX-TCP does not adapt to congestion in a shared network and could cause congestion collapse in which senders continually retransmit data. The combination of an efficient mechanism to switch from slow start without incurring a loss event and the use of a steady data rate in the congestion avoidance phase enables the SteelHead to fill a satellite link while avoiding some loss from buffer overruns and buffer delay for latency-sensitive TCP traffic.

Rate pacing is an ideal setting for many networks. The combination of a tuned TCP stack for satellite environments and MX-TCP to control the rate eases the burden of calculating buffer sizes and making adjustments across the infrastructure.

Rate pacing requires you to configure MX-TCP and as a result does not support IPv6.

For information about how to configure rate pacing, see [“Configuring rate pacing” on page 344](#).

SCPS single-ended rules

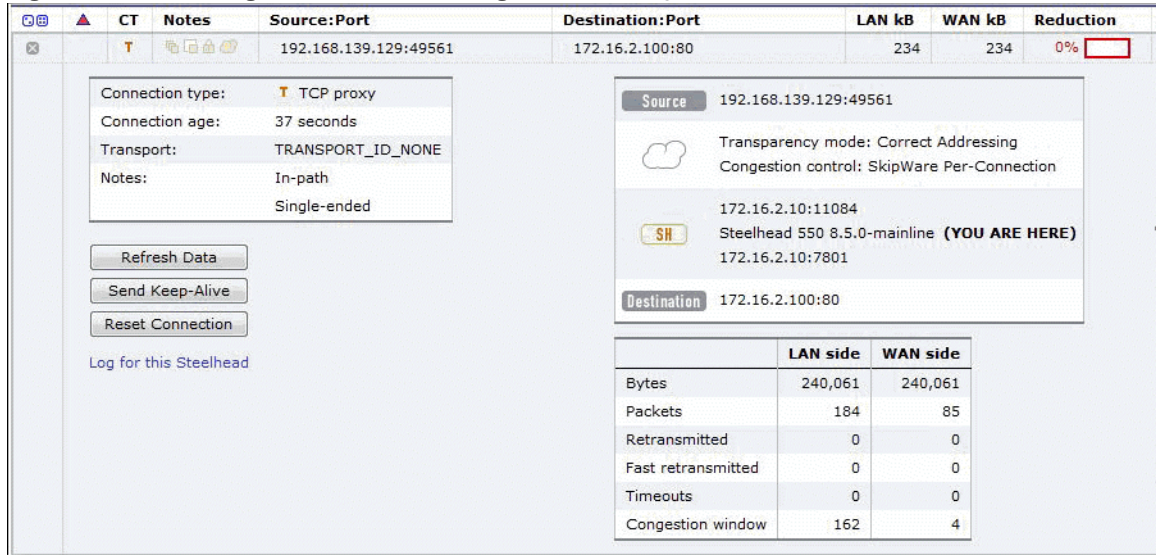
SCPS single-ended rules provide an option to use a TCP stack that is tuned for satellite environments when communicating with a third-party WAN optimizer or TCP-PEP. SCPS single-ended rules is used most commonly for SCPS integration.

In RiOS 8.5 and later, SCPS single-ended rules are enhanced to provide additional options suited for more complex environments. One key addition is the ability to select a transport option (TCP stack) on a per-rule basis, allowing you to use the most appropriate TCP stack for a given environment. You can also enable the rate pacing functionality on a per-rule basis to further adjust the transport optimization to the environment. You can support IPv6 hosts. The default rule is to pass through connections for all IPv4 and IPv6 hosts.

You can use SCPS single-ended rules to take advantage of a single-ended proxy. This functionality can help where there is no WAN optimizer or TCP-PEP on the other side of the connection. Because this feature is a sender-side modification, it provides the most benefit when used at the side of the connection sending the majority of the data.

Figure 15-1 shows an example of a single-ended proxy connection. In this example, the server-side SteelHead uses a single-ended rule to proxy the connection using SCPS per connection as the TCP stack.

Figure 15-1. SCPS single-ended rule as a single-ended proxy



For information about configuring single-ended rules, see [“Configuring single-ended rules” on page 347](#).

SCPS compression

SCPS compression is available in RiOS 8.5 and later. SCPS compression is designed for scenarios in which the SteelHead is interoperating with a third-party WAN optimizer or TCP-PEP using SCPS single-ended rules. When you enable SCPS compression, the SteelHead negotiates SCPS compression functionality with the third-party device. Compression is performed across the TCP header and payload. The TCP header is compressed according to the SCPS-TP standard, and the payload uses LZ-based compression on a packet-by-packet basis. SCPS-compressed IP packets have a next-protocol value of 105, which can require changes to intervening security devices.

Configuring automatic detect TCP optimization

Automatic detect TCP optimization is in RiOS 7.0 or later. Automatic detect TCP optimization enables a SteelHead to detect the transport setting (in other words, the TCP stack) advertised by a peer SteelHead and implement the respective transport setting for the applicable TCP flows to the peer SteelHead. TCP optimization is useful in complex topologies where several different types of networks terminate into a hub or server-side SteelHead.

Use automatic detect TCP optimization on the hub SteelHead to enable an organization that uses the best transport optimization for the respective network segment. You enable the TCP optimization you want on remote SteelHeads. Automatic detect TCP optimization is advertised to a peer via the out-of-band connection between the appliances. The transport setting is communicated by the client-side or server-side SteelHead. However, at least one SteelHead must support automatic detect TCP optimization; otherwise, each SteelHead uses its defined transport setting.

Use automatic detect TCP optimization when you have lower-speed terrestrial connected remote sites, high-speed data centers, and satellite sites all terminating into a SteelHead at an aggregation point: for example, a data center. When you enable automatic detect TCP optimization on the aggregation point SteelHead, and the desired transport settings on the remote SteelHeads, each network segment uses the appropriate transport setting.

To configure automatic detect TCP optimization

- Connect to the CLI and enter the following commands:

```
tcp cong-ctrl mode auto
write memory
```

You can also choose Optimization > Network Services: Transport Settings, select Auto-Detect in the TCP Optimization box, and click **Apply** and **Save**.

Integrating the SteelHead with existing satellite modem TCP acceleration

In some networks, the satellite modems might have built-in TCP acceleration and possibly even LZ compression. Typically, these solutions do not perform well as advanced WAN optimizers such as a SteelHead. However, when you deploy a SteelHead into an environment with TCP acceleration in the satellite modems, the challenge is learning if your modems have TCP acceleration enabled and then determining the most appropriate integration method.

You might not be aware if the modems you have are providing acceleration service until you have deployed the SteelHeads, and autodiscovery does not work due to the satellite modem stripping the RiOS TCP option options used for autodiscovery.

For information about how to analyze for stripped probes, see [“Verification and troubleshooting” on page 350](#).

You can choose from several options to integrate SteelHeads with satellite modems conducting TCP acceleration. The options include:

- Disable TCP acceleration in the satellite modems.
- Enable TCP option forwarding in the satellite modem for the appropriate TCP options:
 - SCPS = TCP option 20
 - Correct Addressing = TCP option 76
 - Port Transparency = TCP option 76 and 78
 - Full Transparency = TCP option 76 and 78
- Use fixed-target rules in the SteelHeads instead of autodiscovery. For more information about fixed-target in-path rules, see the *SteelHead User Guide*.

For information about how to disable TCP acceleration or configure TCP option forwarding, contact the satellite modem vendor technical support.

Licensing SCPS on a SteelHead

To determine if your SteelHead has a valid SCPS license, choose Administration > Maintenance: Licenses and look in the description column for SCPS. You can also use the **show licenses** command. The format of the actual license is LK1-SH55SCPS-XXXX-XXXX-1-XXXX-XXXX-XXXX.

When you order an SCPS license, it is delivered by email from riverbed-license@riverbed.com. You need a license for the data center and branch office SteelHead.

To install an SCPS license

1. Choose Administration > Maintenance: Licenses.
2. Click **Add License**.
3. Paste the license into the text field.
4. Click **Apply**.

Or, connect to the CLI and enter the following commands:

```
enable
configure terminal
license install <license-key>
write memory
restart
```

You must perform an optimization service restart to complete the installation of the SCPS license.

Configuring satellite optimization features

This section describes the following tasks:

- [“Configuring transport optimization” on page 340](#)
- [“Configuring rate pacing” on page 344](#)
- [“Configuring single-ended connection rule table settings” on page 345](#)
- [“Configuring single-ended rules” on page 347](#)

Configuring transport optimization

To properly configure transport settings for the target environment, you must understand its characteristics. This section describes how to configure, monitor, and troubleshoot the transport settings in RiOS 7.0 or later.

To capture your performance characteristics

1. Connect to the SteelHead CLI, using an account with administration rights.
2. Enter the enable mode and then configure terminal mode:

```
enable
configure terminal
```

3. If your environment does not support path MTU discovery, use the **ping** command to measure the maximum transmission unit (MTU) by pinging a remote host.

Start with a full-size packet and decrease the size of the packet in small increments until the ping successfully reaches the target host.

Write the MTU here: _____

Write the round trip time here: _____

If you are deploying your SteelHead through WCCP, you might need to take into account the additional GRE overhead of WCCP. Use the following **ping** command for measuring maximum packet size along a specified path:

```
ping -I inpath0_0 -s <Bytes> <target host>
```

4. Use the following command with a full-size packet for a count of 1000 or more packets:

```
ping -I inpath0_0 -c 1000 -s <your MTU> <target host>
```

Write the percentage of loss during this test here: _____

5. Configure the SteelHead WAN buffers for the target network.

Using the data you have collected, calculate two times bandwidth delay project (BDP) for your satellite network using the following formula or table. For satellite networks that vary in capacity, use the maximum potential speed the network can achieve. If your satellite link speeds differ in either direction, you might have different size *send* and *receive* buffer sizes. If this is the case, your send buffer on the transmitting side should match your receive buffer on the receiving side.

SteelHead WAN Buffers = ((Your RTT in milliseconds * 0.001) * (Your Circuit Speed in bps/8) * 2)

For example, ((600 ms * 0.001) * (5,000,000bps/8) * 2) = 750,000 byte WAN buffers.

Use the following table as a quick reference to help estimate appropriate SteelHead WAN buffers.

Link speed (bps)	256,000	768,000	1,544,000	6,000,000	10,000,000	20,000,000	45,000,000
RTT (ms)							
600	38,400	115,200	231,600	900,000	1,500,000	3,000,000	6,750,000
700	44,800	134,400	270,200	1,050,000	1,750,000	3,500,000	7,875,000
800	51,200	153,600	308,800	1,200,000	2,000,000	4,000,000	9,000,000
900	57,600	172,800	347,400	1,350,000	2,250,000	4,500,000	10,125,000
1000	64,000	192,000	386,000	1,500,000	2,500,000	5,000,000	11,250,000
1100	70,400	211,200	424,600	1,650,000	2,750,000	5,500,000	12,375,000
1200	76,800	230,400	463,200	1,800,000	3,000,000	6,000,000	13,500,000
1300	83,200	249,600	501,800	1,950,000	3,250,000	6,500,000	14,625,000
1400	89,600	268,800	540,000	2,100,000	3,500,000	7,000,000	15,750,000
1500	96,000	288,000	579,000	2,250,000	3,742,800	7,500,000	16,875,000
1600	102,400	307,200	617,600	2,400,000	4,000,000	8,000,000	18,000,000

Link speed (bps)	256,000	768,000	1,544,000	6,000,000	10,000,000	20,000,000	45,000,000
RTT (ms)							
1700	108,800	326,400	656,200	2,550,000	4,250,000	8,500,000	19,125,000
1800	115,200	345,600	694,800	2,700,000	4,500,000	9,000,000	20,250,000
1900	121,600	364,800	733,400	2,850,000	4,750,000	9,500,000	21,375,000
2000	128,000	384,000	772,000	3,000,000	5,000,000	10,000,000	22,500,000

Write the WAN buffer size, in bytes, here: _____

6. Configure the satellite modem or router in the path with at least 1 BDP size buffer. This device is sometimes called the *bottleneck buffer*. Satellite modems might configure this satellite modem or router in terms of time such as milliseconds. In this case, the RTT provides the easiest measurement to use. Routers commonly configure this in terms of packets and thus 1 BDP/MTU gives the best approximation for the queue size.

Write the LAN buffer size, in bytes, here: _____

For information about bottleneck buffer, see [“Potential under performance due to short bottleneck buffer” on page 357](#).

To configure transport settings

1. Configure all the SteelHead WAN buffers with the following commands:

```
protocol connection wan send def-buf-size <buffer-size>
protocol connection wan receive def-buf-size <buffer-size>
```

Or, choose Optimization > Network Services: Transport Settings, select WAN and LAN receive and send buffers, and click **Apply**.

2. Configure your remote SteelHeads with the desired transport options from the commands in the following table.

Transport optimization option	CLI command
Enable BW estimation	<code>tcp-cong-ctrl mode bw-est</code>
Enable error recovery	<code>tcp-err-recovery loss-recovery mode always</code>
Disable error recovery	<code>tcp-err-recovery loss-recovery mode disable</code>
Enable SCPS per connection	<code>tcp cong-ctrl mode per-conn-tcp</code>
Enable SCPS error tolerance	<code>tcp cong-ctrl mode err-tol-tcp</code>
Set back to default TCP	<code>tcp cong-ctrl mode default</code>

Or, choose Optimization > Network Services: Transport Settings, select the appropriate radio button, and click **Apply** (Figure 15-2).

Figure 15-2. Transport Settings page

The screenshot shows the 'Transport Settings' page in the SteelHead Management Console. The breadcrumb trail is 'Network Services > Transport Settings'. The page is divided into three main sections: TCP Optimization, Buffer Settings, and Single-Ended Connections. At the bottom is an 'Apply' button.

Transport Settings

Network Services > Transport Settings ?

TCP Optimization

Congestion Control Algorithm

- ☐ Auto-Detect
- ☒ Standard (RFC-Compliant)
- ☐ HighSpeed
- ☐ Bandwidth Estimation
- ☐ SkipWare Per-Connection (SCPS License Required)
- ☐ SkipWare Error-Tolerant (SCPS License Required)

☐ Enable Rate Pacing

Buffer Settings

LAN Send Buffer Size: bytes

LAN Receive Buffer Size: bytes

WAN Default Send Buffer Size: bytes

WAN Default Receive Buffer Size: bytes

Single-Ended Connections

☐ Enable Single-Ended Connection Rules Table

Apply

3. If you have a mixed environment, configure your hub SteelHead to use automatic detect TCP optimization to reflect the various transport optimization mechanisms of your various remote site SteelHeads.

You can also hard code your hub SteelHead to the desired setting.

4. Restart the optimization service, either with the Management Console or the CLI.

We recommend that you test a few different transport settings, such as the WAN buffer sizes, at different remote sites and determine which settings work best for your environment.

For information about automatic detect TCP, see [“Configuring automatic detect TCP optimization” on page 338](#). For information about gathering performance characteristics and configuring transport settings, see the *Riverbed Command-Line Interface Reference Manual* and the *SteelHead User Guide*.

Configuring rate pacing

The following steps are required for rate pacing to function:

1. After you choose the transport option, select the Enable Rate Pacing check box in the SteelHead Management Console. You can also use the **tcp rate-pacing enable** command.
2. Configure MX-TCP under Advanced QoS.

For more information about rate pacing, see [“Rate pacing” on page 337](#). For more information about MX-TCP and QoS, see [“MX-TCP” on page 123](#).

The relationship between the overall link rate and MX-TCP rate dictates how the rate pacing mechanism operates. Rate pacing exits TCP slow start at the MX-TCP rate if the MX-TCP rate is less than the link rate. If you configure rate pacing in this way, it avoids the first loss on exiting slow start and uses MX-TCP as a scheduler for sending data while still adapting to congestion on a shared network in the congestion avoidance phase.

Alternatively, if the MX-TCP rate is greater than the link rate, then rate pacing exits at the link rate. This exit rate can incur a loss on exiting slow start, or packets are buffered in the bottleneck queue. The sending rate during congestion avoidance is based on a calculation between the rate the transport option (TCP stack) determines and the MX-TCP rate. Over time, the rate pacing mechanism continually probes for the higher MX-TCP rate.

In summary, the relationship works as follows:

- **Link rate greater than MX-TCP rate**—Exit slow start at MX-TCP rate and maintain MX-TCP rate in congestion avoidance.
- **Link rate is greater than 50% of the MX-TCP rate but less than the MX-TCP rate**—Exit slow start at MX-TCP rate and use the congestion avoidance rate determined by the underlying TCP stack selected as the transport option.
- **Link rate less than 50% of the MX-TCP rate or MX-TCP not enabled**—Use the underlying transport option for exiting slow start and the congestion avoidance algorithm.

Because a hub site can be connected to multiple satellite networks and remote sites can use a variety of TCP stacks, it is appropriate for you to use automatic detect on the hub site for rate pacing. You can set up MX-TCP on a site-by-site basis to refine the data rate for each remote. MX-TCP follows the QoS configuration for matching on a site and rule.

The following example shows a configuration for the hub site for two remote sites using rate pacing with different bandwidths:

- Site 1 has subnet 172.16.1.0/24 and a link rate of 2 Mbps
- Site 2 has subnet 172.16.2.0/24 and a link rate of 8 Mbps

Use the following commands on the hub-site SteelHead:

```
tcp cong-ctrl mode auto
tcp rate-pacing enable
```


Configuring single-ended connection rule table settings

Use the single-ended connection rule table to manage which flows are optimized or passed through for SCPS optimization. Configuration of the single-ended optimization rule table is similar to the in-path rules, but you must enable the single-ended connection rule table to apply the rules.

To enable the single-ended connection rule table

- Connect to the CLI and enter the following command:

```
tcp sat-opt scps scps-table enable
```

You must have RiOS 8.5 or later to enable the single-ended connection rule table and SCPS compression with third-party WAN optimizers or TCP-PEPs.

To enable the single-ended connection rule table and SCPS compression with third-party WAN optimizers or TCP-PEPs

- Connect to the CLI and enter the following command:

```
tcp sat-opt scps scps-table enable  
tcp sat-opt scps legacy-comp enable
```

You can also complete the following procedure from the SteelHead Management Console Optimization > Network Services: Transport Settings page.

Figure 15-3. Transport Settings page with single-ended connection rule and SCPS compression

Transport Settings Network Services > Transport Settings ?

TCP Optimization

Congestion Control Algorithm

- ☐ Auto-Detect
- ☐ Standard (RFC-Compliant)
- ☐ HighSpeed
- ☒ Bandwidth Estimation
- ☐ SkipWare Per-Connection
- ☐ SkipWare Error-Tolerant

☐ Enable Rate Pacing

Buffer Settings

LAN Send Buffer Size: bytes

LAN Receive Buffer Size: bytes

WAN Default Send Buffer Size: bytes

WAN Default Receive Buffer Size: bytes

Single-Ended Connections

☒ Enable Single-Ended Connection Rules Table

☒ Enable SkipWare Legacy Compression

Apply

Enabling the SCPS single-ended connection rule table or SCPS compression requires a service restart.

To see the current rules in the table, use the **show tcp sat-opt scps rules** command. Following is an example single-ended connection rule table:

```
ssh (config) # show tcp sat-opt scps rules
Rule S P VLAN Source Addr      Dest Addr      Port
-----
  1 Y Y all  all              all            Interactive
  2 Y Y all  all              all            RBT-Proto
def Y Y all  all              all            all

(S) SCPS setting:          Y=Allow SCPS
                        N=SCPS Bypass
(P) Allow only SCPS peering: Y=Enabled
                        N=Disabled
```

Rules are matched from top to bottom. A flow matching a rule stops at the first rule it matches and applies one of the SCPS modes: pass-through or enable. To pass through all flows for SCPS optimization, add a rule at the start of the table to match all sources, all destinations, all destination ports, and all VLANs.

To create a pass through all flows rule

- Connect to the CLI and enter the following command:

```
tcp sat-opt scps rule srcaddr all dstaddr all dstport "all" allow-scps disable scps-peer-only
disable rulenum start
```

Figure 15-6 shows an example of a pass-through rule in the Management Console.

Configuring single-ended rules

The following procedures show how to configure single-ended rules. Configuration of single-ended rules in RiOS 8.5 and later differs from configuration in RiOS 7.0 and 8.0. There are additional options available in single-ended rules: for example, using a single-ended proxy, enabling SCPS discovery or third-party integration and using rate pacing.

Figure 15-4 shows an example of adding a single-ended rule configured in the SteelHead Management Console using SCPS per connection as the TCP stack and rate pacing enabled. You can use this configuration for when you interpolate with a third-party WAN optimizer or TCP-PEP and your network could benefit from using rate pacing with SCPS per connection as the transport option. Rate pacing requires that you configure MX-TCP with advanced QoS and MX-TCP only supports IPv4 traffic.

Figure 15-4. Single-ended rule with SCPS per connection and rate pacing

Single-Ended Connection Rules:

▼ Add New Rule ✕ Remove Selected Rules ⇅ Move Selected Rules...

Position: End ▼

Source Subnet: All-IP

Destination Subnet: All-IP Port or Port Label: all

VLAN Tag ID: all

Traffic

For "passthrough" (no optimization) uncheck both "SCPS Discover" and "TCP Proxy."

Status: Optimized

☒ SCPS Discover

☐ TCP Proxy

TCP Optimization

Congestion Control Algorithm: SkipWare Per-Connection ▼

☒ Enable Rate Pacing

Add

For more information about single-ended rules, see [“SCPS single-ended rules” on page 337](#). For information about configuring single-ended before RiOS 8.5, see earlier versions of the *SteelHead Deployment Guide* and *Riverbed Deployment Guide* on the Riverbed Support site at <https://support.riverbed.com>.

To edit a single-ended connection rule

1. Choose Optimization > Network Services: Transport Settings page.
2. Expand the rule that you want to edit.

Figure 15-5. Edit single-ended connection rules

Single-Ended Connection Rules:
 + Add New Rule - Remove Selected Rules ⇄ Move Selected Rules...

Rule	Source	Destination	VLAN	Traffic	SCPS Discover	TCP Proxy	Congestion Control Algorithm	Rate Pacing
▶ 1	All-IP	All-IP:Interactive	All	Passthrough	--	--	--	--
▶ 2	All-IP	All-IP:RBT-Proto	All	Passthrough	--	--	--	--
▼ 3	All-IP	All-IP:All	All	Passthrough	--	--	--	--

Source Subnet:

Destination Subnet: Port or Port Label:

VLAN Tag ID:

Traffic

For "passthrough" (no optimization) uncheck both "SCPS Discover" and "TCP Proxy."

Status: Optimized

☒ SCPS Discover

☐ TCP Proxy

TCP Optimization

Congestion Control Algorithm:

☒ Enable Rate Pacing

Apply

default All-IP All-IP:All All Optimized Enabled Disabled SkipWare Per-Connection Disabled

To add a single-ended connection rule

1. Choose Optimization > Network Services: Transport Settings page.
2. Select Add New Rule.
3. Populate the appropriate fields and settings.
4. Click **Add**.

The changes take place immediately to all new flows.

Figure 15-6 shows how to configure a pass-through rule for all traffic. Clear the SCPS Discover and TCP Proxy check boxes.

Figure 15-6. Configure a pass-through rule for all traffic

Single-Ended Connection Rules:
 + Add New Rule - Remove Selected Rules ⇅ Move Selected Rules...

Position: End ▼

Source Subnet: All-IP

Destination Subnet: All-IP Port or Port Label: all

VLAN Tag ID: all

Traffic

For "passthrough" (no optimization) uncheck both "SCPS Discover" and "TCP Proxy."

Status: Passthrough

☐ SCPS Discover

☐ TCP Proxy

TCP Optimization

Congestion Control Algorithm: SkipWare Per-Connection ▼

☐ Enable Rate Pacing

Add

Figure 15-7, Rule 1, shows an example of a single-ended optimization pass-through rule for all traffic initiated from the client-side SteelHead.

Figure 15-7. Single-ended optimization pass-through rule

Single-Ended Connection Rules:
 + Add New Rule - Remove Selected Rules ⇅ Move Selected Rules...

<input type="checkbox"/>	Rule	Source	Destination	VLAN	Traffic
<input type="checkbox"/>	▶ 1	All-IP	All-IP:All	All	Passthrough
<input type="checkbox"/>	▶ 2	All-IP	All-IP:Interactive	All	Passthrough
<input type="checkbox"/>	▶ 3	All-IP	All-IP:RBT-Proto	All	Passthrough
	default	All-IP	All-IP:All	All	Optimized

The Management Console passes through only locally initiated sessions through the LAN interface. Inbound SCPS sessions (SYNs with SCPS negotiation headers) arriving at the WAN interface are terminated. To bypass these inbound SCPS sessions, use the CLI. To pass through inbound SCPS sessions in the single-ended connection table, use the syntax option **scps-peer-only disable**.

Verification and troubleshooting

This section describes common satellite deployment problems and solutions. This section includes the following topics:

- [“Analyzing connection optimization information” on page 350](#)
- [“Analyzing packets for discovery probe stripping” on page 354](#)
- [“Understanding the health of the satellite signal” on page 356](#)
- [“Potential under performance due to short bottleneck buffer” on page 357](#)
- [“Potential performance impact of loss at the start of flow” on page 358](#)
- [“Variance in SCPS performance” on page 359](#)

Analyzing connection optimization information

After you configure the SteelHead transport settings, you can verify if the solution is working as expected. You can determine which transport optimization method a connection is using on the Current Connections page in the Management Console and the CLI. This section includes the following topics:

- [“Using the SteelHead Management Console to investigate connection details” on page 351](#)
- [“Using the Riverbed command-line interface to Investigate connection details” on page 353](#)

Using the SteelHead Management Console to investigate connection details

The Current Connections page in the Management Console provides extensive information about flows observed by the SteelHead. You can efficiently determine various information about flows using this report. To see the details for a flow, including transport settings, click the magnifying glass icon next to a specific connection.

Figure 15-8. Current Connections page

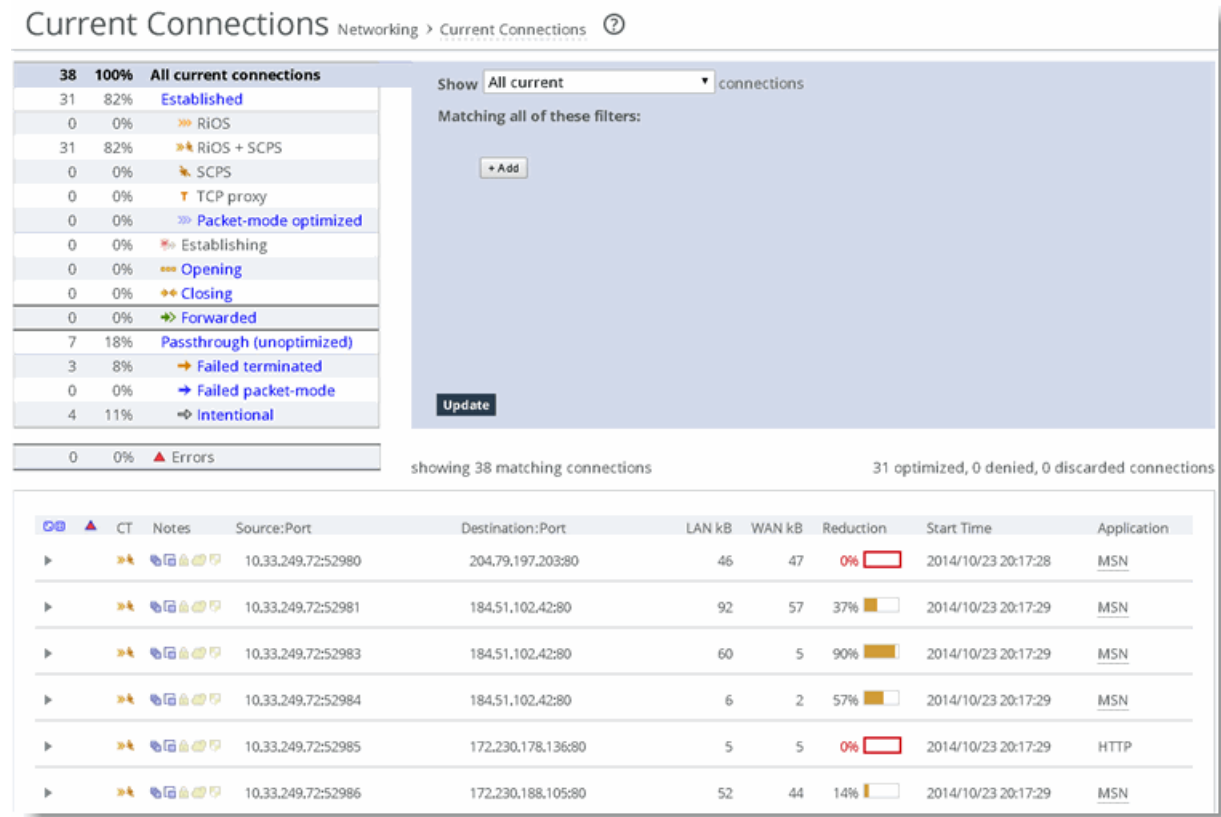
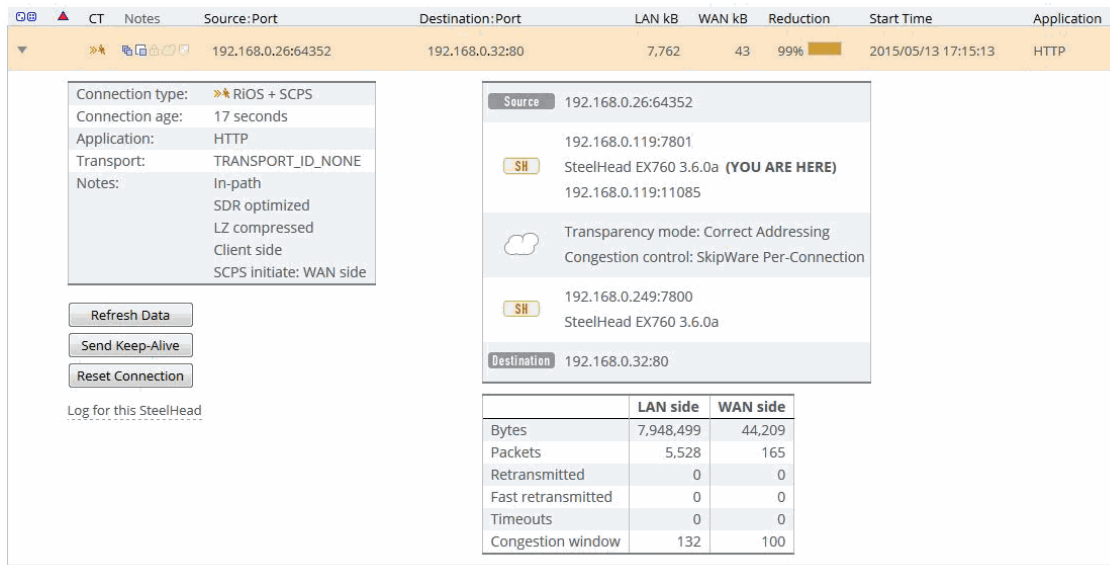


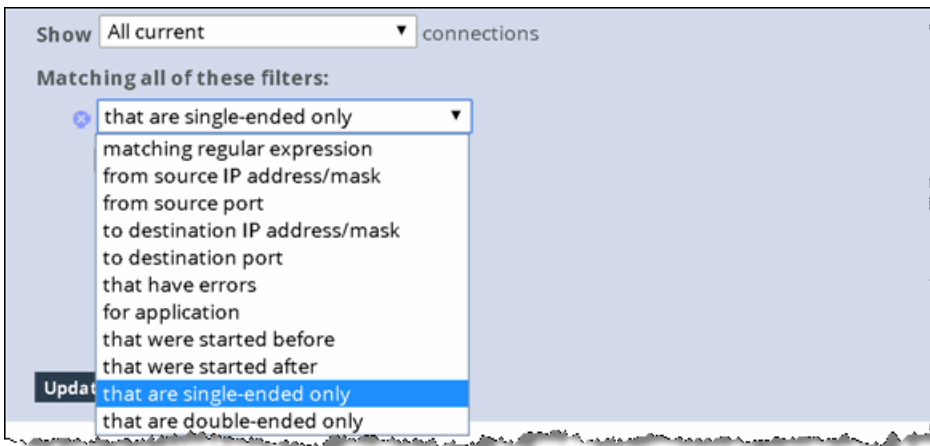
Figure 15-9 shows the details of a specific connection. This flow is optimized and the Congestion control indicates SCPS per connection, which confirms it is using SCPS. The SCPS Initiate is set to WAN, which indicates that this SteelHead is on the client side of this session. If the SCPS Terminate is WAN, then this SteelHead is the server side for the flow. The SCPS Initiate and Terminate cannot both read WAN, because a TCP flow can be initiated only in a single direction. The WAN Congestion Control indicates the transport setting.

Figure 15-9. Connection information



If the current connections report has a lot of flows, you can filter your view. To see only single-ended optimization flows, show All current connections, add a filter, and in the drop-down list select *that are single-ended only* as shown in Figure 15-10.

Figure 15-10. Filtering flow view



Using the Riverbed command-line interface to Investigate connection details

The CLI provides extensive information about flows observed by the SteelHead. The **show connections** command provides a summary list of the connections flowing through a SteelHead. You can use this command with SCPS to quickly see which flows are optimized (O), single-ended optimized (S), and which flows are pass-through (P, PI, PU). Singled-ended optimized flows are included in the established optimized flow total of the **show connections** command. If you want more detail, use the **show connections optimized full** command. See the following examples for details.

```
SH # show connections
T Source Destination App Rdn Since
-----
SO 192.168.139.129:49588 172.16.2.100:80 TCP 0% 2013/07/01 05:26:03
SO 192.168.139.129:49589 172.16.2.100:80 TCP 0% 2013/07/01 05:26:03
SO 192.168.139.129:49590 172.16.2.100:80 TCP 0% 2013/07/01 05:26:03
SO 192.168.139.129:49591 172.16.2.100:80 TCP 0% 2013/07/01 05:26:05
SO 192.168.139.129:49592 172.16.2.100:80 TCP 0% 2013/07/01 05:26:05
SO 192.168.139.129:49593 172.16.2.100:80 TCP 0% 2013/07/01 05:26:05
SO 192.168.139.129:49594 172.16.2.100:80 TCP 0% 2013/07/01 05:26:05
SO 192.168.139.129:49595 172.16.2.100:80 TCP 0% 2013/07/01 05:26:05
SO 192.168.139.129:49596 172.16.2.100:80 TCP 0% 2013/07/01 05:26:05
-----
All V4 V6
-----
Established Optimized: 9 9 0
  RiOS Only (O): 0 0 0
  SCPS Only (SO): 9 9 0
  RiOS+SCPS (RS): 0 0 0
  TCP Proxy (TP): 0 0 0
Half-opened optimized (H): 0 0 0
Half-closed optimized (C): 0 0 0
Establishing (E): 0 0 0
Pass Through: 0 0 0
  Passthrough Intentional (PI): 0 0 0
  Passthrough Unintentional (PU): 0 0 0
Forwarded (F): 0 0 0
Discarded (not shown): 0
Denied (not shown): 0
-----
Total: 9 9 0
```

For more detail, use the **show connection** command options. The syntax requires very specific inputs, and it must be executed while the flow is established through the SteelHead:

```
show connection srcip <ip address> srcport <port> dstip <ip address> dstport <port>
```

You can look at the IP address and port requirements in the show connections flow table. An example of this command follows. The TCP congestion control mechanism is listed in the middle after WAN visibility mode:

```
SH # show connection srcip 192.168.139.129 srcport 49588 dstip 172.16.2.100 dstport 80
Connection not found.
SH # show connection srcip 192.168.139.129 srcport 49597 dstip 172.16.2.100 dstport 80
Type: Single-ended optimized
Optimization Policy: None
Source: 192.168.139.129:49597
Destination: 172.16.2.100:80
```

```

Application:                TCP
Reduction:                  0%
Since:                      2013/07/01 05:27:19

Cloud Acceleration State:   None

Source Side Statistics:
TCP Congestion Algorithm:   Skipware Per Connection
  Bytes:                    328301
  Packets:                  273
  Retransmitted:            0
  Fast Retransmitted:       0
  Timeouts:                 0
  Congestion Window:        235

Destination Side Statistics:
TCP Congestion Algorithm:   New Reno
  Bytes:                    328301
  Packets:                  105
  Retransmitted:            0
  Fast Retransmitted:       0
  Timeouts:                 0
  Congestion Window:        4

```

In most situations, it is easier to use the Current Connections page rather than the CLI for flow investigation. For details, see [“Using the SteelHead Management Console to investigate connection details” on page 351](#).

Analyzing packets for discovery probe stripping

RiOS autodiscovery and SCPS both rely on TCP options to function properly. Some network devices might strip the TCP options and negatively impact discovery or SCPS. In satellite environments, the satellite modems can have TCP acceleration enabled, which might strip TCP options and prevent the SteelHeads from automatically discovering one another.

This section describes how to troubleshoot this issue. You can confirm that TCP options are being stripped by capturing the SYN and SYN/ACK packets on the WAN interface of the server-side SteelHead and looking for TCP options decimal 76 or 78 or both. If you are using SCPS, also look for TCP option decimal 20.

On the server-side SteelHead, you can use the following commands to capture only SYN and SYN/ACK packets for the wan0_0 interface:

```

enable
tcpdump -i wan0_0 -s 150 -w myfilename.cap 'tcp[13] & 2 = 2'

```

Note: Press Ctrl+C to stop the packet capture from the CLI.

You can also execute and stop the capture in the Management Console Reports > Diagnostics: TCP Dumps page. From this page, you can download the capture file. If you are running RiOS 7.0 or later, you can use Pilot Enterprise to remotely start, stop, and analyze packet captures.

After you have downloaded the capture file, open it with a packet analyzer.

If you are using Wireshark 1.6.1 or later to analyze packets, the information row in the Pack List pane begins with S+ or SA+ if a RiOS autodiscovery probe is present in the TCP option field of a SYN or SYN/ACK, respectively. If you have many packets, use the Wireshark display filter `tcp.options.rvbd.probe==1` to display packets with RiOS discovery probes in the TCP option field.

To filter just for SCPS TCP options, use the display filter `tcp.options.scps==1`. Remember that SCPS TCP options are applied to only SYN or SYN/ACK packets. To filter for both RiOS and SCPS discovery probes, use the Wireshark display filter `tcp.options.rvbd.probe==1 || tcp.options.scps==1`. If you do not see any packets with RiOS or SCPS discovery probes, you likely have satellite modems stripping the TCP options field due to TCP acceleration.

After you select the desired packet, inspect the TCP option field to confirm that if the appropriate discovery probes are present. If they are not present at the server-side SteelHead, some device is stripping the probes: for example, a satellite modem or firewall.

Figure 15-11 shows that the SYN packet (#16) is highlighted in Wireshark. In the Packet List pane, the Information column starts with S+, which denotes that the packet has a RiOS discovery probe in the TCP option field. In the Packet Details pane, the TCP option entry from the SteelHead is highlighted in gray, and the details of the probe are decoded. In the Packet Bytes pane, you can see that the actual bytes for the RiOS discovery probe are highlighted and begin with 4c (0x4c is the hexadecimal representation of decimal 76).

Figure 15-11. Packet information in Wireshark

No.	Time	Src	Src Port	Dst	Dst Port	Protocol	Info
16	2011-09-13 10:50:30.799718	192.168.32.111	54829	192.168.32.241	21	TCP	S+, 54829 > ftp [SYN]
17	2011-09-13 10:50:30.803977	192.168.32.241	ftp	192.168.32.111	54829	TCP	SA+, ftp > 54829 [SYN]
18	2011-09-13 10:50:31.405639	192.168.32.61	40276	192.168.32.63	7800	TCP	40276 > rbt-ca [SYN] S
19	2011-09-13 10:50:31.405752	192.168.32.63	rbt-ca	192.168.32.61	40276	TCP	rbt-ca > 40276 [SYN, A
20	2011-09-13 10:50:32.007558	192.168.32.61	40276	192.168.32.63	7800	TCP	40276 > rbt-ca [ACK] S
21	2011-09-13 10:50:32.007593	192.168.32.61	40276	192.168.32.63	7800	TCP	40276 > rbt-ca [PSH] A

Destination port: tcp (21)
[Stream index: 2]	
Sequence number: 1733448814	
Header length: 56 bytes	
Flags: 0x02 (SYN)	
Window size value: 65535	
[Calculated window size: 65535]	
Checksum: 0xdb2e [validation disabled]	
Options: (36 bytes)	
Maximum segment size: 1460 bytes	
No-Operation (NOP)	
Window scale: 2 (multiply by 4)	
No-Operation (NOP)	
No-Operation (NOP)	
Timestamps: TSval 319132692, TSecr 0	
TCP SACK Permitted Option: True	
No-Operation (NOP)	
No-Operation (NOP)	
Rvrbd Probe: Probe Query, CSH IP: 192.168.32.61	
Length: 10	
0000 = Type: 0	
.... 0001 = Version: 1	
Reserved	
CSH IP: 192.168.32.61 (192.168.32.61)	
Application Version: 5	
No-Operation (NOP)	
End of Option List (EOL)	

0010	00 4c 00 34 40 00 40 06	77 c7 c0 a8 20 6f c0 a8	.L.4@. w... o..
0020	20 f1 d6 2d 00 15 67 52	54 6e 00 00 00 00 e0 02	...gR Tn.....
0030	ff ff db 2e 00 00 02 04	05 b4 01 03 03 02 01 01
0040	08 0a 13 05 94 14 00 00	00 00 04 02 01 01 4c 0a
0050	01 01 c0 a8 20 3d 00 05	01 00 =.. ..

If you use Wireshark often to analyze SteelHead performance, you can use color filters to differentiate traffic.

To create Wireshark color filters

1. In Wireshark, choose View > Coloring Rules.

The Coloring Rules dialog box opens.

2. Click **New**.
3. Create a filter name, enter the desired display filter, and set your desired colors.
4. Click **OK**.

You can move the new color rule up or down so that it matches traffic accordingly. Remember that the first coloring rule that is matched is applied to the packet, so the order of color rules is very important.

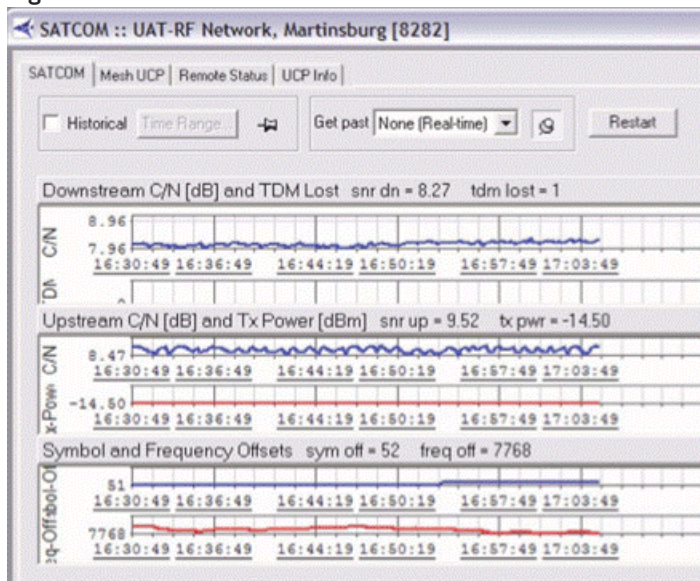
Understanding the health of the satellite signal

Terms such as *signal-to-noise ratio*, *TDM loss*, and other satellite words and abbreviations might be foreign to you. To assist in troubleshooting, you should have a team of satellite experts in your NOC or your service provider's NOC/teleport. When you contact them, the primary questions you want to understand are:

- What is the utilization of the remote site's channel?
- What is the bit error rate for the specific remote site, and is it within an environment's comfort zone?

To analyze a problem, most satellite engineers have a management tool available to track performance. For example, you can use iDirect iMonitor to monitor the health of iDirect hub and remote equipment. Using iMonitor's SATCOM view, you can track performance on a satellite link for an individual remote on the upstream and downstream channels. [Figure 15-12](#) shows a graph from the SATCOM view in iDirect iMonitor.

Figure 15-12. SATCOM



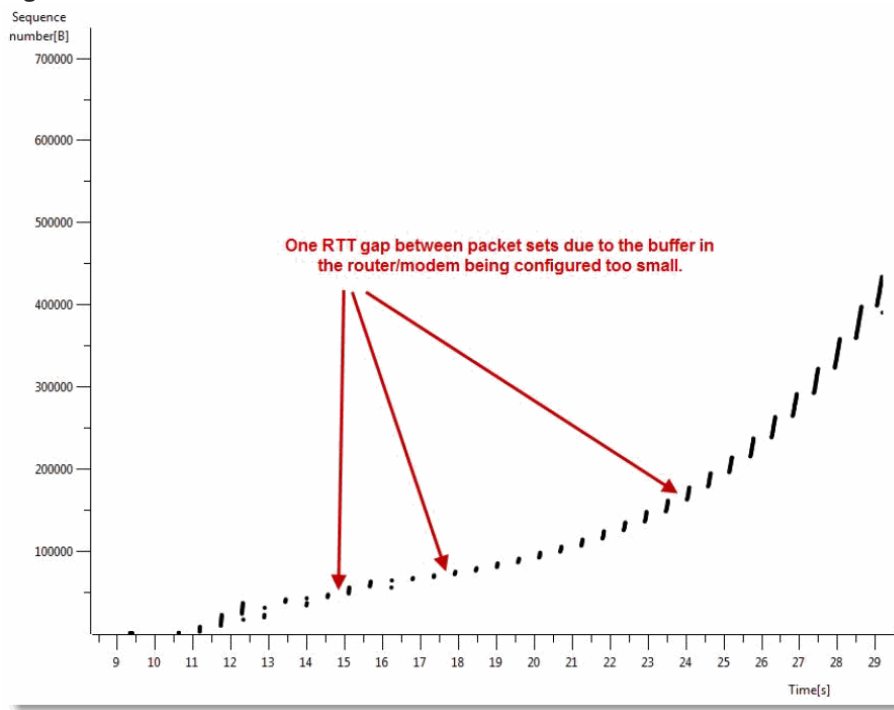
Having your own network monitoring equipment to monitor TCP health, specifically TCP loss/retransmissions, provides an additional tool in the network infrastructure to monitor network health. Riverbed Cascade, NetShark, and Pilot are capable of monitoring these metrics at various granularities.

Potential under performance due to short bottleneck buffer

The bottleneck buffer in the path is most commonly associated with the device connecting to both the high-speed LAN and lower-speed WAN. This device is responsible for absorbing the high rate of packets from the LAN and holding them until these packets can be transmitted over the WAN. Because this device can hold packets the size or length of this buffer, it can have an impact on performance. Remember that the bottleneck buffer can be a satellite modem or a router in the path.

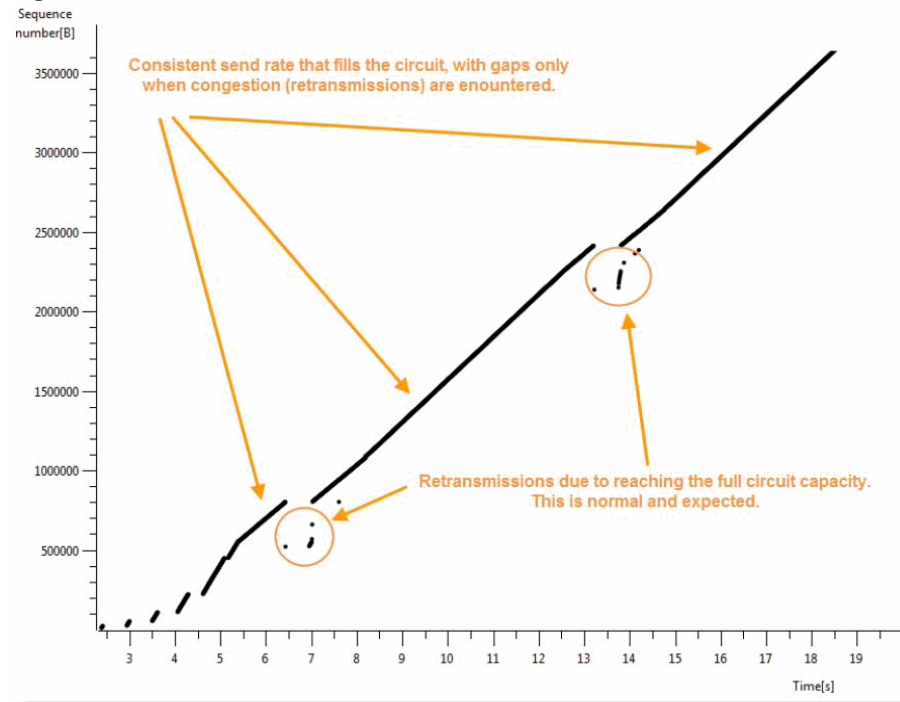
Figure 15-13 shows the bottleneck buffer configured with a buffer size equivalent to 20 milliseconds when the link speed is 2 Mbps with an RTT of 550 milliseconds. The graph shows loss occurs early in the connection and is consistent throughout the connection. A perceptible one RTT gap can be noticed.

Figure 15-13. Small-sized buffer



After the buffer size has been changed, a different pattern emerges. The SteelHead can send data at a continuous rate to fill the WAN circuit. The loss that occurs is normal due to TCP continually detecting if there is a higher available rate.

Figure 15-14. Proper-sized buffer



Potential performance impact of loss at the start of flow

TCP flows are most vulnerable to loss at the beginning of the flow. This loss is due to the initial TCP window size, which is very small at startup. When a TCP flow detects a lost packet in the first several turns, it can negatively impact the acceleration of the flow. Due to the latency in satellite networks, when this occurs, some TCP stacks take significantly longer to recover and accelerate the flow up to a reasonable speed.

When testing in labs, it is very important that you execute adequate flows against each test so that you capture a valid statistical sampling. This testing is critical because loss that coincidentally occurs at the start of flow negatively influences a single test for a certain vendor (Vendor A). Whereas the same test for Vendor B might not realize loss at the beginning of a flow, it might perform much better due to where the loss occurred, relative to the flow's life, and not specifically due to a more superior technology.

Variance in SCPS performance

You might find that the TCP stacks of third-party SCSP solutions vary significantly. These variations can lead to different performance results when running the same transaction or test. When interoperating, you can find variance in performance between third-party devices, or variance depending on the direction data is transmitted. If you have questions or concerns about variance between SCPS solutions, it is best to engage all vendors in a joint discussion. We recommend that you have device configurations and packet captures from all devices to analyze during the discussion.

VPN Routing and Forwarding

This chapter describes how to deploy SteelHeads in an MPLS/VRF environment using Not-So-VRF (NSV) and VRF-aware WCCP. It includes the following sections:

- [“NSV with VRF Select” on page 361](#)
- [“VRF-aware WCCP” on page 372](#)

NSV is a Riverbed network design option that leverages the Riverbed WAN optimization solution by deploying SteelHeads in an existing MPLS deployment using virtual routing and forwarding (VRF). You can use NSV when the WCCP router or the Layer-3 switch operating system does not support VRF-aware WCCP. You can use VRF-aware WCCP as an alternative deployment option to NSV when the WCCP router or Layer-3 switch operating system supports VRF-aware WCCP.

NSV with VRF Select

This section provides an overview of NSV. This section includes the following topics:

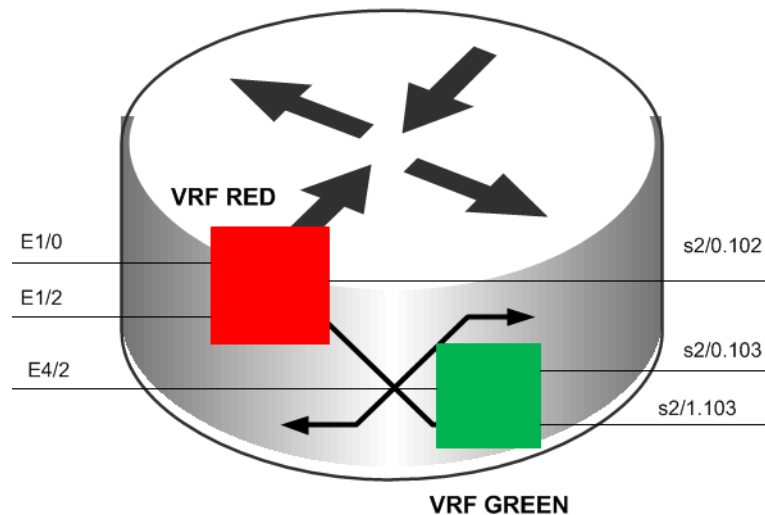
- [“Virtual routing and forwarding ” on page 362](#)
- [“NSV with VRF Select ” on page 363](#)
- [“IOS requirements” on page 364](#)
- [“Prerequisites for NSV” on page 364](#)
- [“Example NSV network deployment” on page 364](#)
- [“Configuring NSV” on page 366](#)

Virtual routing and forwarding

Virtual routing and forwarding (VRF) is a technology used in computer networks that allows multiple instances of a routing table to coexist within the same router at the same time. VRF partitions a router by creating multiple routing tables and multiple forwarding instances. Because the routing instances are independent, you can use the same or overlapping IP addresses without conflict.

Note: The VRF table is also referred to as the *VPNv4 routing table*.

Figure 16-1. Partitioned router using two routing tables

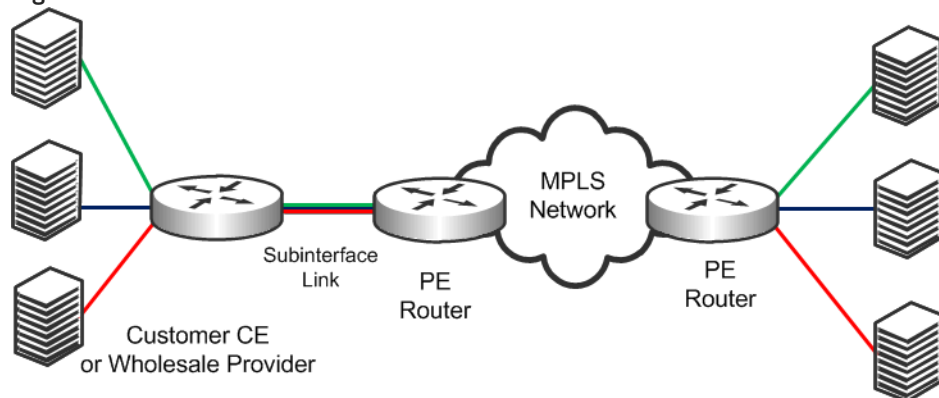


You can implement VRF in a network device by having distinct routing tables, one per VRF. Dedicated interfaces are bound to each VRF.

In [Figure 16-1](#), the red table can forward packets between interfaces E1/0, E1/2, and S2/0.102. The green table, on the other hand, forwards between interfaces E4/2, S2/0.103, and S2/1.103.

The simplest form of VRF implementation is VRF Lite, as shown in [Figure 16-2](#). VRF Lite uses VRFs without multiprotocol label switching (MPLS). In this implementation, each router within the network participates in the virtual routing environment in a peer-based fashion. This implementation extends multiple VPNs from a provider edge (PE) device onto non-MPLS customer edge (CE) devices, which support multiple VRFs. It also replaces the requirement for separate, physical CE devices.

Figure 16-2. VRF lite



NSV with VRF Select

NSV is a Riverbed network design option that leverages the Riverbed WDS solution by deploying SteelHeads in an existing MPLS deployment using VRF. We recommend using NSV in an MPLS/VRF environment to deploy SteelHeads while retaining existing overlapping address spaces.

The concept of NSV originates in an MPLS VPN environment with multiple hosts in the same source VPN. The hosts require access to different servers in various destination VPNs. This deployment is difficult to implement if a particular subinterface is VRF-attached. A subinterface is a way to partition configuration information for certain subsets of traffic that arrive or leave a physical interface.

NSV uses the IOS MPLS VPN VRF Select feature, which essentially eases the requirement of a VRF-attached subinterface.

The VRF Select feature uses policy-based routing (PBR) at the ingress interface of the VRF router to determine which VRF to forward traffic to. In most cases, the VRF router is a PE device. In a VRF-Lite implementation, the VRF router is a CE device. The VRF router determines the routing and forwarding of packets coming from the customer networks (or VPNs). The access control list (ACL) defined in the PBR route map matches the source IP address of the packet. If it finds a match, it sends the packet to the appropriate MPLS VPN (the VRF table).

For more information about PBR, see [“Policy-Based Routing Virtual In-Path Deployments” on page 287](#).

The VRF table contains the virtual routing and forwarding information for the specified VPN. It forwards the selected VPN traffic to the correct MPLS label switched path (LSP), based upon the destination IP address of the packet.

NSV with VRF Select removes the association between the VRF and the subinterface. Decoupling the VRF and the subinterface allows you associate more than one MPLS VPN to the subinterface. The subinterface remains in the IPv4 dimension in VRF Select (as compared to the VPNv4 address space, in which it resides when it is VRF-attached). The subinterface is still IPv4-based, but it becomes aware of VRF Select by replacing the `ip vrf forwarding` Cisco command with `ip vrf receive` command.

The result is that the subinterface becomes *Not-So-VRF*. The subinterface still resides in the global IPv4 table, but it now uses PBR for the VRF switch. The PBR route map matches criteria based on traffic flows to be optimized.

IOS requirements

Cisco recommends the following minimum IOS releases for an MLPS VPN VRF Selection using PBR deployment.

Cisco hardware	Cisco IOS
Most Router Platforms	12.3(7)T or later
C76xx	12.2(33)SRB1, 12.2(33)SRB2, 12.2(33)SRC, 12.2(33)SRC1, 12.2(33)SRC2
ASR 1000 Series Router	XE 2.1.0, 2.1.1, 2.1.2, 2.2.1

Note: Regardless of how you configure a SteelHead, if the Cisco IOS version on the router or switch is below the current Cisco minimum recommendations, it might be impossible to have a functioning NSV implementation, or the implementation might not have optimal performance.

Prerequisites for NSV

Before configuring NSV, review the following information:

- A detailed network diagram illustrating the logical connectivity between the data centers and branch offices
- A running configuration of the multiple VRF CE devices
- The exact IOS versions and hardware platforms in use

Example NSV network deployment

The examples in this section shows the following deployment scenarios:

- One SteelHead, configured as a logical in-path (data center)
- One SteelHead, configured as a physical in-path (branch office)
- Both SteelHeads are running RiOS 5.0.3 or later
- The operating system is 12.3(15)T7
- Two units of 3640 series routers
- Two units of WinXP VM hosts
- IP Service Level Agreement (SLA)
- Static routes with tracking

This deployment is the basis for the configuration shown in [“Configuring NSV” on page 366](#).

Figure 16-3 shows a logical in-path NSV deployment in a VRF network environment.

Figure 16-3. Sample NSV network setup

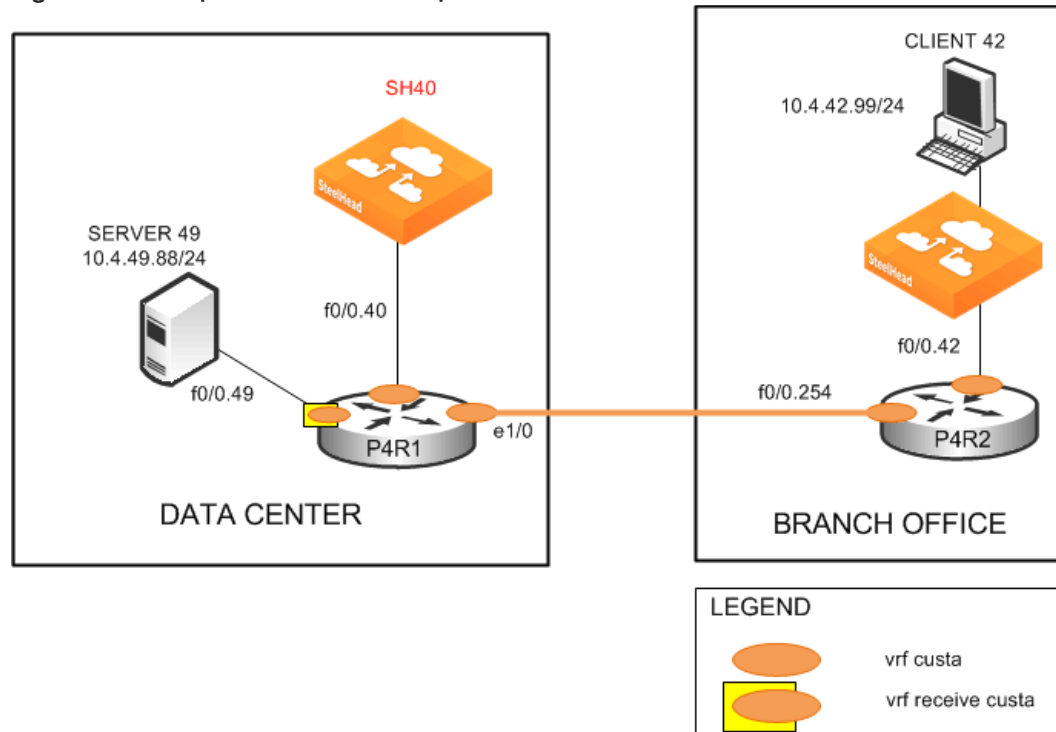


Figure 16-4 shows the NSV deployment shown in Figure 16-3 with intercepted and optimized flows.

Figure 16-4. NSV deployment with intercepted and optimized flows

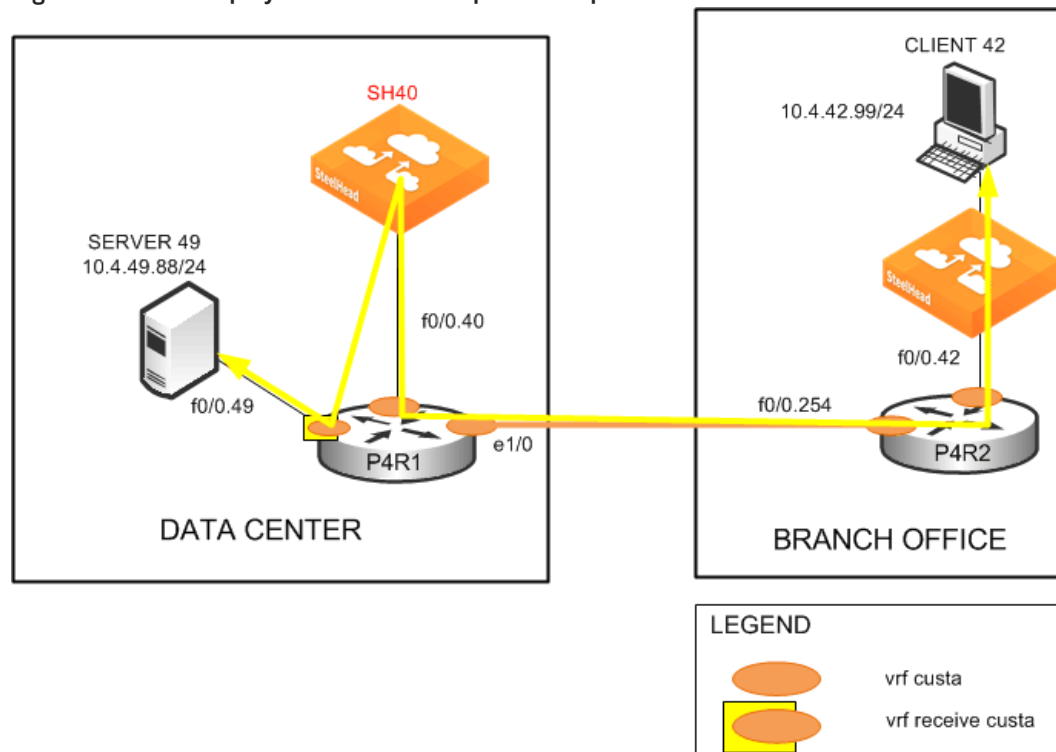
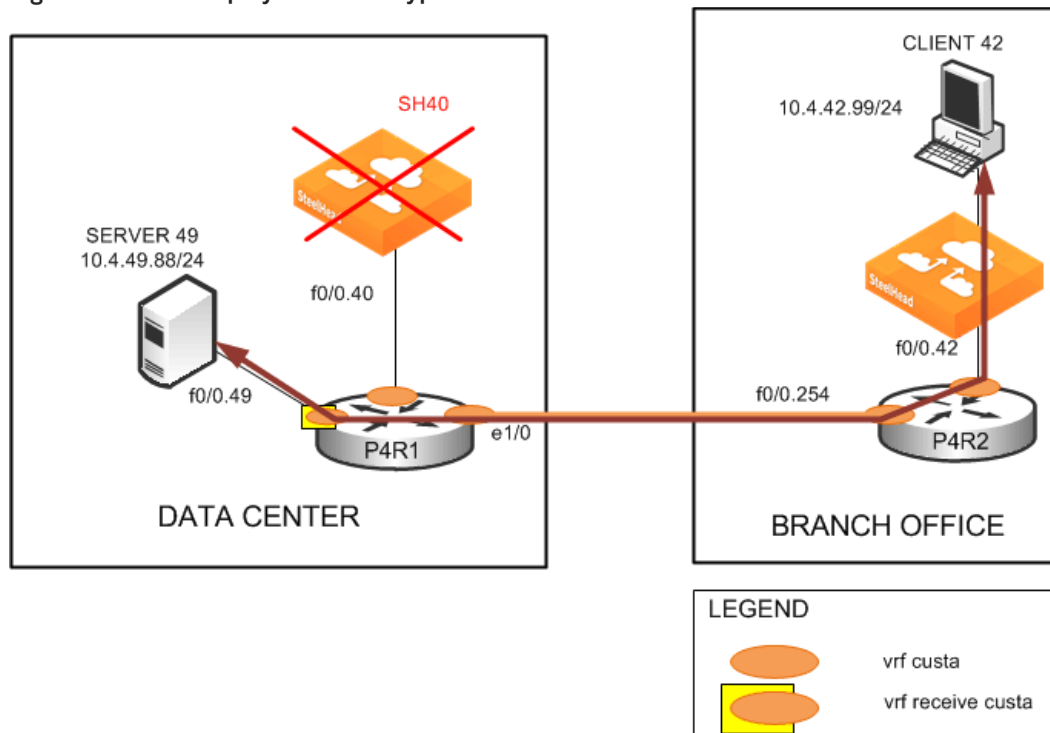


Figure 16-5 shows the NSV deployment with bypassed flows in the event that the data center SteelHead fails.

Figure 16-5. NSV deployment with bypassed flows



Configuring NSV

This section describes how to configure NSV. This section includes the following topics:

- “Basic steps for configuring NSV” on page 366
- “Configuring the data center router” on page 367
- “Configuring the PBR route map” on page 368
- “Decoupling VRF from the subinterface to implement NSV” on page 369
- “Configuring static routes” on page 369
- “Configuring the branch office router ” on page 370
- “Configuring the data center SteelHead” on page 371
- “Configuring the branch office SteelHead ” on page 371

The configuration in this section uses the parameters set up in “Example NSV network deployment” on page 364.

Basic steps for configuring NSV

This section shows the overview of the basic steps to configure NSV with VRF Select. The following sections describe each step in detail.

To configure basic NSV with VRF Select

1. Configure the data center PE or CE router, which includes defining:

- the VRF tables.
- the subinterfaces.
- the PBR route map.
- PBR.
- static routes.
- parameters for monitoring the SteelHead availability.

2. Configure the branch office router.

3. Configure the data center SteelHead.

4. Configure the branch office SteelHead.

The following sections describe each of these steps in detail.

Configuring the data center router

The data center PE or CE router determines the routing and forwarding of packets coming from the customer networks or VPNs. This device requires the most configuration.

The first step is to define the VRF tables for the SteelHead. For example, you define two VRF tables for SteelHead 40: *custa* for the customer and *wds_a* to use as a dummy VRF table. The dummy VRF table is not tied to any interface. It redirects traffic with a corresponding default route, which points to or exits at the subinterface to the SteelHead.

Note: You cannot enter the **set ip next-hop** Cisco command on a PBR route map configured for VRF Select.

The next step configures the subinterfaces and VRF routing protocol. In this example, you configure the following subinterfaces and define the OSPF VRF routing protocol:

- f0/0.40 (the LAN-to-SteelHead 40)
- e1/0 (the WAN)

Note: This example uses OSPF as the routing protocol, but you can use other protocols, such as RIP, EIGRP, ISIS, and BGP, as well. OSPF uses a different routing process for each VRF. For the other protocols, a single process can manage all the VRFs.

To define the VRF tables and subinterfaces

1. Define the VRF tables for the SteelHead. On the data center router (in this example, P4R1), enter the following commands:

```
hostname p4R1
!
ip cef
!
ip vrf custa
    rd 4:1
    !
```

```
ip vrf wds_a
  rd 4:9
  !
```

2. Configure the VRF subinterfaces and corresponding VRF routing protocol. On the data center router, at the system prompt, enter the following commands:

```
interface FastEthernet0/0.40
  encapsulation dot1Q 40
  ip vrf forwarding custa
  ip address 10.4.40.1 255.255.255.0
  !
interface Ethernet1/0
  ip vrf forwarding custa
  ip address 10.254.4.1 255.255.255.0
  half-duplex
  !
router ospf 4 vrf custa
  redistribute static subnets
  network 10.4.40.0 0.0.0.255 area 0
  network 10.254.4.0 0.0.0.255 area 0
```

This example configures the LAN subinterface f0/0.40, which interconnects SteelHead 40 to use VRF custa. Later, you point the dummy VRF wds_a to a default route (in this example, f0/0.40). This enables a PBR route map at f0/0.49 to redirect incoming traffic from Server 49 to Client 42 to SteelHead 40 for optimization.

In this example, because Client 42 is in the VPN custa (VRF custa), the traffic must return to the VRF custa routing path after optimization. For this redirection to work, the SteelHead 40 must reside in VRF custa and *not* VRF wds_a.

Configuring the PBR route map

VRF Select requires a control mechanism such as PBR to select which particular VRF table a data packet goes to. The next step is to configure a PBR route map, which provides matching criteria for incoming traffic and sets the VRF table.

To configure the PBR route map

- On the data center router, enter the following commands:

```
route-map wds_a permit 10
  match ip address 104
  set vrf wds_a
  !
route-map wds_a permit 20
  set vrf custa
  !
access-list 104 permit tcp host 10.4.49.88 host 10.4.42.99
```


The route map `wds_a` matches incoming traffic from Server 49 to Client 42. When it finds a match, it sets the VRF to `wds_a`, which, in turn, points to default route `f0/0.40`, where SteelHead 40 resides. Binding `f0/0.40` with VRF `custa` ensures that the returning optimized traffic eventually reaches Client 42. The route map also sets any incoming traffic to VRF `custa`—except Server 49 to Client 42.

Note: Ensure that the PBR route map contains a default set `vrf` in the PBR route map to always match a packet that does not match any of the previous criteria.

Note: Because BGP control packets are required to remain categorized as global-IPv4, use an ACL to ensure that these packets do not get forwarded to a VRF table.

Decoupling VRF from the subinterface to implement NSV

The following procedure decouples the association between VRF and a subinterface. It implements NSV by replacing the `ip vrf forwarding` Cisco command with `ip vrf receive` command.

The result is that the subinterface becomes *Not-So-VRF*. The subinterface still resides in the global IPv4 table, but it now uses PBR for the VRF switch. The PBR route map matches criteria based on traffic flows to be optimized.

Note: You must have already defined the PBR route map as described in [“Configuring the PBR route map” on page 368](#) before completing the next step.

To implement VRF Select and PBR

- On the data center router, enter the following commands:

```
interface FastEthernet0/0.49
  encapsulation dot1Q 49
  ip vrf receive custa
  ip address 10.4.49.1 255.255.255.0
  ip policy route-map wds_a
```

The absence of the `ip vrf forwarding` command in this example configuration implies that `f0/0.49` is not associated with any particular VRF and remains in the IPv4 global address space. This makes it possible for the SteelHeads to communicate with the subinterface.

Configuring static routes

Static routes play a crucial role in an NSV deployment, because you use them to fine-tune the routing. The primary, default static route points to the in-path interface to redirect incoming traffic for optimization. (In the following example, traffic is redirected to 10.4.40.101 of SteelHead 40).

The command keyword **track 1** determines whether the in-path IP address of the SteelHead is reachable. The primary, default static route is used only when the in-path IP address for the SteelHead is reachable. If it becomes unreachable, the primary route is removed from the routing table. The second, floating route serves as a backup to avoid black holing traffic and ensure flow continuity.

In this example, when the primary route is removed from the routing table because the SteelHead is unreachable, the second route becomes effective at an administrative weight of 250, points to the WAN interface `e1/0`, and avoids blackholing traffic to ensure flow continuity.

Also, in this example, because FastEthernet0/0.49 (where Server 49 is connected) is still in the IPv4 global address space, you must make it visible in VRF custa. To do this, you assign a third static route, associating it with VRF custa. The third static route points to Server 49 (10.4.49.88) in VRF custa and redistributes it into OSPF.

To define static routes

- On the data center router, enter the following commands:

```
ip route vrf wds_a 0.0.0.0 0.0.0.0 FastEthernet0/0.40 10.4.40.101 track 1
ip route vrf wds_a 0.0.0.0 0.0.0.0 Ethernet1/0 10.254.4.2 250
ip route vrf custa 10.4.49.88 255.255.255.255 FastEthernet0/0.49 10.4.49.88
!
```

To monitor SteelHead availability

- On the P4R1, at the system prompt, enter the following commands:

```
ip sla monitor 1
  type echo protocol ipIcmpEcho 10.4.40.101
  vrf custa
  frequency 5
  !
ip sla monitor schedule 1 life forever start-time now
!
track 1 rtr 1 reachability
```

IP SLA uses the ICMP echo protocol to monitor the availability status of the SteelHead in-path IP address every 5 seconds (in this example, IP address 10.4.40.101 for SteelHead 40:custa). This is tied to the primary default route through the tracking mechanism. The tracking mechanism prevents routing to an unavailable IP destination when the in-path IP address for the SteelHead is down (in this example, SteelHead 40:custa).

Configuring the branch office router

A typical branch office router is a PE VRF or CE VRF-Lite device. Its configuration is minimal and standard. In most environments you probably do not need to configure this device.

To configure the P4R2

- On the P4R2, enter the following commands:

```
hostname P4R2
ip cef
ip vrf custa
  rd 4:1
interface FastEthernet0/0.42
  encapsulation dot1Q 42
  ip vrf forwarding custa
  ip address 10.4.42.1 255.255.255.0
interface FastEthernet0/0.254
  encapsulation dot1Q 254
  ip vrf forwarding custa
  ip address 10.254.4.2 255.255.255.0
router ospf 4 vrf custa
  network 10.4.42.0 0.0.0.255 area 0
  network 10.254.4.0 0.0.0.255 area 0
```

Configuring the data center SteelHead

The data center SteelHead (in this example, VRF *custa*) is another vital component of an NSV deployment. Its configuration is very simple; you simply enable the logical in-path interface.

To configure the data center SteelHead

- On the data center SteelHead, connect to the CLI and enter the following commands:

```
hostname "SH40"
interface inpath0_0 ip address 10.4.40.101 /24
ip in-path-gateway inpath0_0 "10.4.40.1"
in-path enable
in-path oop enable
write memory
restart
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

Configuring the branch office SteelHead

The SteelHead deployed at the branch office needs slightly more configuration than the data center SteelHead. Because you are only implementing VRF Select for redirecting the data center LAN-side traffic, you must define fixed-target rules for the WAN-side traffic.

The following example uses SteelHead 42:*custa*.

To configure the branch office SteelHead:

- On the branch office SteelHead, connect to the CLI and enter the following commands:

```
hostname "SH42"
interface inpath0_0 ip address 10.4.42.101 /24
ip default-gateway "10.4.42.1"
in-path enable
in-path rule fixed-target target-addr 10.4.40.101 target-port 7800 dstaddr
10.4.49.88/32 dstport "all" srcaddr 10.4.42.99/32 rulenum 4
```

Note: You must save your changes or they are lost upon reboot. Restart the optimization service for the changes to take effect.

You can also use autodiscovery to eliminate configuring fixed-target rules if you disassociate the WAN interface (in this example, P4R1 e1/0) from the VRF (in this example, *custa*) the same way you disassociated the LAN interface using VRF Select as described in [“Decoupling VRF from the subinterface to implement NSV” on page 369](#).

The branch office SteelHead could also be a SteelHead Mobile. In this deployment, you could use the Mobile Controller to facilitate configuring the fixed-target rules.

For information about the Mobile Controller, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

VRF-aware WCCP

This section describes how to deploy SteelHeads in an MPLS and VRF environment using VRF-aware WCCP. VRF-aware WCCP is an alternative deployment option to NSV when the WCCP router or Layer-3 switch operating system supports VRF-aware WCCP.

This section includes the following topics:

- [“VRF-aware WCCP design examples” on page 373](#)
- [“VRF-aware WCCP best practices” on page 384](#)

The following table shows, by platform, which minimum Cisco software you need to support VRF-aware IPv4.

Cisco WCCP server platforms	IOS/IOS-XE/NX-OS
ISR	ISR G1 - 15.0(1)M or later
C7200	ISR G2 - 15.2(3)T or later
	C7200 - 15.0(1)M or later
	C7206VX
	<ul style="list-style-type: none"> ■ 12.2SRC5 ■ 12.2SRE1
Catalyst 6500 with Sup2T	Catalyst 6500 with Sup2T - 15.1(1)SY or later
C7600	C7600 - 15.1(3)S or later
ASR 1000	3.1.0S or later
Nexus 7000	4.2(1)
	5.1(5) or later
	6.x or later

Note: The table is subject to change over time. For the latest information, see the platform release notes.

VRF allows multiple instances of a routing table to coexist within the same router or Layer-3 switch at the same time. This increases functionality by allowing network (or Layer-3) paths to be segmented without using multiple devices. Because traffic is automatically segregated, you can use VRF to create separate virtual private networks (VPNs) and can be extrapolated to support multitenant architectures.

Multitenancy is an architecture in which a single instance of a software application serves multiple users. Each user is called a tenant. Tenants can be given the ability to customize some parts of the application.

VRF also increases network security and can eliminate the need for encryption and authentication. Because the routing instances are independent, you can use the same or overlapping IP addresses without conflicting with each other.

We recommend a separate SteelHead per each VRF and tenant. Currently, we do not recommend you join a SteelHead to WCCP service groups that could redirect overlapping IP segments.

From the network design perspective, VRF-aware WCCP associates a separate WCCP instance (WCCP router ID) on each defined VRF. This configuration enables you to configure WCCP service groups on a per-VRF basis. The WCCP service group definition is local to the VRF on which it is defined. The WCCP service groups defined within a VRF are logically separated from each other.

Because the WCCP service group is locally significant to the VRF on which it is defined, you can reuse the service group IDs across different VRFs. This reuse is particularly useful for deployments in which the same addressing scheme is adopted for different tenants. In this case, you can reuse the same WCCP configurations (including redirect ACLs) for the various tenants.

By associating a separate WCCP instance for each VRF, you can allocate independent WCCP service group IDs to different VRFs residing within the same router or Layer-3 switch. This is analogous to server virtualization in which multiple virtual machines (VMs) running different applications are hosted on the same physical server.

In short, you can view the VRF-aware WCCP feature as a separate router (or Layer-3 switch) with its own WCCP instance. On the corresponding SteelHead, you define the corresponding WCCP service group IDs as if you were connecting to a single router (or Layer-3 switch).

VRF-aware WCCP design examples

This section shows design examples pertaining to VRF-aware WCCP with virtual in-path SteelHead deployment:

- [“XYZ Data Services design study \(Nexus 7000/NX-OS Example\)” on page 373](#)
- [“IJK Enterprise design study \(ASR 1000/IOS-XE example\)” on page 378](#)
- [“ABC Retail design study \(ISR/IOS Example\)” on page 381](#)

For brevity, only the VRF-aware WCCP configurations on the WCCP router/Layer-3 switch are shown in these examples. The WCCP client (SteelHead) configurations are not shown.

For information on configuring WCCP and the SteelHead, see [“WCCP Virtual In-Path Deployments” on page 249](#).

Note: XYZ Data Services, IJK Enterprise and ABC Retail are fictitious names. Any resemblance is purely coincidental.

XYZ Data Services design study (Nexus 7000/NX-OS Example)

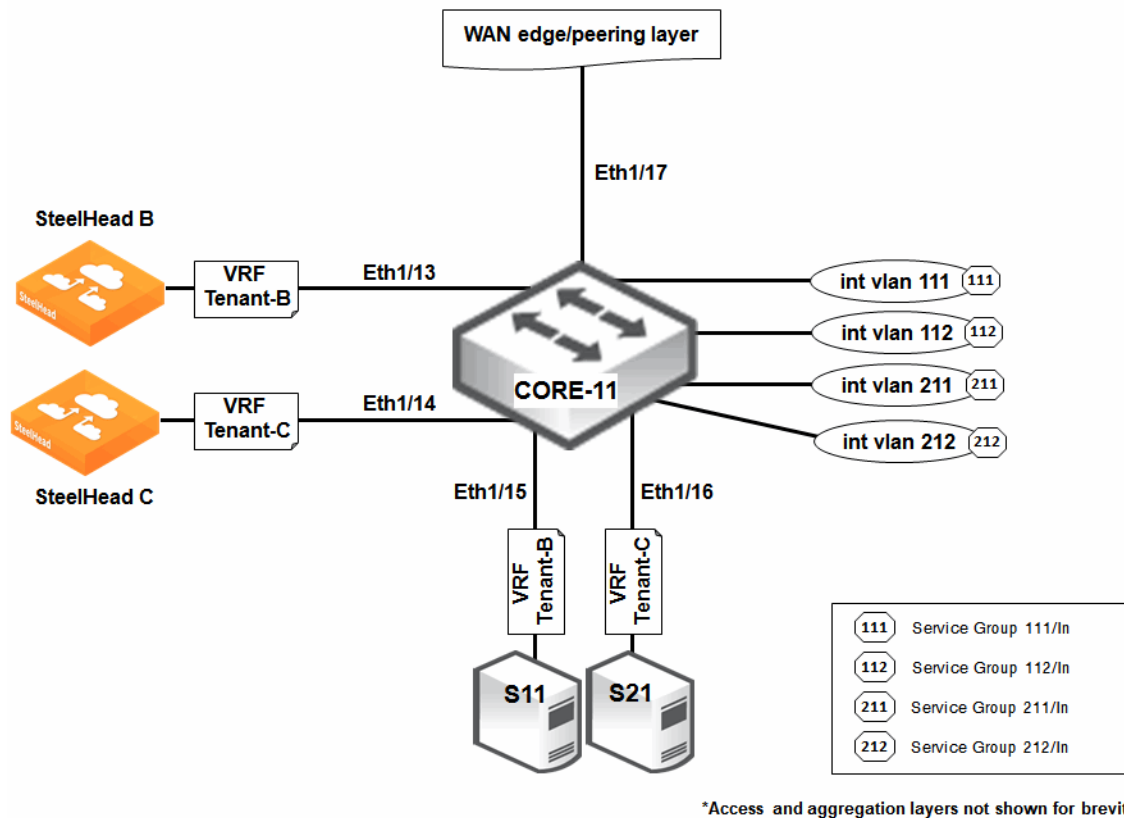
XYZ Data Services is a small IT hosting company that provides premium data services to enterprise customers. One of the data centers of the is located in San Francisco Bay Area. The data center core switches consist of Cisco Nexus 7010 (NX7K) switches that run on NX-OS version 6.2(1).

Note: The Nexus 7000 (NX7K) switch is used as the example platform in this design study to further illustrate the VRF-aware WCCP configurations running on NX-OS in a multitenant environment.

Figure 16-6 shows a portion of the WAN optimization setup on an NX7K switch (CORE-11) in XYZ Data Services Bay Area data center. CORE-11 resides at the data center core layer. The basic design is summarized as follows:

- WAN optimization is provisioned for both Tenant-B and Tenant-C.
- CORE-11 is functioning as a multi-VRF Layer-3 switch, with VRF Tenant-B and VRF Tenant-C created for the corresponding tenants.
- Because there are now two different VRFs in CORE-11, VRF-aware WCCP (or two separate instances of WCCP) has to be implemented on CORE-11 for proper interception and redirection of TCP traffic to the correct SteelHeads:
 - TCP traffic to and from server S11 is intercepted by the WCCP instance for VRF Tenant-B and redirected to SteelHead B for optimization.
 - TCP traffic to and from server S21 is intercepted by the WCCP instance for VRF Tenant-C and redirected to SteelHead C for optimization.
- The SteelHeads in this example can be in physical or virtual form factor.

Figure 16-6. WAN optimization with VRF-aware WCCP on a Nexus 7000 switch



To configure the scenario shown in Figure 16-6, use the following procedures.

The Layer-2 configurations are summarized as follows:

- Five ports are used; four as access ports and the remaining one as an 802.1Q trunk port.
- Out of the four access ports:
 - Two (Eth1/13 and Eth1/15) are for Tenant-B connecting SteelHead B and server S11.
 - The other two access ports (Eth1/14 and Eth1/16) are for Tenant-C connecting SteelHead C and server S21.
- The trunk port (Eth1/17) connects to a WAN edge/peering-layer router (not shown in the diagram).

To configure the Layer-2 baseline configurations for NX7K/CORE-11

```
vlan 11,21,111-112,211-212

interface Ethernet1/13
description "Connects SteelHead B"
switchport
switchport access vlan 11
no shutdown

interface Ethernet1/14
description "Connects SteelHead C"
switchport
switchport access vlan 21
no shutdown

interface Ethernet1/15
description "Connects Server S11"
switchport
switchport access vlan 112
no shutdown

interface Ethernet1/16
description "Connects Server S21"
switchport
switchport access vlan 212
no shutdown

interface Ethernet1/17
description "802.1Q Trunk from Peering Layer"
switchport
switchport mode trunk
switchport trunk allowed vlan 111,211
no shutdown
```

The Layer-3 configurations are summarized as follows:

- Based on the six VLANs created in Code Listing 14-1, six corresponding Layer-3 switch virtual interfaces (SVIs) are also created:
 - SVIs Vlan11 (in IP subnet 10.1.11.0/24), Vlan111 (in IP subnet 10.1.111.0/24), and Vlan112 (in IP subnet 10.1.112.0/24) are associated with VRF Tenant-B.
 - SVIs Vlan21 (in IP subnet 10.2.21.0/24), Vlan211 (in IP subnet 10.2.211.0/24), and Vlan212 (in IP subnet 10.2.212.0/24) are associated with VRF Tenant-C.

- The default VDC is used for VRF Tenant-B and VRF Tenant-C.
- Two OSPFv2 processes are instantiated:
 - One (router ospf 100) for VRF Tenant-B.
 - The other (router ospf 200) for VRF Tenant-C.

To configure the NX7K/CORE-11 Layer-3 baseline configurations

```
feature ospf
feature interface-vlan

vrf context Tenant-B
vrf context Tenant-C

interface Vlan11
no shutdown
vrf member Tenant-B
ip address 10.1.11.1/24
ip router ospf 100 area 0.0.0.0

interface Vlan111
no shutdown
vrf member Tenant-B
ip address 10.1.111.1/24
ip router ospf 100 area 0.0.0.0

interface Vlan112
no shutdown
vrf member Tenant-B
ip address 10.1.112.1/24
ip router ospf 100 area 0.0.0.0

interface Vlan21
no shutdown
vrf member Tenant-C
ip address 10.2.21.1/24
ip router ospf 200 area 0.0.0.0

interface Vlan211
no shutdown
vrf member Tenant-C
ip address 10.2.211.1/24
ip router ospf 200 area 0.0.0.0

interface Vlan212
no shutdown
vrf member Tenant-C
ip address 10.2.212.1/24
ip router ospf 200 area 0.0.0.0

router ospf 100
vrf Tenant-B

router ospf 200
vrf Tenant-C
```


The VRF-aware WCCP configurations are summarized as follows:

- To facilitate the WCCP mask assignment scheme on the NX7K switch, two WCCP service groups are created for each tenant:
 - WCCP service group IDs 111 and 112 for VRF Tenant-B.
 - WCCP service group IDs 211 and 212 for VRF Tenant-C.
- The WCCP interception is based on the inbound direction for each service group.

To configure NX7K/CORE-11 VRF-aware WCCP configurations

```
feature wccp

vrf context Tenant-B
  ip wccp 111 redirect-list 111
  ip wccp 112 redirect-list 112

vrf context Tenant-C
  ip wccp 211 redirect-list 211
  ip wccp 212 redirect-list 212

interface Vlan111
  no shutdown
  vrf member Tenant-B
  ip address 10.1.111.1/24
  ip router ospf 100 area 0.0.0.0
  ip wccp 111 redirect in

interface Vlan112
  no shutdown
  vrf member Tenant-B
  ip address 10.1.112.1/24
  ip router ospf 100 area 0.0.0.0
  ip wccp 112 redirect in

interface Vlan211
  no shutdown
  vrf member Tenant-C
  ip address 10.2.211.1/24
  ip router ospf 200 area 0.0.0.0
  ip wccp 211 redirect in

interface Vlan212
  no shutdown
  vrf member Tenant-C
  ip address 10.2.212.1/24
  ip router ospf 200 area 0.0.0.0
  ip wccp 212 redirect in

ip access-list 111
  10 permit tcp 192.168.100.0/24 10.1.112.100/32

ip access-list 112
  10 permit tcp 10.1.112.100/32 192.168.100.0/24

ip access-list 211
  10 permit tcp 192.168.200.0/24 10.2.212.100/32

ip access-list 212
  10 permit tcp 10.2.212.100/32 192.168.200.0/24
```

Note: HA and other more complex WCCP configurations are omitted in Code Listing 14-3.

For Tenant-B:

- Inbound TCP traffic from remote branch B (192.168.100.0/24) to server S11 (10.1.112.100/32) is intercepted by service group 111 and redirected to SteelHead B for optimization.
- Inbound TCP traffic from server S11 (10.1.112.100/32) to remote branch B (192.168.100.0/24), is intercepted by service group 112 and redirected to SteelHead B for optimization.
- Service groups 111 and 112 are local to VRF Tenant-B.

For Tenant-C:

- Inbound TCP traffic from remote branch C (192.168.200.0/24) to server S21 (10.2.212.100/32) is intercepted by service group 211 and redirected to SteelHead C for optimization.
- Inbound TCP traffic from server S21 (10.2.212.100/32) to remote branch C (192.168.200.0/24), is intercepted by service group 212 and redirected to SteelHead C for optimization.
- Service groups 211 and 212 are local to VRF Tenant-C.

IJK Enterprise design study (ASR 1000/IOS-XE example)

IJK Enterprise specializes in automotive component manufacturing. The enterprise headquarters is based in Stuttgart, Germany. The headquarter WAN routers are made up Cisco ASR 1004 routers running on IOS-XE version 3.13S.

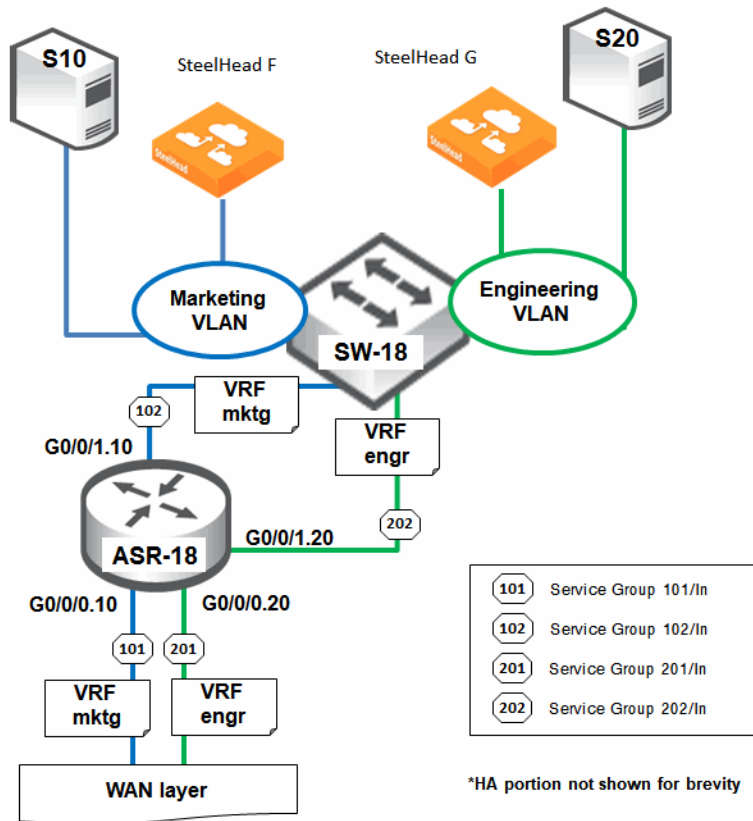
Note: The ASR 1000 (ASR1K) router is used as the example platform in this design study to further illustrate the VRF-aware WCCP configurations running on IOS-XE in a multi-VRF environment.

Figure 16-7 shows a portion of the WAN optimization setup on an ASR1K router (ASR-18) in IJK Enterprise Stuttgart headquarters. ASR-18 resides at the enterprise HQ WAN edge. The basic design is summarized as follows:

- WAN optimization is provisioned for both the marketing and engineering departments in IJK Enterprise.
- ASR-18 is functioning as a multi-VRF router (implementing VRF Lite), with VRF Mktg and VRF Engr created for the corresponding departments.
- Because there are now two different VRFs in ASR-18, VRF-aware WCCP (or two separate instances of WCCP) has to be implemented on ASR-18 for proper interception and redirection of TCP traffic to the correct SteelHeads:
 - TCP traffic to and from server S10 is intercepted by the WCCP instance for VRF Mktg and redirected to SteelHead F for optimization.
 - TCP traffic to and from server S20 is intercepted by the WCCP instance for VRF Engr and redirected to SteelHead G for optimization.
- The SteelHeads in this example can be in physical or virtual form factor.

Note: The deployment of VRFs without MPLS is known as VRF Lite. In VRF Lite, each routed interface (whether physical or virtual) typically belongs to one VRF.

Figure 16-7. WAN optimization with VRF-aware WCCP on an ASR 1000 router



To configure the scenario shown in Figure 16-7, use the following procedure.

The baseline configurations for ASR-18 are summarized as follows:

- Two physical interfaces (G0/0/0 and G0/0/1) are used:
 - G0/0/0 is WAN facing and connects to a WAN-layer router (not shown in the diagram).
 - G0/0/1 is LAN facing and connects to LAN switch, SW-18.
 - Two 802.1Q VLAN subinterfaces (G0/0/0.10 and G0/0/0.20) are derived from G0/0/0.
 - Two 802.1Q VLAN subinterfaces (G0/0/1.10 and G0/0/1.20) are derived from G0/0/1.
- Out of the four subinterfaces created:
 - Two (G0/0/0.10 and G0/0/1.10) are for the Marketing department (marketing server S10 and SteelHead F).
 - The other two (G0/0/0.20 and G0/0/1.20) are for the engineering department (engineering server S20 and SteelHead G).
- Subinterfaces G0/0/0.10 (in IP subnet 10.1.9.0/24) and G0/0/1.10 (in IP subnet 10.1.10.0/24) are associated with VRF Mktg.
- Subinterfaces G0/0/0.20 (in IP subnet 10.2.19.0/24) and G0/0/1.20 (in IP subnet 10.2.20.0/24) are associated with VRF Engr.

- Two OSPFv2 processes are instantiated:
 - One (router ospf 10) for VRF Mktg.
 - The other (router ospf 20) for VRF Engr.

Note: The baseline configurations in this example are not illustrated.

The VRF-aware WCCP configurations are summarized as follows:

- To facilitate the WCCP mask assignment scheme on the ASR1K router, two WCCP service groups are created for each department:
 - WCCP service group IDs 101 and 102 for VRF Mktg.
 - WCCP service group IDs 201 and 202 for VRF Engr.
- The WCCP interception is based on the inbound direction for each service group.

To configure the ASR1K/ASR-18 VRF-aware WCCP configurations

```
vrf definition Engr
rd 20:20
address-family ipv4
exit-address-family

vrf definition Mktg
rd 10:10
address-family ipv4
exit-address-family

ip wccp vrf Engr 201 redirect-list ACL201
ip wccp vrf Engr 202 redirect-list ACL202
ip wccp vrf Mktg 101 redirect-list ACL101
ip wccp vrf Mktg 102 redirect-list ACL102

interface GigabitEthernet0/0/0.10
encapsulation dot1Q 10
vrf forwarding Mktg
ip address 10.1.9.1 255.255.255.0
ip wccp vrf Mktg 101 redirect in

interface GigabitEthernet0/0/0.20
encapsulation dot1Q 20
vrf forwarding Engr
ip address 10.2.19.1 255.255.255.0
ip wccp vrf Engr 201 redirect in

interface GigabitEthernet0/0/1.10
encapsulation dot1Q 10
vrf forwarding Mktg
ip address 10.1.10.1 255.255.255.0
ip wccp vrf Mktg 102 redirect in

interface GigabitEthernet0/0/1.20
encapsulation dot1Q 20
vrf forwarding Engr
ip address 10.2.20.1 255.255.255.0
ip wccp vrf Engr 202 redirect in

ip access-list extended ACL101
permit tcp 192.168.10.0 0.0.0.255 host 10.1.10.100
```

```
ip access-list extended ACL102
  permit tcp host 10.1.10.100 192.168.10.0 0.0.0.255

ip access-list extended ACL201
  permit tcp 192.168.20.0 0.0.0.255 host 10.2.20.100

ip access-list extended ACL202
  permit tcp host 10.2.20.100 192.168.20.0 0.0.0.255
```

Note: HA and other more complex WCCP configurations are omitted in Code Listing 14-4 for brevity.

Marketing department TCP traffic flow and session interception:

- Inbound TCP traffic from remote branch F (192.168.10.0/24) to server S10 (10.1.10.100/32) is intercepted by service group 101 and redirected to SteelHead F for optimization.
- Inbound TCP traffic from server S10 (10.1.10.100/32) to remote branch F (192.168.10.0/24), is intercepted by service group 102 and redirected to SteelHead F for optimization.
- Service groups 101 and 102 are local to VRF Mktg.

Engineering department TCP traffic flow and session interception:

- Inbound TCP traffic from remote branch G (192.168.20.0/24) to server S20 (10.2.20.100/32) is intercepted by service group 201 and redirected to SteelHead G for optimization.
- Inbound TCP traffic from server S20 (10.2.20.100/32) to remote branch G (192.168.20.0/24), is intercepted by service group 202 and redirected to SteelHead G for optimization.
- Service groups 201 and 202 are local to VRF Engr.

ABC Retail design study (ISR/IOS Example)

ABC Retail is a convenience store chain that retails a range of everyday items such as groceries, snack foods, candy, toiletries, soft drinks, tobacco products, and newspapers. The main store office (headquarters) is based in Akron, Ohio. The headquarter WAN routers are made up Cisco 3945 Integrated Services Routers running on IOS version 15.4.2T1.

Note: The Cisco 3900 Integrated Services Router (ISR) is used as the example platform in this design study to further illustrate the VRF-aware WCCP configurations running on IOS in a multi-VRF environment.

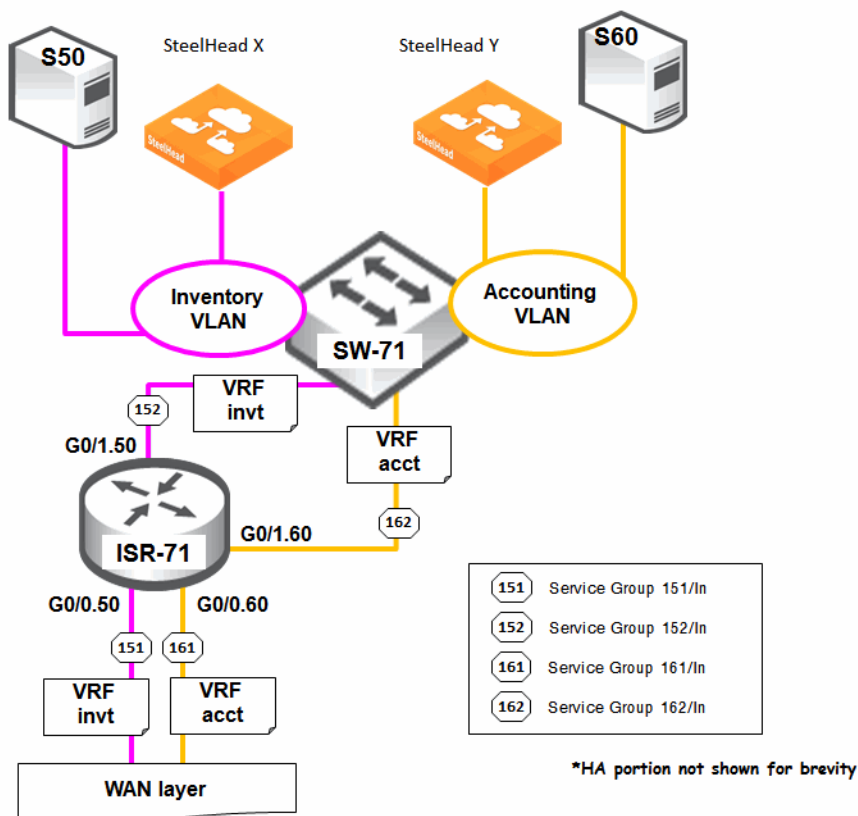
Figure 16-8 shows a portion of the WAN optimization setup on an ISR router (ISR-71) in ABC Retail HQ office. ISR-71 resides at the WAN edge of the headquarter office. The basic design is summarized as follows:

- WAN optimization is provisioned for both the inventory and accounting servers in ABC Retail.
- ISR-71 is functioning as a multi-VRF router (implementing VRF Lite), with VRF Invt and VRF Acct created for the corresponding servers.

- Because there are now two different VRFs in ISR-71, VRF-aware WCCP (or two separate instances of WCCP) will have to be implemented on ISR-71 for proper interception and redirection of TCP traffic to the correct SteelHeads:
 - TCP traffic to and from server S50 is intercepted by the WCCP instance for VRF Invt and redirected to SteelHead X for optimization.
 - TCP traffic to and from server S60 is intercepted by the WCCP instance for VRF Acct and redirected to SteelHead Y for optimization.
- The SteelHeads in this example can be in physical or virtual form factor.

Note: The deployment of VRFs without MPLS is known as VRF Lite. In VRF Lite, each routed interface (whether physical or virtual) typically belongs to one VRF.

Figure 16-8. WAN optimization with VRF-Aware WCCP on a 3900 ISR



To configure the scenario shown in Figure 16-8, use the following procedures.

The baseline configurations for ISR-71 are summarized as follows:

- Two physical interfaces (G0/0 and G0/1) are utilized:
 - G0/0 is WAN facing and connects to a WAN layer router (not shown in the diagram for brevity).
 - G0/1 is LAN facing and connects to LAN switch, SW-71.
 - Two 802.1Q VLAN subinterfaces (G0/0.50 and G0/0.60) are derived from G0/0.
 - Two 802.1Q VLAN subinterfaces (G0/1.50 and G0/1.60) are derived from G0/1.

- Out of the four subinterfaces created:
 - Two (G0/0.50 and G0/1.50) are for inventory purposes (inventory server S50 and SteelHead X).
 - The other two (G0/0.60 and G0/1.60) are for accounting purposes (accounting server S60 and SteelHead Y).
- Subinterfaces G0/0.50 (in IP subnet 10.1.49.0/24) and G0/1.50 (in IP subnet 10.1.50.0/24) are associated with VRF Invt.
- Subinterfaces G0/0.60 (in IP subnet 10.2.59.0/24) and G0/1.60 (in IP subnet 10.2.60.0/24) are associated with VRF Acct.
- Two OSPFv2 processes are instantiated:
 - One (router ospf 50) for VRF Invt.
 - The other (router ospf 60) for VRF Acct.

Note: For brevity, the baseline configurations in this example are not illustrated.

The VRF-aware WCCP configurations are summarized as follows:

- Based to facilitate the WCCP mask assignment scheme on the ISR router, two WCCP service groups are created for each division:
 - WCCP service group IDs 151 and 152 for VRF Invt.
 - WCCP service group IDs 161 and 162 for VRF Acct.
- The WCCP interception is based on the inbound direction for each service group.

To configure the ISR/ISR-71 VRF-aware WCCP configurations

```
ip vrf Acct
 rd 60:60

ip vrf Invt
 rd 50:50

ip wccp vrf Acct 161 redirect-list ACL161
ip wccp vrf Acct 162 redirect-list ACL162
ip wccp vrf Invt 151 redirect-list ACL151
ip wccp vrf Invt 152 redirect-list ACL152

interface GigabitEthernet0/0.50
 encapsulation dot1Q 50
 ip vrf forwarding Invt
 ip address 10.1.49.1 255.255.255.0
 ip wccp vrf Invt 151 redirect in

interface GigabitEthernet0/0.60
 encapsulation dot1Q 60
 ip vrf forwarding Acct
 ip address 10.2.59.1 255.255.255.0
 ip wccp vrf Acct 161 redirect in

interface GigabitEthernet0/1.50
 encapsulation dot1Q 50
 ip vrf forwarding Invt
 ip address 10.1.50.1 255.255.255.0
 ip wccp vrf Invt 152 redirect in
```

```

interface GigabitEthernet0/1.60
 encapsulation dot1Q 60
 ip vrf forwarding Acct
 ip address 10.2.60.1 255.255.255.0
 ip wccp vrf Acct 162 redirect in

ip access-list extended ACL151
 permit tcp 192.168.50.0 0.0.0.255 host 10.1.50.100

ip access-list extended ACL152
 permit tcp host 10.1.50.100 192.168.50.0 0.0.0.255

ip access-list extended ACL161
 permit tcp 192.168.60.0 0.0.0.255 host 10.2.60.100

ip access-list extended ACL162
 permit tcp host 10.2.60.100 192.168.60.0 0.0.0.255

```

Note: HA and other more complex WCCP configurations are omitted in Code Listing 14-5 for brevity.

Inventory division's TCP traffic flow and session interception:

- Inbound TCP traffic from remote branch X (192.168.50.0/24) to server S50 (10.1.50.100/32) is intercepted by service group 151 and redirected to SteelHead X for optimization.
- Inbound TCP traffic from server S50 (10.1.50.100/32) to remote branch X (192.168.50.0/24), is intercepted by service group 152 and redirected to SteelHead X for optimization.
- Service groups 151 and 152 are local to VRF Invt.

Accounting division's TCP traffic flow and session interception:

- Inbound TCP traffic from remote branch Y (192.168.60.0/24) to server S60 (10.2.60.100/32) is intercepted by service group 161 and redirected to SteelHead Y for optimization.
- Inbound TCP traffic from server S60 (10.2.60.100/32) to remote branch Y (192.168.60.0/24), is intercepted by service group 162 and redirected to SteelHead Y for optimization.
- Service groups 161 and 162 are local to VRF Acct.

VRF-aware WCCP best practices

A list of VRF-aware WCCP best common practices is as follows:

- In VRF-aware WCCP implementations, you can view each VRF as a separate router (or Layer-3 switch) with its own WCCP instance. On the corresponding SteelHead you define the corresponding WCCP service group IDs as if you were connecting to a single router (or Layer-3 switch).
- We do not recommend that you join a SteelHead to WCCP service groups that could redirect overlapping IP segments. We recommend a separate SteelHead (physical or virtual form factor) per VRF (or per tenant).
- In VRF-aware WCCP, a service group definition is local to the VRF in which it is defined. The WCCP service IDs defined within a VRF are logically separated from each other, implying that you can reuse the service group IDs across VRFs. We recommend this configuration in scenarios in which cloud service providers adopt the same addressing scheme for different tenants. In this case, you can implement the same WCCPv2 configurations (including redirect ACLs) for the various tenants.

- Although VRF-aware WCCP is supported on the NX7K switch, it is not required if the virtual device context (VDC) is configured on the switch instead of VRF. In this case, the WCCP configurations use the usual global IPv4 routing table. Nevertheless, the Nexus 7000 switch supports only a maximum of four VDCs including the default VDC, so you would still need VRF when the number of tenants exceeds four. Therefore, on the NX7K switch, we recommend that you use both VDC and VRF together, with VRF being the subset of the VDC. Each VDC can be further virtualized to support multiple VRFs.
- The NX7K switch or NX-OS (version 5.1(1) or later) also supports WCCP variable timers. The configuration command is **ip wccp <service-group-number> hia-timeout <here_i_am-timeout-value>**. The default hia-timeout is 10 seconds (implying the WCCP variable timers are not enabled by default on NX-OS). We do not recommend that you change the default HIA-timeout value when configuring WCCP on the NX7K switch.
- In physical in-path deployment, all traffic traverses the respective WAN optimizers and there is no traffic fan-in control. Virtual in-path deployment with WCCP supports traffic fan-in control through the use of redirect access-control lists (ACLs). We recommend the use of redirect ACLs (or in short, redirect-lists) on the WCCP router (or Layer-3 switch) because they provide an explicit fan-in control mechanism that minimizes unnecessary routing, redirection, and packet processing. In most cases, we recommend that you refine the ACLs according to the number of TCP applications that requires WAN optimization for better fan-in control.
- We do not recommend to use a virtual gateway address (derived from HSRP, VRRP, or GLBP) as the WCCP router ID.
- The **in-path module wccp-adjust-mss enable** command adjusts the appropriate MSS size for the SteelHead during WCCP operation. We recommend that you enable this command on the SteelHead to avoid unnecessary packet fragmentation or drops due to outsize packets when WCCP GRE redirection and return are selected.
- The SteelHead enables the WCCP redirection of ICMP messages to support Path MTU Discovery (PMTUD). PMTUD uses ICMP type 3, subtype 4 (Fragmentation needed and DF set) for notification. This is equivalent to the packet-too-big ICMP message defined on ICMP extended ACLs on the router (or Layer-3 switch). We recommend that you enable this feature on the SteelHead and the router (or Layer-3 switch) when end-to-end PMTUD support is required in the WCCP deployment.

Out-of-Path Deployments

This chapter describes out-of-path deployments and summarizes the basic steps for configuring them. This chapter includes the following sections:

- [“Overview of out-of-path deployment” on page 387](#)
- [“Limitations of out-of-path deployments” on page 388](#)
- [“Configuring out-of-path deployments” on page 389](#)

For information about the factors you must consider before you design and deploy the SteelHead in a network environment, see [“Choosing the right SteelHead model” on page 28](#).

Note: Riverbed refers to WCCP and PBR deployments as virtual in-path deployments. This chapter discusses out-of-path deployments, which do not include WCCP or PBR deployments.

This chapter requires that you be familiar with the installation and configuration process for the SteelHead. For details, see the *SteelHead Installation and Configuration Guide*.

Overview of out-of-path deployment

In an out-of-path deployment, only a SteelHead primary interface is required to connect to the network. The SteelHead can be connected anywhere in the LAN. An out-of-path SteelHead deployment does not have a redirecting device. You configure fixed-target in-path rules for the client-side SteelHead. The fixed-target in-path rules point to the primary IP address of the out-of-path SteelHead. The out-of-path SteelHead uses its primary IP address when communicating to the server. The remote SteelHead must be deployed either in a physical or virtual in-path mode.

[Figure 17-1](#) shows an out-of-path deployment.

An out-of-path deployment is generally located on the server side and is often described as a *server-side out-of-path deployment*.

You can achieve redundancy by deploying two SteelHeads out-of-path at one location, and by using both of their primary IP addresses in the remote SteelHead fixed-target rule. The fixed-target rule allows the specification of a primary and a backup SteelHead. If the primary SteelHead becomes unreachable, the remote SteelHeads use the backup SteelHead until the primary comes back online. If both out-of-path SteelHeads in a specific fixed-target rule are unavailable, the remote SteelHead passes through this traffic unoptimized. The remote SteelHead does not look for another matching in-path rule in the list.

You can use RiOS data store synchronization between the out-of-path SteelHeads for additional benefits in case of a failure. For details, see [“RiOS data store synchronization” on page 27](#).

You can also implement load balancing with out-of-path deployments by using multiple out-of-path SteelHeads and configuring different remote SteelHeads to use different target out-of-path SteelHeads.

You can target an out-of-path SteelHead for a fixed-target rule. You can do this simultaneously for physical in-path and virtual in-path deployments, and is referred to as a *hybrid deployment*.

For information about fixed-target in-path rules, see [“Fixed-target in-path rules” on page 48](#).

Limitations of out-of-path deployments

Although the ease of deploying an out-of-path SteelHead might seem appealing, there are serious disadvantages to this method:

- Connections initiated from the site with the out-of-path SteelHead cannot be optimized.
- Servers at the site detect the optimized traffic coming not from a client IP address, but from the out-of-path SteelHead primary IP address.
- In an out-of-path deployment, the primary interface cannot use DHCP.
- Simplified routing cannot be used with out-of-path deployments.

In certain network environments, a change in the source IP address can be problematic. For some commonly used protocols, SteelHeads automatically make protocol-specific adjustments to account for the IP address change. For example, with CIFS, MAPI, and FTP, there are various places where the IP address of the connecting client can be used within the protocol itself. Because the SteelHead uses application-aware optimization for these protocols, it is able to make the appropriate changes within optimized connections and ensure correct functioning when used in out-of-path deployments. However, there are protocols, such as NFS, that cannot function appropriately when optimizing in an out-of-path configuration.

Important: If you use out-of-path deployments, ensure correct operation by carefully selecting which applications you optimize. Even with protocols in which RiOS specifically adjusts for the change in source IP address on the LAN, there might be authentication, IDS, or IPS systems that generate alarms when this change occurs.

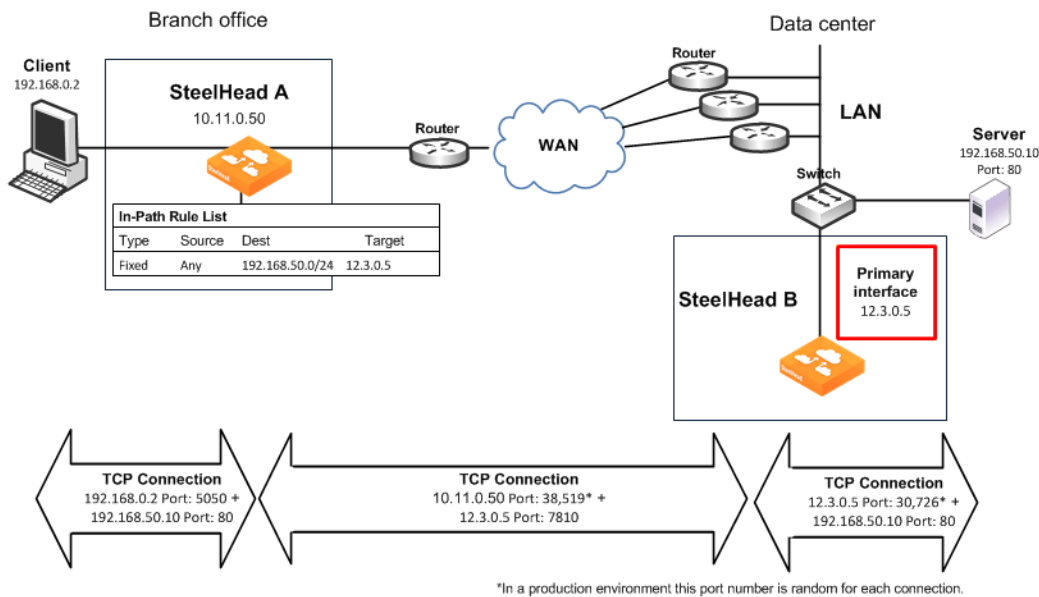
Because of the disadvantages specific to out-of-path deployments, and the requirement of using fixed-target rules, out-of-path deployment is not as widely used as physical or virtual in-path deployments. Out-of-path is primarily used as a way to rapidly deploy a SteelHead in a site with very complex or numerous connections to the WAN.

Configuring out-of-path deployments

Figure 17-1 shows a scenario in which fixed-target in-path rules are configured for an out-of-path SteelHead primary interface.

Note: This section provides the basic steps for out-of-path network deployments. It does not provide detailed procedures. Use this section as a general guide.

Figure 17-1. A Fixed-target in-path rule to an out-of-path SteelHead primary IP address



In this example, you configure:

- SteelHead A with a fixed-target in-path rule specifying that traffic destined to a particular web server at the data center is optimized by the out-of-path SteelHead B.
- The TCP connection between the out-of-path SteelHead, SteelHead B, and the server uses the SteelHead primary IP address as the source, instead of the client IP address.

To configure a basic out-of-path deployment

1. On SteelHead A, connect to the CLI and enter the following commands:

```
enable
configure terminal
in-path rule fixed-target target-addr 12.3.0.5 dstaddr 192.168.50.0/24 dstport 80 rulenum end
```

2. On SteelHead B, connect to the CLI and enter the following commands:

```
enable
configure terminal
out-of-path enable
```


Data Protection Deployments

This chapter describes the configuration and deployment of SteelHeads for data protection solutions. By leveraging SteelHeads, you can achieve higher levels of data protection, streamlined IT operations, and reduced WAN bandwidth.

This chapter includes the following sections:

- [“Overview of data protection” on page 391](#)
- [“Planning for a data protection deployment” on page 392](#)
- [“Configuring SteelHeads for data protection” on page 398](#)
- [“Common data protection deployments” on page 403](#)
- [“Designing for scalability and high availability” on page 405](#)
- [“Enhanced visibility and control for SnapMirror” on page 407](#)
- [“Troubleshooting and fine-tuning” on page 409](#)
- [“Third-party interoperability” on page 410](#)

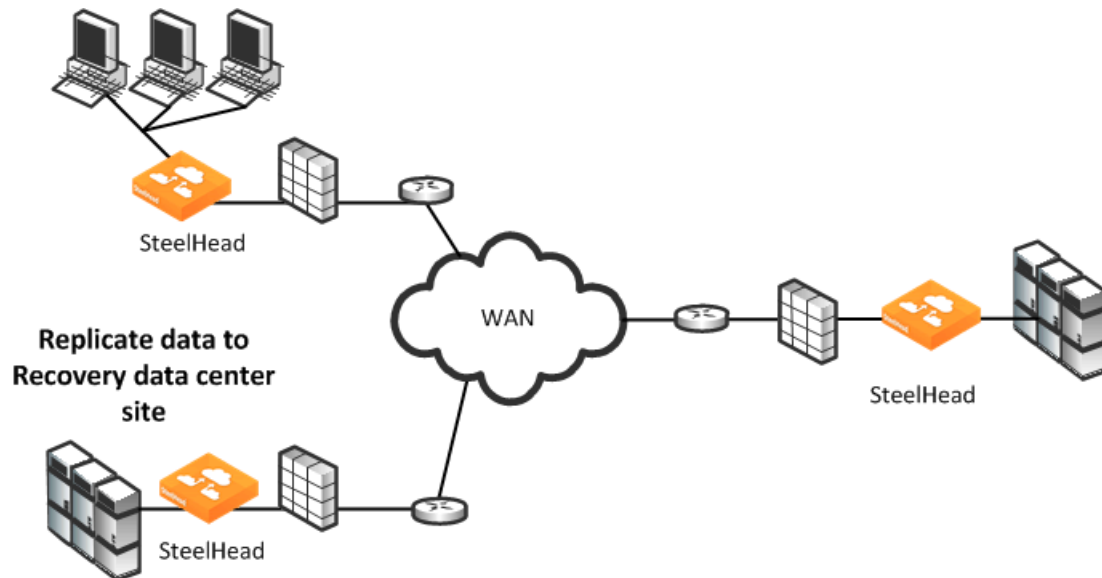
Overview of data protection

To secure and recover important files and data, more data center-to-data center environments (or branch office-to-data center environments) are using WAN-based backup and data replication (DR). WAN optimization is now a critical part of data protection environments because it can substantially reduce the time it takes to replicate data, perform backups, and recover data. Backup and replication over the WAN ensures that you can protect data safely at a distance from the primary site, but it can also introduce new performance challenges.

To meet these performance challenges, Riverbed provides hardware and software capabilities that help data protection environments in the following ways:

- **Reduce WAN bandwidth** - By reducing WAN bandwidth, SteelHeads can lower the total cost of current data protection procedures and, in some cases, make WAN-based backup or replication possible where it was not before.
- **Accelerate data transfer** - By accelerating data transfer, SteelHeads meet or improve time targets for protecting data.

Figure 18-1. Data protection deployment using WAN-based replication



Planning for a data protection deployment

This section describes methods for planning a successful data protection deployment. You must consider several variables, each of which can have a significant impact on the model, number, and configuration of SteelHeads required to deliver the required result. This section includes the following topics:

- [“LAN-side throughput and data reduction requirements” on page 393](#)
- [“Predeployment questionnaire” on page 394](#)

We strongly recommend that you read both of these sections and complete the questionnaire. We also recommend that you consult with Riverbed Professional Services or an authorized Riverbed Delivery Partner when planning for a data protection deployment.

For information about the other factors to consider before you design and deploy the SteelHead in a network environment, see [“Choosing the right SteelHead model” on page 28](#).

LAN-side throughput and data reduction requirements

This section describes requirements and configurations from LAN-side throughput and data reductions. This section includes the following topics:

- “Configuring a nightly full database backup” on page 393
- “Configuring a daily file server replication” on page 394
- “Configuring a very large nightly incremental backup” on page 394

The basis for correctly qualifying, sizing, and configuring SteelHeads for use in a data protection environment depends on that the deployed SteelHeads can:

- receive and process data on the LAN at the required rate (LAN-side throughput), and
- reduce the data by a certain X-Factor, to
- transfer data given certain WAN-side bandwidth constraints.

These constraints are defined by the following formula:

$$\text{LAN-side Throughput} / \text{X-Factor} \leq \text{WAN-side Bandwidth}$$

You derive the LAN-side throughput requirements from an understanding of the maximum amount of data that must be transferred during a given time period. Often, the time allotted to transfer data is defined as a target Recovery Point Objective (RPO) for your organization.

The RPO describes the acceptable amount of data loss measured in time. You must recover data at this time. RPO is generally a definition of what an organization determines is an acceptable data loss following a disaster; it is measured in seconds, minutes, hours, days, or weeks. For example, an RPO of 2 hours means that you can always recover the state of data 2 hours in the past.

Note: The following link provides an Excel throughput calculator that you can use to calculate bandwidth requirements expressed in other forms of time objectives: <https://splash.riverbed.com/message/8478>.

The X-Factor describes the level of data reduction necessary to fit the LAN data into the WAN link. For example, if LAN-side throughput required to meet RPO is 310 Mbps and WAN-side bandwidth available is 155 Mbps, then X-Factor is 2x. X-Factor is highly dependent on the nature of the data, but in practice it generally ranges from 2x (for LZ-only compression) to 4-8x (for default SDR mode).

Configuring a nightly full database backup

Objective:

“I want to copy 1.8 TB of nightly database dumps over my OC-3 within a 10-hour window.”

Formula:

$$1.8 \text{ TB} / 10 \text{ hours} = 400 \text{ Mbps}$$

Solution:

An OC-3 link has a capacity of 155 Mbps. To deliver 400 Mbps, the SteelHead must reduce the total bandwidth over the WAN by $400/155 = 2.58x$.

Configuring a daily file server replication

Objective:

“After consolidating the NetApp file servers from branch offices, I expect daily SnapMirror updates from my data center to go from 400 GB to 4 TB per day. I have a designated DS-3 that is nearly maxed out. Can the SteelHead help me replicate all 4 TB each day using my DS-3?”

Formula:

$$4 \text{ TB} / 1 \text{ day} = 370 \text{ Mbps}$$

Solution:

A DS-3 link has a capacity of 45 Mbps. To deliver 370 Mbps, the SteelHead must reduce the total bandwidth over the WAN by $370/45 = 8.2x$. This bandwidth is within the range of data reduction that the SteelHead can achieve using default SDR, depending on the amount of redundancy present in the data streams.

Configuring a very large nightly incremental backup

Objective:

“The incremental Tivoli Storage Manager (TSM) backup at a remote site is typically 600 GB and the backup window each night is 8 hours. Can I perform these backups over the WAN using a T1 link?”

Formula:

$$600 \text{ GB} / 8 \text{ hours} = 166 \text{ Mbps}$$

Solution:

A T1 link has a capacity of 1.5 Mbps. To deliver 166 Mbps, the SteelHeads must reduce the total bandwidth over the WAN by $166/1.5 = 110x$. This bandwidth is a very high level of reduction that is typically out of range for data protection deployments.

To support backups over the WAN, you must upgrade the WAN link. A T3 link, for example, has a capacity of 45 Mbps. Using a T3 link, the SteelHeads needs to achieve a data reduction of $166/45 = 3.7x$, which is attainable for many deployments.

Predeployment questionnaire

To organize and take a survey of the WAN-side, LAN-side, and X-Factor considerations, use the predeployment questionnaire in the following tables. Discuss your completed survey with Riverbed Professional Services or an authorized delivery partner to determine the best model, number, and initial configuration of the SteelHeads to deploy.

For a Microsoft Word version of the Data Protection questionnaire go to <http://splash.riverbed.com/message/3194>.

Question	Why this is important
WAN-side considerations	
Is this a two-site or a multisite (fan-in, fan-out) data protection opportunity?	In a two-site deployment, the same SteelHead models are often selected for each site. In a multisite (fan-in, fan-out) deployment, the SteelHead at the central site is sized to handle the data transfers to and from the edge sites.
What is the WAN link size?	Knowing the WAN link size is essential in determining: <ul style="list-style-type: none"> ■ which models are feasible for deployment because the SteelHeads specifications are partially based on the WAN rating. ■ the level of data reduction the SteelHeads must deliver to meet the ultimate data protection objective.
What is the network latency between sites?	Knowing the latency in the environment is essential for providing accurate performance estimates. Network latency and WAN link size are used together to calculate buffer sizes on the SteelHead to provide optimal link utilization. Although SteelHeads are generally able to overcome the effects of latency for network protocols used in data protection solutions, some are still latency sensitive.
Is there a dedicated link for disaster recovery?	Environments with a dedicated link are typically easier to configure. Environments with shared links must employ features such as QoS to ensure that data protection traffic receives an adequate amount of bandwidth necessary to meet the ultimate objective.
Question	Why this is important
LAN-side considerations	
Which backup or replication products are you using?	<p>Certain backup or replications products require special configuration. Knowing what is currently in use is essential for providing configuration recommendations and performance estimates. Riverbed has experience with different data protection products and business relationships with many different replication vendors. Many have similar configuration options and network utilization behaviors.</p> <p>Some examples of backup and replication products:</p> <ul style="list-style-type: none"> ■ EMC - SRDF/A RecoverPoint ■ NetApp - SnapMirror, SnapVault ■ IBM - GlobalMirror, XIV replication ■ HDS - TrueCopy, Hitachi Universal Replicator ■ Symantec - NetBackup ■ Vision Solutions - Double-Take ■ CA - ARCserve ■ HP - Continuous Access EVA ■ HDS - TrueCopy ■ IBM - PPRC

Question	Why this is important
Are you using synchronous or asynchronous replication?	<p>Asynchronous replication is typically a very good fit. By comparison, synchronous replication has very stringent latency requirements and is rarely a good fit for WAN optimization.</p> <p>Many types of data protection traffic are not typically considered <i>replication</i> of either type, such as backup jobs.</p>
What is your backup methodology?	<p>Knowing the backup type and schedule provides insight into the frequency of heavy data transfers and the level of repetition within these transfers.</p> <p>Some examples of backup methodologies are:</p> <ul style="list-style-type: none"> ■ A single full backup and an incremental backup for life (synthetic full). ■ A daily full backup. ■ A weekly full backup and a daily incremental backup.
Are your data streams single or multistream? What is the total number of replication streams?	<p>Knowing the number of TCP streams is essential in providing a configuration recommendation and performance estimate. Because SteelHeads proxy TCP/IP, the number of TCP streams created by the data protection solution can impact the SteelHead resource utilization.</p> <ul style="list-style-type: none"> ■ RiOS 5.0 and earlier have a constraint that each TCP session (<i>stream</i>) is serviced by a single CPU core, so splitting the load across many streams is essential to fully use the resources in larger, multicore SteelHeads. ■ RiOS 5.5 or later has multicore features that allow multiple CPU cores to process a single stream. <p>When considering the number of streams, of primary importance is the number of heavyweight data streams that carry significant amounts of traffic. In addition, consider that any smaller control streams that carry a small amount of traffic (such as these streams present in many backup systems and some FCIP systems).</p> <p>Depending on the data protection technology in use, there might be options to increase the number of streams in use. As a first step, determine how many streams are observed in the current environment. Determine whether there is a willingness to increase the number of data streams if a method to do so is suggested.</p>
Is there a FCIP/iFCP gateway? If yes, what is the make, model, and firmware version?	<p>Some FCIP/iFCP gateways (or particular firmware versions of some gateways) do not adhere fully to the TCP/IP or FCIP standards. Depending on what is in use they might require firmware upgrades, special configuration, or cannot be optimized at this time.</p> <p>Gateways are mainly seen in fibre channel SAN replication environments such as SRDF/A, MirrorView, and TrueCopy.</p> <p>Typical firmware versions: Cisco MDS, FCIP v4.1(3) Brocade 7500 FOS v6.3.1, QLogic isr6142 v2.4.3.2.</p>

Question	Why this is important
Is compression enabled on the gateway or the replication product? If yes, what is the current compression ratio?	Most data protection environments using FCIP or iFCP gateways use their built-in compression method, because this is a best practice of the product vendors and the SAN vendors who configure them. However, the best practice for WAN optimization of these technologies is to disable any compression currently in use and employ the SteelHead optimization instead. The first-pass LZ compression in the SteelHead typically matches the compression already in use and then RiOS SDR allows for an overall level of data reduction that improves the previous compression ratio. Knowing the current compression ratio achieved using the built-in compression method is important in determining whether the SteelHeads can improve upon it.
Are SteelHeads already deployed? If yes, what is their make and the RiOS version?	If the environment already has SteelHeads deployed and data protection is a new requirement, knowing the current appliance models in use can determine if adequate system resources are available to meet the objectives without adding additional hardware. Knowing the current RiOS version is essential in determining what features and tuning opportunities are available in the RiOS release to provide the optimal configuration for data protection. If the environment does not already use SteelHeads, Riverbed can recommend the ideal RiOS version based on the environment and data protection objective.
Question	Why this is important
X-factor considerations	
How much new incremental data is added daily or hourly?	The rate of change information is extremely useful alongside the dataset size information to provide accurate performance estimates. If a dataset is too large for a single RiOS data store to find the data patterns for the entire dataset without wrapping continuously, Riverbed can plan system resources based on servicing the amount of data that changes hourly or daily.
What is the total size of the dataset?	For some data protection solutions such as backup, knowing the dataset size is extremely important for RiOS data store sizing. Ideally you want to select SteelHeads that can find the data patterns for the entire dataset without continuously wrapping the RiOS data store. For SAN-based solutions this information can be more difficult to gather, but even rough estimates can help. For example, you can estimate the size of the Logical Unit Number (LUNs) that are subject to replication or the size of the databases stored on an array.
What is the dataset type? For example, Exchange, VMware, SQL, or file system.	Different types of data exhibit different characteristics when they appear on the network as backup or replication traffic. For example, file system data or VMware images often appear as large, sequential bulk transfers and lend themselves well to disk-based data reduction. On the other hand, real-time replication of SQL database updates can often present a workload that requires heavy amounts of disk seeks. These types of workloads can lend themselves better to a memory-based approach to data reduction.

Question	Why this is important
Is the data pre-compressed?	You must determine if precompressed data is present for accurate performance estimates. Data stored at the point of origin in a precompressed format (such as JPEG images, video, or any type of data that has been compressed separately with utility tools such as WinZip), might see limited data reduction from SteelHeads.
Is the data encrypted?	Data stored at the point of origin in a preencrypted format (such as DPM-protected documents or encrypted database fields and records) might see limited data reduction from the SteelHead.
How repeatable is the data?	You must determine if repeatable data is present for accurate performance estimates. Data that contains internal repetition (such as frequent, small updates to large document templates) typically provide very high levels of data reduction.
What LAN-side throughput is needed to meet the data protection goal?	It is the speed of data going in and out of the systems on the LAN that establishes whether the data protection objectives can be met. The LAN-side throughput can be calculated by dividing the total amount of changed data by the time window for the replication or backup job. The WAN-side throughput and level of data reduction represent the level of optimization.

Configuring SteelHeads for data protection

After you deploy the SteelHeads and perform the initial configuration, you can use the features described in this section to deliver an optimal data protection deployment. This section includes the following data protection features:

- [“Adaptive data streamlining feature settings” on page 399](#)
- [“CPU settings” on page 400](#)
- [“Best practices for data streamlining and compression” on page 401](#)
- [“MX-TCP settings” on page 402](#)
- [“SteelHead WAN buffer settings” on page 402](#)
- [“Router WAN buffer settings” on page 403](#)

You can configure the SteelHead features relevant to data protection in the Management Console in the Optimization > Data Replication: Performance page.

Figure 18-2. Performance page data streamlining features

The screenshot shows the 'Adaptive Data Streamlining Modes' configuration page. It is divided into two main sections: 'Maximize Data Reduction' and 'Maximize LAN Throughput'. Under 'Maximize Data Reduction', there are three radio button options: 'Default' (selected), 'SDR-Adaptive', and 'SDR-M'. Under 'Maximize LAN Throughput', there are two radio button options: 'Legacy' and 'Advanced' (selected). Below these sections is a 'CPU Settings' section with a 'Compression Level' dropdown menu set to 'Default'. There are two checkboxes: 'Adaptive Compression' and 'Multi-Core Balancing', both of which are currently unchecked. At the bottom of the page is a dark blue 'Apply' button.

Adaptive data streamlining feature settings

Adaptive data streamlining provides you with the ability to fine tune the data streamlining capabilities and enables you to obtain the right balance between optimal bandwidth reduction and optimal throughput.

The following table describes the adaptive data streamlining settings.

Adaptive data streamlining setting	Benefit	Description
Default SDR/ Classic Data Streamlining	Best data reduction	By default, SteelHeads use their disk-based RiOS data store to find data patterns that traverse the network. Previously seen data patterns do not traverse the network in their fully expanded form. Instead, a SteelHead sends a unique identifier for the data to its peer SteelHead, which sends the fully expanded data. In this manner, data is streamlined over the WAN because unique content only traverses the link once.
SDR-Adaptive	Good data reduction and LAN-side throughput	<p>Dynamically blends different data streaming modes to enable sustained throughput during periods of high disk/CPU-intensive workloads.</p> <p>Legacy - Monitors disk I/O response times and CPU load, and based on statistical trends employs a blend of disk-based deduplication and compression-based data reduction techniques.</p> <p>Use caution with the Legacy setting, particularly when optimizing CIFS or NFS with prepopulation. For more information, contact Riverbed Support.</p> <p>Advanced - Monitors disk I/O response times, CPU load, and WAN utilization, and based on statistical trends employs a blend of disk-based deduplication, memory-based deduplication and compress-based data reduction techniques.</p>
SDR-M	Excellent LAN-side throughput	<p>Performs data reduction entirely in memory, which prevents the SteelHead from reading and writing to and from the disk. Enabling this option can yield high LAN-side throughput because it eliminates all disk latency. SDR-M is typically the preferred configuration mode for SAN replication environments.</p> <p>SDR-M is most efficient between two identical high-end SteelHead models. When SDR-M is configured between two different SteelHead models, the smaller model limits the performance.</p> <p>When you use RiOS SDR-M, RiOS data store synchronization is not possible because none of the data is written to the disk-based data store. For information about data store synchronization, see “RiOS data store synchronization” on page 27.</p>

CPU settings

CPU settings provide you with the ability to balance throughput with the amount of data reduction and balance the connection load. The CPU settings are useful with high-traffic loads to scale back compression, increase throughput, and maximize Long Fat Network (LFN) utilization. This section includes the following topics:

- [“Compression level” on page 401](#)
- [“Adaptive compression” on page 401](#)
- [“Multicore balancing” on page 401](#)

Compression level

The compression level specifies the relative trade-off of LZ data compression for LAN throughput speed. Compression levels 1 to 9 can be specified for fine-tuning. Generally, a lower number provides faster throughput and slightly less data reduction. Setting the optimal compression level provides greater throughput, although maintaining acceptable data reduction.

We recommend setting the compression to level 1 in high-throughput environments such as data-center-to-data-center replication.

Note: The setting is ignored on SteelHead models that are equipped with hardware compression cards.

Adaptive compression

The adaptive compression feature detects the LZ data compression performance for a connection dynamically and turns it off (that is, sets the compression level to 0) momentarily if it is not achieving optimal results. Enabling this feature can improve end-to-end throughput in cases where the data streams are not further compressible.

Multicore balancing

Multicore balancing distributes the load across all CPUs, which maximizes throughput. Multicore balancing improves performance in cases where there are fewer connections than the total number of CPU cores on the SteelHead. Without multicore balancing, the processing of a given connection is bound to a single core for the life of the connection. With multicore balancing, even a single connection leverages all CPU cores in the system.

Continuous code improvements to the performance of single connections has made it unnecessary to enable multicore balancing in RiOS 8.0 or later when 8 or more connections are used. Enabling multicore balancing in these conditions will cause a lower throughput. If multicore balancing is required, use the **datastore codec multicore-bal** CLI command.

Best practices for data streamlining and compression

We recommend the following best practices for data protection scenarios:

- For SAN replication environments (especially with high bandwidth), start with a SDR-M setting and deploy the same model SteelHead on each side.

For information about SAN replication deployments, see [“Storage area network replication” on page 405](#).
- When replicating database log files, the LZ-only compression level typically provides optimal results because database log files contain few repetitive data sequences that can be deduplicated using RiOS SDR.
- For replication of email repositories such as Microsoft Exchange and Lotus Notes, select an appropriate mode of SDR. If WAN capacity is a bottleneck to end-to-end throughput, select a mode that delivers the highest levels of data reduction, such as default SDR or SDR-A. If WAN capacity is not the bottleneck, select a less aggressive form of data reduction, such as SDR-M or Turbo SDR.
- Always set the compression level to 1 in high-throughput data center-to-data center replication scenarios. For more information about best practice guidelines and configuration settings, see [“Common data protection deployments” on page 403](#).

MX-TCP settings

Maximum TCP (MX-TCP) enables data flows to reliably reach a designated level of throughput, which is useful in data protection scenarios where either:

- a dedicated link is used for data protection traffic.
- a known percentage of a given link can be fully consumed by data protection traffic.

For example, if an EMC SRDF/A replication deployment is using peer SteelHeads that are connected to a dedicated OC-1 link (50 Mbps), then you can create an MX-TCP class of 50 Mbps on each SteelHead. In this example, SRDF/A uses port 1748 for data transfers.

See [“Configuring QoS and MX-TCP” on page 165](#) to configure MX-TCP on SteelHeads.

If you cannot allocate a given amount of bandwidth for data protection traffic, but you still require high bandwidth, enable High-Speed TCP (HS-TCP) on peer SteelHeads.

For information about fat pipes, see [“Underutilized fat pipes” on page 476](#). For information about MX-TCP, see [“MX-TCP” on page 123](#).

For more information about MX-TCP as a transport streaming lining mode, see [“Overview of transport streamlining” on page 22](#).

SteelHead WAN buffer settings

In all data protection scenarios, set the SteelHead WAN buffers to at least 2 x BDP. For example, if NetApp SnapMirror traffic is using a dedicated OC-1 link (50 Mbps) with 30 ms of latency (60 ms round-trip time) between sites, then set the SteelHead WAN-side buffers to:

BDP Calculator

Bandwidth	50,000 kbit/sec
Latency	60 ms (round-trip)

$$2 \times \text{BDP} = \frac{50,000 \times 60 \times 2}{8} = 750,000 \text{ bytes}$$

$$\text{BDP} = \frac{50,000 \times 60}{8 \times 1500} = 250 \text{ packets}$$

On all SteelHeads in this environment that send or receive the data protection traffic, enter the following commands:

```
protocol connection wan send def-buf-size 750000
protocol connection wan receive def-buf-size 750000
write memory
restart
```

Router WAN buffer settings

In environments where a small number of connections are transmitting high-throughput data flows, you must increase the WAN-side queues on the router to the BDP. For example, consider an OC-1 link (50 Mbps) with 60 ms latency (RTT):

$$\begin{aligned} \text{BDP} &= 50 \text{ Mbps} * 1,000,000 \text{ b/Mb} * 60 \text{ ms} * (1/1000) \text{ s/ms} * (1/8) \text{ Bytes/bit} * (1/1500) \text{ Bytes/packet} \\ &= 250 \text{ Packets} \end{aligned}$$

On the Cisco router, enter the following hold-queue interface configuration command:

```
hold-queue 250 out
```

You do not need to increase the router setting when using MX-TCP because MX-TCP moves bottleneck queueing onto the SteelHead. This feature allows WAN traffic to enter the network at a constant rate, eliminating the need for excess buffering on router interfaces.

Common data protection deployments

This section describes common data protection deployments. This section includes the following topics:

- [“Remote office, branch office backups” on page 403](#)
- [“Network attached storage replication” on page 404](#)
- [“Storage area network replication” on page 405](#)

Remote office, branch office backups

The remote office, branch office (ROBO) data protection deployment is characterized by one or more small branch office locations, each of which backs up file data from one or more file servers, PCs, and laptops to a central data center. Common applications include Veritas NetBackup, EMC Legato, CommVault Simpana, Sun StorageTek, and backups performed over standard protocols like CIFS and FTP.

In these deployments, WAN links are relatively small, commonly ranging from 512 Kbps on the low end to 10 Mbps on the high end. Also distinct from data center-to-data center replication scenarios where dedicated SteelHeads are typically used exclusively for replication, ROBO backup procedures commonly use the same branch office SteelHeads that are used to accelerate other applications, like CIFS and MAPI. For both of these reasons, ROBO backups commonly require relatively larger levels of WAN bandwidth reduction.

In the Performance page ([Figure 18-2](#)), enter the initial configuration of the peer SteelHeads as follows:

- **Set the Adaptive Streamlining mode to Default** - Due to limited WAN bandwidth in these deployments, it is important to maximize WAN data reduction. The default setting uses disk-based SDR to provide maximum data reduction. File backup workloads typically result in sequential disk access, which works well for disk-based SDR.
- **Set the Compression Level to 6** - Start with aggressive compression to minimize WAN bandwidth.
- **Enable Multicore balancing** - This option allows the SteelHead to use all CPU cores even when there are a small number of connections. Small connection counts can occur if backups are performed nightly, when minimal or no additional traffic is generated.

Network attached storage replication

Network attached storage (NAS) data protection deployment sends primary file data over the WAN to online replicas. Common applications include NetApp SnapMirror, EMC VNX Replicator, and VNX Celerra Replicator.

For information about EMC qualification matrix for Riverbed Technology, see the Riverbed Knowledge Base article *Deploying SteelHeads with EMC Storage*, at <https://supportkb.riverbed.com/support/index?page=content&id=s13363>.

In NAS replication deployments, WAN links are typically large, ranging from T3 (45 Mbps) to OC-48 (2.5 GB). Often NAS replication solutions require dedicated links used exclusively by the NAS replication solution.

As a best practice for high-speed NAS replication solutions, use SteelHeads that are dedicated to only optimizing high-speed NAS replication workloads and that do not optimize large amounts of general application or end-user traffic. Doing this benefits you for the following reasons:

- Increase both the level and predictability of performance delivered by SteelHeads, leading to consistent delivery of recovery point and time objectives (RPO/RTO).
- With separate SteelHeads, the large data sets commonly associated with high-speed replication do not compete for SteelHead data store resources with other user-based traffic, and the reverse.
- You can optimally tune separate SteelHeads for their respective workloads.

Disable any data compression applied on the storage device so that data enters the SteelHead in its raw form. Disabling data compression enables the SteelHead to perform additional bandwidth reduction using SDR.

In the Performance page ([Figure 18-2](#)), enter the initial configuration of the peer SteelHeads as follows:

- **Set the Compression Level to 1** - Higher compression levels produce additional gains in WAN-side bandwidth reduction, but often at a large cost to the CPU resources, which ultimately throttles LAN-side throughput.
- **Enable Multicore Balancing** - Often there are a small number of connections made between storage devices. This option enables the optimization services to balance their processing across all CPU cores.
- **Enable MX-TCP or HS-TCP** - If there is a dedicated WAN-link for the NAS replication traffic or if you know how much bandwidth on a shared link can be allocated to the data transfer, create an MX-TCP class covering the data traffic. If not, enable HS-TCP. If HS-TCP is enabled, increase the router queue length to the BDP. Configure MX-TCP on the QoS Classification page.
- **Set the SteelHead WAN buffers to 2 x BDP** - This option allows the SteelHeads to buffer enough data to continue accepting data from the LAN—even in cases of WAN packet loss.

In cases where WAN links exhibit high-packet loss, you might need to increase the SteelHead WAN buffers higher than 2 x the BDP for optimal throughput.

Storage area network replication

Storage area network (SAN) data protection deployment includes SAN replication products such as EMC Symmetrix Remote Data Facility/Asynchronous (SRDF/A), IBM Global Mirror/IBM Global Mirror, and Hitachi Universal Replicator, including full and incremental backups of databases like Oracle and Exchange.

For more information about SAN replications, see [“Storage Area Network Replication” on page 411](#).

Designing for scalability and high availability

Scalability and high availability are often required in data protection deployments. This section describes the design of data protection solutions which address both requirements. This section includes the following topics:

- [“Overview of N+M architecture” on page 405](#)
- [“Using MX-TCP in N+M deployments” on page 405](#)

For more information about high availability, see [“Multiple WAN router deployments” on page 221](#).

Overview of N+M architecture

The most cost-effective way to provide scalability and high availability is by using an N+M SteelHead architecture or an N+M deployment. In an N+M architecture, N represents the minimum number of SteelHeads that are required to process the total amount of traffic from site to site. M represents the number of additional SteelHeads needed to provide a desired amount of redundancy. For example, a common requirement is to maintain availability in the presence of a single failure. In this case, you can use a N+1 SteelHead deployment architecture.

Using MX-TCP in N+M deployments

This section describes how to use MX-TCP in N+M deployments. This section includes the following topics:

- [“Interceptor and N+M active and backup deployment” on page 406](#)
- [“Interceptor and pass-through connection blocking rules” on page 407](#)

MX-TCP is typically used in data protection deployments when all or part of the WAN bandwidth is dedicated to the data transfers. When using MX-TCP with multiple SteelHeads, MX-TCP settings are set on each SteelHead so that the collection of SteelHeads uses the available WAN bandwidth.

For details, see [“QoS in multiple SteelHead deployments” on page 133](#) and [“MX-TCP” on page 123](#).

In an N+M deployment, the following options effect how to configure MX-TCP:

- **All active, or N+M active** - All N+M SteelHeads participate in optimizing the data transfer. Configure MX-TCP on each SteelHead to use $1/(N+M)$ th of the total available WAN bandwidth. For example, in a 2+1 All Active deployment, configure MX-TCP on each SteelHead to use one-third of the available bandwidth. Less WAN bandwidth is used when one or more SteelHeads are offline. For example, in a 2+1 All Active deployment with one SteelHead offline, two-third of the allocated WAN bandwidth is used by the SteelHeads that remain online.

- **Active and backup, or N primary + M backup** - Exactly N SteelHeads participate in optimizing the data transfer. Configure MX-TCP on each SteelHead to use 1/Nth of the total available WAN bandwidth. If one or more active SteelHeads are offline, backup SteelHeads are used to keep the WAN fully utilized. For example, in a 2+1 Active and Backup deployment, configure MX-TCP on each SteelHead to use one-half of the available bandwidth. If one active SteelHead is offline, the backup SteelHead participates in optimizing the data transfer, keeping the WAN fully utilized.

For information about how to configure an Active and Backup deployment using the SteelHead Interceptor, see [“Interceptor and N+M active and backup deployment” on page 406](#)).

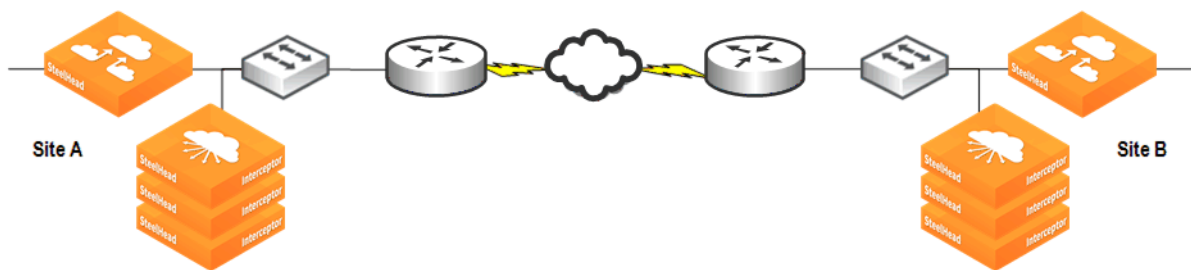
Interceptor and N+M active and backup deployment

When configuring the SteelHead Interceptor for an N+M Active and Backup deployment using the SteelHead Interceptor, load-balance rules are defined which carry out the following actions:

- Balance load across the primary SteelHeads
- Use backup SteelHead in the event of a failure

Figure 18-3 shows a 2+1 Active and Backup deployment.

Figure 18-3. Interceptor N+M



In each site there is a SteelHead Interceptor and three SteelHeads: two are primary and one is the backup. Connections are established from Site A to Site B, and there are four hosts (not depicted) at each site that process equal amounts of data. The following list shows IP addresses for the hosts and SteelHeads at Site A:

- Hosts 1-4: 10.30.50.11 - 10.30.50.14
- Primary SteelHead 1: 10.30.50.15
- Primary SteelHead 2: 10.30.50.16
- Backup SteelHead: 10.30.50.17

The following load-balance rules are used on each SteelHead Interceptor to evenly split the connections established from the four hosts at Site A across the two primary SteelHeads (odd-numbered hosts are redirected to primary SteelHead 1, and even-numbered hosts are redirected to primary SteelHead 2).

```
load balance rule redirect addrs 10.30.50.15 src 10.30.50.11/32
load balance rule redirect addrs 10.30.50.16 src 10.30.50.12/32
load balance rule redirect addrs 10.30.50.15 src 10.30.50.13/32
load balance rule redirect addrs 10.30.50.16 src 10.30.50.14/32
```

The following load-balance rules allow the SteelHead Interceptor to use the backup SteelHead in case either of the primary SteelHeads fails:

```
load balance rule redirect addrs 10.30.50.17 src 10.30.50.11/32
load balance rule redirect addrs 10.30.50.17 src 10.30.50.12/32
load balance rule redirect addrs 10.30.50.17 src 10.30.50.13/32
load balance rule redirect addrs 10.30.50.17 src 10.30.50.14/32
```

The same configuration would be used for the SteelHead Interceptor at Site B, using instead of the IP addresses for the SteelHeads in site B.

Interceptor and pass-through connection blocking rules

In some data protection deployments, it is important to prevent backup and replication connections from being established as unoptimized, or pass-through, connections. These unoptimized connections can have a negative impact on meeting LAN and WAN throughput objectives. Interceptor 2.0.3 or later supports *Pass-through Connection Blocking Rules*. This feature adds a set of rules that can break existing pass-through connections and prevent formation of new ones.

For example, you can create a pass-through blocking rule for port 1748, connect to the Interceptor CLI and enter the following command:

```
in-path passthrough rule block port start 1748 end 1748
```

For details, see the *SteelHead Interceptor User Guide* and the *Riverbed Command-Line Interface Reference Manual*.

Enhanced visibility and control for SnapMirror

The two varieties of SnapMirror are volume based and qtree based. SnapMirror replicates data from one volume or qtree (the source) to another volume or qtree (the mirror). SnapMirror periodically updates the mirror to reflect incremental changes to the source. The result of this process is an online, read-only volume (the mirror), that contains the same data as the source volume at the time of the most recent update.

You can use the information on the mirror to:

- provide quick access to data in the event of a disaster that makes the source volume or qtree unavailable. The secondary copy is nearly identical to the primary copy; every snapshot on the primary copy also exists on the backup copy. You can schedule updates as frequently as every minute.
- update the source to recover from disaster, data corruption (mirror qtrees only), or user error.
- archive the data to tape.
- balance resource loads.
- back up or distribute the data to remote sites.

With data streamlining, the SteelHead optimizes WAN performance for SnapMirror by removing repetitive data from the WAN. Transport streamlining enables TCP to be more efficient, minimizing round trips and maximizing end-to-end throughput.

For environments using NetApp Data ONTAP v7 or Data ONTAP v8 operating in 7-mode, Riverbed provides additional capabilities to enhance the visibility and control of SnapMirror on a volume granular or qtree-granular basis. RiOS 8.5 and later allow you to:

- apply QoS traffic shaping policies on per-volume or qtree basis. You can assign mappings by filer and volume name to one of five volume priorities. Using advanced QoS, you can assign a service class and DSCP value to each volume priority when creating a rule for SnapMirror traffic. Multipath operations are not supported.
- customize SnapMirror traffic optimization on per-volume or qtree basis. You can apply desired optimization algorithms (SDR-Default, LZ-only, and None) for different data types that reside on targeted volumes or qtrees.
- collect and chart SnapMirror statistics such as the total LAN and WAN bytes in and out, throughput, the data reduction at the filer, and volume/qtree granularity.

To benefit from the improved SnapMirror optimization, both SteelHeads must be running RiOS 8.5 or later. SnapMirror optimization is disabled by default.

The following example shows a company with a NetApp filer that is replicating four volumes from New York to San Francisco. These four volumes contain the following data:

- **Volume 1** - Contains archival data of the previous year. The data continues to reside on the primary storage because it is used regularly by the analytics department. However, to preserve server space, the data is stored in compressed format.
- **Volume 2** - Contains graphics and videos. The data is encrypted or not compressible.
- **Volume 3** - Contains an MS Exchange data store of the company users email mailboxes.
- **Volume 4** - Contains lab data that for historical reasons is produced and stored in plain 7-bit ACSII format, but doesn't have repeatable patterns as usual text would.

Because the data composition varies by volume, to best use the SteelHead optimization resources, and to accomplish the best result on the data optimization, you can apply different optimization methods for each volume.

Data in Volume 1 and Volume 2 is neither compressible nor dedupable. If you enable SDR, you have a higher CPU optimization on the SteelHeads and you do not achieve the best use of the SDR data store. The SDR data store can provide higher data reduction to the rest of the data. However, the SteelHead can add value by sending the data using MX-TCP protocol. MX-TCP protocol enables the most of available bandwidth. Traditional TCP cannot accomplish this because of packet loss, high latency, or some combination of these and other impairments.

Data in Volume 3 benefits from SDR because it contains repeatable patterns in the data stream that can both be compressed and replaced with references.

Data in Volume 4 benefits from conventional compression or LZ-only type of SteelHead optimization.

Different volumes can have different change rates and, therefore, have replication service level agreements (SLAs) or recovery point and/or time objectives (RPO/RTO): for example, 4-hour RTO for MS Exchange and 24-hour RTO for images and video. To meet varied requirements, you can apply different QoS policies for those volumes to make increase priority of MS Exchange (Volume 3) data or image data (Volume 2).

For more information about the configuring SnapMirror on the SteelHead, see the *SteelHead User Guide*.

Troubleshooting and fine-tuning

If your data protection deployment is not meeting performance targets after configuring the SteelHeads using the methods described in this chapter, examine the following system components for potential bottlenecks:

- **Application Servers** - Are the server and client fast enough? To perform a LAN baseline check, put the SteelHeads in bypass mode and connect the servers directly through a high-bandwidth network with zero latency to see how fast they are. Time permitting, you might want to do this LAN baselining before introducing the SteelHeads into the test environment.
- **LAN-Side Network** - Make sure that there are no issues with the LAN-side network between the SteelHeads and any data protection hosts. In particular, on the LAN, there should be no packet loss, and the round trip latency between the SteelHeads and hosts should be less than one millisecond for the fastest possible throughput. Interface errors, especially those related to Ethernet duplex negotiation, are a leading factors of LAN-side network issues.
- **WAN-Side Network** - Use MX-TCP to overcome any WAN-side packet loss caused by deficient links or undersized router interface queues. If the WAN bandwidth is being fully utilized during optimized data transfers, then the WAN is the bottleneck. If the WAN link is not fully utilized, options like RiOS SDR-A or SDR-M can increase the LAN-side throughput.
- **CPU** - Check the CPU reports to see if the CPU cores are the bottleneck. If some cores are busy but some are not, enable multicore load balancing. If you enable multicore load balancing and all cores are fully utilized, you might require a larger model SteelHead.
- **Disk** - You can use disk-related metrics to determine that the disk is the bottleneck for higher levels of throughput. Always assess these metrics relative to empirical application performance. Even if they indicate heavy disk utilization, it does not necessarily mean that the disk is the bottleneck. In cases where the disk is the bottleneck, then you can adjust the adaptive data streamlining settings progressively upward to either SDR-A, SDR-M or, finally, compression-only. In some cases, you might need to upgrade to a higher model SteelHead. Consult with your Riverbed Sales or Professional Services representative.
- **Data Store Disk Load** - If the RiOS Data Store Read Efficiency report, accessible from the Management Console, shows that read efficiency falls below 50% consistently, this might indicate that the disk is the bottleneck.

Third-party interoperability

Riverbed optimizes data protection utilities from many storage vendors, including but not limited to the following:

- EMC

For information about EMC qualification matrix for Riverbed Technology, see the Riverbed Knowledge Base article *Deploying SteelHeads with EMC Storage*, at <https://supportkb.riverbed.com/support/index?page=content&id=s13363>.

- NetApp

- HP

- Hitachi Data Systems (HDS)

- IBM

- Dell

- QLogic

- Symantec/Veritas

- Microsoft

- Commvault

- DoubleTake

- CA

- Compellent

- 3Par

- BlueArc

For additional information go to the following websites:

- <http://riverbed.com>

- <http://support.riverbed.com>

Alternatively, you can consult with your authorized Riverbed Solutions Provider.

Storage Area Network Replication

Storage area network (SAN) data protection deployment includes SAN replication products such as EMC Symmetrix Remote Data Facility/Asynchronous (SRDF/A), EMC RecoverPoint, IBM Global Mirror, IBM XIV replication, and Hitachi Universal Replicator. This chapter includes the following sections:

- [“Overview of SAN replication” on page 411](#)
- [“Storage optimization modules” on page 412](#)
- [“Best practices for SAN replication using TCP/IP” on page 420](#)
- [“Best practices for SAN replication using Cisco MDS FCIP” on page 421](#)

SAN replication is one of the common deployments for data protection. For information about the other options, see [“Common data protection deployments” on page 403](#).

Overview of SAN replication

In SAN replication deployments, WAN links are typically large, often ranging from T3 (45 Mbps) to OC-48 (2.5 Gbps) or more. Often SAN replication solutions require dedicated links used exclusively by the SAN replication solution.

As a best practice for high-speed SAN replication solutions, use SteelHeads that are dedicated to only optimizing high-speed SAN replication workloads and that do not optimize large amounts of general application or end-user traffic. Doing this benefits you for the following reasons:

- Increase both the level and predictability of performance delivered by SteelHeads, leading to consistent delivery of recovery point and time objectives (RPO/RT0).
- With separate SteelHeads, the large data sets commonly associated with high-speed replication do not compete for SteelHead data store resources with other user-based traffic, and the reverse.
- You can optimally tune separate SteelHeads for their respective workloads.

Disable any data compression on the SAN array (for example, EMC Symmetrix Gigabit Ethernet connectivity) and on the FCIP or iFCP gateways (for example, Cisco MDS, and QLogic), so the data enters the SteelHead in raw form. Disabling data compression allows the SteelHeads the opportunity to perform additional bandwidth reduction using RiOS SDR.

Use dedicated SteelHeads of the same model for this type of data protection scenario. Consult with your SAN vendor's customer service representative for best practice configuration of their arrays for use with SteelHeads.

For information about EMC qualification matrix for Riverbed Technology, see the Riverbed Knowledge Base article *Deploying SteelHeads with EMC Storage*, at <https://supportkb.riverbed.com/support/index?page=content&id=s13363>. To ensure a successful integration, Riverbed requires EMC involvement in any SRDF deployment.

Storage optimization modules

This section describes the storage optimization module options. This section includes the following topics:

- [“FCIP optimization module” on page 412](#)
- [“SRDF optimization module” on page 415](#)

RiOS 6.0.1 or later includes storage optimization modules for the FCIP and SRDF protocols. These modules provide enhanced data reduction capabilities. The modules use explicit knowledge of where protocol headers appear in the storage replication data stream to separate out headers from the payload data that was written to storage. In absence of a module, these headers represent an interruption to the network stream, reducing the ability of RiOS SDR to match on large, contiguous data patterns.

The modules must be configured based on the types of storage replication traffic present in the network environment. The following sections describe these options and when they would be applied.

FCIP optimization module

This section describes the storage optimization for FCIP and how to configure it. This section includes the following topics:

- [“Configuring base FCIP module” on page 412](#)
- [“Configuring FCIP module rules” on page 413](#)

The module for FCIP is appropriate for environments using storage technology that originates traffic as fibre channel (FC) and then uses a Cisco MDS gateway to convert the FC traffic to TCP for WAN transport.

For information about storage technologies that originate traffic via FC, see [“Storage Area Network Replication” on page 411](#). For configuration best-practice details for Cisco MDS deployments, see [“Best practices for SAN replication using Cisco MDS FCIP” on page 421](#).

All configuration for FCIP must be applied on the SteelHead closest to the FCIP gateway that opens the FCIP TCP connection by sending the initial SYN packet. If you are unsure which gateway initiates the SYN in your environment, we recommend that you apply the module configuration to the SteelHeads on both ends of the WAN.

Configuring base FCIP module

By default, the FCIP module is disabled. When only the base FCIP module has been enabled, all traffic on the well-known FCIP TCP destination ports 3225, 3226, 3227, and 3228 are directed through the module for enhanced FCIP header isolation. In most environments, no further FCIP module configuration is required beyond enabling the base module.

To enable the base FCIP module

1. Connect to the SteelHead CLI and enter the following command:

```
protocol fcip {enable | disable}
```

2. If an environment uses one or more nonstandard TCP ports for FCIP traffic, the module can be configured to handle traffic on additional ports by entering the following command:

```
protocol fcip ports <port-list>
```

Where <port-list> is a comma-separated list of TCP ports. Prefix this command with **no** to remove one or more TCP ports from the list of those currently directed to the FCIP module.

You can check whether the module is currently enabled or disabled, and you can determine on which TCP ports the module is looking for FCIP traffic.

To show current base FCIP module settings

- Connect to the SteelHead CLI and enter the following command:

```
show protocol fcip settings
```

The Current Connections report shows optimized connections with the *App* label for each connection shown as *FCIP*, if the base FCIP module is enabled and connections are established. If the report shows an application of a connection as *TCP*, the module is not used and you must check the configuration.

To observe the current base FCIP module connections

- Connect to the SteelHead CLI and enter the following command:

```
show connections
```

T	Source	Destination	App	Rdn	Since
O	10.12.254.2	4261 10.12.254.34	3225 FCIP	18%	2010/03/09 18:50:02
O	10.12.254.2	4262 10.12.254.34	3226 FCIP	86%	2010/03/09 18:50:02
O	10.12.254.142	4315 10.12.254.234	3225 FCIP	2%	2010/03/09 18:50:02
O	10.12.254.142	4316 10.12.254.234	3226 FCIP	86%	2010/03/09 18:50:02

Configuring FCIP module rules

An environment that has RF-originated SRDF traffic between VMAX arrays requires additional configuration beyond enabling the FCIP base module. Specifically, the SRDF protocol implementation used to replicate between two VMAX arrays uses an additional *Data Integrity Field* (DIF) header, which further interrupts the data stream. For Open Systems environments (such as Windows and UNIX/Linux), the DIF header is injected into the data stream after every 512 bytes of storage data. For IBM iSeries (AS/400) environments, the DIF header is injected after every 520 bytes. Do not add a module rule isolating DIF headers in mainframe environments, because SRDF environments that replicate mainframe traffic do not currently include DIF headers.

Note: FCIP module rules are only required for VMAX-to-VMAX traffic.

If your environment includes RF-originated SRDF traffic between VMAX arrays, the module can be configured to look for DIF headers.

To configure the FCIP module to look for DIF headers in the FCIP data stream

- Connect to the SteelHead CLI and enter the following command:

```
protocol fcip rule src-ip <ip-address> dst-ip <ip-address> dif {enable | disable} dif-blocksize  
<number-of-bytes>
```

For example, if the only FCIP traffic in your environment is RF-originated SRDF between VMAX arrays, you can allow for isolation of DIF headers on all FCIP traffic by modifying the default rule as follows:

```
protocol fcip rule src-ip 0.0.0.0 dst-ip 0.0.0.0 dif enable
```

Environments that have a mix of VMAX-to-VMAX RF-originated SRDF traffic along with other FCIP traffic require additional configuration, because SteelHeads must be informed where DIF headers are expected. This configuration is made based on IP addresses of the FCIP gateways. In such a mixed environment, SAN zoning needs to be applied to ensure that DIF and non-DIF traffic are not carried within the same FCIP tunnel.

Assume your environment consists mostly of regular, non-DIF FCIP traffic but also some RF-originated SRDF between a pair of VMAX arrays. Assume a pair of FCIP gateways are configured with a tunnel to carry the traffic between these VMAX arrays, and that the source IP address of the tunnel is 10.0.0.1 and destination IP is 10.5.5.1. The preexisting default rule tells the module not to expect DIF headers on FCIP traffic. This setting allows for correct handling of the all the non-VMAX FCIP. To obtain the desired configuration, enter the following command to override the default behavior and perform DIF header isolation on the FCIP tunnel carrying the VMAX-to-VMAX SRDF traffic:

```
protocol fcip rule src-ip 10.0.0.1 dst-ip 10.5.5.1 dif enable
```

When configured, the FCIP module looks for a DIF header after every 512 bytes of storage data, which is typical for an Open Systems environment. If your environment uses IBM iSeries (AS/400) hosts, use the **dif-blocksize** to inform the module to look for a DIF header after every 520 bytes of storage data. Enter the following command to modify the default rule to look for DIF headers on all FCIP traffic in a VMAX-based, IBM iSeries (AS/400) environment:

```
protocol fcip rule src-ip 0.0.0.0 dst-ip 0.0.0.0 dif enable dif-blocksize 520
```

To observe the current base FCIP module connections

- Connect to the SteelHead CLI and enter the following command:

```
show protocol fcip rules
```

You can display each rule currently configured, whether DIF header isolation is enabled or disabled for that rule, and how much storage data is expected before each DIF header in traffic matching that rule.

SRDF optimization module

This section describes the storage optimization for SRDF and how to configure it. This section includes the following topics:

- [“Configuring the base SRDF module” on page 415](#)
- [“Detecting Symmetrix VMAX microcode” on page 416](#)
- [“Configuring SRDF module rules” on page 417](#)
- [“Configuring SRDF selective optimization” on page 418](#)
- [“Viewing SRDF reports” on page 419](#)

The module for SRDF is appropriate for environments using EMC's Symmetrix Remote Data Facility (SRDF) with DMX and VMAX storage arrays when the traffic is originated directly from Gigabit Ethernet ports on the arrays (also referred to as *RE* ports). When in this configuration, the SRDF traffic appears on the network immediately as TCP. The SRDF protocol injects headers into the data stream; these headers interrupt the continuity of SteelHead Data Reduction (SDR). The SRDF module removes these headers from the data stream before performing data reduction, and then it reinjects them before sending data to the receiving EMC Symmetrix. In addition, the SRDF module automatically disables native EMC SRDF compression for SRDF transfers, which prevents the need for a BIN file change to disable compression. In the event of a SteelHead or network failure, the Symmetrix arrays fall back on native compression instead of transmitting at uncompressed bandwidth rates.

Note: Environments with SRDF traffic originated through Symmetrix fibre channel ports (*RF* ports) require configuration of the RiOS FCIP module, not the SRDF module. For information about RF ports, see [“FCIP optimization module” on page 412](#).

All configuration for SRDF must be applied on the SteelHead closest to the Symmetrix array that opens the SRDF TCP connection by sending the initial SYN packet. If you are unsure which array initiates the SYN in your environment, we recommend that you apply module configuration to the SteelHeads on both ends of the WAN.

Configuring the base SRDF module

By default, the SRDF module is disabled. When only the base SRDF module has been enabled, all traffic on the well-known SRDF TCP destination port 1748 is directed through the module for enhanced header isolation. In most environments using SRDF only between DMX arrays or VMAX-to-DMX, no further SRDF module configuration is required beyond enabling the base module.

To enable the base SRDF module

1. Connect to the SteelHead CLI and enter the following command:

```
protocol srdf enable
```

To disable SRDF use the **no protocol srdf enable** command.

2. If an environment used one or more nonstandard TCP ports for RE-originated SRDF traffic, the module can be configured to handle traffic on additional ports by entering the following command:

```
protocol srdf ports <port-list>
```

Where **<port-list>** is a comma-separated list of TCP ports. Use the **no** command option to remove one or more TCP ports from the list of those currently directed to the SRDF module.

You can see whether the module is currently enabled or disabled, and you can determine on which TCP ports the module is looking for SRDF traffic.

To observe current base SRDF module settings

- Connect to the SteelHead CLI and enter the following command:

```
show protocol srdf settings
```

The Current Connections report shows optimized connections with the *App* label for each connection shown as *SRDF*, if the base SRDF module is enabled and connections are established. If the report shows a connection's *App* as *TCP*, the module is not used and the configuration must be checked.

To observe the current base SRDF module connections

- Connect to the SteelHead CLI and enter the following command:

```
show connections
```

T	Source	Destination	App	Rdn	Since
O	10.12.254.80	4249 10.12.254.102	1748 SRDF	82%	2010/03/09 16:35:40
O	10.12.254.80	4303 10.12.254.202	1748 SRDF	83%	2010/03/09 16:35:40
O	10.12.254.180	4250 10.12.254.102	1748 SRDF	85%	2010/03/09 16:35:40
O	10.12.254.180	4304 10.12.254.202	1748 SRDF	86%	2010/03/09 16:35:40

Detecting Symmetrix VMAX microcode

For Symmetrix VMAX running Enginuity microcode levels newer than 5874, you do not need to configure SRDF module rules (for details, see [“Configuring SRDF selective optimization” on page 418](#)). For Symmetrix VMAX running Enginuity level 5874 or older, configure SRDF module rules as described in the [“Configuring SRDF module rules”](#) section.

To detect Symmetrix microcode level for Open Systems-connected Symmetrix, use the **symcfg** command in EMC's Solutions Enabler software. Solutions Enabler is EMC software is typically used for managing Symmetrix storage arrays. The following example shows sample output:

```
# symcfg -sid 000194900363 list -v
Symmetrix ID: 000194900363
Time Zone    : PST
Product Model      : VMAX-1SE
Symmetrix ID      : 000194900363

Microcode Version (Number) : 5875 (16F30000)
Microcode Registered Build  : 0
Microcode Date           : 11.22.2010

Microcode Patch Date      : 11.22.2010
Microcode Patch Level     : 122
```


Configuring SRDF module rules

An environment that has RE-originated SRDF traffic between VMAX arrays requires additional configuration beyond enabling the base module. Specifically, the SRDF protocol implementation used to replicate between two VMAX arrays employs an additional *Data Integrity Field* (DIF) header, which further interrupts the data stream. For Open Systems environments (such as Windows and UNIX/Linux), the DIF header is injected into the data stream after every 512 bytes of storage data. For IBM iSeries (AS/400) environments the DIF header is injected after every 520 bytes. Do not add a module rule isolating DIF headers in mainframe environments, because SRDF environments that replicate mainframe traffic do not currently include DIF headers.

Note: SRDF module rules are only required for VMAX-to-VMAX traffic.

If your environment includes RE-originated SRDF traffic between VMAX arrays, the module can be configured to look for DIF headers.

To configure the SRDF module to look for DIF headers

- Connect to the SteelHead CLI and enter the following command:

```
(config) # protocol srdf rule src-ip <ip-address> dst-ip <ip-address> dif {enable | disable}
dif-blocksize <number-of-bytes>
```

For example, if the only RE-originated SRDF traffic in your environment is between VMAX arrays, you can allow for isolation of DIF headers on all SRDF traffic by modifying the default rule as follows:

```
(config) # protocol srdf rule src-ip 0.0.0.0 dst-ip 0.0.0.0 dif enable
```

Environments that have a mix of VMAX-to-VMAX and DMX-based SRDF traffic require additional configuration, because SteelHeads must be informed where DIF headers are expected. This configuration is made based on RE port IP addresses.

Assume your environment contained RE-originated SRDF traffic mostly between DMX arrays but also some between a pair of VMAX arrays. Assume the VMAX array in the primary location had RE ports of IP addresses 10.0.0.1 and 10.0.0.2 and the VMAX array in the secondary location had RE ports of IP addresses 10.5.5.1 and 10.5.5.2. The preexisting default rule tells the module not to expect DIF headers on all RE-originated SRDF traffic. This behavior allows for correct handling of the main DMX-based SRDF traffic. To obtain the desired configuration, enter the following commands to override the default behavior and perform DIF header isolation on the VMAX SRDF connections:

```
(config) # protocol srdf rule src-ip 10.0.0.1 dst-ip 10.5.5.1 dif enable
(config) # protocol srdf rule src-ip 10.0.0.1 dst-ip 10.5.5.2 dif enable
(config) # protocol srdf rule src-ip 10.0.0.2 dst-ip 10.5.5.1 dif enable
(config) # protocol srdf rule src-ip 10.0.0.2 dst-ip 10.5.5.2 dif enable
```

When configured, the SRDF module looks for a DIF header after every 512 bytes of storage data, which is typical for an Open Systems environment. If your environment uses IBM iSeries (AS/400) hosts, rules that use the **dif-blocksize** to inform the module to look for a DIF header after every 520 bytes of storage data. Enter the following command to modify the default rule to look for DIF headers on all SRDF traffic in a

VMAX-based, IBM iSeries (AS/400) environment:

```
(config) # protocol srdf rule src-ip 0.0.0.0 dst-ip 0.0.0.0 dif enable dif-blocksize 520
```

To observe the current SRDF rule settings

- Connect to the SteelHead CLI and enter the following command:

```
show protocol srdf rules
```

This command displays each rule currently configured, whether DIF header isolation is enabled or disabled for that rule, and how much storage data is expected before each DIF header in traffic matching that rule.

Configuring SRDF selective optimization

RiOS 6.1.2 or later feature *selective optimization*. Selective optimization provides different types of optimization to different RDF Groups, and it allows you to tune for the best optimization setting for each RDF group to maximize the SteelHead usability. Selective optimization also depends on Symmetrix VMAX Engenuity microcode levels newer than 5874.

Consider an example with three types of data:

- Oracle logs (RDF group 1)
- Encrypted check images (RDF group 2)
- Virtual machine images (RDF group 3)

In this example, assign LZ-only compression to the Oracle logs, no optimization to the encrypted check images, and default SDR to the virtual machine images. To assign these levels of optimization, configure the SteelHead to associate specific RE port IP addresses with specific Symmetrix arrays, and then assign rules to specific RDF groups for different optimization policies.

To configure the SteelHead to associate RE ports with a specific Symmetrix

1. Connect to the SteelHead CLI and enter the following command:

```
(config) # protocol srdf symm id <symm-id> address <ip-address>
```

The Symmetrix ID is an alphanumeric string that can contain hyphens and underscores (for example, a standard Symmetrix serial number is 000194900363). Do not use spaces or special characters.

2. Add a rule to affect traffic coming from the RE ports associated with this SYMMID:

```
(config) # protocol srdf symm id <symm-id> rdf_group <rdf-group> optimization <opt-policy>
[description *]
```

OPT_POLICY is one of: none, lz-only, or sdr-default.

RDF_GROUP in RiOS is specified as a decimal number; however some EMC utilities report RDF group numbers in hexadecimal.

When you have Symmetrix arrays serving Open Systems hosts, and you are using EMC Solutions Enabler, RDF group numbers are reported in decimal—ranging from 1 to 255. By default, this is how RDF_GROUP is entered in RiOS and shown in reports. For mainframe-attached Symmetrix arrays, tools report RDF group numbers in hexadecimal, starting from 0. Use the following command for SteelHeads serving only mainframe-attached Symmetrix arrays, and you want the representation of RDF_GROUP to be in the range 0 to 254:

```
(config) # protocol srdf symm id base_rdf_group 0
```

When you have three RDF groups (assuming a SYMMID of 123), enter the following commands:

```
(config) # protocol srdf symm id 123 rdf group 1 optimization lz-only description Oracle1_DB
(config) # protocol srdf symm id 123 rdf group 2 optimization none description Checkimages
(config) # protocol srdf symm id 123 rdf group 3 optimization sdr-default description VMimages
```

The following example shows sample output:

```
(config) # show protocol srdf symm stats
Time          SYMM RDF group opt policy Reduction LAN Mbps WAN Mbps LAN KB  WAN KB description
-----
10/2/2010 10:14:49 0123 1      lz-only      68%      222      71.04 222,146 71,040 Oracle1_DB
10/2/2010 10:14:49 0123 2      none         0%       79       79     79,462 79,462 Checkimages
10/2/2010 10:14:49 0123 3      sdr-default  94%      299      17.94 299,008 17,943 VMimages
```

Note: Data reduction is highest for RDF Group 3, which is treated with default SDR.

For more information about commands that show the current SRDF configuration (**show protocol srdf symm id**) or show reduction statistics for all or specific SYMMIDs (**show protocol srdf symm stats**), see the *Riverbed Command-Line Interface Reference Manual*.

To fine-tune SRDF optimization on a per-RDF-group basis

1. Check the level of data reduction currently achieved on each RDF group with the **show protocol srdf symm stats** command.
2. For RDF groups achieving low data reduction (for example, less than 20%), change the optimization policy to LZ-only.
3. For RDF groups achieving no data reduction (0%), first check to determine whether the RDF groups contain information that is intentionally encrypted. If so, change the optimization policy to none. If not, we recommend investigating whether source encryption can be disabled.

Viewing SRDF reports

In RiOS 7.0 or later, you can report SRDF statistics on a per-RDF-group basis. The following command displays statistics for all RDF groups being optimized:

```
(config) # show stats protocol srdf [interval <interval>] | [start-time <date> end-time <date>]
```

This command shows statistics from a specific Symmetrix machine (indicated by `symm_id`):

```
(config) # show stats protocol srdf symm id <symm-id> [interval <interval>] | [start-time <date> end-time <date>]
```

This command shows statistics from a specific RDF group (indicated by `rdf_group`):

```
(config) # show stats protocol srdf symm id <symm-id> rdf-group <rdf-group> [interval <interval>] | [start-time <date> end-time <date>]
```

To view these reports, open the Management Console, and choose Reports > Optimization: SRDF.

Best practices for SAN replication using TCP/IP

Many SAN arrays support replication using direct connectivity via TCP/IP. In this case, SteelHeads optimize connections that are initiated directly between the SAN arrays participating in the replication. The following table shows a best practice configuration running RiOS 8.0 or later with TCP/IP connectivity directly from storage array.

Feature	Best practice
Multicore Balancing	<p>Continuous code improvements to the performance of single connections has made it unnecessary to enable multicore balancing in RiOS 8.0 or later when 8 or more connections are used. Enabling multicore balancing in these conditions will cause a lower throughput.</p> <p>If multicore balancing is required, use the datastore codec multi-core-bal CLI command.</p>
Enable MX-TCP class covering replication traffic.	See “Configuring QoS and MX-TCP” on page 165 to enable and configure MX-TCP
Set WAN TCP buffers	<p>Enter these CLI commands:</p> <p>protocol connection wan receive def-buf-size <2*BDP></p> <p>protocol connection wan send def-buf-size <2*BDP></p>
Set LAN TCP buffers	<p>Enter these CLI commands:</p> <p>protocol connection lan send buf-size 1048576</p> <p>tcp adv-win-scale -1</p> <p>Note: The tcp adv-win-scale -1 command is for RiOS 5.5.6c or later.</p>
Reset existing connections on start up	<p>Enter the following CLI commands:</p> <p>in-path kickoff</p> <p>in-path kickoff-resume</p> <p>Note: The in-path kickoff-resume command is for RiOS 6.0.1a or later.</p>
Never pass-through SYN packets	Enter the in-path always-probe enable CLI command.
SRDF/A optimization	Enter the protocol srdf enable CLI command.
Note: Use only with SRDF/A Replication and RiOS 6.0.1 or later.	
VMAX DIF header optimization	<p>Enter the protocol srdf rule src-ip <x.x.x.x> dst-ip <y.y.y.y> dif enable CLI command.</p> <p>Replace <x.x.x.x> and <y.y.y.y> with IP address pairs for RE ports. For details, see “Storage optimization modules” on page 412.</p> <p>Note: Use only with EMC VMAX and RiOS 6.0.1 or later.</p>
Restart the optimization service	Enter the restart CLI command.

Best practices for SAN replication using Cisco MDS FCIP

This section describes the key concepts and recommended settings in the MDS. This section includes the following topics:

- [“FCIP profiles” on page 421](#)
- [“FCIP tunnels ” on page 422](#)
- [“Configuring a Cisco MDS FCIP deployment” on page 422](#)
- [“Best practices for RiOS 5.5.3 and later with Cisco MDS FCIP configuration” on page 423](#)

FCIP profiles

An *FCIP profile* defines characteristics of FCIP tunnels that are defined through a particular MDS Gigabit Ethernet interface. Profile characteristics include the:

- IP address of the MDS Gigabit Ethernet interface that is originating the tunnel.
- TCP port number.
- bandwidth and latency characteristics of the WAN link.
- advanced settings that are typically left to their default values.

The MDS enables you to define up to three FCIP profiles per physical MDS Gigabit Ethernet interface. Because a tunnel can be created for each profile, a Cisco MDS switch with two physical Gigabit Ethernet ports can have up to six profiles. Most configurations have only one profile per Gigabit Ethernet interface. We recommend maximizing the number of profiles configured for each GigE port to increase the total number of TCP connections.

In the profile setting, the default maximum and minimum bandwidth settings per FCIP profile are 1000 Mbps and 500 Mbps, respectively. You can achieve better performance for unoptimized and optimized traffic using 1000 Mbps and 800 Mbps. These bandwidth setting is the rate of the LAN-side TCP entering the SteelHead, so that setting it aggressively high does not have any downside, because the SteelHead terminates TCP locally on the LAN side and the MDS can slow down if it tries to go too fast by advertising a smaller TCP window.

Similarly, leave the round-trip setting at its default (1000 ms in the Management Console, 1 ms in the CLI), because the *network* in this context is effectively the LAN connection between the MDS and the SteelHead.

If you are doing *unoptimized runs*, configure the bandwidth and latency settings in the MDS to reflect the actual network conditions of the WAN link. These settings improve performance in terms of enabling the MDS to fill-the-pipe with unoptimized runs in the presence of latency.

FCIP tunnels

An FCIP *tunnel* configuration is attached to a profile and defines the IP address and TCP port number of a far-side MDS to which an FCIP connection is established. You can keep the tunnel configuration default settings, with the following key exceptions:

- In the Advanced tab of the MDS GUI:
 - Turn on the Write Accelerator option. Always use this option when testing with SteelHeads in the presence of latency. This is an optimization in the MDS (and similar features exist in other FCIP/iFCP products) to reduce round trips.
 - Set the FCIP configuration for each tunnel to Passive on one of the MDS switches. By default, when first establishing FCIP connectivity, each MDS normally tries to constantly initiate new connections in both directions, and it is difficult to determine which side ends up with the well-known destination port (for example, 3225). This behavior can make it difficult to interpret SteelHead reports. When you set one side to Passive, the nonpassive side always initiates connections, hence the behavior is deterministic.

FCIP settings allow you to specify the number of TCP connections associated with each FCIP tunnel. By default, this setting is 2: one for Control traffic, and one for the Data traffic. Do not change the default value. The single-TCP mode only exists to maintain compatibility with older FCIP implementations. Separating the Control and Data traffic has performance implications because FC is highly jitter sensitive.

You can set whether the MDS compresses the FCIP data within the FCIP tunnel configuration. You must disable it when the SteelHead is optimizing. On the MDS the default setting is off. The best practices of common SAN replication vendors (for example, EMC) recommend turning on this setting when there are no WAN optimization controller (WOC) systems present. However, when adding SteelHeads to an existing environment, it should be disabled.

Configuring a Cisco MDS FCIP deployment

The following example shows a Cisco MDS FCIP gateway configuration. Cisco-style configurations typically do not show the default values (for example, compression is off by default, and it is not present in this configuration dump). Also, this configuration does not show any non-FCIP elements (such as the FC ports that connect to the SAN storage array and VSANs). This example shows a standard and basic topology that includes an MDS FCIP gateway at each end of a WAN link, MDS1, and MDS2.

To configure a standard and basic topology that includes an MDS FCIP gateway

1. Configure MDS1.

```
fcip profile 1
  ip address 10.12.254.15
  tcp max-bandwidth-mbps 1000 min-available-bandwidth-mbps 800 round-trip-time-ms 1
fcip profile 2
  ip address 10.12.254.145
  tcp max-bandwidth-mbps 1000 min-available-bandwidth-mbps 800 round-trip-time-ms 1
interface fcip1
  use-profile 1
  peer-info ipaddr 10.12.254.45
  write-accelerator
  no shutdown
interface fcip2
```

```

use-profile 2
peer-info ipaddr 10.12.254.245
write-accelerator
no shutdown
ip route 10.12.254.32 255.255.255.224 10.12.254.30
ip route 10.12.254.224 255.255.255.224 10.12.254.130
interface GigabitEthernet1/1
ip address 10.12.254.15 255.255.255.224
switchport description LAN side of mv-emcsh1
no shutdown
interface GigabitEthernet1/2
ip address 10.12.254.145 255.255.255.224
switchport description LAN side of mv-emcsh1
no shutdown

```

2. Configure MDS2.

```

fcip profile 1
ip address 10.12.254.45
tcp max-bandwidth-mbps 1000 min-available-bandwidth-mbps 800 round-trip-time-ms 1
fcip profile 2
ip address 10.12.254.245
tcp max-bandwidth-mbps 1000 min-available-bandwidth-mbps 800 round-trip-time-ms 1
interface fcip1
use-profile 1
passive-mode
peer-info ipaddr 10.12.254.15
write-accelerator
no shutdown
interface fcip2
use-profile 2
passive-mode
peer-info ipaddr 10.12.254.145
write-accelerator
no shutdown
ip route 10.12.254.0 255.255.255.224 10.12.254.60
ip route 10.12.254.128 255.255.255.224 10.12.254.230
interface GigabitEthernet1/1
ip address 10.12.254.45 255.255.255.224
switchport description LAN side of mv-emcsh2
no shutdown
interface GigabitEthernet1/2
ip address 10.12.254.245 255.255.255.224
switchport description LAN side of mv-emcsh2
no shutdown

```

Best practices for RiOS 5.5.3 and later with Cisco MDS FCIP configuration

We recommend the following best practices regarding a Cisco MDS FCIP configuration:

- Enable the RiOS 5.5 or later multicore balancing feature due to the small number of data connections.
- Use an in-path rule to specify the neural-mode as *never* for FCIP traffic.
- Set the *always-probe* port to 3225 to ensure that MDS aggressive SYN-sending behavior does not result in unwanted pass-through connections.

The following table summarizes the CLI commands RiOS 5.5.3 or later with Cisco MDS FCIP.

Feature	CLI command
Multicore Balancing	Continuous code improvements to the performance of single connections has made it unnecessary to enable multicore balancing in RiOS 8.0 or later when 8 or more connections are used. Enabling multicore balancing in these conditions will cause a lower throughput. If multicore balancing is required, use the datastore codec multi-core-bal CLI command.
Turn Off Nagle	Use the following CLI command: in-path rule auto-discover srcaddr all-ip dstaddr all-ip dstport "3225" preoptimization "none" optimization "normal" latency-opt "normal" vlan -1 neural-mode "never" wan-visibility "correct" description "" rulenum start
MX-TCP class covering FCIP traffic	See “Configuring QoS and MX-TCP” on page 165 to enable and configure MX-TCP on SteelHeads.
Set WAN TCP buffers	Enter the following CLI commands: protocol connection wan receive def-buf-size <2*BDP> protocol connection wan send def-buf-size <2*BDP>
Set LAN TCP buffers	protocol connection lan send buf-size 1048576 tcp adv-win-scale -1 Note: tcp adv-win-scale -1 is for RiOS 5.5.6c or later.
Reset existing connections on startup	in-path kickoff in-path kickoff-resume Note: in-path kickoff-resume is for RiOS 6.0.1a or later.
Never pass-through SYN packets	in-path always-probe enable
Change always-probe port to FCIP	in-path always-probe port 3225
FCIP optimization	protocol fcip enable Note: Use only with RiOS 6.0.1 or later.
DIF header optimization	protocol fcip rule src-ip <x.x.x.x> dst-ip <y.y.y.y> dif enable Replace <x.x.x.x> and <y.y.y.y> with IP address pairs for MDS Gigabit Ethernet ports. For details, see “Storage optimization modules” on page 412 . Note: Use only with EMC VMAX and RiOS 6.0.1 or later.
Restart the optimization service	restart

If you increase the number of FCIP profiles, you must also create separate in-path rules to disable Nagle for other TCP ports (for example, 3226 and 3227).

Similarly, if you decide to set QoS rules to focus on port 3225 to drive traffic into a particular class, you must create rules for both ports 3226 and 3227.

Authentication, Security, Operations, and Monitoring

This chapter describes how to configure RADIUS, TACACS+, or SAML authentication for the SteelHead, including best practices for securing the SteelHead, and provides information about operations and flow data monitoring.

This chapter includes the following sections:

- [“Overview of secure transport” on page 425](#)
- [“Overview of authentication” on page 426](#)
- [“Authentication features” on page 427](#)
- [“Configuring SAML” on page 428](#)
- [“Configuring a TACACS+ server” on page 432](#)
- [“Securing SteelHeads” on page 433](#)
- [“Changing encryption for domain replication passwords” on page 446](#)
- [“REST API access” on page 446](#)
- [“Capacity planning” on page 446](#)
- [“Overview of exporting flow data” on page 450](#)
- [“SNMP monitoring” on page 451](#)
- [“Configuring SNMPv3 authentication and privacy” on page 457](#)

Overview of secure transport

Today’s enterprises are embracing hybrid networking. Key features of the hybrid network are leveraging multiple paths between sites to achieve diversity across a variety of networks and to deliver application traffic over the most efficient path. However, as new network models are built, the security of traffic on the path is a concern.

RiOS 9.0 working together with SCC 9.0 introduces the secure transport feature. Secure transport is integrated with path selection and provides a way to configure and enable encryption services for traffic over a path. Functionality is separated into the management plane, control plane, and data plane. You configure the management plane on the SCC, in which you also configure and distribute the path selection policy.

When a network is marked as securable (Networking > Network Services: Sites & Networks page), it indicates that the SteelHead will join a group of other SteelHeads that can encrypt traffic. The SCC pushes that policy to all the SteelHeads, and you can also select one of the available SteelHeads to be the secure transport controller (the controller) for the control plane. Over the control plane, the SteelHead selected as the controller communicates with the SteelHeads in the group and coordinates the encryption keys to use over the data plane. Traffic is secured on the data plane for the path marked as securable. The SteelHeads use the encryption keys received from the controller and the Path Selection policy received from the SCC.

Secure transport uses standards-based IPSec with the highest level of commercially available security based on AES-256 and SHA-2 to secure traffic over a path. The control plane between the SteelHead acting as controller and other SteelHeads performing encryption is secured using SSL over TCP port 9443. The management plane from the SCC is secured using SSL and SSH.

One key advantage of secure transport is that management of encryption services is done centrally through the SCC, which serves as the user interface for configuration of secure transport.

For further configuration details, see the *SteelCentral Controller for SteelHead User Guide* and the *SteelCentral Controller for SteelHead Deployment Guide*.

Note: In RiOS 9.0, IPSec secure peering and the secure transport service are mutually exclusive. The secure transport service is enabled by default. Before you enable IPSec secure peering, you must disable the secure transport service. Also, SSL secure peering and secure transport traffic can coexist.

Overview of authentication

You can log in to a SteelHead with a RADIUS, TACACS+, or SAML authentication system for administrative and monitoring purposes. The following methods for user authentication are provided with the SteelHead:

- Local
- SAML
- RADIUS
- TACACS+

For information about per-command authorization and per-command accounting, see the *Riverbed Command-Line Interface Reference Manual*.

The order in which authentication is attempted is based on the order specified in the AAA method list. The authentication list provides backup authentication methods in case one method fails to authenticate the server. If the first server is unavailable, the next server in the list is contacted depending on your settings.

If there are multiple servers within a method (assuming the method is contacting authentication servers) and a server time-out is encountered, the next server in the list is tried. If the current server being contacted issues an authentication reject, another server is contacted according to your settings. If none of the methods validate a user, the user is not allowed access to the server.

The SteelHead does not have the ability to set a per interface authentication policy. The same default authentication method list is used for all interfaces. You cannot configure authentication methods with subsets of the servers specified (that is, there are no server groups).

For information about Windows domain authentication for encrypted MAPI and SMB signed CIFS traffic see the *SteelHead Deployment Guide - Protocols*.

Authentication features

RiOS 5.0.x and later support the following features (available only through the CLI):

- **Per-command authorization** - Your TACACS+ server can authorize all CLI commands with the **aaa authorization per-command default** command. The methods available for per-command authorization are local (default) and TACACS+.

To use TACACS+ for per-command authorization, configure the SteelHead for TACACS+ and define the users and commands authorized to run on your TACACS+ server. For information about how to configure your TACACS+ server, see the TACACS+ server documentation.

Per-command authorization applies to the CLI only.

If you do not have a TACACS+ server, use role-based accounts locally on the SteelHead to limit the Management Console and CLI commands available to users.

For more information about configuring TACACS+ on the SteelHead, see [“Configuring a TACACS+ server” on page 432](#). For details, see how to restrict user roles on [“Best practices for securing access to SteelHeads” on page 433](#).

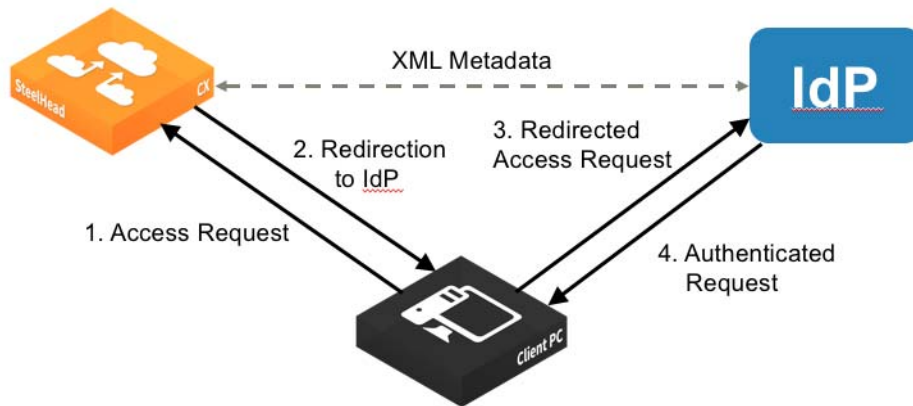
- **Per-command accounting** - You always enable per-command accounting locally. You must specifically enable the command for TACACS+ by defining the TACACS+ method using the **aaa accounting per-command default** command. TACACS+ per-command accounting is always sent to all the configured TACACS+ servers. The local method logs the command in the system logs.
- **TACACS+ server first hit** - When the server first hit CLI command (**tacacs-server first-hit**) is enabled, the SteelHead rejects authentication after the first rejection received from a TACACS+ server rather than continuing through all the TACACS+ servers in the list. This feature applies to user authentication and per-command authorization.
- **Fallback** - The *fallback* option decides how the successive authentication methods are tried. When you enable fallback, if authentication fails, the system continues through all authentication methods (TACACS+, RADIUS, local) in the order you configure them in the authentication method list. Fallback is enabled by default. When you enable conditional fallback (**aaa authentication cond-fallback**) you can configure the system to only proceed beyond TACACS+ or RADIUS if the servers are unreachable. Conditional fallback enables you to reject the login once the first method rejects the attempt, instead of proceeding to the next method in the authentication method list.
- **Remote and console method lists** - There are two method lists: remote (**ssh**, web UI) and console (serial, terminal, SteelHead, telnet). The console method requires a local method to be present, but the remote list does not. You enable the remote method using the **aaa authentication login default** command. You enable the console method using the **aaa authentication console-login default** command.

Configuring SAML

Security Assertion Markup Language (SAML) 2.0 is an XML standard that acts as an authentication interface between a SteelHead and an identity provider (IdP). You can use the IdP to provide additional requirements for authentication, such as a multifactor authentication based on a common access card (CAC) or personal identity verification (PIV) or any other authentication method.

When a SteelHead receives a login request, it determines if SAML is enabled. If SAML is enabled, user authentication through AAA is disabled and the SteelHead redirects the authentication request to the IdP. The IdP authenticates the user and redirects the user to the SteelHead, which allows access.

Figure 20-1. SAML authentication process



To enable IdP authentication, you configure the SteelHead and the IdP with XML metadata that provides detailed appliance identification. The metadata also establishes a trust relationship between the SteelHead and the IdP.

Administrators must add users to the IdP server to provide them login access, and those users need to correspond to SteelHead users. You can have one-to-one mapping of users between IdP and SteelHead, or you can have multiple users on IdP map to single account on the SteelHead, such as the admin account. (You have to create individual user accounts on the SteelHead for one-to-one mapping as the user accounts determine the access permissions.)

If a user who has not been set up in the IdP tries to log in to the SteelHead, the login fails on the IdP login page. (This failed login is not tracked in the SteelHead logs.) If the user has been set up but their user mapping has not been defined in the IdP, the login succeeds but the SteelHead displays an error page (instead of the dashboard).

Note: SAML authentications are only available in the Management Console web interface; they are not available through the CLI. Users can log in to a SAML-enabled SteelHead through the CLI but they are authenticated using the local, RADIUS, or TACACS+ authentication methods.

If you cannot log in using SAML (for example, if the IdP server is unavailable), you can log in through the CLI and disable SAML using the **no aaa saml** command. Once SAML is disabled, you revert to the previously configured authentication method for the web interface. For command details, see the *Riverbed Command-Line Interface Reference Manual*.

You must be logged in as the administrator to enable or disable SAML. For details on configuring SAML, see the *SteelHead User Guide*.

Configuring a RADIUS server

This section describes how to configure a RADIUS server for the SteelHead. This section includes the following topics:

- “Configuring a RADIUS server with FreeRADIUS” on page 429
- “Configuring RADIUS authentication in the SteelHead” on page 430
- “Configuring RADIUS CHAP authentication” on page 431

Configuring a RADIUS server with FreeRADIUS

On a per-user basis, you can specify a different local account mapping by using a vendor-specific attribute. This section describes how to configure the FreeRADIUS server to return an attribute (which specifies the local user account as an ASCII string). The file paths are the default values. If the RADIUS server installation has been customized, the paths might differ.

Dictionary files are stored in the directory `/usr/local/share/freeradius`. You can define RADIUS attributes in this directory. Assuming the vendor does not have an established dictionary file in the FreeRADIUS distribution, begin the process by creating a file called **dictionary.<vendor>** in this directory.

The contents of the **dictionary.<vendor>** file define a vendor identifier (which should be the Structure of Management Information [SMI] Network Management Private Enterprise Code of the Vendor) and any vendor-specific attributes.

In the following example, the Vendor Enterprise Number for Riverbed is **17163** and the Enterprise Local User Name Attribute is 1. These numbers specify that a given user is an **admin** or **monitor** user in the RADIUS server (instead of using the SteelHead default for users not named **admin** and **monitor**).

These instructions assume you are running FreeRADIUS v.1.0, which is available from <http://www.freeradius.org>. You can also find more details in the *SteelHead User Guide*.

To install FreeRADIUS on a Linux computer

1. Download FreeRADIUS from <http://www.freeradius.org>.
2. At your system prompt, enter the following commands:

```
tar xvzf freeradius-$VERSION.tar.gz
cd freeradius-$VERSION
./configure
make
make install #as root
```

To add acceptance requests on the RADIUS server

1. In a text editor, open the `/usr/local/etc/raddb/clients.conf` file.
2. To create the key for the RADIUS server, add the following text to the **clients.conf** file:

```
client 10.0.0.0/16 {
    secret = testradius
    shortname = main-network
    nastype = other
}
```

```
}
```

The secret you specify here must also be specified in the SteelHead when you set up RADIUS server support.

3. In a text editor, create a `/usr/local/share/freeradius/dictionary.rbt` file for Riverbed.

4. Add the following text to the `dictionary.rbt` file.

```
VENDOR      RBT      17163
ATTRIBUTE   Local-User 1      string      RBT
```

5. Add the following line to the `/usr/local/share/freeradius/dictionary`:

```
$INCLUDE dictionary.rbt
```

6. Add users to the RADIUS server by editing the `/usr/local/etc/raddb/users` file, for example:

```
"admin"      Auth-Type := Local, User-Password == "radadmin"
              Reply-Message = "Hello, %u"
"monitor"    Auth-Type := Local, User-Password == "radmonitor"
              Reply-Message = "Hello, %u"
"raduser"    Auth-Type := Local, User-Password == "radpass"
              Local-User = "monitor", Reply-Message = "Hello, %u"
```

7. Start the server using `/usr/local/sbin/radiusd`. Use the `-X` option if you want to debug the server.

Note: The `raduser` is the monitor user as specified by Local, User-Password.

Configuring RADIUS authentication in the SteelHead

This section describes the basic steps for configuring RADIUS authentication in the SteelHead. For details, see the *SteelHead Installation and Configuration Guide* and the *SteelHead User Guide*.

You prioritize RADIUS authentication methods for the system and set the authorization policy and default user.

Note: Put the authentication methods in the order in which you want authentication to occur. If authorization fails on the first method, the next method is attempted, and the order is continued until all the methods have been attempted.

Perform the following basic steps to configure RADIUS support.

To configure RADIUS support

1. Add the IP address of the RADIUS server and specify the key used when you added the device to the ACS server:

```
(config)# radius-server host 192.168.1.200 key rvbd
```

2. Enable AAA.

3. Define the authentication method.

The following configuration attempts to use RADIUS and then local:

```
(config)# aaa authentication login default radius local
```

Configuring RADIUS CHAP authentication

In RiOS 8.0 and later, you can configure RADIUS CHAP authentication through the CLI or, in RiOS 8.5 and later, the SteelHead Management Console. Choose which method to use based on the appropriate risk mitigation strategy provided by either option. For example, CHAP transmits the password in a more secure manner, but various RADIUS servers can store the password in an unencrypted format.

```
radius-server host 192.168.198.136 auth-type chap timeout 3 retransmit 1 key testradius
```

To configure CHAP authentication in the SteelHead Management Console, configure the RADIUS server (Administration > Security: RADIUS).

Figure 20-2. RADIUS CHAP authentication

RADIUS Servers:

▼ Add a RADIUS Server ✕ Remove Selected

Hostname or IP Address:

Authentication Port:

Authentication Type: ☐ PAP ☒ CHAP

☒ Override the Global Default Key

Server Key:

Confirm Server Key:

Timeout (seconds): (1 - 60)

Retries: (0 - 5)

☒ Enabled

Add

After you add the server, include RADIUS in the order of authentication methods. A best practice to ensure that you can still perform authentication in the absence of the RADIUS server is to:

- use the RADIUS server first for authentication, but
- fall back to the SteelHead username and password database if the RADIUS server is unavailable.

Figure 20-3. RADIUS authentication

General Settings Security > General Settings ?

Authentication Methods

▼

☒ For RADIUS/TACACS+, fallback only when servers are unavailable

Authorization Policy: ▼

Apply

Configuring a TACACS+ server

This section describes how to configure a TACACS+ server for the SteelHead. This section includes the following topics:

- “Configuring TACACS+ with Cisco Secure Access Control Servers” on page 432
- “Configuring TACACS+ authentication in the SteelHead” on page 432

Configuring TACACS+ with Cisco Secure Access Control Servers

This task requires that you are running a Cisco Secure Access Control Server (ACS) and you want to configure it for TACACS+.

The TACACS+ Local User Service is **rbt-exec**. The Local User Name Attribute is **local-user-name**. This attribute controls whether a user who is not named **admin** or **monitor** is an administrator or monitor user (instead of using the SteelHead default value). For the SteelHead, the users listed in the TACACS+ server must have PAP authentication enabled.

Use the following procedures to configure TACACS+ with Cisco Secure ACS.

- To configure TACACS+ with Cisco ACS 4.x, go to <http://supportkb.riverbed.com/support/index?page=content&id=S14831>.
- To configure TACACS+ with Cisco ACS 5.x, go to <http://supportkb.riverbed.com/support/index?page=content&id=S16158>.

Configuring TACACS+ authentication in the SteelHead

This section describes the basic steps for configuring TACACS+ authentication in the SteelHead. You prioritize TACACS+ authentication methods for the system and set the authorization policy and default user.

For more information and detailed procedures, see the *SteelHead Installation and Configuration Guide* and the *SteelHead User Guide*.

Note: Put the authentication methods in the order in which you want authentication to occur. If authorization fails on the first method, the next method is attempted, and the order is continued until all the methods have been attempted.

Perform the following basic steps to configure TACACS+ support.

To configure TACACS+ support

1. Add the IP address of the ACS server and specify the key used when you added the device to the ACS server.

```
(config)# tacacs-server host 192.168.1.200 key rvbd
```

2. Enable AAA.
3. Define the authentication method.

The following configuration attempts to use TACACS+ and then local:


```
(config)# aaa authentication login default tacacs+ local
```

Securing SteelHeads

This section describes security features you can use to harden your network, including ways to secure the SteelHeads and some common sense security policies. This section includes the following topics:

- [“Overview of securing SteelHeads” on page 433](#)
- [“Best practices for securing access to SteelHeads” on page 433](#)
- [“Best practices for enabling SteelHead security features” on page 440](#)
- [“Best practices for policy controls” on page 444](#)
- [“Best practices for security monitoring” on page 444](#)
- [“Configuring SSL certificates for web user interface” on page 445](#)

Overview of securing SteelHeads

Organizations have typically focused attention on securing their networks by providing security for and preventing attacks against hosts. Unfortunately, there are also many security risks associated with networking devices. Attacks against such devices can be used to gather valuable information. For example, an attacker could use tools to fill up the MAC address tables of Ethernet switches, causing the switches to flood packets. These packets might contain passwords that can easily be captured.

The SteelHead has been certified and subsequently deployed for internal use by several highly security-conscious organizations, including military, government, and financial organizations. However, SteelHeads are complex network-facing systems and must be treated accordingly.

Note: Because security requirements vary by organization, consider these recommendations with your particular security goals in mind. Before implementing any security measure described in this section, you must have a thorough understanding of its impact. For example, you do not want to disable access to a SteelHead by mistake and not be able to undo the change because you inadvertently blocked your own access.

If you have a specific security concern, we recommend that you consult with Riverbed Professional Services.

Best practices for securing access to SteelHeads

This section describes best practices for securing access to your SteelHeads. These practices are not requirements, but we recommend that you consider these suggestions as implementing them can enforce a secure deployment:

- **Restrict physical access** - You must restrict physical access to any network device. An unauthorized user can easily gain access to a SteelHead if that person has physical access. Every device has the ability to recover lost passwords. By acquiring physical access to a device, an attacker can gain control by using the lost password recovery procedures. Even without breaking into the SteelHead software, it is possible to gain access to the contents of disks by gaining access to the SteelHead itself. You should treat the SteelHead as comparable in value to the servers or clients that hold sensitive data. For example, if servers are in locked rooms, we recommend the SteelHeads also be in those locked rooms.

Someone with physical access could also remove the SteelHead without authorization, allowing an attacker to gain access to confidential data. In general, SteelHeads are less valuable to an attacker than application servers or file servers because of an intrinsic scrambling of the RiOS data store. SteelHeads also support encryption of the RiOS data store to further reduce the likelihood of a successful attack, and SteelCentral Controller for SteelHead Mobile likewise allows the use of file encryption for the RiOS data store on a Windows PC.

Physical access increases the susceptibility of the networking device to denial-of-service attacks. A disgruntled employee could conceivably power the SteelHead down, disarrange the cabling, swap hard drives, or even steal the SteelHead.

- **Use an appropriate login message** - The login message appears on the Management Console Home page. You must display a login message that reinforces your organization access and security policies. Have your organization legal council approve an appropriate login message.

Typical login messages include, but are not limited to:

- statements pertaining to authorized access only
- consequences of unauthorized access
- elimination of right to privacy
- acknowledgment that the user might be monitored

The default login message is "Welcome to the Management Console for SteelHead_name!" You can change this message by navigating to the Administration > System Settings: Announcements page in the Management Console and specifying another message. You can also use the CLI as shown in the following example.

Syntax:

```
[no] banner login <message-string>
```

Example:

```
banner login. "This computer system is the property of Company XYZ Inc. Disconnect NOW if you
have not been expressly authorized to use this system. Unauthorized use is a criminal offence
under the Computer Misuse Act 1990.
Communications on or through Company XYZ Inc. computer systems may be monitored or recorded to
secure effective system operation and for other lawful purposes."
```

- **Allow management only from the primary interface** - Limiting SSH and HTTPS access to the Primary interface allows administrators to restrict who can access the SteelHeads by the use of filters or Access Control Lists. These filters are typically based on the source IP addresses of hosts and are applied on network devices like routers, Layer-3 switches, or firewalls. Limiting remote management access to SteelHeads helps prevent unauthorized user access.

Syntax:

```
[no] web httpd listen enable
[no] web httpd listen interface <interface>
[no] ssh server listen enable
[no] ssh server listen interface <interface>
```

Example:

```
web httpd listen enable
web httpd listen interface primary
ssh server listen enable
ssh server listen interface primary
```

- **Use SSH version 2** - SSH version 2 is more secure than previous versions of SSH. The major differences between SSH1 and SSH2 fall into two main categories: technical and licensing. SSH2 uses different encryption and authentication algorithms.

SSH1 offers four encryption algorithms (DES, 3DES, IDEA, and Blowfish); however, SSH2 dropped support for DES and IDEA but added three algorithms. SSH1 also uses the RSA authentication algorithm; however, SSH2 uses the Digital Signature Algorithm (DSA). These changes were designed to increase the base level of security in SSH2 by using stronger algorithms.

Syntax:

```
[no] ssh server v2-only enable
```

Example:

```
ssh server v2-only enable
```

- **Disable unencrypted communication protocols such as Telnet and HTTP** - An attacker can easily gain access to usernames and passwords by sniffing network communications. You might consider a switched Ethernet environment secure because packets are only forwarded out ports based on the destination MAC address; however, this is not necessarily the case.

Several hacking tools are available that can generate large amounts of bogus MAC addresses. These packets flood the switch's MAC address table in an attempt to overflow the table. A switch typically floods packets out all ports if it does not have an entry in its MAC address table. Therefore, after the MAC address table for the switch is filled, the switch floods packets out all ports.

The attacker can now use a packet-capturing application to capture the flooded packets and can then look for remote management connections. After the attacker discovers remote management connections, the attacker can reset those TCP connections, causing the user to log in again allowing the attacker to capture the username and password.

If only HTTPS and SSH are used, the attacker cannot obtain the usernames and passwords because they are encrypted.

Syntax:

```
[no] telnet-server enable  
[no] web http enable
```

Example:

```
no telnet-server enable  
no web http enable
```

In RiOS 8.5 and later, you can configure the SteelHead to redirect HTTP access to HTTPS access with the **web http redirect** command. We recommend this method unless your security policy requires the SteelHead to not listen to HTTP.

- **Use TLS only for the Management Console** - Only permit TLS between the browser and the Management Console.

Syntax:

```
web ssl protocol tlsv1  
no web ssl protocol sslv3  
web ssl protocol tlsv1.1  
web ssl protocol tlsv1.2
```

- **Restrict user roles** - Be sure to restrict the roles of users. For example, if a Help desk administrator is supposed to only view statistics and generate reports, their account restricts them to those roles.

Syntax:

```
[no] rbm user <username> role <role> permissions <permissions>
[no] rbm role <role> primitive <primitive>
```

Refer to the *Riverbed Command-Line Interface Reference Manual* for more information about this command.

- **Remove the default username from the web preference settings** - The default username in the login field is admin. Do not display a default username because it gives an attacker an example of a username against which to wage a brute-force password attack. Brute-force attacks typically go through an extensive list of words (for example, a dictionary attack) in an attempt to guess the password.

Syntax:

```
web prefs login default
```

Example:

```
web prefs login default ""
```

- **Change settings for SOAP server** - Disable the Simple Object Access Protocol (SOAP) server if it is not in use. You can use the SOAP server to send configuration commands to the SteelHead. We recommend that you change the port (default TCP port 9001), if you want to enable the SOAP server.

Syntax:

```
[no] web soap-server enable
```

Example:

```
no web soap-server enable
```

Syntax:

```
[no] web soap-server port <port>
```

Example:

```
web soap-server port 1234
```

- **Disable unused default accounts** - Disable any default accounts that are not in use.

The monitor account is disabled by default, unless you upgrade the SteelHead from an earlier release in which the monitor account was enabled. The monitor account is a read-only account that allows you to view all settings, but does not allow you to make any changes.

Syntax:

```
[no] username <user-id> disable
```

Example:

```
username monitor disable
```

- **Change all default passwords and community strings** - Be sure to change the default password for the administrator and monitor accounts.

The most common problem with SNMP is that it uses the default community string of public. Change the default to something different.

Syntax:

```
username <user-id> password 0 <cleartext-password>
```

Example:

```
username admin password 0 o2fMu5TS!  
username monitor password 0 o2fMu5TD!
```

Syntax:

```
[no] snmp-server community <name>  
[no] snmp-server trap-community <trap-community-name>
```

Example:

```
snmp-server community o2fMu5TS!
```

- **Use strong passwords** - Strong passwords typically include combinations of letters, numbers, special characters, and combinations of uppercase and lowercase letters with at least eight characters in length. Strong passwords reduce the likelihood of a successful brute-force attack because they are not found in dictionaries and exponentially increase the complexity of the passwords.

An example of a strong password is o2fMu5TS!

- **Use AAA authentication** - One of the challenges with using local usernames and passwords is that when an employee leaves an organization, an administrator must touch every device that has a username and password configured for that former employee.

By leveraging TACACS+, you gain the advantage of having a single location for configuring usernames and passwords. When a person leaves your organization, you can simply disable that single account thereby preventing the user from access to all of the network devices configured to use TACACS+. Another benefit of TACACS+ is the ability to lock out an account after several unsuccessful login attempts.

TACACS+ also provides greater reporting capabilities regarding who is accessing which devices at what time. With a global username and password, you have no idea which administrator actually logged in at a specific time. These reports can be invaluable for tracking network changes and identifying who is making changes. Therefore, it is a critical tool for change management controls.

Refer to *Riverbed Command-Line Interface Reference Manual* and the *SteelHead User Guide* for more detailed information about how to configure AAA.

- **Configure the CLI session time-out** - By default, the SteelHead closes the SSH session to the command line after 15 minutes. You can configure this interval to be more or less with the following command:

Syntax:

```
cli default auto-logout *
```

Example:

```
cli default auto-logout 10
```

This command only affects new SSH sessions. If you want to modify the time-out session only for the current session (and not affect the default settings), use the following command:

Syntax:

```
cli session auto-logout *
```

You can disable the auto-logout feature with the following command:

```
no cli default auto-logout
```

This command changes both the current and the default settings.

You can display the current auto-logout settings with the following command:

```
show cli
```

- **Use strong SSL ciphers for management communications** - Be sure to use strong encryption ciphers for any HTTPS management communications. The cipher is the key that encrypts management communications to the SteelHead. An attacker could still use the hacking tools to crack the encrypted username and password if the encryption ciphers are too weak. An example of a weak cipher is only 56 or 64 bits. A strong cipher is greater than 128 bits.

Syntax:

```
web ssl cipher
```

Example:

```
web ssl cipher "HIGH:-aNULL:-kKRB5:-MD5"
```

- **Set an inactivity timer for console, SSH, and HTTPS sessions** - Be sure to set a proper inactivity time-out value for management sessions. Do not set a console inactivity time-out value to 0. This setting could allow an attacker to take over a previous management session if the previous administrator did not manually log off.

Syntax:

```
[no] web auto-logout <minutes>
[no] cli default auto-logout <minutes>
```

Example:

```
web auto-logout 10
cli default auto-logout 10
```

- **Ensure SNMP is listening on the management interface only** - To prevent unauthorized SNMP access, we recommend enabling SNMP access on the Primary interface only. This configuration allows administrators to control who can access SteelHeads through SNMP by way of using filters applied to routers, Layer-3 switches, or firewalls.

Syntax:

```
[no] snmp-server listen enable
[no] snmp-server listen interface <interface>
```

Example:

```
snmp-server listen enable
snmp-server listen interface Primary
```

- **Enable link state alarms** - Enable the link state alarms, which are disabled by default. These alarms can alert you to any attempt to modify the cabling on the SteelHeads by inserting a tap for illegal sniffing functions.

Syntax:

```
[no] stats alarm <type> <options>
```

Example:

```
stats alarm linkstate enable
```

- **Disable the autodiscovery SCC feature** - By default, all SteelHeads try to register with the SCC using the default hostname riverbedscc. If you do not have an SCC, disable this feature.

Syntax:

```
[no] cmc enable
```

Example:

```
no cmc enable
```

If you do have an SCC, we recommend that you use it to manually discover SteelHeads, thereby reducing the possibility that an attacker could compromise the DNS environment and change the IP address of the riverbedscc A record to a rogue SCC.

- **Configure a password policy in the SteelHead** - In RiOS 8.0 and later, you can configure user accounts on the SteelHead with a password policy to enforce minimum security standards for passwords, number of password attempts, and password expiration. The following password policies are predefined in the SteelHead:
 - **Strong security template** - Corresponds to the typical recommendations in Federal password guidelines.
 - **Basic security template** - Provides a minimal set of standards.

For more information about the type of accounts, how to configure role-based accounts, and information about the options for the password policy, see the *SteelHead User Guide*.

- **Use a BIOS password** - Enable a password for BIOS, which prevents admin password recovery without the supervisor password. To configure a BIOS password:
 - Connect a null modem cable to a SteelHead.
 - Open up a terminal on your host to the SteelHead.
 - Power up the SteelHead.
 - Press F4 to enter BIOS.
 - Navigate to the Security tab.
 - Specify a supervisor password.
 - Make sure that the user password option is set to OFF.
 - Save your configuration and continue to boot the SteelHead.

- **Use a boot loader password** - When you reboot the SteelHead, you can select from one of the two RiOS images on the menu. You can perform password recovery at this point by pressing the E key for edit. Pressing E alters the boot sequence to change the administrative password. Using the following commands, you can enable the boot loader to lock the password recovery process until a password is entered.

```
Steelhead (config) # boot bootloader password test1234
Steelhead (config) # write memory
Steelhead (config) # reload
...
```

```
-----
0: Riverbed Steelhead Software v. 5.5.3 (64bit)
1: Riverbed Steelhead Software v. 5.5.3 (64bit)
-----
```

```
Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS or 'p' to enter a
password to unlock the next set of features.
```

```
Press 'p'
Password: *****
```

- **SSL Issues with Internet Explorer 6 and Oracle R12** - Previously, RiOS fixed a vulnerability found in CBC-based ciphers prior to versions 0.9.6e by inserting an empty frame on the wire to avoid a Chosen Plaintext Attack on cipher-block chaining (CBC) ciphers. Some versions of client and server applications do not understand the insertion of empty frames into the encrypted stream and close the connection when they detect these frames. Therefore, RiOS no longer inserts empty frames by default. Examples of applications that close the connection when they detect these empty frames are IE6 and Oracle R12. SharePoint under IIS has also exhibited this behavior.

The failure occurs when the SSL application fails to understand the data payload when either the client or server is using a block cipher using CBC mode as the chosen cipher. This failure can be with DES, AES, or 3DES using CBC. Note that when SteelHeads are deployed, the chosen cipher can be different than when the client is negotiating directly with the SSL server.

Note: Because current web browsers do not protect themselves from this vulnerability, SteelHeads are no less secure than other vendor's appliances. From a security perspective, fixing this vulnerability is the responsibility of a server, not a patched client.

To determine whether the SteelHeads are inserting empty frames to avoid an attack, capture TCP dumps on the server-side SteelHead LAN interface and look at the Server Hello message that displays the selected cipher. Verify that DES, AES, or 3DES is the cipher. Also, check for the existence of 32-byte length SSL application data (this is the empty frame) on the LAN traces, followed by an SSL Alert.

To change the default and insert empty frames, enter the CLI command **no protocol ssl bug-work-around dnt-insrt-empty**.

Best practices for enabling SteelHead security features

The following best practices enable important security features provided by RiOS. These best practices are not requirements, but we recommend that you follow these suggestions as implementing them can enforce a secure deployment:

- **Use peering rules to control enhanced autodiscovery** - Enhanced autodiscovery is a feature that allows SteelHeads or SteelHead Mobiles to discover other SteelHeads using TCP options. This feature greatly reduces the complexities and time it takes to deploy SteelHeads. It works so seamlessly that it can occasionally have the undesirable effect of peering with SteelHeads on the internet that are not in your organization's management domain.

Another scenario could be that your organization has a decentralized management approach where different business units might make their own purchasing and management decisions. You might not want SteelHeads from two or more business units to peer with one another.

In these situations, we recommend using peering rules. Peering rules determine which connections your SteelHead optimizes connections with, based on the source and destination IP addresses or TCP ports. This feature lets you deny peering with any unwanted connections. Another option is to create an Accept peering rule for your corporate network that allows peering from your own IP addresses, and denies it otherwise.

For more information about using peering rules, see the *SteelHead User Guide*.

For more information about the in-path peering rule command, see the *Riverbed Command-Line Interface Reference Manual*.

- **Enable a secure inner channel between SteelHeads when using the SMB-signing proxy feature** - When sharing files, Windows provides the ability to sign CIFS messages to prevent man-in-the-middle attacks. Each CIFS message has a unique signature which prevents the message from being tampered with. This security feature is called SMB signing.

SMB signing is mandatory on all CIFS connections to domain controllers. Therefore, any CIFS connection to a domain controller must use SMB-signed packets.

You can enable the RiOS SMB signing feature on the server-side SteelHeads communicating with servers that have SMB signing set to **Required**. This feature alleviates latency in file access with CIFS acceleration while maintaining message security signatures. With SMB signing on, the SteelHead optimizes CIFS traffic by providing bandwidth optimization (RiOS SDR and LZ), TCP optimization, and CIFS latency optimization—even when the CIFS messages are signed.

However, because there is no packet signing taking place between the SteelHeads for these connections, we recommend that you configure a secure inner channel to encrypt the traffic between the SteelHeads.

- **Enable a secure inner channel between SteelHeads when using Exchange 2007 encryption** - Outlook 2007 has encryption enabled by default. The SteelHeads are able to decrypt this traffic; however, the connections between the SteelHeads are unencrypted by default. Configure a secure inner channel to encrypt all MAPI traffic between the SteelHeads.
- **Enable a secure inner channel to encrypt all optimized traffic between SteelHeads** - When you enable a secure inner channel, all data between the client-side and the server-side SteelHeads is sent over the secure inner channel. You configure the peer SteelHead as SSL peers so that they are trusted entities. The SteelHeads authenticate each other by exchanging certificates as part of the encrypted inner-channel setup. After the SteelHeads establish the secure inner channel, you can encrypt and optimize all optimized traffic between SteelHeads using the channel. The trust between the SteelHeads is bidirectional; the client-side SteelHead trusts the server-side SteelHead, and vice versa.

- **Authenticate WCCP service groups** - By default, WCCP peers in a WCCP group do not use authentication when registering. This default setting could allow an attacker to join a WCCP group and potentially cause a denial of service attack. Also an administrator could accidentally misconfigure a router to use a WCCP group that already is in use. Authentication controls would prevent these rogue devices from peering, thereby preventing possible network outages or degradation of performance.

Syntax:

```
[no] wccp service-group <service-id> {routers <routers> | assign-scheme [either | hash | mask]
| src-ip-mask <mask> | dst-ip-mask <mask> | src-port-mask <mask> | dst-port-mask <mask>}
{protocol [tcp | icmp] | encap-scheme [either | gre | l2] | dst-ip-mask <mask> flags <flags> |
password <password> | ports <ports> | priority <priority> | weight <weight> | assign-scheme
[either | hash | mask] | src-ip-mask <mask> | dst-ip-mask <mask> | src-portmask <mask> | dst-
port-mask <mask>}
```

Example:

```
wccp service-group 91 routers x.x.x.x password S3cuRity!
```

- **Encrypt the RiOS data store** - RiOS SDR takes all TCP traffic and segments using a rolling data-driven computation. The segmentation produced is not readily predictable without running the computation, so an attacker interested in reconstructing a particular file does not know how many segments are involved or what the file boundaries are within the segments. The segmentation is stable, so that two identical bit sequences produce the same segmentation. Each new segment identified is written to the RiOS data store, because each previously seen segment is reused.

Even though there is inherent security in the obfuscation of the RiOS data store, Riverbed still provides a mechanism for enabling strong encryption of the RiOS data store. Encrypting the RiOS data store significantly limits the exposure of sensitive data in the event a SteelHead is compromised by loss, theft, or other types of security violations. The secured data is impossible for a third party to retrieve.

Syntax:

```
[no] datastore encryption type {none | aes_128 | aes_192 | aes_256}
```

Example:

```
datastore encryption type aes_256
```

Next, select Clear the Data Store on Reboot and reboot the SteelHead.

- **Change the secure vault** - The secure vault contains sensitive information from your SteelHead configuration, including SSL private keys and the RiOS data store encryption key. These configuration settings are encrypted on the disk at all times using AES 256-bit encryption.

Initially, the secure vault is keyed with a default password known only to RiOS. This allows the SteelHead to automatically unlock the vault during system startup. You can change the password, but the secure vault does not automatically unlock on start up. To optimize SSL connections or to use RiOS data store encryption, the secure vault must be manually unlocked if the SteelHead is rebooted.

Therefore, we recommend using this feature only in conjunction with an SCC. The SCC can automatically unlock the Secure Vault when the SteelHead connects to the SCC after a reload.

Syntax:

```
secure vault {new-password <password> | reset-password <old-password> | unlock <password>}
```

Example:

```
Secure vault unlock o2fMu5TS!
```

- **Disable unused features** - Disable any features that are not in use. For example, MAPI Exchange is on by default. If your organization uses Lotus Notes, we recommend that you disable Exchange optimization.

Refer to the *Riverbed Command-Line Interface Reference Manual* or the *SteelHead User Guide* for the specific features you might want to disable.

- **Disable automatic email notification** - This feature proactively sends email notification of critical issues on the SteelHead (such as significant alarms and events) to Riverbed Support. Your organization might not want to send these automatic notifications.

Syntax:

```
[no] email autosupport enable
```

Example:

```
no email autosupport enable
```

- **Disable SteelHead reporting** - This feature proactively reports some very basic information back to Riverbed Support once a week. This reporting is initially disabled, however, if the user configures name server IP addresses for the SteelHead it is automatically enabled. Your organization might not want to send this report.

Syntax:

```
[no] support uptime-report enable
```

Example:

```
no support uptime-report enable
```

- **Delete the preconfigured NTP servers** - If your organization has NTP configured internally, we recommend removing the preconfigured NTP servers.

Syntax:

```
[no] ntp server {<hostname | ip-address>} [version <number>]
```

Example:

```
no ntp server 66.187.224.4
```

- **Configure Network Time Protocol (NTP) settings** - We recommend that you synchronize the SteelHead to an NTP server of your choice. By default, the appliance uses the Riverbed-provided NTP server. Time is a critical function for the SteelHead and other network devices. Networks rely on accurate time determination for managing, securing, planning, and debugging. Tampering with time sources or posing as a rogue time server can lead to critical issues such as network authentication, or less critical issues such as conflicting log message time stamps.

RiOS 8.0 and later support MD5-based NTP authentication at the CLI, and RiOS 8.5 and later support both MD5-based and SHA-based NTP authentication on the CLI and Management Console.

- **Disable any interfaces not in use** - Be sure to disable any interfaces that are not being used. Examples include the Auxiliary interface and any unused in-path interfaces.

Syntax:

```
[no] interface <interface name> <options>
```

Example:

```
interface inpath0_1 shutdown
```

For more information about SteelHead security, see the *SteelHead User Guide*.

Best practices for policy controls

Follow these best practices for implementing secure policy controls:

- **Use the Simple Certificate Enrollment Protocol (SCEP)** - In RiOS 5.5.2 and later, SCEP allows SteelHeads to request signed certificates for enrollment and reenrollment from the certificate server.
- **Use a Certificate Revocation List (CRL)** - In RiOS 5.5.2 and later, SteelHeads can download CRL lists that contain revoked certificates from certificate servers through LDAP. Revoked certificates are considered invalid and are not used by the SteelHead.

For more information about SCEP and CRL, see the *SteelHead Deployment Guide - Protocols*.

Best practices for security monitoring

After implementing security measures for your organization, we recommend enabling the following security monitoring features:

- **Enable Logging** - Be sure to enable logging and log to a syslog server. At a minimum, set logging to the **notice** level to capture failed login attempts. You can also change the default logging facility (CLI only) (system=local0, user=local1, and per-process=local2). Use the **logging facility user local# system <local-facility> perprocess <local-facility>** command to configure the syslog facility.

Example—A failed login attempt:

```
Apr 13 05:19:49 BRANCH webasd[6004]: [web.NOTICE]: web: Attempt to Authenticate admin
Apr 13 05:19:49 BRANCH webasd(pam_unix)[6004]: authentication failure; logname= uid=0 euid=0
tty= ruser= rhost= user=admin
Apr 13 05:19:49 BRANCH webasd[6004]: [web.NOTICE]: web: Failed to authenticate user admin: You
must provide a valid account name and password.
```

After you enable syslog and log to a server, remember to review the logs daily.

RiOS 6.0 also includes several SNMP traps to notify you of SteelHead configuration changes, successful logins, and system dump initiation. For more information, see the *SteelHead User Guide*.

Syntax:

```
[no] logging <ip-address> [trap <log-level>]
```

Example:

```
logging x.x.x.x trap notice
```

- **Email alerts** - Be sure to enable email alerts internally.

Syntax:

```
[no] email mailhub <hostname | ip-address>
[no] email notify events enable
[no] email notify failures enable
[no] email notify events recipient <email-address>
[no] email notify failures recipient <email-address>
```

Example:

```
email mailhub x.x.x.x
email notify events enable
email notify failures enable
email notify events recipient helpdesk@companyxyz.com
email notify failures recipient helpdesk@companyxyz.com
```

Refer to the *Riverbed Command-Line Interface Reference Manual* for more information about configuring email alerts.

- **Register with the Riverbed forums** - Riverbed has several forums that enable you to receive advanced notifications for:
 - general announcements and updates
 - software releases
 - features

To register with Riverbed forums, go to <https://splash.riverbed.com/welcome>.

Configuring SSL certificates for web user interface

The SteelHead automatically generates and uses a self-signed certificate to provide HTTPS access to the web UI to manage the appliance.

This certificate is separate from the SSL feature set. Management of SSL certificates for the web UI pertains to the SSL certificate used by the appliance's web UI when HTTPS is used.

You can replace the self-signed certificate with one created by the administrator or generated by a third-party certificate authority.

To upload a key or certificate

- Connect to the SteelHead CLI and enter the following command:

```
web ssl cert import pem <pem text>
```

Do not enter more than one certification and more than one key. Because neither is required, you can opt to update only the certificate.

To generate a new certificate for the existing key for use with HTTPS on the SteelHead

- Connect to the SteelHead CLI and enter the following command:

```
web ssl cert update
```

The new certificate is authorized for one year (365 days).

To generate a brand new self-signed certificate and key pair for use with HTTPS management on the SteelHead

- Connect to the SteelHead CLI and enter the following command:

```
web ssl cert generate
```

This command overwrites the existing certificate and key pair regardless of whether the previous one was self-signed or user added. This command generates a self-signed certificate that is authorized for one year (365 days).

Changing encryption for domain replication passwords

SteelHead appliances add the protocol domain-auth encrypt-upgd command to change all domain replication user passwords from Data Encryption Standard (DES) to Advanced Encryption Standard (AES). For more information, see the *Riverbed Command-Line Interface Reference Manual*.

REST API access

You enable access to the Riverbed REST API in the Administration > Security > REST API Access page. Representational State Transfer (REST) is a framework for API design. REST builds a simple API on top of the HTTP protocol. It is based on generic facilities of the standard HTTP protocol, including the six basic HTTP methods (GET, POST, PUT, DELETE, HEAD, INFO) and the full range of HTTP return codes. You can discover REST APIs by navigating links embedded in the resources provided by the REST API, which follow common encoding and formatting practices.

You can invoke the REST API to enable communication from one Riverbed appliance to another through REST API calls. For example:

- A NetProfiler communicating with a NetShark.
A NetProfiler retrieving Layer-7 application mapping information from a SteelHead.
- A NetProfiler appliance retrieving a QoS configuration from a SteelHead.

Note: There is an incompatibility between RiOS 9.0 and NetProfiler 10.0 preventing the QoS reporting functionality in NetProfiler from working. If you want to use NetProfiler QoS reporting, you must use a previous version than RiOS 9.0.

For all uses you must preconfigure an access code to authenticate communication between parties and to authorize access to protected resources.

For more information about the SteelHead REST API, see the *SteelHead User Guide* and the *SteelHead REST API Guide*.

Capacity planning

This section describes capacity planning for the SteelHead. This section includes the following topics:

- [“Model characteristics” on page 447](#)
- [“Admission control” on page 448](#)

Model characteristics

This section describes the characteristics of the optimization resources available for the SteelHeads. These resources are the primary determining factors for supported WAN capacity, maximum number of optimized TCP connections, and RiOS data store capacity. For example, the amount of hard drive space and RAM determine how large the RiOS data store can be.

This section includes the following topics:

- “TCP connections” on page 447
- “WAN capacity limits” on page 447
- “RiOS data store size” on page 447
- “Disk performance” on page 448

For more information about individual model specs, go to <http://www.riverbed.com/products-solutions/products/>.

TCP connections

Each SteelHead model has a maximum number of optimized TCP connections. The larger the SteelHead, the more CPU, memory, and disk resources are available, increasing the amount of supportable connections. This is one of the primary considerations you use for sizing branch offices. Typically, we recommend a guideline of five to ten connections per user. Sizing for data center SteelHeads must take in account all optimized connections coming into the data center. For planning purposes, use the high end numbers for these types of connections. Large amounts of active connections (connections that are actively transmitting data), such as HTTP, have more impact on the SteelHead resources.

To view the connection history use the **show stats connection <timeframe>** command.

You can see the average and maximum number of connections for the time frame entered. These numbers are useful to determine if the current SteelHead is properly sized for the number of connections. You can also use this command to review when adding addition users or applications that can increase the number of optimized connections.

WAN capacity limits

WAN capacity limit is the amount of optimized traffic that is sent outbound from a SteelHead. This is also commonly used for sizing branch office SteelHeads. For data center SteelHeads, the WAN capacity is a recommendation on how much typical data throughput a SteelHead can process. Model CX7055M and below are limited to the rated outbound capacity (outbound after optimization). No rate limit or hard restriction is applied to throughput for a data center SteelHead CX7055H, though excessive amounts of certain types of traffic can strain the resources available on the SteelHead.

RiOS data store size

RiOS data store size is the amount of disk space, in GB, available for SDR use. The data center SteelHeads use RAID 10 or FTS 7050 for disk redundancy and optimal performance.

Disk performance

Take disk performance into account for high-end data center deployments. Certain types of data put more load on the disk systems and can be monitored if performance issues are suspected. High throughput data replication deployments typically use dedicated SteelHeads.

Monitor the disk systems with the following OID.

OID	Description
1.3.6.1.4.1.17163.1.1.4.0.8	RAID Errors

Use the following Data Store Disk Load report in the Management Console to monitor disk performance.

If the Disk Load report is showing 80 to 90% for a sustained amount of time or multiple times a day that coincide with periods of lower performance and average RiOS data store cost greater than 5000, the disk load might be impacting overall performance.

Admission control

This section describes admission control and includes the following topics:

- [“Connection limits” on page 448](#)
- [“Memory limits” on page 449](#)

Admission control prevents the SteelHead from processing traffic when overloaded. It also controls the connection count limits. Admission control stops the interception of connections for optimization but still allows the connections to pass through without optimization. Admission control is in one of two states:

- **Flowing** - In the flowing state, connections are intercepted as normal. Every 30 seconds or every 20 connections, admission control reevaluates whether the system is within limits. If the system exceeds certain limits, admission control moves into the paused state.
- **Paused** - In the paused state, the SteelHead does not intercept connections. The connections currently intercepted continue to be optimized, although new connections are passed through. Every 30 seconds or every 20 connections, admission control reevaluates whether the system falls below certain limits. If so, admission control moves back into the flowing state.

Connection limits

Each model contains connection limits to limit the total number of connections that is accepted into the system. The connection limits have rising and falling thresholds. The rising threshold is the cutoff limit. While the system is in flowing state, if the connection count rises above this threshold, admission control moves to the paused state. The falling threshold is the enable limit. While the system is in the paused state, until the connection count falls below this threshold, admission control keeps the system in the paused state.

Some leeway is given for connection limits before admission control is triggered. The minimum number of additional allowed connections before entering admission control on any model is 10. For example, a SteelHead with a rating of 30 connection does not enter admission control until it passes 40 connections. The SteelHead does not exit admission control until the number of connections falls back below the rated limit. Using the same example, a SteelHead entering admission control at 40 connections does not start optimizing new connections again until it is back down below 30 connections (rated limit).

The SteelHead sends out an SNMP alert, called *Admission Control Error*, to the SNMP host that you define. It alerts you that the licensed optimization limit is reached. You can purchase a bigger SteelHead that can take all the optimized TCP connections; or you can limit the type of traffic to be optimized with in-path rules configuration, and ensure maximum optimization benefits are limited only to most critical traffic or specific traffic that is hogging the WAN bandwidth.

Trap and OID	SteelHead state	Text	Description
admissionConnError (enterprises.17163.1.1.4.0.11)	Control Admission	Admission control connections alarm has been triggered.	The SteelHead has entered admission control due to the number of connections and is unable to handle the amount of connections going over the WAN link. During this event, the SteelHead continues to optimize existing connections, but new connections are passed through without optimization. The alarm clears automatically when the traffic has decreased and no other action is needed.

We recommend polling the number of optimized connections periodically, so you can take proactive steps before all the optimized TCP connections are consumed. The following table shows the SNMP MIB for the number of optimized connections.

Trap and OID	SteelHead state	Text	Description
optimizedConnections (enterprises.17163.1.1.5.2.1.0)		Current total number of optimized connections.	The optimized connections count is the total of half opened, half closed, and established/flowing connections.

Memory limits

Each SteelHead model contains memory limits to limit the total amount of memory that is used. The memory limits have rising and falling thresholds. The rising threshold is the cutoff limit. While the system is in flowing state, if the memory usage rises above this threshold, admission control moves to the paused state. The falling threshold is the enable limit. When the system is in the paused state, until memory usage falls below this threshold, admission control keeps the system in the paused state.

Trap and OID	SteelHead state	Text	Description
admissionMemError (enterprises.17163.1.1.4.0.10)	Admission Control	Admission control memory alarm has been triggered.	The SteelHead has entered admission control due to memory consumption. The SteelHead is optimizing traffic beyond its rated capability and is unable to handle the amount of traffic passing through the WAN link. During this event, the SteelHead continues to optimize existing connections, but new connections are passed through without optimization. The alarm clears automatically when the traffic has decreased and no other action is needed.

Note: Use the **show admission** command to display the cutoff and enable settings for your SteelHead.

For additional information about admission control, see the Riverbed Knowledge Base article *Understanding Admission Control* at <https://supportkb.riverbed.com/support/index?page=content&id=s15140>.

Overview of exporting flow data

NetFlow and other Flow Data Collectors gather network statistics about network hosts, protocols and ports, peak usage times, and traffic logical paths. The flow data collectors update flow records with information pertaining to each packet traversing the specified network interface.

The flow data components are as follows:

- **Exporter** - When you enable flow data support on a SteelHead, it becomes a flow data Exporter. The SteelHead exports raw flow data records to a flow data collector. You only need one SteelHead with flow data enabled to report flow records.
- **Collector** - A server or appliance designed to aggregate the data the SteelHead exports. The Cascade Profiler or Cascade Gateway are examples of flow data collectors, which process and present this data in a meaningful way to the administrator. The collector captures:
 - enough information to map the outer-connection to its corresponding inner-connection.
 - the byte and packet reduction for each optimized connection.
 - information about which SteelHead interface optimized the connection, including which peer it used during optimization.
- **Analyzer** - A collection of tools used to analyze the data and provide relevant data summaries and graphs. Flow data analyzers are available for free or from commercial sources. An analyzer is often provided in conjunction with a collector.

For smaller networks, the flow data collector and analyzer are typically combined into a single device. For larger networks, a more distributed architecture might be used. In a distributed design, multiple flow data exporters export their data to several flow data collectors which in turn send data back to the flow data analyzer.

Some environments configure NetFlow on the WAN routers to monitor the traffic traversing the WAN. However, when the SteelHeads are in place, the WAN routers only see the inner-channel traffic and not the real IP addresses and ports of the client and server. By enabling flow data on the SteelHead, this becomes a nonissue altogether. The SteelHead can export the flow data instead of the router without compromising any functionality. By doing so, the router can spend more CPU cycles on its core functionality: routing and switching of packets.

Before you enable flow data support in your network, consider the following:

- Generating flow data can use large amounts of bandwidth, especially on low bandwidth links and thereby impact SteelHead performance.
- To reduce the amount of data exported, you can export only optimized traffic.

For information about SteelHead MIB and SNMP traps, see the *SteelHead User Guide*.

SNMP monitoring

This section describes the SNMP traps. It does not list the corresponding clear traps. Every SteelHead supports SNMP traps and email alerts for conditions that require attention or intervention. An alarm triggers for most (but not every) event and subsequently, the related trap is sent. For most events, when the condition is fixed, the system clears the alarm and sends out a clear trap. The clear traps are useful in determining when an event has been resolved.

RiOS 5.0 supports the following protocols:

- SNMPv1
- SNMPv2c

RiOS 6.0 and later support the following protocols:

- SNMPv3, which provides authentication through the User-based Security Model (USM).
- View-based Access Control Mechanism (VACM), which provides richer access control.

RiOS 7.0 and later support the SNMPv3 authentication with AES 128 and DES encryption described in the following table. We recommend the following OIDs as a good starting point from which to monitor your deployment. Additional variables can be added or removed as needed.

Note: The following OIDs are for xx55 SteelHeads only.

OID	Object type	Description
1.3.6.1.4.1.17163.1.1.5.2.1.0	optimizedConnections	Current total number of optimized connections
1.3.6.1.4.1.17163.1.1.5.2.2.0	passthroughConnections	Current total number of pass-through connections
1.3.6.1.4.1.17163.1.1.5.2.3.0	halfOpenedConnections	Current total number of half-opened (optimized) connections
1.3.6.1.4.1.17163.1.1.5.2.4.0	halfClosedConnections	Current total number of half-closed (optimized) connections

OID	Object type	Description
1.3.6.1.4.1.17163.1.1.5.2.5.0	establishedConnections	Current number of established (optimized) connections
1.3.6.1.4.1.17163.1.1.5.2.6.0	activeConnections	Current number of active (optimized) connections
1.3.6.1.4.1.17163.1.1.5.2.7.0	totalConnections	Total number of connections
1.3.6.1.4.1.17163.1.1.5.1.1.0	cpuLoad1	1-minute CPU load in hundredths
1.3.6.1.4.1.17163.1.1.5.1.2.0	cpuLoad5	5-minute CPU load in hundredths
1.3.6.1.4.1.17163.1.1.5.1.3.0	cpuLoad15	15-minute CPU load in hundredths
1.3.6.1.4.1.17163.1.1.5.1.4.0	cpuUtil1	Percentage CPU utilization, aggregated across all CPUs, rolling average over the past minute
1.3.6.1.4.1.17163.1.1.5.1.5.1.1.1	cpuIndivIndex	A synthetic number numbering the CPUs
1.3.6.1.4.1.17163.1.1.5.1.5.1.2.1	cpuIndivId	Name of the CPU, also serves as the Index for the table
1.3.6.1.4.1.17163.1.1.5.1.5.1.3.1	cpuIndivIdleTime	Idle time for this CPU
1.3.6.1.4.1.17163.1.1.5.1.5.1.4.1	cpuIndivSystemTime	System time for this CPU
1.3.6.1.4.1.17163.1.1.5.1.5.1.5.1	cpuIndivUserTime	User time for this CPU
1.3.6.1.4.1.17163.1.1.4.0.8	raidError	RAID errors

In multiple CPU systems, the last digit corresponds to the CPU number.

The following table summarizes SNMP traps that represent serious issues, and we recommend that you address them immediately.

Trap and OID	SteelHead state	Text	Description
procCrash (enterprises.17163.1.1.4.0.1)		A procCrash trap signifies that a process managed by PM has crashed and left a core file. The variable sent with the notification indicates which process crashed.	A process crashed and subsequently restarted by the system. The trap contains the name of the process that crashed. A system snapshot associated with this crash is created on the SteelHead and is accessible through the CLI or the Management Console. Riverbed Support might need this information to determine the cause of the crash. The crashed process automatically restarts and no other action is required on the SteelHead.
procExit (enterprises.17163.1.1.4.0.2)		A procExit trap signifies that a process managed by PM has exited unexpectedly, but not left a core file. The variable sent with the notification indicates which process exited.	A process unexpectedly exited and subsequently restarted by the system. The trap contains the name of the process. The process might have exited automatically due to other process failures on the SteelHead. Review the release notes for known issues related to this process exit. If none exist, Contact Riverbed Support to determine the cause of this event. The crashed process automatically restarts and no other action is required on the SteelHead.
bypassMode (enterprises.17163.1.1.4.0.7)	Critical	The SteelHead has entered bypass (failthru) mode.	The SteelHead entered bypass mode and passes through all traffic unoptimized. This is the result of the optimization service locking up or crashing. It can also happen when the system is first turned on or turned off. If this trap is generated on a system that was previously optimizing and is still running, contact Riverbed Support.
storeCorruption (enterprises.17163.1.1.4.0.9)	Critical	The RiOS data store is corrupted.	Corruption is detected in the RiOS data store. Contact Riverbed Support immediately.
haltError (enterprises.17163.1.1.4.0.12)	Critical	The service is halted due to a software error.	The optimization service halts due to a serious software error. Check to see if a core dump or sysdump was created. If so, retrieve the information and contact Riverbed Support immediately.

Trap and OID	SteelHead state	Text	Description
serviceError (enterprises.17163.1.1.4.0.13)	Degraded	There has been a service error. Consult the log file.	The optimization service encountered a condition that might degrade optimization performance. Consult the system log for more information.
licenseError (enterprises.17163.1.1.4.0.57)	Critical	The main SteelHead license has expired, been removed, or become invalid.	A license on the SteelHead has been removed, has expired, or is invalid. The alarm clears when a valid license is added or updated.
hardwareError (enterprises.17163.1.1.4.0.58)	Either Critical or Degraded, depending on the state	Hardware error detected.	<p>Indicates that the system has detected a problem with the SteelHead hardware. These issues trigger the hardware error alarm:</p> <ul style="list-style-type: none"> the SteelHead does not have enough disk, memory, CPU cores, or NIC cards to support the current configuration the SteelHead is using a memory Dual In-line Memory Module (DIMM), a hard disk, or a NIC that is not qualified by Riverbed other hardware issues <p>The alarm clears when you add the necessary hardware, remove the unqualified hardware, or resolve other hardware issues.</p>
lanWanLoopError (enterprises.17163.1.1.4.0.63)	Critical	LAN-WAN loop detected. System will not optimize new connections until this error is cleared.	A LAN-WAN network loop has been detected between the LAN and WAN interfaces on a SteelHead (virtual edition). This can occur when you connect the LAN and WAN virtual NICs to the same vSwitch or physical NIC. This alarm triggers when a SteelHead (virtual edition) starts up, and it clears after you connect each LAN and WAN virtual interface to a distinct virtual switch and physical NIC (through the vSphere Networking tab) and then reboot the SteelHead (virtual edition).

Trap and OID	SteelHead state	Text	Description
optimizationServiceStatusError (enterprises.17163.1.1.4.0.64)	Critical	Optimization service currently not optimizing any connections.	<p>The optimization service has encountered an optimization service condition. The message indicates the reason for the condition:</p> <ul style="list-style-type: none"> ■ optimization service is not running This message appears after a configuration file error. For more information, review the SteelHead logs. ■ in-path optimization is not enabled This message appears if an in-path setting is disabled for an in-path SteelHead. For more information, review the SteelHead logs. ■ optimization service is initializing This message appears after a reboot. The alarm clears on its own; no other action is necessary. For more information, review the SteelHead logs. ■ optimization service is not optimizing This message appears after a system crash. For more information, review the SteelHead logs. ■ optimization service is disabled by user This message appears after entering the no service enable command or shutting down the optimization service from the Management Console. For more information, review the SteelHead logs. ■ optimization service is restarted by user This message appears after the optimization service is restarted from either the CLI or Management Console. You might want to review the SteelHead logs for more information.

Trap and OID	SteelHead state	Text	Description
storageProfSwitchFailed (enterprises.17163.1.1.4.0.73)	Either Critical or Needs Attention, depending on the state	Storage profile switch failed	<p>An error has occurred while repartitioning the disk drives during a storage profile switch. A profile switch changes the disk space allocation on the drives, clears the Granite and VSP data stores, and repartitions the data stores to the appropriate sizes.</p> <p>You switch a storage profile by entering the disk-config layout CLI command at the system prompt or by choosing Administration > System Settings: Disk Management on an EX or EX+Granite SteelHead and selecting a storage profile.</p> <p>These reasons can cause a profile switch to fail:</p> <ul style="list-style-type: none"> ■ RiOS cannot validate the profile. ■ The profile contains an invalid upgrade or downgrade. ■ RiOS cannot clean up the existing VDMKs. During clean up RiOS uninstalls all slots and deletes all backups and packages. <p>When you encounter this error, try to switch the storage profile again. If the switch succeeds, the error clears. If it fails, RiOS reverts the SteelHead to the previous storage profile.</p> <ul style="list-style-type: none"> ■ If RiOS is unable to revert the SteelHead to the previous storage profile, the alarm status becomes critical. ■ If RiOS successfully reverts the SteelHead to the previous storage profile, the alarm status displays needs attention.

Trap and OID	SteelHead state	Text	Description
flashProtectionFailed (enterprises.17163.1.1.4.0.75)	Critical	Flash disk hasn't been backed up due to not enough free space on /var filesystem.	Indicates that the USB flash drive has not been backed up because there is not enough available space in the /var filesystem directory. Examine the /var directory to see if it is storing an excessive amount of snapshots, system dumps, or TCP dumps that you could delete. You could also delete any RiOS images that you no longer use.
datastoreNeedClean (enterprises.17163.1.1.4.0.76)	Critical	The data store needs to be cleaned.	You need to clear the RiOS data store. To clear the data store, choose Administration > Maintenance: Services and select the Clear Data Store check box before restarting the appliance. Clearing the data store degrades performance until the system repopulates the data.

If an error condition exists, there are several alarms that are generated along with the SNMP traps. If the email feature is configured, you receive an email notification in addition to the alarms.

To limit the number of alarms generated over a given period of time, use the **stats alarm <alarm-name> rate-limit count <thresholds> <count>** command.

There are three sets of thresholds: short, medium, and long. Each threshold has a window, which is several seconds, and a maximum count. If, for any threshold, the number of alarms exceeds the maximum during the window, an alarm is not generated and emails are not sent.

For more information about configuring SNMP and other important traps, see the *SteelHead User Guide*.

Configuring SNMPv3 authentication and privacy

RiOS 7.0 and later include privacy to the SNMPv3 feature to support authentication and privacy encryption of SNMPv3 messages. You can use AES 128 and DES to send an SNMPv3 encryption for GET action.

All SNMPv3 passwords (authentication/privacy) are stored as hashed (MD5/SHA), and they are all master keys, even if you provide plain text password during configuration.

An SNMP agent runs in every SteelHead that supports SNMP GET request action. Among the techniques to secure SNMP traffic, such as access control lists, you can use SNMPv3 to provide authentication and privacy. The main benefit for SNMPv3 authentication is to ensure the integrity of SNMP traffic, while privacy provides encryption protecting data from being seen by a third party.

Configuring an SNMPv3 GET request encryption is a two-part process:

- **Configure USM user**

The user corresponds with the authentication and privacy mechanism that a management station uses to access the SteelHead.

- **Configure ACLs**

To configure the ACLs, you need to add or edit a group, view and access policy. You cannot add an access policy with a group and a view. Security names are not supported by SNMPv3. To restrict SNMPv3 USM users from polling a specific subnet, use the RiOS ACL feature on the Administration > Security: Management ACL page.

Views represent the OIDs a management station is allowed to access. You can create multiple views and restrict specific OIDs. A view starts with the highest level OID that you specify, and you can view all OIDs further down in the hierarchy, unless you specifically restrict them. You can only view OIDs in the hierarchy.

You must associate a group with a view. After you associate a group with a view, you can define an access policy to link the user, group, and view together.

The following procedure shows an example of a user named *Cascade* created with SHA authentication and AES encryption for privacy.

To configure a USM user

1. From the SteelHead Management Console, choose Administration > System Settings: SNMPv3.

2. Select Add a New User.

Figure 20-4. Add a New USM User

The screenshot shows the 'SNMP v3' configuration page under 'System Settings'. The main heading is 'SNMP v3' with a help icon. Below it, the text 'Create User-based Security Model users.' is displayed. The 'Users:' section contains two buttons: 'Add a New User' (selected) and 'Remove Selected'. A note states: 'Users can be authenticated with either a password or a key.' The form fields are as follows:

- User Name:
- Authentication Protocol:
- Authentication:
- Password: (at least 8 characters)
- Password Confirm:
- ☒ Use Privacy Option
 - Privacy Protocol:
 - Privacy:
 - Privacy Password: (at least 8 characters)
 - Privacy Password Confirm:

An 'Add' button is located at the bottom left of the form.

3. Select the Use Privacy Option check box.
4. Select AES or DES from the Privacy Protocol drop-down list.
5. Select any of the options in the Privacy drop-down list and complete any corresponding steps.

Figure 20-4 shows Supply a Password and the corresponding password.

6. Click Add.

The following procedure shows an example of a group *NetProfiler* created, and then user *Cascade* is associated with the group *Profiler*.

To configure SNMP ACLs

1. From the SteelHead Management Console, choose Administration > System Settings: SNMP ACLs.
2. Select the Add a New Group tab.

Figure 20-5. Add a New Group

Groups:

▼ Add a New Group ✕ Remove Selected

Specify the group name and select the security models. For v1 and v2c security models, select the security name. For usm security models, select the user name.

Group Name:

Security Model and Name Pairs: - +

Add

3. Specify a group name.
4. Select usm and select the user you created in [“To configure a USM user” on page 458](#).
5. Click **Add**.
6. Select the Add a New View tab.

Figure 20-6. Add a New View

Views:

▼ Add a New View ✕ Remove Selected

View Name:

Includes:

(one .x.y.z per line)

Excludes:

(one .x.y.z per line)

Add

View Name	Includes	Excludes
No Views.		

7. Specify a view name.
8. Specify the OIDs to include and exclude from the view.
9. Click **Add**.
10. Select Add a New Access Policy.

Figure 20-7. Add a New Access Policy

Access Policies:

▼ Add a New Access Policy ✕ Remove Selected

Group Name:

Security Level:

Read View:

Add

Group Name	Security Level	Read View
No Access Policies.		

11. Select the group name you created from the Group Name drop-down list.
12. Select AuthPriv from the Security Level drop-down list.
13. Select the view you created from the Read View drop-down list.
14. Click **Add**.

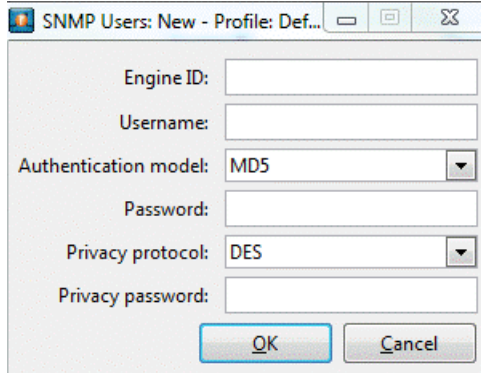
You can verify your configuration in Wireshark. Make sure the SNMP PDUs are encrypted.

Figure 20-8. Wireshark verification

Filter: <input type="text" value="snmp.encryptedPDU"/>												
No.	Time	Delta Time	Conv.	VLAN	Src	Dest	S Port	D Port	Proto	Len	DSCP	Info
3874	391.899305				192.168.0.17	192.168.0.10	58700	161	SNMP	223	0	encryptedPDU: privkey unknown
3875	391.899881				192.168.0.10	192.168.0.17	161	58700	SNMP	240	0	encryptedPDU: privkey unknown
3896	396.898033				192.168.0.17	192.168.0.10	58701	161	SNMP	223	0	encryptedPDU: privkey unknown
3897	396.898695				192.168.0.10	192.168.0.17	161	58701	SNMP	240	0	encryptedPDU: privkey unknown
3908	401.897648				192.168.0.17	192.168.0.10	58702	161	SNMP	223	0	encryptedPDU: privkey unknown

To decrypt the SNMP packets for further troubleshooting

1. From the Wireshark menu, choose Edit > Preferences > Protocols > SNMP.
2. Select the Edit for the SNMP Users window.

Figure 20-9. SNMP Users window

3. Complete the information in the SNMP Users window.

The engine ID is available on the SteelHead through the **show snmp** command or near the end of the running-configuration. The username, authentication model, password, privacy protocol, and privacy password are the same settings you configured for the SNMPv3 user on the SteelHead.

4. Click **OK**.

Wireshark decrypts the SNMP encrypted packets and you can analyze further for troubleshooting.

Troubleshooting SteelHead Deployment Problems

This chapter describes common deployment problems and solutions. This chapter includes the following sections:

- [“Common deployment issues” on page 463](#)
- [“MTU sizing” on page 476](#)

For information about SteelHead installation issues, see the *SteelHead Installation and Configuration Guide*.

For information about the factors to consider before you deploy the SteelHead, see [“Choosing the right SteelHead model” on page 28](#).

Common deployment issues

This section provides solutions to the following deployment issues:

- [“Duplex mismatches” on page 463](#)
- [“Network asymmetry” on page 466](#)
- [“Unknown \(or unwanted\) SteelHead appears on the current connections list” on page 468](#)
- [“Outdated antivirus software” on page 468](#)
- [“Packet ricochets” on page 469](#)
- [“Router CPU spikes after WCCP configuration” on page 469](#)
- [“Server Message Block signed sessions” on page 471](#)
- [“Unavailable opportunistic locks” on page 475](#)
- [“Underutilized fat pipes” on page 476](#)

Duplex mismatches

This section describes common problems that can occur in networks in which duplex settings do not match. Duplex mismatch occurs when the speed of a network interface that is connected to the SteelHead does not match.

The number one cause of poor performance issues with SteelHead installations is duplex mismatch. A duplex mismatch can cause performance degradation and packet loss.

Signs of duplex mismatch:

- You cannot connect to an attached device.
- You can connect with a device when you choose auto-negotiation, but you cannot connect with the same device when you manually set the speed or duplex.
- Minimal or no performance gains.
- Loss of network connectivity.
- Intermittent application or file errors.
- All of your applications are slower after you have installed in-path SteelHeads.

To determine whether the slowness is caused by a duplex mismatch

1. Create a pass-through rule for the application on the client-side SteelHead and ensure that the rule is at the top of the in-path rules list. You add a pass-through rule with the **in-path rule pass-through** command, or you can use the Management Console.
2. Restart the application.
3. Check that all connections related to the application are being passed through. If all connections related to the application are being passed through and the performance of the application does not return to the original levels, the slowness is most likely due to duplex mismatch.

The following sections describe several possible solutions to duplex mismatch.

Solution: Manually set matching speed and duplex

One solution for mismatched speed and duplex settings is to manually configure the settings.

1. Manually set (that is, hard set) matching speed and the duplex settings for the following four ports:
 - Devices (switches) connected on the SteelHead LAN port
 - Devices (routers) connected on the SteelHead WAN port
 - The SteelHead LAN port
 - The SteelHead WAN port

We recommend the following speeds:

- Fast Ethernet Interfaces: 100 megabits full duplex
- Gigabit Interfaces: 1000 megabits full duplex

For more details, see the Riverbed Knowledge Base article, *Problems manually setting 1000 Mbps/Full on SteelHead*, at <https://supportkb.riverbed.com/support/index?page=content&id=s14623>.

We recommend that you avoid using half-duplex mode whenever possible. If you are using a modern interface, and it appears to not support full duplex, double check the duplex setting. It is likely that one side is set to auto and the other is set to fixed. To manually change interface speed and duplex settings, use the **interface** command. For details, see the *Riverbed Command-Line Interface Reference Manual*.

2. Verify that each of the above devices:

- have settings that match in optimizing mode.
- is configured to see interface speed and duplex settings, using the **show configuration** command. By default, the SteelHead automatically negotiates speed and duplex mode for all data rates and supports full duplex mode and flow control. To change interface speed and duplex settings, use the **interface** command.
- have settings that match in bypass mode.
- are not showing any errors or collisions.
- does not have a half-duplex configuration (forced or negotiated) on either the WAN or the LAN.
- has at least 100 Mbps speed, forced or negotiated, on the LAN.
- has network connectivity in optimization and in failure mode.

For information about failure mode, see [“Failure modes” on page 200](#).

3. Test connectivity with the SteelHead powered off to ensure that the SteelHead does not sever the network in the event of a hardware or software problem. This must be done last, especially after making any duplex changes on the connected devices.

4. If the SteelHead is powered off and you cannot pass traffic through it, verify that you are using the correct cables for all devices connected to the SteelHead. The type of cable is determined by the device connecting to the SteelHead:

- Router to SteelHead: use a crossover cable.
- Switch to SteelHead: use a straight-through cable.
- Do not rely on Auto MDI/MDI-X to determine which cables you are using.

For information about cables, see [“Choosing the correct cables” on page 205](#).

5. Use a cable tester to verify that the SteelHead in-path interface is functioning properly: turn off the SteelHead, and connect the cable tester to the LAN and WAN port. The test result must show a crossover connection.

6. Use a cable tester to verify that all of the cables connected to the SteelHead are functioning properly.

For information about how to choose the correct cables, see [“Choosing the correct cables” on page 205](#).

Solution: Use an intermediary switch

If you have tried to manually set matching speed and duplex settings, and duplex mismatch still causes slow performance and lost packets after you deploy in-path SteelHeads, introduce an intermediary switch that is more compatible with both existing network interfaces. We recommend that you use this option only as a last option.

Important: To use an intermediary switch, you must also change your network cables appropriately.

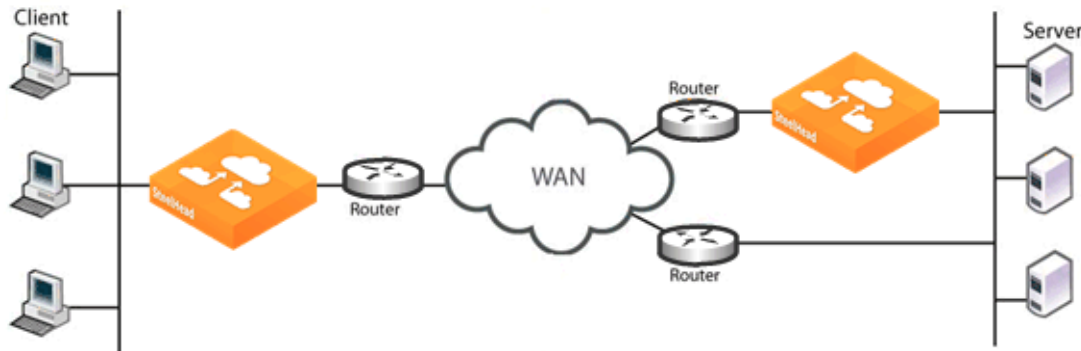
Network asymmetry

If some of the connections in a network are optimized and some are passed through unoptimized, it might be due to network asymmetry. Network asymmetry causes a client request to traverse a different network path than the server response. Network asymmetry can also break connections.

If SYN packets that traverse from one side of the network are optimized, but SYN packets that traverse from the opposite side of the network are passed-through unoptimized, it is a symptom of network asymmetry.

Figure 21-1 shows an asymmetric server-side network in which a server response can traverse a path (the bottom path) in which a SteelHead is not installed.

Figure 21-1. Server-side asymmetric network



The following sections describe several possible solutions to network asymmetry.

With RiOS 3.0.x and later, you can configure your SteelHeads to automatically detect and report asymmetric routes within your network. Whether asymmetric routing is automatically detected by SteelHeads or is detected in some other way, use the solutions described in the following sections to work around it.

For information about configuring auto-detection of asymmetric routes, see the *SteelHead User Guide*.

Solution: Use connection forwarding

For a network connection to be optimized, packets traveling in both network directions (from server to client and from client to server) must pass through the same client-side and server-side SteelHead. In networks in which asymmetric routing occurs because client requests or server responses can traverse different paths, you can solve it by:

- ensuring that there is a SteelHead installed on every possible path a packet can traverse. You would install a second server-side SteelHead, covering the bottom path. For details, see [Figure 21-1](#).
- setting up connection forwarding to route packets that traversed one SteelHead in one direction to traverse the same SteelHead in the opposite direction. Connection forwarding can be configured on the client side or server side of a network.

To set up connection forwarding, use the Management Console or CLI as described in the *SteelHead User Guide* and the *Riverbed Command-Line Interface Reference Manual*.

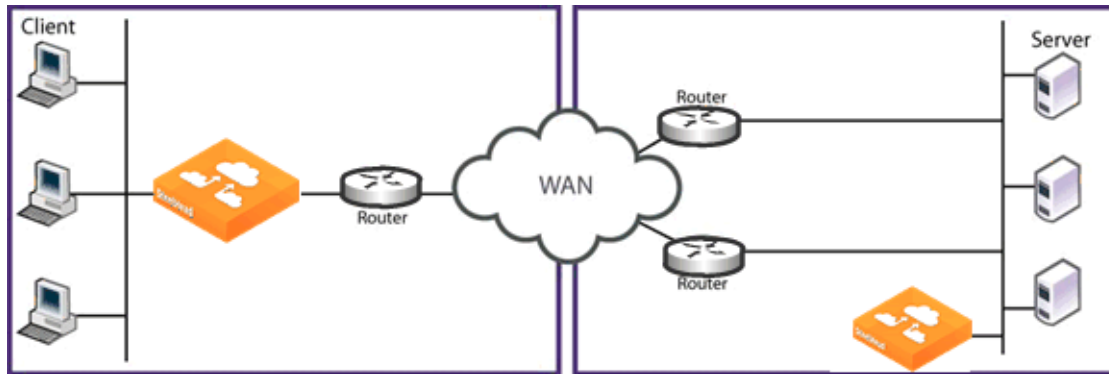
For more information, see [“Connection forwarding” on page 56](#).

Solution: Use virtual in-path deployment

Because a connection cannot be optimized unless packets traveling in both network directions pass through the same client-side SteelHead and the same server-side SteelHead, you can use a virtual in-path deployment to solve network asymmetry.

In the example network shown in [Figure 21-1](#), changing the server-side SteelHead that is deployed in-path on the top server-side path to a virtual in-path deployment, ensures that all server-side traffic passes through the server-side SteelHead.

Figure 21-2. Virtual in-path deployment to solve network asymmetry



A virtual in-path deployment differs from a physical in-path deployment in that a packet redirection mechanism directs packets to SteelHeads that are not in the physical path of the client or server. Redirection mechanisms include a Layer-4 switch (or server load balancer), WCCP, and PBR. These redirection mechanisms are described in:

- [“Virtual In-Path Deployments” on page 245](#)
- [“Out-of-Path Deployments” on page 387](#)
- [“WCCP Virtual In-Path Deployments” on page 249](#)
- [“Policy-Based Routing Virtual In-Path Deployments” on page 287](#)

Solution: Deploy a four-port SteelHead

If you have a SteelHead that supports a Four-Port Copper Gigabit-Ethernet PCI-X card, you can deploy it to solve network asymmetry in which a two-port SteelHead or one of the solutions described in the previous sections is not successful.

For example, instead of the two-port SteelHead deployed to one server-side path as shown [Figure 21-1](#), you deploy a four-port SteelHead on the server side of the network. All server-side traffic passes through the four-port SteelHead and asymmetric routing is eliminated.

For information about two-port and four-port SteelHeads, see the *Network and Storage Card Installation Guide*.

Unknown (or unwanted) SteelHead appears on the current connections list

Enhanced autodiscovery greatly reduces the complexities and time it takes to deploy SteelHeads. It works so seamlessly that occasionally it has the undesirable effect of peering with SteelHeads on the internet that are not in your organization's management domain or your corporate business unit. When an unknown (or unwanted) SteelHead appears connected to your network, you can create a peering rule to prevent it from peering and remove it from your list of peers. The peering rule defines what to do when a SteelHead receives an autodiscovery probe from the unknown SteelHead.

To prevent an unknown SteelHead from peering

1. Choose Configure > Optimization > Peering Rules.
2. Click **Add a New Peering Rule**.
3. Select Passthrough as the rule type.
4. Specify the source and destination subnets. The source subnet is the remote location network subnet (in the format XXX.XXX.XXX.XXX/XX). The destination subnet is your local network subnet (in the format XXX.XXX.XXX.XXX/XX).
5. Click **Add**.

In this example, the peering rule passes through traffic from the unknown SteelHead in the remote location.

When you use this method and add a new remote location in the future, you must create a new peering rule that accepts traffic from the remote location. Place this new Accept rule before the Pass-through rule.

If you do not know the network subnet for the remote location, there is another option: you can create a peering rule that allows peering from your corporate network subnet and denies it otherwise. For example, create a peering rule that accepts peering from your corporate network subnet and place it as the first rule in the list.

Next, create a second peering rule to pass through all other traffic. In this example, when the local SteelHead receives an autodiscovery probe, it checks the peering rules first (from top to bottom). If it matches the first Accept rule, the local SteelHead peers with the other SteelHead. If it does not match the first Accept rule, the local SteelHead checks the next peering rule, which is the pass-through rule for all other traffic. In this case, the local SteelHead just passes through the traffic and does not peer with the other SteelHead.

After you add the peering rule, the unknown SteelHead appliance appears in the Current Connections report as a Connected Appliance until the connection times out. After the connection becomes inactive, it appears dimmed. To remove the unknown appliance completely, restart the optimization service.

Outdated antivirus software

After installing SteelHeads, if application access over the network does not speed up or certain operations on files (such as dragging and dropping) speed up greatly but application access does not, it might be due to old antivirus software installed on a network client.

For similar problems, see:

- [“Server Message Block signed sessions” on page 471](#)
- [“Unavailable opportunistic locks” on page 475](#)

Solution: Upgrade antivirus software

If it is safe to do so, temporarily disable the antivirus software and try opening files. If performance improves with antivirus software disabled, we recommend that you upgrade the antivirus software.

If performance does not improve with antivirus software disabled or after upgrading antivirus software, contact Riverbed Support site at <https://support.riverbed.com>.

Packet ricochets

Signs of packet ricochet are:

- Network connections fail on their first attempt but succeed on subsequent attempts.
- The SteelHead on one or both sides of a network has an in-path interface that is different from that of the local host.
- You have not defined any in-path routes in your network.
- Connections between the SteelHead and the clients or server are routed through the WAN interface to a WAN gateway, and then they are routed through a SteelHead to the next-hop LAN gateway.
- The WAN router drops SYN packets from the SteelHead before it issues an ICMP redirect.

Solution: Add in-path routes

To prevent packet ricochet, add in-path routes to local destinations. For details, see *SteelHead User Guide*.

For information about packet ricochet, see [“In-path redundancy and clustering examples” on page 213](#).

Solution: Use simplified routing

You can also use simplified routing to prevent packet ricochet. To configure simplified routing, use the **in-path simplified routing** command or the Management Console.

For information about simplified routing and how to configure it, see the *Riverbed Command-Line Interface Reference Manual* or the *SteelHead User Guide*.

Router CPU spikes after WCCP configuration

If the CPU usage of the router spikes after WCCP configuration, it might be because you are not using a WCCP-compatible Cisco IOS release or because you must use inbound redirection.

The following sections describe several possible solutions to router CPU spike after WCCP configuration.

Solution: Use mask assignment instead of hash assignment

The major difference between the hash and mask assignment methods lies in the way traffic is processed within the router/switch. With a mask assignment, traffic is processed entirely in the hardware, which means the CPU of the switch is minimal. A hash assignment uses the switch CPU for part of the load distribution calculation and hence places a significant load on the switch CPU. The mask assignment method was specifically designed for hardware-based switches and routers (such as Cisco 3560, 3750, 4500, 6500, and 7600).

For information about mask assignment, see [“WCCP Virtual In-Path Deployments” on page 249](#).

Solution: Check internetwork operating system compatibility

Because WCCP is not fully integrated in every IOS release and on every platform, ensure that you are running a WCCP-compatible IOS release. If you have questions about the WCCP compatibility of your IOS release, contact Riverbed Support site at <https://support.riverbed.com>.

If you are certain that you are running a WCCP-compatible IOS release and you experience router CPU spike after WCCP configuration, review the remaining sections for possible solutions.

Solution: Use inbound redirection

One possible solution to router CPU spike after WCCP configuration is to use inbound redirection instead of outbound redirection. Inbound redirection ensures that the router does not waste CPU cycles consulting the routing table before handling the traffic for WCCP redirection.

For information about redirection, see [“WCCP Virtual In-Path Deployments” on page 249](#)

Solution: Use inbound redirection with fixed-target rules

If inbound redirection, as described in [“Solution: Use inbound redirection” on page 470](#), does not solve router CPU spike after WCCP is configured, try using inbound redirection with a fixed-target rule between SteelHeads. The fixed-target rule can eliminate one redirection interface.

Fixed-target rules directly specify server-side SteelHeads near the target server that you want to optimize. You determine which servers you would like the SteelHead to optimize (and, optionally, which ports), and you add fixed-target rules to specify the network of servers, ports, and out-of-path SteelHeads to use.

For information about how to configure inbound redirection and fixed-target rules, see [“WCCP Virtual In-Path Deployments” on page 249](#)

Solution: Use inbound redirection with fixed-target rules and redirect list

If the solutions described in the previous sections do not solve router CPU spike after WCCP is configured, try using inbound redirection with a fixed-target rule and a redirect list. A redirect list can reduce the load on the router by limiting the amount of unnecessary traffic that is redirected by the router.

For details, see [“WCCP Virtual In-Path Deployments” on page 249](#)

Solution: Base redirection on ports rather than ACLs

If the solutions described in the previous sections do not solve router CPU spike after WCCP configuration, consider basing traffic redirection on specific port numbers rather than using ACLs.

Solution: Use PBR

If the solutions described in the previous sections do not solve router CPU spike after WCCP configuration, consider using PBR instead of WCCP.

For information about PBR, see [“Policy-Based Routing Virtual In-Path Deployments” on page 287](#)

Server Message Block signed sessions

This section provides a brief overview of problems that can occur with Windows Server Message Block (SMB) signing. For information about SMB signing, the performance cost associated with it, and solutions to it, see the *SteelHead User Guide*.

If network connections appear to be optimized but there is no performance difference between a cold and warm transfer, it might be due to SMB-signed sessions.

SMB-signed sessions support compression and RiOS SDR, but render latency optimization (for example read-ahead, and write-behind) unavailable.

Signs of SMB signing:

- Access to some Windows file servers across a WAN is slower than access to other Windows file servers across the WAN.
- Connections are shown as optimized in the Management Console.
- The results of a **TCP dump** show low WAN utilization for files where their contents do not match existing segments in the segment store.
- Copying files via FTP from the slow server is much faster than copying the same files via mapped network drives (CIFS).

When copying FTP from a slow server is much faster than copying from the same server via a mapped network drive, the possibility of other network problems (such as duplex mismatch or network congestion) with the server is ruled out.

- Log messages in the Management Console such as:

```
error=SMB_SHUTDOWN_ERR_SEC_SIG_ENABLED
```

The following sections describe possible solutions to SMB-signed sessions.

For similar problems, see:

- [“Unknown \(or unwanted\) SteelHead appears on the current connections list” on page 468](#)
- [“Unavailable opportunistic locks” on page 475](#)

Solution: Fully optimize SMB-signed traffic

Before you use any of the following solutions, configure your SteelHead to optimize SMB-signed traffic. For information about how to configure SMB-signed traffic, see the *SteelHead Deployment Guide - Protocols*.

If you do not have the privileges or the correct information for SMB-signed traffic optimization, try the following solutions.

Solution: Enable secure-CIFS

Enable Secure-CIFS using the **protocol cifs secure-sig-opt enable** command.

The Secure-CIFS feature automatically stops Windows SMB signing. SMB signing prevents the SteelHead from applying full optimization on CIFS connections and significantly reduces the performance gain from a SteelHead deployment. SMB-signed sessions support compression and RiOS SDR, but render latency optimization (read-ahead, write-behind) unavailable.

With Secure-CIFS enabled, you must consider the following factors:

- If the client-side machine has **Required** signing, enabling the Secure-CIFS feature prevents the client from connecting to the server.
- If the server-side machine has **Required** signing, the client and the server connect but you cannot perform full latency optimization with the SteelHead. (Domain Controllers default to **Required**.)

For information about SMB signing, see the *SteelHead Installation and Configuration Guide*.

Alternatively, if your deployment requires SMB signing, you can optimize signed CIFS messages by selecting **Enable SMB Signing** in the Optimization > Protocols: CIFS (SMB1) page of the Management Console. Before you enable SMB signing, make sure you *disable* **Optimize Connections with Security Signatures**. For detailed information about optimizing signed CIFS messages, including procedures for your Windows server, see the *SteelHead User Guide*.

Note: Secure-CIFS is enabled by default beginning with RiOS 2.x.

Tip: If a log file shows messages such as error=SMB_SHUTDOWN_ERR_SEC_SIG_REQUIRED, use the solution described in [“Solution: Disable SMB signing with Active Directory” on page 472](#). Enabling secure-CIFS has no effect when SMB signing has been set to **required**.

For details, see the *SteelHead Management Console Online Help* or the *SteelHead User Guide*.

Solution: Disable SMB signing with Active Directory

If you have tried enabling Secure-CIFS as described in [“Solution: Enable secure-CIFS” on page 472](#) but SMB signing still occurs, consider using Active Directory (AD) to disable SMB signing requirements on servers or clients.

If the Security Signature feature does not disable SMB signing, you must revise the default SMB registry parameters. SMB signing is controlled by the following registry parameters:

```
enablesecuritysignature (SSEn)  
requiresecuritysignature (SSReq)
```

The registry settings are located in:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\lanmanserver\parameters
```


The following table summarizes the default SMB signing registry parameters.

Machine role	SSEn	SSReq
Client/Workstation	ON	OFF
Member Server	OFF	OFF
Domain Controller	ON	ON

With these default registry parameters, SMB signing is negotiated in the following manner:

- SMB/CIFS exchanges between the Client/Workstation and the Member Server are not signed.
- SMB/CIFS exchanges between the Client/Workstation and the Domain Controller are always signed.

The following table lists the complete matrix for SMB registry parameters that ensure full optimization (that is, bandwidth and latency optimization) using the SteelHead.

Number	Parameters on workstation		Parameters on server		Result
	SSReq	SSEn	SSReq	SSEn	
1	OFF	OFF	OFF	OFF	Signature Disabled; SteelHead full optimization
2	OFF	OFF	OFF	ON	Signature Disabled; SteelHead full optimization
3	OFF	OFF	ON	ON	Cannot establish session
4*	OFF	OFF	ON	ON	Cannot establish session
5	OFF	ON	OFF	OFF	Signature Disabled; SteelHead full optimization
6	OFF	ON	OFF	ON	Signature Enabled; SteelHead bandwidth optimization
7	OFF	ON	ON	ON	Signature Enabled; SteelHead bandwidth optimization
8*	OFF	ON	OFF	ON	Signature Enabled; SteelHead bandwidth optimization
9	ON	ON	OFF	OFF	Cannot establish session
10*	ON	ON	OFF	ON	Signature Enabled; SteelHead bandwidth optimization
11	ON	ON	ON	ON	Signature Enabled; SteelHead bandwidth optimization
12	ON	ON	OFF	ON	Signature Enabled; SteelHead bandwidth optimization
13+	ON	OFF	OFF	OFF	Cannot establish session
14+	ON	OFF	OFF	ON	Signature Enabled; SteelHead bandwidth optimization
15+	ON	OFF	ON	ON	Signature Enabled; SteelHead bandwidth optimization
16+	ON	OFF	OFF	ON	Signature Enabled; SteelHead bandwidth optimization

Note: Rows with an asterisk (*) and a plus sign (+) are illegal combinations of **SSReq** and **SSEn** on the server and the workstation respectively.

This table represents behavior for Windows 2000 workstations and servers with service pack 3 and Critical Fix Q329170. Prior to the critical fix, the security signature feature was not enabled or enforced even on domain controllers.

Each computer has the following set of parameters: one set for the computer as a server and the other set for the computer as a client.

Note: On the client, if SMB signing is set to Required, do not disable it on the server. For the best performance, enable the clients, disable the file servers, and enable domain controllers.

The following procedures assume that you have installed and configured the SteelHeads in your network.

To disable SMB signing on Windows 2000 Domain Controllers, member servers, and clients

1. Open Active Directory Users and Computers on the Domain Controller.
2. Right-click **Domain Controllers** and select Properties.
3. Select the Group Policy tab.
4. Select Default Domain Controllers Policy and click **Edit**.
5. Select Default Domain Controllers Policy/Computer Configuration/Windows Settings/Security Settings/Local Policies/Security Options.
6. Disable Digitally sign client communication (always) and Digitally sign server communication (always).
7. Disable Digitally sign client communication (when possible) and Digitally sign server communication (when possible).
8. Reboot all the Domain Controllers and member servers that you want to optimize.

Note: You can also open a command prompt and enter **gpupdate.exe /Force** that forces the group policy you just modified to become active without rebooting.

You can verify that SMB signing has been disabled on your domain controllers, member servers, and clients. The following procedures assume that you have installed and configured the SteelHeads in your network.

To verify that SMB signing has been disabled

1. Copy some files in Windows from the server to the client through the SteelHeads.
2. Connect to the Management Console. For detailed information, see the *SteelHead User Guide*.
3. On the server-side SteelHead, choose Reports > Diagnostics: System Logs.
4. Look for the SMB signing warnings (in red). For example, look for the following text:

```
SFE: error=SMB_SHUTDOWN_ERR_SEC_SIG_ENABLED
```
5. If you see error messages, repeat **Step 6** and **Step 7** in procedure “To disable SMB signing on Windows 2000 Domain Controllers, member servers, and clients”.

To disable SMB signing on Windows 2003 Domain Controllers, member servers, and clients

1. Open Active Directory Users and Computers on the Domain Controller.
2. Right-click **Domain Controllers** and select Properties.
3. Select the Group Policy tab.
4. Click Default Domain Controllers Policy.
5. Click **Edit**.
6. Click Default Domain Controllers Policy/Computer Configuration/Windows Settings/Security Settings/Local Policies/Security Options.
7. Reboot all the Domain Controllers and member servers that you want to optimize.

Unavailable opportunistic locks

If a file is not optimized for more than one user at a time, it might be because an application lock on it prevents other applications and the SteelHead from obtaining exclusive access to it. Without an exclusive lock, the SteelHead cannot perform latency (for example, read-ahead and write-behind) optimization on the file.

Without opportunistic locks (**oplocks**), RiOS SDR and compression are performed on file contents, but the SteelHead cannot perform latency optimization because data integrity cannot be ensured without exclusive access to file data.

The following behaviors are signs of unavailable oplocks:

- Within a WAN:
 - A client, PC1, in a remote office across the WAN can open a file it previously opened in just a few seconds.
 - Another client, PC2, on the WAN has also previously opened the file but cannot open it quickly because PC1 has it open. While PC1 has the file open, it takes PC2 significantly longer to open the file.
 - When PC1 closes the file, PC2 can once again open it quickly. However, because PC2 has the file open, PC1 cannot open it quickly; it takes significantly longer for PC1 to open the file because PC2 has it open.
 - If no client has the file open, and PC1, PC2, and a third client on the WAN (PC3) simultaneously copy but do not open the file, each client can copy the file quickly and in nearly the same length of time.
- The results of a **tcpdump** show that WAN utilization is low for files that take a long time to open.
- In the Management Console, slow connections appear optimized.

Note: You can check connection bandwidth reduction in the Bandwidth Reduction report in the Management Console.

For similar problems, see:

- [“Unknown \(or unwanted\) SteelHead appears on the current connections list” on page 468](#)
- [“Server Message Block signed sessions” on page 471](#)

Solution: None needed

To prevent any compromise to data integrity, the SteelHead only accelerates access to data when exclusive access is available. When unavailable **oplocks** prevent the SteelHead from performing latency optimization, the SteelHead still performs RIOS SDR and compression on the data. Therefore, even without the benefits of latency optimization, SteelHeads might still increase WAN performance, but not as effectively as when application optimized connections are available.

Underutilized fat pipes

A *fat pipe* is a network that can carry large amounts of data without significantly degrading transmission speed. If you have a fat pipe that is not being fully utilized and you are experiencing WAN congestion, latency, and packet loss as a result of the limitations of regular TCP, consider the solutions outlined in this section.

Solution: Enable high-speed TCP

To better utilize fat pipes such as in GigE WANs, consider enabling high-speed TCP (HS-TCP). HS-TCP is a feature that you can enable on SteelHeads to ease WAN congestion caused by limitations with regular TCP that results in packet loss. Enabling the HS-TCP feature enables more complete utilization of *long fat pipes* (high-bandwidth, high-delay networks).

Important: We recommend that you enable HS-TCP only after you have carefully evaluated whether it will benefit your network environment. For detailed information about the trade-offs of enabling HS-TCP, see the **tcp highspeed enable** command in the *Riverbed Command-Line Interface Reference Manual*.

To display HS-TCP settings, use the **show tcp highspeed** command. To configure HS-TCP, use the **tcp highspeed enable** command. Alternatively, you can configure HS-TCP in the Management Console.

For details, see the *Riverbed Command-Line Interface Reference Manual* or the *SteelHead User Guide*.

MTU sizing

This section describes how SteelHeads work with PMTU Discovery and references RFC 1191 to negotiate Maximum Transmission Unit (MTU) sizing. This section includes the following topics:

- [“MTU issues” on page 477](#)
- [“Determining MTU size in deployments” on page 478](#)
- [“Connection-forwarding MTU considerations” on page 479](#)

The MTU specifies the largest datagram packet (Layer-3 packet) that a device supports. In SteelHead optimized environments, MTU sizing is typically automatic and not a concern. The default MTU size for a SteelHead is 1500 bytes, which is the standard for many client and networking devices, especially across WAN circuits.

For pass-through traffic for an in-path SteelHead without RSP, the SteelHead passes packets up to the supported packet size of the configured in-path MTU. The in-path MTU supports jumbo frame configuration. For 1-Gbps interface cards, the supported MTU is 9216 or 16110, and all 10-Gbps cards support 16110. For a full list of interface cards and their MTU support, go to <https://supportkb.riverbed.com/support/index?page=content&id=s14344>.

For optimized traffic, the SteelHeads act as a proxy. A separate inner TCP connection is established between SteelHeads, with a potentially different MTU size from the original client-to-server connection.

When a SteelHead detects that a session can be optimized, it initiates a TCP session to the remote SteelHead using the IP flag *don't fragment* with packet size up to the value configured in the interface MTU (default 1500 bytes). In line with RFC 1191, if a router or device along the TCP path of the session (possibly originating a GRE tunnel) does not support the packet size, and because it is not allowed to fragment the packet, it can request the originator (the SteelHead) to reduce the packet size. It does this with an ICMP type 3, code 4 (34) packet that carries the desired maximum size and the sequence number of the packet exceeding the router's interface MTU.

A common reason devices support less than 1500 bytes is the presence of GRE tunnels used to establish VPNs. The 24-byte overhead GRE incurs effectively gives the tunnel interface an MTU of 1476 bytes.

Similar to the Path MTU Discover (PMTUD) behavior for most clients and servers, the SteelHead reduces the packet size for a given session after it receives the ICMP message from the device with the lower MTU. According to RFC 1191, Section 6.3, the SteelHead tries to send larger packets every 10 minutes. For details, go to <http://www.faqs.org/rfcs/rfc1191.html>.

In environments that support PMTUD, we recommend that you leave the SteelHead MTU configuration to its default setting of 1500 bytes.

Note: In environments that support PMTUD, use the `ip rt-cache rebuilt-count 0` command on communicating SteelHeads (RiOS 8.0 and later).

For information about MTU and path selection, see “[MTU and MSS adjustment when using firewall path traversal](#)” on page 190.

MTU issues

In most cases two hosts dynamically negotiate path MTU. Networks that contain firewalls or tunnels (VPN, GRE, IPSec transport mode) between SteelHeads sometimes require manual tuning of the MTU values. Firewalls and tunnels interfere in the following ways:

- Firewalls can contain rules that explicitly prevent path MTU by blocking or not sending the ICMP type 3 packets, causing all attempts at dynamically negotiating MTU to fail.
- Tunnels require additional per-packet overhead to encapsulate data, reducing possible MTU size for connections being carried.

SteelHeads set the DF bit for inner channel communication to peer SteelHeads. If the device in the network path does not support the SteelHead packet size and also does not send an ICMP type 3 to notify the SteelHead to reduce packet size, the packet is dropped without the SteelHead knowing to reduce future packet sizes. This can result in poor optimization performance.

Determining MTU size in deployments

A simple method of determining MTU size across a path is to send *do not fragment* ping requests from the client PC, or client-side SteelHead, with varying packet sizes to a remote server or SteelHead. The following procedure shows the method from a windows client.

In the following example, a ping with the don't fragment (-f) and length (-l) 1500 bytes is sent to the remote server or SteelHead. This results in 100% loss with *Packet needs to be fragmented but DF set* in the reply. Pinging 10.0.0.1 with 1500 bytes of data:

```
C:\>ping -f -l 1500 10.0.0.1

Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.

Ping statistics for 10.0.0.1:

    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss)
```

Decrease the size of the packet to 1400 and repeat, pinging 10.0.0.1 with 1400 bytes of data:

```
C:\> ping -f -l 1400 10.0.0.1

Reply from 10.0.0.1: bytes=1400 time=222 ms TTL=251
Reply from 10.0.0.1: bytes=1400 time=205 ms TTL=251
Reply from 10.0.0.1: bytes=1400 time=204 ms TTL=251
Reply from 10.0.0.1: bytes=1400 time=218 ms TTL=251
Ping statistics for 10.0.0.1:

    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)

    Approximate round trip times in milliseconds:

        Minimum = 204 ms, Maximum = 222 ms, Average = 212 ms
```

This command gets the desired result. Repeat the ping test increasing or decreasing in increments of 10 or 20 until reaching the optimum value.

Note: Packet size of 1400 is shown only as an example, typical values can range from 1280 and higher.

When you specify the `-l <size>` command on a Windows machine, you are actually specifying the data payload, not the full IP datagram, including the IP and ICMP headers. To calculate the appropriate MTU, you must add the IP header (20 bytes) and ICMP header (8 bytes) to the Windows ping size. For example, for sending a 1400 byte payload, the SteelHead in-path MTU should be set to 1428 bytes. Although specifying ping sizes on Cisco routers, the specified size includes the IP and ICMP header (28 bytes). If using a Cisco device to test, set the MTU to the specified size; adding the 28 bytes manually is not necessary.

How to configure the in-path MTU value

1. Choose Networking > Networking: In-Path Interfaces, and expand the interface you want to edit.
2. Change the MTU value and apply the setting.

In RiOS 8.0 and later, the SteelHead does not pass through packets larger than the MTU value of its interfaces, nor does it send ICMP notifications to the sending host of the dropped packets. In environments in which the in-path MTU is lowered to account for a smaller MTU in the WAN network, we recommend that you use the **interface mtu-override enable** command.

Connection-forwarding MTU considerations

In networks in which SteelHeads are connection-forwarding neighbors, it is critical that you configure the LAN or WAN links that are expected to carry forwarded traffic so they can support the configured in-path MTU. Connection-forwarded traffic does not support PMTUD.

When forwarded packets are too large, ICMP Type 3, Code 4 messages are generated on intermediate routers are sent back to the sending client or server. The ICMP header does not match a TCP connection in the client or server, which causes poor optimization or failed connections. To prevent poor optimization or failed connections, make sure that you configure the interface MTUs on links carrying forwarded traffic the same size as the SteelHead in-path interface.

SteelHead and AppResponse Integration

This chapter provides a high-level overview of the SteelHead and AppResponse. It includes the following sections:

- “Overview of SteelHead and AppResponse integration” on page 481
- “AppResponse and SteelHead deployment scenarios” on page 483

This chapter assumes you are familiar with the AppResponse documentation listed on the Riverbed Support site, at <https://support.riverbed.com/content/support/software/steelcentral-npm/appresponse-appliance.html>.

Note: You must be running AppResponse 9.5 or later with RiOS 9.0 or later.

Overview of SteelHead and AppResponse integration

AppResponse provides real-time visibility of more than 60 TCP/IP metrics for each IP flow conversation between a client and server, using passive, IP flow-based monitoring. You typically deploy AppResponse in a data center and configure the appliance to use a network SPAN or tap that exposes it to IP traffic.

Prior to AppResponse 9.5, metrics from WAN-optimized network environments—such as SteelHead—were less reliable due to the acceleration and IP address translation that can occur when they pass through WAN-acceleration devices. AppResponse 9.5 or later communicates with SteelHeads running RiOS 9.0 or later that includes a Web Transaction Analysis (WTA) engine that provides extensive real-time monitoring and analysis of web applications.

You can view detailed data on all page views observed: response times for individual pages and objects (broken down by server versus network delays), slow pages, optimized versus nonoptimized content, user counts, view rates, page views by geographic region, HTTP response codes, and so on. SteelHeads and AppResponse use the following process:

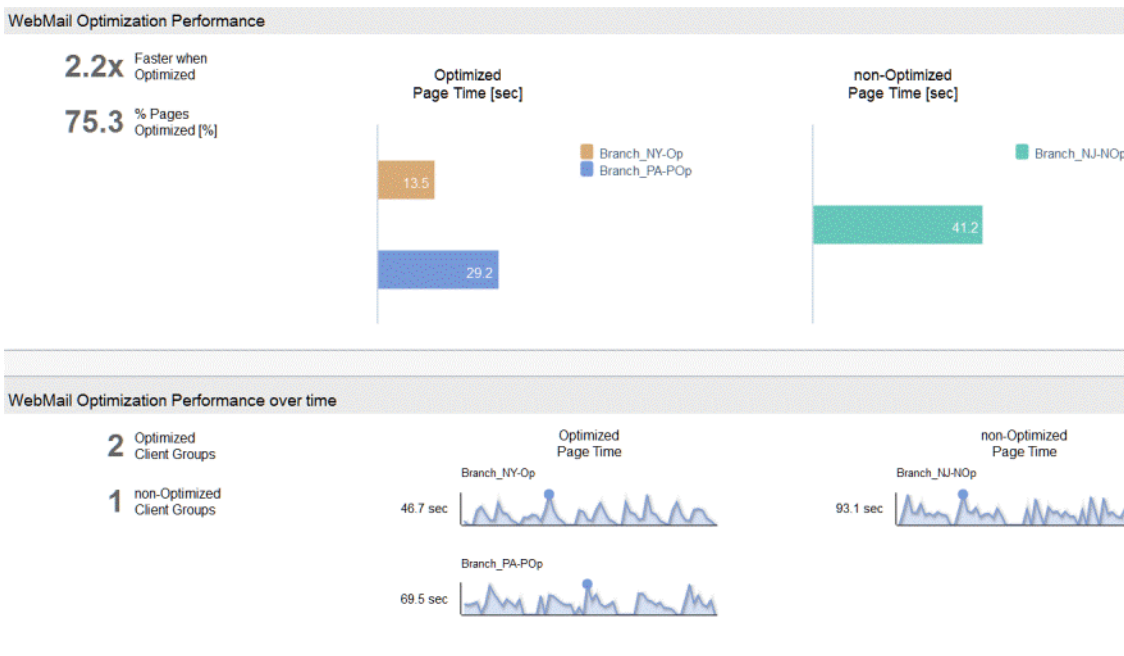
1. The SteelHead collects HTTP payload data and time stamps for the web objects it optimizes.
2. The SteelHead transfers the data to AppResponse (called *SteelFlow WTA*) through the REST API.
3. The WTA process in AppResponse coalesces the data into page views and then calculates metrics.

AppResponse can identify:

- the components of delay (network, server, client).
- the components that are accelerated.
- how well the components performed when rendered on the desktop device of the client.

This unique capability enables network and application engineers to accurately identify what is the source of traffic congestion when a user is viewing web pages at a data center or a cloud-hosted environment. Using AppResponse helps you to understand the positive impact the SteelHeads are having in their environment. **Figure 22-1** shows, on two different AppResponse reports, how you can identify what remote sites are benefiting the most from optimization and what performance they are receiving on average (Choose Quick Views > My SteelHeads > Top Optimized vs non-Optimized Client Groups for my Web App, and then select an application.)

Figure 22-1. Remote site WebMail performance



When you configure AppResponse and SteelHead to work together, you can extend the visibility of WTA from the data center to branch offices. You can evaluate the effects of SteelHead optimization of web pages between your data center and branch offices and between branch offices and cloud servers. Specifically you can:

- quantify end-user experience of your web applications at branch offices.
- compare branch office web application response times to each other.
- compare optimized branch office web application response time to each other.
- compare end-user experience in nonoptimized branch offices to optimized ones.
- compare user experience over time.
- measure optimization performance for business-critical applications.
- monitor and evaluate the performance of SaaS applications such as Salesforce.com.
- quickly identify web applications, pages, objects, sites, and users with high response times.
- break down HTTP response times into network versus application delays for specific pages, objects, and applications.
- tell how many users per site using this application (helps to find application adoption).

- tell what are the devices they are accessing application from (Windows, Mac).
- tell what are the browser type or versions.
- tell who are the top users.
- tell what are the slowest pages.
- tell what is the average user experience in terms of page load time.
- detect any bottlenecks

Figure 22-2 shows the actual end-user response time on the average for applications that are being optimized, and who is benefiting the most from the optimization. (Choose Quick Views > My SteelHeads > Top Optimized Client Groups and Web Apps.)

Figure 22-2. Optimization benefit



AppResponse and SteelHead deployment scenarios

This section describes the deployment scenarios available for AppResponse and SteelHead. It includes the following topics:

- “Data center deployment” on page 483
- “Cloud deployment” on page 486

Data center deployment

In the data center deployment scenario, AppResponse monitors optimization between branch offices and the data center, with SteelHeads in the data center and branch offices. One AppResponse can communicate with many SteelHeads, while one SteelHead can share SteelFlow WTA records with only one AppResponse.

Correct sizing for this configuration can be difficult. You want to fully understand the volume and type of traffic you want to monitor to ensure the AppResponse can efficiently process the traffic sent to it. AppResponse 9.5.2 and later (depending on the size) can receive and process 50,000 to 70,000 SteelFlow WTA records per minute from 1 to N number of SteelHeads. You must check the processing capacity and headroom of each AppResponse as you configure more SteelHeads to send SteelFlow WTA records. This check is performed on AppResponse by using the Appliance Health Check Insight, which indicates pass, fail, or warning for the volume of traffic it receives in each category (Figure 22-3).

Figure 22-3. Key performance metrics

Metric	Average	Max	95th Percentile	Performance Check
CPU Busy (Overall) [%]	5	15	9	Pass
Disk Busy [%]	0	28	2	Pass
Disk Array Status	-	-	-	Pass
Swap [%]	0	0	0	Pass
Packet Rate (before dedup) [#/sec]	43883	94636	81477	Pass
Packet Drops (Flow Engine) [%]	0	0	0	Pass
Packet Duplicates [%]	27	44	42	Pass
HSC Max Thread CPU [%]	0	3	2	Pass
HSC Usage (Flow Engine) [%]	4	7	6	Pass
HSC Usage (ASX) [%]	4	7	6	Pass
HSC Usage (VoIP) [%]	4	13	10	Pass
Flow Engine Max Thread CPU [%]	4	8	5	Pass
Dropped SteelFlow WTA Records [%]	0	0	0	Pass
WTA Dropped Requests (SteelFlow WTA) [%]	0	0	0	Pass
WTA Dropped Requests (span) [%]	0	0	0	Pass
WTA Request Rate (SteelFlow WTA) [#/min]	1080	46360	0	Pass
WTA Request Rate (span) [#/min]	0	0	0	Pass
WTA Request Rate (total) [#/min]	8600	13247	11982	Pass
WTA Abandoned Pairs [%]	0	2	1	Pass
WTA Page Rate [#/min]	6934	10825	8326	Pass
WTA Dropped Pages [%]	0	0	0	Pass

Design considerations:

- One AppResponse can talk to and receive SteelFlow WTA records from N number of SteelHeads
- A SteelFlow WTA record represents some portion of a client HTTP/HTTPS request-response pair, which is optimized by the SteelHead.
- AppResponse is limited by the aggregate total volume of SteelFlow WTA records received per minute. There is no hard limit number of SteelHeads an AppResponse can talk to.
- The ARX 9.5.2 and 9.0.3 performance tables contain sizing information.

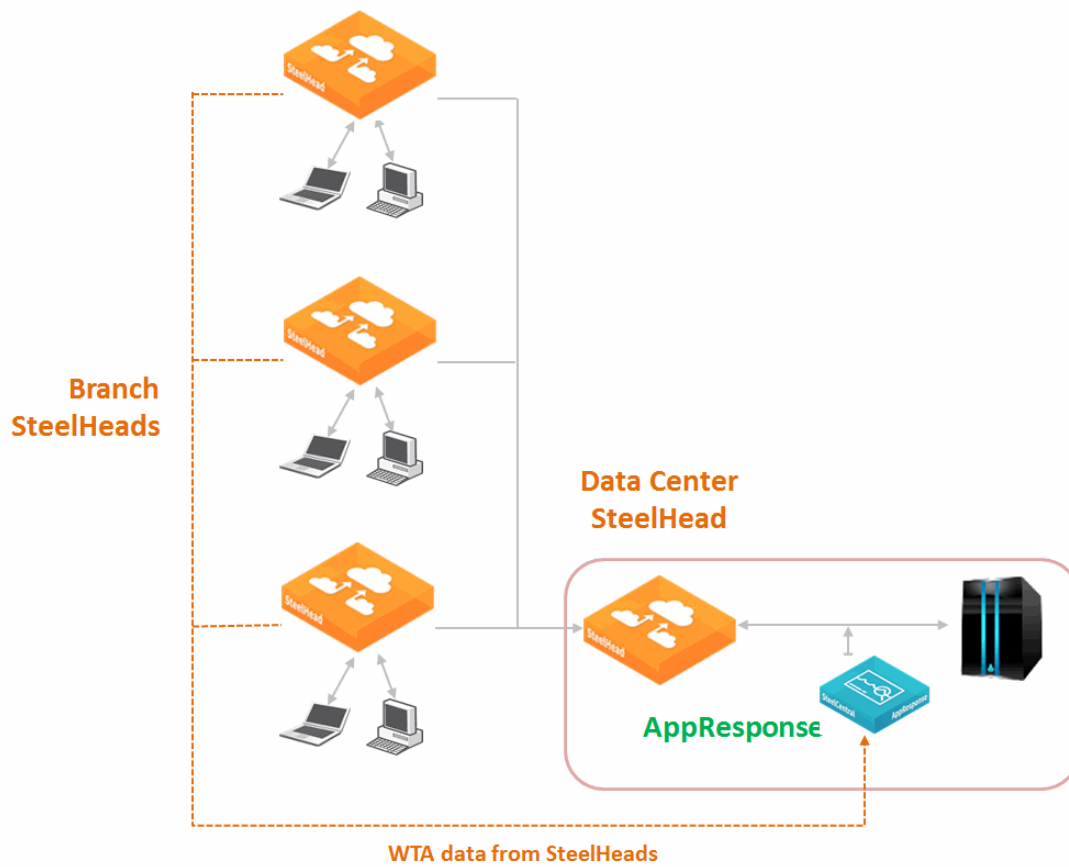
SteelFlow WTA collection is only supported on physical appliances. The smaller appliances can process 50K SteelFlow WTA records per minute and larger appliances can scale up to 70K SteelFlow WTA records per minute.

- We recommend that you add SteelHead and SteelFlow WTA configurations to an existing AppResponse incrementally while monitoring how close the AppResponse is to its maximum SteelFlow WTA processing limit using the Appliance Health Check Insight.
- A SteelFlow WTA record is a record of metrics about an HTTP/HTTPS request/response pair.

A web page GET generates many different requests for content on that page. On average a web page GET generates about 10 to 20 SteelFlow WTA records because 10 to 20 components of the page are requested and the server responds back with the requests.

Figure 22-4 shows one AppResponse in the data center, one SteelHead in the data center, and the other SteelHead in the branch office that you want to monitor.

Figure 22-4. Data center deployment

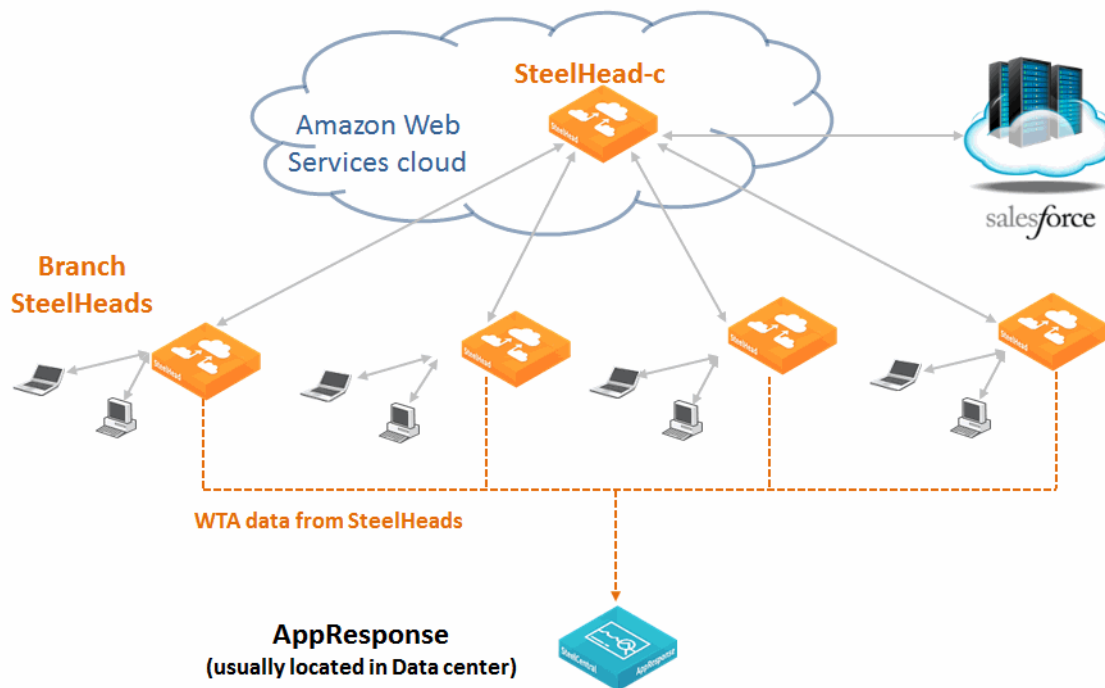


Cloud deployment

In the cloud deployment scenario, AppResponse monitors optimization between a branch office and a cloud server, with SteelHead-c and a SteelHead CX or EX in the branch offices.

Figure 22-5 shows one AppResponse in the data center, one SteelHead-c in the cloud, and the other SteelHead in the branch offices sending their SteelFlow WTA data records to an AppResponse, usually located in a data center.

Figure 22-5. Cloud deployment



Deploying SteelHead-v in OpenStack

This chapter describes how to download, create, and deploy a new SteelHead (virtual edition) (SteelHead-v) appliance in an OpenStack environment and includes the following sections:

- [“What you need” on page 487](#)
- [“Deploying SteelHead-v appliances in OpenStack” on page 487](#)
- [“Using a Heat template to deploy SteelHead-v appliances” on page 496](#)

What you need

Knowledge requirements

You must be familiar with the deployment options described in the *SteelHead (Virtual Edition) Installation Guide*.

Software requirements

You must use a next-generation virtual SteelHead VCX software image in KVM format. CX series SteelHead-v images are not supported.

Deployment guidelines

Do not use Layer 2 redirection mechanisms in your OpenStack deployment. As with many public clouds, OpenStack is a Layer 3 environment, and there are no Layer 2 redirections available. Devices can only receive traffic if traffic is directly forwarded to their IP addresses.

If you use a Heat template to deploy the VCX, see [“Heat template deployment guidelines” on page 496](#) for additional configuration recommendations.

Deploying SteelHead-v appliances in OpenStack

To deploy SteelHead-v appliances in an OpenStack environment

1. Go to the support.riverbed.com site and download the image named Next Generation Virtual SteelHead VCX Software Image (KVM). Do not download an image for a CX model.

2. Extract the files in the image by entering the **tar** CLI command.

In this CLI example, the .tar file image name is image_rbt_sh_9_7_0_n6_ng_x86_64.tgz.

```
$ tar -zxvf image_rbt_sh_9_7_0_n6_ng_x86_64.tgz
mgmt.qcow2
install.sh
riverbed_model_tmp
all_bypass_pci
```

3. Using Secure Copy (SCP) or another secure copy utility, upload the extracted mgmt.qcow2 image to the OpenStack client workstation.
4. Import the mgmt.qcow2 image into the image service using the **glance image-create** CLI command.

This example shows how to create an image named NextGen Virtual Steelhead 9.7.

```
$ glance image-create --name "NextGen Virtual SteelHead 9.7" --container-format bare --disk-format qcow2 < mgmt.qcow2
```

Property	Value
checksum	b192306c2d7b459e57063d9b93cba4e9
container_format	bare
created_at	2017-06-12T20:36:02Z
disk_format	qcow2
id	304c7ee3-62fb-4881-995d-8a4d8f960bbc
min_disk	0
min_ram	0
name	NextGen Virtual Steelhead 9.7
owner	2091e12a67f844bf86a0fa22965d393d
protected	False
size	2680881152

5. Determine the memory requirements for the VCX model by opening and viewing the riverbed_model_tmp file using the **cat** CLI command.

The riverbed_model_tmp file shows the following information in this order:

- Model
- Model description
- Number of vCPUs
- RAM
- Management HDD size
- DataStore1 HDD Size
- DataStore2 HDD Size

This example shows the output of the `riverbed_model_tmp` file.

```
$ cat riverbed_model_tmp
.
.
.
VCX10 = New-Model 1 2048MB 20GB 50GB 50GB;
VCX20 = New-Model 1 2048MB 20GB 80GB 80GB;
VCX30 = New-Model 1 2048MB 20GB 100GB 100GB;
VCX40 = New-Model 2 4096MB 26GB 150GB 150GB;
VCX50 = New-Model 4 8192MB 38GB 400GB 400GB;
VCX60 = New-Model 4 8192MB 38GB 400GB 400GB;
VCX70 = New-Model 6 24576MB 70GB 800GB 80GB;
VCX80 = New-Model 12 32768MB 86GB 1600GB 160GB;
VCX90 = New-Model 24 49152MB 118GB 2240GB 160GB;
VCX100 = New-Model 32 65536MB 160GB 3600GB 300GB;
VCX110 = New-Model 44 131072MB 300GB 4800GB 300GB;
```

6. Create an OpenStack flavor based on the requirements of the license. Virtual hardware templates are called flavors in OpenStack.

If you have already created a flavor, skip to [Step 7 on page 490](#).

- To create an OpenStack flavor using the GUI, select the System tab, select the Flavors category, click **Create Flavor**, and enter values for the VCX appliance in the Flavor Information tab.

Enter the RAM, Root Disk, Ephemeral Disk, and Swap Disk values based on the requirements for the VCX model. This figure shows a flavor being created for a VCX10.

Figure 23-1. Creating an OpenStack flavor using the GUI

The screenshot shows a 'Create Flavor' dialog box with two tabs: 'Flavor Information' (active) and 'Flavor Access'. The 'Flavor Information' tab contains the following fields and values:

- Name ***: vcx10
- VCPUs ***: 1
- RAM (MB) ***: 2048
- Root Disk (GB) ***: 20
- Ephemeral Disk (GB)**: 20
- Swap Disk (MB)**: 0

At the bottom right, there are 'Cancel' and 'Save' buttons. A mouse cursor is visible over the 'Save' button.

- To create an OpenStack flavor using the CLI, see [“Creating OpenStack flavors: CLI examples” on page 506](#).

7. Create networks for the appliance. Create one network for each network interface.

- To create networks using the OpenStack GUI, choose System > Networks, click Create Network, and enter the IP address and subnet values for each network.

This figure shows four networks being created: one network each for the primary, WAN, LAN, and auxiliary (aux) network interfaces.

Figure 23-2. Creating networks for SteelHead-v network interfaces

Project	Network Name	Subnets Associated	DHCP Agents	Shared	Status	Admin State	Actions
demo	primary	primary 192.168.1.0/24	1	Yes	Active	UP	Edit Network
admin	public	ip6-public-subnet 2001:db8::/64 public-subnet 172.24.4.0/24	1	No	Active	UP	Edit Network
demo	private	ip6-private-subnet fdd:fa7:5f53::/64 private-subnet 10.0.0.0/24	1	No	Active	UP	Edit Network
demo	wan	wan 192.168.4.0/24	1	Yes	Active	UP	Edit Network
demo	lan	lan 192.168.3.0/24	1	Yes	Active	UP	Edit Network
demo	public	-	0	Yes	Active	UP	Edit Network
demo	aux	aux 192.168.2.0/24	1	Yes	Active	UP	Edit Network

- To create networks using the CLI, continue to [Step 8](#). You create networks at the same time as you create the instance when using the CLI.

You can also use a Heat template to create the interfaces as shown in [“Using a Heat template to deploy SteelHead-v appliances” on page 496](#).

8. Create an OpenStack instance using either the Horizon dashboard or OpenStack CLI.

- To create an OpenStack instance from the Horizon dashboard, complete these steps:
 - Expand Compute under the Project topic and click **Image**.
 - Select the SteelHead-v KVM image.
 - Click **Launch Instance** in the Actions column.

Figure 23-3. Launching an OpenStack instance using Horizon GUI

Image Name	Type	Status	Public	Protected	Format	Size	Actions
NextGen Virtual Steelhead 9.7	Image	Active	Yes	No	QCOW2	1.3 GB	Launch Instance

- Select the Details tab in the Launch Instance window and enter the VCX flavor and image name in the Flavor and Instance Boot Source fields.

Figure 23-4. Specifying parameters for the instance using the Horizon GUI

Launch Instance

Details * Access & Security Networking * Post-Creation Advanced Options

Availability Zone
nova

Instance Name
ngVCX-A

Flavor ⓘ
vcx10
Some flavors not meeting minimum image requirements have been disabled.

Instance Count ⓘ
1

Instance Boot Source ⓘ
Boot from image

Image Name
NextGen Virtual Steelhead 9.7

Specify the details for launching an instance.
The chart below shows the resources used by this project in relation to the project's quotas.

Flavor Details

Name	sh.V150M
VCPUs	1
Root Disk	30 GB
Ephemeral Disk	44 GB
Total Disk	74 GB
RAM	1,024 MB

Project Limits

Number of Instances 0 of 10 Used

Number of VCPUs 0 of 20 Used

Total RAM 0 of 51,200 MB Used

Cancel Launch

- Select the Networking tab and associate the network interfaces with the networks you created.

This example shows how to associate networks for the Primary (NIC 1), Aux (NIC 2), LAN (NIC 3), and WAN (NIC 4) interfaces.

Figure 23-5. Associating the networks with the network interfaces

Launch Instance

Details * Access & Security **Networking *** Post-Creation Advanced Options

Selected networks

- NIC:1 ↕ primary (aa18e8f4-82e3-4b7f-a246-5a649da4a7a7) -
- NIC:2 ↕ aux (82c589b9-cd33-47b8-8d50-c51b28dcdb0b) -
- NIC:3 ↕ lan (8c96aa5c-4442-4602-94cc-30479eb8c89b) -
- NIC:4 ↕ wan (5a24c2f1-8df9-484f-84f7-6c9936b63ad1) -

Available networks

- ↕ primary (79cc026b-27ce-4bf5-9e89-4582b0ac780a) +

Choose network from Available networks to Selected networks by push button or drag and drop, you may change NIC order by drag and drop as well.

Cancel Launch

- Select the Post-Creation tab, and then create a script to add a primary and auxiliary IP address, network mask, default gateway, DNS server address, domain name, and administrative password.

This example shows how to create a SteelHead-v with an IP address of 192.168.1.2/24, an auxiliary address of 10.0.0.2/16, a default gateway of 10.0.0.1, a DNS server address of 10.16.32.48, and a domain name of nbttech.com.

Figure 23-6. Creating a post-creation script for the instance

Launch Instance

Details * Access & Security Networking * **Post-Creation** Advanced Options

Customization Script Source: Direct Input

Script Data ②

```
nxbd_sh:
primary_ipv4: 192.168.1.2
primary_masklen: 24
aux_ipv4: 10.0.0.2
aux_masklen: 16
default_gw: 10.0.0.1
name_server: 10.16.32.48
domain_name: nbttech.com
admin_password: $1$xcuHq/$a/qZ8zGpzy.NHsKjJ8Yla.
```

You can customize your instance after it has launched using the options available here.

"Customization Script" is analogous to "User Data" in other systems.

Cancel Launch

- Click **Launch**.
- To create the OpenStack instance using the CLI, either use a Heat template to create the interfaces as shown in [“Using a Heat template to deploy SteelHead-v appliances” on page 496](#), or enter commands similar to the commands shown in this example.

Note: Entering the CLI in this example creates a single network interface and limits your SteelHead-v deployment type to out-of-path only. With out-of-path deployments, the primary interface is used to receive and optimize traffic. Use the autodiscovery process or fixed-target in-path rules for this configuration.

This example shows how to create a network interface for a VCX10; change the flavor if you are configuring another VCX model.

```
$ openstack server create --image "NextGen Virtual Steelhead 9.7" --flavor vcx10 --nic net-id=d23be009-7340-41a3-83e0-734775b81407 ngVCX-A
```

```
+-----+
| Field                                     | Value -----+
| OS-DCF:diskConfig                       | MANUAL |
| OS-EXT-AZ:availability_zone             | |
| OS-EXT-SRV-ATTR:host                    | None |
| OS-EXT-SRV-ATTR:hypervisor_hostname     | None |
| OS-EXT-SRV-ATTR:instance_name           | |
| OS-EXT-STS:power_state                   | NOSTATE |
| OS-EXT-STS:task_state                   | scheduling |
| OS-EXT-STS:vm_state                     | building |
| OS-SRV-USG:launched_at                  | None |
| OS-SRV-USG:terminated_at                | None |
| accessIPv4                              | |
| accessIPv6                              | |
| addresses                               | |
| adminPass                               | ZGF84SuKbTgp |
| config_drive                            | |
| created                                 | 2017-06-13T18:13:13Z |
| flavor                                  | vcx10 (ffba79ff-2715-4bd1-a081-0bfc304dde7a) |
| hostId                                  | |
| id                                       | 855f0253-c502-46c0-8ae0-c08547446c95 |
| image                                   | NextGen Virtual Steelhead 9.7 (304c7ee3-62fb-4881-995d-8a4d8f960bbc) |
| key_name                                | None |
| name                                    | ngVCX-A |
| os-extended-volumes:volumes_attached   | [] |
| progress                                | 0 |
| project_id                              | 53ad8fb1ddd1457aa68d14f531f07616 |
| properties                              | |
| security_groups                         | [{u'name': u'default'}] |
| status                                  | BUILD |
| updated                                 | 2017-06-13T18:13:13Z |
| user_id                                 | 70bcd75fd6bf4553a15fc84b2f5454b2 |
+-----+
```

9. Create a datastore volume by entering the **openstack volume create** command.

This example shows how to create a datastore called `vcx_1` with a size of 50 gigabytes (GB), which is sufficient for a VCX10.

```
$ openstack volume create --size 50 vcx_1
```

Field	Value
attachments	[]
availability_zone	nova
bootable	false
consistencygroup_id	None
created_at	2017-06-12T21:00:41.038935
description	None
encrypted	False
id	6c82f307-7489-43d5-98fc-00353c6ff479
migration_status	None
multiattach	False
name	vcx_1
properties	
replication_status	disabled
size	50
snapshot_id	None
source_volid	None
status	creating
type	None
updated_at	None
user_id	70bcd75fd6bf4553a15fc84b2f5454b2

10. Attach the datastore volume to the running instance by entering the **openstack server add volume** command.

This example shows how to add the datastore volume `vcx_1` to the instance name `ngVCX-A`.

```
$ openstack server add volume ngVCX-A vcx_1
```

11. Restart the virtual machine.

12. Deploy the SteelHead-v appliance into the network.

See the “Deployment Options” section in the “Overview of SteelHead-v” chapter of the *SteelHead (Virtual Edition) Installation Guide* for more information.

Deleting an OpenStack instance

To delete an OpenStack instance, enter this command:

```
openstack server delete <instance-name>
```

For example, to delete the `ngVCX-A` instance created for the examples in this chapter, enter the `openstack server delete ngVCX-A` command.

Using a Heat template to deploy SteelHead-v appliances

OpenStack lets you create network interfaces between devices using the Neutron networking service. You use a Heat Orchestration Template (HOT) to create the required Neutron interfaces for devices in your network, including but not limited to the SteelHead-v appliances.

The Heat template is a YAML (YAML Ain't Markup Language) file and has a file type of .yaml or .yml.

Heat template deployment guidelines

- **SteelHead-v cabling guidelines** - Cable your network so that all traffic passes through the SteelHead-v appliances for end-to-end connectivity. Cabling an alternative route might cause traffic to bypass the SteelHead-v appliances.
- **Port security guidelines** - Disable port security on all SteelHead LAN and WAN ports, all firewall or other security appliance ports, and the ports of any device that performs routing functions in your network. Alternatively, configure allowed IP pairs for the ports of all SteelHead, firewall, and routing devices. This configuration is required for any traffic that passes through an OpenStack virtual machine (VM) to a destination that is not on the VM.

By default, OpenStack enables port security and security groups. OpenStack's port security is based on a Layer 2 MAC address, and security groups filter traffic based on a Layer 3 IP address and Layer 4 port number. When traffic passes through a device (such as a ping packet through a SteelHead), Layer 2 and Layer 3 packet headers are not modified, which could trigger port security and security group alerts.

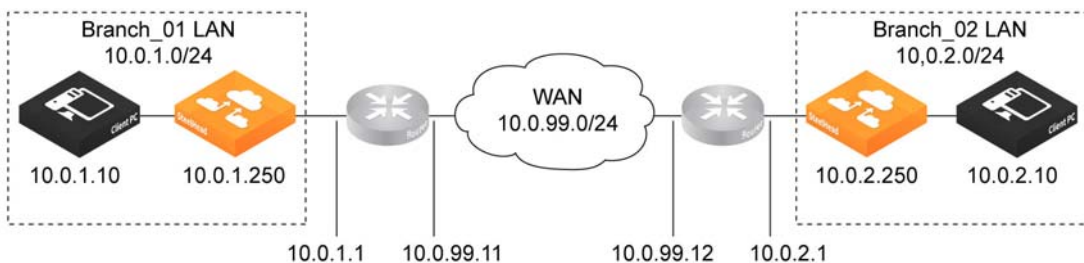
Disabling port security at a port level allows the VM to send and receive traffic to and from any destination, which is the standard configuration for optimization or firewall devices.

- **Routing guidelines** - If you have any routes between trusted and untrusted sites that use Layer 2 addresses, delete those routes; otherwise, the network can have connectivity issues. Instead, allow OpenStack to use Address Resolution Protocol (ARP) to find Layer 2 destinations.

Heat template example

This example provides a networking environment for the LAN and WAN deployment shown in this figure.

Figure 23-7. Network diagram for sample Heat template



Sample Heat template .yaml file

```
heat_template_version: '2015-04-30'

description: |
  Basic inpath setup.
```


branch_01				branch_02	
lan	wan	WAN	wan	lan	
10.0.1.0/24		10.0.99.0/24		10.0.2.0/24	
client	--- SteelHead	--- Router	---- Router	--- SteelHead	--- client
10.0.1.10	10.0.1.250	10.0.1.1	10.0.2.1	10.0.2.250	10.0.2.10
		10.0.99.11	10.0.99.12		

All devices have their first interface on a mgmt network (10.0.0.0/24) with an IP assigned by Neutron. Floating IPs are also allocated as a convenience.

SteelHeads also have an interface on an aux network (10.0.10.0/24)

parameters:

```
external_network:
  type: string
  description: External network for public IP.
  constraints:
    - custom_constraint: neutron.network
  default: "public"
key_name:
  type: string
  description: An existing key pair to use for authentication to Client VMs
  constraints:
    - custom_constraint: nova.keypair
client_image:
  type: string
  description: >
    Image to use for clients. This template assumes the image uses
    predictable network naming (ens3, ens4, etc) and has cloud-init installed.
  constraints:
    - custom_constraint: glance.image
client_flavor:
  type: string
  description: Flavor to use for clients
  constraints:
    - custom_constraint: nova.flavor
  default: "m1.small"
sh_image:
  type: string
  description: Image to use for SteelHeads
  constraints:
    - custom_constraint: glance.image
sh_flavor:
  type: string
  description: Flavor to use for SteelHeads
  constraints:
    - custom_constraint: nova.flavor
sh_datastore_size:
  type: number
  description: Datastore size of SteelHead
  default: 50
```

resources:

```
#### Utility Networks
# mgmt: All devices get mgmt for remote access and configuration
mgmt_net:
  type: OS::Neutron::Net
```

```

mgmt_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: mgmt_net }
    cidr: 10.0.0.0/24
    dns_nameservers:
      - 8.8.8.8

# SteelHeads have aux interface
aux_net:
  type: OS::Neutron::Net

aux_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: aux_net }
    cidr: 10.0.10.0/24

# A WAN network between several provider edge routers.
wan:
  type: OS::Neutron::Net

wan_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: wan }
    cidr: 10.0.99.0/24

### Management router (for Internet and Floating IP access)

router_mgmt:
  type: OS::Neutron::Router
  properties:
    external_gateway_info:
      network: { get_param: external_network }

router_mgmt_interface:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: router_mgmt }
    subnet: { get_resource: mgmt_subnet }

#### Branch 01

### LAN L3 segment is split into two L2 segments bridged by the SteelHead inpath:
#   _lan for client instances
#   _wan for the uplink to wan via a router
#
#   In this topology's test networks, DHCP and gateway addresses are
#   disabled; every device must be configured with static addresses and
#   routes.
#   The mgmt interfaces do use DHCP however.
#
branch_01_lan:
  type: OS::Neutron::Net

branch_01_lan_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: branch_01_lan }
    cidr: 10.0.1.0/24
    gateway_ip: null
    enable_dhcp: False

```

```

branch_01_wan:
  type: OS::Neutron::Net

branch_01_wan_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: branch_01_wan }
    cidr: { get_attr: [ branch_01_lan_subnet, cidr ] }
    gateway_ip: null
    enable_dhcp: False

### Client VM

client_branch_01_ens3:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: mgmt_net }

client_branch_01_ens4:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_01_lan }
    fixed_ips:
      - ip_address: 10.0.1.10

client_branch_01:
  type: OS::Nova::Server
  properties:
    image: { get_param: client_image }
    flavor: { get_param: client_flavor }
    key_name: { get_param: key_name }
    networks:
      - port: { get_resource: client_branch_01_ens3 }
      - port: { get_resource: client_branch_01_ens4 }
    user_data_format: RAW
    # Use cloud config to bring up ens4 with the correct IP and route.
    # NB: This is not persistent across reboots; it will only apply
    # on first boot.
    user_data: |
      #cloud-config
      runcmd:
        - ip link set ens4 up
        - ip addr add 10.0.1.10/24 dev ens4
        - ip route add 10.0.2.0/24 via 10.0.1.1
        - ip route add 10.0.99.0/24 via 10.0.1.1

### SteelHead

steelhead_branch_01_primary:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: mgmt_net }

steelhead_branch_01_aux:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: aux_net }

```

```

steelhead_branch_01_lan0_0:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_01_lan }
    fixed_ips:
      - ip_address: 10.0.1.250
      # The inpath interfaces pass through traffic from client VMs
      # without modifying the src/dst MAC address. OpenStack Neutron
      # port security always enables rules to prevent MAC address
      # spoofing, so port security must be disabled for inpath
      # interfaces.
    port_security_enabled: False

steelhead_branch_01_wan0_0:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_01_wan }
    fixed_ips:
      - ip_address: { get_attr: [steelhead_branch_01_lan0_0, fixed_ips, 0, ip_address] }
      # The inpath interfaces pass through traffic from client VMs
      # without modifying the src/dst MAC address. OpenStack Neutron
      # port security always enables rules to prevent MAC address
      # spoofing, so port security must be disabled for inpath
      # interfaces.
    port_security_enabled: False

steelhead_branch_01:
  type: OS::Nova::Server
  properties:
    image: { get_param: sh_image }
    flavor: { get_param: sh_flavor }
    # Steelhead networks must be primary, aux, lan, wan [, lan, wan, ...]
    networks:
      - port: { get_resource: steelhead_branch_01_primary }
      - port: { get_resource: steelhead_branch_01_aux }
      - port: { get_resource: steelhead_branch_01_lan0_0 }
      - port: { get_resource: steelhead_branch_01_wan0_0 }

sh_branch_01_cinder_volume:
  properties:
    availability_zone: nova
    size: {get_param: sh_datastore_size}
  type: OS::Cinder::Volume

sh_branch_01_volume_attachment:
  properties:
    instance_uuid: {get_resource: steelhead_branch_01}
    mountpoint: /dev/vdb
    volume_id: {get_resource: sh_branch_01_cinder_volume}
  type: OS::Cinder::VolumeAttachment

### Provider Edge Branch 01 Router

pe_branch_01_mgmt:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: mgmt_net }

```

```

pe_branch_01_branch:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_01_wan }
    fixed_ips:
      - ip_address: 10.0.1.1
      # ARP packets may be sent on behalf of other addresses, which may get
      # filtered by the implicit MAC + IP security rule. To avoid this,
      # disable port security entirely. An alternative that would allow
      # other security groups to be used on the routers that might work:
      #   allowed_address_pairs:
      #     - ip_address: 0.0.0.0/0
      # NB. This isn't required for neutron routers, but setting it if the
      # router is replaced by a VM based software router.
    port_security_enabled: False

pe_branch_01_wan:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: wan }
    fixed_ips:
      - ip_address: 10.0.99.11
      # ARP packets may be sent on behalf of other addresses, which may get
      # filtered by the implicit MAC + IP security rule. To avoid this,
      # disable port security entirely. An alternative that would allow
      # other security groups to be used on the routers that might work:
      #   allowed_address_pairs:
      #     - ip_address: 0.0.0.0/0
      # NB. This isn't required for neutron routers, but setting it if the
      # router is replaced by a VM based software router.
    port_security_enabled: False

# Note: This router is using the default Neutron implementation,
# but the template explicitly creates the ports so it can easily be
# switched out with a VM-based software router from another vendor.
# The below resources would be replaced with a OS::Nova::Server.
# For example:
#   pe_branch_01:
#     type: OS::Nova::Server
#     properties:
#       image: { get_param: router_image }
#       flavor: { get_param: router_flavor }
#       networks:
#         - port: { get_resource: pe_branch_01_mgmt }
#         - port: { get_resource: pe_branch_01_branch }
#         - port: { get_resource: pe_branch_01_wan }

pe_branch_01:
  type: OS::Neutron::Router

pe_branch_01_mgmt_int:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: pe_branch_01 }
    port: { get_resource: pe_branch_01_mgmt }

pe_branch_01_branch_int:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: pe_branch_01 }
    port: { get_resource: pe_branch_01_branch }

```

```

pe_branch_01_wan_int:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: pe_branch_01 }
    port: { get_resource: pe_branch_01_wan }

# Next hop to Branch 02 is the wan side of pe_branch_02.
pe_branch_01_route_branch_02:
  type: OS::Neutron::ExtraRoute
  depends_on: pe_branch_01_wan_int
  properties:
    router_id: { get_resource: pe_branch_01 }
    destination: { get_attr: [branch_02_lan_subnet, cidr] }
    nexthop: { get_attr: [pe_branch_02_wan, fixed_ips, 0, ip_address] }

#### Branch 02

### LAN L3 segment is split into two L2 segments bridged by the SteelHead inpath:
#   _lan for client instances
#   _wan for the uplink to wan via a router
#
#   In this topology's test networks, DHCP and gateway addresses are
#   disabled, every device must be configured with static addresses and
#   routes.
#
branch_02_lan:
  type: OS::Neutron::Net

branch_02_lan_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: branch_02_lan }
    cidr: 10.0.2.0/24
    gateway_ip: null
    enable_dhcp: False

branch_02_wan:
  type: OS::Neutron::Net

branch_02_wan_subnet:
  type: OS::Neutron::Subnet
  properties:
    network_id: { get_resource: branch_02_wan }
    cidr: { get_attr: [ branch_02_lan_subnet, cidr ] }
    gateway_ip: null
    enable_dhcp: False

### Client VM

client_branch_02_ens3:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: mgmt_net }

client_branch_02_ens4:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_02_lan }
    fixed_ips:
      - ip_address: 10.0.2.10

```

```

client_branch_02:
  type: OS::Nova::Server
  properties:
    image: { get_param: client_image }
    flavor: { get_param: client_flavor }
    key_name: { get_param: key_name }
    networks:
      - port: { get_resource: client_branch_02_ens3 }
      - port: { get_resource: client_branch_02_ens4 }
    user_data_format: RAW
    # Use cloud config to bring up ens4 with the correct IP and route.
    # NB: This is not persistent across reboots, will only apply on first boot.
    user_data: |
      #cloud-config
      runcmd:
        - ip link set ens4 up
        - ip addr add 10.0.2.10/24 dev ens4
        - ip route add 10.0.1.0/24 via 10.0.2.1
        - ip route add 10.0.99.0/24 via 10.0.2.1

### SteelHead

steelhead_branch_02_primary:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: mgmt_net }

steelhead_branch_02_aux:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: aux_net }

steelhead_branch_02_lan0_0:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_02_lan }
    fixed_ips:
      - ip_address: 10.0.2.250
      # The inpath interfaces pass through traffic from client VMs
      # without modifying the src/dst MAC address. OpenStack Neutron
      # port security always enables rules to prevent MAC address
      # spoofing, so port security must be disabled for inpath
      # interfaces.
      port_security_enabled: False

steelhead_branch_02_wan0_0:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_02_wan }
    fixed_ips:
      - ip_address: { get_attr: [steelhead_branch_02_lan0_0, fixed_ips, 0, ip_address] }
      # The inpath interfaces pass through traffic from client VMs
      # without modifying the src/dst MAC address. OpenStack Neutron
      # port security always enables rules to prevent MAC address
      # spoofing, so port security must be disabled for inpath
      # interfaces.
      port_security_enabled: False

```

```

steelhead_branch_02:
  type: OS::Nova::Server
  properties:
    image: { get_param: sh_image }
    flavor: { get_param: sh_flavor }
    # Steelhead networks must be primary, aux, lan, wan.
    networks:
      - port: { get_resource: steelhead_branch_02_primary }
      - port: { get_resource: steelhead_branch_02_aux }
      - port: { get_resource: steelhead_branch_02_lan0_0 }
      - port: { get_resource: steelhead_branch_02_wan0_0 }

sh_branch_02_cinder_volume:
  properties:
    availability_zone: nova
    size: {get_param: sh_datastore_size}
  type: OS::Cinder::Volume

sh_branch_02_volume_attachment:
  properties:
    instance_uuid: {get_resource: steelhead_branch_02}
    mountpoint: /dev/vdb
    volume_id: {get_resource: sh_branch_02_cinder_volume}
  type: OS::Cinder::VolumeAttachment

#### Provider Edge Branch 02 Router

pe_branch_02_mgmt:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: mgmt_net }

pe_branch_02_branch:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: branch_02_wan }
    fixed_ips:
      - ip_address: 10.0.2.1
      # ARP packets may be sent on behalf of other addresses, which may get
      # filtered by the implicit MAC + IP security rule. To avoid this,
      # disable port security entirely. An alternative that would allow
      # other security groups to be used on the routers that might work:
      #   allowed_address_pairs:
      #     - ip_address: 0.0.0.0/0
      # NB. This isn't required for neutron routers, but setting it if the
      # router is replaced by a VM based software router.
    port_security_enabled: False

pe_branch_02_wan:
  type: OS::Neutron::Port
  properties:
    network_id: { get_resource: wan }
    fixed_ips:
      - ip_address: 10.0.99.12
      # ARP packets may be sent on behalf of other addresses, which may get
      # filtered by the implicit MAC + IP security rule. To avoid this,
      # disable port security entirely. An alternative that would allow
      # other security groups to be used on the routers that might work:
      #   allowed_address_pairs:
      #     - ip_address: 0.0.0.0/0
      # NB. This isn't required for neutron routers, but setting it if the
      # router is replaced by a VM based software router.
    port_security_enabled: False

```



```

pe_branch_02:
  type: OS::Neutron::Router

pe_branch_02_mgmt_int:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: pe_branch_02 }
    port: { get_resource: pe_branch_02_mgmt }

pe_branch_02_branch_int:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: pe_branch_02 }
    port: { get_resource: pe_branch_02_branch }

pe_branch_02_wan_int:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: pe_branch_02 }
    port: { get_resource: pe_branch_02_wan }

pe_branch_02_route_branch_01:
  type: OS::Neutron::ExtraRoute
  depends_on: pe_branch_02_wan_int
  properties:
    router_id: { get_resource: pe_branch_02 }
    destination: { get_attr: [branch_01_lan_subnet, cidr] }
    nexthop: { get_attr: [pe_branch_01_wan, fixed_ips, 0, ip_address] }

#### Floating IPs

client_branch_01_ip:
  type: OS::Neutron::FloatingIP
  depends_on:
    - router_mgmt_interface
  properties:
    floating_network: { get_param: external_network }
    port_id: { get_resource: client_branch_01_ens3 }

client_branch_02_ip:
  type: OS::Neutron::FloatingIP
  depends_on:
    - router_mgmt_interface
  properties:
    floating_network: { get_param: external_network }
    port_id: { get_resource: client_branch_02_ens3 }

steelhead_branch_01_ip:
  type: OS::Neutron::FloatingIP
  depends_on:
    - router_mgmt_interface
  properties:
    floating_network: { get_param: external_network }
    port_id: { get_resource: steelhead_branch_01_primary }

steelhead_branch_02_ip:
  type: OS::Neutron::FloatingIP
  depends_on:
    - router_mgmt_interface
  properties:
    floating_network: { get_param: external_network }
    port_id: { get_resource: steelhead_branch_02_primary }

```

Creating an OpenStack stack from a Heat template

To create an OpenStack stack from a Heat template, enter this command, where `<heat_template.yaml>` is the name of Heat template and `<stack_name>` is the name of the stack to create.

```
openstack stack create -t <heat_template.yaml> <stack_name>
```

Deleting an OpenStack stack created by a Heat template

To delete an OpenStack stack created by a Heat template, first delete the VMs (servers) associated with the SteelHead-v appliances in the Heat template.

Use these commands to delete an OpenStack stack created by a Heat template, where `<steelhead_name>` is the name of the SteelHead (virtual edition) appliance and `<stack_name>` is the name of the stack to delete.

```
openstack server delete <steelhead_name>
openstack stack delete <stack_name>
```

If there are multiple SteelHead (virtual edition) appliances in the Heat template, delete them all before deleting the stack.

Creating OpenStack flavors: CLI examples

This example shows how to create OpenStack flavors for VCX10 through VCX90 models.

```
$ openstack flavor create --public vcx10 --id auto --ram 2048 --disk 20 --vcpus 1 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	20
id	f763d2db-cf11-40f9-9b57-002801e79621
name	vcx10
os-flavor-access:is_public	True
properties	
ram	2048
rxtx_factor	1.0
swap	
vcpus	1

```
$ openstack flavor create --public vcx20 --id auto --ram 2048 --disk 20 --vcpus 1 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	20
id	ab0de064-3c42-497f-9e1f-921d210e29cc
name	vcx20
os-flavor-access:is_public	True
properties	
ram	2048
rxtx_factor	1.0
swap	
vcpus	1

```
$ openstack flavor create --public vcx30 --id auto --ram 2048 --disk 20 --vcpus 1 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	20
id	a25f7ff7-3055-49c7-b2b2-6c1f6bf763b6
name	vcx30
os-flavor-access:is_public	True
properties	
ram	2048
rxtx_factor	1.0
swap	
vcpus	1

```
$ openstack flavor create --public vcx40 --id auto --ram 4096 --disk 20 --vcpus 2 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	20
id	e38834a0-b424-4dcc-a9ea-70c7d6167af8
name	vcx40
os-flavor-access:is_public	True
properties	
ram	4096
rxtx_factor	1.0
swap	
vcpus	2

```
$ openstack flavor create --public vcx50 --id auto --ram 8192 --disk 38 --vcpus 4 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	38
id	2674952a-84d0-4942-9e80-08b51b5a8342
name	vcx50
os-flavor-access:is_public	True
properties	
ram	8192
rxtx_factor	1.0
swap	
vcpus	4

```
$ openstack flavor create --public vcx60 --id auto --ram 8192 --disk 38 --vcpus 4 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	38
id	8a73bd53-d52c-4788-845e-80b72019816a
name	vcx60
os-flavor-access:is_public	True
properties	
ram	8192
rxtx_factor	1.0
swap	
vcpus	4

```
$ openstack flavor create --public vcx70 --id auto --ram 24576 --disk 70 --vcpus 6 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	70
id	7e583004-0914-481c-971f-4e0dc59fb9e7
name	vcx70
os-flavor-access:is_public	True
properties	
ram	24576
rxtx_factor	1.0
swap	
vcpus	6

```
$ openstack flavor create --public vcx80 --id auto --ram 32768 --disk 86 --vcpus 12 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	86
id	f58a60d4-42d5-4b6d-bd0e-3f7f4cefcdd
name	vcx80
os-flavor-access:is_public	True
properties	
ram	32768
rxtx_factor	1.0
swap	
vcpus	12

```
$ openstack flavor create --public vcx90 --id auto --ram 49152 --disk 118 --vcpus 24 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	118
id	80d3ee37-c13c-40e3-8ca1-972bd70f0ba6
name	vcx90
os-flavor-access:is_public	True
properties	
ram	49152
rxtx_factor	1.0
swap	
vcpus	24

```
$ openstack flavor create --public vcx90 --id auto --ram 65536 --disk 160 --vcpus 32 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	118
id	80d3ee37-c13c-40e3-8ca1-972bd70f0ba6
name	vcx90
os-flavor-access:is_public	True
properties	
ram	49152
rxtx_factor	1.0
swap	
vcpus	24

```
$ openstack flavor create --public vcx90 --id auto --ram 131072 --disk 300 --vcpus 44 --rxtx-factor 1
```

Field	Value
OS-FLV-DISABLED:disabled	False
OS-FLV-EXT-DATA:ephemeral	0
disk	118
id	80d3ee37-c13c-40e3-8ca1-972bd70f0ba6
name	vcx90
os-flavor-access:is_public	True
properties	
ram	49152
rxtx_factor	1.0
swap	
vcpus	24

SteelCentral Controller for SteelHead Mobile Deployments

The following types of users exist in a branch office and remote access user deployment scenario:

- “Overview of SteelCentral Controller for SteelHead Mobile deployment” on page 511
- “Multiple Mobile Controller deployments” on page 514
- “Ports used with Mobile Controllers and SteelHead Mobiles” on page 524
- “Interaction between Mobile Controllers and SteelHead Mobile clients” on page 524
- “Location awareness” on page 525
- “SSL with SteelCentral Controller for SteelHead Mobile” on page 528
- “Mobile Controller best practices and other considerations” on page 531

This chapter requires that you be familiar with the *SteelCentral Controller for SteelHead Mobile User Guide* and the *SteelCentral Controller for SteelHead Mobile Installation Guide*.

Note: If you are using a release previous to SteelCentral Controller for SteelHead Mobile 4.0, see earlier versions of the *SteelHead Deployment Guide* and the *SteelCentral Controller for SteelHead Mobile User Guide* on the Riverbed Support site at <https://support.riverbed.com>.

Overview of SteelCentral Controller for SteelHead Mobile deployment

Before you begin the installation and configuration process for the Mobile Controller, you must select a network deployment. This section describes the Mobile Controller deployment options. This section includes the following topics:

- “Basic setup for deploying Mobile Controller” on page 511
- “Mobile Controller with VPN deployments” on page 512
- “Mobile Controller with firewall deployments” on page 513
- “Branch office and remote access deployments” on page 514

Basic setup for deploying Mobile Controller

The Mobile Controller ships with default policies. You can install and deploy the Mobile Controller without modifying the default policies, or you can modify them to suit your environment.

If your network environment requires the deployment of multiple Microsoft Installer (MSI) packages, create the packages you need before you deploy the default package.

To install the Mobile Controller using the default *Initial* policy provided, deploy the MSI package named *Default*. The default MSI package installs the default policies.

For the basic steps for how to install and configure the Mobile Controller and how to deploy the default MSI package to the SteelHead Mobile in your network, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Mobile Controller with VPN deployments

When you deploy Mobile Controller components in environments with VPNs, make sure that you do not optimize the VPN tunnel. If the VPN tunnel uses TCP for transport, add a pass-through rule to the policy for the VPN port number connected to by the client. Depending on your deployment scenario, this rule might be the first rule in the list.

VPNs that use IPsec as the transport protocol do not need a pass-through rule.

You can configure the Mobile Controller with a VPN as follows:

- In-path
- Out-of-path

Figure 24-1 shows a deployment in which both the mobile employee and the branch office use the same in-path SteelHead.

Figure 24-1. In-path Mobile Controller deployment and VPN tunnel

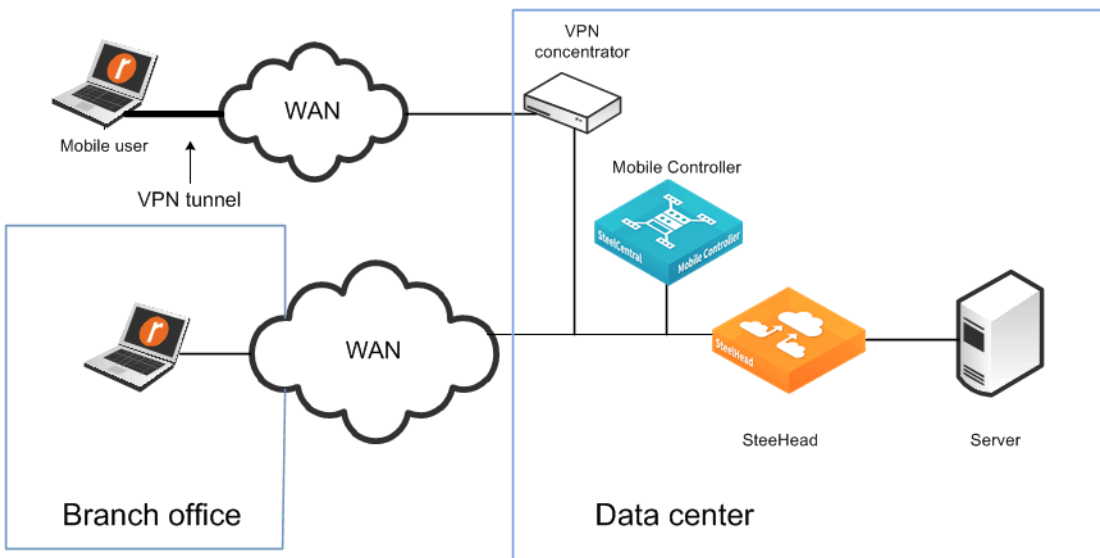
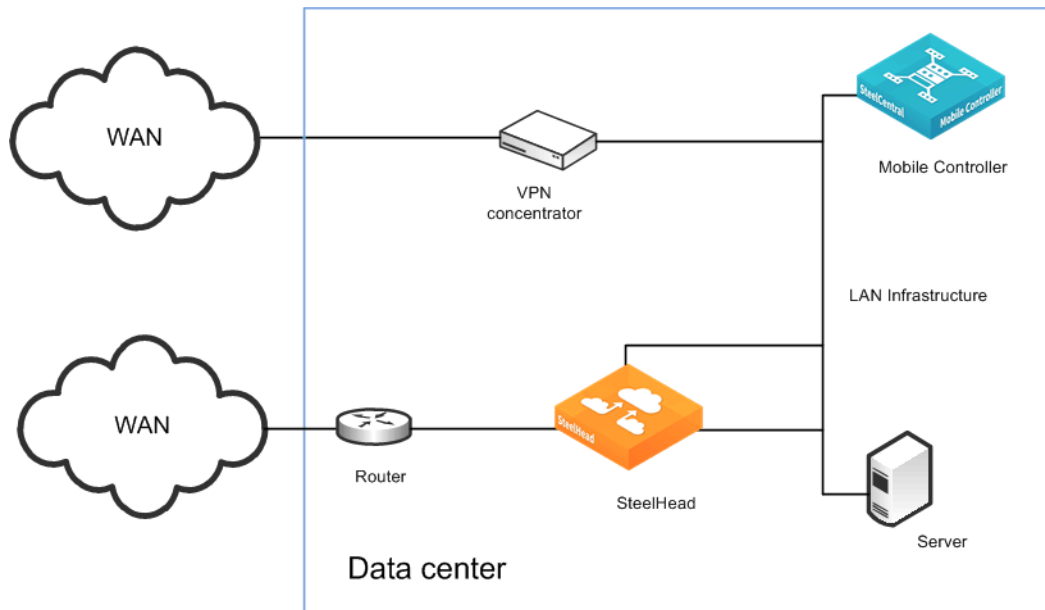


Figure 24-2 shows an in-path deployment in which the mobile employee and the branch office use the same SteelHead, but for the branch office SteelHead is in-path; for the mobile employees, it is out-of-path.

Figure 24-2. Out-of-path deployment Mobile Controller deployment and VPN tunnels



For more information about policies and pass-through rules, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Mobile Controller with firewall deployments

External firewalls, such as home firewall router appliances commonly found with broadband internet connections, do not require special settings for SteelHead Mobile when operating with VPN software on the client computer. The VPN software can have special requirements for external firewalls.

If you are using a firewall that does not allow outgoing connections, you must allow `rbtdebug.exe`, `rbtmon.exe`, `rbtspport.exe`, `rbtlogger.exe`, and `shmobile.exe`.

If you must access the Mobile Controller without the use of a VPN, both the client-side and server-side network firewalls must have some or all of ports 22, 80, 443, 7800, 7810, and 7870 open, as follows:

- Port 22 allows SSH access to the Mobile Controller from a remote site.
- Ports 80 and 443 allow web access (including HTTP and HTTPS).
- Port 7800 is the default port between the SteelHead Mobile and the remote SteelHead for all optimized TCP sessions.
- You need to open Port 7810 on the network firewalls if you configure SteelHead Mobile to optimized connections with server-side out-of-path SteelHeads.
- SteelHead Mobile uses Port 7870 to send statistics to the Mobile Controller.

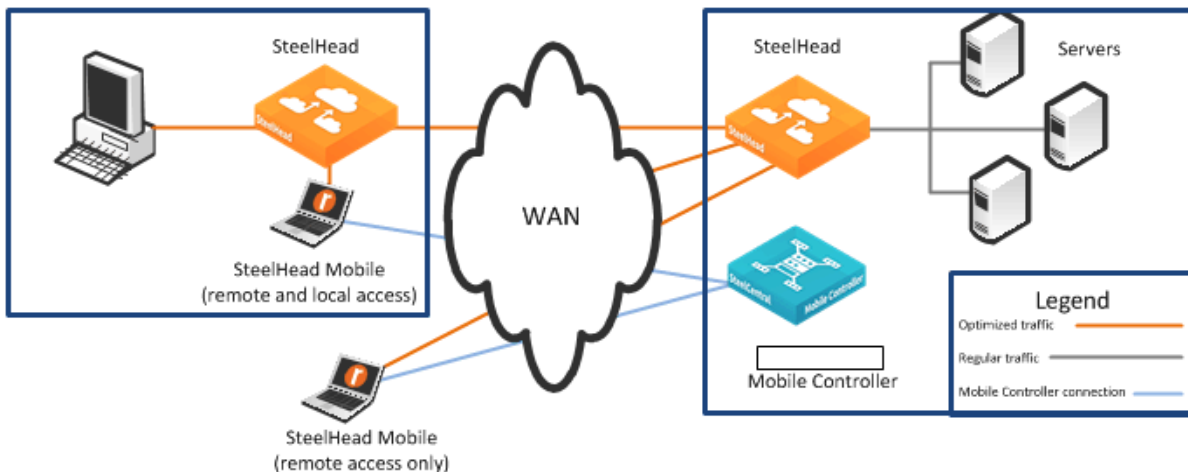
If you are using a VPN originating on the client machine, you do not need to open any of these ports mentioned previously.

Branch office and remote access deployments

In a branch office and remote access user deployment scenario, there are the following types of users:

- Local branch office users with systems that are already optimized by the local SteelHead. These users do not need the SteelHead Mobile software.
- Local branch office users who also remotely access the network. These users need SteelHead Mobile software, and their systems are optimized by the server-side SteelHead.

Figure 24-3. Deploying SteelCentral Controller for SteelHead Mobile for branch office and remote users



If SteelHead Mobiles are connecting to a branch office that already has a SteelHead, you can enable enhanced autodiscovery on all SteelHeads. Autodiscovery allows the SteelHead Mobile to bypass the local SteelHead and optimize with the remote SteelHead at the data center.

If you configure the branch warming feature in SteelCentral Controller for SteelHead Mobile 3.0 or later, the SteelHead Mobile automatically detects the local SteelHead when it is in the branch office (using location awareness). The SteelHead Mobile does not consume a license when it is at the branch office. The SteelHead Mobile continues to optimize with the remote SteelHead, and it also warms the SteelHead Mobile RiOS data store, the local SteelHead, and the remote SteelHead.

For information about location awareness and branch warming, see [“Location awareness” on page 525](#). For information about enhanced autodiscovery, see [“Peering rules” on page 39](#) and the *SteelHead User Guide*.

Multiple Mobile Controller deployments

This section describes the benefits of deploying more than one Mobile Controller, including deployment methods. This section includes the following topics:

- [“Overview of multiple Mobile Controller deployments” on page 515](#)
- [“Mobile Controller Concurrent User Limits” on page 517](#)
- [“Configuring multiple Mobile Controllers for redundancy” on page 517](#)
- [“Preparing to join Mobile Controllers in a high-availability cluster” on page 519](#)
- [“Sizing considerations in a high-availability cluster” on page 520](#)

- [“Endpoint license pooling” on page 521](#)
- [“Communication between HA cluster members” on page 523](#)

Overview of multiple Mobile Controller deployments

If you deploy multiple Mobile Controllers, you gain the following benefits:

- **Federation** - Different IT teams can manage designated areas.
- **Scale** - You can support a greater number of concurrently connected users.

There is a limit of concurrent user licenses per Mobile Controller. This limit is either 100, 4000, or 20,000 depending on the Mobile Controller model. For details, see [“Mobile Controller Concurrent User Limits” on page 517](#).
- **Redundancy** - In case of a network outage or a failure of a Mobile Controller, users can continue to receive a license from another Mobile Controller and gain optimized access to network resources.

By default, when you deploy multiple Mobile Controllers, they operate as separate entities, with their own SteelHead Mobile policies and concurrent user licenses. If you need identical policies across multiple Mobile Controllers, then you must individually update each instance. Concurrent user licenses on a failed Mobile Controller are not available for use by other Mobile Controllers. To ensure policies and licenses can be automatically synchronized and shared, deploy multiple Mobile Controllers in a high-availability cluster.
- **High-availability cluster** - In case of a network outage or failure of a Mobile Controller, users can receive a license from another Mobile Controller and gain optimized access to network resources. If you configure a high-availability cluster, multiple Mobile Controllers automatically synchronize the SteelHead Mobile policies among themselves. Concurrent user licenses on all Mobile Controllers in a cluster are pooled together as a single resource that is available to all Mobile Controllers in the cluster. To configure two or more Mobile Controllers as a high-availability cluster, the controllers must be running SteelCentral Controller for SteelHead Mobile 4.0 or later.

With more than one data center, you are not required to deploy multiple Mobile Controllers. If you have multiple data centers, but only one Mobile Controller, the SteelHead Mobile obtains a user license from the Mobile Controller no matter in which data center the Mobile Controller is located. The Mobile Controller is not directly involved with the optimized connections. After the SteelHead Mobile has a license, it optimizes connections with the SteelHeads in the data centers where the application servers are located.

You must distinguish between a deployment of multiple Mobile Controllers for redundancy and multiple Mobile Controllers as a high-availability cluster. If you are using SteelCentral Controller for SteelHead Mobile 4.0 or later, we recommend that you use a high-availability cluster. The following table shows the differences.

Requirement	Multiple Mobile Controllers	Multiple Mobile Controllers in a high-availability cluster (SteelCentral Controller for SteelHead Mobile 4.0 or later)
Federated SteelCentral Controller for SteelHead Mobile	Yes	Yes
Autonomous SteelHead Mobile policy management per SteelCentral Controller for SteelHead Mobile device	Yes	No
Global SteelHead Mobile policy management	Partial - requires manual policy synchronization between SteelCentral Controller for SteelHead Mobile devices.	Yes
Ability to service license requests on failure of the Mobile Controller	Partial - requires sufficient additional endpoint licenses per SteelCentral Controller for SteelHead Mobile device.	Yes
Scalable beyond 4000 concurrent SteelHead Mobile	Yes	Yes

Mobile Controller Concurrent User Limits

The following tables show the different Mobile Controller models and concurrent user limits.

Mobile Controller appliance	Default number of concurrent users	Maximum concurrent users
SMC-8650-BASE	40	4000
SMC-9000-BASE	40	20,000

Mobile Controller-v licenses for VMware ESX host and SteelHead EX 2.x	Default number of concurrent users	Maximum concurrent users
SMC-VRT-V100	10	4000
SMC-VRT-V100-E	100	100

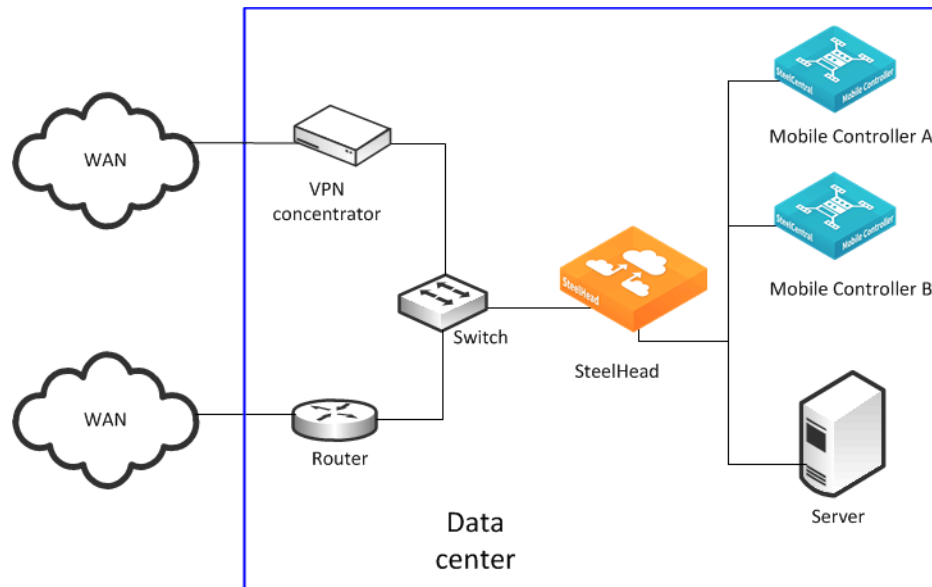
Mobile Controller-v licenses for SteelHead EX 1.x	Default number of concurrent users	Maximum concurrent users
VSMC-VSP	10	100
VSMC-VSP-E	100	100

Configuring multiple Mobile Controllers for redundancy

This section describes how to deploy two or more Mobile Controllers in a redundant configuration. This section requires that you be familiar with [“Multiple Mobile Controller deployments” on page 514](#).

Figure 24-4 shows Mobile Controller A and Mobile Controller B installed in the data center and configured with a basic setup. You can install Mobile Controllers in different data centers and continue to provide redundancy.

Figure 24-4. Two active Mobile Controllers



Complete the following steps on both of the Mobile Controllers.

To configure two Mobile Controllers for redundancy

1. From the Mobile Controller Console, choose Manage > Policies.
2. Select an existing policy to edit, or create a new policy.
3. Select Endpoint Settings.
4. Select Controller Settings.
5. Under Controller Options, add the hostname or IP address of each Mobile Controller.

To avoid any potential issues with invalid certificates when deploying in an SSL environment, make sure that the name and IP address in the Mobile Controller list is consistent with the hostname field in the Mobile Controller configuration.

- Under Controller Options, select Use Random Ordering of Controllers when Connecting to ensure a random but even distribution of clients per Mobile Controller.

Figure 24-5. Endpoint Settings page

Manage > Policies ?

+ Create New Policy — Remove Selected Policies

Policy ↑↓	Last Modified Time ↑↓	Description
Initial	2017/05/23 00:52:44	

General Settings In-Path Rules Protocol Settings SSL Location Awareness **Endpoint Settings**

Controller Settings Desktop Settings

☐ Auto-update List with all Controllers from the Cluster

Controller Options

+ Add New Controller — Remove Selected Controllers ↑↓ Move Selected Controllers

#	Controller Hostname	Port
1	main-vsmc2.lab.nbttech.com (localhost)	7870

☐ Use Random Ordering of Controllers when Connecting

Update Policy

You can repeat these steps to add additional Mobile Controllers to the redundant configuration later. In each case, add the hostname or IP address of the additional Mobile Controllers to the Endpoint Settings in the relevant policy on all the Mobile Controllers in the redundant configuration.

To avoid clients receiving an untrusted certificate message, multiple Mobile Controllers in a redundant configuration must have the same Certificate Authority (CA) certificates. For more information, see [“Multiple Mobile Controllers and SSL” on page 531](#).

Preparing to join Mobile Controllers in a high-availability cluster

This section describes how to prepare two or more Mobile Controllers to form a Mobile Controller high-availability cluster. Mobile Controller clusters work together as a single entity with respect to client policies and some elements of the reporting. You can administer the Mobile Controller cluster from any member of the cluster.

Any client policy that you update on any cluster member automatically replicates and synchronizes with the other members of the cluster. Conflicts in which the same policy is inadvertently edited at the same time on different cluster members resolve automatically. Policies, packages, and some reporting statistics are automatically passed among cluster members. By default, this communication is through a TCP connection on port 7870.

Before a Mobile Controller can join a cluster, the Mobile Controller must:

- have a valid IP address.
- be able to ping other cluster members.
- be running SteelCentral Controller for SteelHead Mobile 4.0 or later.

- have its own set of licenses (CIFS, MAPI, concurrent user endpoint licenses, and so on).
- have the same signing CA certificate as the other Mobile Controllers already in the cluster.

No geographic or latency restrictions exist between the nodes of a Mobile Controller cluster, because the Mobile Controllers in a cluster use the same protocol that a Mobile Controller uses with a SteelHead Mobile. Typically, the current connection between the Mobile Controller and the SteelHead Mobiles supports round-trip times in excess of 3 seconds.

After a Mobile Controller successfully joins the cluster, it is automatically populated with the packages, policies, and other client-related settings from the existing members of the cluster, and any existing settings are deleted. If you want to retain a copy of the original settings, make sure that you back up the Mobile Controller configuration before joining it to an existing cluster. You do not need to configure existing cluster members for a new Mobile Controller to join.

To set up a cluster, make sure that all Mobile Controllers meet the requirements listed above, choose one Mobile Controller to start with, and join the remaining Mobile Controllers to the cluster one-by-one.

The members of the cluster can be different types and models of Mobile Controller. For example, it is possible to have SMC-9000, SMC-8650 and a Mobile Controller-v all in the same cluster so long as they meet the requirements previously listed.

For information about how to join the Mobile Controller to the cluster, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Sizing considerations in a high-availability cluster

An HA cluster provides the ability to survive the failure of an individual Mobile Controller and potentially multiple Mobile Controller failures if you have a configuration that supports this (for example, the number of cluster members is greater than two).

Clustering also enables scalability for the number of supported endpoints. When multiple Mobile Controllers are joined together to form an HA cluster, the endpoint licenses that each Mobile Controller instance had prior to joining are put into a license pool. The license pool is available for all cluster members to use.

The following guidelines ensure a supported cluster configuration:

- We recommend that a cluster of SMC-9000s does not exceed a total of 80,000 endpoint licenses and a cluster of SMC-8650 does not exceed a total of 20,000 endpoint licenses. A cluster of SMC-8650 with 20,000 endpoint licenses can be 5 x SMC-8650 with 4,000 licenses each. Another example is that you can have a cluster of 8 x SMC-8650 with 2,500 licenses each.
- While you can have a mixture of different Mobile Controller platforms as part of a cluster, in a cluster that comprises a mixture of SMC-9000 and SMC-8650 models make sure that surviving cluster members can sustain the total number of endpoint licenses in the pool. For more information about pooling of endpoint licenses, see [“Endpoint license pooling” on page 521](#).

As a result of ongoing development and enhancements, this information is always subject to change. We recommend that you contact your Riverbed account team if you have a need to extend beyond these limits.

Endpoint license pooling

There are many different scenarios related to HA clusters and endpoint licensing. To have a successful deployment, you must understand how these endpoint licenses move in and out of the pool. This section includes the following topics:

- [“General process for pooled license distribution” on page 521](#)
- [“License thresholds” on page 522](#)
- [“Pooled license counts during cluster member failure” on page 523](#)

Every cluster member must have some number of endpoint licenses before it joins the cluster. Generally, a new Mobile Controller has a number of endpoint licenses included by default. The actual quantity of default endpoint licenses depends on the model of the Mobile Controller. Additional endpoint licenses are supplied in packs, with each pack containing 10 licenses. For example, to add 100 licenses to a Mobile Controller, you purchase ten packs. When a Mobile Controller joins a cluster, it gives licenses to the pool if it has more than 100 licenses (this is the default setting).

For example, if a Mobile Controller has 130 licenses when it joins the cluster, it gives 30 licenses to the pool and keeps 100 licenses. If a Mobile Controller has 60 licenses, it does not give any licenses to the pool when it joins the cluster. Giving licenses to the pool is known as *check in*.

You can change the default of how many licenses the Mobile Controller keeps initially with the following command on the Mobile Controller CLI:

```
(config) # cluster license initial-count <number>
```

For example:

```
(config) # cluster license initial-count 50
```

This command changes the default of licenses kept from 100 to 50. In that, when a Mobile Controller joins a cluster for the first time, if it has more than 50 endpoint licenses, it checks in the excess to the pool.

Note: Because the Mobile Controller is part of a cluster, you can enter the command once, and it automatically applies to all other members of the cluster.

General process for pooled license distribution

A Mobile Controller cluster member assigns a license from the cluster pool to a SteelHead Mobile client when the client requests one.

When the license count of a Mobile Controller drops below a certain threshold, the Mobile Controller takes more licenses from the pool. Taking licenses from the pool is known as *check out*. If there are no licenses in the pool, the Mobile Controller continues to assign licenses on request until it either runs out of licenses or the pool is populated with more available licenses.

If there are no more licenses for the Mobile Controller and there are no licenses in the pool, the SteelHead Mobile client automatically contacts another Mobile Controller for a license (assuming the client is configured with multiple Mobile Controllers). If the other Mobile Controllers do not have an available license, the SteelHead Mobile client passes through connections while continuing to request a license at 10-second intervals. After a license is issued, any new connections are optimized.

There are a number of threshold settings related to license check out and check in.

When a Mobile Controller takes licenses from the pool, by default it checks out up to 100 licenses. You can change the default value with the following command on the Mobile Controller CLI:

```
(config) # cluster license checkout-count <number>
```

For example:

```
(config) # cluster license checkout-count 40
```

This command changes the default threshold from 100 to 40. In that, when a Mobile Controller reaches its *low threshold* value, it checks out 40 more licenses. For more information about thresholds, see [“License thresholds” on page 522](#).

Note: Because the Mobile Controller is part of a cluster, you can enter the command once, and it automatically applies to all other members of the cluster.

If the pool contains fewer licenses than the value for *checkout-count*, then the Mobile Controller takes all the remaining licenses, leaving the pool empty.

After a Mobile Controller has joined a cluster, checking in its excess licenses to the pool, it also detects how many licenses it has kept.

License thresholds

You can configure the Mobile Controller with two thresholds associated with the license usage. The *high threshold* is when a Mobile Controller has assigned most of its licenses (in other words, license usage is high) and needs to check out more licenses from the pool.

The *low threshold* is when the number of assigned licenses drops (in other words, license usage is low). When a client machine running the SteelHead Mobile client goes offline for a while (shuts down, hibernates, disconnects from the network, and so on) the Mobile Controller that assigned the license to the client marks the license as available for another request. When the number of assigned licenses in the Mobile Controller drops beneath the low-threshold value, the Mobile Controller has an excess of available licenses and checks in excess licenses to the pool.

The high-threshold and low-threshold values are expressed as a percentage. These two values are by default 90 and 70 respectively. You can change the default value with the following command on the Mobile Controller CLI:

```
(config) # cluster license high-threshold <percentage>
```

and

```
(config) # cluster license low-threshold <percentage>
```

For example:

```
(config) # cluster license high-threshold 80
```

This command changes the default value from 90% to 80%. In that, when a Mobile Controller has assigned 80% of its initial count of licenses, it checks out more licenses from the pool as specified by the *checkout-count* value.

Note: Because the Mobile Controller is part of a cluster, you can enter the command once, and it automatically applies to all other members of the cluster.

Pooled license counts during cluster member failure

One of the benefits of using multiple Mobile Controllers in a cluster rather than having them as individual (autonomous) instances is that you can pool the licenses in such a way that they remain available with one surviving member in the cluster.

When a Mobile Controller fails as part of a cluster, any licenses that it has are automatically placed in the pool and can be used by surviving Mobile Controllers in the cluster.

When the failed Mobile Controller recovers, it automatically checks out licenses as described in [“Endpoint license pooling” on page 521](#).

In a cluster of mixed Mobile Controller models that include one or more SMC-9000-BASE and one or more SMC-8650-BASE, a failure scenario might result in fewer licenses being available due to the capacity of the Mobile Controller.

For example, consider a cluster composed of two Mobile Controllers (1 x SMC-9000-BASE and 1 x SMC-8650-BASE) each with 3,000 licenses. The cluster can support 6,000 endpoints connected concurrently. But if the SMC-9000-BASE were to fail, only 4,000 licenses are available from the SMC-8650-BASE because that is its maximum capacity.

Communication between HA cluster members

Individual Mobile Controllers that are part of an HA cluster share the same SteelHead Mobile group assignments, policies, packages and other general configuration details. This information is sent to a new cluster member at the time of joining the cluster. During the lifetime of the cluster, any changes or additions to deployment configuration data that are completed on one of the cluster members is automatically communicated to the other cluster members. Likewise, the data associated with endpoint reports is also communicated between the cluster members.

As a new cluster member joins, there can be a comparatively large amount (10s of MB) of information sent from an existing member to the new member. There is no real time limit for this transfer to complete, but if the link between the members is low bandwidth and high latency (for example, a WAN link, then the initial transfer can take some time. The new member only transitions to the Connected, Synced status after the transfer has completed. When this status is reached, the member is fully operational within the cluster.

The cluster members use a persistent connection on TCP port 7870 to send the configuration and report data. This port is the same TCP port that is used by SteelHead Mobile to communicate with the Mobile Controller.

After the cluster members are all in sync, any configuration changes to one member are automatically broadcast to the other members. Even if there is no change, there is a poll every 5 minutes between random pairs of cluster members to compare the configuration between the pair. This ensures that any changes that have been missed are properly synced.

Ports used with Mobile Controllers and SteelHead Mobiles

Mobile Controllers and SteelHead Mobile clients use the following TCP ports to communicate with each other:

- **7870** - reporting, statistics, license requests, and policy pushes between SteelHead Mobile and the Mobile Controller

This port is the default port for the communication between Mobile Controllers in a cluster.

- **80/443** - SteelHead Mobile software upgrade
- **80/443** - SteelHead Mobile upload of sysdump and tcpdump files using HTTP POST

Interaction between Mobile Controllers and SteelHead Mobile clients

This section describes some of the communication between SteelHead Mobile clients and the Mobile Controller. For more information about the terminology used in this section such as *packages*, *policies*, *group assignments*, and so on, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

A client machine with the SteelHead Mobile client software installed and enabled begins optimizing traffic after it has completed some initial interaction with a Mobile Controller. This initial contact with the Mobile Controller is triggered by the SteelHead Mobile client when it intercepts the first connection that is to be optimized.

The SteelHead Mobile begins by contacting a Mobile Controller that is already included on the list of controllers defined in the policy of the client. In a small deployment, there is probably only one Mobile Controller on the list. In a larger deployment, there might be more than one Mobile Controller. In such a case, the first Mobile Controller contacted is either the one at the top of the list (depending on the endpoint settings in the policy) or one that is chosen at random.

Contact is made through a TCP connection on port 7870. If there are no problems, the Mobile Controller issues a license. If the Mobile Controller does not have a license available, it instructs the client to contact the next Mobile Controller on its list. The SteelHead Mobile client continues through its list of Mobile Controllers, contacting each one in turn, until it receives a license. If no licenses are available, the client continues to try and contact Mobile Controllers every 20 seconds until it receives a license. After a license has been issued, the SteelHead Mobile client can continue to optimize any new connections that it intercepts that match relevant in-path rules in the client policy.

During this initial interaction after a license has been issued, the Mobile Controller takes the opportunity to check the policy that the client already has, and if a newer one exists, the controller automatically downloads it to the client. If a newer policy is downloaded, it takes immediate effect.

The Mobile Controller also checks the current version of the SteelHead Mobile client software, and if a newer version is available for the client, it might prompt the client to update its software. This step can occur differently depending on the way you have configured the endpoint settings for the client policy: for example, if you have a manual update process.

The connection on TCP port 7870 to the Mobile Controller from the SteelHead Mobile client is kept open to allow the client to send updates for reporting purposes (for example, optimized connections, data reduction, and so on). These updates are typically sent every 5 minutes.

If the end user logs off the client machine but there are still optimized connections in the background, the SteelHead Mobile client remains licensed and the connections continue to be optimized.

The Mobile Controller releases the client license under one of the following circumstances:

- The client is shut down.
- The client is disconnected from the network.
- The client hands back the license because it has entered branch mode.
- The connection to the Mobile Controller is lost for a continuous period of 24 hours.
- The client hands back the license because it has no optimized connections of any kind for a period of 10 minutes.

In this last case, the client instantly requests a license once more if a new connection is intercepted for optimization.

Location awareness enables the SteelHead Mobile client to enter branch mode and may use branch warming. To learn more about these features and how it affects whether licenses are issued, see [“Location awareness” on page 525](#).

Location awareness

This section describes how to configure location awareness on the SteelHead 6.0 and later and the SteelCentral Controller for SteelHead Mobile 2.0 and later. This section includes the following topics:

- [“Overview of location awareness” on page 525](#)
- [“Branch warming” on page 526](#)

Overview of location awareness

Location awareness enables SteelHead Mobiles using SteelCentral Controller for SteelHead Mobile 2.0 or later to detect that they are in a branch office with a SteelHead, and it enables the branch office SteelHead to optimize the traffic of SteelHead Mobile clients.

When a SteelHead Mobile client is at a branch office that has a SteelHead, location awareness enables you to choose which device performs optimization. If the SteelHead Mobile performs optimization and is not in branch mode, the SteelHead Mobile warms its local RiOS data store. It also consumes a SteelHead Mobile license.

If a branch office and data center have a SteelHead, the SteelHead Mobile client can use location awareness to put the client into branch mode. In branch mode you can use the branch office SteelHead for optimization so that the end user does not consume a SteelHead Mobile license. But in that case, the SteelHead Mobile does not warm its local RiOS data store. Branch warming enhances location awareness by warming the SteelHead Mobile RiOS data store. By default, branch warming is disabled.

For more information about branch warming, see [“Branch warming” on page 526](#).

You can configure location awareness in the following ways:

- **Adapter-based location awareness** - SteelHead Mobile optimizes traffic over only the adapters you specify. For example, most VPNs implement a virtual Ethernet adapter, but you can configure a rule to always optimize over VPN adapters and not over LAN adapters.

You cannot use adapter-based location awareness in:

- remote offices with a small number of users and no branch office SteelHead.
 - hardware-based VPNs that do not terminate at the endpoint or client.
- **Latency-detection location awareness** - Enables the SteelHead Mobile clients to optimize data only if latency to the closest SteelHead is greater than the threshold value. The default latency threshold value is 10 ms. To configure the value on the Mobile Controller, choose Policies > Location Awareness.

Branch warming

This section describes how branch warming on the SteelHead Mobile interacts with the SteelHeads in the branch office and the data center. Branch warming requires the Mobile Controller and SteelHead Mobile to run SteelCentral Controller for SteelHead Mobile 3.0 or later, and the SteelHead to run RiOS 6.0 or later. Earlier versions of RiOS provide only location awareness, not branch warming.

This section includes the following topics:

- [“Branch warming and SteelHead Mobile licenses” on page 527](#)
- [“Branch warming and enhanced autodiscovery” on page 528](#)

Branch warming works in conjunction with location awareness, enabling the SteelHead Mobile user to experience warm acceleration regardless of the location. Branch warming tracks the data segments created while a SteelHead Mobile is in a SteelHead-enabled branch office. Location awareness enables SteelHead Mobile to detect that it is in a branch office with a SteelHead, and it enables the SteelHead to optimize SteelHead Mobile traffic.

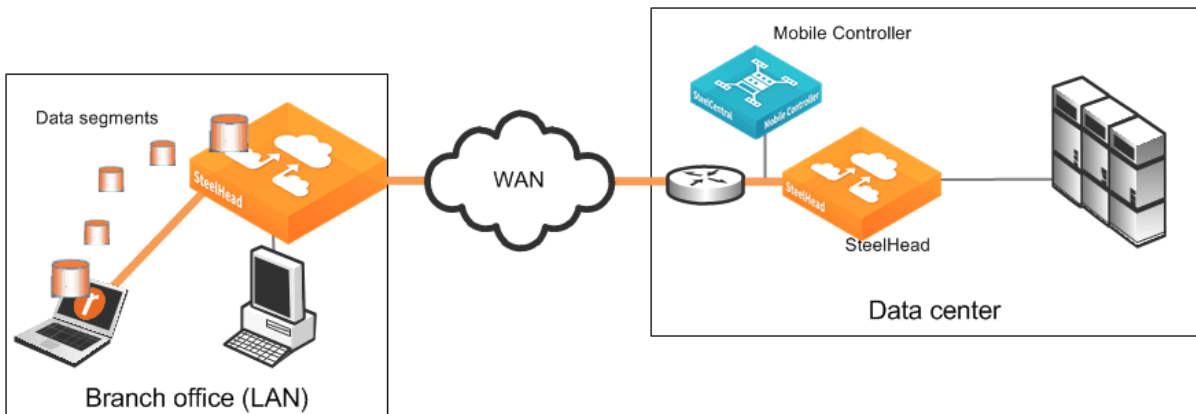
When you enable branch warming, SteelHead Mobile and the SteelHead cooperate to provide warm data for out-of-branch use. SteelHead Mobile shares segments with the SteelHead, thereby providing warm data wherever possible. Branch warming populates new data transfers that are occurring between the client and server, placing them into the RiOS data stores of SteelHead Mobile, the branch office SteelHead, and the server-side SteelHead.

When you download data from the server, the server-side SteelHead checks to see if either SteelHead Mobile or the branch office SteelHead has the data in its RiOS data store. If either device already has the data segments, the server-side SteelHead sends only references to the data. SteelHead Mobile and the branch office SteelHead communicate with each other to resolve the references.

Other clients at the branch office also benefit from branch warming, because data transferred by one client at a branch office populates the branch office SteelHead RiOS data store. Performance improves for all clients at the branch office because they receive warm performance for that data.

Figure 24-6 shows how branch warming enables mobile users to optimize traffic with the server-side SteelHead, while feeding segments that these users generate into the branch office SteelHead RiOS data store.

Figure 24-6. Branch warming example



For each data request, the server-side SteelHead checks whether the branch office SteelHead or the SteelHead Mobile RiOS data store making the request already has the data. If either one has the data, the SteelHead sends a reference to the SteelHead Mobile.

After the SteelHead Mobile gets the reference, it checks if its RiOS data store already has the reference. If it does, the SteelHead Mobile communicates with the server-side SteelHead that it need not send the data again. Simultaneously, the SteelHead Mobile checks whether the branch office SteelHead has the same reference. If the branch office SteelHead has the reference, the communication concludes; otherwise, the SteelHead Mobile shares the reference and data with it.

If the SteelHead Mobile does not have the reference or if its RiOS data store is deleted, it checks with the branch office SteelHead to determine if it has the reference. If it does, then the SteelHead Mobile takes the data segments from the branch-side SteelHead and communicates with the server-side SteelHead that it need not send the data again.

However, if the branch office SteelHead does not have the reference, the SteelHead Mobile requests the new data from the server-side SteelHead and shares the new data and reference with the branch office SteelHead so that at the end this communication, all three—the server-side SteelHead, the branch office SteelHead, and the SteelHead Mobile—have the reference.

Branch warming and SteelHead Mobile licenses

A SteelHead Mobile with branch warming enabled, inside a branch office using the branch SteelHead, uses one connection on the server-side SteelHead and one connection on the branch office SteelHead. It does not use a SteelCentral Controller for SteelHead Mobile license when in the branch mode. A single SteelCentral Controller for SteelHead Mobile license allows an unlimited number of connections.

The SteelHead Mobile uses a license only when it detects that the SteelHead with which it has optimized connections is not in branch mode.

Branch warming and enhanced autodiscovery

Enable enhanced autodiscovery on all SteelHeads to ensure that branch warming is successful. By default, enhanced autodiscovery is enabled.

With enhanced autodiscovery enabled, the last SteelHead along the network path from the client to the server is automatically detected. Optimization then occurs between the SteelHead Mobile and the last SteelHead.

You can display, add, and modify enhanced autodiscovery settings on a SteelHead in the Optimization > Network Services: Peering Rules page.

SSL with SteelCentral Controller for SteelHead Mobile

SSL is a cryptographic protocol that provides secure communications between two parties over the internet. This section includes the following topics:

- [“Traditional SSL optimization” on page 528](#)
- [“Advanced high-security SSL optimization” on page 529](#)
- [“Configuring SteelCentral Controller for SteelHead Mobile and SSL” on page 530](#)
- [“Using SteelHead Mobile with SSL proxy devices” on page 530](#)
- [“Supported TLS versions with SteelHead Mobile” on page 531](#)
- [“Multiple Mobile Controllers and SSL” on page 531](#)

Typically in a web-based application, the client authenticates the server. You install an SSL certificate on a web server for the client to check the credentials of the certificate to make sure it is valid and signed by a trusted third party. Trusted third parties that sign SSL certificates are called CA certificates.

SteelHead Mobile 2.0 and later support both traditional SteelHead SSL optimization and advanced high-security SSL optimization. We recommend that you use advanced high-security SSL optimization to protect your system. You must have RiOS 5.5 or later installed on your SteelHead to use advanced high-security SSL.

For more information about traditional SteelHead SSL optimization, see the *SteelHead Deployment Guide - Protocols*.

Traditional SSL optimization

In traditional SteelHead SSL optimization with a SteelHead Mobile, a client-side SteelHead and the SteelHead Mobile can optimize traffic from any client. The traditional SSL security mode enables clients to optimize SSL traffic to all SteelHeads before RiOS 5.5. RiOS 5.5 and later can run in *mixed* deployments where one SteelHead is running RiOS 5.5 and another SteelHead in the network is running an earlier RiOS version.

In traditional SSL optimization, the client-side SteelHead runs on the client machine. An attacker can redirect network traffic to their SteelHead Mobile-enabled system and obtain the client session key sent from the server-side SteelHead, thereby decrypting the client traffic.

To prevent such man-in-the-middle attacks, and to ensure that SteelHead Mobile can decrypt traffic originating on only one machine, SteelCentral Controller for SteelHead Mobile 2.0 and later provide advanced high-security SSL optimization.

Advanced high-security SSL optimization

This section provides an overview of advanced high-security SSL optimization. We recommend that you use advanced high-security SSL optimization to protect your system.

Advanced SSL optimization:

- enables SteelHead Mobile to optimize traffic from applications on the local system but not from any other system.
- requires RiOS 5.5 or later. If you are running a version earlier than RiOS 5.5, SteelHead Mobile supports the traditional SteelHead SSL optimization.
- has specific browser requirements. For the most current requirements, see the release notes.

Figure 24-7. Advanced high-security SSL optimization using a SteelHead Mobile

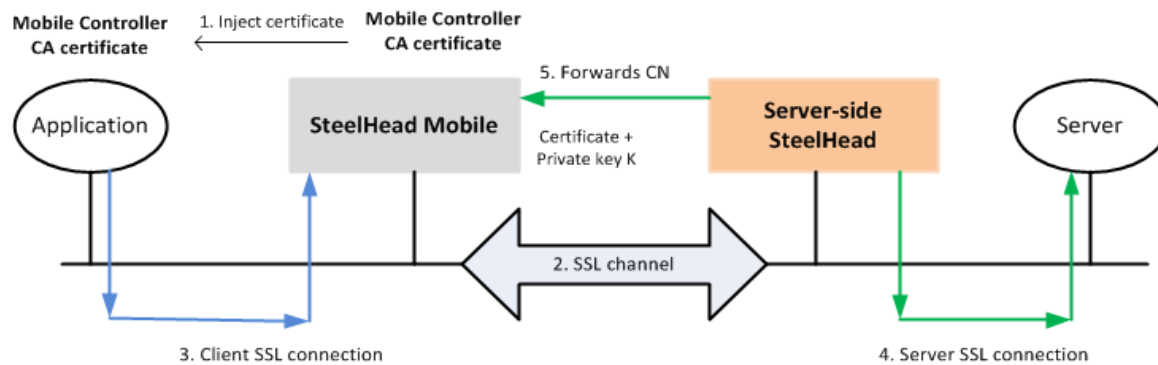


Figure 24-7 shows the steps in the advanced high-security SSL optimization using SteelHead Mobile. The steps are as follows:

1. SteelHead Mobile inserts a CA certificate into the trusted CA list using Internet Explorer or Firefox. The CA certificate is local to SteelHead Mobile.
2. When connections are initiated for SSL optimization, either on demand or proactively, the SSL inner channel is initiated.
3. SteelHead Mobile intercepts the client SSL connection from the client and terminates it.
4. The server-side SteelHead connects to the server and extracts the common name (CN) from the server certificate. CN is the DNS name or IP address.
5. The server-side SteelHead forwards the CN to the SteelHead Mobile.

SteelHead Mobile takes the CN and uses the CA certificate it injected to generate a signed server certificate that it passes to the application. The application can trust this certificate because it is signed by the CA that exists in its trusted certificates list.

Configuring SteelCentral Controller for SteelHead Mobile and SSL

This section provides an overview of the basic steps required to configure SSL using SteelCentral Controller for SteelHead Mobile 4.0 or later and the SteelHead 6.0 or later.

1. Obtain valid Enhanced Cryptography License Keys. This is required for every SteelHead that peers with SteelHead Mobile and Mobile Controller.
2. Configure the server-side SteelHead for SSL optimization with the SSL servers.
3. Configure a trust relationship between the Mobile Controller and the server-side SteelHead.
 - Export the SSL certificate from the Mobile Controller.
 - Configure a proxy certificate and private key for the SSL back-end server on the server-side SteelHead. This step enables the server-side SteelHead to act as a proxy for the back-end server, which is necessary to intercept the SSL connection and to optimize it.
 - Import the Mobile Controller CA certificate into the server-side SteelHead.
 - Export the server-side SteelHead peering certificate.
 - Add the server-side SteelHead peer to the Mobile Controller.

These mutual trust relationships establish secure inner channels between the Mobile Controller and the server-side SteelHead.

For information about the secure inner channel, see the *SteelHead Deployment Guide - Protocols*.

4. Create or edit the policy on the Mobile Controller so that it allows the SteelHead Mobile to intercept SSL connections.
5. Run a test to verify your configuration.

For information about configuring SteelCentral Controller for SteelHead Mobile and SSL, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Using SteelHead Mobile with SSL proxy devices

With SteelHead Mobile 4.8 and later, the client can continue to provide SSL optimization in deployments in which you use SSL proxy devices. SteelHeads have been supporting optimized traffic of such deployments since RiOS 7.0. For more information about SSL deployments, see the *SteelHead Deployment Guide - Protocols*.

The overall configuration steps on the server-side SteelHead are exactly the same as for deployments in which there is a client-side SteelHead. When using SteelHead Mobile with SSL proxy devices, we recommend you use a minimum of RiOS 8.6.0 on the server-side SteelHead. The client-side configuration follows exactly the same steps as for general SSL optimization that are described in [“Configuring SteelCentral Controller for SteelHead Mobile and SSL” on page 530](#), with the additional step of enabling the SSL Proxy Support option in the Mobile client policy.

When you configure this option with the Management Console, use the setting in the SSL section of the policy page. When configuring the policy using the CLI on the Mobile Controller, use the following command:

```
(config) # policy id <Policy ID> ssl proxy-support enable
```

For information about configuring SteelCentral Controller for SteelHead Mobile and SSL, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Supported TLS versions with SteelHead Mobile

By default, after you have completed the steps in [“Configuring SteelCentral Controller for SteelHead Mobile and SSL” on page 530](#), the SteelHead Mobile client supports SSL connections that use TLS 1.0.

SteelHead Mobile 4.7 supports TLS 1.1 or 1.2. You must enable this capability using the following CLI command on the Mobile Controller:

```
(config) # policy id <id-number> ssl backend client-tls-1.2
```

Even though TLS 1.1 is not mentioned in the syntax of this command, using this command automatically enables support for both TLS 1.1 and 1.2.

You must make sure that the server-side SteelHead is configured to support TLS 1.2 by using the following commands on the SteelHead CLI:

```
(config) # protocol ssl backend server-tls-1.2  
(config) # protocol ssl backend client-tls-1.2
```

For information about configuring SteelHeads and SSL, see the *SteelHead Deployment Guide - Protocols*, the *SteelHead User Guide*, and the *Riverbed Command-Line Interface Reference Manual*.

Multiple Mobile Controllers and SSL

An SSL configuration requires there be a *trust relationship* between the SteelHead Mobile and Mobile Controller. Without this trust relationship, the Mobile Controller cannot connect to and provide the SteelHead Mobile with configuration details.

Mobile Controllers in a redundant deployment or a high-availability cluster must have identical signing CA certificates. Having identical signing CA certificates is a prerequisite for a Mobile Controller to join a cluster with other Mobile Controllers. Mobile Controllers generate their own certificates. If the Mobile Controllers do not have the identical signing CA certificates, the SteelHead Mobile clients receive an untrusted certificate message.

For more information about configuring trust relationships for Mobile Controllers and SSL, see the *SteelCentral Controller for SteelHead Mobile User Guide*.

Mobile Controller best practices and other considerations

This section lists best practices and includes other factors to consider when deploying Mobile Controller. This section includes the following topics:

- [“Deployment scenarios” on page 532](#)
- [“Management best practices” on page 533](#)
- [“Migration Mobile Controller hardware” on page 534](#)
- [“Licensing best practices” on page 535](#)
- [“Antivirus software” on page 535](#)

- “Signed SMB support” on page 535
- “SSL client authentication support” on page 536
- “SMC and Federal Information Processing Standard (FIPS)” on page 536
- “Optimization before user log in” on page 537

Deployment scenarios

Consider the following types of deployment scenarios. **Figure 24-8** shows the same SteelHead for office and mobile employees—use fixed-target rules if the VPN ingress point is different.

Figure 24-8. Same SteelHead for mobile employees and office employees

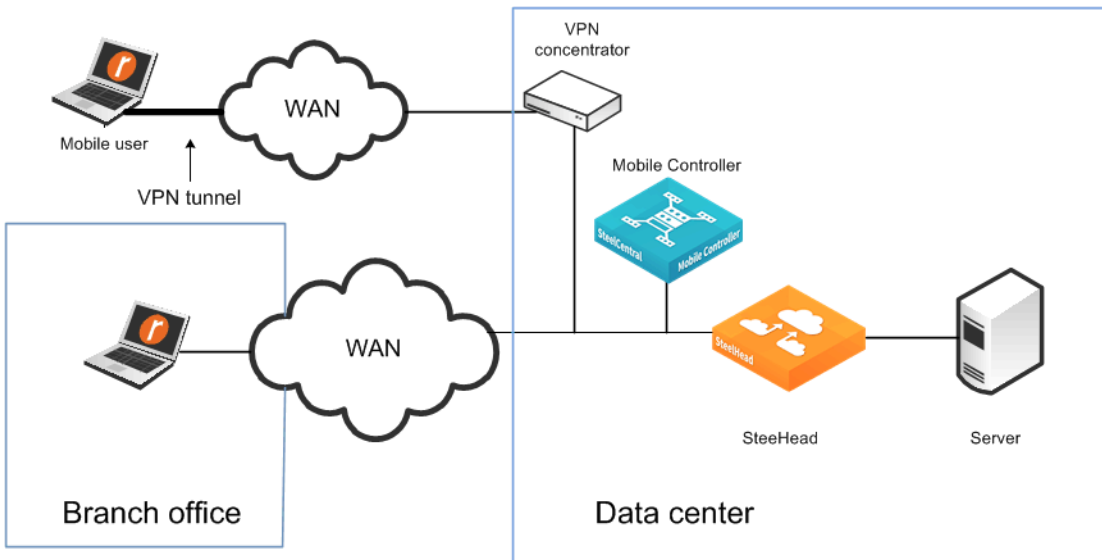
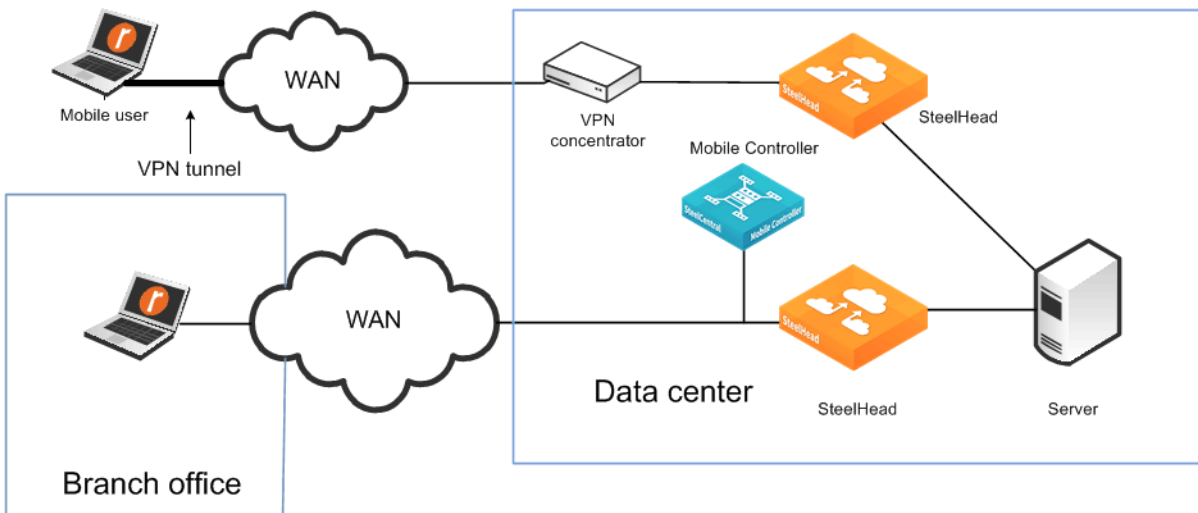


Figure 24-9 shows one SteelHead for mobile employees and one SteelHead for office employees.

Figure 24-9. Using one SteelHead for mobile employees and one SteelHead for office employees



Consider the following factors when deciding between using a SteelHead or SteelCentral Controller for SteelHead Mobile for a branch office:

- SteelCentral Controller for SteelHead Mobile is best for small offices or offices with employees that spend the majority of their time out of the office.
- SteelHead Mobile has an individual RiOS data store, requiring software on each laptop. A SteelHead has a shared RiOS data store.
- SteelHeads ensure that a resource is dedicated to acceleration.
- SteelHeads have features that are not available on SteelCentral Controller for SteelHead Mobile, such as VSP, prepopulation, and QoS.
- You can deploy Mobile Controller appliances in the data center for easier accessibility and connectivity.
- For installations where you expect to have fewer than 100 concurrent mobile users, consider installing Mobile Controller virtual edition using on the SteelHead in the data center.

If you are deploying the Windows operating system and software by cloning, you might run into an issue in which you create duplicate SIDs (ghosting). For more information, go to <http://supportkb.riverbed.com/support/index?page=content&id=S15558>.

Management best practices

The best practices to deploy and manage SteelCentral Controller for SteelHead Mobile are as follows:

- Understand your mobile user population by geography, client type, and division.
- Design systems that do not modify the default endpoint settings in the policy on the Mobile Controller.

If you decide to modify the default endpoint settings for groups of users, consider using fewer groups with more members, versus more groups with fewer members. For information about endpoint settings and policies, see the *SteelCentral Controller for SteelHead Mobile User Guide*.
- Use MSI packages to push SteelCentral Controller for SteelHead Mobile software to clients. MSI packages can also enable rollback or upgrade, ensuring easy maintenance. Mobile Controller does support automatic upgrades to clients. For more information, go to <https://supportkb.riverbed.com/support/index?page=content&id=S15231>.
- The initial installation of the client software cannot be pushed out from the Mobile Controller, but you can use Microsoft Windows Group Policy Object (GPO) to automatically push out and install client software for the first time to Windows clients. For more information, see *Managing Mobile Clients Using Group Policy Templates* at <https://supportkb.riverbed.com/support/index?page=content&id=S14393>.
- We recommend that you back up your Mobile Controller-v using SCC. For more information about backing up and restoring Mobile Controller-v, especially when running on ESXi in VSP on SteelHead EX go to <https://supportkb.riverbed.com/support/index?page=content&id=S17487>.
- Installation can be done in *visible* or *invisible* mode. In visible mode, end users have a Riverbed icon in the system tray that they can click for basic information and settings. In invisible mode, there are no visible icons to show that SteelCentral Controller for SteelHead Mobile is running on the end-user machine.

- Do not use branch warming if you:
 - always want the client to optimize.
 - are using RiOS earlier than 6.0.

Migration Mobile Controller hardware

Due to hardware refresh or replacement, you might need to migrate the configuration data (assignments, policies, packages, and so on) from your legacy Mobile Controller to a new hardware platform.

To perform a successful migration, use the following steps as guidance. Depending on the versions of software running on the old versus new Mobile Controller, you might not need to perform all the steps.

To migrate Mobile Controller hardware

1. Back up older Mobile Controller configurations to the SCC.
2. Install the new Mobile Controller running the same software version (or close to it) on the network, using the new IP address.
3. Manage the new Mobile Controller in the SCC.
4. Unplug the old Mobile Controller from the network.

At this time, the SteelHead Mobile clients that already have an active license continue to optimize, but they cannot upload any reports and no policies are pushed out.
5. Restore the old Mobile Controller configuration (including IP address) to the new Mobile Controller. SteelHead Mobile client users connect automatically.
6. Upgrade the new Mobile Controller to the most current required version of software.
7. If needed, push out the software update to the Mobile Clients using the preferred method.

Note: You cannot migrate or transfer the endpoint license packs from old Mobile Controller platform to new Mobile Controller platform. Contact your Riverbed account team to obtain new license packs.

You can migrate to new hardware by temporarily creating a cluster of the two Mobile Controllers, allowing the policies and packages from the old Mobile Controller to move to the new Mobile Controller. After the transfer is complete, you can update the policies with the new Mobile Controller address to ensure that SteelHead Mobile clients connect to it. You can next remove the old Mobile Controller from the cluster. The main caveat to this method is that both Mobile Controllers must be running the same version of Mobile Controller software. You cannot use this method if the old Mobile Controller does not support the newer software version.

For information on another approach to migrate from one Mobile Controller hardware to a new Mobile Controller hardware, go to <https://supportkb.riverbed.com/support/index?page=content&id=S24136>.

Licensing best practices

The best practices for end-user licensing are as follows:

- When considering licenses, do not count mobile employees who are behind the SteelHead and are using branch warming.
- On the basis that not all mobile users are connected and active at the same time, estimate a 3:1 ratio of licensed versus connected users.
- SteelHead Mobile is issued a license by the Mobile Controller only after SteelHead Mobile initiates its first optimized connection request.
- For configurations in which you deploy multiple Mobile Controllers for high availability, use version 4.0 or later to allow pooling of end-user licenses from all the Mobile Controllers. Pooling of end-user licenses is an efficient use of the licenses if there is an unplanned Mobile Controller outage.

Antivirus software

You can configure certain antivirus tools installed on a Windows or Mac platform to scan files that have recently changed. Configure the antivirus scanner to ignore the SteelHead Mobile RiOS data store.

Because SteelHead Mobile is constantly updating its RiOS data store when the user is sending and receiving data with optimized connections, the SteelHead Mobile RiOS data store is scanned frequently. This might lead to end-user performance problems. Because the SteelHead Mobile RiOS data store does not contain files of any type (just unique data segments), there is no need to scan it for viruses.

Signed SMB support

You do not have to configure the SteelHead Mobile policy to support signed SMB connections. For traffic to optimize correctly:

- SteelHead Mobile must run 3.1 or later.
- you must configure the server-side SteelHead to optimize signed SMB traffic (joined to a Windows domain, configured with a suitable authentication mode, and so on).

If configured correctly, the Current Connections report on the server-side SteelHead shows CIFS-SIGNED.

With Mobile Controller 4.6 and later, Signed SMB connections are supported for SMB1, SMB2, and SMB3 dialects. Previous versions of Mobile Controller supported only SMB1 and SMB2.

While the SteelHead Mobile software is supported with Windows 8.1 operating systems, it is possible that the Windows client connects to a Windows 2012 or 2012-R2 file server SMB 3.02 or later. SMB 3.02 is only fully supported in SteelHead Mobile 4.7 and later, and SMB 3.1.1 is only fully supported in SteelHead Mobile 5.0 and later. If you are using an earlier version of the client, you can provide data reduction, but no Layer-7 latency optimization is possible. To ensure the correct behavior for SteelHead Mobile clients with version prior to 4.7, you must modify the policy for SteelHead Mobile to ensure the correct dialect is used.

Enter the following commands on the Mobile Controller substituting <id> for the relevant policy ID used by SteelHead Mobile:

```
no policy id <id> smb2 neg-whitelist enable
no policy id <id> smb2 basic-dialect
write memory
```

Note: This setting does not affect the optimization of the other SMB dialects that continue to receive full latency and data optimization benefits. You do not need this setting if the SteelHead Mobile client is 4.7 or later.

When using Windows 10 operating systems with the SteelHead Mobile client, you must use SteelHead Mobile 4.8 or later to ensure support for signed SMB or any other optimization benefits.

For more information about signed SMB support, see the *SteelHead Deployment Guide - Protocols*.

SSL client authentication support

Mobile Controller 4.6 and later include support for SSL client authentication. This feature enables SteelHead Mobile to optimize traffic when the client PC is using a Common Access Card (CAC). Mobile Controller 4.6 and later support only client PCs running Windows 7 and using TLS 1.0.

To enable support for client authentication, you need to configure both the Mobile Controller and the server-side SteelHead.

Follow these steps on the Mobile Controller:

1. Enable SSL optimization.
2. Enable Client Certificate support.
3. Import the server-side SteelHead peering certificate.

Perform Steps 1 and 2 within the Mobile Controller policy assigned to the relevant client(s). Perform Step 3 on the Mobile Controller Management Console on the Configure > SSL > Peering page.

For information about configuring the server-side SteelHead and general information on CAC and client certificate support, see the *SteelHead Deployment Guide - Protocols*.

SMC and Federal Information Processing Standard (FIPS)

With Mobile Controller 4.6 and later, there is an option to deploy the Mobile Controller in a FIPS-compliant mode. The FIPS mode is only applicable to the Mobile Controller itself and does not apply to SteelHead Mobile that it might be responsible for.

Before enabling FIPS mode, the Mobile Controller must have a FIPS license installed.

Mobile Controllers that are part of an HA cluster can be FIPS enabled. We strongly recommend that you have all members of the cluster FIPS enabled rather than have a mixture of FIPS and non-FIPS Mobile Controllers.

You can enable FIPS mode only through the CLI of the Mobile Controller. For more information about the configuration of FIPS, see the *FIPS Administrator's Guide*.

Optimization before user log in

As part of a SteelHead Mobile installation, several processes run in the background, including *rbtsport*. Windows or MacOS starts *rbtsport* when the host operating system starts up. Therefore, even before a user has logged in, the optimization service is already running. This means that the SteelHead Mobile can optimize a user login process. It can also provide optimization for other system processes that are occurring automatically in the background, such as backups.

