Steelhead® Appliance Deployment Guide - Protocols

April 2014

riverbed

© 2014 Riverbed Technology. All rights reserved.

Riverbed®, Cloud Steelhead®, Granite™, Interceptor®, RiOS®, Steelhead®, Think Fast®, Virtual Steelhead®, Whitewater®, Mazu®, Cascade®, Shark®, AirPcap®, BlockStream™, SkipWare®, TurboCap®, WinPcap®, Wireshark®, TrafficScript®, FlyScript™, WWOS™, and Stingray™ are trademarks or registered trademarks of Riverbed Technology, Inc. in the United States and other countries. Riverbed and any Riverbed product or service name or logo used herein are trademarks of Riverbed Technology. 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 Technology 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, Incorporated 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 2012 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.

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 Technology. This documentation may not be copied, modified or distributed without the express authorization of Riverbed Technology 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 Technology assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation.



Riverbed Technology
199 Fremont Street
San Francisco, CA 94105

Phone: 415.247.8800
Fax: 415.247.8801
Web: http://www.riverbed.com

Part Number
712-00127-04

Contents

Preface **1**

About This Guide 1
Audience 2
Types of Steelhead Appliances 2
Document Conventions 3
Additional Resources 3
Release Notes 3
Riverbed Documentation and Support Knowledge Base 4
Online Documentation 4
Contacting Riverbed 4
Internet 4
Technical Support 4
Riverbed Professional Services 4
Documentation 4
What Is New 5

Chapter 1 - CIFS Optimization **7**

Overview of CIFS Protocol 8
CIFS File Locking and Oplocks 9
Safe Reading and Writing 10
Security Signatures 11
Overview of SMB Versions 2, 2.1, 3, and 3.02 11
More Information 12
RiOS CIFS Optimization Techniques 12

Chapter 2 - Microsoft Exchange Email Optimization **15**

MAPI Client and Server Communication 16
Auto-Discovery and MAPI Connections 17
RiOS MAPI Optimization 18
Encrypted MAPI Optimization 19
MAPI Admission Control for Microsoft Outlook 21
MAPI Optimization with Steelhead Appliances in a Serial Cluster or a Parallel Deployment 22

MAPI Optimization with Exchange Clusters 22
Outlook Anywhere Optimization 23
Microsoft Exchange 2013 Optimization 29
MAPI Destination Port Handling 31

Chapter 3 - Signed SMB and Encrypted MAPI Optimization 33

Windows Security Concepts 34
Domain Relationships 35
Choosing an Authentication Mode for the Server-Side Steelhead Appliance 37
Transparent Versus Delegation Mode 38
Overview of Configuring SMB Signing and Encrypted MAPI 40
SMB3 Optimization with Windows 8 Clients and Windows 2012 Server 41
Joining a Steelhead Appliance to a Domain 42
One-Way Trust Configuration 43
Enabling Kerberos in a Restricted Trust Environment 44
Configuring Constrained Delegation for Delegation Mode 45
Kerberos 45
Overview of Kerberos 46
Optimization in a Native Kerberos Environment 48
Domain User with Replication Privileges 49
Configuring Traffic Optimization for HTTP (SharePoint), Encrypted MAPI, and Signed SMB/SMB2/SMB3 49
Configuring the Server-Side Steelhead Appliance for Active Directory Integrated (Windows 2003/2008) 50
Best Practices for the Steelhead Appliance in a Secure Windows Deployment 51
Domain Authentication Scaling 52
When to Use Domain Authentication Scaling 52
General Improvements in RiOS v8.6 52
Domain Controller Load Balancing 53
Domain Health Check and Domain Authentication Automatic Configuration 53
Domain Health Check 54
Domain Authentication Automatic Configuration 58
Example Configurations 60
Single Domain 60
Multiple Domains with Windows 7 Clients 61

Chapter 4 - HTTP Optimization 63

HTTP and Browser Behavior 64
Multiple TCP Connections and Pipelining 65
HTTP Authentication 66
Connection Jumping 68
HTTP Proxy Servers 69
Configuring HTTP SSL Proxy Interception 71

RiOS HTTP Optimization Techniques 73
Primary Content Optimization Methods 73
HTTP Vary Headers 75
Connection Pooling 76
HTTP Authentication Optimization 76
HTTP Automatic Configuration 79
HTTP Settings for Common Applications 82
HTTP Optimization for SharePoint 83
HTTP Optimization Module and Internet-Bound Traffic 84
Tuning Microsoft IIS Server 85
Determining the Current Authentication Scheme on IIS 85
Per-Connection or Per-Request NTLM Authentication 86
Per-Connection or Per-Request Kerberos Authentication 86
Changing the Authentication Scheme 87
Changing the Per-Connection/Per-Request NTLM Authentication Mode 87
Changing the Per-Connection/Per-Request Kerberos Authentication Mode 88
HTTP Authentication Settings 88
HTTP Optimization Module and Proxy Servers 89
Determining the Effectiveness of the HTTP Optimization Module 89
Info-Level Logging 90
Use Case 90

Chapter 5 - Citrix ICA Optimization 95

Overview of Citrix ICA 95
Citrix Version Support 96
Citrix ICA Traffic Optimization with Steelhead Appliances 96
Secure Citrix ICA Encryption 97
Citrix Drive-Mapping Optimizations 98
Citrix Multi-Stream ICA Traffic Optimization with Steelhead Appliances 99
Citrix Virtual Channels and Traffic Priorities 100
Single-Stream and Multi-Stream ICA 100
QoS Classification for Citrix Traffic 104
Reduction for Citrix Small Packet Real-Time Traffic 106
Citrix ICA Optimization Over SSL 106

Chapter 6 - SMTP-Over-TLS Optimization 109

Configuring Microsoft Exchange Servers for SMTP Over TLS 110
Configuring the Steelhead Appliance for TLS Over SSL 112

Chapter 7 - FTP Optimization 115

Overview of FTP 115

Active Mode 115
Passive Mode 116
Configuring In-Path Rules 117
Optimizing FTP 117
Passing through FTP 117
QoS Classification for the FTP Data Channel 118
Active FTP Classification 118
Passive FTP Classification 118
FTP Optimization Considerations 119
Steelhead Mobile FTP Considerations 120

Chapter 8 - Other Protocol Optimization 121

Oracle Forms Optimization 121
Determining the Deployment Mode 122
NFS Optimization 122
Implementing NFS Optimization 123
Configuring IP Aliasing 124
Lotus Notes Optimization 124
Optimizing Encrypted Lotus Notes 124

Chapter 9 - CIFS and HTTP Prepopulation 131

CIFS Prepopulation 131
Design Considerations 135
HTTP Prepopulation 137
Microsoft Silverlight, Apple HLS, and Adobe Flash 138

Chapter 10 - SSL Deployments 141

The Riverbed SSL Solution 141
Overview of SSL 143
How Steelhead Appliances Terminate SSL 144
Configuring SSL on Steelhead Appliances 146
SSL Required Components 146
Setting Up a Simple SSL Deployment 148
Server SSL Optimization Proxy Certificate and Private Key Scenarios 152
Steelhead Appliance Secure Peering Scenarios 154
Deploying Secure Steelhead Appliance Peering 162
Advanced SSL Features 163
Client Certificate Support 164
Proxy Server Support 166
Mid-Session SSL Support 166
Server Name Indication 166
Steelhead Mobile SSL High-Security Mode 167

Troubleshooting and Verification 168
Interacting with SSL-Enabled Web Servers....169
Obtaining the Server Certificate and Private Key....169
Generating Self-Signed Certificates 171

Chapter 11 - Configuring SCEP and Managing CRLs 173

Using SCEP to Configure On-Demand and Automatic Re-Enrollment 173
Configuring On-Demand Enrollment 175
Configuring Automatic Re-Enrollment....175
Viewing SCEP Settings and Alarms....176
Managing Certificate Revocation Lists....177
Managing CRLs 179
Viewing CRL Alarm Status 180

Appendix A - RiOS Version Compatibility with Domains and Domain Relationships 183

User Domain Is the Same as Server Domain—Delegation Mode....184
User Domain Is the Same as Server Domain—Transparent Mode....184
User Domain Is the Different from Server Domain (Bi-Directional)—Delegation Mode 185
User Domain Is Different from Server Domain (Bi-Directional)—Transparent Mode....185
Server-Side Steelhead Appliance is in a Different Domain to the Server with One-Way Trust 186

Index 187

Preface

Welcome to the *Steelhead Appliance Deployment Guide - Protocols*. Read this preface for an overview of the information provided in this guide and contact information. This preface includes the following sections:

- "About This Guide" on page 1
- "Additional Resources" on page 3
- "Contacting Riverbed" on page 4
- "What Is New" on page 5

About This Guide

The *Steelhead Appliance Deployment Guide - Protocols* describes why and how to configure Riverbed appliances with common protocols.

This guide includes information relevant to the following products:

- Riverbed Optimization System (RiOS)
- Riverbed Steelhead appliance (Steelhead appliance)
- Riverbed Steelhead CX appliance (Steelhead CX)
- Riverbed Steelhead EX appliance (Steelhead EX)
- Riverbed Steelhead DX appliance (Steelhead DX)
- Riverbed Virtual Steelhead (VSH)
- Riverbed Cloud Steelhead (CSH)
- Riverbed Central Management Console (CMC)
- Riverbed Central Management Console Virtual Edition (CMC-VE)
- Riverbed Steelhead Mobile software (Steelhead Mobile)
- Riverbed Steelhead Mobile Controller appliance (Mobile Controller)
- Riverbed Steelhead Mobile Client (Mobile Client)
- Riverbed Interceptor appliance (Interceptor appliance)
- Riverbed Virtual Services Platform (VSP)

- Riverbed Services Platform (RSP)
- Riverbed Steelhead Cloud Accelerator software (SCA)

Audience

This guide is written for storage and network administrators familiar with administering and managing WANs. You must be familiar with TCP, CIFS, Citrix, HTTP, Lotus Notes, MAPI, NFS, FTP, and SSL.

You must also be familiar with the following:

- the Management Console. For details, see the *Steelhead Appliance Management Console User's Guide*.
- connecting to the RiOS CLI. For details, see the *Riverbed Command-Line Interface Reference Manual*.
- the installation and configuration process for the Steelhead appliance. For details, see the *Steelhead Appliance Installation and Configuration Guide* and the *Virtual Steelhead Appliance Installation Guide*.
- the Interceptor appliance. For details, see *Interceptor Appliance User's Guide*.
- the CMC. For details, see *Riverbed Central Management Console User's Guide*.
- the Steelhead Mobile Controller. For details, see *Steelhead Mobile Controller User's Guide*.

Types of Steelhead Appliances

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

- **Steelhead appliance (*xx*50)** - includes WAN optimization and Proxy File service (PFS). RSP is available with an additional license. For details, see the *Steelhead Appliance Management Console User's Guide* for Steelhead appliance (*xx*50).
- **Steelhead CX (*xx*55)** - is a WAN optimization-only device. Desktop models have two in-path interfaces. For details, see the *Steelhead Appliance Management Console User's Guide* for Steelhead CX (xx55).
- **Steelhead EX (*xx*60)** - 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 Appliance Management Console User's Guide fo*r Steelhead EX (xx60).
- **Steelhead DX** - addresses the demands of disaster recovery and data protection over WANs for data center-to-data center, with up to 60-times acceleration of data replication workloads. For details, see the *Steelhead Appliance Management Console User's Guide* for Steelhead DX.
- **VSH** - is a virtualized version of the Steelhead appliance that runs under VMware ESX/ESXi and the Cisco Services-Ready Engine (SRE) platform. For details, see the *Virtual Steelhead Appliance Installation Guide*.
- **CSH** - is the Steelhead appliance for public cloud computing environments. You deploy the CSH differently from the Steelhead appliance and the VSH. For details, see the *Riverbed Cloud Services User's Guide*.
- **Mobile Client** - 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 *Steelhead Mobile Controller User's Guide*.
- **Mobile Controller** - provides management functionality for Mobile Clients. For details, see the *Steelhead Mobile Controller User's Guide*.

- **CMC** - provides management functionality for various Riverbed products, including Steelhead appliances, Mobile Controllers, and Interceptor appliances. For details, see the *Riverbed Central Management Console User's Guide*.

For more details on the Steelhead appliance family, see http://www.riverbed.com/us/products/steelhead_appliance/.

Document Conventions

This guide uses the following standard set of typographical conventions.

Convention	Meaning
italics	Within text, new terms, emphasized words, and REST API URIs appear in *italic* typeface.
boldface	Within text, CLI commands, CLI parameters, and REST API properties appear in **bold** typeface.

Courier	Code examples appears in Courier font:
	`amnesiac > enable`
	`amnesiac # configure terminal`

< >	Values that you specify appear in angle brackets: **interface <ipaddress>**
[ ]	Optional keywords or variables appear in brackets: **ntp peer <addr> [version <number>]**
{ }	Required keywords or variables appear in braces: **{delete <filename>}**
|	The pipe symbol represents a choice to select one keyword or variable to the left or right of the symbol. The keyword or variable can be either optional or required: **{delete <filename> | upload <filename>}**

Additional Resources

This section describes resources that supplement the information in this guide. It includes the following:

- "Release Notes" on page 3
- "Riverbed Documentation and Support Knowledge Base" on page 4
- "Online Documentation" on page 4

Release Notes

The following online file supplements the information in this guide. It is available on the Riverbed Support site at https://support.riverbed.com.

Release Notes	Purpose
<product>_<version_number> <build_number>.pdf	Describes the product release and identifies fixed problems, known problems, and work-arounds. This file also provides documentation information not covered in the manuals or that has been modified since publication.

Examine this file before you begin the installation and configuration process. It includes important information about this release of the Steelhead appliance.

Riverbed Documentation and Support Knowledge Base

For a complete list and the most current version of Riverbed documentation, log in to the Riverbed Support site at https://support.riverbed.com.

The Riverbed Knowledge Base is a database of 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.

Online Documentation

The Riverbed documentation set is periodically updated with new information. To access the most current version of Riverbed documentation and other technical information, consult the Riverbed Support site at https://support.riverbed.com.

Contacting Riverbed

This section describes how to contact departments within Riverbed.

Internet

You can learn about Riverbed products at http://www.riverbed.com.

Technical Support

If you have 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.

Riverbed Professional Services

Riverbed has a staff of professionals who can help you with installation, provisioning, network redesign, project management, custom designs, consolidation project design, and custom coded solutions. To contact Riverbed Professional Services, email proserve@riverbed.com or go to http://www.riverbed.com/services-training/Services-Training.html.

Documentation

Riverbed continually strive to improve the quality and usability of the documentation. Riverbed appreciates any suggestions you might have about the online documentation or printed materials. Send documentation comments to techpubs@riverbed.com.

What Is New

Since the December 2013 release of the *Steelhead Appliance Deployment Guide - Protocols*, the following information has been added or updated:

- **Updated** - "Overview of SMB Versions 2, 2.1, 3, and 3.02" on page 11
- "Microsoft Exchange Email Optimization" on page 15 (formerly the MAPI Optimization chapter)
  - **New** - "Support for Common Access Card" on page 26
  - **New** - "Microsoft Exchange 2013 Optimization" on page 29
- "Signed SMB and Encrypted MAPI Optimization" on page 33
  - **Updated** - "SMB3 Optimization with Windows 8 Clients and Windows 2012 Server" on page 41
  - **New** - "Domain Authentication Scaling" on page 52
- **Updated** - "Citrix ICA Optimization" on page 95
- "HTTP Optimization" on page 63
  - **Updated** - "Configuring HTTP SSL Proxy Interception" on page 71
  - **Updated** - "Primary Content Optimization Methods" on page 73
  - **New** - "HTTP Vary Headers" on page 75
  - **Updated** - "HTTP Optimization for SharePoint" on page 83
- **Updated** - "HTTP Prepopulation" on page 137
- **Updated** - "Mid-Session SSL Support" on page 166

CHAPTER 1 CIFS Optimization

This chapter describes the Common Internet File System protocol (CIFS) optimization module on the Steelhead appliance. This chapter includes the following sections:

- "Overview of CIFS Protocol" on page 8
- "RiOS CIFS Optimization Techniques" on page 12

CIFS, also referred to as the Server Message Block (SMB) protocol, has been in existence since the early 1990s. The protocol provides shared access to files and printers, along with other communication between hosts on a network, including an authenticated Inter-Process Communication (IPC) mechanism. It is one of the most common protocols used for network file access by Microsoft operating systems.

The protocol originated as SMB, and was a created by combined work from IBM and Microsoft. SMB was initially designed to run on top of NetBIOS/NETBEUI but has run directly on top of TCP since the Microsoft Windows 2000 version. SMB was primarily a LAN-based protocol, which accounts for some of the challenges you see when it is used across WAN links. The rename and relaunch from SMB to CIFS was around 1996, which was the same time as Sun Microsystems announced WebNFS.

Although CIFS is known as the generic name for the protocol, technical discussions and documents continue to use the term SMB. Since the initial release of SMB, Microsoft has continued to change and enhance the protocol, each time assigning a new version number to SMB. The most recent versions are SMB2, SMB2.1, SMB3, and SMB3.02.

For details on SMB versions, see "Overview of SMB Versions 2, 2.1, 3, and 3.02" on page 11.

Note: For the purpose of this chapter, the terms SMB and CIFS are used interchangeably.

RiOS has provided CIFS optimization since version 1.0, with ongoing enhancements in each subsequent version. CIFs optimization focuses on reducing the impact of WAN round-trip latency of common application and system behavior, including the Microsoft Office product suite, general file access, and remote printing. Ongoing work has been necessary to adjust to new incremental changes introduced to the protocol from Microsoft, typically with each new Microsoft operating system version. Additionally, new features have been added that enhance and broaden the applicability of the CIFS optimization.

RiOS includes optimization for so-called *SMB Signed* traffic. SMB Signed refers to an optional feature in which the client and server, using the CIFS protocol, use cryptographic techniques to sign each protocol datagram exchanged in a CIFS session. This technique does not encrypt data passing through the network; it merely authenticates that the client and server receive datagrams that have not been modified by unauthorized middle devices. Through specific configuration and integration into the Windows security domain, RiOS can perform full latency and bandwidth optimization securely for SMB Signed traffic, while still maintaining the end-to-end authentication that SMB signing was designed to achieve.

While SMB signing does not encrypt the data, SMB3 can encrypt traffic. If you use the correct RiOS release with the appropriate configuration settings, RiOS can continue to provide full-latency and bandwidth optimization while maintaining a secure client-server communication.

For details on configuring CIFS optimization, see the *Steelhead Appliance Management Console User's Guide*.

Overview of CIFS Protocol

This section describes the base CIFS protocol and how RiOS performs CIFS optimization. For example, for a client to access a file on a CIFS server, the following steps generally occur on the client:

1. Parse the full filename to determine the server name.
2. Establish a TCP connection to the server.
3. Negotiate what type of SMB dialect and option is used with the server.
4. Transfer credentials and authenticate access to the server.
5. Open the file for operations by sending the filename and desired access to the server.
6. Perform read, write, or other operations on the file, typically in a serial fashion.
7. Close the file for operation.

After a connection to the server is established, client systems tend to leave the connection open for some amount of time, even when there are no open files by the client. This has an impact on CIFS optimization, because RiOS can optimize CIFS connections only when it detects the beginning of the CIFS connection. This is due to the negotiation of SMB options and SMB dialect that allow RiOS to understand what client and server operating systems are in use, which revision of the CIFS protocol is in use, and other options relevant to optimizing the connection.

The following table shows CIFS protocol commands that traverse the network during a session set up and subsequent simple file read.

Client Command	Server Response
SMB_COM_NEGOTIATE	Must be the first message sent by a client to the server. Includes a list of SMB dialects supported by the client. Server response indicates which SMB dialect should be used.
SMB_COM_SESSION_SETUP_ANDX	Transmits the user authentication information. Successful server response has the user ID (UID) field set in the SMB header and is used for subsequent SMBs on behalf of this user.

Client Command	Server Response
SMB_COM_TREE_CONNECT_ANDX	Transmits the name of the disk share the client wants to access. Successful server response has a Tree Connect ID (TID) field set in the SMB header that is used for subsequent SMBs referring to this resource.
SMB_COM_OPEN_ANDX	Transmits the name of the file, that relative to the TID, the client wants to open. Successful server response includes a file ID (FID) that the client supplies for subsequent operations on this file.
SMB_COM_READ_ANDX	Client supplies TID, FID, file offset, and number of bytes to read. Successful server response includes the requested file data.
SMB_COM_CLOSE	Client closes the file represented by TID and FID. Server responds with success code.
SMB_COM_TREE_DISCONNECT	Client disconnects from resource represented by TID.

Source: http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf (Sep. 4, 2010)

This sequence represents the example protocol flow for a specific use case. Notice that SMB_COM_SESSION_SETUP_ANDX contains the user's authentication information. This is also particularly relevant when there are signed SMB sessions and Windows domains as part of the client/server environment. For details, see "Windows Security Concepts" on page 34.

READ_ANDX and other commands can be used (in some cases) several hundreds of times before the *end of file* is reached. There are many additional round trips across a WAN link that contribute to the overall performance degradation compared to the same operation across a LAN.

CIFS File Locking and Oplocks

The CIFS protocol allows for different types of file locking and concurrent access to be used by client and servers. Separately, the CIFS protocol allows servers to notify clients (when requested) when they are the only client accessing a file, or when all clients for a file are performing only reads.

Requesting an *oplock* is when a client opens a potentially shared file, and requests to be notified by the server when it is the only client accessing the file, or when it is accessing the file with other clients that are performing only reads (and no writes). When a client is granted an oplock on a file, it can perform some types of actions on the file completely locally, without notifying the server. Oplock breaks occur when servers notify a client if additional clients request access to the file and remote operations performed at the client must be synchronized with the server before granting that access.

The following types of oplocks are available:

- **Batch** - Generally used by command-line applications that repeatedly open many of the same files. The term *batch* is used for this type of access, which is most often experienced by batch (or script) command sets. Batch applocks allow clients to perform file opens completely, without notifying the server.
- **Exclusive** - This oplock is granted to a client if the file is not already opened by another client. This means that the client has sole access to the file, so it is safe to cache changes until the file closes or when requested by the server.
- **Level 2** - This oplock is never requested by a client directly. Instead, a client that holds an exclusive oplock might be downgraded to a level 2 oplock at any time by the server, indicating that some additional client has accessed the file but has not yet performed any write operations.

Important: Do not confuse oplocks with file locks. File locks control access to files, and oplocks notify clients about the state of concurrency for a file.

Clients can request a specific type of shared-access control for a file when they open it. Examples of the types of shared access are do not share, share for read only, share for write only, or share for read/write. Clients can also apply more granular control by requesting locks on specific portions of a file, known as *byte-range locks*. The CIFS server tracks all of the access requests that it detects, and grants or denies operations based on the requests from live clients. The shared access control, or *Shared mode* access, is the most prevalent in most user applications. Typically, only database and similar applications can make use of byte range locks.

RiOS registers these concurrency controls—file locks, share access controls, and oplocks. The RiOS CIFS module tracks the state of every file opened between the client and server, and uses this state to determine what types of optimized connections can safely be applied to file requests on the clients.

For example, when a client is granted an exclusive oplock on a file, RiOS can safely perform read-ahead and write-behind of file data because it is the only client accessing the file. Some of the CIFS features in RiOS enable more in-depth recognition of concurrency control, like the overlapping open optimization and the applock optimization feature.

Safe Reading and Writing

The concurrency controls—file locks, shared-access controls, and oplocks—ensure that clients can coordinate their reads and writes in an orderly fashion. In particular, when a client wants to read from a file, it knows that reading from a local cache is safe (for example, through oplocks), or that its reads and writes are coordinated with other clients (for example, through byte range locks).

When a client wants to write data to a file, there are two modes of writing that are supported in CIFS environments: *write-through* and *write-back*. The application using CIFS can decide which of these two modes to use every time it needs to write.

For write-through mode, the process falls into a pattern in which the client sends data to the server for writing, waits for an acknowledgment from the server, sends some more data, waits, and so on, until all the data is written and acknowledged. At this point the client sends a CIFS command to close the file on the server. The acknowledgement from the server that the file is closed signals that all data is safely written to disk. The server releases any lock on the file and makes it available for read-write access to other clients. Although this might be the safest way when writing data, when sending data across the WAN, it is not the most efficient method because the round-trip time across the WAN is typically several magnitudes higher than across the LAN.

Write-back mode is a form of optimization in which the client does not need to wait until the write is acknowledged before proceeding with the next operation. Synchronization is forced by the client flushing its cache or closing the file. Microsoft Word and Excel are good examples of applications that use CIFS in a write-back mode. If for any reason there is a problem when the file is closed, the user is informed and the data is recovered using the automatic save mechanism.

In write-back mode, there are several places where data can linger on its way to the disk: the client machine can gather writes, the server file system can delay pushing to disk, and even the server's physical disk subsystem can cache writes in RAID cards or on memory in the disk itself. RiOS provides another place for this temporary buffering between client and long-term storage to occur during write-back mode.

RiOS honors the client-specified write-back or write-through mode for each write operation. In general, most clients use write-back mode, because the performance of write-through mode—even on local disks, without the added performance impact of WAN based usage—is unacceptable to end users. RiOS can deliver fast write performance over the WAN in a way that is analogous to how file servers and network attached storage (NAS) appliances deliver fast write performance.

Security Signatures

The CIFS protocol includes a feature to cryptographically sign every CIFS protocol datagram between the client and server. The file data that traverses the network is still sent in the clear, even when this option is used, so this CIFS feature does not protect against network snooping. However, it does protect against man-in-the-middle attacks.

The security signature feature is typically controlled through Windows group policy settings, with a separate client-side and server-side three-way setting that controls whether signing is used or not. This policy setting is then used during the initial CIFS session by the client and server to determine whether or not to sign their CIFS sessions. The following table shows the result when a client attempts to connect to a server—either the connection fails due to incompatible settings, or the session can sign the CIFS datagrams.

	Server - Disabled	Server - Enabled	Server - Required
Client - Disabled	Not signed	Not signed	No connections
Client - Enabled	Not signed	Signed	Signed
Client - Required	No connections	Signed	Signed

Several options are available for RiOS to optimize connections that are signed:

- **Do nothing** - RiOS can apply bandwidth and TCP-level optimization but does not perform CIFS latency optimized connections. This is because the client-side Steelhead appliance needs to generate CIFS datagrams to the client as if it were the server, and the server-side Steelhead appliance needs to generate CIFS datagrams to the server as if it were the client.
- **Enable the Optimize Connections with Security Signatures (that do not require signing) feature** - This is only useful when neither the client nor the server use the required setting. When you enable this feature, RiOS alters the negotiation of signing between the client and server so that they each detect the disabled setting.
- **Join the server-side Steelhead appliance to the Windows domain, and possibly configure user delegation if necessary** - Although this requires the most configuration, it maintains the end-to-end security guarantees that security signatures allow. Sessions continue to be signed between the clients and client-side Steelhead appliance, between the Steelhead appliances, and between the server-side Steelhead appliance and server. Maintaining this end-to-end status can be a requirement for some security mandates. For details, see "Windows Security Concepts" on page 34.

Overview of SMB Versions 2, 2.1, 3, and 3.02

Since the initial release of SMB, Microsoft has continued to change and enhance the SMB protocol. These changes have usually coincided with the release of a new version of the Windows client and server operating system:

- SMB2 was introduced with Windows Vista and Windows Server 2008
- SMB2.1 was introduced in Windows 7 and Windows Server 2008 R2
- SMB3 was introduced with Windows 8 and Windows Server 2012
- SMB3.02 was introduced with Windows 8.1 and Windows Server 2012 R2

Using the appropriate version of RiOS, you can configure your environment so that full-latency and bandwidth optimizations are possible regardless of the SMB version.

Not all SMB features are relevant when considering optimizing SMB traffic with RiOS; however consider the following features:

- SMB2
  - Request compounding, which allows multiple SMB 2 requests to be sent as a single network request
  - Larger reads and writes
  - Caching of folder and file properties
  - Improved message signing (HMAC SHA-256 replaces MD5 as hashing algorithm)
- SMB2.1
  - Client oplock leasing model
  - Large MTU support
  - Support for previous versions of SMB (allowing Windows 7 clients and Server 2008 R2 servers to negotiate for backwards compatibility)
- SMB3
  - Secure negotiation (allows the client and server to detect a man-in-the-middle that modifies the negotiation process)
  - Directory leases (allows the client to safely cache directory contents)
  - Encrypted traffic (encryption of the SMB3 traffic on a per-server or per-share basis)
- SMB3.02
  - Some additional capabilities (for example, Instancing, and read and write flags). These capabilities are not relevant for WAN optimization on their own, but because they are in the client/server session setup dialog, the Steelhead appliance needs to understand the dialect to respond correctly.

For more information on SMB3, see "SMB3 Optimization with Windows 8 Clients and Windows 2012 Server" on page 41.

More Information

Although the low-level detail of the CIFS protocol is beyond the scope of this document, there are many public reference materials available. For a starting point, go to http://media.server276.com/codefx/CIFS_Explained.pdf.

RiOS CIFS Optimization Techniques

The CIFS optimization module comprises many separate techniques. Many of these techniques are enabled by default, so when configuring a new Steelhead appliance (or creating a new Steelhead Mobile package), little, if any, CIFS specific configuration is required.

Note: Typically, new features in RiOS, even ones that are widely applicable, are not enabled by default until they have been present for several releases. Some optimization features that are not enabled by default can provide benefit in your environment, and you can consider enabling them on an individual basis.

Modern CIFS servers generally listen on TCP port 445; for compatibility, they also listen to port 139. The CIFS optimization module within RiOS is designed to intercept traffic on these two ports for acceleration of file-sharing and remote Windows printing traffic.

After a CIFS connection is intercepted for optimization, the RiOS CIFS optimization module monitors the negotiation and session setup commands to detect which client and servers are in use, and what types of CIFS options are understood by each. As clients initiate requests to the server, RiOS applies its optimization techniques to reduce the time that the client awaits a response from the server.

For specific details on CIFS optimization features, see *Steelhead Appliance Management Console User's Guide*.

CHAPTER 2 Microsoft Exchange Email Optimization

This chapter describes the optimization of the Messaging Application Programming Interface (MAPI) used between Microsoft Outlook clients and Exchange servers. The combination of Outlook and Exchange server is one of the most common client and server combinations for email in a corporate business environment. This chapter includes the following sections:

- "MAPI Client and Server Communication" on page 16
- "RiOS MAPI Optimization" on page 18
- "MAPI Destination Port Handling" on page 31

Note: The MAPI communication mechanism is also known by a more recent name: Microsoft as Outlook-Exchange Transport Protocol.

MAPI comes in two forms: extended and simple. This section discusses the extended form, which is used by Outlook. The simple form is included as part of Microsoft Outlook Express and is not discussed here.

In the last ten years, MAPI has evolved through Exchange v5.5, Exchange 2000, Exchange 2003, Exchange 2007, and Exchange 2010. Outlook versions have followed in close coordination. RiOS can perform optimization for each of these Exchange versions.

For information about Exchange 2013, see "Microsoft Exchange 2013 Optimization" on page 29.

Because of the requirements of Exchange 2003, the Exchange server communicates using an encrypted conversation with compatible versions of the Outlook client. By default, the encryption mode is enabled in Exchange 2007 and Exchange 2010. RiOS securely optimizes this encrypted MAPI traffic by using the RiOS MAPI optimization along with the Windows-domain related RiOS features.

Microsoft Outlook Cache mode was introduced with Outlook 2003. Cache mode attempts to hide performance issues over the WAN by pre-emptively downloading new email and their attachments from the Exchange server to a cache of storage on the local hard disk of the Outlook client workstation. Although this gives an appearance of fast performance, it has no effect on emails going to the sent items or trash folder of the user. Using Cache mode also means that any mail, including unwanted or junk mail (and attachments), is downloaded to the Outlook client. RiOS provides optimization for Outlook clients using Cache mode, both by reducing the increased WAN bandwidth impact of Cache mode and decreasing the wait time for arriving and transmitted emails.

For more details on configuring MAPI optimization, see the *Steelhead Appliance Management Console User's Guide*.

MAPI Client and Server Communication

The Exchange server includes an end point mapper (EPM) that listens on TCP port 135. The Outlook client connects to this port, and is assigned random TCP server ports to communicate with the Exchange server using the MAPI protocol. These MAPI connections are used to send and receive emails, calendaring, address lookup, and so on. You can configure the Exchange server to use a fixed TCP port for the MAPI connections.

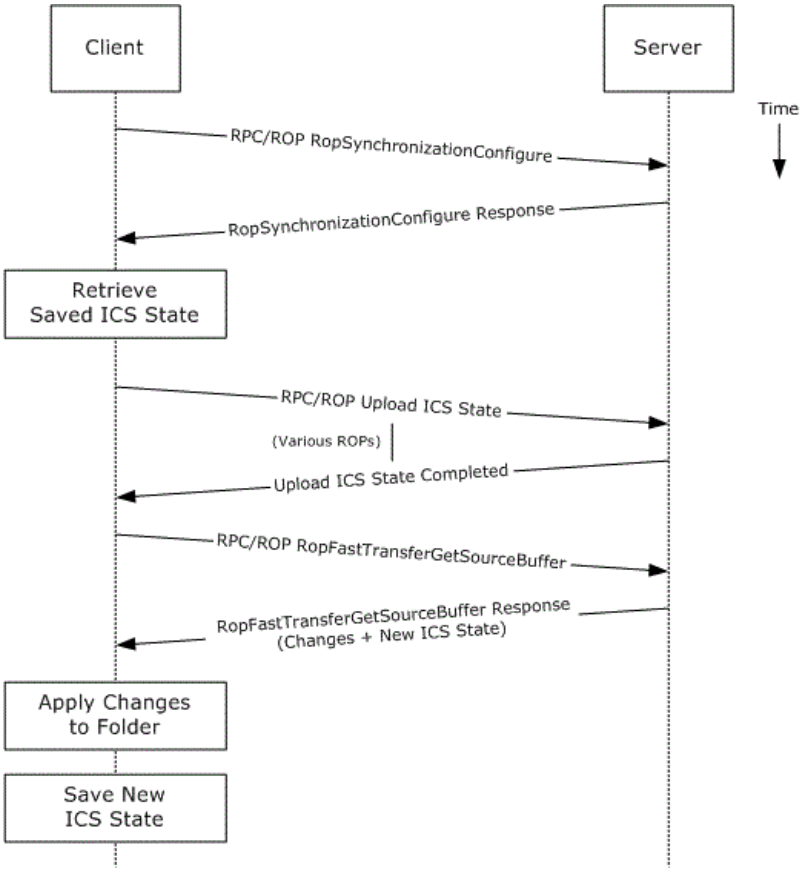
The conversation between the Outlook and the Exchange server is quite complex. The following is a basic overview for the Exchange server communication:

1. The Outlook client contacts the directory service and DNS to establish the universal resource identifier (URI) for the Exchange server.
2. The Outlook client uses the URI to connect to the EPM server, then logs on to the Exchange server.
3. The Outlook client performs remote operations (ROPs) to open (that is, create and write) a new email message, saves the changes, and creates and adds an attachment, if needed.
4. Using Name Service Provider Interface (NSPI), the Outlook client resolves the name of a recipient.
5. The Outlook client submits the message to the Exchange server for sending.

Note: For details on Exchange server communication, go to http://msdn.microsoft.com/en-us/library/cc307725%28EXCHG.80%29.aspx.

Other conversations between Outlook and the Exchange server include opening folders for reading or searching, deleting items, and synchronizing folders. Synchronizing folders is required to download and read emails from the Exchange server to the Outlook inbox. Figure 2-1 shows the synchronization process. The incremental change synchronization (ICS) state is how the client knows what new items to download from the Exchange server.

Figure 2-1. Outlook Client and Exchange Server Synchronization



Source: http://download.microsoft.com/download/5/D/D/5DD33FDF-91F5-496D-9884-0A0B0EE698BB/[MS-OXPROTO].pdf (Sep. 2, 2010)

Steelhead appliance MAPI optimization is based on the streamlining of ROPs.

Auto-Discovery and MAPI Connections

RiOS intercepts and optimizes the EPM conversation between the Outlook client and Exchange server on TCP port 135. For Outlook client connections that are configured for correct addressing, the client-side Steelhead appliance alters the port that the EPM service sends to the Outlook client for MAPI connections (default is port 7830). For port or full transparency addressing modes, no changes are made to the reported MAPI port, and the Outlook client connects to whichever port EPM assigns to MAPI. RiOS intercepts and optimizes subsequent connections made to the MAPI port and performs MAPI-specific optimization on them.

If the Outlook client connections to the Exchange server are intercepted and optimized, MAPI-specific optimized connections are not used when the client-side Steelhead appliance has an in-path rule to pass-through traffic on TCP port 135 (or the server-side Steelhead appliance has a peering rule to pass-through traffic on TCP port 135).

After RiOS learns the MAPI port for an Exchange server, RiOS no longer goes through the auto-discovery process for connections to the Exchange server IP address and learned MAPI port. Although this gives some performance benefits to new MAPI connections (especially in large latency WANs), it also means that configuring pass-through or other types of in-path rules might not have the immediate desired impact on traffic.

Note: Always use the **in-path probe-mapi-data** CLI command to perform auto-discovery for MAPI connections, and to allow in-path rules to have immediate impact, even on Outlook clients that are currently optimized.

When using fixed-target rules to optimize MAPI traffic, configure the client-side Steelhead appliance to optimize traffic where the destination IP address is the Exchange server, and where the destination port is 7830 (or whatever port is configured on the client-side Steelhead appliance for MAPI optimization). If you want optimization for NSPI connections, a fixed-target rule in which the destination IP address is the Exchange server and where the destination port is 7840 (or whatever port is configured on the client-side Steelhead appliance for NSPI optimization).

RiOS MAPI Optimization

The MAPI optimization module comprises several optimization techniques:

- Read Ahead on attachments
- Read Ahead on emails
- Write Behind on attachments
- Write Behind on emails
- Folder Synchronization
- Prepopulation

Note: : Each separate MAPI connection is used by one user only, but it is common for one Outlook to have several MAPI connections active simultaneously—and it usually does. For example, in a branch office where a Steelhead appliance optimizes MAPI traffic for several users, there are several optimized MAPI connections per user, but no single MAPI connection is shared between users. However, when a user closes Outlook and triggers the start of a MAPI prepopulation session, there is only one optimized MAPI prepopulation connection per user.

The following are MAPI optimization methods on the Steelhead appliance:

- **MAPI optimization** - This is the fundamental component of the MAPI optimization module and includes optimization for Read, Write (Receive, Send), and Synchronize operations. This setting uses port 7830 and is enabled by default. If the Exchange server is not configured to use random ports (allocated by EPM) and needs to use a fixed port configured by the administrator, you must change port 7830 to the fixed port number used by the Exchange server.
- **MAPI with Exchange 2000** - Increases the latency optimization benefit for Outlook to Exchange 2000 traffic. Optimized traffic for Exchange depends on having previous optimization levels enabled. For example, to support Exchange 2007, you must enable optimized traffic for Exchange 2000 and 2003 in addition to Exchange 2007. Exchange 2003 and 2007 optimization are enabled by default in RiOS v6.5 and later. MAPI with Exchange 2000 is enabled by default.

- **MAPI Exchange 2003 optimization** - Enables optimization for Exchange 2003.
- **MAPI Exchange 2007 optimization** - Enables optimization for Exchange 2007 and greater.
- **Encrypted optimization** - Enables the client-side Steelhead appliance functionality required for encrypted MAPI optimization. For details, see "Encrypted MAPI Optimization" on page 19.
- **MAPI prepopulation** - Allows a Steelhead appliance to warm its RiOS data store with the data patterns of new mail and attachments that arrive after a user shuts down Outlook. This is especially useful in global organizations where communication to a user continues long after local business hours. MAPI prepopulation considerably reduces the bandwidth consumed by a branch office when users start their day, and can greatly accelerate the reception of mail when a user first connects to an Exchange server.

  MAPI prepopulation does not use any additional Client Access Licenses (CALs) from the Exchange server. To start a MAPI prepopulation session, the Steelhead appliance automatically holds open an existing authenticated MAPI connection after Outlook has shut down on the client. At the same time, the client-side Steelhead appliance sends a message to the server-side Steelhead appliance to prepare a MAPI prepopulation session. The session begins if the following conditions are met:

  – The server-side Steelhead appliance has MAPI prepopulation enabled

  – The existing total client-side Steelhead appliance active connections for MAPI prepopulation has not reached its maximum configured limit

  – A MAPI prepopulation session has not already been started for the client

  No user credentials are used or saved by the Steelhead appliance when performing prepopulation.

  The remote Steelhead appliance uses these pre-authenticated connections to pull the data for mail from the Exchange server over the WAN link, automatically pre-populating the client-side RiOS data store.

  If a user starts a new Outlook session, the MAPI prepopulation session terminates. If for some reason the MAPI prepopulation session does not terminate (for example, the user starts a new session in a location that is different than the Steelhead appliance that has the MAPI prepopulation session active), the MAPI prepopulation session will eventually time-out per the configuration setting.

  In RiOS v6.5 and later, the control of MAPI prepopulation is on the client-side Steelhead appliance. This allows for a higher rate of prepopulated sessions, and enables MAPI prepopulation to take advantage of the read-ahead mechanisms in the MAPI optimization feature.

  MAPI prepopulation v2 is supported in RiOS v6.0.4 and later, v6.1.2 and later, and v6.5 and later. The client-side and server-side Steelhead appliances can be at any of these code train levels and provide prepopulation v2 capabilities. For example, a client-side Steelhead appliance running RiOS v6.0.4 connecting to a server-side Steelhead appliance running RiOS v6.5 or later provides prepopulation v2 capabilities. In contrast, a 6.0.1a client-side Steelhead appliance connecting to a RiOS v6.5 or later server-side Steelhead appliance supports prepopulation v1, but does not provide prepopulation v2.

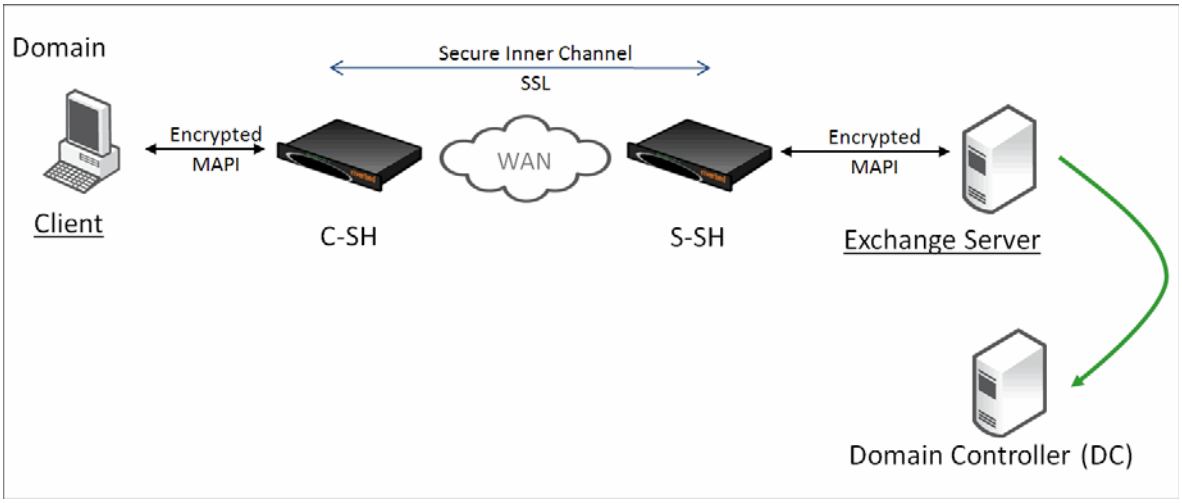
  MAPI prepopulation v2 tracks the User ID as well as the client IP address. This is helpful in scenarios where the user returns to the branch office and is assigned a different IP address by DHCP. In MAPI prepopulation v1 (where just the IP address was tracked), the prepopulation session could have remained until the connection timed out (default 96 hours).

Encrypted MAPI Optimization

When optimizing encrypted MAPI traffic, normal encryption methods are maintained between the Outlook client and client-side Steelhead appliance, and the Exchange server and server-side Steelhead appliance.

To ensure the optimized MAPI connection between the two Steelhead appliances is also encrypted, configure RiOS Secure Inner Channel. For detail, see the *Steelhead Appliance Deployment Guide*.

Figure 2-2. Encrypted Connections Between Client and Server



To enable the Steelhead appliance to optimize encrypted MAPI traffic between Outlook and the Exchange Server:

1. On the server-side Steelhead appliance, choose Configure > Networking > Windows Domain.
2. Join the server-side Steelhead appliance to the same Windows domain that the Exchange server belongs to and operates as a member server.

   Note: An adjacent domain can be used (via Cross-Domain support) if the Steelhead appliance is running RiOS v6.1 or later.

3. Verify that Outlook is encrypting traffic.
4. Enable the **Encrypted Optimization** option on client-side and server-side Steelhead appliances involved in optimizing MAPI encrypted traffic. Alternatively, use the CLI command **protocol mapi encrypted enable**.
5. If the Outlook client is running on a Windows 7 machine, use Delegation mode. If you are not using Windows 7 hosts, use the default Transparent mode.
6. Ensure that both Enable MAPI Exchange 2003 Acceleration and Enable MAPI Exchange 2007 Acceleration are enabled. In RiOS v6.1 and later, by default, both options are enabled.
7. Restart the service on all Steelhead appliances that have the Encrypted Optimization option enabled.

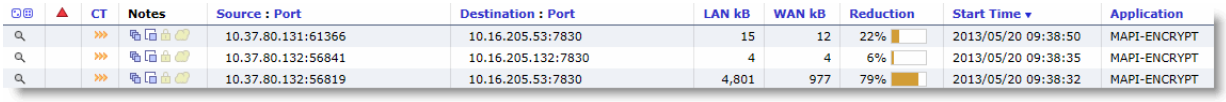
Note: The server-side and client-side Steelhead appliances must be running RiOS v5.5.x or later.

The Windows Security section also provides information about the relevant, minimum version of RiOS to ensure support for different Windows client operating systems and Windows domain structures.

For more details on how to ensure the Steelhead appliance is joined to the domain, as well as configuring Transparent or Delegation modes, see "Domain Relationships" on page 35.

You can confirm that the Steelhead appliance is successfully performing encrypted MAPI optimization by going to the Steelhead Management Console Current Connections report and looking for MAPI-ENCRYPT connections listed in the Applications column of the report page.

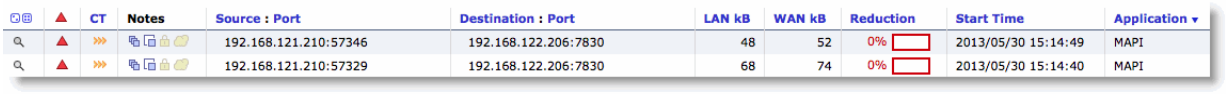
Figure 2-3. Current Connections Report



CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
		10.37.80.131:61366	10.16.205.53:7830	15	12	22%	2013/05/20 09:38:50	MAPI-ENCRYPT
		10.37.80.132:56841	10.16.205.132:7830	4	4	6%	2013/05/20 09:38:35	MAPI-ENCRYPT
		10.37.80.132:56819	10.16.205.53:7830	4,801	977	79%	2013/05/20 09:38:32	MAPI-ENCRYPT

If for some reason this is not successful, you see a red triangle in the Notes column

Figure 2-4. Unsuccessful Current Connections Report

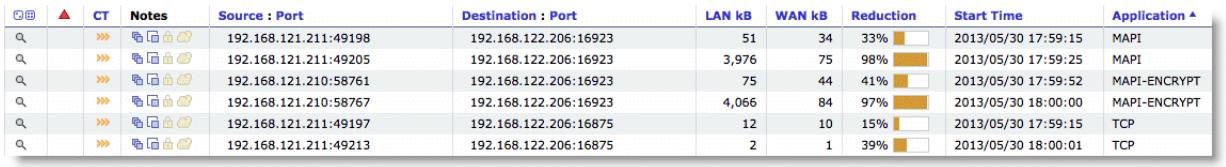


CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
		192.168.121.210:57346	192.168.122.206:7830	48	52	0%	2013/05/30 15:14:49	MAPI
		192.168.121.210:57329	192.168.122.206:7830	68	74	0%	2013/05/30 15:14:40	MAPI

You can see the WAN KB values are higher than the LAN KB values. For details on how to troubleshoot this problem, see the *Steelhead Appliance Management Console User's Guide*.

Enabling support for encrypted MAPI when there is a mixture of both encrypted Outlook clients and native Outlook clients allows the native clients to benefit from optimization.

Figure 2-5. Encrypted Outlook Clients and Native Outlook Clients



CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
		192.168.121.211:49198	192.168.122.206:16923	51	34	33%	2013/05/30 17:59:15	MAPI
		192.168.121.211:49205	192.168.122.206:16923	3,976	75	98%	2013/05/30 17:59:25	MAPI
		192.168.121.210:58761	192.168.122.206:16923	75	44	41%	2013/05/30 17:59:52	MAPI-ENCRYPT
		192.168.121.210:58767	192.168.122.206:16923	4,066	84	97%	2013/05/30 18:00:00	MAPI-ENCRYPT
		192.168.121.211:49197	192.168.122.206:16875	12	10	15%	2013/05/30 17:59:15	TCP
		192.168.121.211:49213	192.168.122.206:16875	2	1	39%	2013/05/30 18:00:01	TCP

MAPI Admission Control for Microsoft Outlook

MAPI admission control for Microsoft Outlook are control measures to prevent the Steelhead appliance from entering admission control when there are many TCP connections. A*dmission control* is a state a Steelhead appliance enters when it has stopped optimizing new connections because it has exhausted a certain type of resource. During admission control, new connections pass through, while existing connections are optimized. Upon exiting admission control, the Steelhead appliance once again intercepts and optimizes new connections.

For more information about admission control, see the *Steelhead Appliance Deployment Guide* and the *Steelhead Appliance Management Console User's Guide*.

When you use an Exchange server and the clients are using Microsoft Outlook, Outlook initiates and maintains multiple TCP connections with the Exchange server. A *MAPI session* is the collection of multiple TCP connections from an Outlook client.

When Mircosoft Outlook connections are optimized, the Steelhead appliance remaps contexts to prevent inadvertent mixing of optimized and non-optimized traffic. MAPI differs from other protocols because the TCP connections of a session are not independent of other TCP connections from the same session.

Because of how Exchange servers interact with MAPI sessions, you must optimize all client MAPI connections by the same pair of client-side and server-side Steelhead appliances, or have them all set to pass through. If you optimize only some of the connections, the session fails. If the connections are not optimized by the same pair of Steelhead appliances, the session fails.

Prior to RiOS v7.0, MAPI admission control v1 used a threshold percentage used to calculate control. For example, if a Steelhead appliance had a 1000 connection limit, the admission control cutoff value of 85% was 850 connections. The remaining 150 connections were made available only for existing MAPI sessions.

MAPI admission control v2 in RiOS v7.0 and later is enhanced to include:

- Server-side Steelhead appliance awareness.

  MAPI admission control v1 was a client-side Steelhead appliance implementation only. MAPI admission control v2 continues as a client-side Steelhead appliance implementation, but it is also aware of the server-side Steelhead appliance.
- MAPI admission control v2 preemptively closes MAPI sessions to reduce the connection count in an attempt to bring the Steelhead appliance out of admission control. MAPI sessions are closed in the following order:
  - MAPI prepopulation connections
  - MAPI sessions with the largest number of connections
  - MAPI sessions with the most idle connections
  - The oldest MAPI session
  - MAPI sessions exceeding the memory threshold

While the Steelhead appliance is in the admission control state, a special handling of MAPI sessions is activated. Because new connections cannot be optimized for an existing client session, the Steelhead appliance closes all connections from this existing client. Outlook re-establishes connectivity to the Exchange server, and because the Steelhead appliance is in an admission control state, the new connections of the client are detected as pass-through connections.

MAPI Optimization with Steelhead Appliances in a Serial Cluster or a Parallel Deployment

MAPI admission control is helpful when there are two client-side Steelhead appliances in a serial cluster. MAPI optimization expects all connections from an Outlook client to be optimized by the same Steelhead appliance and not split across two Steelhead appliances. However, the thresholds in RiOS v7.0 and later significantly reduce the likelihood of this situation occurring.

In a typical parallel deployment in which you configure the Steelhead appliances as connection forwarding neighbors, the Steelhead appliances forwards new MAPI connections to the Steelhead appliance that detects the initial MAPI connection for the user session. This effectively avoids failure situations in which the MAPI connections for a single session are not optimized on the same Steelhead appliance.

MAPI Optimization with Exchange Clusters

In larger Microsoft Exchange deployments it is possible that the Exchange server comprises several nodes working in a cluster. With Exchange 2010, the cluster can have some form of load balancer to distribute connections between the various nodes in the cluster, including one or more client access servers (CAS) that act as a front end to the mailbox servers. You must configure the client-side Steelhead appliance with one of the following settings:

- enable port transparency for MAPI traffic
- enable full transparency for MAPI traffic

- disable MAPI port remapping with the CLI command **no protocol mapi port-remap enable**.

Note: You must restart the service for this command to take effect.

CAS requires RiOS v6.1.1 or later. There is not a specific configuration required on the server-side Steelhead appliance to support optimization for Exchange clusters.

Riverbed recommends RiOS v6.1.4 or v6.5.1 for Exchange clusters that use multiple CAS and Microsoft NLB for load balancing.

Outlook Anywhere Optimization

Outlook Anywhere (OA) is a new feature in Exchange 2003. It allows Outlook clients to connect to Exchange over HTTP or HTTPS. Remote users can connect to Exchange without using specialized VPN software. The Outlook Anywhere protocol uses a variant of the existing MAPI protocol transported over HTTP. In releases earlier than RiOS v6.5, the Steelhead appliance was unable to provide latency specific optimization for this traffic.

Outlook Anywhere optimized traffic uses the new RPC over HTTP optimization engine, as well as the existing MAPI, HTTP, and SSL optimization features.

Note: The Steelhead appliance must be properly licensed to use SSL. For details, see "SSL Required Components" on page 146.

Outlook Anywhere can leverage both traditional MAPI and encrypted MAPI. If you use encrypted MAPI, the server-side Steelhead appliance must be a member of the domain.

The following table shows encryption types using HTTP and encrypted MAPI.

Tunnel Type	Encryption
HTTP tunnel regular MAPI	Not encrypted
HTTPS tunnel regular MAPI	Encrypted once
HTTP tunnel encrypted MAPI	Encrypted once
HTTPS tunnel encrypted MAPI	Encrypted twice

The following list describes requirements for Outlook Anywhere:

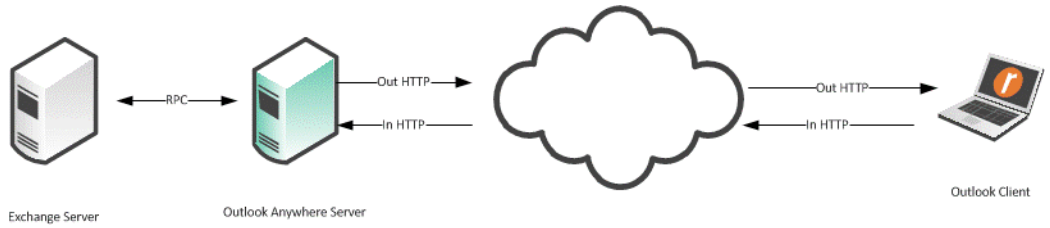
- RiOS v6.5 or later
- Microsoft Outlook and Exchange Server 2003 or later
- Encrypted MAPI requires the server-side Steelhead appliance to join the windows domain of the Exchange server (for details, see "Joining a Steelhead Appliance to a Domain" on page 42)
- HTTPS requires SSL licensing and an SSL certificate and key from the Outlook Anywhere server

Note: Outlook Anywhere optimized connections cannot start MAPI prepopulation.

The communication flow of Outlook Anywhere is between an Outlook client, the Outlook Anywhere server, and an Exchange server. The Outlook Anywhere server is a RPC Proxy server. The server takes RPC calls to and from the Exchange server, removes the RPC headers, and encapsulates the data in HTTP or HTTPS. Because MAPI is a full duplex protocol, two HTTP/HTTPS connections are required for each Outlook Anywhere connection. This is called a *virtual connection*.

Figure 2-6 shows the RPC connection between the Outlook Anywhere server and the Exchange server. The Outlook Anywhere server strips the RPC headers and encapsulates them in either HTTP or HTTPS, and sends it to the Outlook client. The Outlook client then strips the HTTP from the session and reads the traffic as RPC and MAPI.

Figure 2-6. A Virtual Connection



Outlook prefers to make connections with TCP/MAPI. Despite configuring Outlook to use Outlook Anywhere (RPC over HTTP) connections, Outlook sometimes checks to see if it can connect to the Exchange server with a TCP/MAPI connection.

Note: To disable fallback to TCP, enter an in-path rule on the client-side Steelhead appliance to prevent EPM to the Outlook Anywhere server. An in-path rule to deny MAPI TCP is not normally used in most field deployments, as HTTP and HTTPS are the only protocols allowed through the firewall.

To block RPC to the Outlook Anywhere server

- On the client-side Steelhead appliance, connect to the CLI and enter the following command:

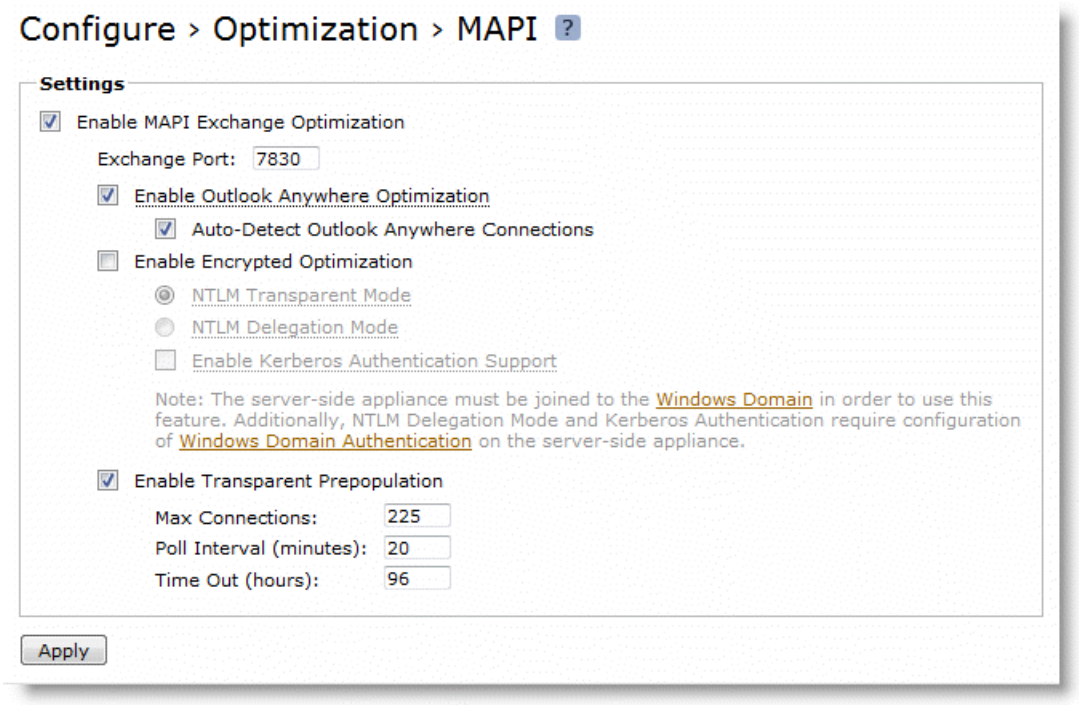
```
in-path rule deny rulenum [number] dstaddr [address] dstport 135 srcaddr 0.0.0.0 description
BlockRPC
```

To configure Outlook Anywhere

1. Configure Outlook Anywhere MAPI:
   - On the client-side and server-side Steelhead appliance, choose Configure > Optimization > MAPI.
   - Select Enable Outlook Anywhere optimization.
   - Select Auto-Detect Outlook Anywhere Connections.

- Click **Apply**.

Figure 2-7. Configuring Outlook Anywhere On The MAPI Page



Note: The corresponding CLI commands are **[no] protocol mapi outlook-anywhr enable** and **[no] protocol mapi outlook-anywhr auto-detect**.

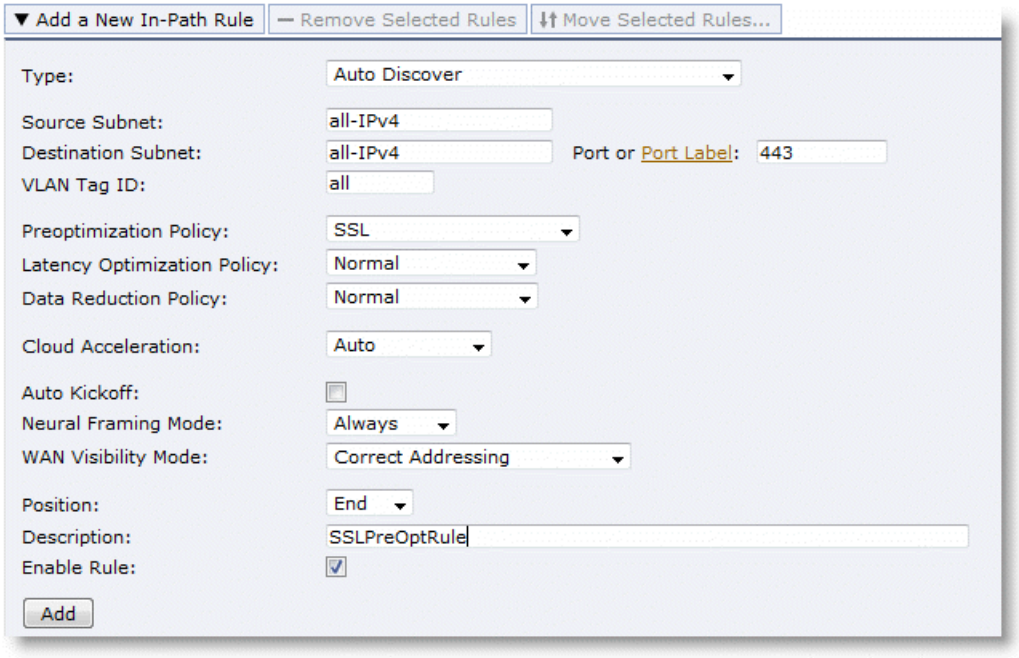
2. Configure an in-path rule for HTTPS connections to enable SSL preoptimization only if the SH has not had port 443 removed from the port label Secure. Normally port 443 is removed as part of the simple SSL configuration. For more details, see "Setting Up a Simple SSL Deployment" on page 148.

   To configure an in-path rule for HTTPS connections:

   - Choose Configure > Optimization > In-Path Rules.
   - Select Add a New In-Path Rule.
   - Select Auto Discover from the Type drop-down list.
   - Specify port 443.
   - Select SSL from the Preoptimization Policy drop-down list.

- Click **Add.**

Figure 2-8. SSL In-Path Rule for Outlook Anywhere Optimization



Note: You can configure an in-path rule for HTTPS connections to enable SSL preoptimization through the CLI by entering **in-path rule auto-discover preoptimization ssl dstport 443 rulenum end description SSLPreOptRule**.

3. Enable HTTP optimization the client-side and server-side Steelhead appliance. For details, see "HTTP Optimization" on page 63.

4. Enable SSL the client-side and server-side Steelhead appliance. The certificate and key from the Outlook Anywhere server must be installed on the server-side Steelhead appliance.

   If you are using an internal CA, the CA root certificate must be installed.

   If you are using encrypted MAPI you must enable secure inner channel. For details, see "Microsoft Exchange Email Optimization" on page 15.

Note: When in-path rules are applied at the Management Console, or when you use the CLI, you must complete a save operation to save your changes permanently.

Support for Common Access Card

A Common Access Card (CAC) is a type of smart card often used in military and other high-security environments. The cards are usually about the size of a credit card and contain a chip that holds one or more public-key infrastructure (PKI) certificates, and an identifier that is unique to the card holder.

You can configure Outlook clients and Exchange servers so that they communicate only with each other after the user has inserted the CAC into a card reader attached to the client and entered a PIN. When a CAC is used, the communication between the Outlook client and Exchange server is sent over a secure connection known as *S-Channel*.

You can configure Steelhead appliances to optimize traffic over S-Channel as part of an Outlook Anywhere over SSL (RPC over HTTPS) session. You must first configure your Steelhead appliances to optimize Outlook Anywhere over SSL, including installing the correct certificates.

After you complete this initial configuration, you must configure the server-side Steelhead appliance to enable support for S-Channel:

```
protocol mapi outlook-anywhr schannel enable
```

Next, configure the client-side Steelhead appliance to enable support for passive key derivation (PKD):

```
protocol ssl client-cer-auth enable
```

Verifying Connection Status

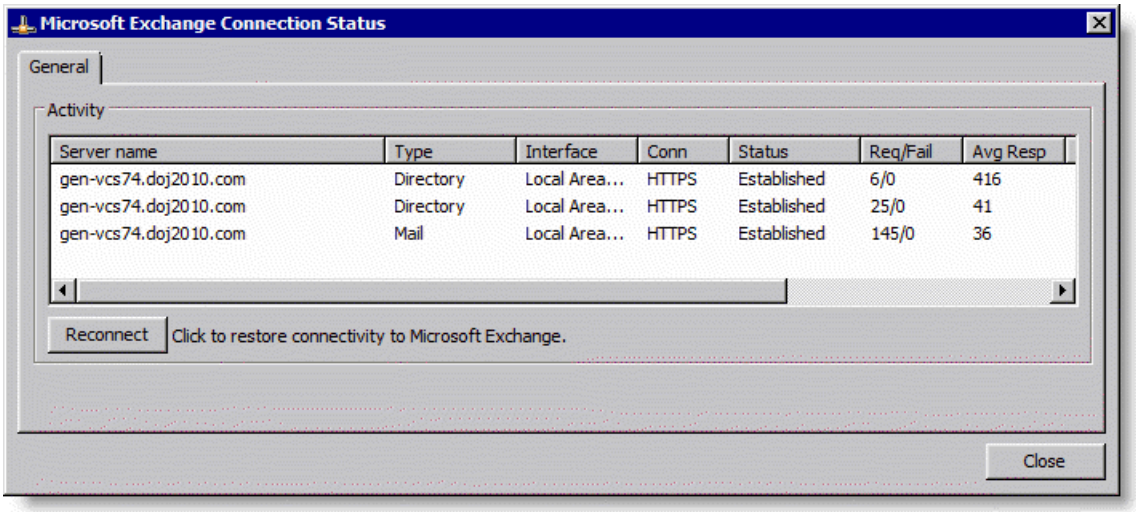
You can verify the Outlook connection status. It is important that you verify that the connection state is HTTP or HTTPS, and not TCP.

To view the Outlook connection status

1. Hold the CTRL key and click the **Outlook** icon in the task tray.
2. Select Connection Status.

   The Connection Status screen opens.

Figure 2-9. Connection Status Screen



View the Reports > Networking > Current Connections page to identify if the active connections are standard MAPI (MAPI-OA), or if they are encrypted MAPI (eMAPI-OA).

Figure 2-10. MAPI-OA Current Connections Page

CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
		10.37.80.20:56651	10.16.205.53:443	298	95	68%	2013/05/20 09:32:08	MAPI-OA
		10.37.80.20:56650	10.16.205.53:443	4	1	74%	2013/05/20 09:32:08	MAPI-OA
		10.37.80.20:56645	10.16.205.53:443	6	5	19%	2013/05/20 09:32:07	MAPI-OA
		10.37.64.21:49657	10.16.205.53:443	4	3	19%	2013/05/20 09:31:56	MAPI-OA
		10.37.64.21:49656	10.16.205.53:443	4	3	18%	2013/05/20 09:31:55	MAPI-OA
		10.37.64.21:49655	10.16.205.53:443	4	3	20%	2013/05/20 09:31:55	MAPI-OA
		10.37.64.21:49654	10.16.205.53:443	4	1	75%	2013/05/20 09:31:54	MAPI-OA
		10.37.64.21:49653	10.16.205.53:443	4	1	75%	2013/05/20 09:31:53	MAPI-OA
		10.37.64.21:49651	10.16.205.53:443	33	12	62%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49650	10.16.205.53:443	33	11	67%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49649	10.16.205.53:443	4	3	16%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49648	10.16.205.53:443	4	1	74%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49647	10.16.205.53:443	4	1	73%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49646	10.16.205.53:443	4	1	75%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49652	10.16.205.53:443	32	9	73%	2013/05/20 09:31:52	MAPI-OA
		10.37.64.21:49645	10.16.205.53:443	4	1	71%	2013/05/20 09:31:51	MAPI-OA
		10.37.64.21:49644	10.16.205.53:443	4	1	71%	2013/05/20 09:31:51	MAPI-OA
		10.37.64.21:49643	10.16.205.53:443	4	1	70%	2013/05/20 09:31:51	MAPI-OA
		10.37.64.21:49642	10.16.205.53:443	22	5	79%	2013/05/20 09:31:51	MAPI-OA
		10.37.80.144:54704	10.16.205.53:443	154	85	44%	2013/05/20 09:31:18	MAPI-OA

Figure 2-11. eMAPI-OA

CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
		10.37.64.20:59634	10.16.205.53:62405	3	2	22%	2013/05/20 04:33:09	TCP
		10.37.64.20:59629	10.16.205.53:7830	10	8	25%	2013/05/20 04:28:46	MAPI-ENCRYPT
		10.37.64.20:56935	10.16.205.53:443	5,781	2,105	63%	2013/05/19 06:50:24	eMAPI-OA
		10.37.64.20:56934	10.16.205.53:443	3,026	1,881	37%	2013/05/19 06:50:23	eMAPI-OA
		10.37.64.20:54239	10.16.205.132:7830	114	104	9%	2013/05/18 21:26:27	MAPI-ENCRYPT

Troubleshooting Outlook Anywhere Optimized Traffic

The following list describes how to troubleshoot issues with Outlook Anywhere optimized traffic.

- Change the Outlook Anywhere server to use HTTP rather than HTTPS
- Disable encrypted MAPI on the Client Access server running Outlook Anywhere

Riverbed recommends that you disable SSL and MAPI encryption to facilitate troubleshooting and easily decipher packet captures.

To change Outlook Anywhere server to use HTTP only

1. Change or create a registry key on the Outlook Anywhere server.

   The key is located at HKLM\Software\Microsoft\Rpc\RpcProxy\AllowAnonymous. It is a DWORD value set to 0x1. Copy and paste the following into a.reg file for faster implementation.

   ```
   Windows Registry Machine Editor Version 5.0
   [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Rpc\RpcProxy]
   "Allow Anonymous"=dword:00000001
   ```

2. Restart the Outlook Anywhere server you modify the registry (by either manual or automated registry modification).

By default, Exchange 2010 only accepts encrypted MAPI connections. This is not a problem for the Steelhead appliance because of the encrypted MAPI feature. However, for troubleshooting purposes, Riverbed recommends that you disable encrypted MAPI.

To disable MAPI Encryption on Exchange 2010

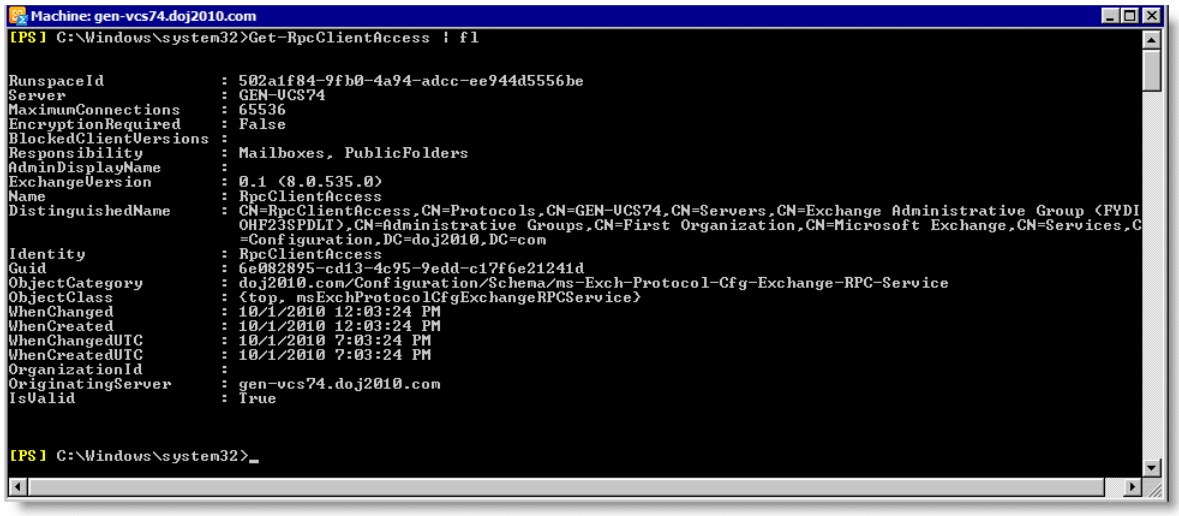
1. To determine if MAPI Encryption is enabled, launch the Exchange server management shell by choosing Start >All Programs> Exchange Server 2010 > Exchange Management Shell.

2. Enter the following command:

   ```
   Get-RpcClientAccess | fl
   ```

   Figure 2-12 shows an example output.

Figure 2-12. Exchange Management Shell Output



3. To determine if MAPI encryption is enabled, view the EncryptionRequired field. It has a value of True or False. A True value means that encryption is required and you must change it to False. If a False value is present, you can exit the shell.

4. Enter the following command to change the value to False:

   ```
   RpcClientAccess -server ServerName -EncryptionRequired $false
   ```

5. Restart the Exchange server.

Microsoft Exchange 2013 Optimization

Microsoft has made significant changes to Outlook MAPI-based traffic optimization in Microsoft Exchange 2013. Client access is simplified by elimination of the traditional End Point Mapper (EPM) and MAPI protocols. This new implementation enables users to more easily access to mailbox servers by leveraging well-known protocols HTTPS and HTTP. This change in protocols also enables organizations to define tighter network security for their Exchange environments by allowing only HTTP and HTTPS traffic from client-facing networks.

Figure 2-13 shows an example of client traffic from Exchange 2013 and Outlook 2013 using Wireshark.

Figure 2-13. Client Traffic Using Wireshark

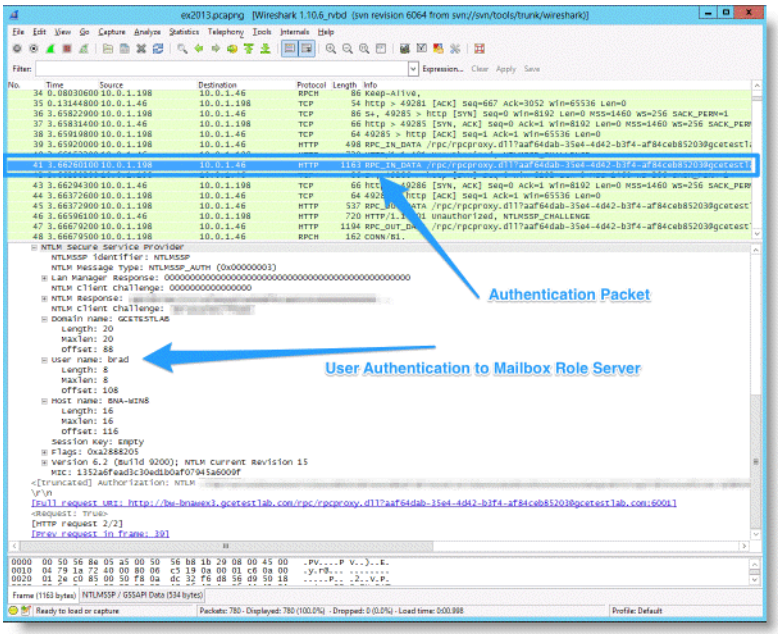


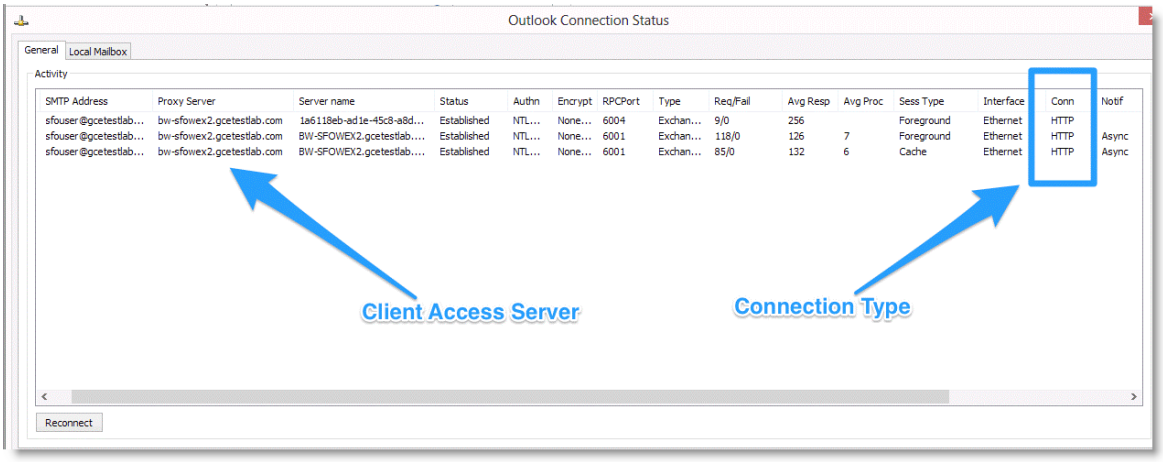
Figure 2-14 shows that the Exchange traffic is traversing HTTP TCP port 80 using a netstat -n output from the client running Outlook 2013.

Figure 2-14. Exchange Traffic on Port 80



Figure 2-15 shows the Outlook Connection Status window.

Figure 2-15. Outlook Connection Status Window



To take advantage of the MAPI-based traffic optimization change, you must configure your Steelhead appliances to optimize Outlook Anywhere traffic. RiOS enables Outlook Anywhere-TCP streams to be initially intercepted by the HTTP protocol optimization function, and then it transparently switches to Remote Procedure Call (RPC) over HTTP.

For details on configuring Outlook Anywhere, see "Outlook Anywhere Optimization" on page 23.

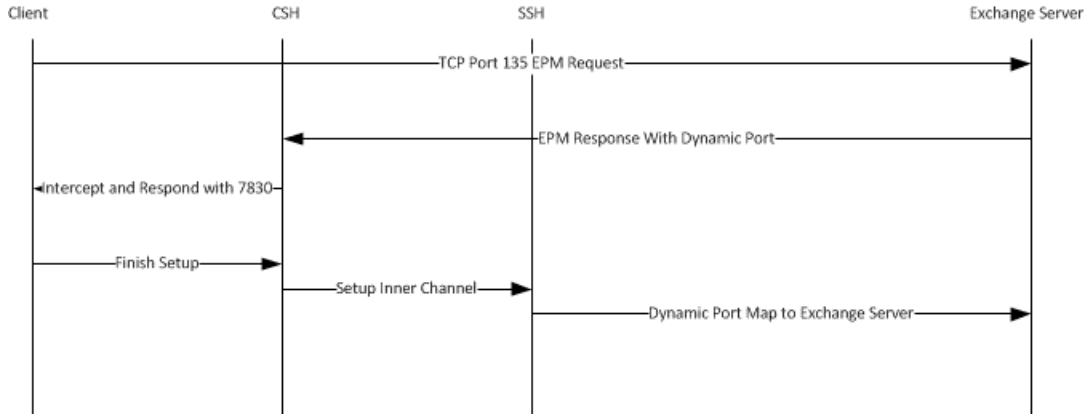
MAPI Destination Port Handling

If you place an Exchange server behind a firewall, you must use static MAPI ports for the firewall to statefully inspect traffic to and from the MAPI servers. Default dynamic port mapping is not available in this scenario. You must enable static MAPI ports to Exchange Client Access servers that are placed behind load balancers for RiOS to correctly track connections.

A typical MAPI connection has the following flow:

- The client initiates a connection to the Exchange server on port 135. The EPM requests a dynamic port from the Exchange server.
- The Exchange server responds with a dynamic port from which the client can connect to MAPI.
- The client-side Steelhead appliance intercepts the dynamic port response and moves it to port 7830. Port 7830 is the standard Steelhead appliance MAPI port.
- The client finishes initiating the MAPI connection with the client-side Steelhead appliance.
- The inner channel connection or optimized connection is established between the client-side Steelhead appliance and server-side Steelhead appliance.
- The server-side Steelhead appliance finishes the connection to the Exchange server on the dynamic port.

Figure 2-16. Flow of a Typical MAPI Connection



In high security environments it is desirable to hard code the Exchange server port in the event that multiple Exchange servers need multiple static ports. With a firewall between the clients and the Exchange server, an issued dynamic port cannot always be interpreted. The default behavior of the Steelhead appliance is to remap the port to a single dynamic port. This causes a problem for MAPI servers running on multiple defined ports and the operation fails.

To disable the remapping capabilities of the MAPI software

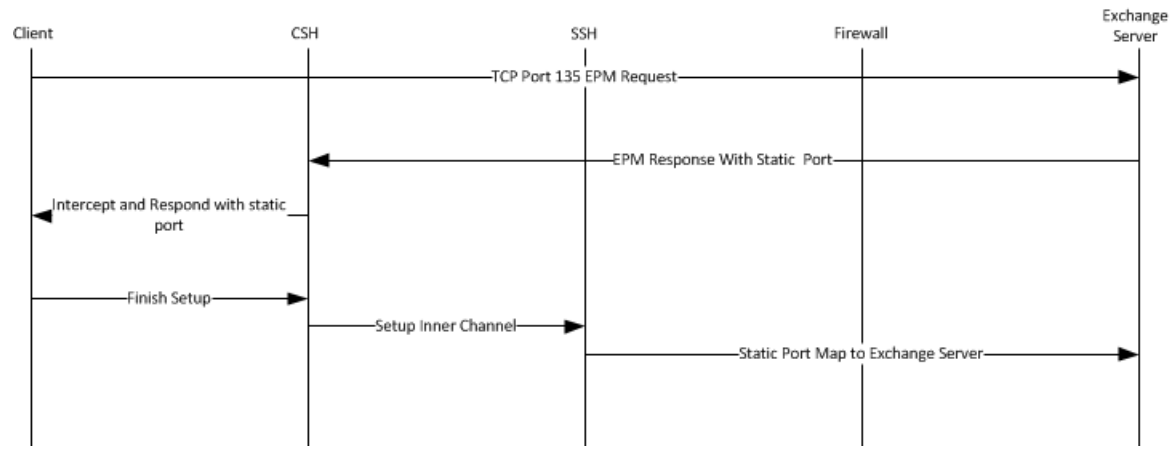
- On the client-side Steelhead appliance, connect to the CLI and enter the following commands:

```
no protocol mapi port-remap enable
write memory
service restart
```

After you disable re-mapping capabilities of the MAPI software, MAPI has the following flow:

1. The client sends the end point mapper request to the Exchange server.
2. The Exchange server responds with a static MAPI port.
3. The client-side Steelhead appliance intercepts the response.
4. The client-side Steelhead appliance issues the client the Exchange server address and the assigned static port.
5. The client-side Steelhead appliance finishes setting up the client side connection
6. The inner channel between the client-side Steelhead appliance and the server-side Steelhead appliance is completed.
7. The server-side Steelhead appliance completes the connection to the Exchange server on the assigned static port.

Figure 2-17. MAPI Flow with Static Ports



CHAPTER 3 Signed SMB and Encrypted MAPI Optimization

This chapter discusses high-level techniques and guidance for configuring signed service message block (SMB) and encrypted MAPI traffic, to ensure data integrity. This chapter includes the following sections:

- "Windows Security Concepts" on page 34
- "Domain Relationships" on page 35
- "Choosing an Authentication Mode for the Server-Side Steelhead Appliance" on page 37
- "Transparent Versus Delegation Mode" on page 38
- "Overview of Configuring SMB Signing and Encrypted MAPI" on page 40
- "SMB3 Optimization with Windows 8 Clients and Windows 2012 Server" on page 41
- "Joining a Steelhead Appliance to a Domain" on page 42
- "Configuring Constrained Delegation for Delegation Mode" on page 45
- "Kerberos" on page 45
- "Configuring the Server-Side Steelhead Appliance for Active Directory Integrated (Windows 2003/2008)" on page 50
- "Best Practices for the Steelhead Appliance in a Secure Windows Deployment" on page 51
- "Domain Authentication Scaling" on page 52
- "Domain Health Check and Domain Authentication Automatic Configuration" on page 53
- "Example Configurations" on page 60

Signed SMB and encrypted MAPI traffic use techniques to protect against unauthorized man-in-the-middle devices from making modifications to their exchanged data. Additionally, encrypted MAPI traffic and encrypted SMB3 traffic ensure data confidentiality by transmitting data with protection across the network. To securely optimize this traffic, a properly configured client and server-side Steelhead appliance:

- decrypts and removes signatures on received LAN side data from the client or server.
- performs bandwidth and application layer optimization.
- uses the secure inner channel feature to maintain data integrity and confidentiality of data transmitted over the WAN.
- converts the received optimized data back to its native form.
- encrypts and applies signatures for LAN side transmission of data to the client or server.

To query the Windows domain controller for the necessary cryptographic information to optimize this traffic, the server-side Steelhead appliance must join a Windows domain. The Steelhead appliance can require other configurations, both on the Steelhead appliance, and in the Windows domain. This cryptographic information is only useful for the lifetime of an individual connection or session. The information is obtained at the beginning of a connection, and transferred to the client-side Steelhead appliance as needed, using the secure inner channel feature. You must configure the secure inner channel to ensure maximum security.

Only the server-side Steelhead appliance is required to join the domain, and it does so using a machine account in the same way that a Windows device joins the domain using a machine account. The Steelhead appliance joins the domain this way to obtain a client user session key (CUSK) or server user session key (SUSK), which allows the Steelhead appliance to sign and/or decrypt MAPI on behalf of the Windows user that is establishing the relevant session.

The server-side Steelhead appliance must join a domain that is either:

- the user domain. The domain must have a trust with the domains that include the application servers (file server, Exchange server, and so on) you want to optimize.
- a domain with a bi-directional trust with the user domain. The domain might include some or all of the Windows application servers (file server, Exchange server) for Steelhead appliance optimization.

Production deployments can have multiple combinations of client and server Windows operating system versions, and can include different configuration settings for signed SMB and encrypted MAPI. Therefore it is possible that the security authentication between clients and servers can use NTLM or Kerberos, or a combination of the two. This chapter includes more details on authentication types and Steelhead appliance configuration requirements.

Windows Security Concepts

The Windows security framework is based on a formal structure known as a *domain*. Inside the domain is a logical group of host resources (primarily clients and servers, but also printers and other peripherals, and so on) that share a central directory database. The database resides on one or more servers known as *domain controllers* and contains user accounts and security information for all the resources in the domain. Other domains can coexist alongside and are joined together through a *Trust Relationship* to allow resources to securely communicate between each other even though they are in different domains with their own domain controllers.

When users and clients are accessing server resources (like file servers and Exchange servers), their credentials are validated against the database on the domain controller. This ensures that the client and the user have the correct security privileges to be able to access resources that provide signed SMB traffic or encrypted MAPI traffic.

There are several techniques and protocols used to validate the credentials of the user and client. These vary according to the Windows operating system version and application configuration (for example, Microsoft Outlook and Microsoft Exchange) on both the client and server. Example protocols include Kerberos, NTLMv1, and NTLMv2. The minimum RiOS version required to perform optimization in a Windows security environment is based on the protocol used, because RiOS has added more capabilities in newer versions.

Domain Relationships

Some organizations might have more than one Windows domain in use in their environment. A Steelhead appliance, like a Windows server, can only join a single domain. Therefore, the choice of which domain the Steelhead appliance should join depends on the domain where the file or mail servers are located, and the type of trust relationship between the Steelhead appliances potential domain, the file or mail servers domain, and the domain containing a user's credentials.

Figure 3-1 shows an example of a simple, single domain structure. All resources (clients, servers, and so on) that have joined the same domain are subject to the domain permissions and authentications, and can access the other available resources. Only a single domain controller is required in this case although you can have more than one domain controller for resilience. The Steelhead appliance must join the one available domain as a precursor to secure Windows protocol optimization.

Figure 3-1. Single Domain Structure

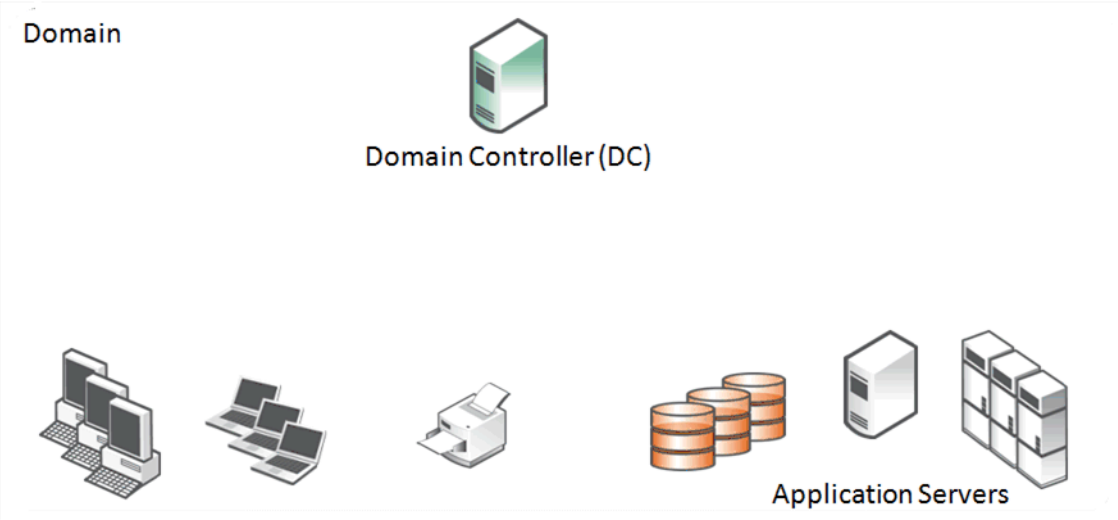


Figure 3-2 shows an example where clients are in one domain and servers are in a second domain. There is a trust relationship between the two domains that allow the clients and servers to access each other. The trust relationship is a *one-way trust*. The client domain is described as the *Trusted Domain* and the server domain is described as the *Trusting Domain*. The arrow that indicates the direction of trust is from the trusting domain to the trusted domain. Because of the one-way trust, only the resources in the client domain are allowed to access the resources in the server domain and not the other way around. Each of the two domains has its own domain controller, each with their own database for the resources in its domain.

Figure 3-2. Two Domains and One-Way Trust Structure

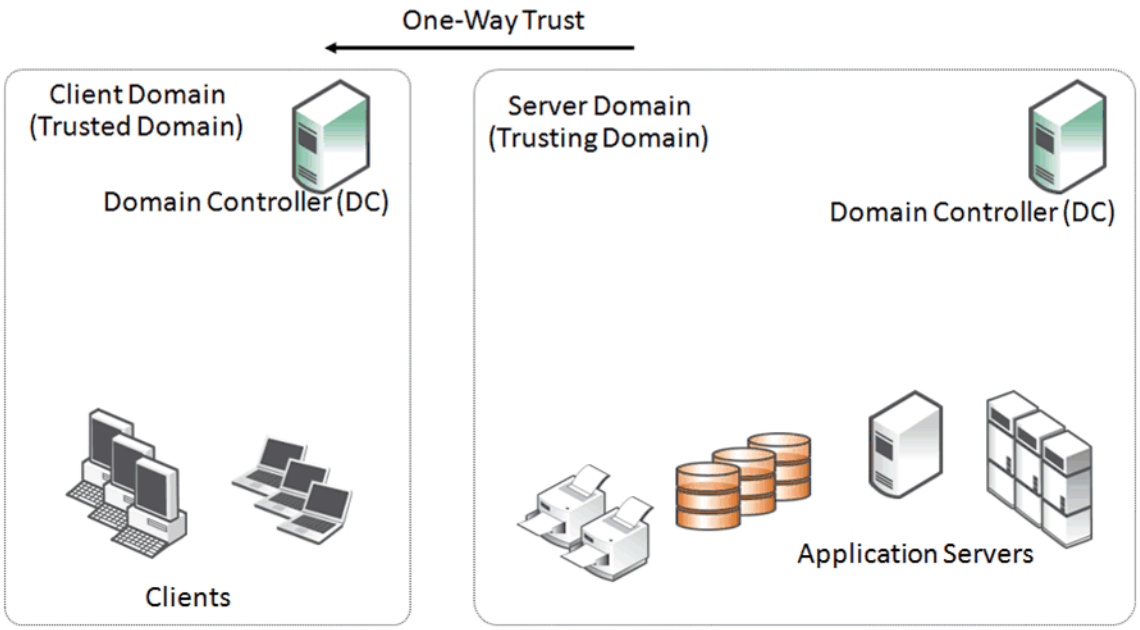
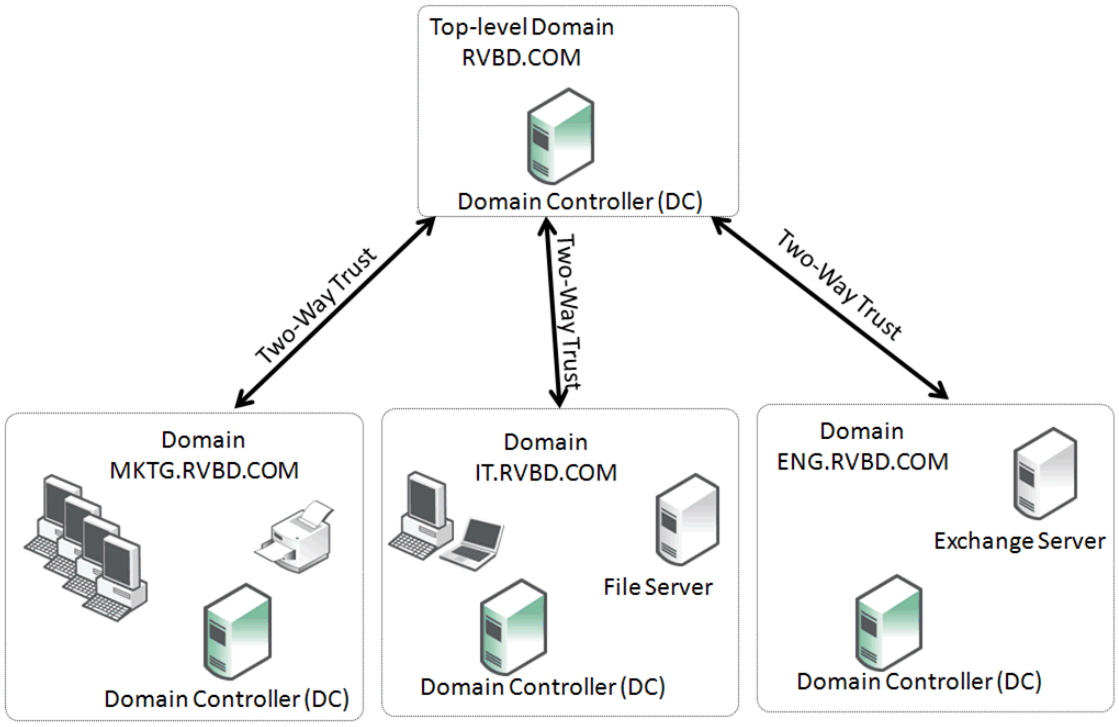


Figure 3-3 shows an example of a configuration where there are multiple domains. In a two-way trust, each of the resources in the child domains can access the other domains through the top-level (parent) domain as long as the correct permissions have been set up within the child domains.

Figure 3-3. Multiple Domains with Two-way Trust Structure



There can be a mixture of domains with domain controllers with different Windows operating system versions and a mixture of trust types. *Native mode* is when all domain controllers run the same version of operating system, and *Mixed mode* is when there are different versions of the operating system.

Choosing an Authentication Mode for the Server-Side Steelhead Appliance

There are many combinations and settings for Windows client operating system, Windows domain level, Windows authentication method, and RiOS version. You can choose from several different configuration options on the server-side Steelhead appliance. In some cases, you can select multiple options to provide the best possible flexibility for your deployment. The options are as follows:

- **Transparent mode** - Authenticate using NTLM pass-through authentication to the Windows domain controller. This option is considered the easiest to deploy with the minimum of administrative overhead. However, Steelhead appliances running RiOS v6.5 or earlier have restrictions on supported Windows clients. With RiOS v7.0 and later, you can use Active Directory integrated join mode (Windows 2008/2003) so that the server-side Steelhead appliance can join the domain with very limited privileges. When you join the domain this way in RiOS v7.0 or later, it supports a much wider selection of Windows client types.

- **End-to-end Kerberos** - When you need the authentication between Windows clients and servers to be Kerberos from end to end, the server-side Steelhead appliance can make use of a replication user account in the Windows domain. While this is the only choice available when the authentication is required to be end-to-end Kerberos, you can combine it with Transparent mode to provide the most flexibility in the event that some clients still negotiate NTLM authentication.
- **Delegation mode** - Authenticate using Kerberos constrained delegation through a delegate user account in the Windows domain. Delegation mode uses a combination of both NTLM and Kerberos authentication in the optimized connection between client and server. These options are discussed in detail throughout the remainder of this chapter. Riverbed recommends that you thoroughly read this chapter to make an informed choice. Delegation mode requires the most administrative effort to maintain configurations, both on the server-side Steelhead appliance and within Active Directory. Consider this option only if Transparent mode is not a viable solution.

Transparent Versus Delegation Mode

The server-side Steelhead appliance obtains the necessary cryptographic information to optimize SMB signed or encrypted MAPI traffic using one of the following modes:

- Delegation mode
- Transparent mode

This section requires that you be familiar with Delegation and Transparent modes.

Constrained delegation is commonly used in production environments to allow a trusted service to authenticate and query specific security information on behalf of a user. Microsoft did not design this feature specifically for use with only Steelhead appliances. For example, you can use constrained delegation in many Sharepoint and SSL VPN architectures to authenticate a user through their Windows credentials.

In a Steelhead appliance deployment, delegation mode uses Kerberos authentication between the server-side Steelhead appliance and any configured servers participating in the signed session (the behavior of RiOS v5.5.x and later). NTLM is used between the client machine and server-side Steelhead appliance. Delegation mode in RiOS v6.1 and later supports all clients, including Windows 7.

Transparent mode uses NTLM authentication end-to-end between the client-side and server-side Steelhead appliances, and the server-side Steelhead appliance and the server. Transparent mode in RiOS v6.1 and later supports all Windows servers, including Windows 2008 R2, that have NTLM enabled or allowed. Transparent mode in RiOS v7.0 and later supports all Windows client operating systems, including Windows 7. Earlier versions of RiOS support all Windows client operating systems, but Windows 7 requires registry changes.

When you use Delegation mode, a service principal name (SPN) for the delegate user is created using the setspn.exe command-line tool. The Windows Server 2003 SP1 Support Tools product CD includes this tool, or you can download it from the Microsoft Download Center.

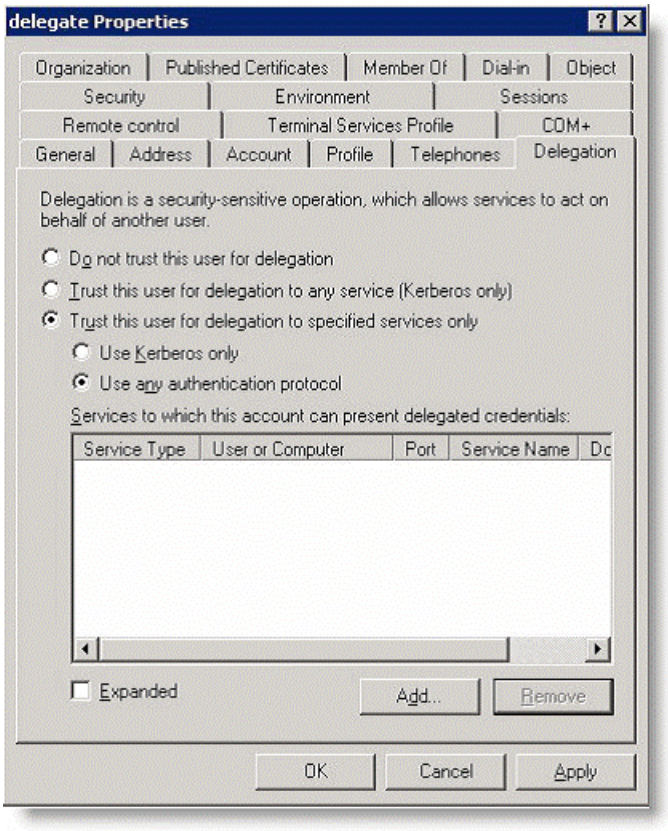
The SPN:

- must be unique because the domain controller assigns the Kerberos ticket for it.
- cannot be used by another service.
- cannot be `cifs/<hostname of domain controller>` or `mapi/<hostname of domain controller>`, which are used by the CIFS and MAPI services (for example, `c:\> setspn.exe -A cifs/delegate delegate_user)`.

To grant a user access for Delegation mode for the CIFS or MAPI service in Windows

1. From the Domain Controller, launch the Active Directory Users and Computers tool.
2. Right-click the user name (in this example the user name is *delegate*).
   This opens the Properties window.
3. Select Trust this user for delegation to specified services only.
4. Select Use any authentication protocol.
5. Click **OK**.

Figure 3-4. Delegate Properties



Transparent mode in RiOS v6.1 does not support:

- Windows 7 clients using default authentication settings.
- Windows 2008 R2 domains that have NTLM disabled.
- Windows servers that are in domains with NTLM disabled.
- Windows Vista SP2 clients when optimizing encrypted MAPI.
- EMC Celerra NAS.

Transparent mode in RiOS v7.0 or later supports:

- Windows 7 clients using default authentication settings
- Windows 2008 R2 clients, in which a Windows 2008 R2 server is initiating the TCP connection.

For details on establishing the correct minimum version of RiOS, see "RiOS Version Compatibility with Domains and Domain Relationships" on page 183.

From a security perspective, it important to note:

- In Transparent mode, the Steelhead appliance does not obtain any Windows user names or passwords. The CUSK and SUSK (hashed by the Windows domain) are only maintained for the duration of the client-to-file server or Exchange server session. No credentials are stored on either the client-side or server-side Steelhead appliance when you use Transparent mode.
- In Delegation mode, only the active directory-controlled delegate account is stored in the server-side Steelhead appliance configuration. The delegate account defined and controlled in Active Directory is granted trust only for delegation to the CIFS or MAPI service on specific servers.

Overview of Configuring SMB Signing and Encrypted MAPI

A deployment that successfully optimizes SMB signing and encrypted MAPI traffic requires gathering information about the Windows environment, configuring the Steelhead appliances with the appropriate RiOS version and configuration; when using Delegation mode or end-to-end Kerberos, this same deployment potentially requires more configuration in the Windows domain.

The following is a general overview on how to configure a typical SMB signing and MAPI encrypted deployment (for complete details, see the *Steelhead Appliance Management Console User's Guide*):

1. Determine if the domain containing users to optimize is different from the server domain.

   If the user domain is different from the server domain, determine the trust relationship between the domains.

2. Gather version information about the Windows environment:

   - Client operating system versions
   - Domain functional level of all server domains involved in the configuration (for example, Native 2003, Mixed mode, Native 2008-R2, Native 2012, and so on)

   Typical environments allow the Windows client and server to negotiate either NTLM authentication or Kerberos authentication for SMB or MAPI traffic. In these environments, the Steelhead appliance can force negotiation to NTLM authentication to provide latency and bandwidth optimization. Certain environments do not negotiate NTLM or Kerberos—instead they require Kerberos authentication. The Kerberos authentication is set either through settings on the client or through domain policy settings. RiOS v7.0 or later supports this type of end-to-end Kerberos authentication. If your environment is Kerberos only, see "Kerberos" on page 45.

3. Using the information provided in "RiOS Version Compatibility with Domains and Domain Relationships" on page 183, determine:

   - the minimum RiOS version needed for the client-side Steelhead appliance.
   - the minimum RiOS version needed for the server-side Steelhead appliance.
   - if Transparent mode is sufficient for optimization (preferred), or if Delegation mode is required.

4. Upgrade the RiOS version on the client- and server-side Steelhead appliances, if necessary.

5. If not already implemented, configure the secure inner channel feature on both the client-side and server-side Steelhead appliance.

6. Join the server-side Steelhead appliance to the domain. The domain can be the user domain or a domain that has a trust with the user domain: for example, the server domain. For details, see "Joining a Steelhead Appliance to a Domain" on page 42.

   In environments where the server domain only has a one-way trust to the user domain, you need a special configuration when joining the server-side Steelhead appliance to a domain. For details, see "One-Way Trust Configuration" on page 43.

7. If Delegation mode is required, set up the delegation accounts in the server domains and configure Delegation mode on the server-side Steelhead appliance. For details, see "Configuring Constrained Delegation for Delegation Mode" on page 45.

8. Verify successful optimization of the SMB Signed or encrypted MAPI traffic.

SMB3 Optimization with Windows 8 Clients and Windows 2012 Server

Microsoft created an update to the SMB protocol with the release of Microsoft Windows 8 and Server 2012 operating systems. This new version is officially known as SMB3, but in some early documentation it might be referred to as SMB2.2.

The following new features included in SMB3 are relevant to your Steelhead appliance deployment:

- Encryption between client and server
- New SMB signing algorithm
- Secure dialect negotiation

You can use all three features only if the client and server are Windows 8 and Windows 2012 servers respectively. If the client is Windows 7 or earlier, or the server is Server 2008-R2 or earlier, the client and server do not use the new features for reasons of backward compatibility—with the exception of secure dialect negotiation.

Secure dialect negotiation is enabled and used by default in Windows 8 and Server 2012 connections. You can optionally use secure dialect negotiation with SMB2 when you are setting up a connection to a server running Server 2008-R2, but this is not normally enabled, for compatibility reasons.

With the release of Windows 8.1 and Windows Server 2012 R2, Microsoft has added support for SMB3.02.

For SMB3 and SMB3.02 specifications, go to http://msdn.microsoft.com/en-us/library/cc246482.aspx.

RiOS prior to v8.5 provides some limited optimization of SMB3, but in some cases, there is no optimization at all. For more information about SMB3 and releases prior to RiOS v8.5, see https://supportkb.riverbed.com/support/index?page=content&id=S16547.

In RiOS v8.5 or later, you can provide full optimization for signed SMB3 traffic between Windows 8 clients and Server 2012 servers, as well as other client/server combinations that result in the use of signed SMB3 traffic.

Additionally, in RiOS v8.5 or later, you can optimize SMB3 traffic that is encrypted at the file share or file server level.

To optimize SMB3 traffic, you must run RiOS v8.5 or later on both the client-side and server-side Steelhead appliances, and join the server-side Steelhead appliance to the domain. Additionally, you must enable SMB3 optimization on both the client-side and server-side Steelhead appliances.

Important: Even if you do not configure Windows 8 clients and Server 2012 servers to require signing and encryption, the SMB3 protocol continues to use secure dialect negotiation during the connection setup. Therefore, the configuration settings outlined in this section are mandatory to ensure SMB3 traffic is optimized.

If you correctly configure the Steelhead appliances to optimize signed SMB3 traffic, there is no additional configuration required to support the optimization of signed SMB3 traffic that is encrypted at the share or server level.

For more details on the configuration settings required for SMB3, see the *Steelhead Appliance Management Console User's Guide*. For information on optimizing traffic between Windows 8.1 and Server 2012R2, see https://supportkb.riverbed.com/support/index?page=content&id=S23053.

Joining a Steelhead Appliance to a Domain

RiOS v6.1 or later supports joining domains in native or mixed modes for the following domain functional levels:

- Windows 2000
- Windows 2003 R2
- Windows 2008
- Windows 2008 R2
- Windows 2012

For more information about maximum latency between a Steelhead appliance and a domain controller, see http://supportkb.riverbed.com/support/index?page=content&id=S22110.

Before you join a Steelhead appliance to a domain, perform the following prerequisite verifications:

- The primary interface on a Steelhead appliance must have IP reachability to any domain controllers, DNS servers, and other resources that a Windows server or workstation typically has in the domain that the Steelhead appliance is joining. By default, the Steelhead appliance uses the primary interface as the source interface for the join operation. Other interfaces, such as the auxiliary and in-path management interfaces, might work if enabled, but they are not supported for authentication traffic in general.
- The source interface for the Steelhead appliance must not have its communication to the Windows domain controller blocked by a firewall or other security mechanism—for example, the RiOS Management ACL mechanism—either during the join or while optimizing connections. For details on the ports and protocols required for operation, go to http://support.microsoft.com/?id=179442.
- Configure the Steelhead appliance to use the same DNS server that other members of the domain use. Active Directory uses the DNS protocol to discover domain resources. When a Steelhead appliance is joining a domain, it needs to make DNS based requests to a DNS server that understands these Active Directory specific operations.
- The source interface for the join must have an A record in the DNS domain associating the Steelhead appliance hostname to its IP address.

- The Steelhead appliance must have its clock synchronized to within a few seconds of the domain controller it communicates with. The best practice is to have the Steelhead appliance use the NTP protocol to synchronize with the NTP server that is used for clock synchronization within the Windows domain.

  Note: A clock that is not synchronized on the Steelhead appliance is the most common cause of domain join errors.

- Credentials for a domain account with sufficient privileges to join a machine to the domain must be temporarily available. Many organizations have specific accounts that are used only for the purpose of joining machines to the domain, and have no other privileges. Enter the credentials on the Steelhead appliance to perform the join operation. The credentials are not stored on the Steelhead appliance. If you have multiple Steelhead appliances in the data center (for example, in a high-availability configuration), then you can add the same replication user account to the configuration of all the server-side Steelhead appliances.

If the server-side Steelhead appliance is running a version of RiOS between v6.1 and v6.5, it can join the domain only to appear as a *Workstation*. In RiOS v7.0 or later, the Steelhead appliance can join the domain and appear as one of three different roles: Workstation, Active Directory integrated (Windows 2003), or Active Directory integrated (Windows 2008). When the Steelhead appliance appears as one of these roles, it has very limited functionality.

For more details, see "Configuring the Server-Side Steelhead Appliance for Active Directory Integrated (Windows 2003/2008)" on page 50.

During the join operation to appear as a Workstation, the Steelhead appliance communicates with domain controllers and other entities related to the Windows domain, including the DNS server. The exact operations performed depend on the domain functional level, but are similar to the operations made when a new Windows server joins the domain. RiOS v6.1 or later can join domains using NTLMv1, NTLMv2, or Kerberos for authentication.

By default, the Steelhead appliance appears in the Computers Organizational Unit (OU) in the domain. This is the default for Windows member servers joining a domain. You can use the CLI for a domain join and specify a different OU.

The following example shows a Steelhead appliance joining the domain RVBD.COM and placing the Steelhead appliance in the WAN-opt OU:

```
domain join domain-name RVBD.COM login join-account password join-password org-unit WAN-opt
```

One-Way Trust Configuration

You need RiOS v6.1 or later when the user is in a different domain from the servers, and when the server domain has a one-way trust relationship with the user domain. In this scenario, the user domain is the *trusted domain*, and the server domain is the *trusting domain*. This setup is occasionally deployed in organizations that have a number of subsidiary companies or between two companies that have recently merged but wish to keep their IT separate for a period of time.

More typically this configuration is found when the user organization has file or mail services hosted by an external service provider. For example, with Microsoft Office 365 Dedicated service, this setup enables the Steelhead appliance to use authentication of the Exchange servers that provide Microsoft Exchange online services.

In environments in which there is a one-way trust between server domains and user domains, the server-side Steelhead appliance is joined to the user domain, or any domain that trusts the domain of the client user. Because of the one-way trust, the server-side Steelhead appliance cannot automatically build a list of domains on the other side of the trust. Therefore, you must explicitly configure the server domains that trust the user domain on the server-side Steelhead appliance, using the **protocol domain-auth oneway-trust** CLI command.

For example, if the users in the RVBD.COM domain use file or mail servers in the SERVERS.PROVIDER.COM domain (whose shortened NetBIOS domain name is SERVERS), you first ensure that the server-side Steelhead appliance is joined to the RVBD.COM domain and then use the following commands on the server-side Steelhead appliance:

```
protocol domain-auth oneway-trust dns-name SERVERS.PROVIDER.COM netbios-name SERVERS
```

You can view the list of configured one-way trust domains with the **show protocol domain-auth oneway-trust** CLI command and, if needed, you can remove domains with the **no protocol domain-auth oneway-trust dns-name …** CLI command.

In RiOS v7.0 or later, the support for one-way trusts is further enhanced to include Windows 7 clients without requiring a registry change on the client. You must join the server-side Steelhead appliance to the domain using the Active Directory integrated (Windows 2003/2008) mode and then execute the CLI command as indicated previously.

Support for Kerberos authentication through a one-way trust is included in RiOS v8.5 or later. Versions of RiOS prior to v8.5 support only NTLM authentication in one-way trust configurations.

Enabling Kerberos in a Restricted Trust Environment

When you use Kerberos authentication, there are some additional one-way trust restrictions if you use external managed services such as Microsoft Office 365 Dedicated. These restricted trust models are deliberately designed with split resource and management Active Directory domains.

Within a hosted services deployment, or any other deployment in which there is a restricted one-way trust, the server-side Steelhead appliance cannot use the replication user account to contact the resource domain controller for the session key.

The server-side Steelhead appliance joins the user account domain and uses the replication user account to communicate with the domain controller in the user domain.

In addition, the client-side and server-side Steelhead appliances use a Kerberos feature in RiOS v8.5 to intercept traffic over TCP port 88, the port most used by Kerberos.

By intercepting the Kerberos exchanges (which are initiated by the client when setting up an authenticated connection to a server), the server-side Steelhead appliance can obtain a copy of the session key used between the client and server. This action enables the Steelhead appliances to optimize the signed SMB or encrypted MAPI session in a restricted trust deployment.

Note: The Kerberos feature must use TCP.

Interestingly, Windows XP clients, by default, use UDP for Kerberos authentication. You must reconfigure Windows XP clients to use TCP, if necessary.

For more information about configuring Steelhead appliances for restricted one-way trusts, see "Configuring the Server-Side Steelhead Appliance for Active Directory Integrated (Windows 2003/2008)" on page 50 and the *Steelhead Appliance Management Console User's Guide*.

Configuring Constrained Delegation for Delegation Mode

When you use Delegation mode to optimize SMB Signed or encrypted MAPI traffic, additional configuration (beyond joining the server-side Steelhead appliance to a domain) is required because Delegation mode uses the Active Directory constrained delegation feature. Configuration is required on both the server-side Steelhead appliance and in the Windows domain that it joins, as well as in any other Windows domain in which there are servers you want optimized by the same server-side Steelhead appliance.

Constrained delegation is an Active Directory feature that allows configured services to obtain security related information for a user. Configuring constrained delegation requires the creation of a special delegate user account in the Windows domain. The account allows the delegate user the privilege of obtaining security information for use with specific applications (like CIFS and MAPI), and then configuring the delegate user credentials on the server-side Steelhead appliance.

For instructions on the Steelhead appliance and Windows domain configuration for constrained delegation, see the *Steelhead Appliance Management Console User's Guide*.

In RiOS v7.0 or later, any new delegation user account you add to the Steelhead appliance configuration is automatically saved in the secure vault.

Note: In RiOS v7.0 or later, the secure vault is locked on a new Steelhead appliance. The secure vault is also locked on a Steelhead appliance that is upgraded to RiOS v7.0 or later. You must unlock the secure vault to view, add, remove, or edit any delegate user configuration details that are stored on the Steelhead appliances.

In RiOS v6.5, the delegate user details are stored in an obscured form within the general Steelhead appliance configuration database. When upgrading from RiOS v6.5 to v7.0 or later, any preexisting delegate user account information is not automatically moved to the secure vault. Instead, you must use the following CLI command to manually migrate the information:

```
protocol domain-auth migrate
```

Use the following command to confirm whether the credentials have been migrated:

```
show protocol domain-auth credentials location
```

Kerberos

Kerberos is a network authentication protocol. In a Microsoft Windows environment, the Active Directory domain controller maintains user account and login information to support the Kerberos service. From a corporate perspective, you can think of Kerberos as guarding against unauthorized access to your IT assets.

This section includes the following topics:

- "Overview of Kerberos" on page 46
- "Optimization in a Native Kerberos Environment" on page 48
- "Domain User with Replication Privileges" on page 49
- "Configuring Traffic Optimization for HTTP (SharePoint), Encrypted MAPI, and Signed SMB/SMB2/SMB3" on page 49

Important: Kerberos authentication is not unique to a Microsoft Windows environment. This guide contains explanations and details limited to a Windows configuration.

Overview of Kerberos

The three components of Kerberos are the key distribution center (KDC), the client user, and the server resource with the service that the client user wants to access. The KDC is a part of the Windows domain controller and performs two service functions: the authentication service (AS) and the ticket-granting service (TGS).

To enable the client user to access a server resource (for example, a Windows file server), three exchanges are involved:

- The AS exchange
- The TGS exchange
- The AP (application protocol) exchange

When users initially log in to a Microsoft Windows network, they must provide a login name and password for access. These credentials are verified in the AS exchange of a KDC within the domain of the user. The KDC has access to Active Directory user account information. When a user is successfully authenticated, the authentication server grants the user a ticket to get tickets (TGT) that is valid for the local domain—you are first approved to get tickets, and next you are approved for each ticket as you ask for them. The TGT (sometimes referred to as the *AS reply*, although the AS reply actually contains more parts) has a default lifetime of 10 hours and can be renewed during the user's session without requiring the user to reenter a password.

The AS reply includes two components: the TGT itself, which is encrypted with a key that only the KDC (TGS) can decrypt; and the first of two session keys encrypted with the user's password hash. Any future communication with the KDC uses this initial session key (SK1). The client uses the TGT, which is cached in volatile memory space on the client machine, to request sessions with services throughout the network.

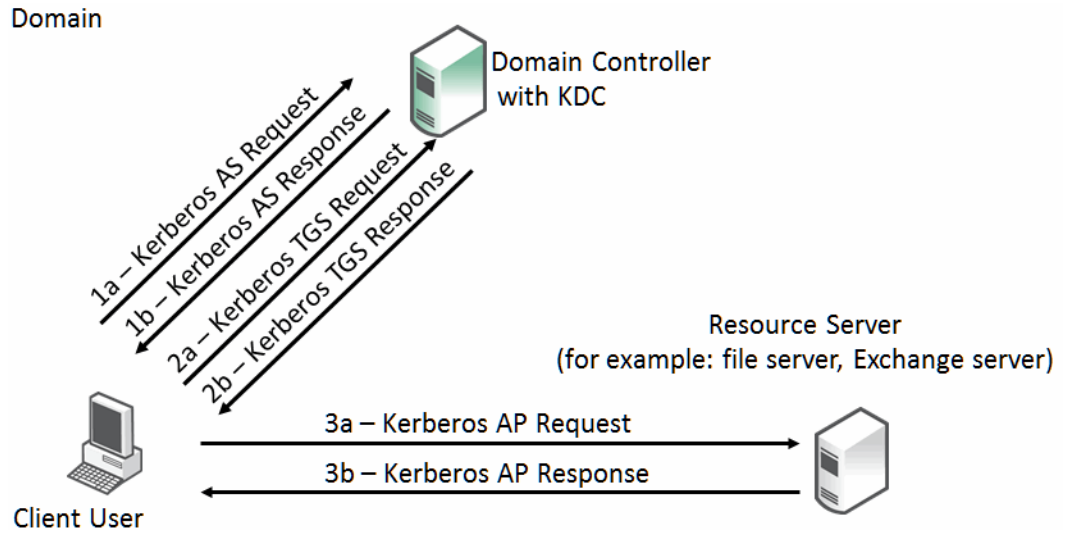
The client sends a request to the TGS, and asks for credentials of the desired server. The client first creates an *authenticator* using the SK1. The authenticator contains the user name, IP address, and a timestamp—all encrypted using SK1. The client next sends a copy of the TGT and authenticator to the TGS. The TGS responds with the server credentials in the form of a service ticket that also has two components: one encrypted in the server's key, which the client cannot read; and the other encrypted with the client's key, which is temporary. The encryption key is the second of the two session keys (SK2) issued during the sequence of exchanges. The SK2 is valid only for communicating with the desired server and lasts for the lifetime of the session with the desired server. A second or subsequent session with the same server needs a completely new SK2 session key requested by the client from the TGS.

Next, the client creates another autenticator, this time using SK2 for encryption. The client transmits the autenticator with the service ticket, which contains the client identity and a copy of the session key—SK2—all encrypted with the server's key, to the server. As a result of this exchange, the SK2 is now held by the client and server. SK2 authenticates the client and can optionally authenticate the server. The SK2 can also be used to encrypt further communication between the two hosts or to exchange a separate subsession key used for encrypting further communication.

The AP exchange is also known as the client/server exchange because the exchange is often completed with whatever application protocol is required between the client and server: for example, SMB and SMB2 for file sharing, MAPI for email, or HTTP for Sharepoint.

The Kerberos exchange process is shown in Figure 3-5.

Figure 3-5. The Three Kerberos Exchanges



Multiple Domain Environments and Referral Tickets

The Kerberos exchange occurs when the client user, KDS, and resource server are in the same Windows domain. Where there are multiple domains, there are steps that include referral tickets.

The AS and TGS functions are separate within the key distribution center. This enables the user to use the TGT obtained from the AS in the domain to obtain service tickets from a TGS in other domains. This is accomplished through referral tickets. A referral ticket is similar to the service ticket that is granted during the TGS exchange described earlier. The difference between the two is that the referral ticket contains a ticket for the KDC in the adjoining domain rather than the server.

If there is a trust established between two domains, referral tickets are granted to clients that request authorization for services in other domains. Because of the trust relationship between the two domains, an interdomain key, based on the trust password, is available for authenticating KDC functions.

For example, consider a client user in the local domain who seeks services in a foreign domain. The two domains are directly linked through a two-way trust. The basic steps are as follows:

1. The client performs a TGS exchange with its local domain KDC. The KDC recognizes a request for a session with a foreign domain server and responds by returning a referral ticket for the KDC in the foreign domain.

2. The client contacts the KDC of the foreign domain with the referral ticket. This ticket is encrypted with the interdomain key. Given that the decryption works, the TGS service for the foreign domain returns a service ticket for the server service in the foreign domain.

3. The client performs the client/server exchange with the server in the foreign domain and begins the user session with the service.

In a multiple domain configuration where the client is in one domain and the server is in a foreign domain, communication between the two can only be through an intermediate domain. Again, assume that there are transitive two-way trusts between the local domain and the intermediate domain, and the intermediate domain and the foreign domain.

The process is similar to the two-domain example. The basic steps are as follows:

1. The client performs a TGS exchange with its local domain KDC. The KDC recognizes a request for a session with a foreign domain server and responds by returning a referral ticket for the KDC in the intermediate domain.

2. The client contacts the KDC of the intermediate domain with the referral ticket. This ticket is encrypted with the interdomain key. The TGS service for the intermediate domain returns a referral ticket for the foreign domain.

3. The client contacts the KDC of the foreign domain with the intermediate domain referral ticket. The TGS service for the foreign domain returns a service ticket for the server service in the foreign domain.

4. The client performs the client/server exchange with the server in the foreign domain and begins the user session with the service.

For more details and reference material on Kerberos that are beyond the scope of this guide, see:

- http://technet.microsoft.com/en-us/library/bb742516.aspx
- http://web.mit.edu/kerberos/
- http://www.ietf.org/rfc/rfc4120.txt

Optimization in a Native Kerberos Environment

From a Riverbed perspective, the optimization of secure protocols in a Microsoft Windows environment includes the following:

- Signed SMB
- Signed SMB2
- Signed SMB3
- Encrypted MAPI
- Encrypted Outlook-Anywhere
- SharePoint

You must understand how RiOS performs optimization on these protocols when the authentication is end-to-end Kerberos before you proceed to more specific details. Remember the following points:

- Secure protocols use a session key to encrypt or digitally sign protocol packets. This session key is exchanged during the authentication phase between the client and server.
- In the case of Kerberos, the session key is extracted from the service ticket by the target server.
- Decrypting the service ticket requires the server machine account password, which is set up during the domain join process and might be periodically refreshed.
- The servers machine account password is only known to the target server and domain controller.
- Active Directory supports account replication between domain controllers for high availability.

Therefore, the only way for RiOS to legitimately optimize these encrypted, or digitally signed, secure protocols is to obtain the machine credentials for any servers that require optimization. RiOS uses Active Directory replication by using the same protocols and tools as the Windows domain controllers do, and by employing a domain user that has replication privileges.

When the server-side Steelhead appliance uses this replication capability, it can transparently participate in the key exchange between the client and server, maintain security, and optimize relevant traffic.

Domain User with Replication Privileges

Although a user with generic domain administrator privileges meets the requirements for replication, this could have some security implications.

Instead, you can assign a restricted set of privileges to a user, known as a *replication user*. You can configure the replication user on a per forest basis so that the user assigned to it can retrieve machine credentials from any domain controller in any trusted domain within the forest. Remember that a forest can comprise of multiple domains with trusts between them.

The replication user begins with a generic user account created by a Windows domain administrator, which might or might not be you. After this is done, you can apply the privileges and restrictions for the account to become a replication user on the Steelhead appliance. If you have multiple Steelhead appliances in the data center (for example, in a high-availability configuration), then you can add the same replication user account to the configuration for all the server-side Steelhead appliances. For details on how to configure the replication user, see the *Riverbed Command-Line Interface Reference Manual* and the *Steelhead Appliance Management Console User's Guide*.

After the configuration is complete, the Steelhead appliance retains the replication user account details by encrypting them in its secure vault.

Note: In RiOS v7.0 or later, the secure vault is locked on a new Steelhead appliance. The secure vault is also locked on a Steelhead appliance that is upgraded to RiOS v7.0 or later. You must unlock the secure vault to view, add, remove, or edit any replication or delegate user configuration details that are stored on the Steelhead appliances.

In Windows 2008 domains, you can apply further restrictions to the replication user by using a password replication policy (PRP). A PRP is a set of rules that describe a list of accounts that you can replicate between domain controllers. When you enable PRP in a Windows domain, the Steelhead appliance is restricted, specifically to using the replication user to replicate accounts as determined by the PRP settings. Therefore, you can use the PRP to limit the accounts known to the replication user. This configuration can create additional administrative overhead in managing the PRP; however it might be worth the additional security.

PRP is not available in Windows domains with a functional level lower than Windows 2008. For details on how to configure PRP, see the *Steelhead Appliance Management Console User's Guide*.

Configuring Traffic Optimization for HTTP (SharePoint), Encrypted MAPI, and Signed SMB/SMB2/SMB3

In RiOS v7.0 or later, you can enable Kerberos authentication support for HTTP for SharePoint, encrypted MAPI, signed SMB, and signed SMB2. To fully optimize signed SMB3 traffic, you must use RiOS v8.5 or later. When you enable the Kerberos setting on the server-side Steelhead appliance, you enable the Steelhead appliance to use the relevant domain user privileges and to access to the session key used by the client and server.

You must have RiOS v7.0 or later on both the server-side and client-side Steelhead appliances. You must use RiOS v8.5 or later for SMB3.

For details on how to configure each application, see the *Steelhead Appliance Management Console User's Guide* and "HTTP Optimization for SharePoint" on page 83.

Configuring the Server-Side Steelhead Appliance for Active Directory Integrated (Windows 2003/2008)

Important: In some cases this can be an alternative to creating and using a specific replication user (described in "Domain User with Replication Privileges" on page 49). However, it depends on the authentication mechanism used by the Windows clients and servers.

You do not necessarily need to use the replication user or delegate user facility to optimize secure Windows traffic if you deploy the server-side Steelhead appliance so that it joins a domain in the Active Directory environment. For the server-side Steelhead appliance to integrate into Active Directory, you must configure the role when the appliance joins the Windows domain. Depending on whether you have a Windows domain with a 2003 functional level or 2008 functional level, select the appropriate role from the Management Console or the command line of the server-side Steelhead appliance.

For details on how to configure Active Directory integration, see the *Riverbed Command-Line Interface Reference Manual* and the *Steelhead Appliance Management Console User's Guide*.

Be aware, that when you integrate the server-side Steelhead appliance in this way, it does not provide any Windows domain controller functionality to any other machines in the domain and does not advertise itself as a domain controller or register any SRV records (service records). In addition, the Steelhead appliance does not perform any replication nor hold any Active Directory objects. The server-side Steelhead appliance has just enough privileges so that it can have a legitimate conversation with the domain controller and then use transparent mode for NTLM authentication.

This scenario is successful only for servers and clients that can make use of NTLM authentication. If you configure any clients and servers exclusively to use Kerberos authentication, they cannot use NTLM authentication. Therefore, the only way to optimize secure Windows traffic between such hosts is to configure the server-side Steelhead appliance for end-to-end Kerberos support with a replication user account.

For details on replication users, see "Domain User with Replication Privileges" on page 49.

The following table shows the different combinations of Windows clients and authentication methods with the required minimum version of RiOS and Windows configuration (delegation, Kerberos, Active Directory integrated) for the server-side Steelhead appliance.

Client OS	Authentication Method	RiOS v6.0/6.5 (Delegation)	RiOS v7.0 (Kerberos)	RiOS v7.0 (Active Directory Integrated)
XP/Vista	Password authentication/NTLM	Optimized	Optimized	Optimized
Windows 7	Password authentication/NTLM	Optimized in delegation mode	Optimized in delegation mode	Optimized
XP/Vista	Negotiate authentication/Simple And Protected Negotiate (SPNEGO)	Optimized using NTLM	Optimized using Kerberos	Optimized using NTLM
Windows 7	Negotiate authentication/ SPNEGO	Optimized using NTLM in delegation mode	Optimized using Kerberos	Optimized using NTLM
Any client	Kerberos	Passthrough	Optimized	Passthrough

Remember, if you configure the server-side Steelhead appliance to support end-to-end Kerberos authentication, you can also join it to the domain in Active Directory integrated mode to support other clients that might be using NTLM authentication. This configuration can provide flexible and broad support for multiple combinations of Windows authentication types in use within the Active Directory environment.

For advice and suggestions on configuration best practices, see "Best Practices for the Steelhead Appliance in a Secure Windows Deployment" on page 51.

Best Practices for the Steelhead Appliance in a Secure Windows Deployment

There are many possible ways to configure the server-side Steelhead appliance to support the different secure Windows application options. The following is a list of best practices to ensure that the majority, if not all, of the secure Windows traffic in your environment is fully optimized:

- Make sure that the server-side Steelhead appliance has a DNS entry. You need only an *A record*.
- Make sure that the server-side Steelhead appliance time-of-day is synchronized through NTP. Riverbed recommends that you synchronize to the same NTP service as the domain controllers.
- Join the server-side Steelhead appliance to the Windows domain of choice using the role of Active Directory integrated Windows 2008. If possible, use the domain of the user domain or the same domain as the majority of the Windows application servers on which you want optimized traffic.
- When joining the domain, specify one or more domain controllers within the same domain that are geographically closest to the server-side Steelhead appliance.
- Enable native Kerberos authentication support for the relevant Layer-7 RiOS features (for example, signed CIFS, signed SMB2, signed SMB3, encrypted MAPI, and HTTP) on the server-side Steelhead appliance.

  Note: For encrypted MAPI and signed SMB3, these settings are also required on the client-side Steelhead appliance.

- Configure a replication user within the Active Directory forest and enter the account details into the server-side Steelhead appliance.
- Configure a PRP within Active Directory to further restrict the replication user account (optional).
- If the server-side Steelhead appliance interacts with other domains through a one-way trust, use the CLI or the Management Console to enable this setting.

  Note: If there are no one-way trusts, then this step is not required.

- Depending on the RiOS version you have on the server-side Steelhead appliance, you can perform the Windows Active Directory configuration steps (domain join, user account creation, and so on) by using the domain authentication automatic configuration feature in the Management Console or through the CLI.

  For information about the domain health and domain authentication automatic configuration features, see "Domain Health Check and Domain Authentication Automatic Configuration" on page 53.

Domain Authentication Scaling

This section describes the ability of the Steelhead appliances to scale the authentication communications that occur between the server-side Steelhead appliance and the Windows domain controller. This section includes the following topics:

- "When to Use Domain Authentication Scaling" on page 52
- "General Improvements in RiOS v8.6" on page 52
- "Domain Controller Load Balancing" on page 53

When to Use Domain Authentication Scaling

In releases previous to RiOS v8.6, the server-side Steelhead appliance communicates only with a single domain controller at a time. Even if the domain that the Steelhead appliance is part of contains additional domain controllers, the Steelhead appliance can choose to communicate only with one. If the domain controller fails or is unavailable for some reason, then the Steelhead appliance automatically discovers an alternative domain controller.

While this process works well in most deployments, the following are examples of environments in which an individual Steelhead appliance can be limited by this single-instance approach and can benefit from domain authentication scaling. Domain authentication scaling is available in RiOS v8.6 and later.

A large-scale deployment (with tens of thousands of users) typically involves a larger number of authentication requests between clients and servers. Because the server-side Steelhead appliance is involved in so many requests when optimizing secure Windows traffic, lightening its communication to an individual domain controller is important.

In another scenario, there might not be a large number of requests but there might be a high degree of latency between the server-side Steelhead appliance and the single domain controller it is communicating with. For example, with a deployment to Microsoft Office 365, there might be *non-LAN* latency between the domain controller and the server-side Steelhead appliance.

In general, Riverbed recommends deployment in which the latency between domain controllers and the server-side Steelhead appliance is no more than a few milliseconds.

For more information about maximum latency between a Steelhead appliance and a domain controller, see http://supportkb.riverbed.com/support/index?page=content&id=S22110.

General Improvements in RiOS v8.6

With RiOS v8.6 or later, the server-side Steelhead appliance can manage higher quantities of communication with the Windows Active Directory (AD) environment it is requesting authentication details from. This has no direct benefit to the optimization of the secure Windows traffic (signed SMB, encrypted MAPI, and so on), but it can improve the authentication workload that the server-side Steelhead appliance is part of. These general improvements include multithreading the RiOS code associated with AD interaction, which enables the server-side Steelhead appliance to respond to more requests in a shorter time. Because these improvements are embedded within the RiOS code, you do not need to configure any additional settings other than to have the server-side Steelhead appliance running RiOS v8.6 or later.

Domain Controller Load Balancing

Windows domains often contain more than one domain controller for redundancy and scalability. With RiOS v8.6 or later you can configure the server-side Steelhead appliance to communicate simultaneously with multiple domain controllers in the same domain. This enables the server-side Steelhead appliance to:

- sustain higher quantities of authentication requests.
- tolerate the loss of a domain controller much more efficiently.
- reduce the load it places on an individual domain controller.

To enable the server-side Steelhead appliance to load balance domain controllers, enter this command on the server-side Steelhead appliance:

```
protocol domain-auth configure load-balancing
```

You can configure this setting only on the CLI. For more details on this command, see *Riverbed Command-Line Interface Reference Manual*.

When you enable domain controller load balancing, the server-side Steelhead appliance balances the traffic load across up to four domain controllers within the same domain. You can configure load balancing across more domain controllers by specifying the desired number as part of the command. The following example shows how to configure the Steelhead appliance to load balance traffic across four domain controllers:

```
protocol domain-auth configure load-balancing 4
```

You can balance traffic across a maximum of eight domain controllers. Use the s**how protocol domain-auth load-balancing** command to display the current status.

You do not need to specify the individual domain controllers to which the Steelhead appliance load balances traffic. After the Steelhead appliance is joined to a domain it automatically discovers any domain controllers that exist by performing a DNS lookup. The results from the DNS lookup enable the server-side Steelhead appliance to automatically build its own list of domain controllers to balance traffic across. However, if you have already configured the Steelhead appliance with a static list of domain controllers, then the static list takes precedence over an automatically discovered list. You can create a static list either by specifying one or more domain controllers on the server-side Steelhead appliance during the join domain procedure or by using the **domain settings dc-list** * command.

Domain Health Check and Domain Authentication Automatic Configuration

This section describes domain health check and domain authentication automatic configuration and includes the following topics:

- "Domain Health Check" on page 54
- "Domain Authentication Automatic Configuration" on page 58

To optimize secure Windows traffic using Steelhead appliances, you:

- need to configure NTP- and DNS-related settings.
- must join the server-side Steelhead appliance to the Windows domain.
- must configure the necessary RiOS features based on the type of protocols you want optimize: for example, Encrypted MAPI/ signed SMB, signed SMB2, and signed SMB3.

- must deploy one or more service accounts configured for delegation (if using constrained delegation) or replication (if using end-to-end Kerberos).

The domain health check comprises of a series of tests related to Active Directory configuration settings. These tests and checks provide troubleshooting help for domain-related problems that might occur between the server-side Steelhead appliance and the Active Directory environment.

The domain authentication automatic configuration includes a series of graphical widgets to assist you in performing the relevant configuration tasks associated with both the Steelhead appliance and the Active Directory setup.

Domain Health Check

In RiOS v8.5 or later, the domain health check feature is available in the Steelhead Management Console and the CLI. Prior to RiOS v8.5, domain health check was only available in the CLI. You can use the domain health check feature to execute a variety of tests that provide diagnostic reports about the status of domain membership, end-to-end Kerberos replication, both manual and automatic constrained delegation, and DNS resolution. This information enables you to resolve issues quickly.

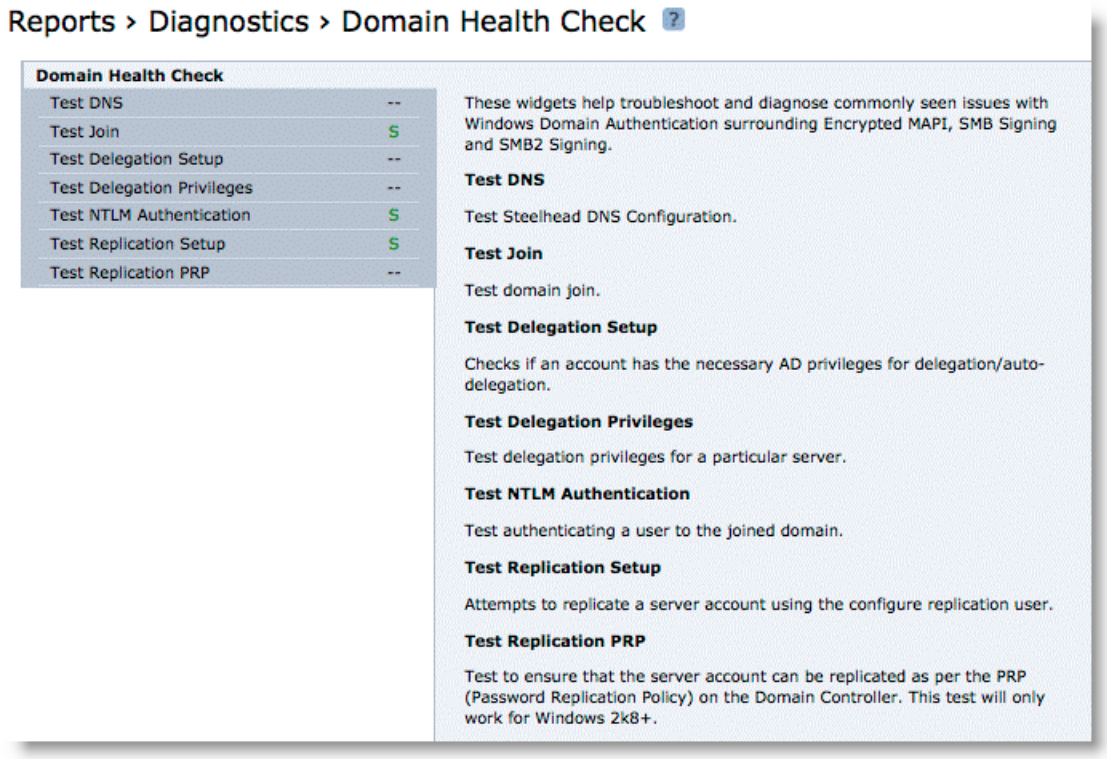
RiOS Riverbed recommends that you use domain health check from the Steelhead Management Console rather than the CLI.

For a full description of how to use domain health check, see the *Steelhead Appliance Management Console User's Guide* and the *Riverbed Command-Line Interface Reference Manual*.

Using the Steelhead Management Console to Test Domain Health Check

Use the Domain Health Check page to run tests on domain health. You can create test parameters by entering specific information into certain fields. Click **Test** to run the relevant test. You receive feedback on whether the test succeeds or fails, along with the option to display a detailed log file of the test as it progresses. The output of the log file can aid in troubleshooting issues that might be found during testing.

Figure 3-6. Domain Health Check Page



You can access the same tests by choosing Configure > Networking > Windows Domain and Configure > Optimization > Windows Domain Auth pages of the Steelhead Management Console.

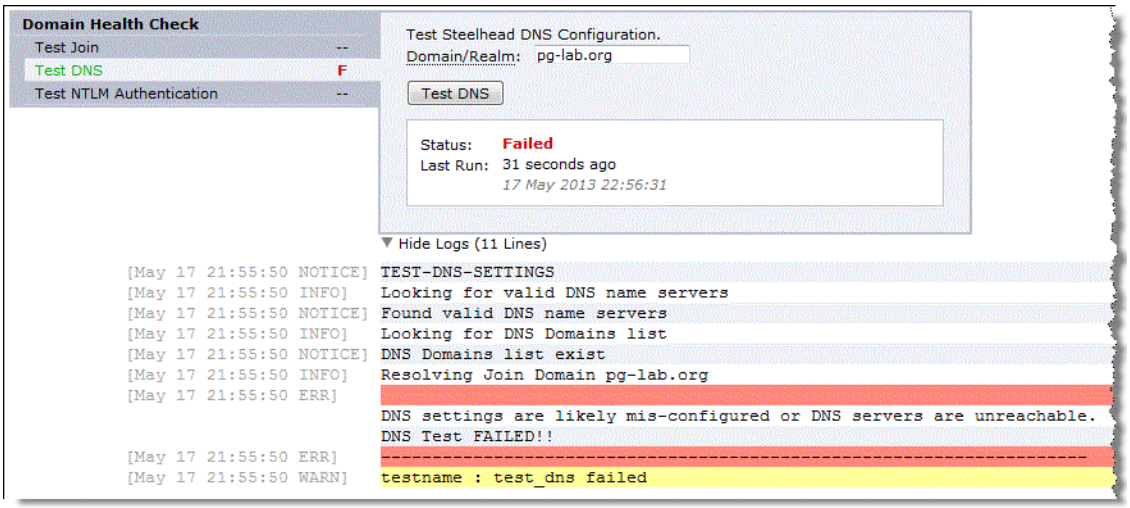
The following examples describe several ways to confirm that the domain health check feature is functioning correctly.

To test DNS setting using the Steelhead Management Console

- From the Management Console, select either Reports > Diagnostics > Domain Health Check or Configure > Networking > Windows Domain.

Figure 3-7 shows the check for Test DNS. In this particular example, the test has failed. You can choose to display or hide the logs for the test.

Figure 3-7. Failed DNS Test in the Management Console

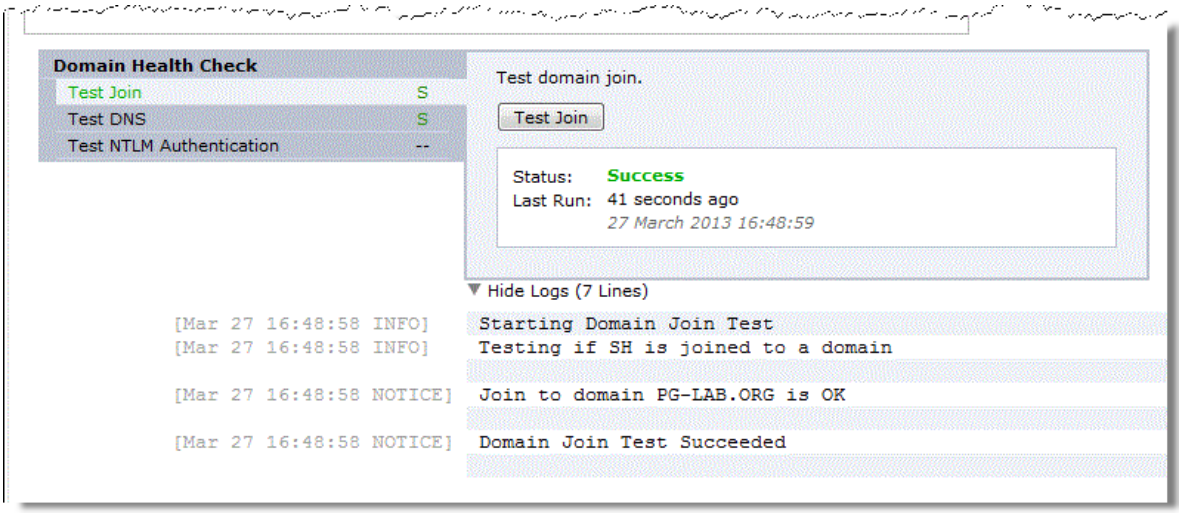


To test Domain Join using the Steelhead Management Console

- From the Management Console, select either Reports > Diagnostics > Domain Health Check or Configure > Networking > Windows Domain.

  Figure 3-8 shows the check for Test Join. In this particular example, the test has passed. You can choose to display or hide the logs for the test.

Figure 3-8. Successful Test Join in the Management Console



Using the RiOS CLI Commands to Test Domain Health Check

To use the RiOS domain health check CLI commands, you must understand that each command performs a test or configuration task, but the result of the command is displayed only by executing a follow-on command. This second command is displayed as part of the output of the previous command and is usually a **show** command.

For example, the command **protocol domain-auth test dns** is followed by **show protocol domain-auth test dns** to display the results of the test.

The main reason for this two-stage process is that the tests themselves perform a request or look-up that is outside of the Steelhead appliance: for example, a DNS query to a DNS server can take a few moments to complete. The two-stage process means the Steelhead appliance CLI does not hang while waiting for the test to execute. As each test is executed, the results are saved to a temporary log file on the Steelhead appliance. After a test is complete, the contents of the log file are displayed in a more user-friendly format when you use the relevant show command.

The following table lists the test or configuration tasks and the associated commands.

Task	CLI Commands
Check DNS settings.	`protocol domain-auth test dns` `show protocol domain-auth test dns`
Confirm that the Steelhead appliance is correctly joined to the Windows domain.	`protocol domain-auth test join` `show protocol domain-auth test join`
Ensure that the Steelhead appliance can authenticate client connections.	`protocol domain-auth test authentication username * password *` `show protocol domain-auth test authentication`
Check to see if constrained delegation is correctly set up.	`protocol domain-auth test delegation setup domain * dc *` `show protocol domain-auth test delegation setup`
Complete the configuration settings for a preexisting Windows user account that is selected to become the delegation or replication account.	`protocol domain-auth auto-conf delegation setup-user adminuser * adminpass * domain * dc *` `show protocol domain-auth auto-conf delegation`

Add or remove servers on the delegation user's msDS-AllowedToDelegateTo list.	`protocol domain-auth auto-conf delegation add-server adminuser * adminpass * domain * dc * service * serverlist *`
	`protocol domain-auth auto-conf delegation del-server adminuser * adminpass * domain * dc * service * serverlist *`
	`protocol domain-auth auto-conf delegation add-server domain * dc * service * serverlist *`
	`protocol domain-auth auto-conf delegation del-server domain * dc * service * serverlist *`
	`show protocol domain-auth auto-conf delegation add-server`
	`show protocol domain-auth auto-conf delegation del-server`

Auto-configure a previously created account in Active Directory with replication privileges over the entire domain.	`protocol domain-auth auto-conf replication adminuser * adminpass * domain * dc *` `show protocol domain-auth auto-conf replication`
Determine if end-to-end Kerberos replication is correctly configured.	`protocol domain-auth test replication try-repl domain * shortdom * rserver *` `show protocol domain-auth test replication try-repl`

The following example describes how to confirm that the domain health check feature is functioning correctly.

To check DNS settings using the CLI

- Connect to the Steelhead CLI in and enter the following commands:

```
protocol domain-auth test dns
show protocol domain-auth test dns
```

Figure 3-9 shows a successful DNS test and Figure 3-10 shows a failed DNS test.

Figure 3-9. Successful DNS Test in the CLI

```
bravo-sh93 # protocol domain-auth test dns
Test DNS Status : STARTED.
Check result using : 'show protocol domain-auth test dns'
bravo-sh93 # show protocol domain-auth test dns

Action                          STATUS                    LAST RUN
------------------------------------------------------------------------------

Test DNS                        SUCCESS              Tue Aug  9 00:17:04 2011

------------------------------------------------------------------------------


RESULT : Testing DNS Configuration
Joined Domain : VCS246.GEN-VCS78DOM.COM


DNS Test Passed
```

Figure 3-10. Failed DNS Test in the CLI

```
bravo-sh93 # show protocol domain-auth test dns

Action                          STATUS                    LAST RUN
------------------------------------------------------------------------------

Test DNS                        FAILED               Tue Aug  9 00:14:37 2011

------------------------------------------------------------------------------


RESULT : Testing DNS Configuration
Joined Domain : VCS246.GEN-VCS78DOM.COM

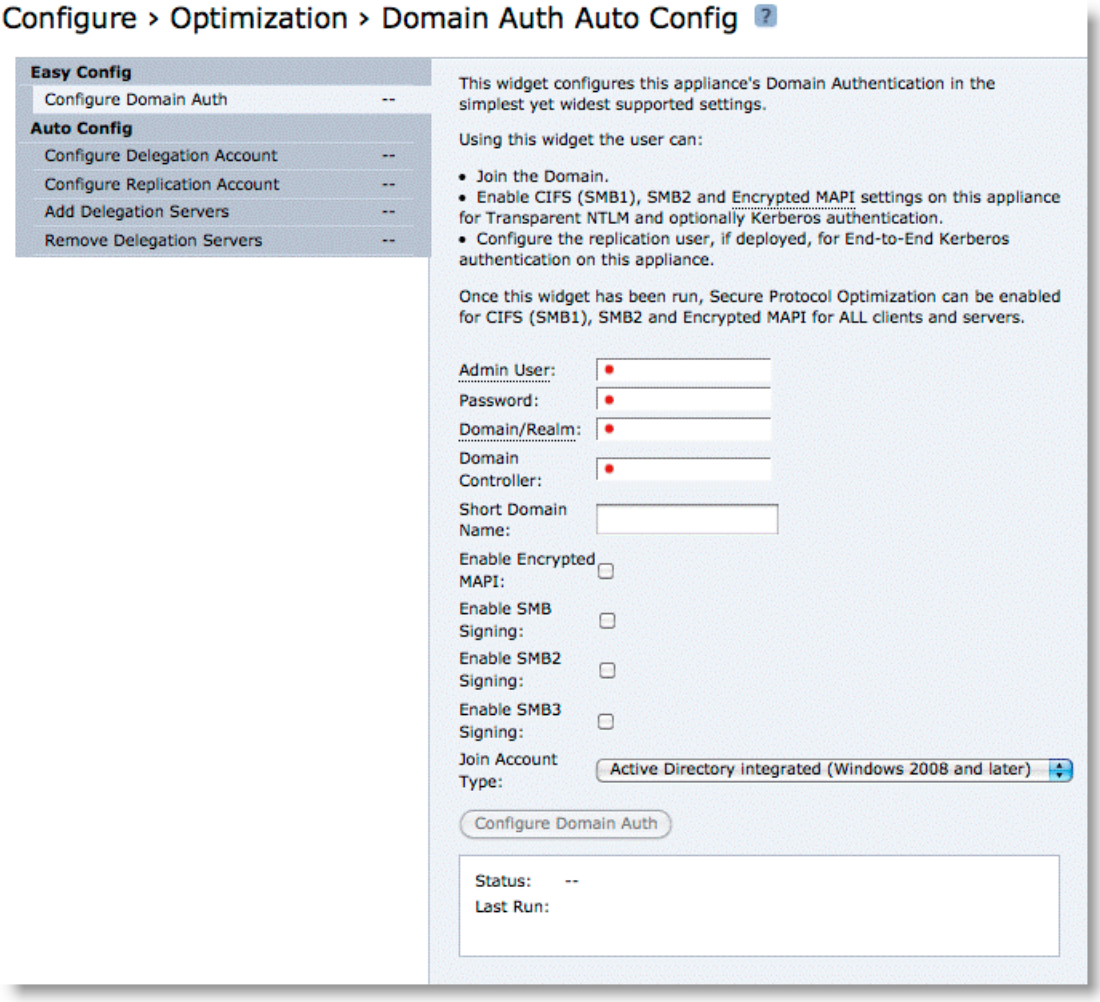
DNS Settings are broken - Looking up VCS246.GEN-VCS78DOM.COM failed with error : Host name lookup failure
DNS settings are likely mis-configured or DNS servers are unreachable.
```

Domain Authentication Automatic Configuration

Domain authentication automatic configuration is available in the Steelhead Management Console in RiOS v8.5 or later. Domain authentication automatic configuration is a powerful set of widgets designed to help you easily configure the server-side Steelhead appliance and Active Directory. In RiOS versions prior to v8.5, you can configure the server-side Steelhead appliance and Active Directory using only the CLI.

For example, Figure 3-11 shows how, in RiOS v8.5 or later, the domain authentication automatic configuration guides you through the steps to join the Steelhead appliance to the domain and enable the relevant Windows features (Encrypted MAPI, signed SMB, signed SMB2, and signed SMB3).

Figure 3-11. Domain Authentication Automatic Configuration



You can use the domain authentication automatic configuration to configure a Windows user account that you can use for delegation or replication purposes. The domain authentication automatic configuration on the Steelhead appliance does not create the delegate or replication user; the Windows domain administrator must create the account in advance, using the preferred standard Active Directory procedures.

After you create the basic user account in Active Directory, you can complete the remaining configuration steps using domain authentication automatic configuration on the Steelhead appliance.

Along with configuring the delegation and replication user accounts, domain authentication automatic configuration enables you to add and remove entries in the lists of delegation servers.

For details on how to use domain authentication automatic configuration, see the *Steelhead Appliance Management Console User's Guide*.

Example Configurations

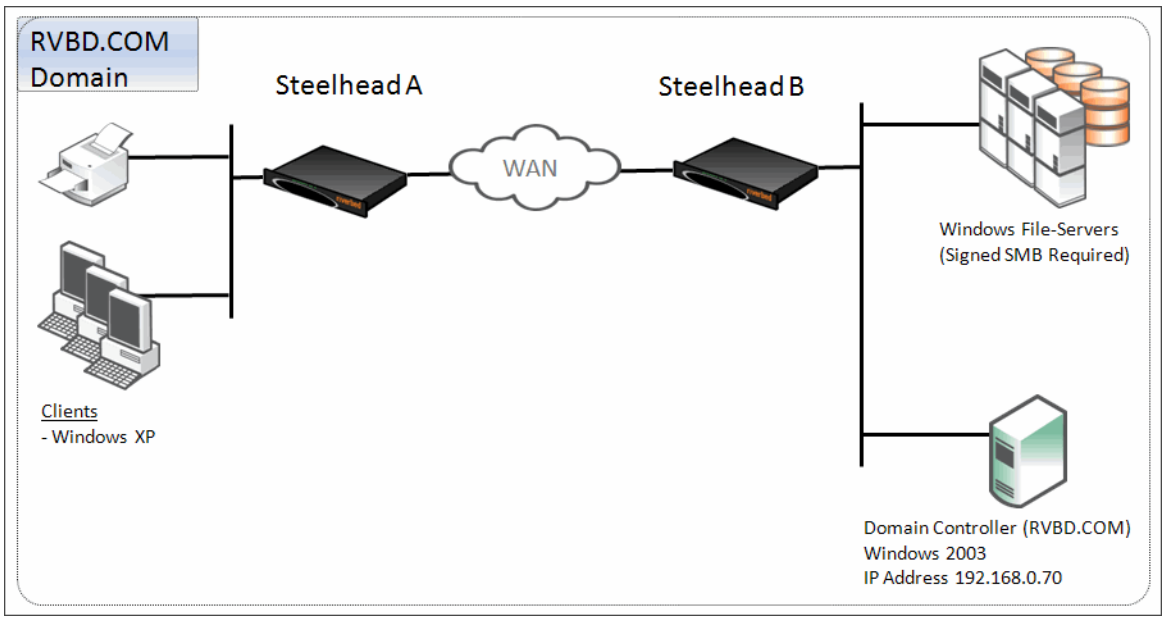
This section shows the following configuration examples:

- "Single Domain" on page 60
- "Multiple Domains with Windows 7 Clients" on page 61

Single Domain

Figure 3-12 shows a data center and a branch office that are both in the RVBD.COM domain hosted by a domain controller running Windows 2003. The Windows XP clients in the remote office want optimized CIFS access to the file servers in the Data Center. The file servers are configured with *Signing Required*. For ease of management, Transparent mode is preferred by the customer.

Figure 3-12. Single Domain Example



The following is true for this configuration:

- Riverbed recommends that all the Steelhead appliances are time synchronized for Steelhead appliance deployments that involve some form of Windows authentication. This is especially true where Kerberos is involved in the authentication process. Consider using NTP to make the time synchronization task easier to maintain.
- Steelhead A on the client side must have a minimum of RiOS v5.5.x running to support signed SMB, but no further client-side configuration is required.
- Steelhead B on the server side needs to join the RVBD.COM domain.
- Steelhead B needs to have a minimum version of RiOS v6.0.x and Transparent mode configured, or a minimum version of RiOS v5.5.x but Delegation mode must be configured.
- Secure inner channel is optional for this configuration.

To configure a data center and branch office in the same domain

1. Steelhead A does not need configuration.
2. On Steelhead B, connect to the CLI in configuration mode and enter the following commands:

```
domain join domain-name RVBD.COM login Administrator password ******** dc-list 192.168.0.70
short-name RVBD
protocol cifs smbv1-mode enable
protocol cifs smb signing enable
protocol cifs smb signing mode-type transparent
write memory
restart
```

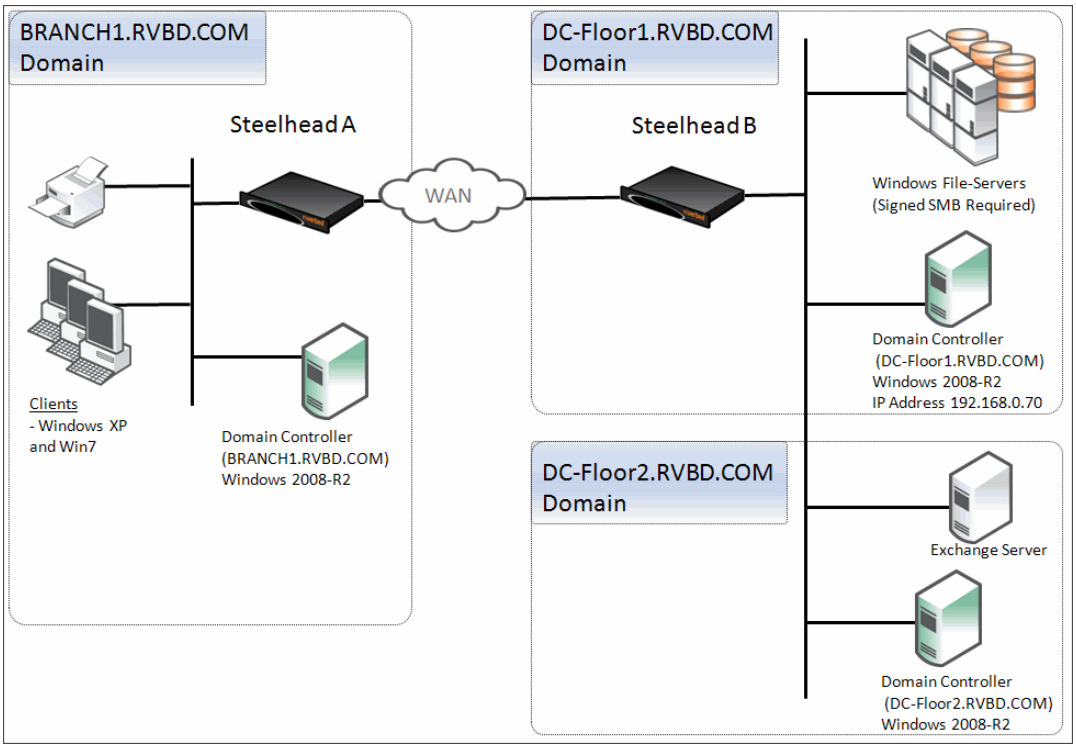
Multiple Domains with Windows 7 Clients

Figure 3-13 shows all offices are in the RVBD.COM domain but several child domains exist. The data center is on two floors with a domain for each: DC-Floor1.RVBD.COM and DC-Floor2.RVBD.COM. The remote office is in the BRANCH1.RVBD.COM domain. DC-Floor1 has file servers with signed SMB required, and DC-Floor2 has Microsoft Exchange servers with encryption enabled.

All domain controllers are running MS Windows 2008-R2 and there are two-way trusts between all domains.

The branch office clients are a mixture of both Windows XP and Windows 7, and the customer wants optimized access to all servers in the data center, including file servers and the Exchange server.

Figure 3-13. Multiple Domain with Windows 7 Clients Example



The following is true for this configuration:

- Steelhead A on the client side needs to have a minimum of RiOS v5.5.x running, and have optimization for encrypted MAPI enabled. Steelhead A does not need any further configuration to support signed SMB traffic.
- Steelhead B on the server side needs to join the DC-Floor1.RVBD.COM domain. It could join the DC-Floor2.RVBD.COM domain, but only one of the two.
- Steelhead B requires a minimum version of RiOS v6.1 and Delegation mode configured because it has to support the Windows 7 clients, and the Windows 2008-R2 domains, and that the Exchange server and clients are in different domains to Steelhead B.

Note: Either Manual or Auto-Delegation mode can be configured. A delegate user is required in each of the domains where a server is going to be optimized. In the following example, Auto-Delegation mode is used and the delegate user names for the domains (DC-Floor1 and DC-Floor2) are delegate_rvbd_cifs and delegate_rvbd_mapi respectively.

To configure multiple domains and windows 7 clients

1. On Steelhead A, connect to the CLI in configuration mode and enter the following commands:

```
protocol cifs smbv1-mode enable
protocol mapi 2k7 native enable
protocol mapi encrypted enable
protocol mapi encrypted delegation enable
write memory
restart
```

2. On Steelhead B, connect to the CLI in configuration mode and enter the following commands:

```
domain join domain-name DC-Floor1.RVBD.COM login Administrator password ******** dc-list
192.168.0.70 short-name DC-Floor1
protocol domain-auth delegation auto-mode enable
protocol domain-auth delegate-user domain DC-Floor1.RVBD.COM user delegate_rvbd_cifs password
******
protocol domain-auth delegate-user domain DC-Floor2.RVBD.COM user delegate_rvbd_mapi password
******
protocol cifs smbv1-mode enable
protocol cifs smb signing enable
protocol cifs smb signing mode-type delegation
protocol mapi 2k7 native enable
protocol mapi encrypted enable
protocol mapi encrypted delegation enable
write memory
restart
```

Confirm the correct configuration settings display in the Current Connections report, shown in Figure 3-14 and Figure 3-15. For details on the Current Connection Report, see the *Steelhead Appliance Management Console User's Guide*.

Figure 3-14. MAPI-ENCRYPT

CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
»		10.37.80.131:61366	10.16.205.53:7830	15	12	22%	2013/05/20 09:38:50	MAPI-ENCRYPT
»		10.37.80.132:56841	10.16.205.132:7830	4	4	6%	2013/05/20 09:38:35	MAPI-ENCRYPT
»		10.37.80.132:56819	10.16.205.53:7830	4,801	977	79%	2013/05/20 09:38:32	MAPI-ENCRYPT

Figure 3-15. CIFS-SIGNED

CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
»		192.168.122.202:49205	192.168.121.201:445	15	16	79%	2013/05/23 14:42:30	CIFS-SIGNED

CHAPTER 4 HTTP Optimization

This chapter includes examples of techniques used by the Hypertext Transfer Protocol (HTTP) optimization module on the Steelhead appliance to improve optimization. This chapter includes the following sections:

- "HTTP and Browser Behavior" on page 64
- "RiOS HTTP Optimization Techniques" on page 73
- "HTTP Authentication Optimization" on page 76
- "HTTP Settings for Common Applications" on page 82
- "HTTP Optimization for SharePoint" on page 83
- "HTTP Optimization Module and Internet-Bound Traffic" on page 84
- "Tuning Microsoft IIS Server" on page 85
- "Info-Level Logging" on page 90
- "Use Case" on page 90

The HTTP protocol has become the accepted standard mechanism for transferring documents on the Internet. The original version, HTTP v1.0, is documented in RFC 1945 but has since been superseded by HTTP v1.1 and documented in RFC 2068. TCP protocol is the most common underlying transport protocol for HTTP.

By default, a Web server listens for HTTP traffic on TCP port 80, although you can reconfigure the Web server to listen on a different port number. HTTP uses a very simple client/server model—after the client has established the TCP connection with the server, it sends a request for information and the server replies back. A Web server can support many different types of requests but the two most common types of requests are as follows:

- **GET request** - asks for information from the server.
- **POST request** - submits information to the server.

A typical Web page is not one file that downloads all at once. Web pages are composed of dozens of separate objects including JPG and GIF images, JavaScript code, cascading style sheets, and more—each of which is requested and retrieved separately, one after the other. Given the presence of latency, this behavior is highly detrimental to the performance of Web-based applications over the WAN. The higher the latency, the longer it takes to fetch each individual object and, ultimately, to display the entire page. Furthermore, the server might be protected and require authentication before delivering the objects. The authentication can be once per connection or it can be once per request.

The HTTP optimization module addresses these challenges by using several techniques. For example:

- The HTTP optimization module learns about the objects within a Web page and pre-fetch those objects in bulk before the client requests them. When the client requests those objects, the local Steelhead appliance serves them out locally without creating extra round trips across the WAN.
- The HTTP optimization module learns the authentication scheme that is configured on the server. It can inform the client that it needs to authenticate against the server without the client incurring an extra round trip to discover the authentication scheme on the server.

HTTP and Browser Behavior

The HTTP protocol uses a simple client/server model where the client transmits a message (also referred to as *a request*), to the server and the server replies with a response. Apart from the actual request, which is to download the information from the server, the request can contain additional embedded information for the server. For example, the request might contain a *Referer* header. The Referer header indicates the URL of the page that contains the hyperlink to the currently requested object. Having this information is very useful for the Web server to trace where the requests originated from, for example:

```
GET /wiki/List_of_HTTP_headers HTTP/1.1
Host: en.wikipedia.org
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.7)
Gecko/20100713 Firefox/3.6.7
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

SOURCE: http://en.wikipedia.org/wiki/HTTP_referrer

The HTTP protocol is a stateless protocol. A stateless protocol means that the HTTP protocol does not have a mechanism to store the state of the browser at any given moment in time. Being able to store the state of the browser is useful particularly in situations where a user who is online shopping, has added items to the shopping cart, but was then interrupted and closed the browser. With a stateless protocol, when the user re-visits the Web site again, the user has to add all the items back into the shopping cart. To address this issue, the server can issue a *cookie* to the browser. There are typically the following types of cookies:

- **Persistent cookie** - remains on the user's PC after the user closes the browser.
- **Non-persistent cookie** - discarded after the user closes the browser.

In the example earlier, assuming that the server issues a persistent cookie, when the user closes the browser and re-visits the Web site again, all the items from the previous visit are populated in the shopping cart. Cookies are also used to store user preferences. Each time the user visits the Web site, the Web site is personalized to the user's liking.

This example shows where to find the cookie in the client's GET request.

```
GET /cx/scripts/externalJavaScripts.js HTTP/1.1
Host: www.acme.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.6)
Gecko/20100625 Firefox/3.6.6
Accept: */*
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

```
Referer: http://www.acme.com/cpa/en_INTL/homepage
Cookie: TRACKER_CX=64.211.145.174.103141277175924680;
CoreID6=94603422552512771759299&ci=90291278,90298215,90295525,90296369,902941
55; _csoot=1277222719765; _csuid=4c20df4039584fb9; country_lang_cookie=SG-en
```

Web browser performance has improved considerably over the years. In the HTTP v1.0 specification, the client opened a connection to the server, downloaded the object, and then closed the connection. Although this was easy to understand and implement, it was very inefficient because each download triggered a separate TCP connection setup (TCP 3-way handshake). In the HTTP v1.1 specification, the browser has the ability to use the same connection to download multiple objects.

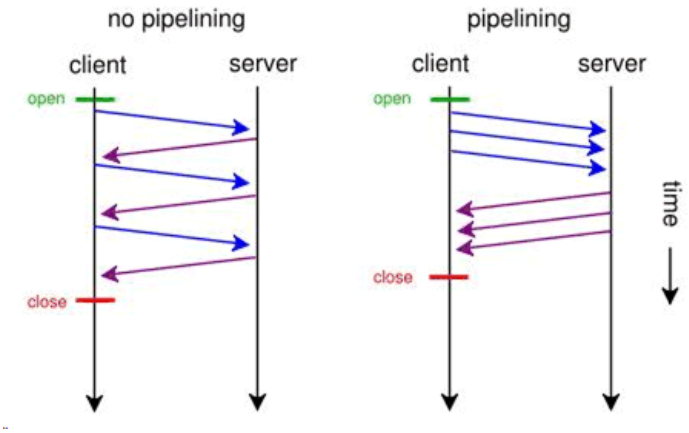
The next section discusses how the browser uses multiple connections, and a technique known as *pipelining*, to improve the browser's performance.

Multiple TCP Connections and Pipelining

Most modern browsers establish two or more TCP connections to the server for parallel downloads. The concept is simple—as the browser parses the Web page, it knows what objects it needs to download. Instead of sending the requests serially over a single connection, the requests are sent over parallel connections resulting in a faster download of the Web page.

Another technique used by browsers to improve the performance is *pipelining*. Without pipelining, the client first sends a request to the server and waits for a reply before sending the next request. If there are two parallel connections, then a maximum of two requests are sent to the server concurrently. With pipelining, the browser sends the requests in batches instead of waiting for the server to respond to each individual object before sending the next request. Pipelining is used with a single HTTP connection or with multiple HTTP connections. Although most servers support multiple HTTP connections, some servers do not support pipelining. Figure 4-1 illustrates the effect of pipelining.

Figure 4-1. Pipelining



Source: http://egonitron.com/2007/05/25/the-truth-about-the-firefox-pipelining-trick/ (Aug. 12, 2010)

HTTP Authentication

To provide access control to the contents on the Web server, authentication is enabled on the server. There are many ways to perform authentication—for example, certificates and smartcards. The most common schemes are NT LAN Manager (NTLM) and Kerberos. Although the two authentication schemes are not specific to HTTP and are used by other protocols such as CIFS and MAPI, there are certain aspects that are specific to HTTP.

Note: For more details on Kerberos, see "CIFS Optimization" on page 7, "Microsoft Exchange Email Optimization" on page 15, and "Kerberos" on page 45.

The following steps describe four-way NTLM authentication shown in Figure 4-2:

1. The client sends a GET request to the server.

2. The server, configured with NTLM authentication, sends back a *401 Unauthorized* message to inform the client that it needs to authenticate. Embedded in the response is the authentication scheme supported by the server. This is indicated by the WWW-Authenticate line.

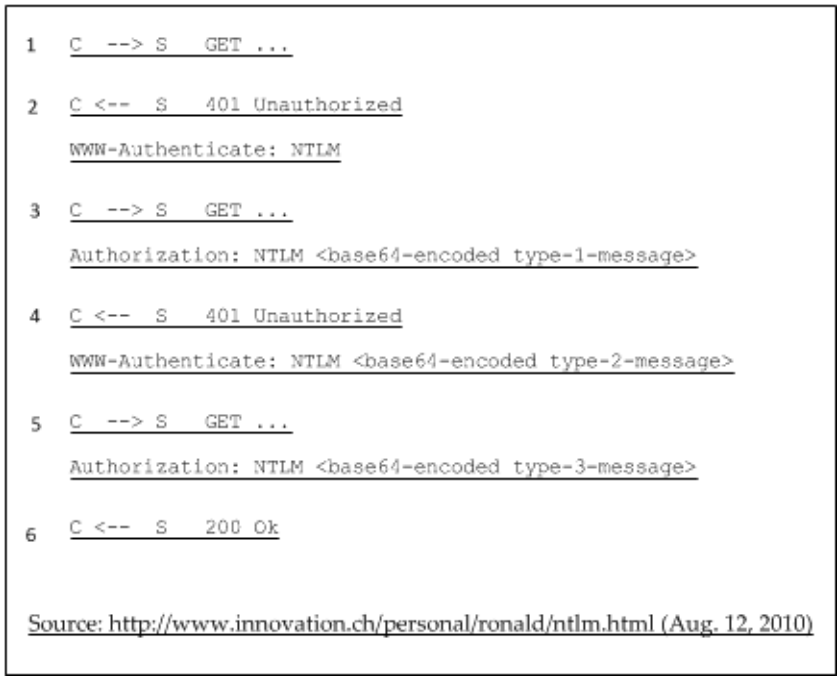
3. The client sends another GET request, and attaches an NTLM Type-1 message to the request. The NTLM Type-1 message provides the set the capability flags of the client (for example, encryption key size).

4. The server responds with another 401 Unauthorized message, but this time it includes an NTLM Type-2 message in the response. The NTLM Type-2 message contains the server's NTLM challenge.

5. The client computes the response to the challenge and once again attaches this to another GET request. Assuming the server accepts the response, the server delivers the object to the client.

6. Assuming a network on 200 ms of round-trip latency, it would take at least 600 ms for the browser begins to download the object.

Figure 4-2. Four-Way NTLM Authentication



For a more in-depth discussion on NTLM authentication, go to:

- http://www.innovation.ch/personal/ronald/ntlm.html

A detailed explanation of the Kerberos protocol is beyond the scope of this guide. For details, go to:

- http://simple.wikipedia.org/wiki/Kerberos_(protocol%)
- http://technet.microsoft.com/en-us/library/cc772815(WS.10).aspx

There is the concept of per-request and per-connection authentication with HTTP authentication. A server configured with per-request authentication requires the client to authenticate every single request before the server delivers the object to the client. If there are 100 objects (for example, .jpg images), it performs authentication 100 times with per-request authentication. With per-connection authentication, if the client only opens a single connection to the server, then the client only needs to authenticate with the server once. No further authentication is necessary. Using the same example, only a single authentication is required for the 100 objects. Whether the Web server does per-request or per-connection authentication varies depending on the software. For Microsoft's IIS server, the default is per-connection authentication when using NTLM authentication, the default is per-request authentication when using Kerberos authentication.

When the browser first connects to the server, it does not know whether the server has authentication enabled. If the server requires authentication, the server responds with a 401 Unauthorized message. Within the body of the message, the server indicates what kind of authentication scheme it supports in the WWW-Authenticate line. If the server supports more than one authentication schemes, then there are multiple WWW-Authenticate lines in the body of the message. For example, if a server supports both Kerberos and NTLM, the following appears in the message body:

```
WWW-Authenticate: Negotiate
WWW-Authenticate: NTLM
```

When the browser receives the Negotiate keyword in the WWW-Authenticate line, it first tries Kerberos authentication. If Kerberos authentication fails, it falls back to NTLM authentication.

Assuming that there are multiple TCP connections in-use, after the authentication succeeds on a first connection, the browser downloads and parses the base page as they authenticate. To improve performance, most browsers parse the Web page and start to download the objects over parallel connections (for details, see "Multiple TCP Connections and Pipelining" on page 65).

Connection Jumping

When the browser establishes the second, or subsequent, TCP connections for the parallel or pipelining downloads, it does not remember if the server requires authentication. Therefore, the browser sends multiple GET requests over the second, or additional, TCP connections, without the authentication header. The server rejects the requests with the 401 Unauthorized messages. When the browser receives the 401 Unauthorized message on the second connection, it is aware that the server requires authentication. The browser initiates the authentication process. Yet, instead of keeping the authentication requests for all the previously requested objects on the second connection, some of the requests jump over the first and are authenticated. This is known as *connection jumping*.

Note: Connection jumping is specific behavior to Internet Explorer, unless you are using Internet Explorer 8 with Windows 7.

Figure 4-3 shows the client authenticated with the server and the client request for the index.html Web page. It parses the Web page and initiates a second connection for parallel download.

Figure 4-3. Client Authenticates with the Server

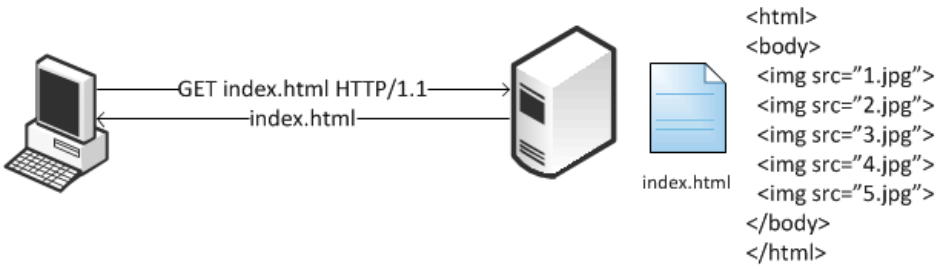


Figure 4-4 shows the client pipelines the requests to download objects 1.jpg, 2.jpg, and 3.jpg on the second connection. However, because the server requires authentication, the requests for those objects are all rejected.

Figure 4-4. Rejected Second Connection

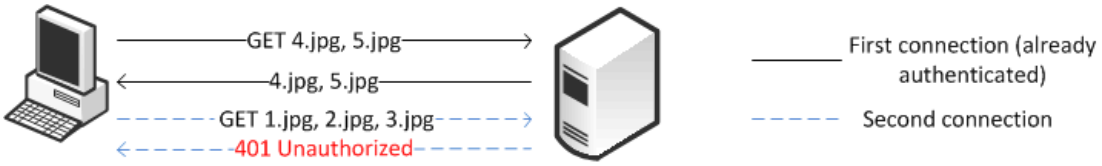
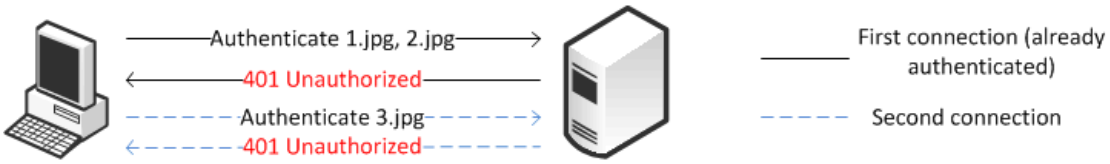


Figure 4-5 shows the client sends the authentication request to the server. Instead of keeping the requests on the second connection, some of the requests have jumped over to the first, and an already authenticated, connection.

Figure 4-5. Connection Authentication Jumping



The following problems, that result performance impact, arise as a result of this connection jumping behavior:

- If an authentication request appears on an already authenticated connection, the server can reset the state of the connection and force it to go through the entire authentication process again.
- The browser has effectively turned this into a per-request authentication even though the server can support per-connection authentication.

Note: For more details, see Strip Auth header in "HTTP Authentication Optimization" on page 76.

HTTP Proxy Servers

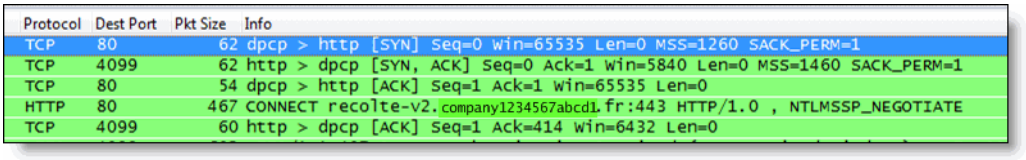
Most enterprises have proxy servers deployed in the network. Proxy servers serve the following purposes:

- Access control
- Performance improvement

Many companies have some compliance policies that restrict what the users can or cannot access from their corporate network. Enterprises meet this requirement by deploying proxy servers. The proxy servers act as a single point through which all Web traffic entering or exiting the network must traverse so the administrator can enforce the necessary policies. In addition to controlling access, the proxy server might act as a cache engine caching frequently accessed data. By doing this, it can eliminate the need to fetch the same content for different users.

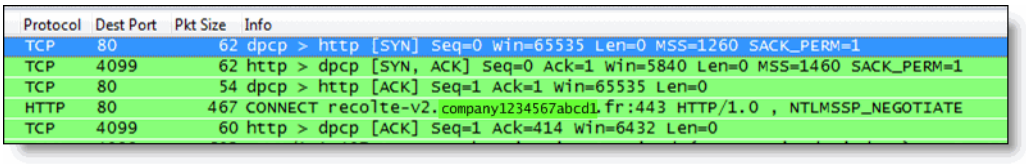
Communication with an HTTP proxy server differs from the browser to the requested HTTP server. When you use a defined proxy server, the browser initiates a TCP connection with that defined proxy server. The initiation starts with a standard TCP three-way handshake. Next, the browser requests a Web page and issues a CONNECT statement to the proxy server, with instructions to which Web server it wants to connect.

Figure 4-6. Connection to a Web Server Through a Proxy



In a standard HTTP request with an open proxy server, the proxy next opens a connection to the requested Web server and returns the requested objects. However, most corporate environments use proxy servers as outbound firewall devices, and require authentication by issuing a code 407 Proxy Authentication Required. After a successful authorization is complete, the proxy returns the originally requested objects. Successful authorization by the proxy server can be verifying username and password, and if the destination web site is on the approved list.

Figure 4-7. Client Sending NTLM Authorization



SSL connections are different. In a standard HTTP request, the Steelhead appliance optimizes proxy traffic without any issues. Standard HTTP optimized traffic is leveraged, and data reduction and latency reductions are prevalent. Due to the nature of proxy server connections, an additional step is required to set up SSL connections. Prior to RiOS v7.0, SSL connections were set up as pass-through connections and were not optimized because the SSL sessions were negotiated after the initial TCP setup. In RiOS 7.0 or later, RiOS can negotiate SSL after the setup of the session.

Figure 4-8. Setting Up an SSL Connection Through a Proxy Server

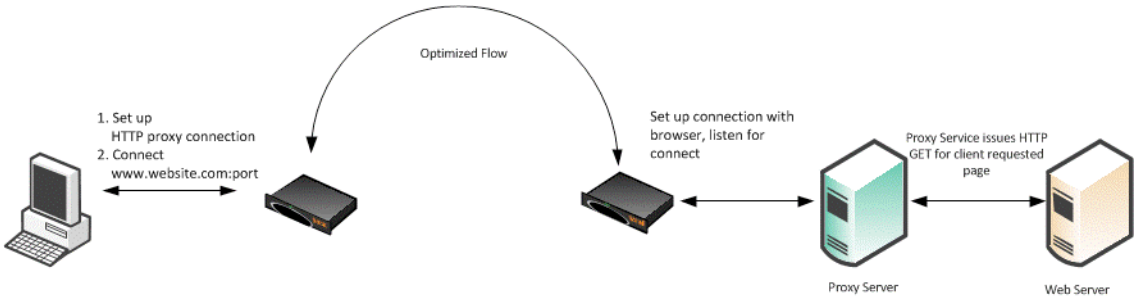


Figure 4-8 shows an SSL connection through a proxy server. The steps are as follows:

1. A client sends a TCP three-way handshake to the proxy server. The proxy HTTP is made on this connection.

2. The client issues a CONNECT statement to the host it wants to connect with, for example:

```
CONNECT www.riverbed.com:443 HTTP/1.0
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1;.NET CLR 1.0.3705;.NET CLR
1.1.4322;.NET CLR 2.0.50727;.NET CLR 3.0.4506.2152;.NET CLR 3.5.30729)
Host: www.riverbed.com
Content-Length: 0
Proxy-Connection: Keep-Alive
Pragma: no-cache
```

3. The proxy server forwards the connect request to the remote Web server.

4. The remote Web server accepts and send back an ACK.

5. The client sends an SSL Client Hello Message.

6. The server-side Steelhead appliance intercepts the Client Hello Message and begins to set up an SSL inner-channel connection with the client. The Steelhead appliance also begins to set up the SSL conversation with the original Web server.

   Note: The private key and certificate of the requested Web server must exist on the server-side Steelhead appliance, along with all servers targeted for SSL optimization.

7. An entry is added to a hosts table on the server-side Steelhead appliance.

   The host table is how a Steelhead appliance discerns which key is associated with each SSL connection, because each SSL session is to the same IP and port pair. The hostname becomes the key field for managing this connection.

8. The server-side Steelhead appliance passes the session to the client-side Steelhead appliance using SSL optimization. For details on SSL optimization, see "SSL Deployments" on page 141

Configuring HTTP SSL Proxy Interception

In some HTTP proxy implementations, the proxy server terminates the SSL session to the Web server to inspect the Web payload for policy enforcement or surveillance. In this scenario, the SSL server that is defined in the SSL Main Settings page on the server-side Steelhead appliance is the SSL proxy device, and not the server being requested by the browser. If you use self-signed certificates from the proxy server, you must add the CA from the proxy server to the Steelhead appliance trusted certificate authority (CA) list.

Note: For more details on SSL, see "Configuring SSL on Steelhead Appliances" on page 146.

Figure 4-9. SSL Session Setup Between the Proxy Server and the Client

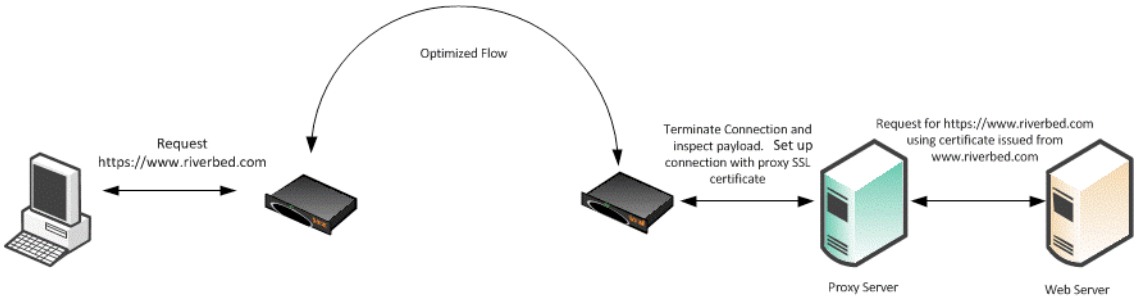
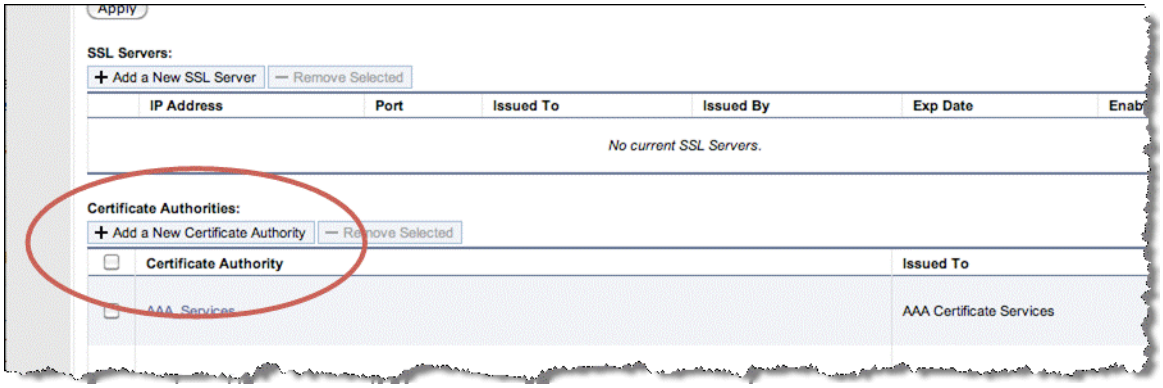


Figure 4-9 shows that the client requests a page from https://www.riverbed.com. This request is sent to the proxy server as a CONNECT statement. Because this is an intercepted proxy device, the SSL session is set up between the proxy server and the client, typically with an internally trusted certificate. You must use this certificate on the server-side Steelhead appliance for SSL optimization to function correctly.

Figure 4-10. Adding a CA



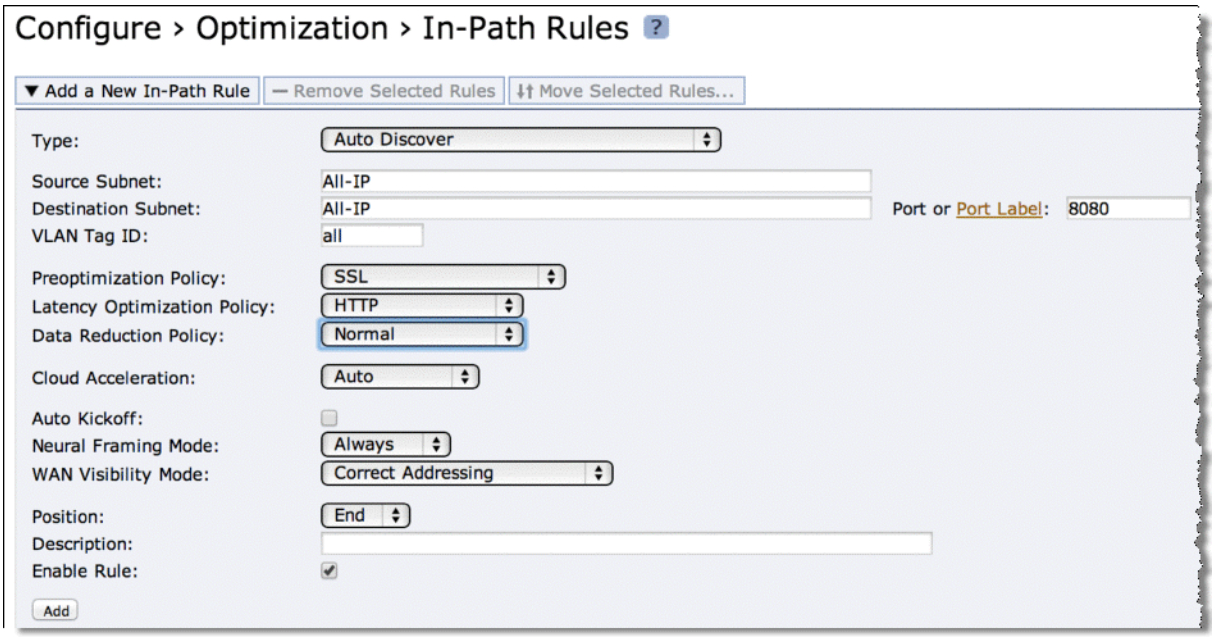
To configure RiOS SSL traffic interception

1. Before you begin:
   - You must have a valid SSL license.
   - The server-side and client-side Steelhead appliances must be running RiOS 7.0 or later.
2. Choose Configure > Optimization > SSL Main Settings and select Enable SSL Optimization.

   The private key and the certificate of the requested Web server, and all the servers targeted for SSL optimization, must exist on the server-side Steelhead appliance.
3. Choose Configure > Optimization > Advanced Settings and select Enable SSL Proxy Support.

4. Create an in-path rule on the client-side Steelhead appliance (shown in Figure 4-11) to identify the proxy server IP address and port number with the SSL preoptimization policy for the rule.

Figure 4-11. In-Path Rule



RiOS HTTP Optimization Techniques

The HTTP optimization module was first introduced in RiOS v4.x. At the time, the only feature available was URL learning. Subsequently, in RiOS v5.x, the parse-and-prefetch and metadata response features were added to the family. In RiOS v6.0, the metadata response feature was replaced with the object prefetch table. In RiOS v6.1, major enhancements were made specifically to optimize HTTP authentication.

RiOS v7.0 introduces HTTP automatic configuration. With HTTP automatic configuration you can:

- enable HTTP automatic configuration to profile HTTP applications. In other words, HTTP automatic configuration collects statistics on applications and dynamically generates a configuration entry with the proper optimization settings and self-tunes its HTTP parameter settings.

  You can also store all objects permitted by the servers—you do not need to specify the extension type of specific objects to prefetch.

- expand the HTTP authentication options by including end-to-end Kerberos authentication. All previous configurations are still available for configuration.

You must run RiOS v7.0 or later on both the client-side Steelhead appliance and server-side Steelhead appliance to use automatic HTTP configuration and end-to-end Kerberos. For more details, see "HTTP Automatic Configuration" on page 79 and "HTTP Authentication Optimization" on page 76.

Primary Content Optimization Methods

The following list explains the primary content optimization methods:

- **Strip compression** - To conserve bandwidth, nearly all browsers support compression. The browser specifies what encoding schemes it supports in the *Accept-Encoding* line in the request. Before the server responds with a reply, it compresses the data with the encoding scheme that the client supports. To maximize the benefit of SDR, data coming from the server must decompress to allow SDR to de-duplicate this data. When Strip Compression is enabled, the HTTP optimization module removes the Accept-Encoding line from the request header before sending the request to the server. Because the modified request does not contain any supported compression scheme, the server responds to the request without any compression. This allows SDR to de-duplicate the data. With this option enabled, the amount of LAN-side traffic increases as the server no longer sends the traffic in a compressed format.
- **Insert cookie** - The HTTP optimization module relies on cookies to distinguish between different users. If the server does not support cookies, the HTTP optimization module inserts its own cookie so that it can distinguish between different users. Cookies that are inserted by the HTTP optimization module start with *rbt-http=* and are followed by a random number. Enable this option only if the server does not issue its own cookies.

  Note: The Steelhead appliance removes the Riverbed cookie when it forwards the request to the server.

- **Insert keep-alive** - Keep-alive, or persistent connection, is required for the HTTP optimization module to perform pre-fetches. If the client does not support keep-alive and the server does, the client-side Steelhead appliance inserts a Connection: keep-alive to the HTTP/1.0 response, unless the server explicitly instructed to close the connection by adding Connection: close.

  This option does not apply to situations where the client supports keep-alive but the server does not. If the server does not support keep-alive, then pre-fetching is not possible and changes must be made on the server to support keep-alive.
- **URL learning** - The Steelhead appliance learns associations between a base request and a follow-on request. This feature is most effective for Web applications with large amounts of static-content images, style sheets, and so on. Instead of saving each object transaction, the Steelhead appliance saves only the request URL of object transactions in a knowledge base, and then generates related transactions from the list. URL learning uses the Referrer header field to generate relationships between object requests and the base HTML page that referenced them, and to group embedded objects. This information is stored in an internal HTTP database. The benefit of URL Learning is faster page downloads for subsequent references to the same page, from the original browser of the requester or from other clients in the same location. When the Steelhead appliance finds a URL in its database, it immediately sends requests for all of the objects referenced by that URL, saving round trips for the client browser. You can think of URL Learning as an aggressive form of pre-fetching that benefits all users in a location instead of just a single user, and that remembers what was fetched for subsequent accesses.
- **Parse-and-prefetch** - The Steelhead appliance includes a specialized algorithm that determines which objects are going to be requested for a given Web page, and pre-fetches them so that they are available when the client makes a request. This feature complements URL Learning by handling dynamically generated pages and URLs that include state information.

  Parse-and-prefetch reads a page, finds HTML tags that it recognizes as containing a pre-fetchable object, and sends out pre-fetch requests for those objects. Typically, a client needs to request the base page, parse it, and then send out requests for each of these objects. This still occurs, but with Parse-and-Prefetch the Steelhead appliance has quietly prefetched the page before the client receives it and has already sent out the requests. This allows the Steelhead appliance to serve the objects as soon as the client requests them, rather than forcing the client to wait on a slow WAN link.

For example, when an HTML page contains the tag <img src="my_picture.gif">, the Steelhead appliance pre-fetches the image my_picture.gif

Like URL Learning, Parse-and-Prefetch benefits many users in a location rather than just a single user. Unlike URL Learning, Parse-and-Prefetch does not remember which objects were referenced in a base request, so it does not waste space on dynamic content that changes each request for the same URL, or on dynamic URLs which is not an efficient use of space.

- **Object prefetch table** - The Steelhead appliance stores objects per the metadata information contained in the HTTP response. The object prefetch table option helps the client-side Steelhead appliance respond to If-Modified-Since (IMS) requests from the client, cutting back on round trips across the WAN. This feature is useful for applications that use a lot of cacheable content.

  Although URL learning and parse-and-prefetch can request information from the HTTP server much sooner than the client would have requested the same information, the Object Prefetch Table stores information to completely eliminate some requests on the WAN. The client browsers receive these objects much faster than they would if the object had to be fetched from the server.

  In releases previous to RiOS v8.6, all HTTP responses above 1 Mb triggered the RiOS HTTP traffic optimization to bypass the remainder of established HTTP sessions. In RiOS v8.6 and later, environments running both client-side and server-side Steelhead appliances remove the 1-Mb response limit. However, single HTTP objects returned as part of the response, which are greater than 1 Mb, are not stored in the HTTP object prefetch table by default.

Although features such as URL learning, parse-and-prefetch, and the object prefetch table (the replacement to metadata response) can help speed up applications, other factors, such as authentication, can negatively impact the performance of applications.

HTTP Vary Headers

RFC 2616 provides HTTP 1.1 the ability for client and server to determine if they can retrieve HTTP responses from local cache or if they must retrieve the responses from the origin server. The vary header consists of HTTP field names that the client evaluates. In releases previous to RiOS v8.6, responses that contained a HTTP vary header were excluded from Steelhead appliance HTTP optimization.

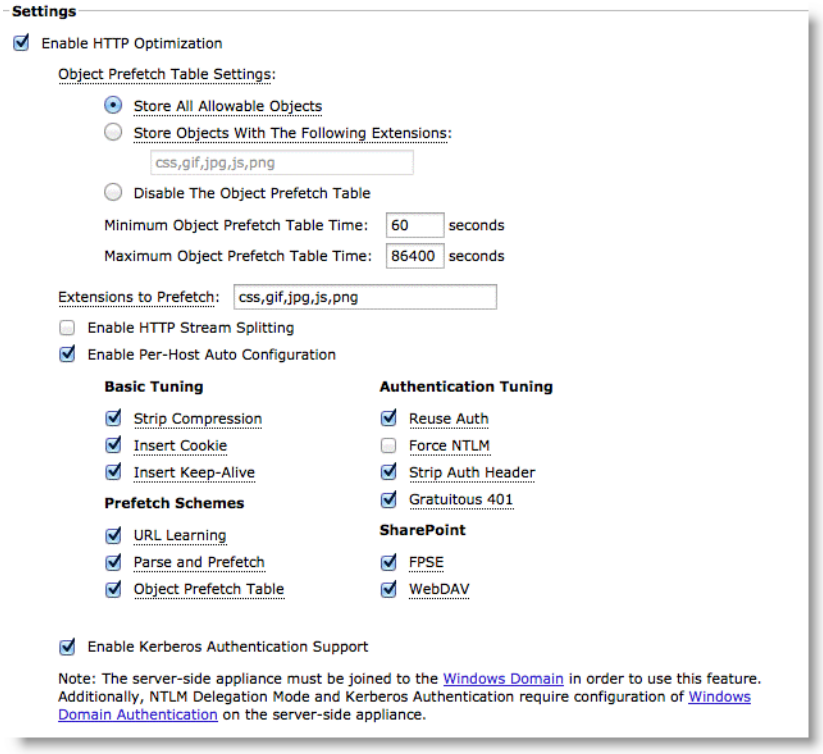
In RiOS v8.6 and later, if HTTP responses contain a vary HTTP header and are not compressed, RiOS HTTP optimization caches the response for client-side consumption. However, if RiOS receives a response with a vary HTTP header and the response is content-encoded, the local RiOS HTTP object cache does not store the HTTP object.

The following example shows an HTTP response with an HTTP vary header set without content encoding enabled:

```
HTTP/1.1 200 OK
Date: Sat, 01 Mar 2014 23:07:28 GMT
Server: Apache
Last-Modified: Fri, 28 Feb 2014 10:23:10 GMT
ETag: "183-4f374d3157f80-gzip"
Accept-Ranges: bytes
Vary: User-Agent
Content-Encoding: gzip
Content-Length: 265
Keep-Alive: timeout=15, max=98
Connection: Keep-Alive
Content-Type: text/css
```

As a best practice, enable the HTTP Strip Compression option on the HTTP page. You can apply this as part of the HTTP automatic configuration settings or manually per HTTP server. For more information on HTTP automatic configuration, see "HTTP Automatic Configuration" on page 79.

Figure 4-12. HTTP Optimization Configuration



Connection Pooling

Connection pooling pre-establishes 20 inner channel connections between each pair of Steelhead appliances. HTTP traffic benefits the most from connection pooling, although connection pooling is not specific to HTTP. When the Steelhead appliance requires an inner channel, it picks one from the pool and therefore eliminates the time for the TCP three-way handshake. The reason HTTP traffic benefits the most is because those connections are typically short-lived.

Note: Connection pooling is available only when using the Correct Addressing as the WAN visibility mode. For details, see the *Steelhead Appliance Deployment Guide*.

HTTP Authentication Optimization

RiOS v6.1 or later has specific HTTP authentication optimization for handling the various inefficient browser authentication behaviors. The HTTP authentication optimization attempts to modify the client-to-server behavior so that it maximizes the benefit of the HTTP optimization module.

RiOS v7.0 or later supports Kerberos as an authentication mechanism, in addition to the NTLM-based authentication supported by previous RiOS versions. Kerberos authentication support is beneficial for access to SharePoint, Exchange, IIS, and other Microsoft applications that use Active Directory and Kerberos for authentication. With this feature, your system is capable of prefetching resources when the Web server employs per-request Kerberos authentication.

Prior to RiOS v7.0, servers that required Kerberos authentication did not take advantage of the parse-and-prefetch optimization feature. RiOS v7.0 or later can decrypt the Kerberos service ticket and generate session keys to authenticate, on a per-request basis, with the Web server. For more details on Kerberos, see "Kerberos" on page 45.

The following is a list of HTTP authentication optimization methods:

- **Re-Use auth** - URL learning and parse-and-prefetch sends a particular request from the browser and triggers the HTTP optimization module to pre-fetch the objects of a Web page. The browser typically opens parallel TCP connections to the server to download the objects (for details, see "Multiple TCP Connections and Pipelining" on page 65). If the Re-use Auth feature is not enabled, the HTTP optimization module does not distribute the objects to the client through an unauthenticated connection, even though it can already have the objects in its database. With Re-use auth enabled, the HTTP optimization module requires that the session has already been authenticated and that it is safe (in other words, without violating any permissions) to deliver the pre-fetched objects to client regardless of whether the connection is authenticated or not. Re-use Auth is as if the client only used a single connection to download all the objects in serial.
- **Force NTLM** - The default authentication behavior on Microsoft's IIS server is per-request authentication for Kerberos and per-connection authentication for NTLM. Thus, the HTTP optimization module is configured to change the client-to-server negotiation so that the client chooses an authentication that maximizes the benefit of the HTTP optimization module. When enabled, Force NTLM removes the WWW-Authenticate: Negotiate line from the 401 Unauthorized message from the server. When the 401 Unauthorized message arrives at the client, the only authentication option available is NTLM. The client has no choice but to use NTLM for authentication.

  Note: Do not use this feature if Kerberos authentication is required.

- **Strip auth header** - When you enable strip auth header, the HTTP optimization module detects an authentication request on an already authenticated connection. It removes the authentication header before sending the request to the server. Because the connection is already authenticated, the server delivers the object to the client without having to go through the entire authentication process again. If you require Kerberos authentication, do not use strip auth header.

  Note: Strip Auth Header specifically addresses the issue described in "Connection Jumping" on page 68.

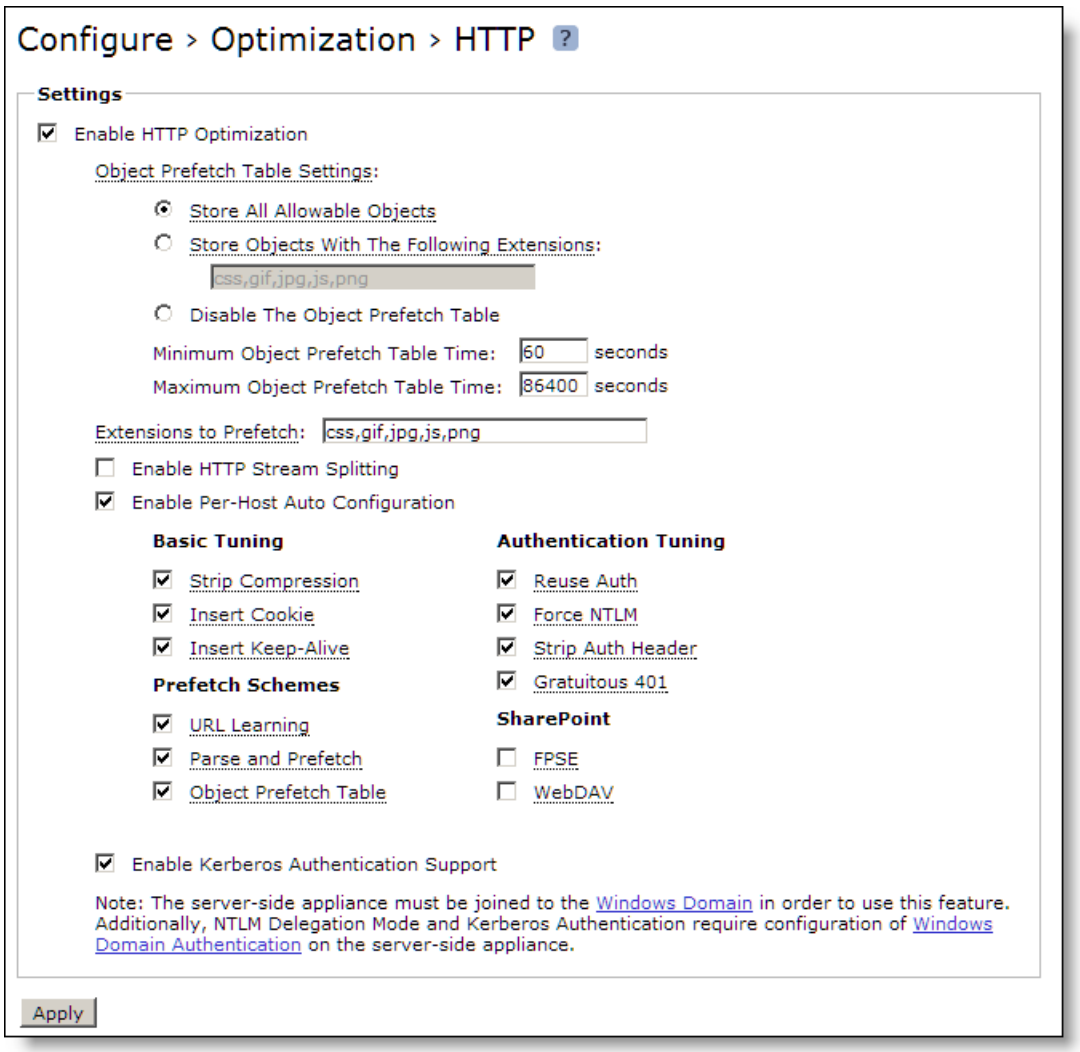
- **Gratuitous 401** - You can use this feature with both per-request and per-connection authentication but it is most effective when used with per-request authentication. With per-request authentication, every request must be authenticated against the server before the server would serve the object to the client. Most browsers do not cache the server response requiring authentication, and this wastes one round-trip for every GET request. With gratuitous 401, the client-side Steelhead appliance caches the server response. When the client sends the GET request without any authentication headers, it responds locally with a 401 unauthorized message and saves a round trip.

  The HTTP optimization module does not participate in the actual authentication. The HTTP optimization module informs the client that the server requires authentication without requiring it to waste one round trip.

To enable HTTP Kerberos authentication

1. Install RiOS v7.0 or later on the client-side and server-side Steelhead appliance.
2. Join the server-side Steelhead appliance to the active domain directory.
3. Enable Kerberos key replication on the server-side Steelhead appliance.
4. On the HTTP page, select Enable Kerberos Authentication Support.
5. Click **Apply**.

Figure 4-13. Configuring HTTP Kerberos Authentication



HTTP Automatic Configuration

RiOS v7.0 or later reduces the complexity of configuring the HTTP optimization features by introducing an automatic configuration method. HTTP optimization can automatically detect new HTTP Web hosts and initiate an evaluation of the traffic statistic to determine the most optimal setting. Rule sets are built automatically. The analysis of each HTTP application is completed and configured independently per client-side Steelhead appliance.

Riverbed recommends that both the client-side and server-side Steelhead appliances are running RiOS v7.0 or later for full statistics gathering and optimization benefits. For more details, see https://supportkb.riverbed.com/support/index?page=content&id=S17039.

HTTP automatic configuration is enabled by default—the Steelhead appliance starts to profile HTTP applications upon startup.

A detailed description of the HTTP automatic configuration process phases are as follows:

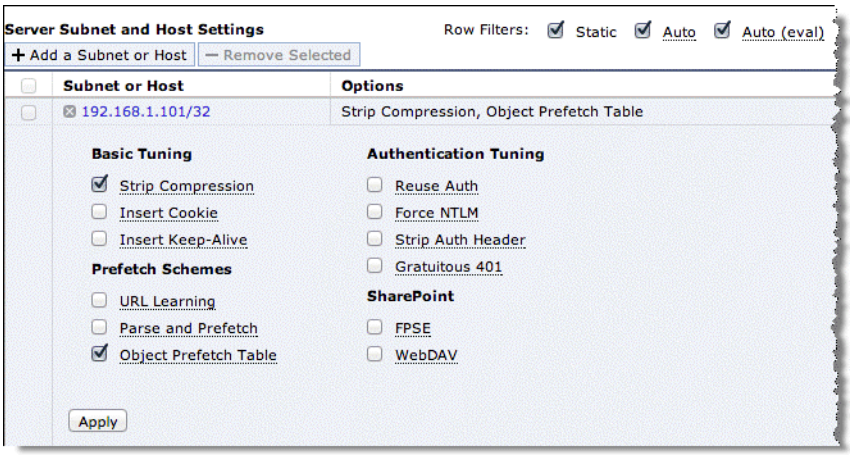
1. **Identification phase** - HTTP applications are identified by a hostname derived from the HTTP request header: for example, host: sharepoint.example.com or www.riverbed.com. This makes applications more easily identifiable when an HTTP application is represented by multiple servers. For example, http://sharepoint.example.com is collectively resolved to multiple IP addresses, each belonging to multiple servers for the purpose of load balancing. A single entry encompasses the multiple servers.

2. **Evaluation phase** - The Steelhead appliance reads the metadata and builds latency and throughput-related statistics on a per-HTTP application basis. After the Steelhead appliance detects a predetermined number of transactions, it moves to the next phase. During the evaluation phase, the Web server host has an object prefetch table, insert keep alive, reuse authentication, strip authentication and gratuitous 401 enabled by default to provide optimization. Strip authentication might be disabled after the evaluation stage if the Steelhead appliance determines that the HTTP server requires authentication.

3. **Automatic phase** - The HTTP application profiling is complete and the HTTP application is configured. At this phase, prefetch (URL learning and parse and prefetch), strip-compression, and insert-cookie optimization features are declared as viable configuration options for this application, in addition to the options from the earlier phase. Evaluation of prefetching is based on the time difference between the server-side Steelhead appliance to the server, and the server-side Steelhead appliance to client-side Steelhead appliance. Prefetch is enabled if the time difference is significantly greater. Stripping the compression is enabled if the server-side Steelhead appliance LAN bandwidth is significantly greater than the WAN of the client-side Steelhead appliance. Insert-cookie optimization is automatically enabled only when the server does not use cookies.

4. **Static phase** - You can insert custom settings for specific hosts or subnets. You can also select an automatically configured rule and override the settings. In RiOS v7.0 or later, you can insert hostnames (for example, www.riverbed.com) along with specific IP hosts/subnets (for example, 10.1.1.1/32). If you use a subnet instead of a hostname, you must specify the subnet mask.

Figure 4-14. Subnet and Subnet Mask



RiOS v7.0 or later stores all cacheable objects, as long as the objects are allowed by the HTTP server and not prohibited by the headers in the HTTP request and response. The restriction to cache objects based on their extensions is removed. You can choose to cache all allowable objects or change to pre-RiOS v7.0 settings and explicitly point out the extension of certain object you wish to store. You can also choose to not store any objects.

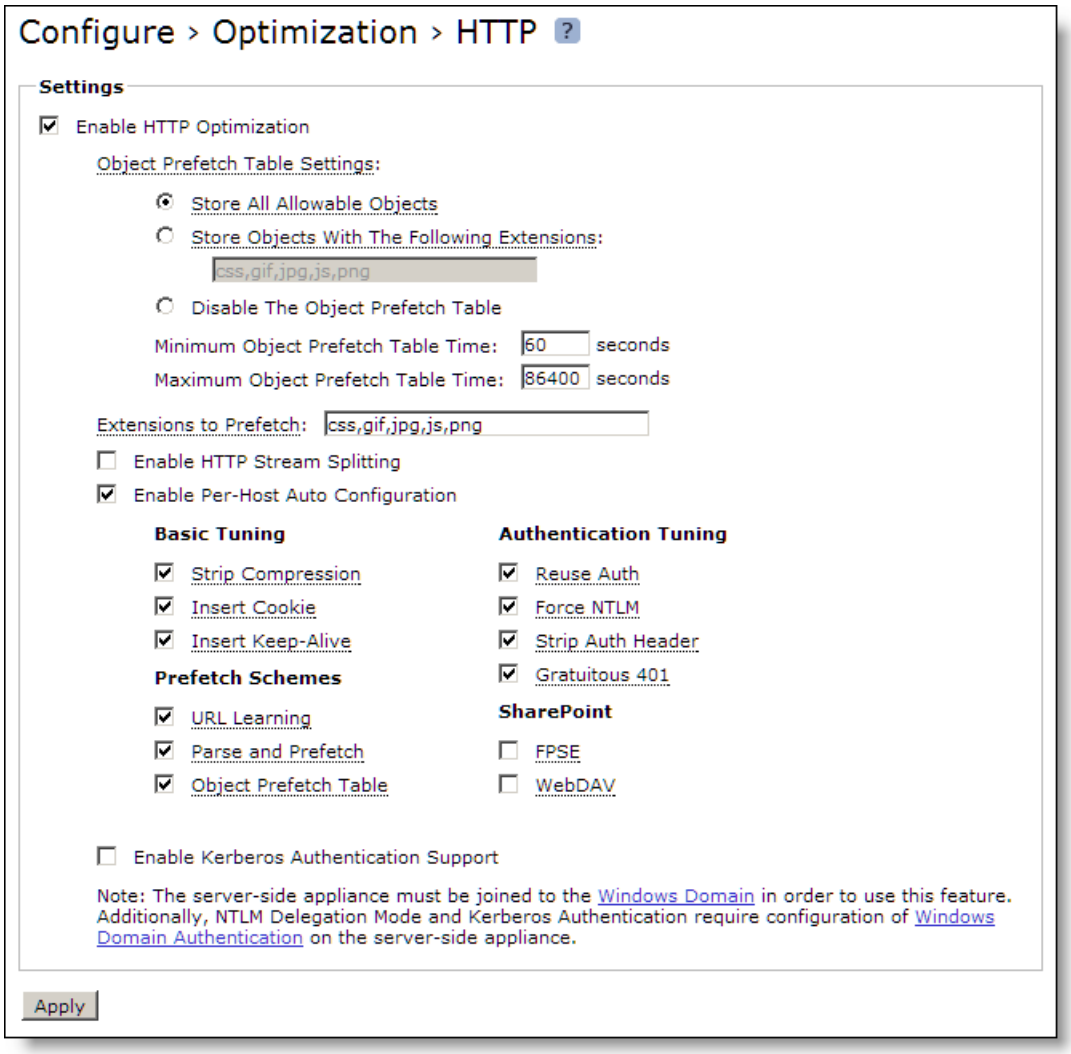
To enable HTTP automatic configuration

1. In the Management Console, choose Configure > Optimization > HTTP.
2. Select Enable Per-Host Auto Configuration.
3. Select the appropriate options. All are enabled by default.

   For more information about HTTP options, see the *Steelhead Appliance Management Console User's Guide*.

4. Click **Apply**.

Figure 4-15. HTTP Configuration Page



You can filter between the discovered and statically added hosts. When you enable HTTP automatic configuration, you can select the row filters to see auto-configured hosts and evaluated hosts. To delete the entry, you must edit an automatically configured host. The host automatically changes its configuration to become static. Only then can you remove the entry.

Note: Removal of an entry allows the Steelhead appliance to re-learn the HTTP application and begin the evaluation phase again.

You can select an automatically configured rule and edit the optimization settings. Upon completion, the rule set is now considered a static rule. This process is similar to previous RiOS versions in which manually adding subnets to optimize hosts was needed.

When you disable the automatic configuration option, only static and the default optimization rule (0.0.0.0/0) apply. The automatic and evaluation rules no longer populate the table.

Figure 4-16. Server and Host Settings Page



Note: If you restart the optimization process, you remove all automatic and evaluation configured entries. The Steelhead appliance profiles the applications again.

To configure HTTP automatic configuration with the command line

- On the client-side Steelhead appliance, connect to the CLI in configuration mode and enter the following command:

```
protocol http auto-config enable
```

To clear the HTTP automatic configuration statics with the command line

- On the client-side Steelhead appliance, connect to the CLI in configuration mode and enter one of the following commands:

```
protocol http auto-config clear-stats all
```

or

```
protocol http auto-config clear-stats hostname <http host name >
```

For more details on how to configure HTTP automatic configuration, see the *Steelhead Appliance Management Console User's Guide* and the *Riverbed Command-Line Interface Reference Manual*.

HTTP Settings for Common Applications

Riverbed recommends that you test the HTTP optimization module in a real production environment. To manually configure HTTP applications, the following table shows the recommended settings for common enterprise applications.

Application	URL Learning	Parse and Prefetch	OPT
SAP/Netweaver	No	Yes	Yes
Microsoft CRM	Yes	No	Yes
Agile	No	No	Yes
Pivotal CRM	No	Yes	Yes
SharePoint	No	Yes	Yes

As discussed in "Primary Content Optimization Methods" on page 73, URL Learning accelerates environments with static URLs and content, Parse and Prefetch accelerates environments with dynamic content, and OPT saves round trips across the WAN, and reduces WAN bandwidth consumption.

HTTP Optimization for SharePoint

You can configure Microsoft SharePoint-specific HTTP optimization options in RiOS v8.5 and later. The SharePoint optimization options are available on the Configure > Optimization > HTTP page (Figure 4-15):

- **Microsoft Front Page Server Extensions (FPSE) protocol** - enables the client application to display the contents of a Website as a file system. FPSE supports uploading and downloading files, directory creation and listing, basic file locking, and file movement in the Web server. The FPSE HTTP subprotocol is used by SharePoint 2007 and 2010 when opening documents in Microsoft Office. To increase performance, the following FPSE requests are cached:
  - URL-to-Web URL request
  - Server-version request
  - Open-service request

  One of the inherent issues with SharePoint communication is that after each request is complete, the Web server closes the connection. Thus, each new request requires a new TCP handshake. If you are using SSL, additional round trips are required. Request caching eliminates the round trips and speeds up the connection.

  FPSE supports SharePoint Office clients 2007 and 2010, installed on Windows 7 (SP1) and Windows 8. SharePoint 2013 does not support FPSE.
- **Microsoft Web Distributed Authoring and Versioning (WebDAV)** - represents a set of standardized extensions (RFC 4918) to the HTTP/1.1 protocol that enables users to collaborate, edit, and manage files on remote Web servers. Specific to SharePoint, WebDAV is a protocol for manipulating the contents of the document management system. You can use WebDAV protocol to map a SharePoint site to a drive on a Windows client machine. To properly set up as a WebDAV drive, you must create it as a mapped drive and not as a new network folder location.

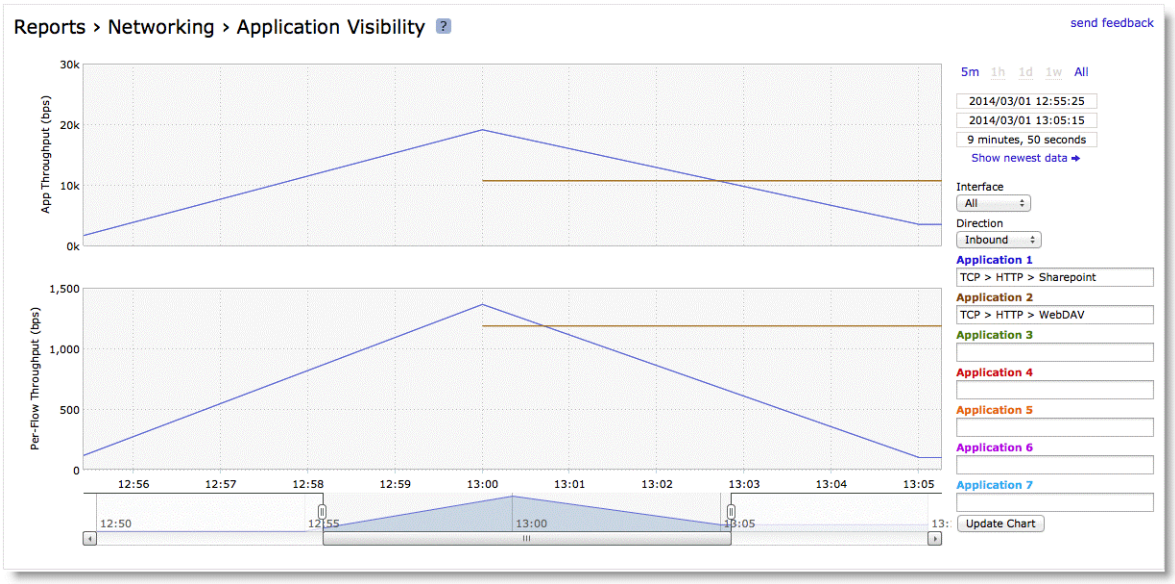
  The WebDAV optimization option in the RiOS cache often repeats metadata and replies locally from the client-side Steelhead appliance. WebDAV protocol is used in both SharePoint 2010 and 2013.
- **File Synchronization via SOAP over HTTP (FSSHTTP) Protocol** - FSSHTTP is a protocol you can use for file data transfer in Microsoft Office and SharePoint. FSSHTTP enables one or more protocol clients to synchronize changes performed on shared files stored on a SharePoint server. FSSHTTP allows coauthoring and is supported with Microsoft Office 2010 and 2013, and SharePoint 2010 and 2013.

  Currently, RiOS does not offer any latency optimization for FSSHTTP. Clients can continue to benefit from HTTP optimization, TCP streamlining, and data reduction.
- **OneDrive for Business** - OneDrive for Business is a personal library intended for storing and organizing work documents. As an integral part of Office 365 or SharePoint 2013, OneDrive for Business enables end users to work within the context of your organization, with features such as direct access to your company address book. OneDrive for Business is usually paired with Sync OneDrive for Business (sync client). This synchronization application enables you to synchronize OneDrive for Business library or other SharePoint site libraries to a local computer. OneDrive for Business is available with Office 2013 or with Office 365 subscriptions that include Microsoft Office 2013 applications.

  RiOS v8.6 and later supports optimization of traffic between OneDrive for Business and Sync OneDrive for Business. Office 365 requires a cloud-based instance of the server-side Steelhead appliance that is hosted closer to Microsoft Office 365 data center.

In RiOS v8.6 and later, you can view byte-level information in regards to the SharePoint-specific optimization in the Reports > Networking > Application Visibility report. If you have enabled AFE, the DPI engine can classify the two specific SharePoint HTTP sub-applications that are shown in the current connections report screen.

Figure 4-17. Application Visibility Report



For details on how to configure SharePoint optimization and more information about the SharePoint optimization options, see the *Steelhead Appliance Management Console User's Guide* and the *Riverbed Command-Line Interface Reference Manual*. For details on AFE, see the *Steelhead Appliance Deployment Guide* and the *Steelhead Appliance Management Console User's Guide*.

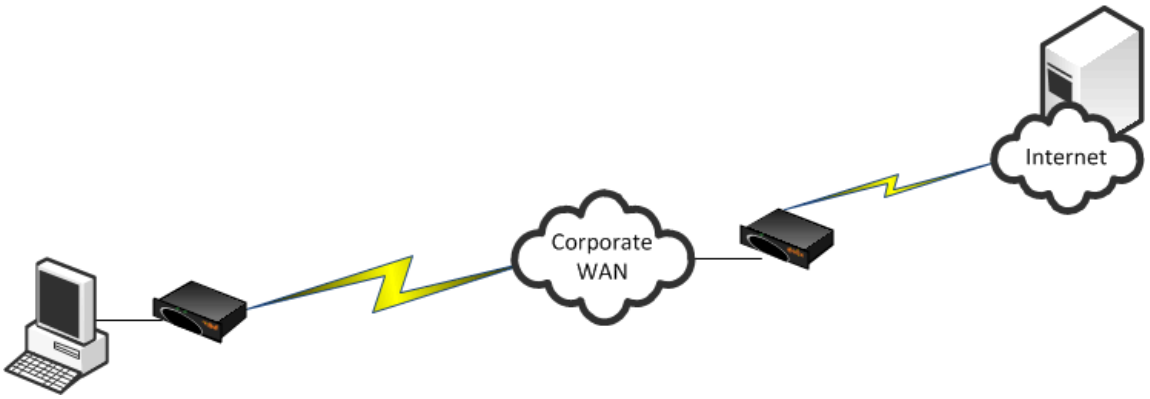
HTTP Optimization Module and Internet-Bound Traffic

Sometimes Internet access from the branch offices is back-hauled to the data center first, before going out to the Internet. Consequently, a significant portion of the WAN traffic is Internet bound traffic and you might be tempted to optimize that traffic.

For Internet-bound traffic, Riverbed recommends that you enable HTTP automatic configuration.

Similar to other Layer-7 optimization modules, the HTTP optimization module expects the latency between the server and the server-side Steelhead appliance to be LAN latency. With Internet-bound traffic, the latency between the server-side Steelhead appliance and the server is going to be far greater than typical LAN latency. In this case, the URL learning and parse-and-prefetch features are not very effective, and in certain cases, are slower than without optimization.

Figure 4-18. Internet Traffic Backhauled from the Branch Office to the Data Center Before Internet



Tuning Microsoft IIS Server

This section describes the steps required to modify the IIS server so that it can maximize the performance of the HTTP Authentication optimization module. In some instances of the HTTP Authentication optimization requires the IIS server to behave in a certain way. For example, if NTLM authentication is in use, then the HTTP optimization module expects the NTLM authentication to use per-connection authentication.

Determining the Current Authentication Scheme on IIS

IIS 6 uses *metabase* to store its authentication settings. Although you can manually modify the metabase, the supported method in querying or modifying the metabase is to use the built-in Visual Basic (VB) script adsutil.vbs. The adsutil.vbs script is located in the C:\inetpub\adminscripts directory on the IIS server.

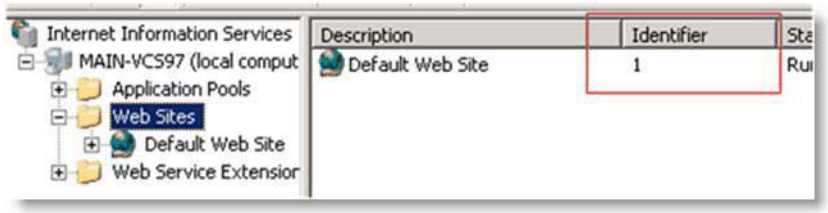
To determine the current setting on the IIS server

- From the command prompt on the Windows server, enter the following command in the c:\inetpub\adminscripts directory:

```
cscript adsutil.vbs get w3svc/$WebsiteID$/root/NTAuthenticationProviders
```

The $WebsiteID$ is the ID of the Web site hosted by the Web server. The $WebsiteID$ can be found in the IIS Manager and it's the number under the Identifier column.

Figure 4-19. Web Site ID in the Identifier Column



If the output of the command is one of the following, it indicates that the server prefers Kerberos authentication and if that fails, it falls back to NTLM:

```
The parameter "NTAuthenticationProviders" is not set at this node.
NTAuthenticationProviders       : (STRING) "Negotiate,NTLM"
```

If the output of the command is the following, it indicates that the server only supports NTLM authentication:

```
NTAuthenticationProviders       : (STRING) "NTLM"
```

Determining the Current Authentication Mode on IIS

Both NTLM and Kerberos authentication can support either per-connection or per-request authentication. To determine the current authentication mode on IIS, use the same VB script, adsutil.vbs, but query a different node:

```
cscript adsutil.vbs get w3svc/$WebsiteID$/root/AuthPersistSingleRequest
```

The output of the command should be similar to the following text:

```
The parameter "AuthPersistSingleRequest" is not set at this node; or

AuthPersistSingleRequest        : (BOOLEAN) False; or

AuthPersistSingleRequest        : (BOOLEAN) True
```

Per-Connection or Per-Request NTLM Authentication

If the authentication scheme is NTLM and the output of the authentication mode is either *not set* or *false*, then the server is configured with per-connection NTLM authentication. If the output of the authentication mode is *true*, then the server is configured with per-request NTLM authentication.

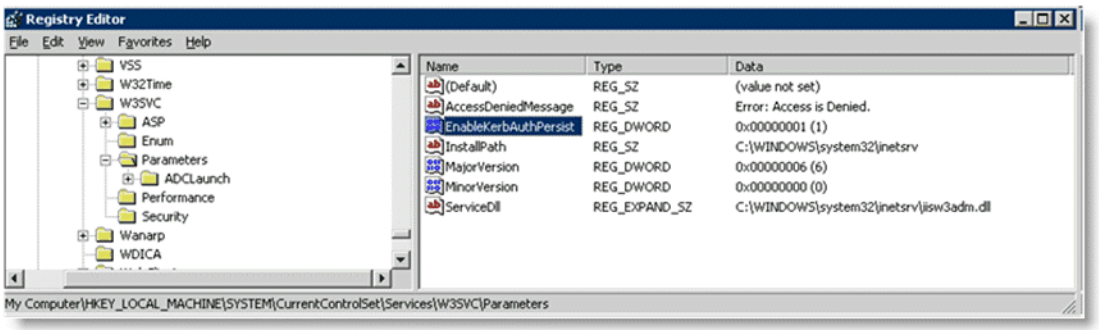
Per-Connection or Per-Request Kerberos Authentication

If the authentication scheme is Kerberos, you must perform extra steps you can determine whether the server is using per-connection or per-request authentication.

To determine whether Kerberos is using per-connection or per-request (applies to IIS v6.0)

1. Check the registry for key named EnableKerbAuthPersist under HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W3SVC\Parameters. If the key does not exist, or if the key does exist but has a value of zero, then Kerberos is using per-request authentication.

Figure 4-20. Registry Editor Window



2. If the key exists but has a non-zero value, and the output from authentication mode is false, then Kerberos might be using per-connection Kerberos. Kerberos is using per-connection authentication if the server running is running IIS v6.0 and has the patch installed per Microsoft's knowledge base article http://support.microsoft.com/kb/917557.

For details on IIS v7.0, go to http://support.microsoft.com/kb/954873.

The following table shows the different combinations and whether Kerberos would perform per-connection or per-request authentication.

		EnableKerbAuthPersist/AuthPersistNonNTLM**		
		Not-set	**Non-zero**	**Zero**
AuthPersistSingleRequest	**Not-set**	Per-request	Per-connection*	Per-request
	TRUE	Per-request	Per-request	Per-request
	FALSE	Per-request	Per-connection	Per-request

* requires IIS patch
** AuthPersistNonNTLM of IIS 7 replaces EnableKerbAuthPersist of IIS 6

Changing the Authentication Scheme

You can change the authentication by using the adsutil.vbs script.

To modify the server so that it only supports NTLM authentication

- From the command prompt on the Windows server, enter the following command:

```
cscript adsutil.vbs set w3svc/$WebsiteID$/root/NTAuthenticationProviders "NTLM"
```

You can configure the server to attempt Kerberos authentication first before NTLM by changing the last parameter to *Negotiate, NTLM*.

Remember to restart the IIS server after the changes have been made.

Changing the Per-Connection/Per-Request NTLM Authentication Mode

By default, NTLM uses per-connection authentication.

To change per-connection NTLM to per-request NTLM

- From the command prompt on the Windows server, enter the following command:

```
cscript adsutil.vbs set w3svc/$WebsiteID$/root/AuthPersistSingleRequest TRUE
```

To change NTLM back to its default, replace the word TRUE with FALSE and restart the IIS server.

Note: When you use NTLM, the HTTP optimization module works best when per-connection authentication is set.

Changing the Per-Connection/Per-Request Kerberos Authentication Mode

By default, Kerberos uses per-request authentication. There are several ways to configure Kerberos to use per-connection authentication. For details, see "Per-Connection or Per-Request Kerberos Authentication" on page 86.

HTTP Authentication Settings

The following table shows some of the recommended configurations for the HTTP Authentication optimization. In this instance, assume that it is not possible to make any modifications on the IIS server.

For example, if the authentication setting on the IIS server is per-request Kerberos, enabling Force NLTM forces the client to use NTLM and in turn, the HTTP optimization module can provide better optimization by using the URL Learning and Parse-and-Prefetch features. If NTLM authentication is not an option, then the only possibility is to enable Gratuitous 401.

If you modify the settings on the IIS server, then Force NTLM might not be necessary. In this case, only the first row of the following table is applicable.

The HTTP optimization module expects the default authentication behavior for both NTLM and Kerberos (in other words, per-connection authentication for NTLM and per-request authentication for Kerberos). Continuing with the example earlier, if the IIS server is configured to use the non-standard authentication scheme by using per-request Kerberos and per-request NTLM, then using Force NTLM does not help as it changes from per-request Kerberos to per-request NTLM. In this case, URL Learning and Parse-and-Prefetch is not effective.

IIS Authentication	Recommended Configurations	Notes
per-connection NTLM	Reuse Auth. + Strip Auth. Header + Grat. 401	
	Reuse Auth. + Strip Auth. Header	
per-request Kerberos	Reuse Auth. +Force NTLM + Strip Auth. Header + Grat. 401	N/A if NTLM is not an option
	Reuse Auth. +Force NTLM + Strip Auth. Header + Grat. 401	N/A if NTLM is not an option
	Grat. 401	
per-request NTLM	Grat. 401	
per -connection Kerberos	Reuse Auth. + Force NTLM + Strip Auth. Header	N/A NTLM is not an option
	Reuse Auth.	
	Change to per-request Kerberos and turn on	
	Grat. 401	
Don't know	Reuse Auth. + Strip Auth. Header + Grat. 401	
	Reuse Auth. + Strip Auth. Header	

HTTP Optimization Module and Proxy Servers

Some network deployments might involve HTTP proxy servers to speed up resource retrieval, apply access control policies, or audit and filter contents. In general, the Steelhead appliances provide full optimization benefits even with proxy servers in place. Optimized connections can be limited or even hindered in the following circumstances:

- **Proxy authentication** - If the proxy server uses authentication, pre-fetching performance can be affected. Similar to Web server authentication, proxy authentication can limit the pre-fetching capability of the connection where pre-fetched resources are served when the proxy server uses per-connection authentication. In the case of the per-request mode, pre-fetching is not possible as every single request requires user authentication. HTTP authentication optimization features apply to both Web server and proxy authentication.
- **Proxy caching** - A proxy server can maintain its own local cache to accelerate access to resources. Generally, proxy caching does not thwart HTTP optimization. If it caches objects for an excessively long time, Object Prefetch Table (OPT) might not provide full benefits as the retrieved resource might not be sufficiently fresh.
- **Selective proxying** - If the same Web server is accessed directly by a client and through a proxy server, URL Learning might not work properly. URL Learning builds a pre-fetch tree by observing ongoing requests. This is based on the assumption that the same URL is requested by the client if the same base page is requested again.

  When a pre-fetch tree is created without a proxy, the observed partial URLs comprise the tree. If another client asks for the same page but through a proxy, the proxy fails to forward the pre-fetched responses because it expects full URLs to be able to send the responses back to the client. Thus URL Learning might not work well when proxy is selectively employed for the same host. This only happens with URL Learning. Other pre-fetching schemes, such as Parse-and-Prefetch and Object Prefetch Table, are not subjected to this issue.
- **Fat Client** - Not all applications accessed through a Web browser use the HTTP protocol. This is especially true for fat clients that run inside a Web browser that might use proprietary protocols to communicate with a server. HTTP optimization does not improve performance in such cases.

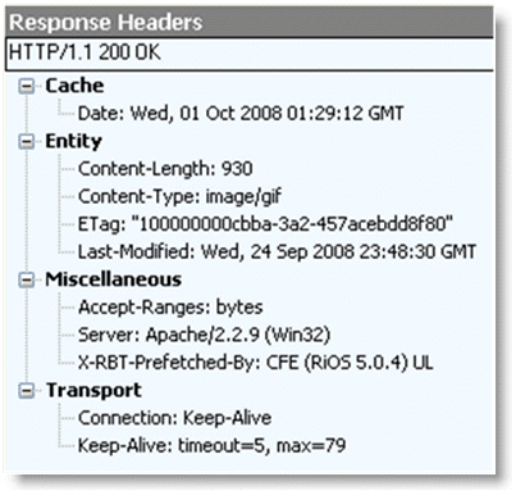
Determining the Effectiveness of the HTTP Optimization Module

Figure 4-21 shows that when an object is optimized by the HTTP optimization module, the response header contains the line X-RBT-Prefetched-By. The X-RBT-Prefetched-By line also contains the name of the Steelhead appliance, the version of system, and the method of optimization. The different methods of optimization are:

- **UL** - URL Learning.
- **PP** - Parse-and-Prefetch.
- **MC** - Metadata Response (pre-RiOS v6.0).

- **PT** - Object Prefetch Table (RiOS v6.0 or later).
- **AC** - Gratuitous 401.

Figure 4-21. Response Header Contained in the Line X-RBT-Prefetched-By



The response header can be captured by using tcpdump, HTTPWatch, Fiddler or similar tools. Because the HTTP optimization module is a client-side driven feature, the capture must be done on the client itself.

Info-Level Logging

You can look at the client-side Steelhead appliance log messages to determine whether or not the HTTP optimization module is functioning. The logging level on the Steelhead appliance must be set at info level logging for the messages to appear in the log. If the HTTP optimization module is pre-fetching objects, a message similar to the following appears in the logs. This only applies to the URL Learning and Parse-and-Prefetch. No log messages are displayed for Metadata Response or Object Prefetch Table.

```
Sep 25 12:28:57 CSH sport[29969]: [http/client.INFO] 2354 {10.32.74.144:1051 10.32.74.143:80}
Starting 40 Prefetches for Key->abs_path="/" host="10.32.74.143" port="65535" cookie="rbt-
http=2354"
```

The key specified in the log message is not necessarily the object that triggered the pre-fetch operation. In RiOS v6.0 or later, the log message includes the object that triggered the pre-fetch operation.

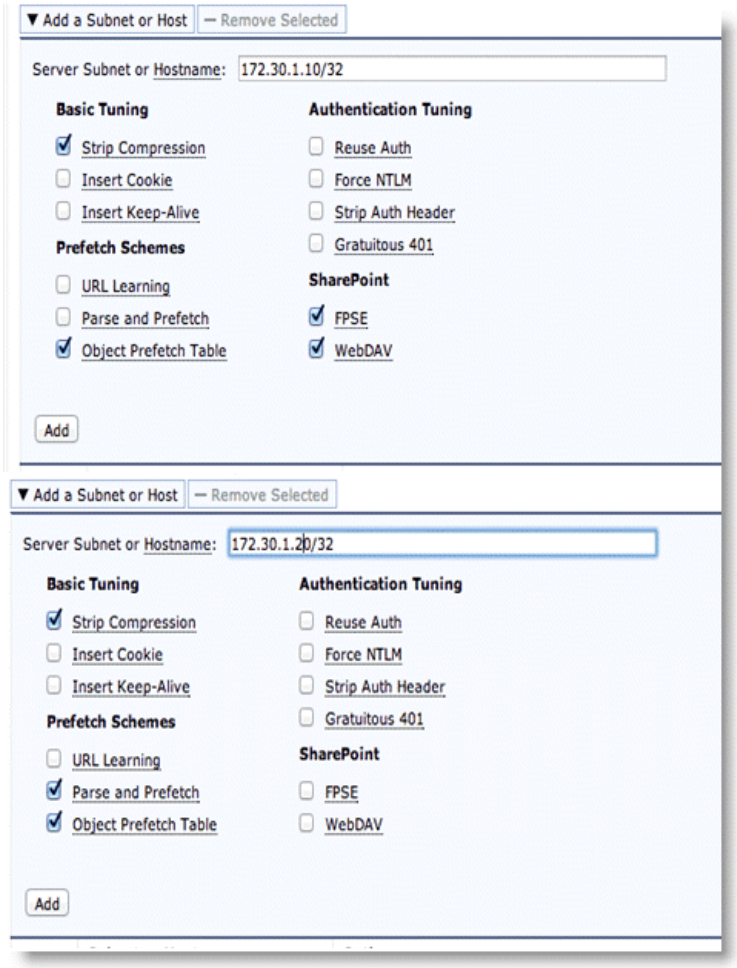
Use Case

While automatic configuration is typically the preferred method of configuration, you have the option of manual configuration. The following use case shows a manual configuration.

A customer has a 1.5 Mbps link with 100 ms latency between the branch office and the data center. The PCs in the remote office are running Microsoft Windows XP with Internet Explorer 7. Users in the remote offices are complaining of slow access for SAP Netweaver and Microsoft SharePoint. The SAP Netweaver server has an IP address of 172.30.1.10 and the Microsoft SharePoint server has an IP address of 172.16.2.20.

Because both SAP Netweaver and Microsoft SharePoint are well-known applications, the customer configured the following on the client-side Steelhead appliance.

Figure 4-22. Two Subnet Server Settings Showing the New Recommended SharePoint Settings



After configuring the settings above, the customer noticed a significant improvement in response time for SAP Netweaver but no changes for Microsoft SharePoint—even though the connections are optimized with good data reduction. One of the users mentioned that the Microsoft SharePoint portal required authentication, which might be the reason why Parse and Prefetch did not work. Unfortunately, the system administrator in charge of the SharePoint portal cannot be reached at this moment and you cannot check the authentication setting on the server.

Instead of checking the authentication on the server, you can capture tcpdump traces and check for the authentication scheme in use. Figure 4-23 shows the server has Kerberos enabled and hence the client attempts to authenticate using Kerberos first.

Figure 4-23. TCP Dump Trace Confirming Kerberos Enabled

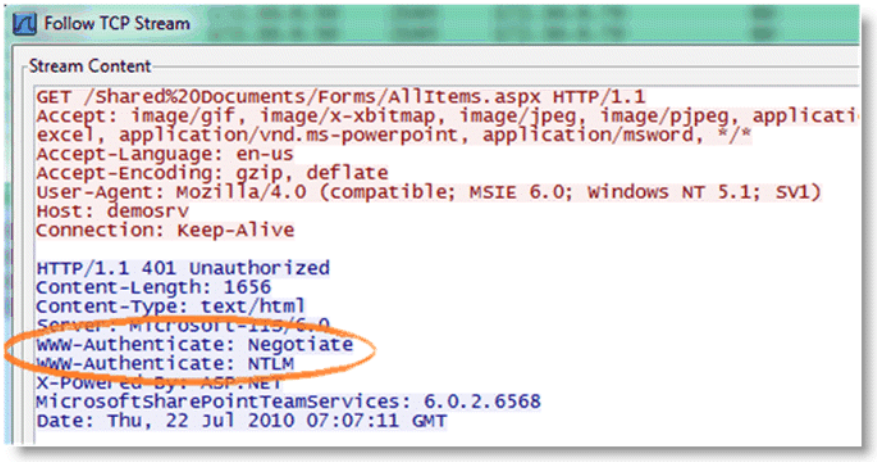


Figure 4-24 confirms that by scrolling through the trace, the per-request Kerberos is configured on the server.

Figure 4-24. TCP Dump Trace Showing Per-Request Kerberos

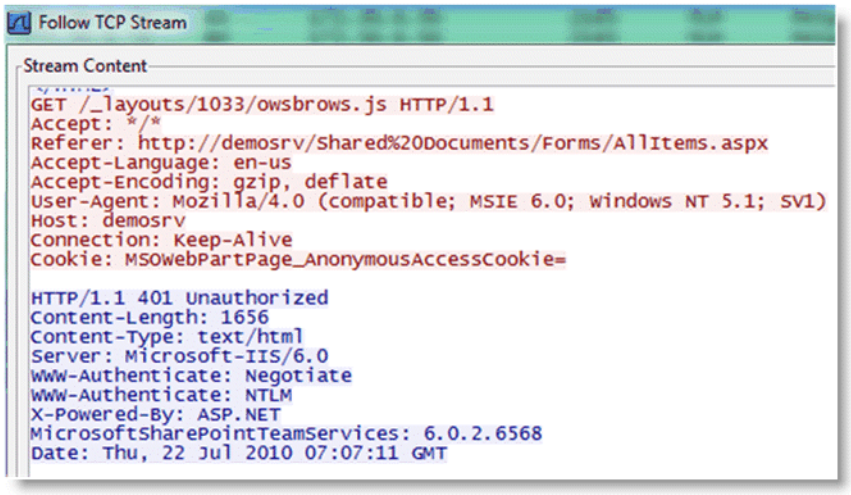
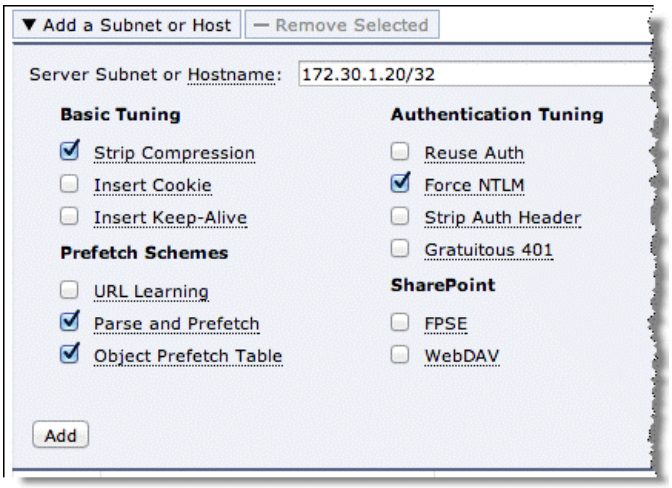


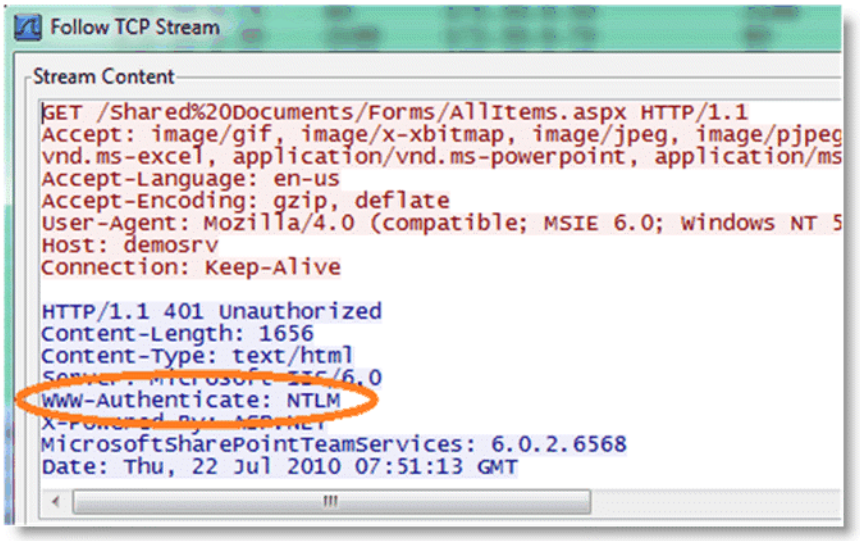
Figure 4-25 shows that given this information, the best option is to enable **Force NTLM** for the SharePoint server.

Figure 4-25. Enable Force NTLM



Taking another trace on the client-side Steelhead appliance confirms that the only authentication option available is NTLM. Because there is no other authentication option but NTLM, the client is forced to authenticate via NTLM and Parse and Prefetch and pre-fetches can once again function as before.

Figure 4-26. Trace Stream Showing NTLM as the Only Authentication Available



In this instance, it is not necessary to enable the other features, as the entire transaction took place over a single connection. If the client uses multiple TCP connections, then it might be necessary to enable re-use auth, strip auth header, and gratuitous 401. Enabling the rest of the features does not provide any benefit in this instance, but it does not cause any problems either.

CHAPTER 5 Citrix ICA Optimization

To consolidate operations, some organizations deploy desktop and application virtualization solutions such as Citrix XenDesktop and XenApp in the data center. Citrix uses a proprietary protocol called Independent Computing Architecture (ICA) to provide connectivity between its clients (called *receivers*) and its published applications and desktops.

This chapter includes the following sections:

- "Overview of Citrix ICA" on page 95
- "Citrix ICA Traffic Optimization with Steelhead Appliances" on page 96
- "Secure Citrix ICA Encryption" on page 97
- "Citrix Drive-Mapping Optimizations" on page 98
- "Citrix Multi-Stream ICA Traffic Optimization with Steelhead Appliances" on page 99
- "QoS Classification for Citrix Traffic" on page 104
- "Reduction for Citrix Small Packet Real-Time Traffic" on page 106
- "Citrix ICA Optimization Over SSL" on page 106

Overview of Citrix ICA

RiOS v6.0 and later provide these optimizations:

- Classification and shaping of Citrix ICA traffic using Riverbed QoS to improve the end-user desktop experience
- Bandwidth reduction of compressed and encrypted Citrix ICA traffic using Steelhead appliance Citrix optimization

RiOS v7.0 and later provides these optimizations:

- Latency optimization for client drive mapping (CDM) in the Citrix ICA session
- Optimization of Citrix sessions over SSL using Citrix Access Gateway (CAG)
- Steelhead appliance Citrix Optimization for Multi-Port ICA traffic

RiOS v7.0.4 and later provides traffic optimization for enhanced data reduction for small Citrix packets.

Citrix Version Support

RiOS v6.0 and later provides support for the following Citrix software components.

Citrix Receiver or ICA client versions:

- Online plug-in version 9.x
- Online plug-in version 10.x
- Online plug-in version 11.x
- Online plug-in version 12.x
- Online plug-in version13.x (Receiver version 3.x)
- Receiver for Windows version 4.x

Citrix XenDesktop:

- XenDesktop 4
- XenDesktop 5
- XenDesktop 5.5
- XenDesktop 5.6

Citrix XenApp:

- Presentation Server 4.5
- XenApp Server 5
- XenApp Server 6
- XenApp Server 6.5

In addition, RiOS supports encrypted and compressed Citrix ICA traffic optimization.

For details on configuring Citrix optimization, see the *Steelhead Appliance Management Console User's Guide*, the *Riverbed Command-Line Interface Reference Manual*, and the white paper *Optimizing Citrix ICA Traffic with RiOS 8.0 (June 2013)*.

Citrix ICA Traffic Optimization with Steelhead Appliances

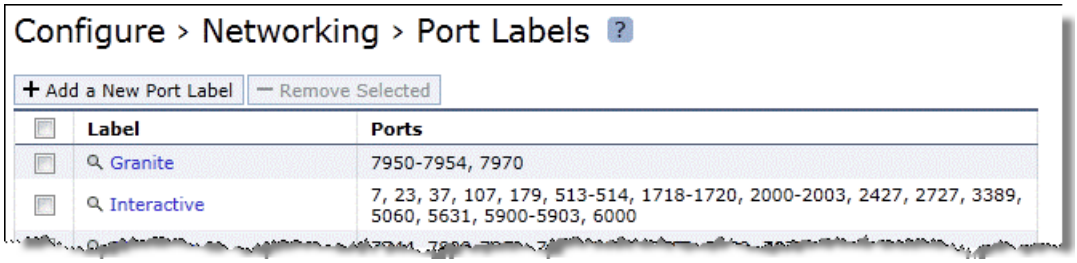
Citrix ICA traffic is usually transported on TCP port 1494. Citrix ICA traffic is transported on TCP port 2598 if Citrix Session Reliability is enabled. When you enable Citrix Session Reliability, the client tunnels its ICA traffic inside the Common Gateway Protocol (CGP) port 2598.

For more information about Citrix session reliability, go to http://support.citrix.com/article/CTX104147.

To configure Citrix ICA traffic optimization with a Steelhead appliance

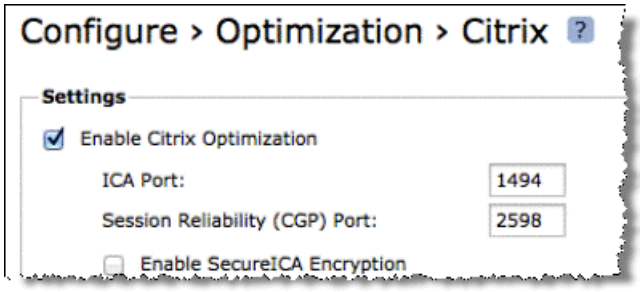
1. From the Steelhead Management Console, choose Configure > Networking > Port Labels and remove port 1494 and 2598 from the Interactive port label.

Figure 5-1. Port Labels



2. Choose Configure > Optimization > Citrix.

Figure 5-2. Citrix Page



3. Select Enable Citrix Optimization.
4. Specify the ICA port and Session Reliability (CPGP) Port.
5. Click **Apply** to apply the settings to the running configuration.
6. Restart the optimization service.

You must perform this configuration on both the client-side and server-side Steelhead appliances.

Secure Citrix ICA Encryption

SecureICA is a Citrix ICA protocol feature that encrypts the session data transmitted between Citrix clients and servers. By default, all Citrix ICA sessions have SecureICA set to basic ICA protocol encryption. The basic ICA protocol encryption setting obfuscates data but does not provide industry standard encryption. Other settings are available to increase the level of encryption using the RC5 algorithm.

For more information on SecureICA, go to:

- http://support.citrix.com/proddocs/topic/xenapp6-w2k8-admin/ps-securing-use-secure-ica.html
- http://support.citrix.com/proddocs/topic/xenapp6-w2k8-admin/ps-securing-sec-clsrv-conn.html

RiOS supports optimization of Citrix ICA sessions with SecureICA set to RC5 40-, 56-, and 128-bit encryption. By default, RiOS can optimize Citrix ICA traffic with SecureICA set to basic ICA protocol encryption. You must enable SecureICA encryption to allow RiOS to optimize ICA sessions with SecureICA encryption set to RC5 on both the client-side and the server-side Steelhead appliances.

To enable RiOS to optimize Citrix Sessions with SecureICA encryption set to RC5 SecureICA

1. From the Steelhead Management Console, choose Configure > Optimization > Citrix.
2. Select Enable Citrix Optimization.
3. Specify the ICA port and Session Reliability port.
4. Select Enable SecureICA Encryption.
5. Click **Apply** to apply the settings to the running configuration.
6. Restart the optimization service.

You must perform this configuration on both the client-side and server-side Steelhead appliances.

Citrix Drive-Mapping Optimizations

Client drive mapping (CDM) is a Citrix ICA protocol feature that enables users to access their local drives (such as floppy disk drives, network drives, USB drives, CD-ROM drives, and hard disk drives) from within an ICA session. When you use CDM to access a mapped drive within an ICA session, the end-user desktop experience can be negatively affected by the underlying network performance.

RiOS v7.0 and later supports CDM latency optimization. CDM latency optimization improves the end-user desktop experience when reading and writing files on a mapped drive in the following ways:

- **Reading a file (client-to-server transfer)** - The Steelhead appliances monitor the CDM virtual channel for files being read from the client drive. When the client-side Steelhead appliance detects a chunk of file data being requested, it begins to read ahead and sends the file data in larger chunks to the server-side Steelhead appliance. The server-side Steelhead appliance buffers the file data until it is requested. This effectively eliminates many round trips across the WAN and improves the user desktop experience.
- **Writing a file (server-to-client transfer)** -The Steelhead appliances monitor the CDM virtual channel for files written to the client drive. Whenever the server sends file data to the client, the server-side Steelhead appliance immediately sends an acknowledgment to the server so that the server continues to send file data quickly. Without this local acknowledgment, the server does not send any more data until it receives the acknowledgment from the client. This optimization eliminates round trips across the WAN, which improves the user desktop experience.

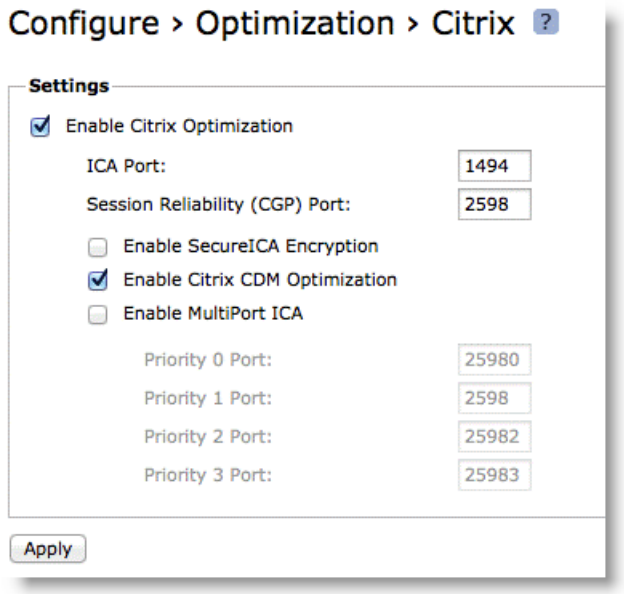
CDM latency optimization is available when both the client-side and server-side Steelhead appliances are running RiOS v7.0 or later. You cannot apply CDM latency optimization to Citrix sessions that use Citrix session reliability.

Note: File transfers greater than 1 GB do not perform as well as smaller file transfers.

To configure Citrix CDM latency optimization

1. Choose Configure > Optimization > Citrix.
2. Select Enable Citrix CDM Optimization.

Figure 5-3. Citrix Settings



3. Click **Apply** to apply the settings to the running configuration.
4. Restart the optimization service.

To configure CDM with the CLI, use the following commands:

```
(config)#protocol citrix cdm enable
(config)#service restart
```

You must perform this configuration on both the client-side and server-side Steelhead appliances.

Citrix Multi-Stream ICA Traffic Optimization with Steelhead Appliances

This section requires that you be familiar with the Citrix ICA protocol and how to configure your Citrix server. This section includes the following topics:

- "Citrix Virtual Channels and Traffic Priorities" on page 100
- "Single-Stream and Multi-Stream ICA" on page 100

Citrix Virtual Channels and Traffic Priorities

The ICA traffic within a Citrix session comprises of many categories of traffic called *virtual channels*. A virtual channel provides a specific function of Citrix ICA remote computing architecture, such as print, client drive mapping, audio, video, and so on. The ICA traffic within a Citrix session is also categorized by priority, in which virtual channels carrying real-time traffic, such as audio and video, are flagged with higher priority than virtual channels carrying bulk transfer traffic such as print and client drive mapping. The ICA priority groups are as follows:

- Very High (priority 0)
- High (priority 1)
- Medium (priority 2)
- Low (priority 3)

For more information about Citrix ICA virtual channel, go to the Citrix Knowledge Base article CTX116890 at http://support.citrix.com/article/CTX116890.

Single-Stream and Multi-Stream ICA

By default, a Citrix session uses a single TCP connection to carry traffic for all virtual channels and all priority groups. This is referred to as *single-stream ICA*.

XenApp 6.5 and XenDesktop 5.5 introduced a new feature that enabled use of multiple TCP connections to carry traffic for a Citrix session. This is called *multi-stream ICA*. Multi-stream ICA is available only when you enable Citrix session reliability. Multi-stream ICA carries traffic on port 2598 and three other user-configurable ports as defined in the multi-port Citrix computer policy. Each port represents an ICA priority group and enables you to apply true network-based QoS policies to the priority groups for the virtual channel traffic that they carry.

For more information about enhanced QoS with multi-stream ICA, go to http://blogs.citrix.com/2011/08/25/enhanced-qos-via-multi-stream-ica/.

This table shows the association of Citrix ICA priority groups to the various ICA virtual channels for multi-stream ICA.

Value	Priority	Description
0	Very high	Audio
1	High	Thin Wire/DX command remoting, seamless, MSFT TS licensing; SmartCard redirection; control virtual channel; and end-user experience monitoring
2	Medium	MediaStream (Windows media and Flash), USB redirection, clipboard, and client drive mapping
3	Low	Printing, client COM port mapping, LPT port mapping, and legacy OEM virtual channels

You must use these software components to enable and optimize multi-stream and multi-port ICA:

- RiOS v7.0 or later
- Citrix XenApp 6.5 or later
- Citrix XenDesktop 5.5 or later
- Citrix Receiver 3.x (online plug-in 13.0) or later

In addition, you must enable Citrix session reliability. Citrix session reliability is enabled by default in XenApp 6.5 and XenDesktop 5.5.

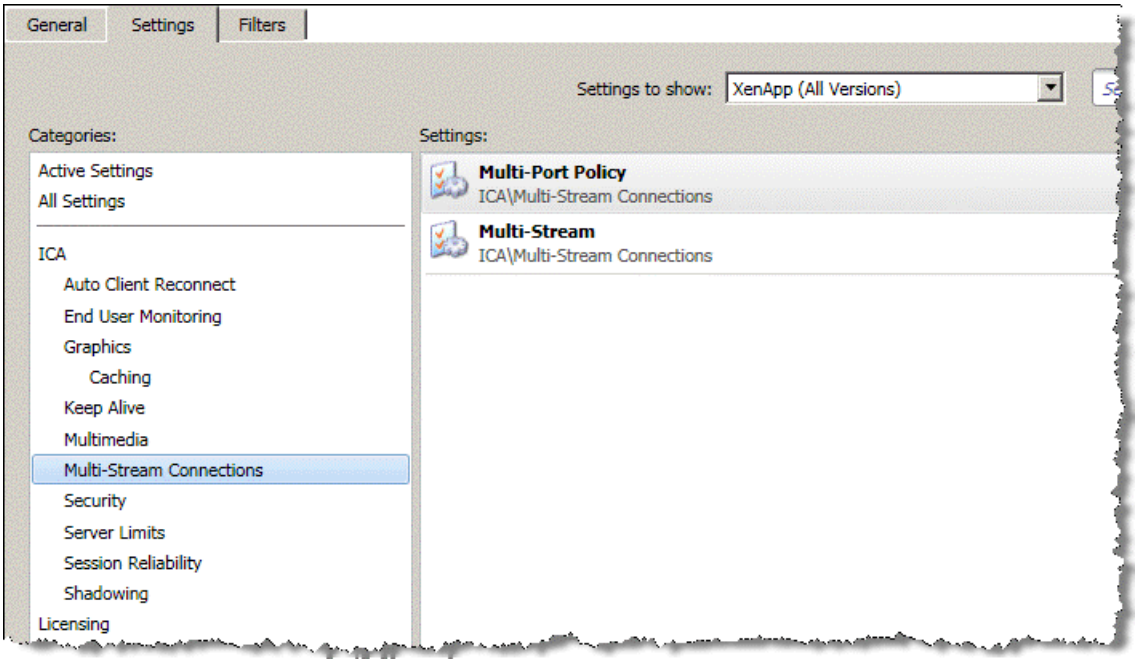
For more information about Citrix session reliability, go to http://support.citrix.com/article/CTX104147.

The following example shows how to configure multi-stream and multi-port ICA in XenApp 6.5.

To configure XenApp 6.5 multi-stream and multi-port ICA traffic optimization with a Steelhead appliance

1. From the Citrix AppCenter or Windows Group Policy Editor, enable multi-stream ICA in the Citrix computer policy.

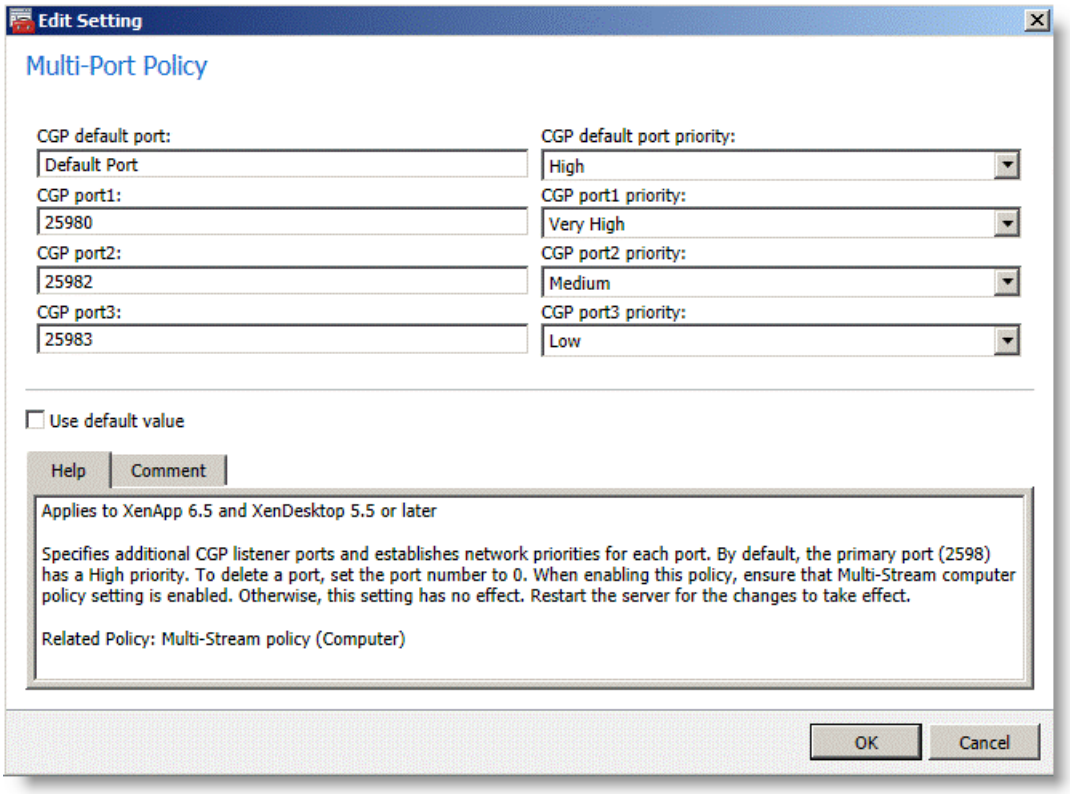
Figure 5-4. Citrix Computer Policy for Multi-Stream and Multi-Port ICA



2. Configure the multi-port policy in the Citrix computer policy.

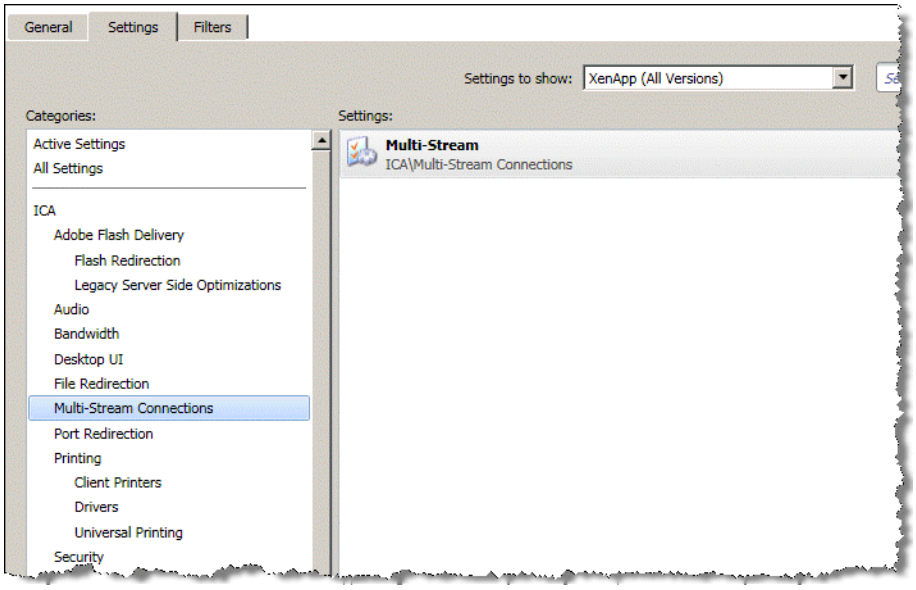
You cannot change the default port assigned to high priority. The default is mapped to the Citrix CGP port 2598 for session reliability.

Figure 5-5. Citrix Computer Policy for Multi-Port ICA



3. Enable multi-stream ICA in the Citrix user policy.

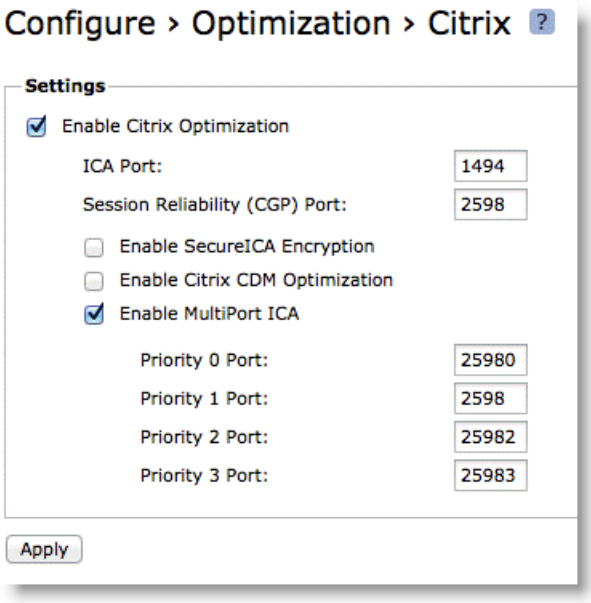
Figure 5-6. Citrix User Policy for Multi-Stream ICA



4. Restart the XenApp Server to apply the Citrix computer policy. Log in the user once more to apply the Citrix user policy.

5. From the Steelhead Management Console, choose the Configure > Optimization > Citrix page and specify the ports according to what is configured in the multi-port policy.

Figure 5-7. Configure the Ports for Multi-Port ICA



6. Click **Apply** to apply the settings to the running configuration.

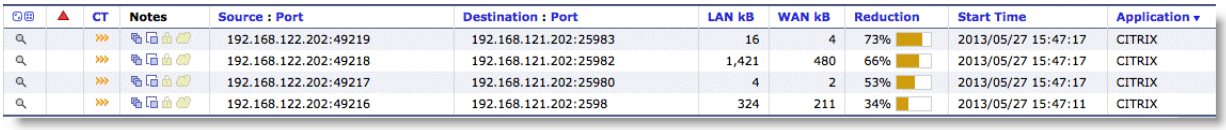
7. Restart the optimization service.

   You must repeat this configuration on both the client-side and the server-side Steelhead appliance.

8. Log in to the XenApp server and launch a published application.

   From the Current Connections page in the Steelhead Management Console, you see four unique TCP connections for the Citrix session, according to the TCP ports that you have configured in the multi-port policy.

Figure 5-8. Current Connections Report



CT	Notes	Source : Port	Destination : Port	LAN kB	WAN kB	Reduction	Start Time	Application
»»		192.168.122.202:49219	192.168.121.202:25983	16	4	73%	2013/05/27 15:47:17	CITRIX
»»		192.168.122.202:49218	192.168.121.202:25982	1,421	480	66%	2013/05/27 15:47:17	CITRIX
»»		192.168.122.202:49217	192.168.121.202:25980	4	2	53%	2013/05/27 15:47:17	CITRIX
»»		192.168.122.202:49216	192.168.121.202:2598	324	211	34%	2013/05/27 15:47:11	CITRIX

You do not need to change your current QoS configuration if you already have rules to prioritize Citrix ICA traffic on port 2598. However, you must configure additional QoS rules if you want to prioritize Citrix ICA traffic on the user-configurable ports that you have configured in the multi-port Citrix computer policy.

QoS Classification for Citrix Traffic

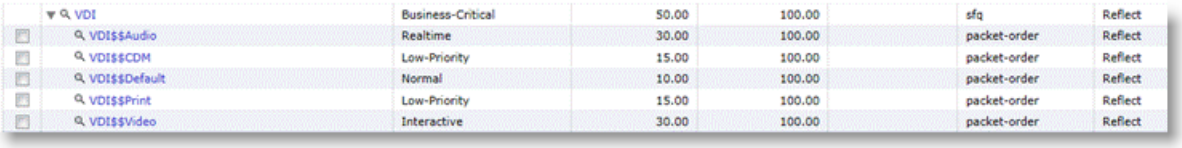
RiOS v6.0 and later enables you to classify Citrix traffic using QoS to prioritize the delivery of Citrix ICA traffic according to the ICA priority groups. QoS classification for Citrix traffic is beneficial in mixed-use environments in which Citrix users perform printing and use client drive-mapping features. Using QoS to classify Citrix traffic in a mixed-use environment improves the desktop computing experience for end users.

These RiOS capabilities ensure optimal delivery of Citrix traffic over the network:

- **Latency priority** - The ICA traffic priority affects traffic latency. For example, latency priority enables you to assign interactive traffic a higher priority than print or client drive-mapping traffic. A typical priority setting for interactive Citrix sessions, such as screen updates, is real-time or interactive. Remember that priority is relative to other classes in your QoS configuration.
- **Bandwidth allocation (also referred to as *traffic shaping*)** - When configuring QoS for Citrix traffic, you must allocate the correct amount of bandwidth for each QoS traffic class. The amount you specify reserves a predetermined amount of bandwidth for each traffic class. Bandwidth allocation is important to ensure that a given class of traffic cannot consume more bandwidth than it is allowed. You must also ensure that a given class of traffic has a minimum amount of bandwidth available for delivery of data through the network.
- **Packet-order queue** -The packet-order queue protects the TCP stream order by monitoring flows that are currently inside the packet-shaping infrastructure. The ordering within the TCP stream is preserved. Use this queue for Citrix traffic, because if part of the Citrix session falls into a non-packet ordered class, the Citrix TCP stream might be rearranged, affecting performance and the ability to classify the remaining packets in the stream.

You can start the QoS configuration for Citrix ICA traffic by creating four QoS traffic classes under a single parent traffic class. Each QoS traffic class represents an ICA priority group. If you do not configure the multi-port policy, you must create an additional default traffic class to catch any Citrix ICA traffic that cannot be classified through the priority group. Each traffic class created for Citrix ICA traffic is set to use the packet-order queue, except for the parent class. Figure 5-9 shows a QoS class configuration for Citrix traffic.

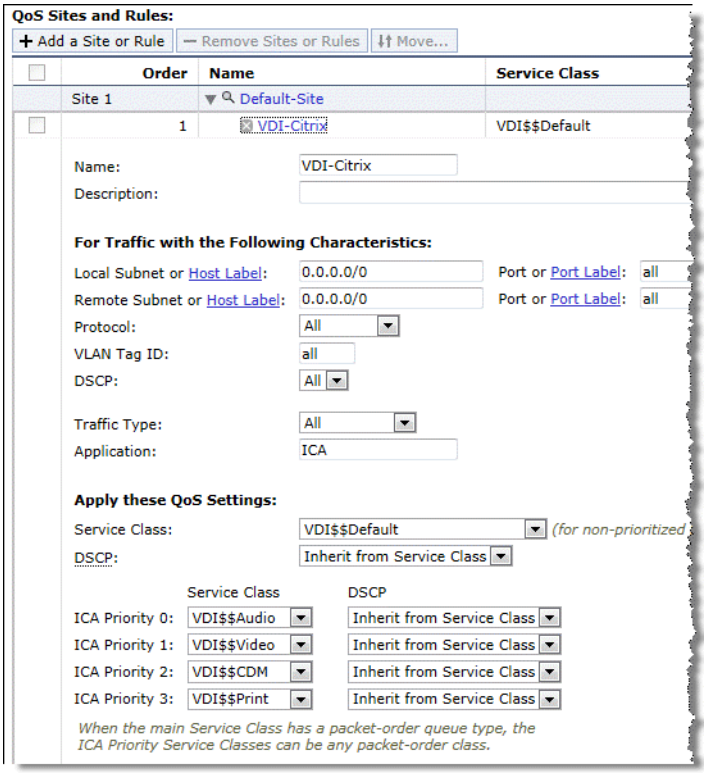
Figure 5-9. Packet-Order Queue



VDI	Business-Critical	50.00	100.00		sfq	Reflect
VDI$$Audio	Realtime	30.00	100.00		packet-order	Reflect
VDI$$CDM	Low-Priority	15.00	100.00		packet-order	Reflect
VDI$$Default	Normal	10.00	100.00		packet-order	Reflect
VDI$$Print	Low-Priority	15.00	100.00		packet-order	Reflect
VDI$$Video	Interactive	30.00	100.00		packet-order	Reflect

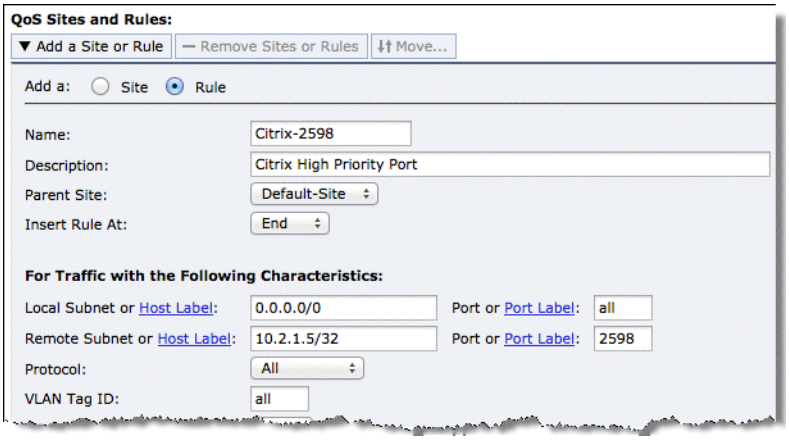
The QoS rule configuration for Citrix ICA traffic is dependent on whether you have configured the multi-port policy. If you have not configured the multi-port policy, you can use the Application Flow Engine (AFE) to identify and classify Citrix ICA traffic according to the ICA priority groups. Figure 5-10 shows a QoS rule configuration using AFE to identify and classify Citrix traffic.

Figure 5-10. QoS Rule Configuration Using AFE



If the multi-port policy is configured, the packet header information can be used to identify and classify Citrix ICA traffic according to the TCP ports configured within the multi-port policy. Figure 5-11 shows a QoS rule configuration using packet header information to identify and classify Citrix ICA traffic.

Figure 5-11. QoS Rule Using Packet Header Information



For detailed configuration examples of Citrix traffic and QoS, see the *Steelhead Appliance Deployment Guide*.

Reduction for Citrix Small Packet Real-Time Traffic

Riverbed recommends that you enable enhanced data reduction for real-time Citrix traffic that is sent in small packets, such as key strokes, mouse clicks, and other Citrix packets that are less than 64 bytes.

Optimization for small Citrix packets is disabled by default.

To enable or disable Citrix optimization for small packets, use the following command:

```
[no] protocol citrix smallpkts enable
```

To check whether or not optimization for small Citrix packets is enabled, use the following command:

```
show protocol citrix smallpkts
```

You must perform this configuration on both the client-side and server-side Steelhead appliances.

Citrix ICA Optimization Over SSL

Citrix Access Gateway (CAG) is an appliance that provides secure remote access to users of XenApp and XenDesktop over SSL VPN. CAG is also known as Access Gateway Enterprise Edition (AGEE) and Netscaler Gateway. CAG proxies the Citrix ICA traffic delivered from these applications and passes them securely over HTTPS or SSL to the end user.

Note: For more details on SSL, see "Configuring SSL on Steelhead Appliances" on page 146.

Figure 5-12. ICA Client Communication Through a Citrix Access Gateway

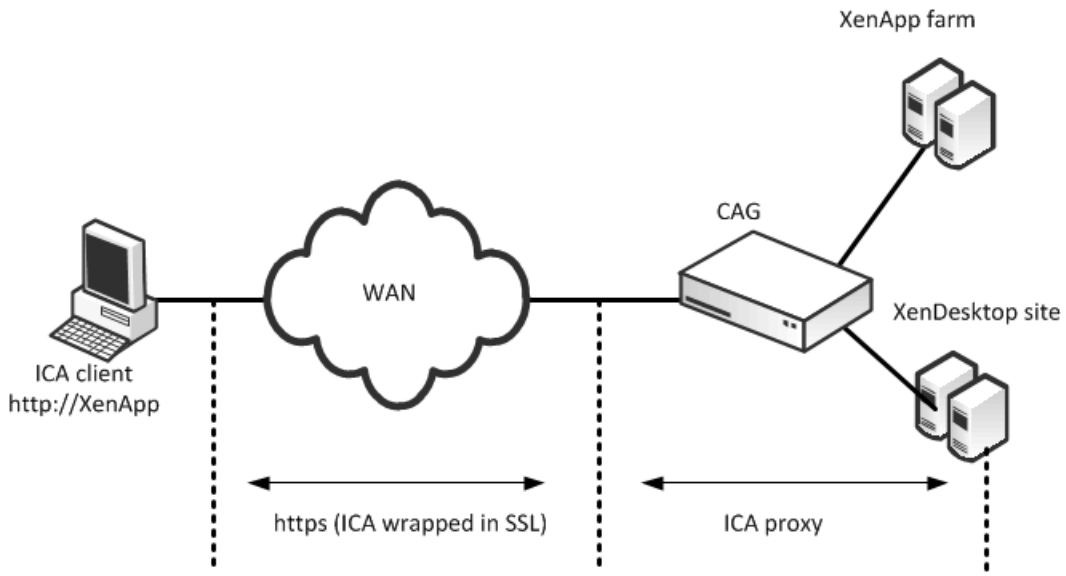


Figure 5-12 shows a CAG deployment. The user reaches the login page by entering the XenApp or XenDesktop secure remote access URL (https://<CAG URL>) in a browser. This page is hosted on the CAG. The user enters their credentials for authentication.

Upon a successful authentication, a list of published applications and desktops is displayed. When the user accesses these applications and desktops, an ICA connection is launched from the user desktop to the XenApp and XenDesktop server. The CAG functions as a gateway to intercept and proxy the user ICA connection to the XenApp and XenDesktop servers on one end, while providing secure remote access over SSL VPN to the user on the other end.

RiOS v7.0 and later can optimize ICA traffic wrapped in SSL using an SSL preoptimization policy to the in-path rule. The in-path rule has several parameters that allow for the chaining of multiple optimization features.

Figure 5-13 shows a Steelhead appliance deployment with CAG.

Figure 5-13. Steelhead Appliance Deployment with CAG

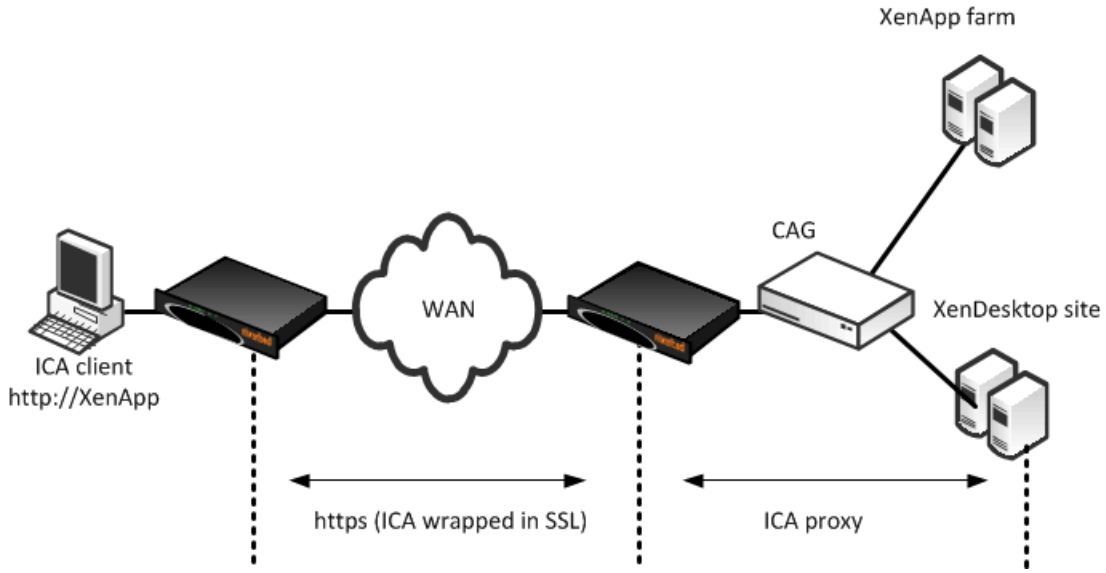
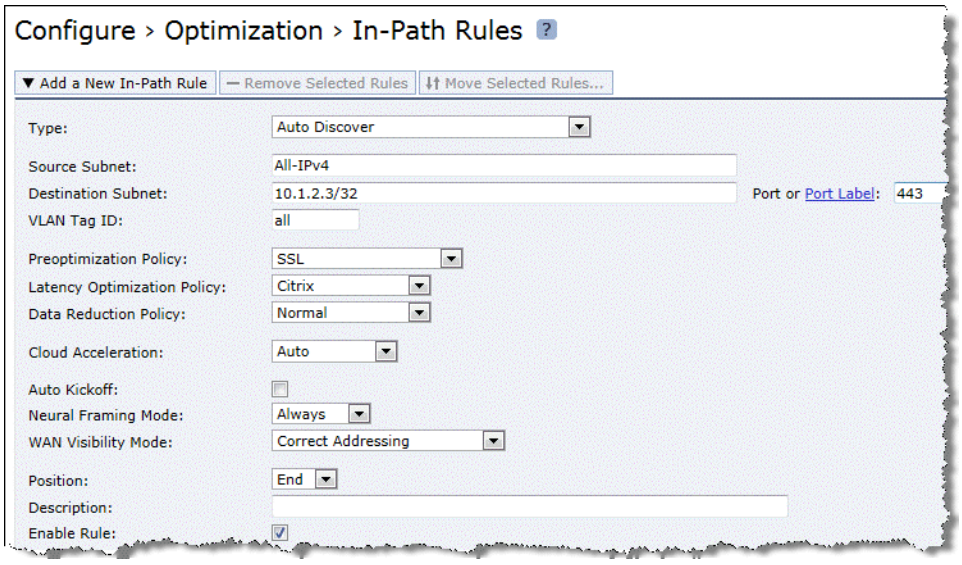


Figure 5-14 shows an in-path rule configuration to optimize Citrix ICA traffic wrapped in SSL.

Figure 5-14. ICA Traffic Wrapped in SSL



Citrix ICA traffic optimization with CAG requires the following:

- Both the client-side and server-side Steelhead appliance must have RiOS v7.0 or later.
- The certificate you use on the Steelhead appliance must be a valid certificate.

  You must import self-signed certificates to the client certificate store. The Citrix client does not connect if you use an invalid certificate.

CHAPTER 6 SMTP-Over-TLS Optimization

In RiOS v7.0 or later, you can securely communicate over the Internet with Simple Mail Transfer Protocol (SMTP) over transport layer security (TLS). This chapter includes the following sections:

- "Configuring Microsoft Exchange Servers for SMTP Over TLS" on page 110
- "Configuring the Steelhead Appliance for TLS Over SSL" on page 112

SMTP is the standard for email transport across the Internet and the standard communication method for Mircosoft Exchange hub servers in Exchange 2007 through to Exchange 2013. Prior to RiOS v7.0, SMTP over TLS or SMTP/S was sent through the Steelhead appliance as a pass-through connection. Because the SMTP session was set up prior to the SSL session, you could not determine when to start SSL optimizations.

In RiOS v7.0 or later, you can determine the start of a TLS connection and the early finish that enables TLS to terminate without disconnecting the SMTP session. This functionality provides more efficient use of the protocol, because it does not perform a handshake on every TLS email sent.

For more details on SSL, see "SSL Deployments" on page 141. For more information about TLS, see "Mid-Session SSL Support" on page 166.

Figure 6-1. SMTP Connection

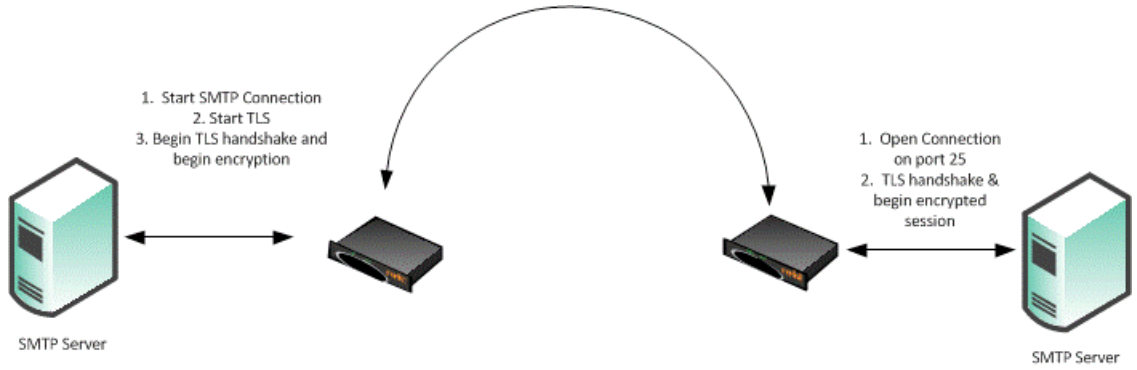


Figure 6-1 shows the START-TLS command issued after the initial SMTP connection is established. Prior to RiOS v7.0, there was no way to determine that an encrypted session had started, so traffic optimization results on the port were low. It was common to enter a pass-through rule for this connection to prevent unnecessary use of RiOS resources.

In RiOS v7.0 or later, SMTP over TLS enables the Steelhead appliance to intercept the START-TLS message after it is issued, which enables the Steelhead appliance to optimize native SSL traffic.

Figure 6-2. SMTP Connection with SMTP Over TLS

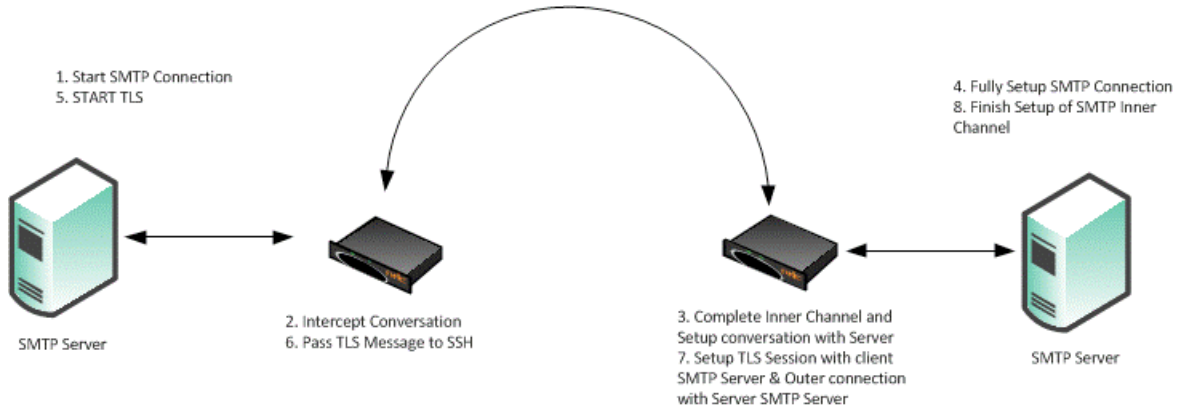


Figure 6-2 shows that an unencrypted SMTP session is intercepted and optimized through the Steelhead appliance. When the START-TLS is issued, RiOS recognizes that a TLS session is imminent and begins the necessary operations to optimize the SSL session. For more details on SSL optimization, see "SSL Deployments" on page 141.

Configuring Microsoft Exchange Servers for SMTP Over TLS

Microsoft Exchange hub servers have additional authentication mechanisms beyond TLS. Two specific environments require additional configuration on the Exchange hub server to allow the SMTP-over-TLS optimization to function properly:

- **Single domain multiple hub servers** - All Exchange hub servers are within the same domain. The servers are inherently trusted, no specific SSL client authorization occurs, and the Exchange hub server uses a null client certificate.
- **Multiple domain multiple hub servers** - Exchange hub servers are in untrusted domains and must be trusted through the Microsoft **set-sendconnector** command. Complete the following steps on each Exchange hub server:

  ```
  set-sendconnector -Identity "<send_conn_name>" -DomainSecurityEnabled $True
  set-recieveconnector -Identity "<receive_conn_name>" -DomainSecurity Enabled $True
  ```

  Next, set the secure domains to secure traffic between one another:

  ```
  set-transportconfig -TLSReceiveDomainSecureList {<remote_domain>}
  set-transportconfig -TLSSendDomainSecureList {<remote_domain>}
  ```

  Note: Some Exchange deployments require a domain trust before mutual authentication will work. Adding a two-way forest trust, preferably with forest-wide authentication, solves this issue.

Configuration with Microsoft Exchange Hub servers requires the following prerequisites:

- You must have an SSL license installed on all Steelhead appliances participating in SSL or TLS optimization.
- The client-side and server-side Steelhead appliances must have RiOS v7.0 or later.
- You must generate and import new certificates in the Exchange hub servers.

An Exchange hub server does not allow its private key to be exported. You cannot retrieve either the certificate or the private key from the hub server. For details regarding receive connectors and certificates in an Exchange hub server environment, go to the following Web sites:

- http://technet.microsoft.com/en-us/library/aa996395.aspx
- http://technet.microsoft.com/en-us/library/aa998327.aspx

You can generate the new certificates with the tool listed in Microsoft Technote 998327 or with OPENSSL. This is an example with OPENSSL:

```
linux#openssl req -x509 -newkey 1024 -keyout my.pem -out my.cert -nodes
linux#openssl pkcs12 -export -inkey my.pem -in my.cert -out my.pfx
```

After you create the new self-signed certificates, you must install them on the remote and local Exchange server—which runs the hub server role—as defined next.

To install a new certificate on the remote and local Exchange server

1. Import the PFX file to the local Exchange server's personal certificate store.
2. Add the SMTP role to the certificate.

   Exchange uses roles to identify which certificates to use in various situations. You need to *tell* the Exchange server that your certificate for encrypted traffic over SMTP.
3. If the Exchange management console (EMC) is not already open, in the search field of the Windows Start menu, search for Exchange management console and open the application.
4. In the left navigation pane, toggle open Microsoft Exchange On-Premises (your server).
5. Click **Server Configuration** and wait for the Exchange certificates in the work (that is, the middle) pane to populate.
6. Right-click the self-signed certificate and select Assign Services to Certificate... This opens the Assign Services to Certificate wizard.

   Note: You can identify your certificate by viewing the Subject and Issuer fields. Your ticket is probably the only one listed that does not have a customized name in the Name field.
7. Select Simple Mail Transfer Protocol (SMTP).
8. Click **Assign**.
9. After the assignment completes successfully, click **Finish**.

   Note: If you are prompted to overwrite the existing SMTP certificate, select Yes.
10. Add the partner's permission group to the local Exchange server's default receive connector.
11. If the EMC is not already open, in the search field of the Windows Start menu, search for exchange management console and open the application.
12. In the left navigation pane, toggle open the Microsoft Exchange On-Premises (your server).

13. Toggle open the Server Configuration.
14. Click **Hub Transport** and wait for the receive connectors in the work pane (that is, the middle) to populate.
15. Click the default receive connection (usually named something similar to Default mach101).
16. In the Actions pane, under the section with the default receive connector's name, select Properties. This opens the default receive connector's properties dialog box.
17. Select the Permission Groups tab.
18. Select Partners.

Note: Deselect Anonymous users if it is selected.

---

19. Click **OK** to accept the changes to the default receive connector's properties.
20. Import the CERT file to the remote Exchange server's Computer Trusted Root Certificate Authorities.

Configuring the Steelhead Appliance for TLS Over SSL

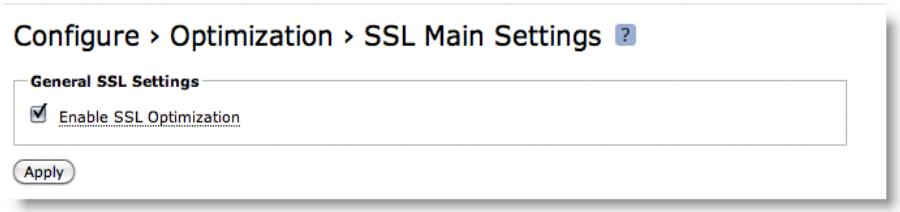
There are two steps to configure the Steelhead appliance to use SMTP over TLS:

- Configure SSL.
- Configure an in-path rule.

To configure the Steelhead appliance for use with SMTP over TLS

1. Enable SSL in on the Steelhead appliance.
   - In the Management Console, choose Configure > Optimization > SSL Main, and select Enable SSL Optimization, or use the CLI command **sh(config)#protocol ssl enable**.

Figure 6-3. SSL Main Settings Page



- In the Management Console, choose Configure > Optimization > Advanced Settings and select Enable Client Certificate Support and Enable Midsession SSL, or use the CLI commands **sh#(config)protocol ssl mid-session-ssl** and **sh#(config)protocol ssl client-cer-auth enable**.

Figure 6-4. SSL Advanced Settings Page



2. Configure an in-path rule.

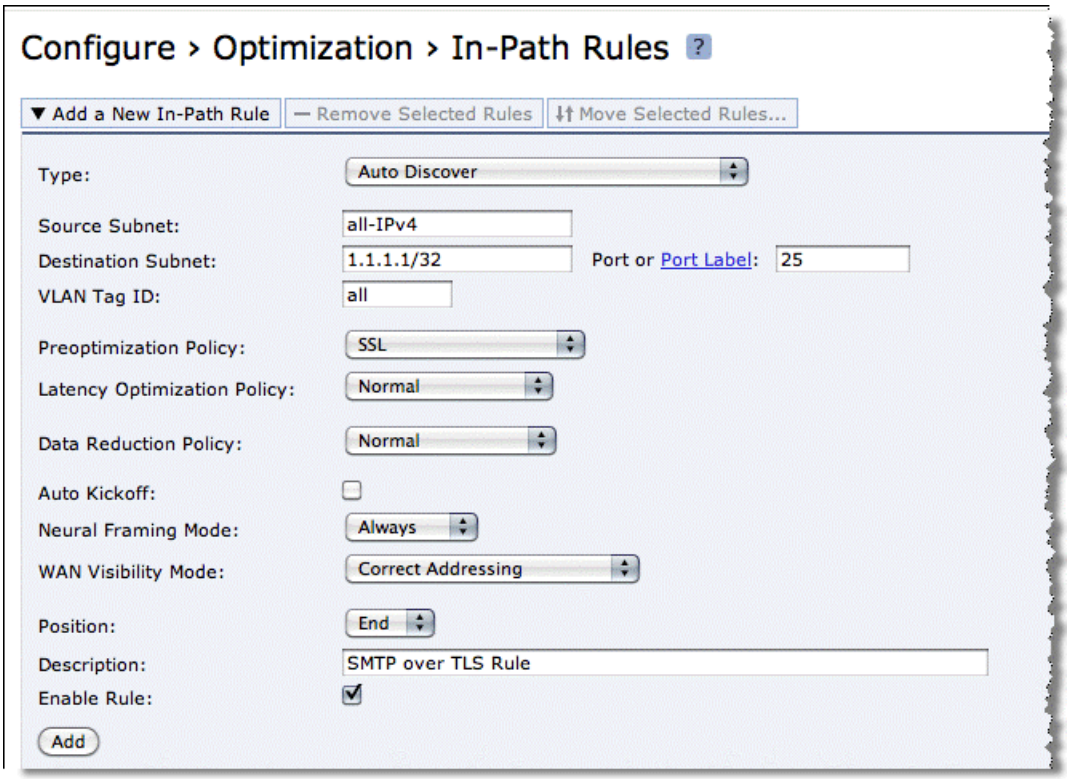
   The in-path rule applies the SSL preoptimization policy to the SMTP-over-TLS traffic.

   - In the Management Console, choose Configure > Optimization > In-Path Rules.

- Select Add New In-Path Rule, enter the following parameters, and click **Add**.

Rule Setting	Value
Type	Auto Discover
Source Subnet	all-IPv4
Destination Subnet	IP address of the SMTP-over-TLS server
Port or Port Labels	SMTP port, typically 25
VLAN tag ID	all
Preoptimization Policy	SSL
Latency Reduction Policy	Normal
Data Reduction Policy	Normal
Auto Kickoff	Off
Neural Framing	Always
WAN Visibility	Correct addressing, port transparency, or full transparency are all valid configuration options.
Position	Prior to any negating rules
Description	a description of the rules
Enable Rules	Yes

Figure 6-5. In-Path Rules Page



- Or, you can use the CLI command **sh(config)# in-path rule auto-discover rulenum 4 srcaddr all-ipv4 dstaddr 1.1.1.1/32 dstport 25 preoptimization ssl**.

CHAPTER 7

FTP Optimization

This chapter discusses File Transfer Protocol (FTP) behavior and configuration on the Steelhead appliances, including how to configure in-path and QoS rules to accomplish specific functions. This chapter includes the following sections:

- "Overview of FTP" on page 115
- "Configuring In-Path Rules" on page 117
- "QoS Classification for the FTP Data Channel" on page 118
- "FTP Optimization Considerations" on page 119
- "Steelhead Mobile FTP Considerations" on page 120

FTP is a standard network protocol used to copy a file from one host to another over a TCP/IP-based network. FTP is built on a client/server architecture and uses separate control and data connections between the client and server.

By default, FTP optimization is enabled on all FTP connections.

Overview of FTP

The FTP protocol consists of two connections: the control connection and the data connection. A client initiates the control connection to the server on TCP port 21. This connection remains open for the duration of the session and sends administrative data (for example, commands, identification, and passwords). After the control connection is established, the data connection transfers the file data.

The data connection uses various originators and ports. When you look at TCP connections and Steelhead appliance optimization, both the control and data connections appear as separate TCP connections. The control connection always appears to go from the FTP client on a random source port to the FTP server on port 21. The data connection properties change significantly, depending on if active or passive mode is used for the FTP transfer.

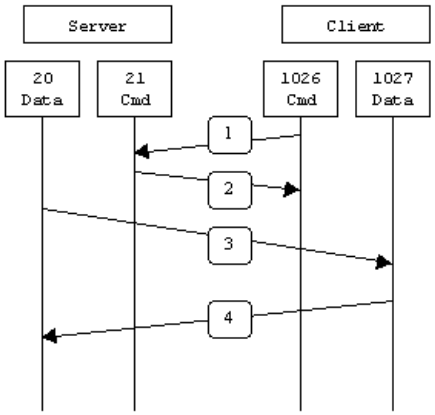
Active Mode

Figure 7-1 shows the control connection and data connection used in FTP active mode:

1. The client from a random TCP source port greater that 1024 (shown here as port 1026) connects to the server on port 21 to establish the control connection. The client sends the server the port to establish the data connection on (shown here as TCP port 1027).
2. The server acknowledges the port number.
3. The server initiates the data connection from TCP port 20 to the client on the specified port in Step 1.
4. The client responds with an ACK to complete the establishment of the data connection.

Most of the time TCP port 21 is associated with FTP control and TCP port 20 with data. TCP port 20 is only true with FTP in active mode.

Figure 7-1. Active Mode



Source: http://slacksite.com/other/ftp.html (Sept 8, 2010)

A potential issue with active mode FTP is the FTP client does not make the actual connection to the data port of the server—it tells the server what port it is listening on and the server connects back to the specified port on the client. From a client-side firewall this appears to be an outside system initiating a connection to an internal client, which can be blocked. For more information, see the *Steelhead Appliance Deployment Guide*.

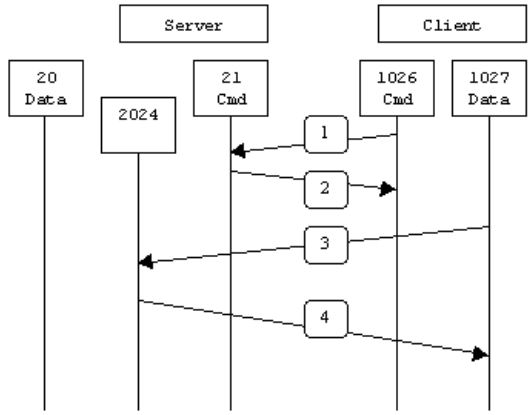
Passive Mode

Figure 7-2 shows the control connection and data connection used in FTP passive mode.

1. The client connects from a random TCP source port to the server on port 21. The client requests the server to provide a port that the client can connect to for the data connection.
2. The server replies with the specified port for the data channel (shown here as TCP port 2024).
3. The client initiates the data connection from a random port to the specified server data port.
4. The server sends an ACK to the client.

With passive FTP, both the control and data connections are both originated from the client and that port 20 is not used.

Figure 7-2. Passive Mode



Source: http://slacksite.com/other/ftp.html (Sept 8, 2010)

Passive mode FTP solves many of the problems from the client-side security perspective, although it does require the server to accept a remote connection to a range of high numbered ports. Most clients today support both active and passive mode FTP. The default Windows client does not support passive FTP, along with some Unix versions, such as Solaris.

Many people prefer to use their Web browser as an FTP client. Most browsers only support passive mode when accessing ftp:// URLs.

Configuring In-Path Rules

Steelhead appliances are aware of FTP connections in either active or passive mode, and correlate the data channels with the control channel.

Optimizing FTP

By default, FTP uses the standard optimization rule for in-path rules. If you use manual optimization rules, follow the standard in-path practice, configuring the client-side Steelhead appliance in-path rules based on IPs or TCP port. With the TCP port, it is only necessary you specify destination port 21, which automatically includes optimization for the data channels, regardless of if active or passive mode is used.

Passing through FTP

To pass through FTP, Riverbed recommends creating pass-through rules based on the FTP server IP, on both the client and server-side Steelhead appliances. You base the rules this way because of varying data connection behavior depending on active or passive mode. Because the source of the data connection is the FTP server in active mode, it is necessary to configure the in-path pass-through rules on the server-side Steelhead appliance—this is unlike other protocols.

QoS Classification for the FTP Data Channel

When configuring QoS classification for FTP, the QoS rules differ depending on whether the FTP data channel is using *active* or *passive* FTP. Active versus passive FTP determines whether the FTP client or the FTP server select the port connection for use with the data channel, which has implications for QoS classification.

Active FTP Classification

With active FTP, the FTP client logs in and enters the PORT command, informing the server which port it must use to connect to the client for the FTP data channel. Next, the FTP server initiates the connection towards the client. From a TCP perspective, the server and the client swap roles. The FTP server becomes the client because it sends the SYN packet, and the FTP client becomes the server because it receives the SYN packet.

Although not defined in the RFC, most FTP servers use source port 20 for the active FTP data channel.

For active FTP, configure a QoS rule on the server-side Steelhead appliance to match source port 20. On the client-side Steelhead appliance, configure a QoS rule to match destination port 20.

You can also use AFE to classify active FTP traffic.

Passive FTP Classification

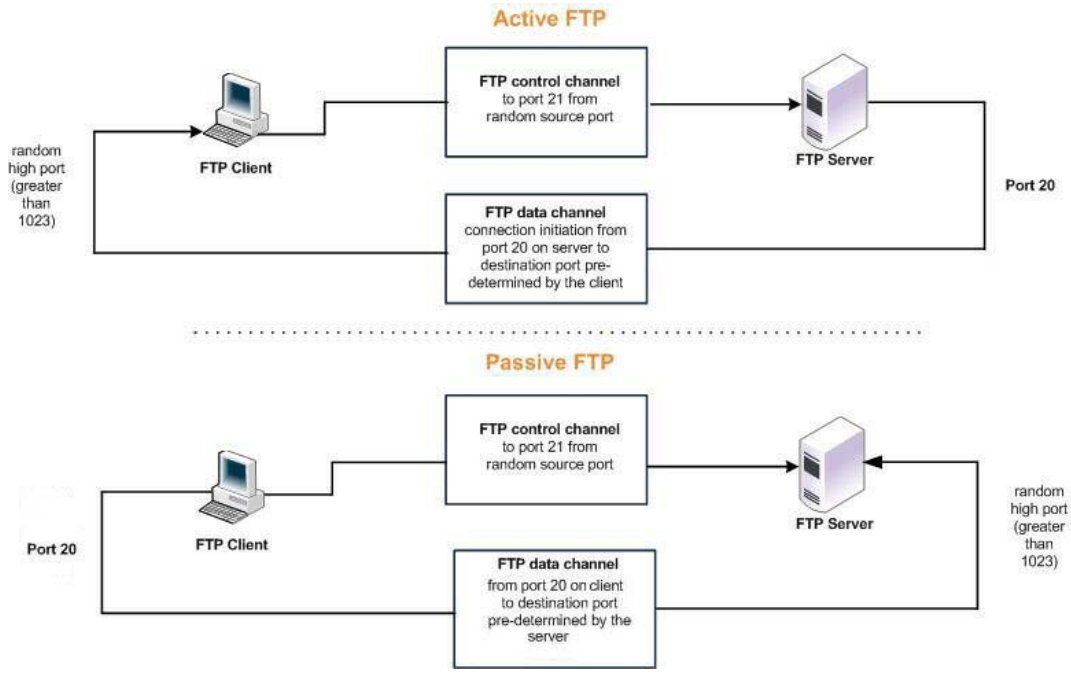
With passive FTP, the FTP client initiates both connections to the server. First, it requests passive mode by entering the PASV command after logging in. Next, it requests a port number for use with the data channel from the FTP server. The server agrees to this mode, selects a random port number, and returns it to the client. Once the client has this information, it initiates a new TCP connection for the data channel to the server-assigned port. Unlike active FTP, there is no role swapping and the FTP client initiates the SYN packet for the data channel.

The FTP client receives a random port number from the FTP server. Because the FTP server cannot return a consistent port number to use with the FTP data channel, RiOS does not support QoS Classification for passive FTP in versions earlier than RiOS v4.1.8, v5.0.6, or v5.5.1. Later RiOS releases support passive FTP and the QoS Classification configuration for passive FTP is the same as active FTP.

When configuring QoS Classification for passive FTP, port 20 on both the server and client-side Steelhead appliances means the port number being used by the data channel for passive FTP, as opposed to the literal meaning of source or destination port 20.

Note: The Steelhead appliance must intercept the FTP control channel (port 21), regardless of whether the FTP data channel is using active or passive FTP.

Figure 7-3. Active and Passive FTP



With RiOS v8.0.4 and later the Riverbed application flow engine (AFE) monitors the FTP control connection. AFE learns the negotiated port numbers and connection initiator from the FTP control connection. You can then use AFE to classify active and passive FTP connections for IPv4.

The AFE is unable to classify FTP correctly in a server-side out-of-path (SSOOP) Steelhead appliance deployment with the exception of optimized FTP in active mode.

Note: FTP with IPv6 is currently not supported in AFE and QoS.

For more information about QoS and AFE, see the *Steelhead Appliance Deployment Guide* and the *Steelhead Appliance Management Console User's Guide*.

FTP Optimization Considerations

Although the FTP data and control connections are separate TCP connections, the two channels are correlated. For example, the sender requests to close the data connection over the control channel when a file has finished transmitting.

With WAN optimization in place, the sender can incorrectly believe that the file transfer has completed, and send a request over the control channel to prematurely close the data connection. Because the Steelhead appliances act as a TCP proxy, the sender is actually communicating with the local Steelhead appliance. In certain environments with high latency or low data reduction, the local Steelhead appliance can still be sending the file, but the sender sends the request to close the data connection before the file is completely finished.

If you find that the data connection is closing prematurely, you can adjust a manual delay to the sender request. Use the command **protocol ftp delay-close-sec <time in seconds>** on the Steelhead appliance local to the FTP server. Riverbed recommends 30 seconds for most environments.

Steelhead Mobile FTP Considerations

Steelhead Mobile only optimizes connections originating from the Mobile Client. With active FTP, the data connection is opened from the server to the client. Steelhead Mobile does not optimize active FTP because the data connection originates from the server. Although the majority of FTP client software supports passive mode, ensure your client is capable and configured correctly to use passive FTP when using Steelhead Mobile.

For more information about Steelhead Mobile, see the *Steelhead Appliance Deployment Guide*.

CHAPTER 8 Other Protocol Optimization

In addition to the protocols previously discussed, this chapter describes the basic steps for configuring Steelhead appliance protocol optimization for the following protocols:

- "Oracle Forms Optimization" on page 121
- "NFS Optimization" on page 122
- "Lotus Notes Optimization" on page 124

Oracle Forms Optimization

You can display and modify Oracle Forms optimization settings in the Configure > Optimization > Oracle Forms page.

Oracle Forms is a platform for developing user interface applications to interact with an Oracle database. It uses a Java applet to interact with the database in either native, HTTP, or HTTPS mode. The Steelhead appliance decrypts, optimizes, and then re-encrypts the Oracle Forms traffic.

You can configure Oracle Forms optimization in these modes:

- **Native** - The Java applet communicates with the backend server, typically over port 9000. Native mode is also known as socket mode.
- **HTTP** - The Java applet tunnels the traffic to the Oracle Forms server over HTTP, typically over port 8000.
- **HTTPS** - The Java applet tunnels the traffic to the Oracle Forms server over HTTPS, typically over port 443. HTTPS mode is also known as SSL mode.

Use Oracle Forms optimization to improve Oracle Forms traffic performance. RiOS v5.5.x and later supports 6i, which comes with Oracle Applications 11i. RiOS v6.0 and later supports 10gR2, which comes with Oracle E-Business Suite R12.

This feature does not need a separate license and is enabled by default. However, you must also set an in-path rule to enable this feature.

Note: Optionally, you can enable IPSec encryption to protect Oracle Forms traffic between two Steelhead appliances over the WAN or use the Secure Inner Channel on all traffic.

Determining the Deployment Mode

Before enabling Oracle Forms optimization, you must know the mode in which Oracle Forms is running at your organization.

To determine the Oracle Forms deployment mode

1. Start the Oracle application that uses Oracle Forms.
2. Click a link in the base HTML page to download the Java applet to your browser.
3. On the Windows taskbar, right-click the Java icon (a coffee cup) to access the Java console.
4. Choose Show Console (JInitiator) or Open *<version>* Console (Sun JRE).
5. Locate the "connectMode=" message in the Java Console window. This message indicates the Oracle Forms deployment mode at your organization: for example,

```
connectMode=HTTP, native
connectMode=Socket
connectMode=HTTPS, native
```

For more details on configuring Oracle Forms optimization, see the *Steelhead Appliance Management Console User's Guide*.

NFS Optimization

NFS optimization provides latency optimization improvements for NFS operations by pre-fetching data, storing it on the client Steelhead appliance for a short amount of time, and using it to respond to client requests. You enable NFS optimization in high-latency environments.

You can configure NFS settings globally for all servers and volumes or you can configure NFS settings that are specific to particular servers or volumes. When you configure NFS settings for a server, the settings are applied to all volumes on that server unless you override settings for specific volumes.

Important: NFS optimization is not supported in an out-of-path deployment.

Note: NFS optimization is only supported for NFS v3.

For each Steelhead appliance, you specify a policy for pre-fetching data from NFS servers and volumes. You can set the following policies for NFS servers and volumes:

- **Global Read/Write** - Choose this policy when the data on the NFS server or volume can be accessed from any client, including LAN clients and clients using other file protocols. This policy ensures data consistency but does not allow for the most aggressive data optimization. **Global Read/Write** is the default value.
- **Custom** - Create a custom policy for the NFS server.
- **Read-only** - Any client can read the data on the NFS server or volume but cannot make changes.

After you add a server, the Management Console includes options to configure volume policies.

For detailed information, see the *Steelhead Appliance Management Console User's Guide*.

Implementing NFS Optimization

This section describes the basic steps for using the Management Console to implement NFS. For detailed information, see the *Steelhead Appliance Management Console User's Guide*.

Basic Steps

Perform the following basic steps to configure NFS optimization.

To configure NFS optimized connections

1. Enable NFS in the Configure > Optimization > NFS page.

   Enable NFS on all desired client and server Steelhead appliances.

2. For each client Steelhead appliance, configure NFS settings that apply by default to all NFS servers and volumes. For details, see the *Steelhead Appliance Management Console User's Guide*.

   Configure these settings on all desired client Steelhead appliances. These settings are ignored on server-side Steelhead appliances. If you have enabled NFS optimization (as described in the previous step) on a server-side Steelhead appliance, NFS configuration information for a connection is uploaded from the client-side Steelhead appliance to the server Steelhead appliance when the connection is established.

Important: If NFS is disabled on a server-side Steelhead appliance, the appliance does not perform NFS optimization.

---

3. For each client-side Steelhead appliance, override global NFS settings for a server or volume that you specify. You do not need to configure these settings on server-side Steelhead appliances. If you have enabled NFS optimization on a server-side Steelhead appliance, NFS configuration information for a connection is uploaded from the client-side Steelhead appliance to the server-side Steelhead appliance when the connection is established.

   If you do not override settings for a server or volume, the global NFS settings are used. If you do not configure NFS settings for a volume, the server-specific settings, if configured, are applied to the volume. If server-specific settings are not configured, the global settings are applied to the server and its volumes.

   Note: When you configure a prefetch policy for an NFS volume, you specify the desired volume by an FSID number. An FSID is a number NFS uses to distinguish mount points on the same physical file system. Because two mount points on the same physical file system have the same FSID, more than one volume can have the same FSID.

   For details, see the *Steelhead Appliance Management Console User's Guide*.

4. If you have configured IP aliasing for an NFS server, specify all of the server IP addresses in the Steelhead appliance NFS-protocol settings.

5. View and monitor NFS statistics in the Management Console Reports > Optimization > NFS Statistics page.

Configuring IP Aliasing

If you have configured IP aliasing (multiple IP addresses) for an NFS server, you must specify all of the server IP addresses in the Steelhead appliance NFS protocol settings for NFS optimization to work properly.

To configure IP aliasing on a Steelhead appliance

1. In the Management Console, choose Configure > Optimization > NFS.
2. Click **Add New NFS Server** to expand the page.
3. In the Name box, specify the name of the NFS server.
4. Enter each server IP address in a comma separated list in the Server IP box.
5. Click **Add Server**.

Lotus Notes Optimization

You can enable and modify Lotus Notes optimization settings in the Configure > Optimization > Lotus Notes page.

Lotus Notes is a client/server collaborative application that provides email, instant messaging, calendar, resource, and file sharing. RiOS provides latency and bandwidth optimization for Lotus Notes v6.0 and later traffic across the WAN, accelerating email attachment transfers and server-to-server or client-to-server replications.

RiOS saves bandwidth by automatically disabling socket compression, which makes SDR more effective. It also saves bandwidth by decompressing Huffman-compressed attachments and LZ-compressed attachments when they are sent or received and recompressing them on the other side. This allows SDR to recognize attachments that have previously been sent in other ways (such as over CIFS, HTTP, or other protocols), and also allows SDR to optimize the sending and receiving of attachments that are slightly changed from previous sends and receives.

To use this feature, both the client-side and server-side Steelhead appliances must be running RiOS v5.5.x or later. To enable optimization of encrypted Lotus Notes connections, both the client-side and server-side Steelhead appliances must be running RiOS v7.0.

Enabling Lotus Notes provides latency optimization regardless of the compression type (Huffman, LZ, or none).

Before enabling Lotus Notes optimization, be aware that it automatically disables socket-level compression for connections going through Steelheads that have this feature enabled.

For details on configuring Lotus Notes optimization, see the *Steelhead Appliance Management Console User's Guide*.

Optimizing Encrypted Lotus Notes

You can optimize encrypted Lotus Notes traffic in RiOS v7.0 or later. When you enable the encrypted Lotus Notes feature, traffic between Steelhead appliances is decrypted and the current Lotus Notes protocol optimization (RiOS v5.5) is applied.

Lotus Notes Authentication

The Lotus Notes and the Domino server relies on a the Notes ID file for proper authentication. This file contains information for authentication and encryption between the client and the server in the Lotus Notes and Domino system. The Notes ID file is usually stored on the client. You must have a password to decrypt the ID file and use its contents, but you do not need a password to authenticate with the server (for example, MS-Exchange or other systems).

This section requires that you be familiar with Lotus Notes and Domino servers.

Optimization Architecture

To optimize an encrypted connection between a Notes client and a Domino server, you must import the Domino servers ID file into the server-side Steelhead appliance, because the Steelhead appliances in the path of the connection need to be able to decrypt and reencrypt the sent data. Next, configure the Domino server with a port on which it accepts unencrypted connections. This can either be the standard port or an auxiliary port. Now, when a Notes client connects to the Domino server, the server-side Steelhead appliance forwards the connection to the auxiliary port of the server.

After the connection is authenticated, the server-side Steelhead appliance resets the connection of the Notes client but maintains the unencrypted connection with the Domino server on the auxiliary port. The Notes client now tries to establish a new encrypted connection, which the server-side Steelhead appliance intercepts and handles as if it were the Domino server.

The server-side Steelhead appliance (acting as the Domino server) generates the information necessary to encrypt the connection to the Notes client. The result is a connection that is encrypted between the Notes client and server-side Steelhead appliance but unencrypted between the server-side Steelhead appliance and the Domino server.

Configuring Optimized Encrypted Lotus Notes

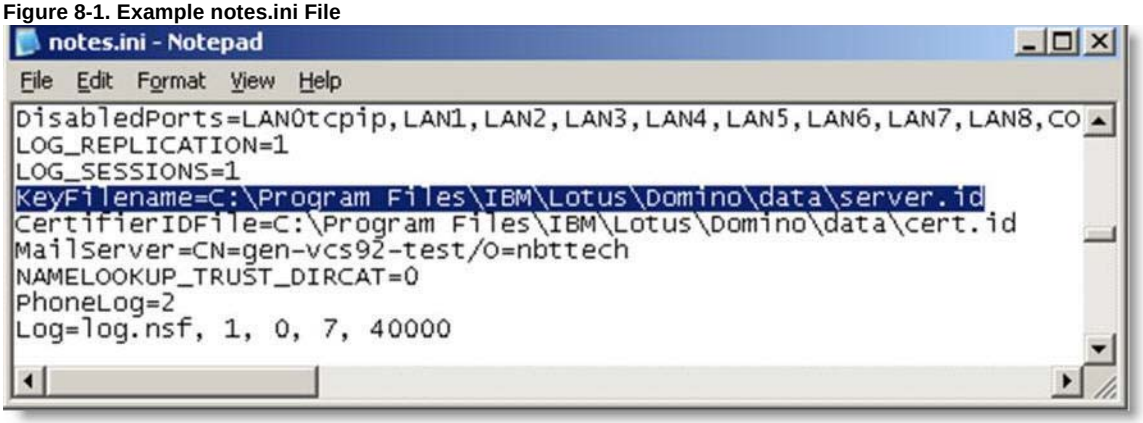
This section describes how to configure optimized encrypted Lotus Notes.

To import the server ID file of Domino servers that require optimization into the server-side Steelhead appliance

1. Log in to the respective Domino servers and identify the location of the server ID file in the notes.ini file. This is usually is located on a Windows server in C:\Program Files\IBM\Lotus\Domino\data.

2. Open the notes.ini file with a text editor.

Figure 8-1. Example notes.ini File



3. Import all server ID files into the server-side Steelhead appliance. You do not need to do this on the client-side Steelhead appliance).
   - From the Management Console, choose Configure > Optimization > Lotus Notes. You can also use the CLI command **protocol notes encrypt import server-id <http, ftp, or scp URL> [password <password>]**.
   - Specify the server ID file URL (or use the Browse button) and password.

   Note: Server IDs might not have passwords.

   - Click **Add**.

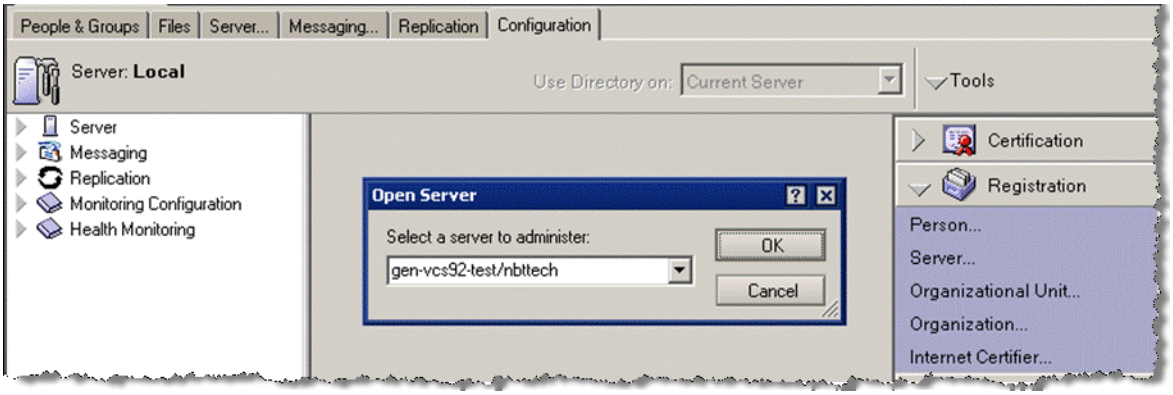
Figure 8-2. Import the Server ID File



To configure Domino servers to accept unencrypted connections on an auxiliary port

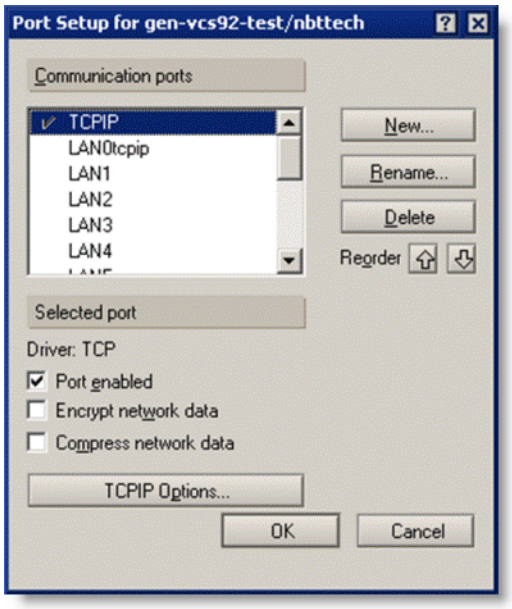
1. Connect to the Domino server.
2. Open the Domino Administrator and connect to the desired Domino server.

Figure 8-3. Domino Server Administrator



3. From the Domino Administrator, choose Configure > Server > Setup Ports to open the Setup Ports page.

Figure 8-4. Setup Ports Page



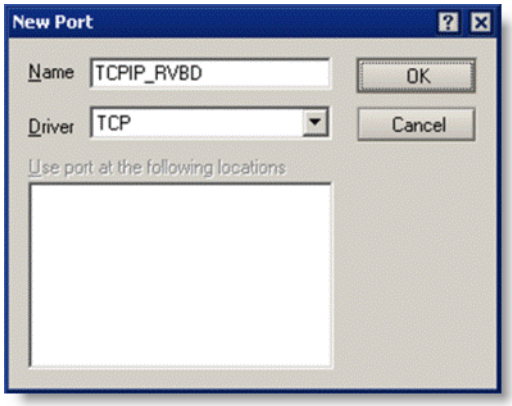
4. In the Setup Ports page, click **New** to create a new port.

5. Name the port.

   This example shows the port named TCPIP_RVBD and has the TCP selected in the Driver drop-down menu.

6. Click **OK**.

Figure 8-5. Creating a new port

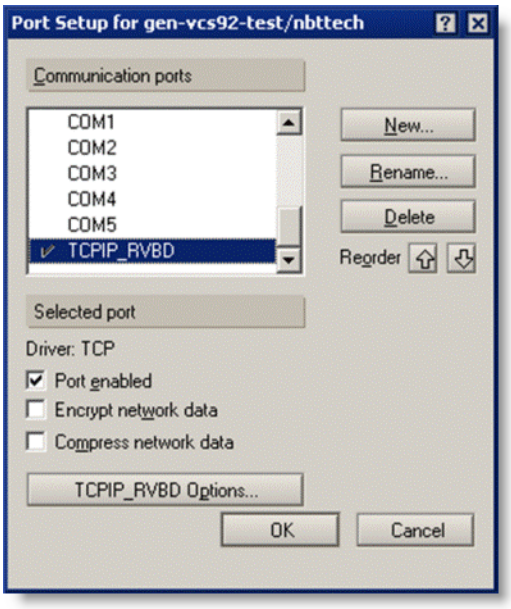


7. To enable the newly created port, select the new port in the Setup Ports page.

   Make sure that Port enabled is selected and Encrypt network data is not selected.

8. Click **OK**.

Figure 8-6. The New Port on the Setup Ports Page



To set a new TCP port number and restart the port

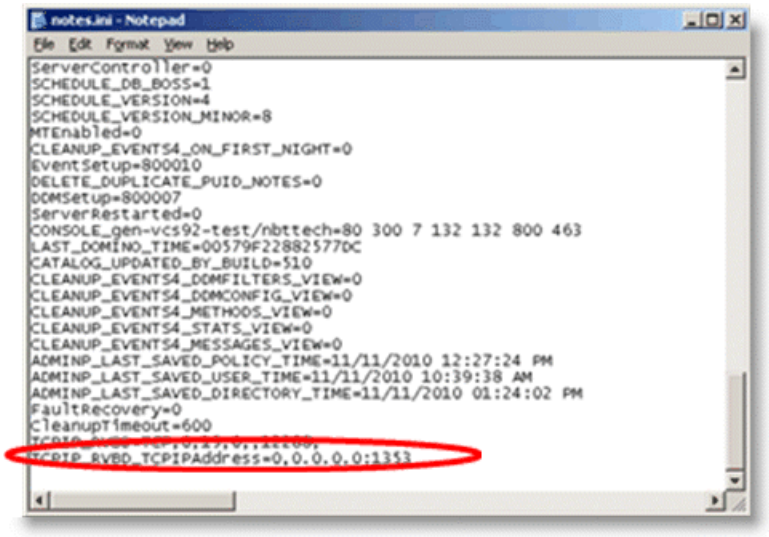
1. Open the Domino server's notes.ini file.

   The file is usually located in C:\Program Files\IBM\Lotus\Domino.

2. Add a line using the format <port_name>_TCPIPAddress=0,<IP_address>:<port>.

   Use the IP address 0.0.0.0; Domino listens on all server IP addresses.

Figure 8-7. Line Added to the notes.ini File

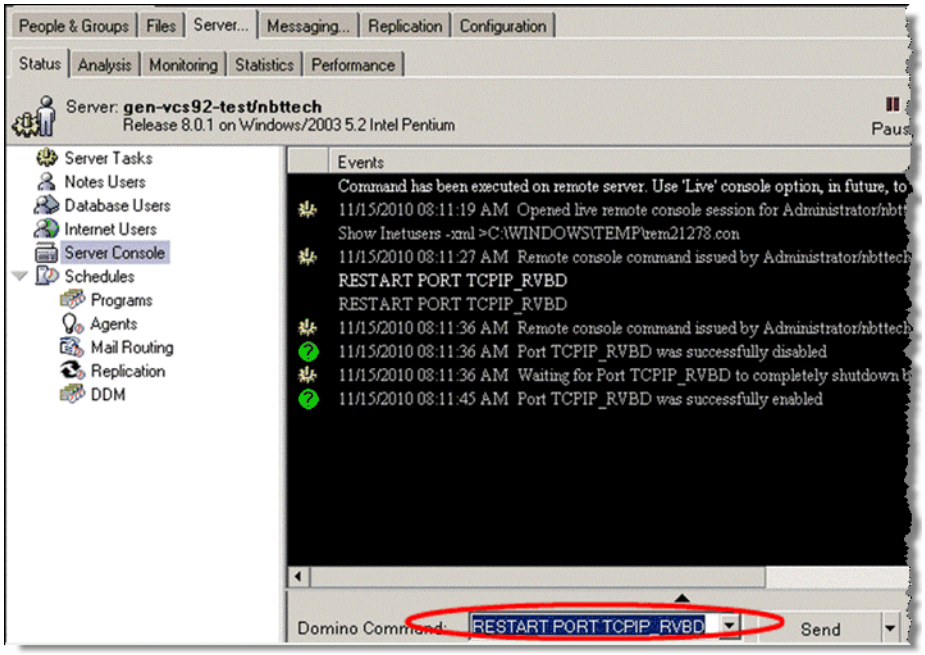


3. Restart the new port.

   The Domino server starts to listen on the new port after a restart of the port or the server.

4. Select the Server tab.
5. Choose Server Console in the left tree structure.
6. In the drop-down menu at the bottom of the page, choose restart port TCPIP_RVBD.

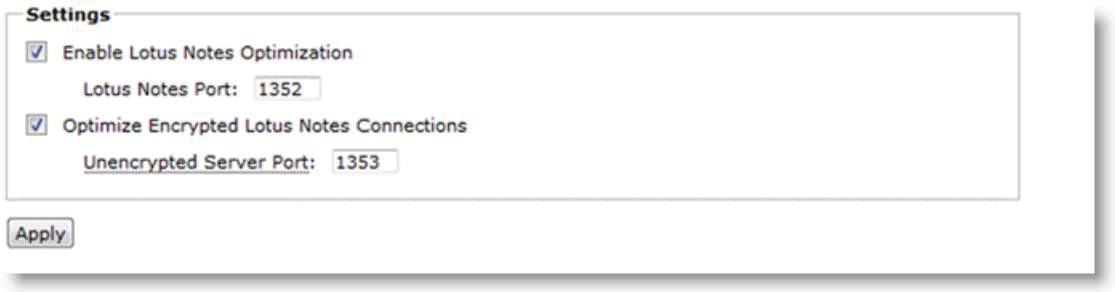
Figure 8-8. Restart the Port



To enable Notes Encryption Optimization on both Steelhead appliances and set the appropriate alternate unencrypted port number on the server-side Steelhead appliance

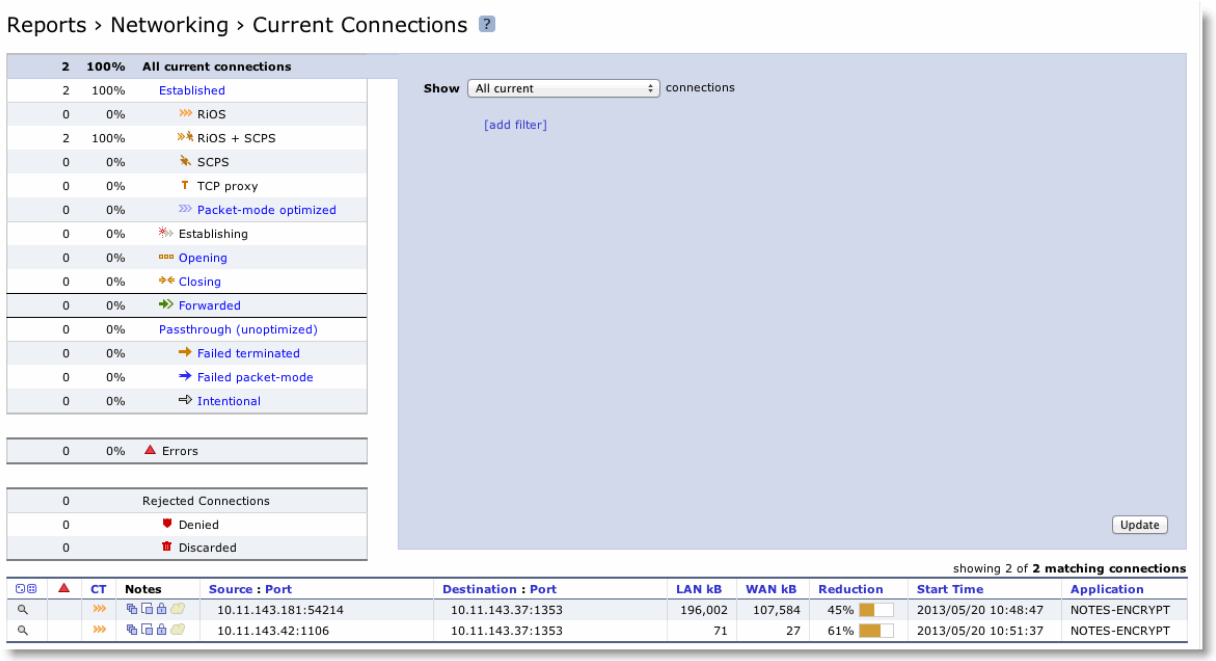
1. From the Management Console, choose Configure > Optimization > Lotus Notes.
2. Select Enable Lotus Notes Optimization.
3. Select Optimize Encrypted Lotus Notes Connections.
4. Specify the configured unencrypted port number.

Figure 8-9. Lotus Notes Page



When the Notes client has connected to the Domino server, you see an optimized encrypted Notes connection in the current connections table. An example client-side Steelhead appliance is shown in Figure 8-10.

Figure 8-10. Client-Side Steelhead Appliance Current Connections Page with Optimized Encrypted Lotus Notes



Riverbed recommends that you also enable secure peering on the Steelhead appliances. The optimized encrypted Notes connection between the client-side and the server-side Steelhead appliance is unencrypted. To secure this connection, you can enable secure peering on the Steelhead appliances. For more details, see "Deploying Secure Steelhead Appliance Peering" on page 162.

Troubleshooting

If the Steelhead appliance encounters an error with the Domino server, the IP address of the Domino server might be added to a blacklist at the server-side Steelhead appliance to avoid disruption to the traffic. This includes configuration errors. New connections to or from the IP address are not optimized.

You can view blacklist entries with the CLI command **show protocol notes encrypt blacklist**. You can clear the entire blacklist or a single entry with the CLI command **protocol notes encrypt blacklist remove-ip <ip-address>**.

The best practice is to check the system logs of the Steelhead appliance to see if something about your Lotus Notes optimization is not working properly: for example, there are messages if the wrong server ID file is imported or the unencrypted port is not configured on the Domino server.

Important notes:

- When you enable encrypted Lotus Notes optimization, you force all connections between a Notes client and a Domino server to be encrypted.
- Enabling Encrypted Notes optimization forces the client to always perform slow authentication.
- The Notes client might try to reuse a previously assigned ticket during authentication by sending an Auth request after the server's Hello response. This is called *fast authentication* because it saves several round trips. Fast authentication does not work when you enable Encrypted Notes optimization.

CHAPTER 9 CIFS and HTTP Prepopulation

You can avoid the penalties of a cold transfer by using CIFS and HTTP prepopulation to warm the Steelhead appliance with data not yet requested by end users. Prepopulation is fully integrated into RiOS v7.0 or later.

The prepopulation of data is extremely beneficial for end users who know of data crossing the WAN: for example, operating system patch updates, antivirus update files, a video training session posted on an internal file server for offline viewing, or a users directory share located at the data center.

You configure prepopulation only on the client-side Steelhead appliance. You must configure and connect the client-side Steelhead appliance Primary interface properly for prepopulation to work. The Primary interface must be able to establish connection with the server.

The Primary interface of the Steelhead appliance acts as a client and requests data from the share you want to use to warm the RiOS data store. To warm both Steelhead appliances, data flows from the server, passes the server-side Steelhead appliance and ends at the client Steelhead appliance.

The traffic leaves the Primary interface and it must traverse a client-side in-path connection (for example, LAN0_0/WAN0_0 for an inline deployment, or WAN0_0 for a logical in-path deployment) before reaching the server-side Steelhead appliance. Prepopulation does not work if the Primary interface bypasses client-side optimization first.

This chapter includes the following sections:

- "CIFS Prepopulation" on page 131
- "HTTP Prepopulation" on page 137

CIFS Prepopulation

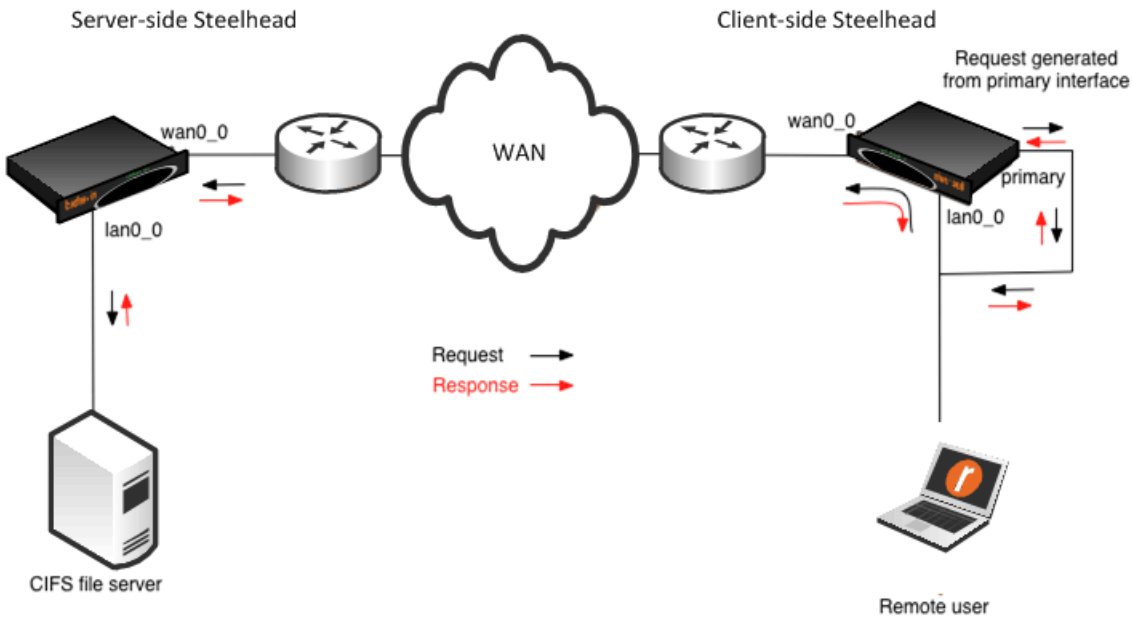
CIFS prepopulation uses a flexible scheduler to prepopulate the RiOS data store. When a client on the remote LAN requests data again, only new or modified data is sent over the WAN, which dramatically increases the rate of data transfers. Unlike with PFS, the client-side Steelhead appliance drops the actual data files it has received, but maintains SDR data in the RiOS data store.

You can use a Windows machine, or any other NAS filer, for the server.

In RiOS v7.0 or later, CIFs shares that require SMB signing are fully supported. To populate SMB-signed CIFs shares, you must join the Steelhead appliance to the domain in which the shares are published or to a trusted domain.

For details on SMB signing, see “Signed SMB and Encrypted MAPI Optimization” on page 33.

Figure 9-1. Traffic Flow for CIFS Prepopulation



When you create a share using CIFS prepopulation, the Steelhead appliance attempts to connect to the destination server on port 139. You must enable NetBIOS over TCP/IP on a Microsoft Windows machine (server or client) to accept the connection on port 139. On a Windows server, enter the following command to see if it is listening on port 139:

```
C:\>netstat -an | find ":139"
TCP 192.168.1.20:139 0.0.0.0:0 LISTENING
```

If you do not see similar output with the correct host IP, NetBIOS over TCP/IP is not enabled on the interface. You can enable it on the relevant interface from the WINS tab in the Advanced TCP/IP settings.

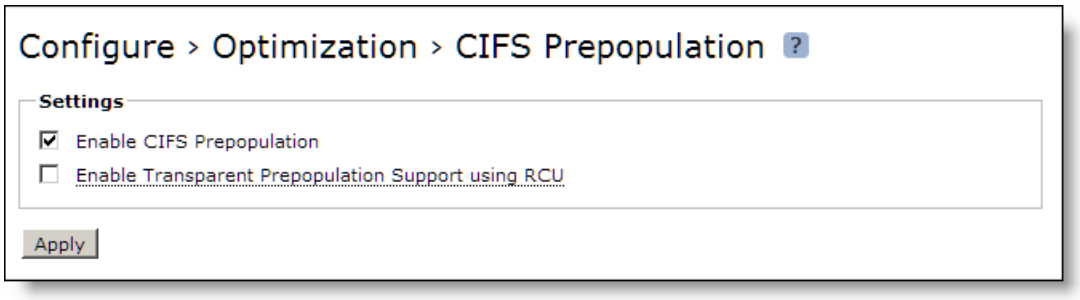
In RiOS v8.5 or later, you can create policies and rules for a higher level of control over which files the system transfers to warm the RiOS data store. A policy is a group of rules that select particular files to prepopulate; for example, you can create a policy that selects all PDF files larger than 300 MB created since January 1st of 2013.

To configure CIFS prepopulation and add a CIFs share using the Management Console

1. From the Management Console, choose Configure > Optimization > CIFS Prepopulation.

2. Select Enable CIFs Prepopulation and click **Apply**.

Figure 9-2. CIFS Prepopulation Page



1. Select Add a Prepopulation Share.
2. Specify the remote path where data is stored in UNC standard: for example, `\\bw-sfowad1\shared`.
3. Specify the account username. This is the login name that has a minimum of read access to the remote path.
4. Specify your password and confirm your password.
5. Add a comment about the share. Comments can not include an ampersand (&).
6. Specify a time limit that the synchronization job should not exceed.
7. Specify a limit on the amount of data in the synchronization job.
8. Select either current files for synchronization or use the latest share snapshot.

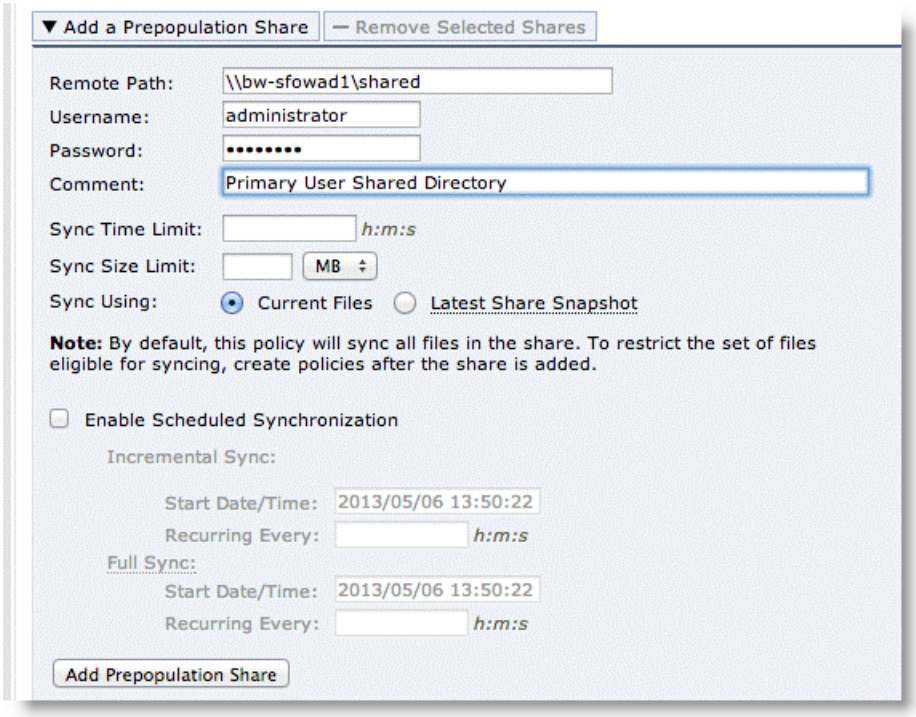
   If no snapshots are available, the system uses the current files.
9. To schedule regular prepopulation events, select Enable Scheduled Synchronization (optional).

   You can schedule full synchronization or incremental synchronization. Use Full Sync for data store wrap to keep potentially evicted data warm.

   If a Full Sync and an Incremental Sync occur simultaneously, the Full Sync takes precedence.
10. Click **Add Prepopulation Share**.

11. Click **Apply**.

Figure 9-3. CIFS Prepopulation Share Information



Share synchronization begins, based on the parameters you entered.

Note: You cannot configure CIFS shares to be guest, anonymous, or public Samba shares without a password.

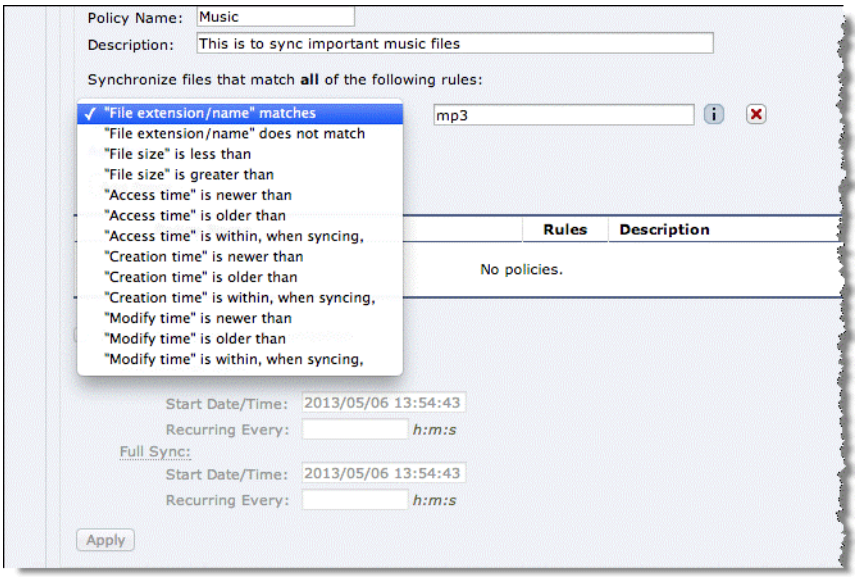
You can create synchronization policies in RiOS v8.5 or later. Synchronization polices enable you to define granular synchronization using filters for inclusion and exclusion specific data types.

To configure CIFS prepopulation policies

1. Open an existing prepopulation share.
2. Select Add a Policy.
3. Specify the policy name and a description.
4. Select data types from the drop-down list to create filers for the policy:
   - File extension
   - File size
   - Access time
   - Creation time

5. Modify time.

Figure 9-4. CIFS Synchronization Policy Filers



For more details on the CIFs prepopulation CLI commands and configuration, see the *Riverbed Command-Line Interface Reference Manual*.

Design Considerations

This section describes several design considerations.

Prewarming the RiOS data store during off-hours helps to reduce WAN bandwidth consumption during peak hours. When you configure CIFS prepopulation, it tries to use all available bandwidth. If you want to control bandwidth usage with CIFS prepopulation, you can configure QoS so that only a portion of the bandwidth is used for CIFs prepopulation.

To configure QoS to work with CIFs prepopulation

1. On the server-side Steelhead appliance, enable QoS and set the proper link speed.
2. Set up a QoS class. Name it PREPOP and set an upper limit (for example, 10%).
3. Set up a QoS rule for the class PREPOP with a destination of the client-side Steelhead appliance primary IP and give it a lower priority.

For more details on QoS, see the *Steelhead Appliance Deployment Guide*.

In RiOS v7.0 or later, clients can authenticate to servers requiring signing and synchronizing transparently from a Windows 2003 server DFS share.

Note: NTLM authentication mode is supported in RiOS v7.0 or later. In previous RiOS versions, CIFS prepopulation could only be performed against unsigned servers and required the use of transparent prepopulation support and the Riverbed Copy Utility (RCU) on the server.

In RiOS v7.0 or later, you can set the length of time a CIFS prepopulation command is to run. The time is reflected in seconds. You can run a test report on the client-side Steelhead appliance to view the expected changes to be synchronized. A sample report follows:

```
(config) # prepop share modify remote-path '\\fileserver\share' max-duration 1800
#--- Set the maximum duration of a prepopulate run
(config) # prepop share dry-run share-name '\\fileserver\share'
#--- Generate a dry run report where they can view a report of the expected changes
(config) # show prepop log dry-run remote-path '\\fileserver\share'
```

RiOS v8.5 or later enhances CIFS prepopulation with a new policy-based system available in the CLI and the Steelhead Management Console. You can create specific policies and associate them with the CIFS prepopulation shares previously created.

The use of policies is a function of the client-side Steelhead appliance. When you use policies, you can synchronize files:

- created since a given set time.
- modified since a given time.
- matching an expression.
- matching a given size range.
- accessed since a given time.

An example policy creation follows:

```
(config) # prepop share policy share-name '\\fileserver\share' policy-name policy1
#--- To create a policy 'policy1' using share '\\fileserver\share'
(config) # prepop share policy share-name '\\fileserver\share' policy-name policy1 create-time time
'2011/10/01 00:00:00' compare-op after
#--- To prepopulate files created after that specific date
(config) # prepop share policy share-name '\\fileserver\share' policy-name policy1 create-time time
'2011/10/05 00:00:00' compare-op before
#--- To prepopulate files created before the earlier specific date
(config) # show prepop share policy share-name '\\fileserver\share' policy-name policy1
#--- To display the specifics of the earlier mentioned policy
(config) # prepop share policy share-name '\\fileserver\share' policy-name policy1 file-name
'A_*.pdf;*.txt' compare-op matches
#--- To only include files with the .pdf and.txt extensions to be prepopulated
#--- Notice the use of the wild character "*". Additional matches need to be separated by a
#--- semi-colon.
(config) # prepop share policy share-name '\\fileserver\share' policy-name policy1 file-name '*.dat'
compare-op not-matches
#--- To exclude files with the.dat extension to be prepopulated
config) # prepop share policy share-name '\\fileserver\share' policy-name policy1 file-size 10M
compare-op less
#--- To prepop files less than 10meg in size
(config) # prepop share policy share-name '\\fileserver\share' policy-name policy1 file-size 5M
compare-op greater
#--- To prepopulate files greater than 5 mb in size
```

You can delete a complete policy directly or delete a specific rule within a policy with the following command.

```
(config) #  no prepop share policy share-name '\\fileserver\share policy-name policy1
```

HTTP Prepopulation

HTTP prepopulation, in RiOS 7.0 or later, is an enhanced HTTP-based data delivery method. HTTP prepopulation delivers data to the remote site by using the HTTP protocol to prewarm your RiOS data store. HTTP prepopulation uses the HTTP protocol to read and deliver data. The feature delivers any data content residing on a Web server, including on-demand video streams at remote site. Remote users can have an enhanced viewing experience. For more details on video optimization, see the *Steelhead Appliance Deployment Guide*.

HTTP prepopulation requires that you configure and connect the primary interface for full functionality. HTTP prepopulation is only available through the command line. Configuration is only on the client-side Steelhead appliance.

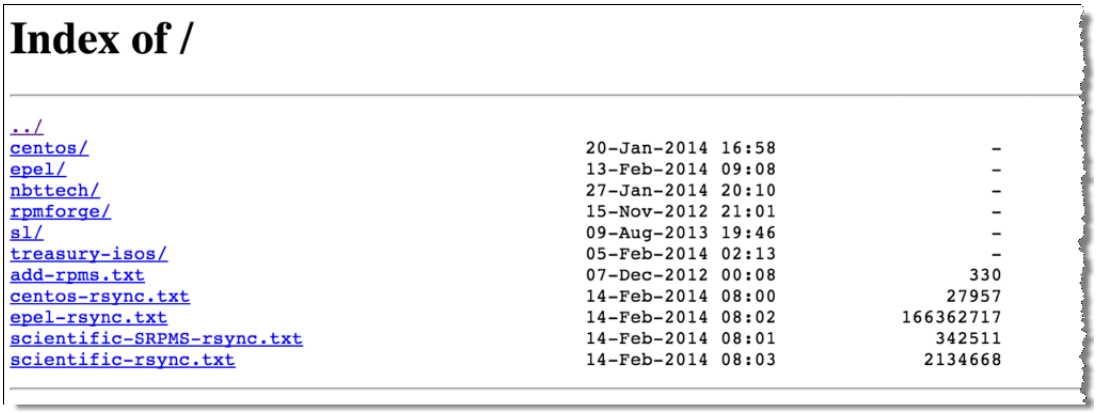
Note: If you are using HTTP prepopulation over IPv6, you must have a an IPv6 address on the primary interface. For more details on IPv6, see the *Steelhead Appliance Deployment Guide*.

To configure HTTP prepopulation you create a list composed of URLs that contain the data you want optimized. You can configure up to 100 lists, and you can include an unlimited number of URLs within each list.

For example, you can combine URL links to multiple Human Resource training videos in one list named *HRvideos*. The URL link must reference a filename with an extension in the URL link such as http://www.example.com/video/example.mp4, and not reference a general URL link such as http://www.youtube.com/videoexample.

Figure 9-5 shows a Web server directory listing. The HTTP prepopulation functionality recursively downloads the entire folder content.

Figure 9-5. Web Server Directory Listing



The following example shows how to configure HTTP prepopulation with the list name HRvideos. List names are case sensitive.

To configure HTTP prepopulation

1. On the client-side Steelhead appliance, connect to the CLI in configuration mode.
2. Build the list name, for example:

```
(config)# protocol http prepop list HRvideos
```

3. Add URLs to the list:

```
protocol http prepop list HRvideos url http://intranet/hr/HowToInterview.mp4
```

You can optimize content located on a secured Web server (HTTPS). If HTTPS is specified as a protocol, you must configure SSL on the client-side Steelhead appliance.

4. Start the process to transfer data across and prewarm your RiOS data store:

```
protocol http prepop list HRvideos start
```

The CLI session is unresponsive until the prepopulation runs to completion.

HTTP prepopulation time varies, depending on transfer time. If you want to maintain access to the CLI, Riverbed recommends that you use the **job** command to run the transfer at a specified time to run the transfer during off hours.

An example of the job command is as follows:

```
(config) # job 1 name prepop
(config) # job 1 command 1 "protocol http prepop list HRvideos start"
(config) # job 1 date-time 12:15:00
(config) # job 1 enable
```

To cancel an HTTP prepopulation job that has not begun

- On the client-side Steelhead appliance, connect to the CLI and enter the following command:

```
job <N> disable
#--- where <N> corresponds to the job number.
```

To cancel HTTP prepopulation in progress

- On the client-side Steelhead appliance, connect to the CLI and enter the following command:

```
protocol http prepop list HRvideos cancel
```

Additional various supporting commands follow:

```
show protocol http prepop lists
show protocol http prepop status all
show protocol http prepop status list HRvideos
```

For more information about HTTP prepopulation commands, see "Microsoft Silverlight, Apple HLS, and Adobe Flash" on page 138 and the *Riverbed Command-Line Interface Reference Manual*.

Microsoft Silverlight, Apple HLS, and Adobe Flash

In RiOS v7.0 and later, the Steelhead appliance supports prepopulation for Microsoft Silverlight On-demand streaming content. In RiOS v8.5 and later, the Steelhead appliance supports prepopulation for Microsoft Silverlight Video, Apple HLS, and Adobe Flash.

To prepopulate video, you must specify the manifest URL containing the manifest file. The manifest file includes information about the different bit rates available, the timeline markers, and so on. Manifest files are generally produced by video encoding software that supports their related format: for example, the Microsoft expressions encoder for Microsoft Silverlight is MP4.

In RiOS v8.5 and later, you can specify a URL (as opposed to specifically referencing the manifest file in the URL link as in previous RiOS versions). RiOS v8.5 and later automatically:

- parses the content.
- identifies the manifest file.

- extracts the proper information.
- extracts the constituent segment URLs.
- represents the videos for all bit rates to download.

You need a browser plug-in on the client machine to play and watch the video. For example, Microsoft Silverlight is compatible with Internet Explorer, Firefox, and Safari.

The Microsoft Silverlight manifest file is an XML file, with a .ISM extension, that describes the live or on-demand video presentation. The Apple HLS manifest file is a text-based manifest file with a .M3U8 or .M3U extension that directs the player to additional manifest files for each of the encoded streams. The Adobe HTTP Dynamic Streaming manifest file has a .F4M extension.

Note: A manifest file is invisible to the end user; only end-user video players use it.

To configure HTTP prepopulation, use the following commands on the client-side Steelhead appliance:

- Create or delete a new prepopulation list with a user-specified name:

```
[no] protocol http prepop list *
```

- Add or remove URL to the list that needs to be recursively prepopulated. All the objects in the page are fetched recursively (1 level deep):

```
[no] protocol http prepop list * url *
```

- Start the prepopulation operation:

```
protocol http prepop list * start
```

- Cancel the prepopulation operation:

```
protocol http prepop list * cancel
```

- Display HTTP prepopulation status:

```
show protocol http prepop status all
```

- Display specified list status info:

```
show protocol http prepop status list *
```

- Display specified list configured URLs:

```
show protocol http prepop list *
```

- Display all the configured list:

```
show protocol http prepop lists *
```

CHAPTER 10 SSL Deployments

This chapter describes how to configure SSL support. This chapter includes the following sections:

- "The Riverbed SSL Solution" on page 141
- "Overview of SSL" on page 143
- "Configuring SSL on Steelhead Appliances" on page 146
- "Advanced SSL Features" on page 163
- "Steelhead Mobile SSL High-Security Mode" on page 167
- "Troubleshooting and Verification" on page 168
- "Interacting with SSL-Enabled Web Servers" on page 169

This chapter requires that you be familiar with the SSL protocol.

The Riverbed SSL Solution

The Riverbed SSL solution accelerates data transfers that are encrypted using SSL, provided Steelhead appliances are deployed locally to both the client-side and server-side of the network. All of the same optimized connections that are applied to normal non-encrypted TCP traffic, you can also apply to encrypted SSL traffic. Steelhead appliances accomplish this without compromising end-to-end security and the established trust model. Your private keys remain in the data center and are not exposed in the remote branch office location where they might be compromised.

The Riverbed SSL solution starts with Steelhead appliances that have a configured trust relationship, enabling them to exchange information securely over their own dedicated SSL connection. Each client uses unchanged server addresses and each server uses unchanged client addresses; no application changes or explicit proxy configuration is required. Riverbed uses a unique technique to split the SSL *handshake*.

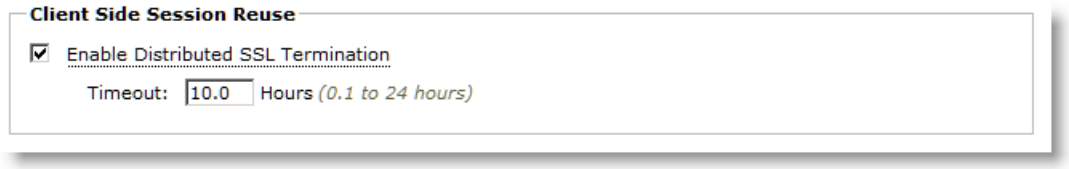
The handshake is the sequence of message exchanges at the start of an SSL connection. In an ordinary SSL handshake, the client and server first establish identity using public-key cryptography, and then negotiate a symmetric session key to use for data transfer. When you use Riverbed's SSL acceleration, the initial SSL message exchanges take place between the client application (for example, a Web browser) and the server-side Steelhead appliance.

Prior to RiOS v6.0, the SSL handshakes from the client are always handled by the server-side Steelhead appliance, including session reuse handshakes.

RiOS v6.0 or later provides an alternative strategy, called *distributed termination*, in which initial full handshakes are terminated on the server-side Steelhead appliance, while subsequent reuse handshakes are terminated by the client-side Steelhead appliance.

Distributed termination is enabled by default and is on the Configure > Optimization > Advanced Settings page. The time-out value specifies the amount of time the client can reuse a session with an SSL server after the initial connection ends.The range is six minutes to 24 hours. The default value is ten hours.

Figure 10-1. Distributed Termination Setting on the Advanced Settings Page



Distributed termination improves performance by reducing the CPU load on the server-side Steelhead appliance and shortens the key negotiation process by avoiding WAN round trips to the server. Distributed termination also shortens the key negotiation process by avoiding WAN round trips to the server. You can configure reuse of a client-side session for distributed termination on the Configure > Optimization > Advanced Settings page in the Management Console. For more details on distributed termination, see "How Steelhead Appliances Terminate SSL" on page 144.

Riverbed has worked with large enterprise design partners to ensure that SSL acceleration delivers real world benefits in real-world deployments, specifically:

- sensitive cryptographic information is kept in the secure vault—a separate, encrypted store on the disk.
- built-in support for popular Certificate Authorities (CAs) such as VeriSign, Thawte, Entrust, and GlobalSign. In addition, Steelhead appliances allow the installation of other commercial or privately-operated CAs.
- import of server proxy certificates and keys in PEM, PKCS12, or DER formats. Steelhead appliances also support the generation of new keys and self-signed certificates. If your certificates and keys are in another format, you must first convert them to a supported format before you can import them into the Steelhead appliance. For details, see "SSL Required Components" on page 146.
- separate control of cipher suites for client connections, server connections, and peer connections.
- bulk export or bulk import server configurations (including keys and certificates) from or to, respectively, the server-side Steelhead appliance.
- that you can use the CMC to streamline setup of Steelhead appliance trust relationships.

Note: For more information, see the *Steelhead Appliance Management Console User's Guide* and *Riverbed Central Management Console User's Guide*.

The Steelhead appliance has a secure vault which stores all SSL server settings, other certificates (that is, the CA, peering trusts, and peering certificates), and the peering private key. The secure vault protects your SSL private keys and certificates when the Steelhead appliance is not powered on.

Initially the secure vault is keyed with a default password known only to the RiOS software. This allows the Steelhead appliance to automatically unlock the vault during system start up. You can change the password, but the secure vault does not automatically unlock on start up. To optimize SSL connections, the secure vault must be unlocked.

Overview of SSL

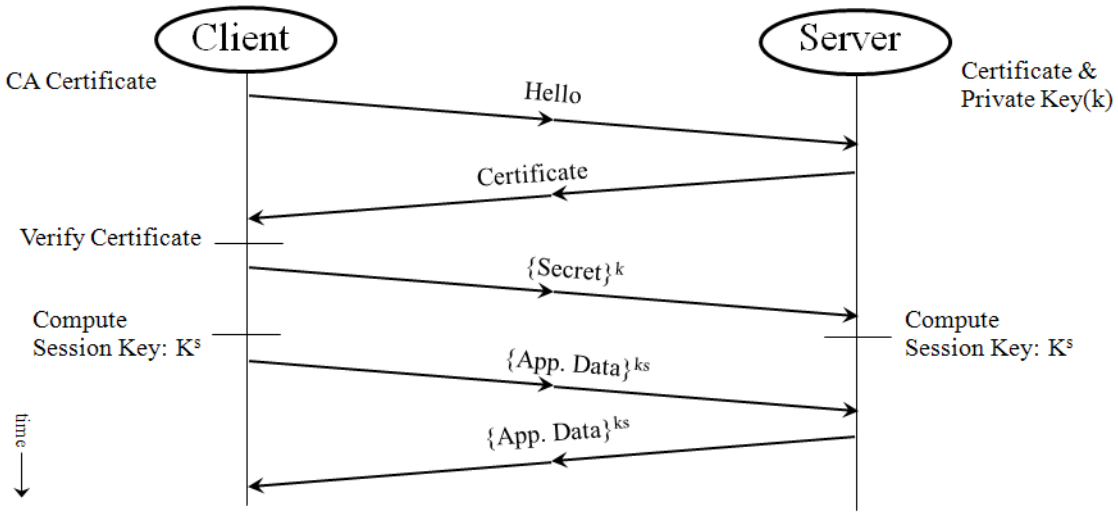
Because a complete description of SSL is outside the scope of this guide, this section provides a brief overview of the relevant components of SSL security.

SSL provides a way for client and server applications to communicate securely over a potentially insecure network. It provides authentication, and prevents eavesdropping and tampering. In the most common use case, you can SSL to transport HTTP traffic, and provide one-way identification: only the Web server authenticates itself to the Web browser.

Important: The SSL features have changed with each release of RiOS. If you are running a version of RiOS that is earlier than v6.0, consult the appropriate documentation for that software release.

Figure 10-2 shows a simplified SSL handshake where the server authenticates itself to the client.

Figure 10-2. A Simple SSL Handshake



In response to the Hello message, the server sends back its certificate. The certificate contains the server's public key, some identifying information about the server (such as its name and location), and a digital signature of all this information. This digital signature is issued by an entity called a Certificate Authority (CA) that both the client and the server trust, and serves as proof that no one has tampered with the certificate.

Upon receiving the certificate, the client verifies that it has not been tampered with and that it does belong to that particular server. Then, the client generates a random number N, encrypts it with the server's public key, and sends it to the server. At this point, both the client and the server use the same function to derive the session key, ks, from N. In the simplest case, after the initial SSL handshake between the client and the server, and the creation of the session keys, the same session keys are used for the duration of the session, without the need to go through another SSL handshake.

How Steelhead Appliances Terminate SSL

At a high level, Steelhead appliances terminate an SSL connection by making the client think it is talking to the server and making the server think it is talking to the client. In fact, the client is talking securely to the Steelhead appliances. You require some special provisioning to accomplish optimization that appears transparent to the client.

To enable SSL connection termination, you must configure the server-side Steelhead appliance to include proxy certificates and private keys for the purpose of emulating the server. For details, see "Server SSL Optimization Proxy Certificate and Private Key Scenarios" on page 152. When the Steelhead appliance poses as the server, there does not need to be any change to either the client or the server. The security model is not compromised—the optimized SSL connection continues to guarantees server-side authentication, and prevents eavesdropping and tampering.

When transferring data over the WAN on behalf of an optimized SSL connection, the client and server-side Steelhead appliances ensure that their inner connection provides all the security features the original SSL connection would have, had it not been optimized. Steelhead appliances accomplish this by establishing their own SSL connection between themselves. To secure the inner Steelhead channel, you must configure each Steelhead appliance with the certificate of the peer Steelhead appliance (secure peering). There are various methods to accomplish a secure inner channel between the Steelhead appliances. For more details, see "Steelhead Appliance Secure Peering Scenarios" on page 154.

To securely terminate an SSL connection to an SSL server, the following is a high-level configuration for the server-side Steelhead appliance (for more details on how to configure SSL on a Steelhead appliance, see the *Steelhead Appliance Management Console User's Guide*):

- A certificate and private key pair for the server. This certificate and private key pair does not have to be the same as the one used by the actual server. In a production environment, it would typically be signed by a CA trusted by the client.

Important: To avoid confusion, in this chapter, the certificate residing on the server-side Steelhead appliance (for the server), is referred to as the *proxy certificate*. The proxy certificate might or might not be the same as the actual server certificate, but it serves the same function. The one exception is if you use client certificates. For details, see "Client Certificate Support" on page 164.

- The certificate of the client-side Steelhead appliance.

The client-side Steelhead appliance has only the server-side Steelhead appliance certificate for secure peering between the client-side and server-side Steelhead appliances. The client-side Steelhead appliance does not need the certificates or keys of the server. This discourages any tampering at the branch office.

The following describes distributed termination:

- Full SSL handshakes terminate on the server-side Steelhead appliance.
- The master secret containing information that allows the computation of the session key for reusing the session is transported to the session cache of the client-side Steelhead appliance.
- The subsequent handshakes are reused and terminate on the client-side Steelhead appliance.

Figure 10-3 shows a high-level view of application data crossing the network from the client to the server. After the SSL connection is terminated, there are only three session keys involved: kc, ks, and kt. The data is encrypted with session key kc; then it is decrypted, optimized, and reencrypted with session key kt for the inner secure SSL channel between the two Steelhead appliances:

- kc is the session key between the client initiating the transmission and the server-side Steelhead appliance.
- ks is the session key between the server-side Steelhead appliance and the server.

- kt is the session key between the client-side Steelhead appliance and the server-side Steelhead appliance for the inner secure SSL channel.

Figure 10-3. Typical Data Transfer Operations on SSL Connections Accelerated by Steelhead Appliances

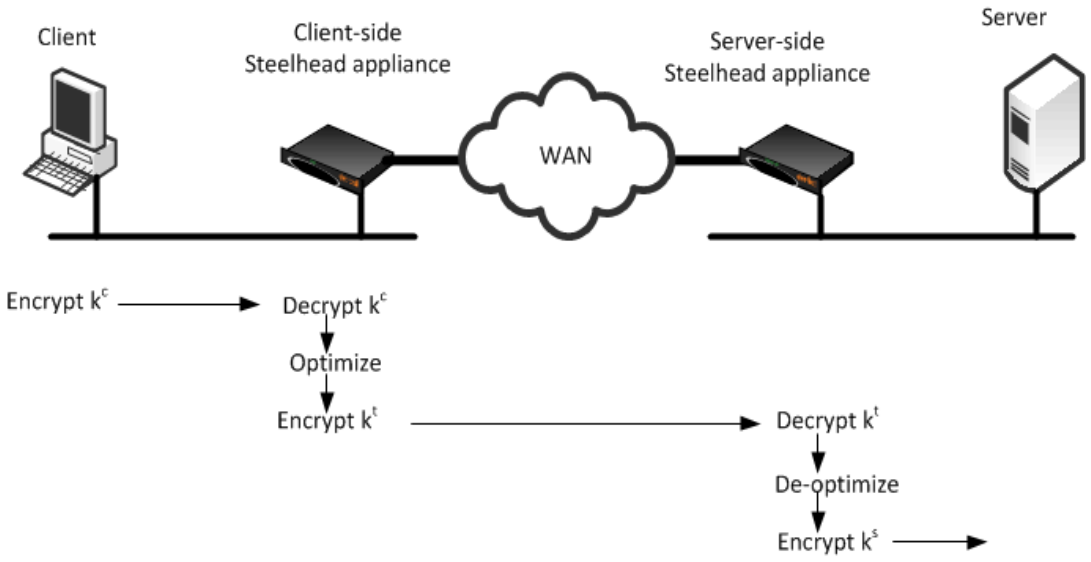
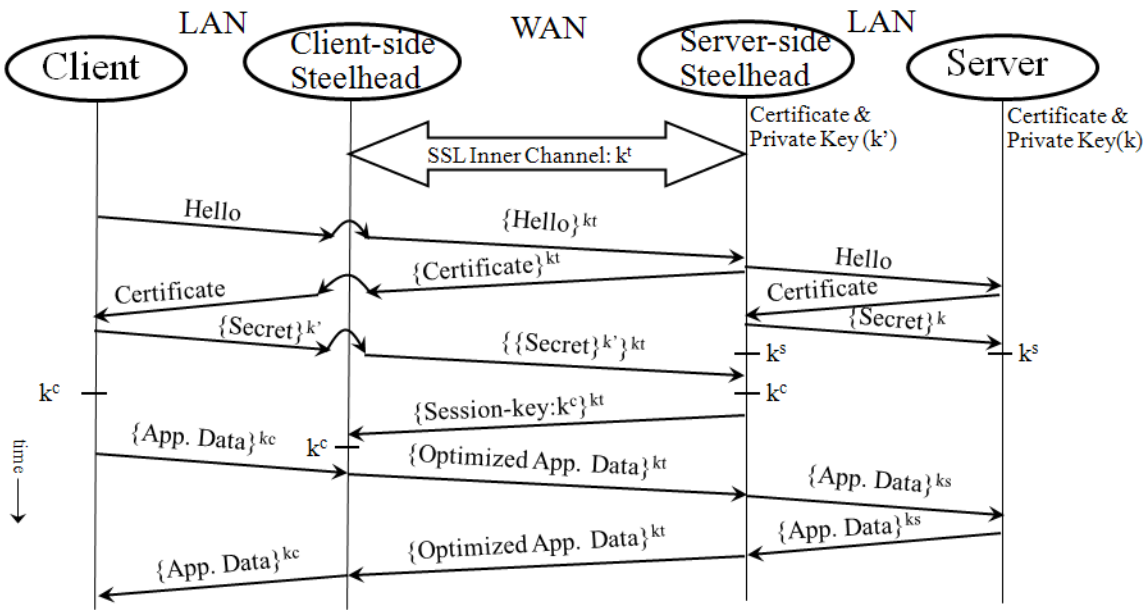


Figure 10-4 shows in time sequence, the complete set of actions to set up an SSL connection.

Figure 10-4. Time Sequence Diagram



Configuring SSL on Steelhead Appliances

This section describes how to deploy SSL on the Steelhead appliance and provides common configuration examples. This section includes the following topics:

- "SSL Required Components" on page 146
- "Setting Up a Simple SSL Deployment" on page 148
- "Server SSL Optimization Proxy Certificate and Private Key Scenarios" on page 152
- "Steelhead Appliance Secure Peering Scenarios" on page 154
- "Deploying Secure Steelhead Appliance Peering" on page 162
- "Client Certificate Support" on page 164

SSL Required Components

You need the following SSL components to deploy SSL on Steelhead appliances:

- "Enhanced Cryptography License Key" on page 146
- "Proxy Certificate and Private Key" on page 146
- "Certificate Chain Discovery" on page 147
- "Certificate Authority Certificates" on page 148
- "Peer Certificates" on page 148

Enhanced Cryptography License Key

US export restrictions require that this license is installed on each Steelhead appliance that has SSL enabled. You can acquire an enhanced cryptography license key by filling out the online form available at http://sslcert.riverbed.com.

Proxy Certificate and Private Key

The proxy certificate is the certificate you configure on the server-side Steelhead appliance for the server. Do not confuse this with the certificate used for the secure inner channel or secure peering between the two Steelhead appliances.

Important: To avoid confusion, in this chapter, the certificate residing on the server-side Steelhead appliance (for the server), is referred to as the *proxy certificate*. The proxy certificate might or might not be the same as the actual server certificate, but it serves the same function. The one exception is if you use client certificates. For details, see "Client Certificate Support" on page 164.

The proxy certificate can be self-signed, signed by a well-known CA, or signed by your organization's own CA. It can be the same as, or different from, the certificate used by the actual server. The correct type of certificate depends on your deployment.

You can use a single wild card certificate and its private key as the proxy certificate for multiple servers.

For example, you might have three origin servers using different certificates: sales.riverbed.com, internal.riverbed.com, and marketing.riverbed.com. You can add each of the three server certificates (as the proxy certificates) and their corresponding private keys, or you can add a single wild card certificate (*.riverbed.com) with its private key.

Note: In RiOS versions earlier than v6.0, you must configure each server. But in RiOS versions v6.x or later, you add the wild card certificate only once.

RiOS v6.0 or later simplifies the SSL configuration process because the server-side Steelhead appliance does not need to have a proxy certificate for each server individually. Prior to v6.0, you had to provide an IP address, port, and certificate to enable SSL optimization for a server. This method is impractical in cases where multiple servers shared the same server certificate because each server had to have a corresponding entry on the server-side Steelhead appliance.

In RiOS v6.0 or later, you simply add unique proxy certificates to a certificate pool on the server-side Steelhead appliance. When a client initiates an SSL connection with a server, the Steelhead appliance matches the common name of the certificate on the server with one in its proxy certificate pool. If it finds a match, it adds the server to the list of optimizable servers, and all subsequent connections to that server are optimized with the same proxy certificate.

If it does not find a match, it adds the server name to the list of bypassed servers and all subsequent connections to that server are not optimized. The optimizable and bypassed server lists appear on the server-side Steelhead appliance SSL Main Settings page of the Management Console.

If your proxy certificate and private key are in a format other than PEM, PCKS12, or DER, you must convert the certificate and key before you import them. The file extension does not necessarily dictate the encoding, as many common extensions such as .crt, .cer or .key can be either PEM or DER encoded.

To convert a certificate and private key, Riverbed recommends using an open source toolkit such as OpenSSL. For details, see http://www.openssl.org/.

For details on converting a Base64 encoded PCKS12 certificate to PEM, see the Riverbed Knowledge Base article Converting a PCKS12 Certificate/Key to PEM at https://supportkb.riverbed.com/support/index?page=content&id=S13301.

Certificate Chain Discovery

You can use intermediary certificates instead of signing all certificates with a root level certificate. In releases earlier than RiOS v5.5, you needed to put all of the intermediate certificates on the server-side Steelhead appliance and keep them up-to-date. With RiOS v5.5 or later, you can configure the server-side Steelhead appliance to automatically pull down intermediate certificates from the back-end server.

Use the following command on the server-side Steelhead appliance to enable certificate chain discovery:

```
protocol ssl backend server chain-cert cache enable
```

Alternatively you can use the Management Console to enable certificate chain discovery. Choose Configure > Optimization > Advanced Settings and select Enable SSL Server Certificate Chain Discovery. By default, this option is disabled.

Certificate Authority Certificates

When the server-side Steelhead appliance establishes a connection to the server, it needs to validate the certificate on the server. It is common to have this certificate signed by a commercial CA (such as VeriSign). Certificates for these common CAs are preinstalled on Steelhead appliances, which allows the appliance to validate the authenticity of the server certificate without any additional configuration. When a private CA is used, you must install the certificate for this CA on the server-side Steelhead appliance.

Peer Certificates

Each Steelhead appliance participating in SSL optimization requires a peer certificate. The peer certificates allow the client-side and server-side Steelhead appliances to establish a secure SSL channel. You can distribute these certificates several different ways: manual cut-and-paste; using the white, gray, and black peering lists; or through the CMC.

The peer certificates between the Steelhead appliances can, and in most cases are, self-signed certificates. The session keys between the Steelhead appliances are already used for encrypted data by the session key formed between the client-side and the server-side Steelhead appliance. The inner SSL channel ensures an end-to-end SSL transaction. For details, see "Setting Up a Simple SSL Deployment" on page 148 and "Managing Steelhead Appliance Trusts with a CMC" on page 158.

Setting Up a Simple SSL Deployment

This section describes the steps required to configure the simplest of all the SSL deployments and helps you to understand and test basic SSL functionality. This simple deployment consciously makes a trade-off between simplicity and security in favor of the former and installs a self-signed server certificate on the server-side Steelhead appliance. Use this configuration for an already-deployed enterprise SSL-based Web application.

Important: Because this deployment actually breaks the SSL security model, you should use it for scenarios where the focus is on improving performance and security is not the primary concern

---

To conduct a series of performance tests, you need two Steelhead appliances, a WAN simulator, a LAN connection between the client and client-side Steelhead appliance, and another LAN connection between the server-side Steelhead appliance and the server. You do not need to acquire or generate a valid certificate for the server, however the configuration steps remain as if this were a CA signed certificate.

This configuration assumes your Web server is configured to support SSL connections and is running, and that the client-side and the server-side Steelhead appliances can optimize non-SSL TCP traffic. The type peering used in the simple SSL deployment is traditional peering with self-signed certificates. For more information about traditional peering, see "Steelhead Appliance Secure Peering Scenarios" on page 154.

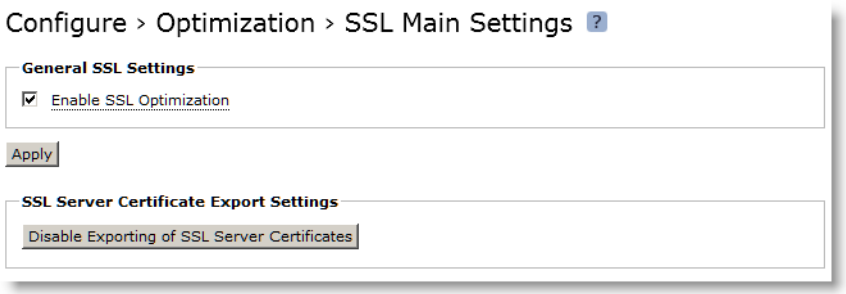
To set up a simple SSL deployment

1. To set up the client-side Steelhead appliance:
   - On the client-side Steelhead appliance, choose Configure > Networking > Port Labels to display the Port Labels page in the Management Console.
   - Select the Secure label, remove port 443, select Remove Selected, and click **Apply**.

     Default bypass rules contain all secure label ports, and hence secure traffic is not optimized by default.
   - Choose Configure > Maintenance > Licenses to verify that the Enhanced Cryptographic License Key is installed on the client-side and server-side Steelhead appliances.
2. To set up the server-side Steelhead appliance:
   - On the server-side Steelhead appliance, choose Configure > Optimization > SSL Main Settings to display the SSL Main Settings page in the Management Console.
   - Select Enable SSL Optimization and click **Apply**.

     Wait until the server-side Steelhead appliance is back to a Healthy state.

Figure 10-5. SSL Main Settings Page

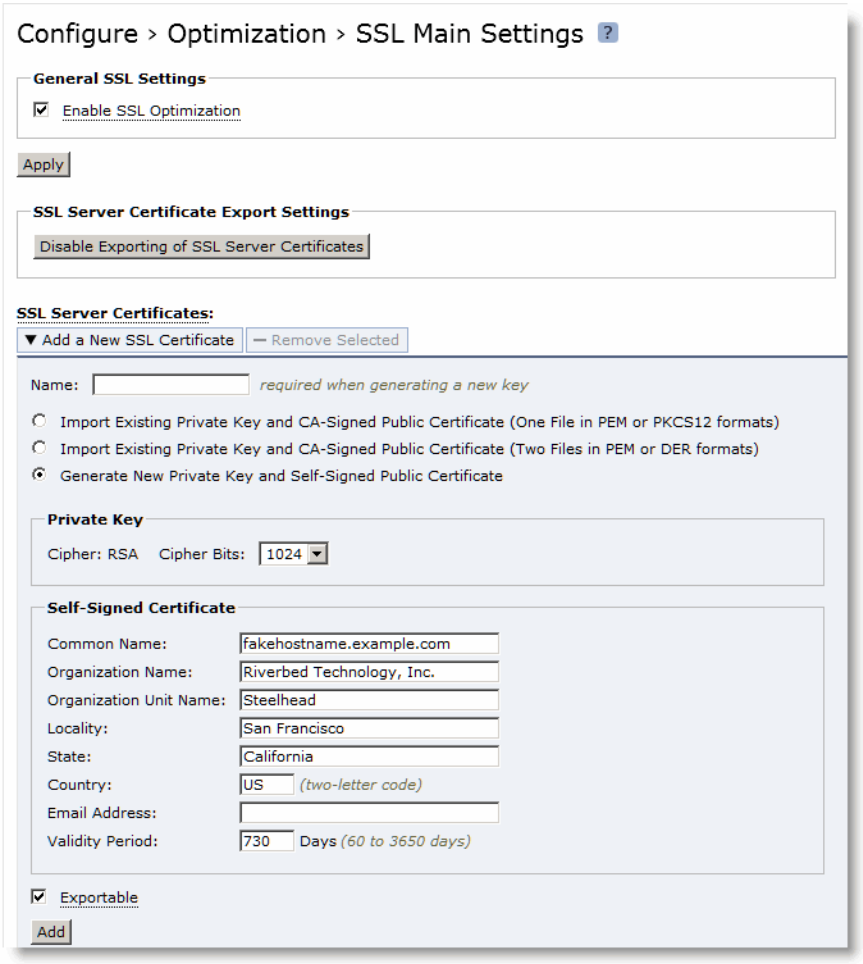


   - Add the proxy certificate and a private key for the server to the server-side Steelhead appliance.

     Select Add a New SSL Certificate, specify the common name of the server (or specify a wildcard server name), select Generate New Private Key and Self-Signed Public Certificate, and click **Add**.

The common name must be the real server common name or one of its subject alternative names.

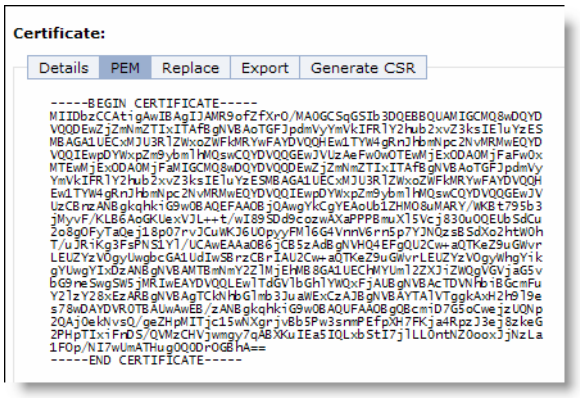
Figure 10-6. SSL Main Settings Page



3. Configure secure peering (traditional peering)
   - Choose Configure > Optimization > Secure Peering (SSL) to display the Secure Peering (SSL) page.
   - Under Certificate, select PEM, and copy and paste the certificate to the clipboard.

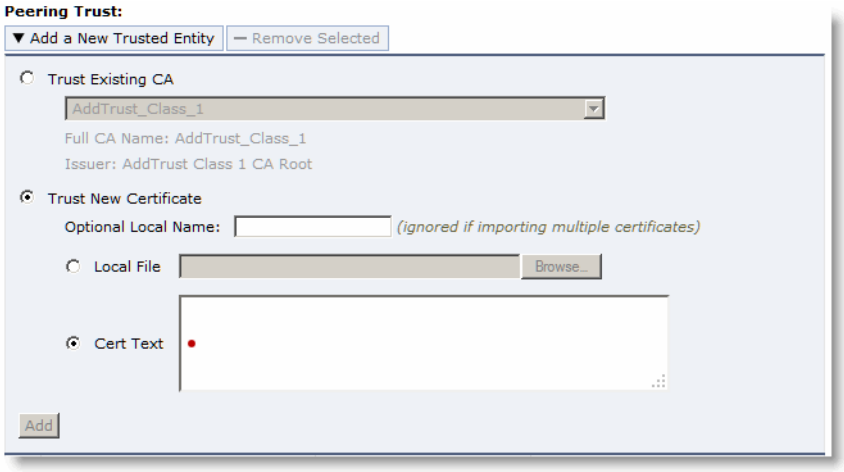
This is the self-signed peering certificate that represents the server side appliance for securing the inner channel. Do not confuse this with the proxy certificate between the server-side Steelhead appliance and the server.

Figure 10-7. PEM Certificate



- On the client-side Steelhead appliance, choose Configure > Optimization > SSL Main Settings, select Enable SSL Optimization and click **Apply**.
- Choose Configure > Optimization > Secure Peering (SSL).
- Under Peering Trust, select Add a New Trusted Entity.

Figure 10-8. Adding a New Trusted Entity

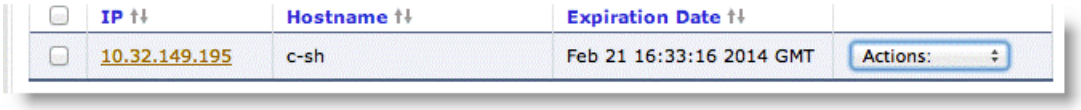


- Select Trust New Certificate, type an Optional Local Name, paste the copied server-side Steelhead self-signed certificate into the Cert Text box, and click **Add**.
- Save the configuration and restart the optimization service on both Steelhead appliances.
- Choose > Reports > Networking > Current Connections report in the Management Console on the server-side Steelhead appliance, to verify that the connections are being optimized. This report summarizes the SSL connection requests and connection rate.

In the Management Console on the server-side Steelhead appliance, the Current Connections report lists connections as optimized without a Protocol Error flag.

If you see a protocol error flag, look to make sure both Steelhead appliances see its adjacent peer in the white list of the Configure > Optimization > Secure Peering (SSL) page. If one of the Steelhead appliances is in the gray list, select that particular Steelhead appliance by checking the box next to it and select Trust in the drop-down Action list. For more details, see "Simplified Peering Using White, Gray and Black Peer Lists" on page 154.

Figure 10-9. Self-Signed Peer Gray List



Another sign that SSL is working is if you receive a warning pop-up window in the browser, notifying the you that the authenticity of the server cannot be established. This is expected, because the server-side Steelhead appliance presents a self-signed certificate to the client, which it does not trust yet. The server-side Steelhead appliance is using a certificate that was automatically generated (server-side Steelhead appliance self-signed certificate), which is not in the client's trusted root authority certificate store.

For details on troubleshooting and verification, see "Troubleshooting and Verification" on page 168.

Note: Prior to RiOS v5.5, port 443 was not in the list of ports that had HTTP-specific optimization enabled. To turn these on for an SSL-enabled Web application in RiOS versions prior to v5.5, you must add the appropriate in-path rule on the client-side Steelhead appliance to turn on HTTP optimization, in addition to the above steps.

Server SSL Optimization Proxy Certificate and Private Key Scenarios

For the server-side SSL handshake, use the following primary ways to configure the server-side Steelhead appliance with proxy certificates and private keys:

- "Real Server Private Key and Certificate" on page 152
- "Real Server Private Key and Different CA Signed Proxy Server Certificate" on page 153
- "Steelhead Appliance Private Key and CA Signed Proxy Server Certificate" on page 153
- "Steelhead Appliance Self-Signed Private Key and Server Proxy Certificate" on page 153

Real Server Private Key and Certificate

You can implement the real server private key and the real server certificate on the server-side Steelhead appliance as the proxy certificate and the key for the server. You do not need to change the client, as the client detects the server-side Steelhead appliance as the real server, and completes the SSL handshake with the server-side Steelhead appliance as if it were the server.

Figure 10-10. Real Server Private Key and Certificate



Real Server Private Key and Different CA Signed Proxy Server Certificate

You can implement the real server private key, and create a second certificate (using the server private key and common name or alternate names of the server by a CA) and key on the server-side Steelhead appliance as the proxy certificate and key for the server.

You might choose to use this scenario if you had to make some compromises. For example, you might want to optimize with client certificates. This requires that you use the server's real private key, but if the real private key of the server is not suitable for Steelhead appliance deployment for some reason—the security requirements preclude server impersonation, the subject alternative names are a problem, you desire a longer validity period than the original certificate, or you want to use a wild card subject name that is broader than the original—you can prepare a different certificate with the same key to facilitate the optimization.

The proxy certificate can be signed by any CA (well-known or customer enterprise CA). The certificate for the particular CA signing the proxy certificate must be present in the client's Trusted Root Certificate Authority Store to avoid SSL warnings on the browser. You can also have the actual server certificate signed by one CA (for example, a well-known CA like Verisign), but the proxy certificate signed by another CA (for example, an enterprise CA), as long as the client trusts the particular CA's (signed) certificates.

Figure 10-11. 2.Real Server Private Key and Different CA Signed Proxy Server Certificate



Steelhead Appliance Private Key and CA Signed Proxy Server Certificate

You can generate and implement a private key on the server-side Steelhead appliance (not imported from the server), and create and sign a proxy certificate (using server's common or alternate name) by a CA using the generated private key.

The proxy certificate can be signed by any CA (well-known or customer enterprise CA). The certificate for the particular CA signing the proxy certificate must be present in the client's Trusted Root Certificate Authority Store to avoid SSL warnings on the browser. You can also have the actual server certificate signed by one CA (for example, a well-known CA like Verisign), but the proxy certificate signed by another CA (for example, an enterprise CA), as long as the client trusts the particular CA's (signed) certificates.

Figure 10-12. Steelhead Appliance Private Key and CA Signed Proxy Server Certificate



Steelhead Appliance Self-Signed Private Key and Server Proxy Certificate

You can generate and implement a private key and self-signed proxy certificate on the server-side Steelhead appliance using the common or alternate names of the server. This is similar to the simple SSL deployment described in "Setting Up a Simple SSL Deployment" on page 148.

The server-side Steelhead appliance is the CA, and you do not need another CA to sign the proxy certificates. Client Web browsers might display a warning screen when accessing the server over SSL because this CA/server-side Steelhead appliance is most likely not trusted by the clients. You would use this scenario for proofs of concepts and testing Steelhead appliance SSL functionality. To avoid seeing the SSL warnings on the clients, you must add the self-signed proxy certificate to the to the client's Trusted Root Certificate Authority Store. A Web browser can use its own certificate store, or the operating systems keystone.

Figure 10-13. Steelhead Appliance Self-Signed Private Key and Server Proxy Certificate



Steelhead Appliance Secure Peering Scenarios

The following are the primary ways you can configure secure peering between the client-side and server-side Steelhead appliances (the secure inner channel):

- "Simplified Peering Using White, Gray and Black Peer Lists" on page 154
- "Traditional Peering with Self-Signed Certificates" on page 155
- "Traditional Peering with CA Signed Certificates" on page 156
- "Managing Steelhead Appliance Trusts with a CMC" on page 158

Simplified Peering Using White, Gray and Black Peer Lists

You can use peering lists to configure peer certificates on Steelhead appliances running RiOS v5.0 or later. Whenever a Steelhead appliance fails to establish a secure SSL channel to its peer due to the lack of a peer certificate, that peer, along with its certificate, is put into the self-signed peer gray list. This indicates that the Steelhead appliance does not know whether it can trust the peer. If a peer is in the gray list, a Steelhead appliance bypasses the SSL connection. No transport, data, or application streamlining applies to this connection because there is no interception.

You can examine the stored peer certificate for authenticity and move any peer from the gray list to either the white list or the black list. Presence in the white list indicates that the peer is trusted, and its certificate is valid. When two Steelhead appliances that are attempting to peer with each other have each other in their respective white lists, they can establish a secure SSL channel between themselves. Presence in the black list indicates that the peer is not trusted, and that the Steelhead appliance cannot create a secure SSL channel to it.

Important: Riverbed highly recommends verifying the fingerprint on the peer certificate to confirm that it does indeed belong to the peer Steelhead appliance.

Using self-signed peer lists is a reasonable way to deploy SSL optimization in smaller deployments or a proof of concept. This method also allows for the simplest set-up, as is shown in the following procedure.

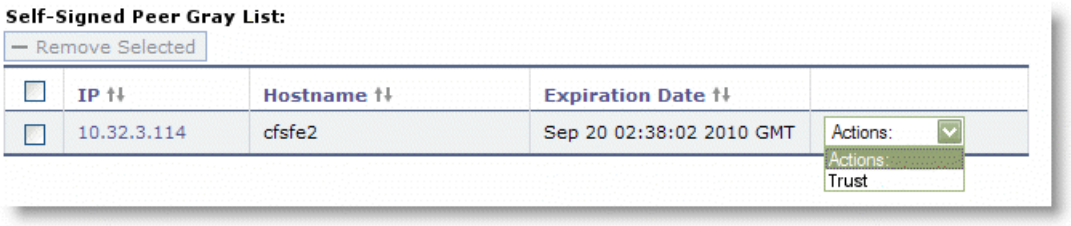
To manage Steelhead appliance trust using the white, gray, and black peering lists

1. On the client-side Steelhead appliances, choose Configure > Networking > Port Labels to display the Port Labels page in the Management Console.

2. Select the Secure label, remove port 443, select Remove Selected, and click **Apply**.
3. On both the client-side and the server-side Steelhead appliances, choose Configure > Optimization > SSL Main Settings and select Enable SSL Optimization.
4. On the server-side Steelhead appliance, open Configure > Optimization > SSL Main Settings page and add the SSL server.
5. Restart the optimization service on both Steelhead appliances.
6. Establish an SSL connection from a client behind the client-side Steelhead appliance to the server.
7. On both the server-side and the client-side Steelhead appliances, choose Configure > Optimization > Secure Peering (SSL).
8. Under Self-Signed Peer Gray list, identify the peer Steelhead appliance. Optionally, you can examine this certificate on the peer and make sure both copies match. After you verify the authenticity of the certificate, select Trust from the Actions drop-down list.

Important: Before moving a peer from the gray list to the trusted peers white list, it is critical to verify that the certificate fingerprint does indeed belong to a peer Steelhead appliance, particularly to avoid the potential risk of a man-in-the-middle attack. Select the IP address in next to the hostname to view the certificate and compare this to the peering certificate installed on the target Steelhead appliance.

Figure 10-14. Self-Signed Peer Gray List



9. Save the configuration.

The Steelhead appliances optimizes connections to the configured SSL servers. To see successful SSL connections, view the Reports > Optimization > SSL to view the SSL report. For details on verifying SSL connections, see "Troubleshooting and Verification" on page 168.

Traditional Peering with Self-Signed Certificates

Traditional peering and self-signed certificates is the easiest implementation—the certificates from both Steelhead appliances are implemented on the adjacent (peer) Steelhead appliance.

Each Steelhead appliance is preinstalled with a self-signed private key and certificate. Steelhead appliances can secure the inner channel, as long as they know and trust each other's certificates, and an SSL handshake can occur.

For details see "Setting Up a Simple SSL Deployment" on page 148.

You do not need the peering lists in this scenario, as both Steelhead appliances trust their adjacent peer Steelhead appliance.

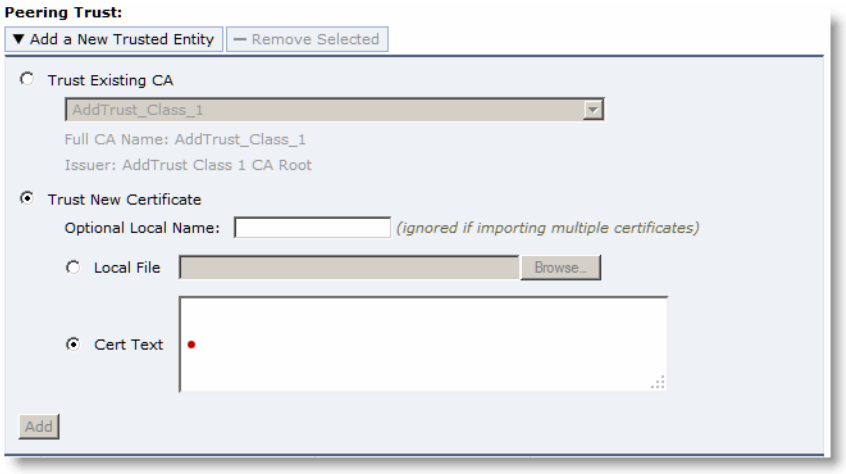
Traditional Peering with CA Signed Certificates

In traditional peering with CA signed certificates, you use your enterprise (or well-known) CAs for all certificates in your domain, and add the CA certificate to each appliance peering trust list. This scenario simplifies configuring secure-peeing trust relationships because each Steelhead appliance automatically trusts a peer Steelhead appliance whose certificate is valid and was signed by the trusted CA.

Note: If you have intermediary certificates, you must add them at the same time as the CA certificate. When using secure peering, both the server-side and the client-side Steelhead appliances are used as client and server (enhanced key usage). For details, go to https://supportkb.riverbed.com/support/index?page=content&id=S14925.

To configure traditional peering with CA signed certificates

1. Choose to Configure > Optimization > Secure Peering (SSL) section.
2. Select on Add a New Trusted Entity.
3. Select Trust New Certificate.
4. Copy and paste the CA certificate (as opposed to each of the other Steelhead appliance's self-signed certificates) in the Cert Text box.
5. Click **Add**.



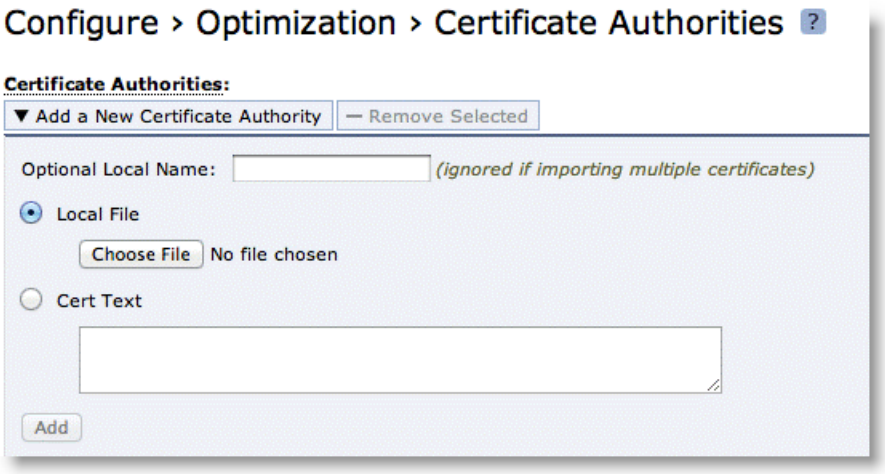
You can accomplish the same results if you use the following procedures.

To add the CA certificate in the Certificate Authority list on each Steelhead appliance

1. Choose Configure > Optimization > Certificate Authorities.
2. Select Add a New Certificate Authority.
3. Select Cert Text.
4. Copy and paste the CA certificate into the text box.

If you have the certificate in a file, you can select Local File, click **Choose File** to browse to the CA certificate file.

5. Click **Add**.

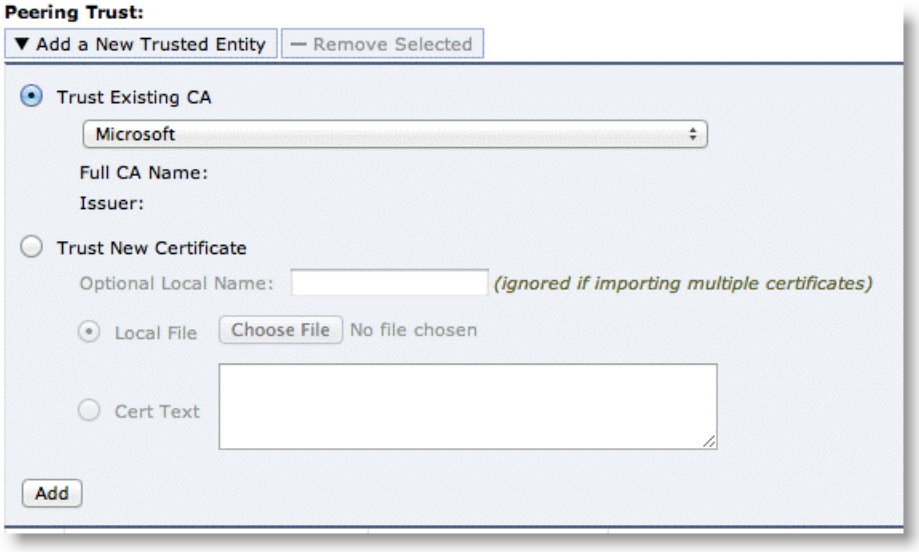


The CA is added to the CA list and does not automatically trust the CA for secure peering connections.

To add the new CA as a trusted entity

1. Choose Configure > Optimization > Secure Peering (SSL).
2. Select Add a New Trusted Entity.
3. Select Trust Existing CA and from the drop-down list.
4. Select the newly added CA.
5. Click **Add**.

The Steelhead appliance certificates is signed by this particular CA, and is now trusted for secure peering.



Managing Steelhead Appliance Trusts with a CMC

Riverbed recommends that you configure a large SSL deployment with the help of a CMC to simplify SSL configuration. The CMC can enable an SSL deployment to scale to an arbitrary size.

A production SSL deployment requires more configuration steps to provide a secure SSL trust model. The configuration complexity arises for the following reasons:

- Because there are typically more than two Steelhead appliances involved, there are more trust relationships to manage.
- In order to preserve the existing trust model, you must be certain that the proxy certificates you install on the server-side Steelhead appliances are valid and trusted by the clients. Self-signed certificates do not generally not meet the trust requirement.
- You must be sure to install the proxy certificates on all the server-side Steelhead appliances that might be optimizing connections to your servers.
- Do not prepare a specific proxy certificate for each server if you can prepare a wild card certificate that is suitably valid, trusted, and matches the names of the target servers.
- You can incorporate other features: for example, you might encrypt the RiOS data store in the branch office or in both the branch office and the data center. Change the secure vault password on the server-side Steelhead appliance to protect the certificate and private key. If you do not change the factory password on the secure vault, you do not need enter a password.

In an SSL deployment consisting of 100 branch sites all linked to a single data center that hosts an SSL server, each branch Steelhead appliance must have a copy of the peer certificate for the server-side Steelhead appliance, and the server-side Steelhead appliance needs a peer certificate for each branch Steelhead appliance. If you configure the system as described in "Setting Up a Simple SSL Deployment" on page 148, you must cut and paste the peer certificate into each Steelhead appliance.

The CMC provides streamlined methods of installing all of the certificates. Some of these reduce the cost of adding one Steelhead appliance or location to the deployment and others involve a fixed startup cost, but allow for a virtually zero-cost scale-out.

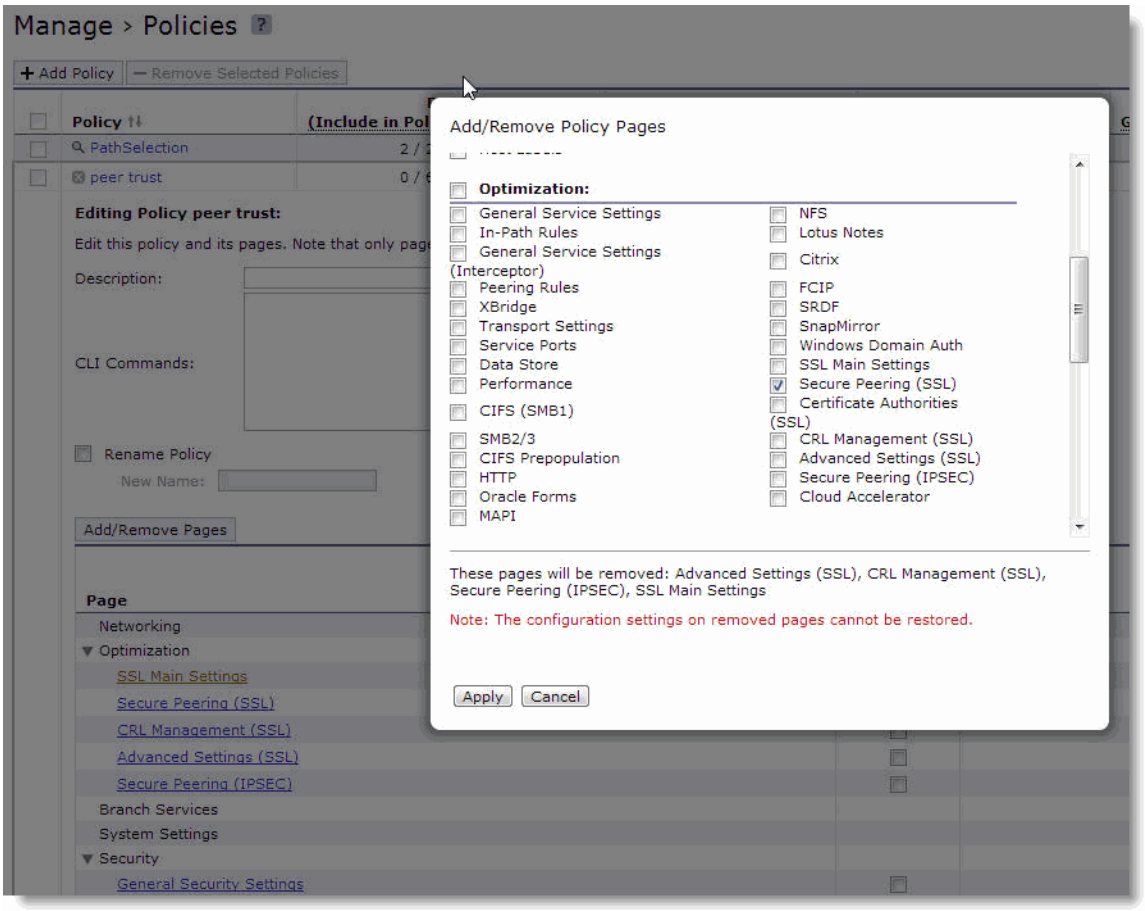
Deployment Example—Managing Steelhead Appliance Trust with a CMC

In this example, you create a policy on the CMC that automatically configures each Steelhead appliance with the peer certificates on all the other Steelhead appliances. Thus, after the initial CMC-based configuration, each Steelhead appliance now trusts all the other Steelhead appliances in the deployment.

To manage Steelhead appliance trust with a CMC

1. Choose Manage > Policies, and select Create New Policy.
2. Enter the policy name into the Policy Name box, and click **Add**.
3. Select Optimization for policy type.
4. Select the peer trust name to display a list of the policy pages.

Figure 10-15. Using A CMC To Manage Trust



5. Select SSL Peering to enable the policy page to take effect for a policy push.
6. Click **Apply**.
7. Select the SSL Peering policy page.

8. Select either specific Steelhead appliances to trust each other, or select Trust All Peers.

Figure 10-16. SSL Peering Page



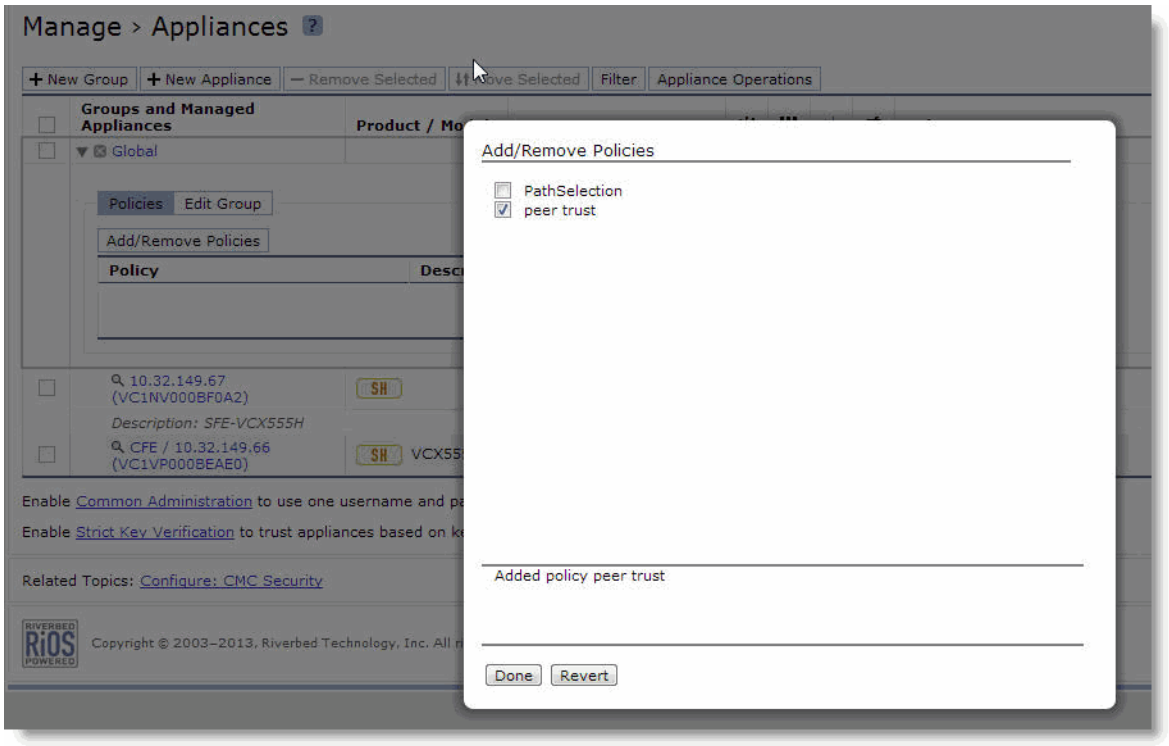
9. Click **Update**.

10. Choose Manage > Appliances.

11. Select Global to display the Edit Group Global page.

12. Select peer trust from the Optimization Policy drop-down list.

Figure 10-17. Global Appliance Option



13. Select Appliance Operations, select Push Policies from the drop-down list, select Global, and click **Push**.

    When the policy push succeeds, all the Steelhead appliance peers now trust each other. When you deploy a new Steelhead appliance, it automatically inherits the peer trust policy, and is included in the *Web of trust* after the policy is updated across all the Steelhead appliances. You must push the policy again to have the new Steelhead appliance receive all of the peering trust certificates from other appliances. You must push the policy again if any the Steelhead appliances peering certificates are updated on either the CMC or the Steelhead appliance.

    Note: You can check whether the policy push has succeeded by checking the Operations History page.

14. To complete the SSL configuration, on the client-side Steelhead appliances, choose Configure > Optimization > SSL Main Settings, select Enable SSL Optimization, and then install the server certificates and private keys on the server-side Steelhead appliance.

    You can do this manually, or you can use the CMC.

Note: For details on the CMC, see the *Riverbed Central Management Console User's Guide* and the *Steelhead Appliance Deployment Guide*.

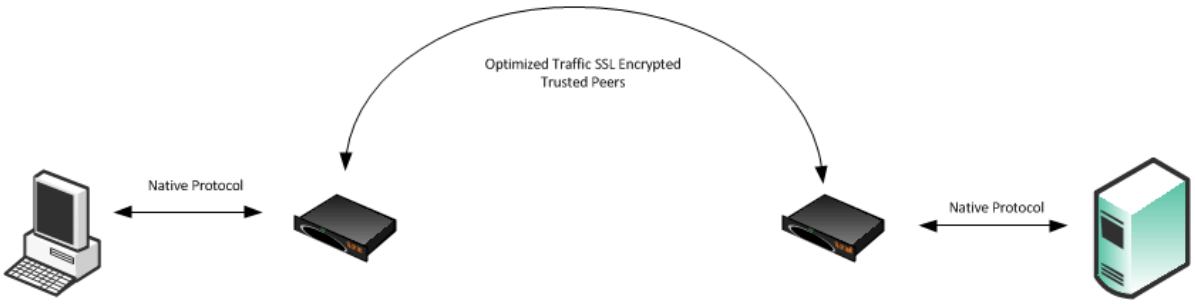
Deploying Secure Steelhead Appliance Peering

SSL secure peering ensures that all data in remote RiOS data store is on a trusted appliance. In a high-security environments, you might want only known and trusted Steelhead appliances to peer for optimized flows. Prior to RiOS v6.0, you could control peering only by modifying the peering tables, which was limiting because the default behavior was to optimize traffic. Modifying the peering tables creates situations in which Steelhead appliances can peer and optimize flows that might not be explicitly trusted. Using SSL secure peering, and forcing all traffic flows to flow through SSL, you can create an environment of explicit trust—only Steelhead appliances that have SSL peering certificates can peer and optimize flows.

In federated networks, it is common for organizationally separate Steelhead appliances to be aware of each other. If you use only secure Steelhead appliance peering, this forces all communication between the Steelhead appliances to be SSL. If the connection fails, then it causes all conversations between Steelhead appliances to pass through without being intercepted. A secure Steelhead appliance peering environment can only be created when you use trusted peering certificates to optimize with one another. For more information about setting up peering certificates, see "Setting Up a Simple SSL Deployment" on page 148 or "Steelhead Appliance Secure Peering Scenarios" on page 154.

Figure 10-18 shows the client on the left starting a conversation with the server on the right with CIFS. The default behavior of the Steelhead appliance is to optimize this connection. In the secure peering infrastructure, the Steelhead appliance sets up an SSL session for the inner channel and peer only with Steelhead appliances that have an SSL peering trust. To prevent any peering between Steelhead appliances without a trusted SSL peering certificate, complete the following steps.

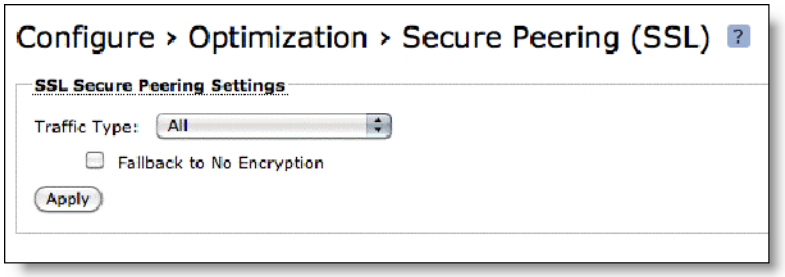
Figure 10-18. Trusted Peers



To ensure all optimized Steelhead appliance communications are secure

1. Chose Configure Optimization > Secure Peering (SSL) in the Management Console.
2. Select All from the Traffic Type drop-down list.
3. Ensure Fallback to No Encryption is not selected.

Figure 10-19. Secure Peering (SSL) Page



4. Click **Apply**.

If you select All from the Traffic Type you can cause up to a 10% performance decline in higher-capacity Steelhead appliances. Take this performance metric into account when sizing a complete secure Steelhead appliance peering environment.

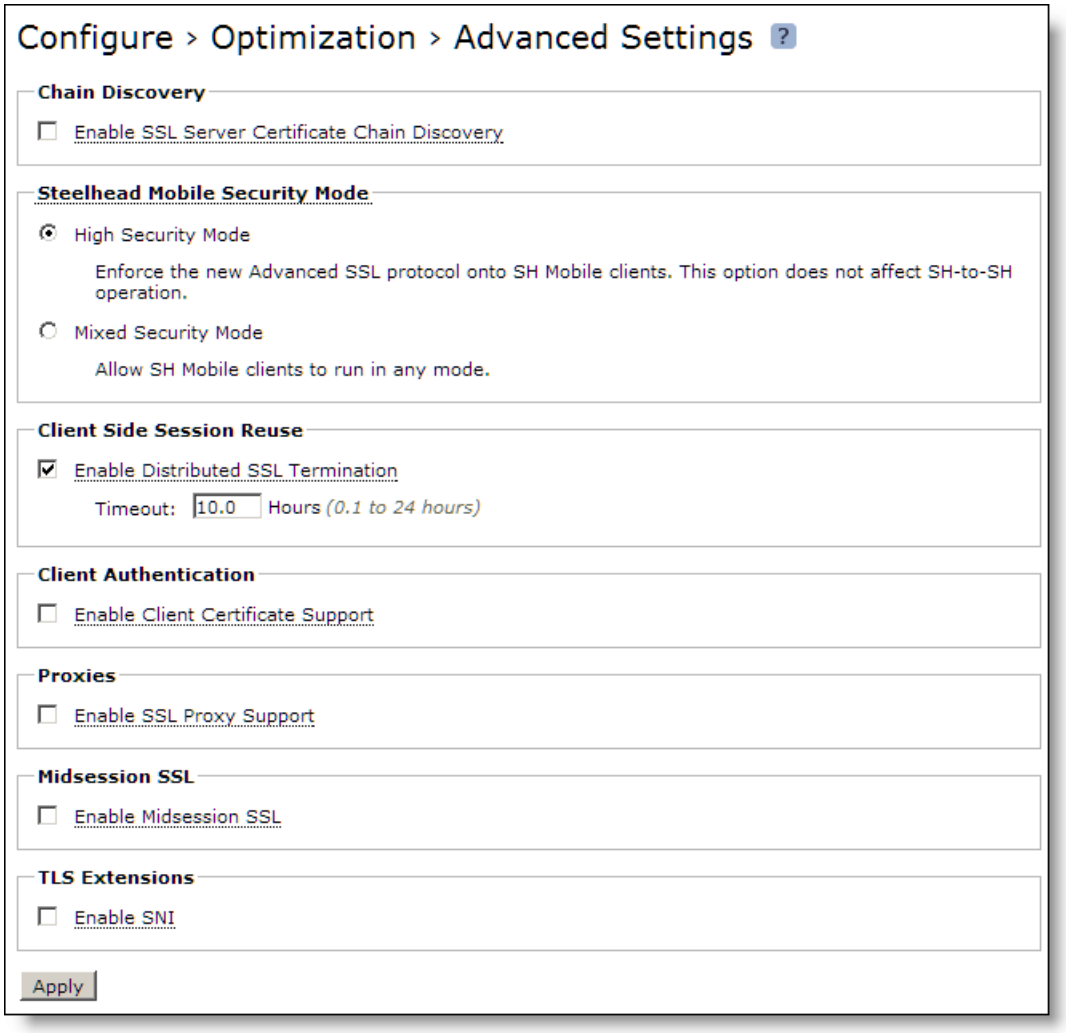
Important: Riverbed recommends that you use NTP time synchronization, instead of manually synchronizing the clocks, on both the server-side and client-side Steelhead appliances. It is critical that the peer Steelhead appliance time is the same for the trust relationship to work. Choose Configure > System Settings > Date and Time to display the Host Settings page. Under Date and Time, click **Use NTP Time Synchronization**.

Advanced SSL Features

This section describes the advanced SSL feature set:

- "Client Certificate Support" on page 164
- "Proxy Server Support" on page 166
- "Mid-Session SSL Support" on page 166

Figure 10-20. Advanced Settings Page



Client Certificate Support

Client certificate support enables acceleration of SSL traffic to SSL servers that authenticate SSL clients. The SSL server verifies the SSL client certificate. In the client authentication SSL handshake, each client has a unique client certificate and the SSL server, in most cases, maintains the state that is specific to each client when answering the client's requests. The SSL server must receive exactly the same certificate that is originally issued for a client on all the connections between the client and the server. Typically the client's unique certificate and private key are stored on a smart card, such as a Common Access Card (CAC), or on a similar location that is inaccessible to other devices on the network.

Enabling the client authentication feature allows Steelhead appliances to compute the encryption key and the SSL server continues to authenticate the original SSL client exactly as it would without the Steelhead appliances. The server-side Steelhead appliance observes the SSL handshake messages as they go back and forth. With access to the SSL server's private key, the Steelhead appliance computes the session key exactly as the SSL server does. The SSL server continues to perform the actual verification of the client, so any dependencies on the uniqueness of the client certificate for correct operation of the application are met. Because the Steelhead appliance does not modify any of the certificates (or the handshake messages) exchanged between the client and the server, there is no change to their trust model. The client and server continue to trust the same set of CAs as they did without the Steelhead appliances accelerating their traffic.

If the data center has a mixed environment with a few SSL servers that authenticate clients along with those that do not authenticate clients, Riverbed recommends enabling client authentication.

There are a few caveats when using client-side certificates:

- Both the client-side and the server-side Steelhead appliances must be running RiOS v6.5. or later.
- Enable client certificate support on the server-side Steelhead appliance.
- The server-side Steelhead appliance must have access to the exact private key used by the SSL server.
- You must configure the SSL server to ask for client certificates.
- The Steelhead appliance must have a compatible cipher chosen by the server.
- SSL sessions that reuse previous sessions that are unknown to the Steelhead appliance cannot be decrypted.
- Client-side certificates with renegotiation handshakes are not supported.
- Client-side certificate supports the RSA key exchange only. It does not support the Diffie-Hellman key exchange.

To enable client authentication

1. Perform the basic steps to enable SSL optimization. For details, see "Setting Up a Simple SSL Deployment" on page 148.

2. On the server-side Steelhead appliance, choose Configure > Optimization > Advanced Settings, select Enable Client Certificate Support, and click **Apply**.

3. Choose Configure > Optimization> SSL Main Settings, import the SSL server private key and certificate, and click **Save** to save the configuration. You do not need to restart the optimization service.

Note: Riverbed recommends that you use strong security ciphers, such as AES. Avoid stream ciphers.

Verification

To verify client authentication, on the server-side Steelhead appliance, check the Discovered Server (Optimizable) table on the SSL Main Settings page. Optimizable servers that are using client authentication appear as optimizable streams. For servers that are not using client authentication, the server appears in the Discovered Server (bypassed, not optimizable streams) table with the reason *No proxy certificate configured for the server*.

Proxy Server Support

RiOS v7.0 or later includes support for enterprise proxy servers that route Web traffic (including secure SSL Web traffic) on behalf of clients to a secure SSL server.

For more information, including a configuration example, see"Configuring HTTP SSL Proxy Interception" on page 71.

Mid-Session SSL Support

RiOS v7.0 or later supports mid-session SSL support. A normal SSL session starts with an SSL handshake. The rest of the session is encrypted using session keys until the session is torn down. There are some scenarios where the session starts unencrypted, but might need to encrypt certain traffic during the session. In these scenarios, an SSL handshake occurs mid-session to secure the specific traffic.

An example mid-session SSL is START Transport Layer Security (TLS) during SMTP sessions. The client starts an unencrypted SMTP session on port 25, to the server. The server accepts the unencrypted session and indicates to the client that it supports STARTTLS. In the middle of session, the client initiates STARTTLS. Normal TLS negotiations resume and the rest of the conversation is encrypted. For more details, see http://en.wikipedia.org/wiki/STARTTLS.

RiOS v8.6 and later supports TLS versions v1.0, v1.1, and v1.2. Note that the Steelhead appliance can negotiate to earlier versions of TLS if the peer Steelhead appliance uses a RiOS release prior to v8.6.

Figure 10-21. Example STARTTLS During SMTP Session

```
S: <waits for connection on TCP port 25>
C: <opens connection>
S: 220 mail.example.org ESMTP service ready
C: EHLO client.example.org
S: 250-mail.example.org offers a warm hug of welcome
S: 250 STARTTLS
C: STARTTLS
S: 220 Go ahead
C: <starts TLS negotiation>
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO client.example.org[3]
. . .
```

Source: http://en.wikipedia.org/wiki/STARTTLS (May 3, 2012)

To enable mid-session SSL support

1. Choose Configure > Optimization > Advanced Settings.
2. Select Enable Midsession SSL.
3. Click **Apply**.

Server Name Indication

Select server name indication (SNI) on the server-side Steelhead appliance while using name-based virtual hosts with SSL. SNI is an extension to the SSL protocol. With SNI, the first SSL client hello handshake message sent to the HTTPS server includes the requested virtual hostname to which the client is connecting. Because the server is aware of the hostname, it returns a host-specific security certificate.

Without SNI, an HTTPS server returns a default certificate that satisfies hostnames for all virtual hosts. The SSL connection setup uses the default virtual host configuration for the address where the connection was received. Browser messages warn that certificates have the wrong hostname. With SNI enabled, RiOS provides the hostname. This enables the server to determine the correct named virtual host for the request and to set up the connection accordingly from the start.

The browser validates the certificate names against the requested URL, and the server-side Steelhead appliance verifies that the selected proxy certificate is compatible with the client hostname. This ensures that the browser does not reject the proxy certificate for the server-side Steelhead appliance.

If SNI provides a hostname that does not exactly match the common name or any of the subject alternate names for the certificate on the server-side Steelhead appliance, the system determines that a valid certificate is not present and bypasses that hostname. No configuration is necessary on the client-side Steelhead appliance.

The client browser must also support SNI.

Steelhead Mobile SSL High-Security Mode

Steelhead Mobile v2.0x or later provides a high-security mode that accelerates SSL-encrypted traffic while maintaining the preferred trust model. Using high-security mode, Steelhead Mobile can apply data streamlining, transport streamlining, and application streamlining mechanisms to SSL-encrypted traffic while keeping all private keys within the data center and without requiring fake certificates in branch offices. High-security mode is enabled by default.

In high-security mode, Steelhead Mobile uses two CA certificates. The Mobile Controller CA resides on the Controller and signs the certificate on the inner channel. The other CA certificate is self-signed and is generated by and resides on each client. Steelhead Mobile ensures that this self-signed certificate is trusted by Internet Explorer, Safari and Firefox. This Steelhead Mobile Internal CA Certificate to generates and signs proxy certificates dynamically.

The authentication process is essentially identical to how a Steelhead appliance verifies the identity of other client-side Steelhead appliances: CAs verify the identity of the certificates presented to the Steelhead appliances. However, when the Mobile Controller does not want a client to optimize SSL traffic (the equivalent of placing the CA in the black peering list on the Steelhead appliance), it assigns the client a package and acceleration policy that does not have SSL enabled. (You can also disable SSL on a client-side appliance if you do not want it to optimize SSL traffic.)

The following sequence of operations occur before Steelhead Mobile optimizes a connection:

1. Steelhead Mobile automatically adds the Steelhead Mobile CA Certificate into the Windows Cert store (used by Internet Explorer) and into the Firefox browser trusted CA list. On Mac OS X, this is added to the Keychain (used by Safari).

2. The browser makes a connection to the server, the connection is intercepted, and splices are set up.

3. The browser initiates an SSL handshake by sending a *client_hello* message.

4. Steelhead Mobile establishes a secure inner channel over the existing inner TCP connection using the certificate signed by the Controller CA.

5. Steelhead Mobile sends the *begin_handshake(cn)* message to the server-side Steelhead appliance.

6. The server-side Steelhead appliance performs a handshake with the server and sends the server's begin_handshake(cn) to Steelhead Mobile.

7. Steelhead Mobile receives the server's begin_handshake(cn) and generates a proxy server certificate signed by the internal Steelhead Mobile CA.

8. Steelhead Mobile completes a handshake with the browser.

Three separate secure connections are now established and traffic is now optimized.

Note: For instructions on how to enable high-security mode for Steelhead Mobile, see the *Steelhead Mobile Controller User's Guide* and the *Steelhead Appliance Deployment Guide*.

Troubleshooting and Verification

Use these tools to verify that you have configured SSL support correctly:

- **SSL Optimization** - After completing the SSL configuration on both Steelhead appliances and restarting the optimization service, access the secure server from the Web browser. These events take place in a successful optimization:
  - If you specified a self-signed proxy certificate for the server on the server-side Steelhead appliance, a pop-up window appears on the Web browser. View the certificate details to ensure that it is the same as the certificate on the server-side Steelhead appliance.
  - In the Management Console, the Current Connections report lists the new connection as optimized without a Protocol Error flag.
  - In the Management Console, the Traffic Summary report displays encrypted traffic (typically, HTTPS).
  - Verify that the back-end server IP appears in the SSL Discovered Server Table (Optimizable) in the SSL Main Settings page.

Note: Because all the SSL handshake operations are processed by the server-side Steelhead appliance, all the SSL statistics are reported on the server-side Steelhead appliance. No SSL statistics are reported on the client-side Steelhead appliance.

- **Monitoring SSL Connections** - Use these tools to verify SSL optimization and to monitor SSL progress:
  - On the client Web browser, click the **Lock** icon to obtain certificate details. The certificate must match the proxy certificate installed on server-side Steelhead appliance.

  - In the Current Connections report, verify the destination IP address, port 443, the Connection Count as Established (three yellow arrows on the left side of the table), SDR Enabled (three cascading yellow squares on the right side of the table), and that there is no Protocol Error (a red triangle on the right side of the table).
  - In the SSL Statistics report (on the server-side Steelhead appliance only) look for connection requests (established and failed connections), connection establishment rate, and concurrent connections.
- **Monitoring Secure Inner Channel Connections** - Use these tools to verify that secure inner channels are in use for the selected application traffic types:
  - In the Current Connections report, look for the **Lock** icon and three yellow arrows, which indicate the connection is encrypted and optimized. If the Lock icon is not visible or is dimmed, click the magnifying glass to view a failure reason that explains why the Steelhead appliance is not using the secure inner channel to encrypt the connection. If there is a red protocol error, click the magnifying glass to view the reason for the error.
  - Search the client-side and server-side Steelhead appliance logs for ERR and WARN.
  - Check that both Steelhead appliances appear in the white peering trust list on the client-side and server-side Steelhead appliances, indicating that they trust each other.

If you are experiencing issues with your SSL traffic being optimized, see the Riverbed Knowledge Base article *Troubleshooting your SSL Configuration* at http://supportkb.riverbed.com/support/index?page=content&id=S15107.

Interacting with SSL-Enabled Web Servers

This section describes how to obtain the server certificate and private key on two Web servers: Apache and Microsoft IIS. This section includes the following topics:

- "Obtaining the Server Certificate and Private Key" on page 169
- "Generating Self-Signed Certificates" on page 171

Obtaining the Server Certificate and Private Key

SSL is a protocol that enables the underlying application to transmit data securely over an insecure network. At the very foundation of SSL is the assumption that an authenticated party (for example, a Web server) has exclusive access to its private key. If any other entity has this private key, it can mount a man-in-the-middle attack on a connection to the authenticated party.

Steelhead appliances optimize SSL traffic when you configure the server-side Steelhead appliance with the server's certificate and private key, which enables it to intercept all SSL connections to the server.

Apache Certificates and Private Keys

The following procedures explain how to locate the Apache server certificate and private key and import them into the server-side Steelhead appliance.

To obtain the server certificate and private key from an Apache-based Web server

1. Locate the Apache httpd.conf configuration file.

2. Look through the file for lines beginning with SSLCertificateFile and SSLCertificateKeyFile, for example:

```
SSLCertificateFile /etc/foo/bar/server.crt
SSLCertificateKeyFile /etc/foo/bar/server.key
```

The filename following SSLCertificateFile is the server certificate. The filename following SSLCertificateKeyFile is the server private key. After you locate these files, you can import them into the server-side Steelhead appliance configuration.

To import the certificate and private key

1. On the server-side Steelhead appliance, choose Configure > Optimization > SSL Main Settings in the Management Console.
2. Select Add a New SSL Certificate.
3. Select Import Existing Private Key and CA-Signed Public Certificate (Two Files in PEM or DER formats).
4. Under Import Private Key, select Local File, click **Browse**, and go to the certificate key file.
5. Under Import Public Certificate, select Local File, click **Browse**, and go to the server certificate file.
6. Click **Add**.

IIS Certificates and Private Keys

The following procedures explain how obtain the server certificate and private key from a IIS Web server and import them into the server-side Steelhead appliance.

To obtain the server certificate and private key from an IIS Web server

1. From the Windows Start > Run menu, enter **mmc** to launch the Microsoft Management Console (MMC).
2. Within the IIS snap-in, go through the tree to the Web server in question. (If the IIS snap-in does not exist, choose File > Add/Remove Snap-in, select the Web server, and click **Add**.)
3. Right-click the server item and select Properties.
4. Select the Directory Security tab.
5. Select View Certificate.
6. Select the Details tab.
7. Select Copy to File.
8. Select Yes, export private key.

Both the certificate and the private key are now stored in a single file with the filename you specified. The filename ends with the .pfx extension.

To import the certificate and private key

1. On the server-side Steelhead appliance, choose Configure > Optimization > SSL Main Settings in the Management Console.
2. Under SSL Server Certificates, select Add a New SSL Certificate.
3. Select Import Existing Private Key and CA-Signed Public Certificate (One File in PEM or PKCS12 formats).
4. Under Import Single File, select Local File, click **Browse**, and go to the .pfx file.

   Note: Because the file .pfx file is not scrambled with a password, you can leave the Decryption Password field blank.

5. Click **Add**.

Generating Self-Signed Certificates

In certain situations you might not want to, or might not be able to, use the server's real private key. If that is the case, you can generate a self-signed certificate and private key pair for the server and install them on the server-side Steelhead appliance. This certificate is not signed by any real certificate authority, but is instead signed by the private key itself, and is thus called a self-signed certificate.

During SSL connection establishment, when the server-side Steelhead appliance presents the self-signed certificate to the client (for example, a Web browser), the client cannot verify the authenticity of the certificate. From the client's point of view, security might have been compromised, and the user is typically alerted with a message to this effect.

Generating Self-Signed Certificates with Apache

A typical SSL-enabled Apache installation comes with a utility called OpenSSL, which you can use to generate the self-signed certificate. Enter the following command:

```
$ openssl req -new -x509 -nodes -out server.crt -keyout server.key
```

This adds two files to the current directory, server.crt and server.key. These files correspond to the certificate and the private key, respectively. The next step is to import the files into the server-side Steelhead appliance configuration.

To import the certificate and private key

1. On the server-side Steelhead appliance, Steelhead appliance, choose Configure > Optimization > SSL Main Settings in the Management Console.
2. Under SSL Server Certificates, select Add a New SSL Certificate.
3. Select Import Existing Private Key and CA-Signed Public Certificate (Two Files in PEM or DER formats).

   Because the file server.key is not scrambled with a password, you can leave the Decryption Password field blank.

4. Click **Add**.

Generating Self-Signed Certificates with IIS

If you want to generate a self-signed certificate for an IIS-based Web server, you have two options.

To generate self-signed certificates with IIS

- Install Cygwin and include the OpenSSL package in the installation. This gives you access to the OpenSSL utility from X.2.1, which you can use to generate the certificate and private key. To install, go to http://www.cygwin.com/.

  —or—

- Download and install a set of IIS 6.0 Resource Kit Tools from Microsoft at http://www.microsoft.com/en-us/download/details.aspx?id=17275.

  This package contains a utility called SelfSSL which you can use to generate a self-signed certificate and private key for a Web server. SelfSSL also automatically installs the certificate for that Web server instance of IIS, so you must follow the steps in X.1.2 to extract the certificate into a file.

  SelfSSL replaces an existing certificate for a Web server instance.

CHAPTER 11 Configuring SCEP and Managing CRLs

This chapter describes how to configure the Simple Certificate Enrollment Protocol (SCEP) and how to manage Certification Revocation Lists (CRLs) using the Riverbed CLI. This chapter includes the following sections:

- "Using SCEP to Configure On-Demand and Automatic Re-Enrollment" on page 173
- "Managing Certificate Revocation Lists" on page 177

This section makes the following assumptions:

- You have configured SSL on the Steelhead appliance (for details, see "SSL Deployments" on page 141).
- You have set up a SCEP server.

Using SCEP to Configure On-Demand and Automatic Re-Enrollment

SCEP is for securely issuing and revoking digital certificates in a simple, scalable manner on network devices. The Steelhead appliance uses SCEP to configure on-demand enrollment and automatic re-enrollment of SSL peering certificates.

Note: Currently, the Steelhead appliance can only enroll peering certificates.

This section describes how to configure on-demand and automatic re-enrollment of SSL peering certificates.

The following table summarizes the SCEP commands.

SCEP Commands	Parameters	Definition
secure-peering scep auto-reenroll	**enable**	Enables automatic re-enrollment of a certificate to be signed by a CA.
	exp-threshold <num of days>	Specify the amount of time (in days) to schedule re-enrollment before the certificate expires.
	last-result clear-alarm	Clears the automatic re-enrollment last-result alarm. The last result is the last completed enrollment attempt.

SCEP Commands	Parameters		Definition
secure-peering scep max-num-polls	**<max number polls>**		Specify the maximum number of polls before the Steelhead appliance cancels the enrollment. The peering certificate is not modified. The default value is **5**. A poll is a request to the server for an enrolled certificate by the Steelhead appliance. The Steelhead appliance polls only if the server responds with **pending**. If the server responds with **fail** then the Steelhead appliance does not poll.
secure-peering scep on-demand cancel	**None**		Cancels any active on-demand enrollment.
secure-peering scep on-demand gen-key-and-csr	**rsa**		Generates a new private key and CSR for on-demand enrollment using the Rivest-Shamir-Adleman algorithm.
		state <string>	Specify the state. No abbreviations allowed.
		org-unit <string>	Specify the organizational unit (for example, the department).
		org <string>	Specify the organization name (for example, the company).
		locality <string>	Specify the city.
		email <email addr>	Specify an email address of the contact person.
		country <string>	Specify the country (2-letter code only).
		common-name <string>	Specify the hostname of the peer.
		key-size <512 | 1024 | 2048>	Specify the key size in bits (for example, 512, 1024, 2048).
secure-peering scep on-demand start			Starts an on-demand enrollment (in the background by default).
	foreground		Starts an on-demand enrollment in the foreground.
secure-peering scep passphrase	**<pass phrase>**		Specify the challenge password phrase.
secure-peering scep poll-frequency	**<minutes>**		Specify the poll frequency in minutes. The default value is **5**.
secure-peering scep trust	**peering-ca <peer ca>**		Specify the name of the existing peering CA.
secure-peering scep url	**<url>**		Specify the URL of the SCEP responder. Use the following format: http://host[:port/path/to/service].

Configuring On-Demand Enrollment

The following example configures the most common on-demand enrollment SCEP settings.

Note: You can only perform one enrollment of a certificate at a time. You must stop enrollment before you begin the enrollment process for another certificate.

To configure on-demand enrollment of certificates

1. To configure SCEP settings, connect to the Steelhead CLI and enter the following commands:

```
enable
configure terminal
secure-peering scep url <http://host[:port/path/to/service>
secure-peering scep trust peering-ca < name of a peering CA >
secure-peering scep poll-frequency 10
secure-peering scep max-num-polls 6
secure-peering scep passphrase “device unique passphrase”
```

2. To perform an on-demand enrollment you must first generate a new key and Certificate Signing Request (CSR), at the system prompt enter the command:

```
secure-peering scep on-demand gen-key-and-csr rsa 1048 country us org mycompany org-unit
engineering
```

3. To display the CSR (including the fingerprint), at the system prompt enter the command:

```
show secure-peering scep peering on-demand csr
```

4. To start an on-demand enrollment, at the system prompt enter the command:

```
secure-peering scep on-demand start
```

5. To view current status and the result of the last attempt (since boot), at the system prompt enter the following commands:

```
show secure-peering scep enrollment status
show secure-peering scep on-demand last-result
```

6. To stop enrollment, at the system prompt enter the following command:

```
secure-peering scep on-demand cancel
show secure-peering scep on-demand last-result
```

   You must stop enrollment before you can begin the enrollment process for another certificate.

Configuring Automatic Re-Enrollment

The following example configures the most common automatic re-enrollment SCEP settings.

To configure automatic re-enrollment of certificates

1. To configure SCEP settings, connect to the Steelhead CLI and enter the following command:

```
enable
configure terminal
secure-peering scep url http://entrust-connector/cgi-bin/pkiclient.exe
secure-peering scep trust peering-ca < name of a peering CA >
```

```
secure-peering scep poll-frequency 10
secure-peering scep max-num-polls 6
secure-peering scep passphrase "device unique passphrase"
```

2. To configure automatic re-enrollment, at the system prompt enter the following commands:

```
secure-peering scep auto-reenroll exp-threshold 30
secure-peering scep auto-reenroll enable
```

3. To view current automatic re-enrollment settings, at the system prompt enter the following commands:

```
show secure-peering scep peering auto-reenroll csr
show secure-peering scep peering on-demand last-result
```

Viewing SCEP Settings and Alarms

This section describes how view SCEP settings and alarms.

The following table summarizes the commands for SCEP settings.

Command	Parameters	Definition
show secure-peering scep	**None**	Displays SCEP information.
show secure-peering scep auto-reenroll	**csr**	Displays the automatic re-enrollment CSR.
	last-result	Displays the result of the last completed automatic re-enrollment.
show secure-peering scep ca	**<ca name> certificate**	Displays a specified SCEP peering CA certificate.
show secure-peering scep enrollment status	**None**	Displays enrollment status information.
show secure-peering scep on-demand	**csr**	Displays on-demand enrollment information.
	last-result	Displays result of the last completed on-demand enrollment.

An SCEP alarm is triggered when the Steelhead appliance requests a SCEP server to dynamically re-enroll an SSL peering certificate and the request fails. The Steelhead appliance uses SCEP to dynamically re-enroll a peering certificate to be signed by a certificate authority. The alarm clears automatically when the next automatic re-enrollment succeeds.

To view SCEP alarm status

1. Connect to the Steelhead CLI and enter enable mode.

2. Enter the following the command:

```
show stats alarm ssl_peer_scep_auto_reenroll
Alarm ssl_peer_scep_auto_reenroll:
    Enabled:                  yes
    Alarm state:              ok
    Rising error threshold:   no
    Rising clear threshold:   no
    Falling error threshold:  no
    Falling clear threshold:  no
    Rate limit bucket counts: { 5, 20, 50 }
    Rate limit bucket windows: { 3600, 86400, 604800 }
```

```
Last checked at:           2009/07/30 17:43:07
Last checked value:        true
Last event at:
Last rising error at:
Last rising clear at:
Last falling error at:
Last falling clear at:
```

To clear the SCEP alarm

1. Connect to the Steelhead CLI and enter configuration mode.
2. Enter the following the command:

```
secure-peering scep auto-reenroll last-result clear-alarm
```

Managing Certificate Revocation Lists

Certificate Revocation Lists allow CAs to revoke issued certificates (for example, when the private key of the certificate is compromised).

Note: CRL s are not used by default in the Steelhead appliance.

A CRL is a database that contains a list of digital certificates that have been invalidated before their expiration date, including the reasons for the revocation, and the names of the issuing certificate signing authorities. The CRL is issued by the CA which issues the corresponding certificates. All CRLs have a lifetime during which they are valid (often 24 hours or less).

CRLs are used when a:

- server-side Steelhead appliance verifies the certificate presented by the server in the SSL handshake between the server-side Steelhead appliance and the server.
- server-side Steelhead appliance verifies the certificate presented by the client-side Steelhead appliance in the handshake between the two Steelhead appliances for establishing a secure inner channel over the WAN.
- client-side Steelhead appliance verifies the certificate presented by the server-side Steelhead appliance in the handshake between the two Steelhead appliances for establishing a secure inner channel over the WAN.

Note: Currently, the Steelhead appliance only supports downloading CRLs from Lightweight Directory Access Protocol (LDAP) servers.

The following table summarizes CRL CLI management commands.

CRL Commands	Parameters		Definition
protocol ssl crl ca			Configures CRL for automatically discovered CAs. You can update automatically discovered CRLs using this command.
	<ca name>		Specify the name of a SSL CA certificate.
	cdp <integer>		Specify an integer index of a Cisco Distribution Point (CDP) in a CA certificate. The **no protocol ssl crl ca * cdp *** command option removes the update.
		ldap server <IP addr or hostname>	Specify the LDAP server IP address or hostname to modify a CDP URI.
		port <port>	Optionally, specify the LDAP service port.
		crl-attr-name <attr-name>	Optionally, specify the attribute name of CRL in a LDAP entry.
protocol ssl crl cas enable			Enables CRL polling and use of CRL in handshake verifications of CA certificates. Enabling CRL allows the CA to revoke a certificate. For example, when the private key of the certificate is compromised, the CA can issue a CRL that revokes the certificate.
protocol ssl crl handshake			Configures handshake behavior for a CRL.
	fail-if-missing		If a relevant CRL cannot be found the handshake fails.
[no] protocol ssl crl manual	**ca**		Specify the CA name to manually configure the CDP. The **no protocol ssl crl manual** command removes manually configured CDPs.
		uri <uri>	Specify the complete CDP URI to manually configure the CDP for the CA.
	peering ca		Specify the CA name to manually configure the CDP for the peering CA.
		uri <uri>	Specify the complete CDP URI to manually configure the CDP for the peering CA.

CRL Commands	Parameters		Definition
protocol ssl crl peering	**ca <ca name>**		Configures a CRL for an automatically discovered peering CA.
		cdp <integer>	Specify an integer index of a cdp in a peering CA certificate. The **no protocol ssl crl peering ca * cdp *** removes the update.
		ldap server <ip-addr or hostname> <cr>	Specify the IP address or hostname of a LDAP server.
		crl-attr-name <string> | port <port num>	Optionally, specify an attribute name of CRL in a LDAP entry.
		port <port num>	Optionally, specify the LDAP service port.
	cas enable		Enables CRL polling and use of CRL in handshake verification.
protocol ssl crl query-now	**ca <string> cdp <integer>**		Download CRL issued by SSL CA. Specify the CA name and CDP integer.
	peering ca <ca name> cdp <integer>		Download CRL issued by SSL peering CA. Specify the CA name and CDP integer.
show protocol ssl crl	**ca <ca name>**		Display current state of CRL polling of a CA.
	crl cas <cr> | crl-file <string> text		Display the CRL in text format version.
	crl peering ca <ca name> | cas crl-file <string> text		Display current state of CRL polling for peering
	crl report ca <ca name> | peering ca <peering ca name>		Display reports of CRL polling from the CA or display reports of CRL polling from peering CA.

Managing CRLs

This section describes how to manage CRLs using the CLI.

To update an incomplete CDP

1. To enable CRL polling and handshakes, connect to the Steelhead CLI and enter configuration mode.
2. Enter the following set commands:

```
protocol ssl crl cas enable
protocol ssl crl peering cas enable
```

3. To view the CRL polling status of all CAs, enter the following command:

```
show protocol ssl crl ca cas
```

```
<<this example lists two CDPs: one complete CDP and one incomplete CDP>>
  CA: Comodo_Trusted_Services
   CDP Index: 1
     DP Name 1: URI:http://crl.comodoca.com/TrustedCertificateServices.crl
     Last Query Status: unavailable
   CDP Index: 2
     DP Name 1: URI:http://crl.comodo.net/TrustedCertificateServices.crl
     Last Query Status: unavailable
<<an incomplete CDP is indicated by the DirName format>>
  CA: Entrust_Client
   CDP Index: 1
     DP Name 1: DirName:/C=US/O=Entrust.net/OU=www.entrust.net/Client_CA_Info/CPS incorp. by
  ref.limits liab./OU=(c) 1999 Entrust.net Limited/CN=Entrust.net Client Certification
Authority
  CN=CRL1
     Last Query Status: unavailable
   CDP Index: 2
     DP Name 1: URI:http://www.entrust.net/CRL/Client1.crl
     Last Query Status: unavailable
```

In this case, the Entrust Client is an incomplete CDP as indicated by **DirName** format. Currently, the Steelhead appliance only supports updates in the **DirName** format.

4. To update the incomplete CDP URI, enter the following commands:

```
protocol ssl crl ca Entrust_Client cdp 1 ldap-server 192.168.172.1
protocol ssl crl peering ca Entrust_Client cdp 1 ldap-server 192.168.172.1
```

5. To view the status of the updated CDP, enter the following command:

```
show protocol ssl crl ca Entrust_Client
```

The status of CRL polling can be either **pending**, **success**, or **error**.

6. To check CRL polling status of all CAs, enter the following command:

```
show protocol ssl crl cas
```

Viewing CRL Alarm Status

This section describes how to view a CRL alarm and how to clear a CRL alarm.

To view CRL alarm status

1. Connect to the Steelhead CLI and enter enable mode.
2. Enter the following the command:

```
show stats alarm crl_error
Alarm crl_error:
    Enabled:                  yes
    Alarm state:              ok
    Rising error threshold:   1
    Rising clear threshold:   1
    Falling error threshold:  no
    Falling clear threshold:  no
    Rate limit bucket counts: { 5, 20, 50 }
    Rate limit bucket windows: { 3600, 86400, 604800 }
    Last checked at:          2009/07/30 17:40:34
    Last checked value:       0
    Last event at:
Last rising error at:
```

```
Last rising clear at:
Last falling error at:
Last falling clear at:
```

To clear a CRL alarm, you must either rectify the problem by updating the incomplete CDP or you must disable CRL polling.

To disable CRL polling and clear a CRL alarm

1. Connect to the Steelhead CLI and enter configuration mode.
2. Enter the following the command:

```
no protocol ssl crl cas enable
```

APPENDIX A RiOS Version Compatibility with Domains and Domain Relationships

Support for full optimization of signed SMB and encrypted MAPI first started in RiOS v5.5. With more recent developments of the configuration and diagnostic features included in RiOS, Riverbed recommends that you use a minimum of RiOS v6.1 for Steelhead appliances in a network where you want to optimize signed SMB and encrypted MAPI traffic. If you need full optimization of signed SMB2 and SMB2.1 traffic, then Riverbed recommends that you use a minimum version of RiOS v6.5 on all Steelhead appliances. If you need support for optimization of SMB2.2 and SMB3 traffic, you must use RiOS v8.5 or later on all Steelhead appliances. RiOS v8.5 is the minimum version that supports AD integration with Windows 2012 domain function level.

With RiOS v8.5 or later, you can use the *Domain Auth Auto Config* widget (Choose Configure > Optimization > Domain Auth Auto Config). The widget automates the majority of the required configuration tasks, avoiding the need to perform step-by-step operations in different configuration tools and using the command line on the Windows Active Directory platforms.

However, Riverbed understands that sometimes, for reasons of policy and change management, there might be a requirement to retain Steelhead appliances on earlier versions of RiOS.

This section explains how to establish the minimum version of RiOS required for optimizing signed SMBv1 or encrypted MAPI traffic. The following list describes factors affecting the minimum version of RiOS:

- Determining the required server-side Steelhead appliance RiOS version depends on a variety of factors:
  - Whether you require Transparent mode, Delegation mode, or support for end-to-end Kerberos
  - The domain functional level and operation system version of the domain controllers in the domain that is being joined by the Steelhead appliance
  - The operating system version of client machines
  - The type of trust relationship between the server domain and the user domain

The following configurations help you determine the RiOS version you want to use on the server-side Steelhead appliance:

- "User Domain Is the Same as Server Domain—Delegation Mode" on page 184
- "User Domain Is the Same as Server Domain—Transparent Mode" on page 184
- "User Domain Is the Different from Server Domain (Bi-Directional)—Delegation Mode" on page 185
- "User Domain Is Different from Server Domain (Bi-Directional)—Transparent Mode" on page 185
- "Server-Side Steelhead Appliance is in a Different Domain to the Server with One-Way Trust" on page 186

You must use RiOS v7.0 or later if your configuration requires end-to-end Kerberos authentication between the client and server.

The *Domain(Server)* listing in the following tables refers to the domain functional level in use by the domain that the server-side Steelhead appliance joins, or the operating system version of the domain controllers used by that domain—whichever is most recent. For example, if the server-side Steelhead appliance is joined to a domain where the domain controllers use Windows 2008, but the domain functional level is Windows 2003, you can reference the Windows 2008 entries in the appropriate table.

There are some configurations where signed SMB communication between two servers running Windows 2008 or Windows 2008-R2 can occur. In this instance, one of the servers is effectively acting as a client to the other server. The same rules apply for the required version of RiOS.

The following tables include two Windows server versions in the client listings. However, you must remember that the default SMB communication between these types of server is SMBv2—either configure the server-side Steelhead appliance with SMBv1 Backward Compatibility enabled, or configure the Steelhead appliances with RiOS v6.5 or later with SMB2 optimization enabled.

User Domain Is the Same as Server Domain—Delegation Mode

Use the following tables when the user is in the same domain as the servers in question and you want to use Delegation mode.

Delegation Mode Same Domain Domain(Server)	Windows XP Clients	Windows Vista and Windows 2008 Clients	Windows 7 and Windows 2008-R2 Clients
Windows 2000	None	None	None
Windows 2003	RiOS v5.5	RiOS v5.5	RiOS v5.5
Windows 2008	RiOS v5.5	RiOS v5.5	RiOS v5.5
Windows 2008-R2	RiOS v6.1	RiOS v6.1	RiOS v6.1

User Domain Is the Same as Server Domain—Transparent Mode

Use the following table when the user is in the same domain as the servers in question and you want to use Transparent mode.

Transparent Mode Same Domain Domain(Server)	Windows XP Clients	Windows Vista and Windows 2008 Clients	Windows 7 and Windows 2008-R2 Clients
Windows 2000	RiOS v6.0	RiOS v6.0	RiOS v7.0
Windows 2003	RiOS v6.0	RiOS v6.0	RiOS v7.0
Windows 2008	RiOS v6.0	RiOS v6.0	RiOS v7.0
Windows 2008-R2	RiOS v6.1	RiOS v6.1	RiOS v7.0

User Domain Is the Different from Server Domain (Bi-Directional)—Delegation Mode

Use the following table when the use is in a different domain from the servers in question, there is a bi-directional trust relationship between the domains, and depending you want to use Delegation mode.

Delegation Mode Different Domain Bi-Directional Trust Domain(Server)	Windows XP Clients	Windows Vista and Windows 2008 Clients	Windows 7 and Windows 2008-R2 Clients
Windows 2000	None	None	None
Windows 2003	RiOS v6.1	RiOS v6.0	RiOS v6.1
Windows 2008	RiOS v6.1	RiOS v6.1	RiOS v6.1
Windows 2008-R2	RiOS v6.1	RiOS v6.1	RiOS v6.1

User Domain Is Different from Server Domain (Bi-Directional)—Transparent Mode

Use the following table when the user is in a different domain from the servers in question, there is a bi-directional trust relationship between the domains, and you want to use Transparent mode.

Transparent Mode Different Domain Bi-Directional Trust Domain(Server)	Windows XP Clients	Windows Vista and Windows 2008 Clients	Windows 7 and Windows 2008-R2 Clients
Windows 2000	RiOS v6.0	RiOS v6.0	RiOS v7.0
Windows 2003	RiOS v6.0	RiOS v6.0	RiOS v7.0
Windows 2008	RiOS v6.0	RiOS v6.0	RiOS v7.0
Windows 2008-R2	RiOS v6.1	RiOS v6.1	RiOS v7.0

In releases prior to RiOS v7.0, Transparent mode cannot be used with the Windows 7 default settings unless you make a registry change to the Windows 7 client. Without the registry change, Windows 7 clients whose SMB Signed or encrypted MAPI sessions are optimized by a Steelhead appliance configured for Transparent mode do operate, but do not experience application specific latency optimization for SMB Signed traffic. Encrypted MAPI connections do not experience application specific latency or bandwidth optimizations.

To use Transparent mode with the default settings in Windows 7 clients, the server-side Steelhead appliance must be running RiOS v7.0 or later and you must join the domain in Active Directory Integrated mode.

For details on setting Transparent mode, Manual and Auto-Delegation mode, see the *Steelhead Appliance Management Console User's Guide*. For details on Active Directory Integrated mode, see "Configuring the Server-Side Steelhead Appliance for Active Directory Integrated (Windows 2003/2008)" on page 50 and the *Steelhead Appliance Management Console User's Guide*.

Server-Side Steelhead Appliance is in a Different Domain to the Server with One-Way Trust

RiOS v6.1 or later is required when the user is in a different domain from the servers in question, and the server domain has a one-way trust relationship with the user domain.

RiOS v7.0 or later is required in a one-way trust scenario, when clients use Windows 7 or Windows 2008 R2.

RiOS v8.5 is required in a one-way trust scenario, when clients use Windows 8 and servers use Windows Server 2012.

You typically see this scenario when an organization uses an external service provider to provide file or mail services. The external service provider creates a one-way trust to the organization domain so that the organization users can authenticate on the service provider servers.

For more details on required configuration when the server domain has a one-way trust to the user domain, see "One-Way Trust Configuration" on page 43.

For details on checking the domain level and settings, and details on checking and creating authentication with Manual and Auto-Delegation modes, see the *Steelhead Appliance Management Console User's Guide*.

Index

A
Active Directory integrated 50
Additional resources 3
Admission control
  Exchange clusters 22
  MAPI 21
  Steelhead appliances in a serial cluster 22
Adobe Flash, HTTP prepopulation 138
Apple HLS, HTTP prepopulation 138
Auto-discovery, MAPI 17
Automatic configuration, HTTP 79

B
Basic steps to configure
  IP aliasing 124
  NFS 123

C
CIFS 7
  oplock 9
  prepopulation 131
  protocol commands 8
  reading and writing 10
  security signatures 11
Citrix
  drive mapping 98
  multi-stream ICA traffic 99
  optimization 95
  QoS classification 104
  SecureICA encryption 97
  small packet real-time packet 106
  SSL 106
  traffic priorities 100
  virtual channels 100
Citrix Access Gateway 106
Citrix ICA traffic 96
Common Access Card 26
Connection jumping 68
Connection pooling 76

D
Delegation mode
  server-side Steelhead appliance 38
  SMB signing 38, 45
Document conventions 3
Domain authentication
  automatic configuration 58
  scaling 52
Domain controller load balancing 53
Domain health check 54
Domino servers 125

E
Encrypted MAPI
  optimization 49
Enhanced cryptography license key 146
Exchange 2013 29
Exchange clusters, MAPI 22

F
FTP 115
  active mode 115, 118
  in-path rules 117
  optimization 115
  passive mode 116, 118
  QoS classification 118
  Steelhead Mobile 120

H
HTTP 63
  authentication settings 88
  automatic configuration 73, 79
  browser behavior 64
  connection jumping 68
  connection pooling 76
  effectiveness of optimization 89
  info-level logging 90
  Internet-bound traffic 84
  Kerberos 66, 86
  NTLM 66, 86
  NTLM authentication 67
  pipelining 65
  prepopulation 137
  proxy servers 69, 89
  refer header 64
  RiOS optimization methods 73
  settings for common applications 82
  SharePoint 83
  SSL proxy interception 71
  use case 90
HTTP prepopulation, Adobe Flash 138
HTTP prepopulation, Apple HLS 138
HTTP prepopulation, Microsoft Silverlight 138
HTTPS Mode for Oracle Forms 121

I
IP aliasing 124

IPSec encryption
  with Oracle Forms 121

J
Java applet in Oracle Forms 121

K
Kerberos 45
  AS reply. see ticket to get tickets
  authentication service 46
  HTTP 66, 86
  key distribution center 46
  multiple domains 47
  native kerberos environment 48
  overview 46
  replication 49
  restricted trust 44
  server-side Steelhead appliance 38
  ticket to get tickets 46
  ticket-granting service 46
  Windows domain 47
  Windows security concepts 34
Known issues 4

L
Licensing
  SSL 23, 146
Load balancing, domain controller 53
Lotus Notes optimization 124
Lotus Notes, Domino servers 125

M
MAPI 15
  admission control 21
  auto-discovery 17
  disabling encrypted 29
  encrypted optimization 19, 33
  Exchange 2013 29
  optimization methods 18
  Outlook anywhere 23
  parallel deployment 22
  serial cluster 22
  static and dynamic ports 31
Microsoft IIS server 85
Microsoft Silverlight, HTTP prepopulation 138
Mixed mode, Windows domain 37
Multiple domains
  Windows 7 clients 61

N
Native mode 121
Native mode, Windows domain 37
NFS
  optimization, enabling 122
  prefetch policies for 122
  server IP aliasing 124
NTLM 34, 38, 66, 86

O
One-way trust 36, 43
Online documentation 4
Optimization
  CIFS 7
  CRLs 173
  encrypted MAPI 33
  FTP 115
  HTTP 63
  Lotus Notes 124
  MAPI 15
  NFS 122
  NFS IP aliasing 124
  protocol 121
  SCEP 173
  signed SMB 33
  SMTP-over_TLS 109
  SSL 141
optimization
  encrypted MAPI 49
  SharePoint 49
  SMB signing 49
Oracle Forms
  determining the deployment mode 122
  optimizing 121
Outlook anywhere 24
  configuration 24
  HTTPS 23
  MAPI 23
  troubleshooting 28
  verifying connection status 27

P
Peering, secure Steelhead appliance 162
Pipelining 65
Prepopulation
  CIFS 131
  HTTP 137
Protocol
  SSL, verifying optimization for 168, 169
Protocol optimization 121
Proxy certificate and private key 146
Proxy servers 69

Q
QoS
  CIFS prepopulation 135
  Citrix 104
  FTP 118

R
Referer header 64
Release notes 3
Restricted trust, Kerberos 44
Riverbed Professional Services, contacting 4

S
SCEP
  automatic re-enrollment 175
  on-demand enrollment 175
  overview 173
  settings and alarms 176
Secure peering, Steelhead appliance 162
SecureICA encryption 97
Server name indication (SNI) 166
SharePoint 83
  optimization 49
SMB signing 33
  delegation mode 38, 45
  different domain 186

multiple domains 61
NTLM 34, 38
one-way trust 43
optimization 49
single domain 60
transparent mode 38
SMB signing and encrypted MAPI
configuration overview 40
SMB2 11
SMB2.1 11
SMB3 11, 41
SMB3.02 11
SMTP, TLS 109
Socket mode 121
SSL
Citrix 106
enable client certificate support 164
enhanced cryptography license key 146
HTTP optimization 71
licensing 23, 146
mode in Oracle Forms 121
proxy certificate and private key 146
proxy certificate and private key
scenarios 152
server name indication (SNI) 166
SMTP 109
TLS 112
verifying 168, 169
Steelhead appliances
secure peering 162
secure Windows 51
Steelhead Mobile, FTP 120
Sun JRE 122

T
Technical support, contacting 4
TLS
Microsoft Exchange hub transport 110
SMTP 109
SSL 112
Transparent mode, SMB signing 38
Transport mode
server-side Steelhead appliance 37
Trusted domain 36
Trusting domain 36

V
Video
Adobe Flash 138
Apple HLS 138
HTTP prepopulation 137
Microsoft Silverlight 138
Virtual channels, Citrix 100

W
Windows domain
authentication automatic configuration 58
constrained delegation 45
Exchange server 20
joining a Steelhead 42
Kerberos 47
mixed mode 37
more than one 35
native mode 37
one-way trust 36
relationships 35
trusted domain 36
trusting domain 36
Windows domain health 53
Windows domain health check 54
Windows security 34