Troubleshooting Riverbed® Steelhead® WAN Optimizers

Edwin Groothuis

Troubleshooting Riverbed® Steelhead® WAN Optimizers

Edwin Groothuis

Publication date Mon 19 May 2014 10:59:17

Abstract

Feedback Information:

If you have any comments regarding how the quality of this book could be improved, please contact the author. He can be reached via email at edwin.groothuis@riverbed.com or at steelhead-troubleshooting@mavetju.org, his personal website can be found at http://www.mavetju.org/.

Table of Contents

Preface

Introduction

Welcome! This book is about troubleshooting Riverbed Steelhead appliances and networks optimized by the Steelhead appliances. It contains the experiences with cases handled by the Riverbed TAC on how to detect the source of problems in networks with Steelhead appliances and how to troubleshoot problems with Steelhead appliances.

For many years, networking has been fun but it has become relative boring: Moving packets around, processing routing updates for reconnected networks and linking up different transport layers. Everybody knows and understands IP routing and troubleshooting it is a pretty straight forward process. WAN optimization is a new field, a Steelhead appliance is not just a router or switch where packets go through, it is a hop in the network where packets get mangled, contents get changed and some magic happens: Shazam! Knowing how this magic works will help you understand what is happening and resolve issues with the Steelhead appliances faster and more efficient.

Edwin.

Intended Audience

This is not a book on how to manage and deploy Steelhead appliances, the Riverbed Deployment Guide and the Steelhead Management Console User's Guide (both available from the Riverbed Support website) are the right books for that. This is a book about what can go wrong with Steelhead appliances in a network. This book is for everybody who has two or more Steelhead appliances in their network and wants to understand how the Steelhead appliances interact with other devices in the network, what can go wrong and how to use the features on the Steelhead appliance to troubleshoot it.

The easiest way to troubleshoot Steelhead appliances is to know what is expected to happen when you poke and prod it. Depending on the behaviour seen, the next steps can be easily determined. Very simple steps, very logical steps, sounds nearly like a normal networking troubleshoot approach. And that is often all what is required.

This book describes experiences, possible issues and troubleshooting with the xx20, xx50, CX and EX series Steelhead appliances and software versions up to, and including, RiOS 8.5.

Organization of this book

This book is split in several chapters; The first chapters with some background on WAN optimization, the next set of chapters are about tools available on Steelhead appliances, followed by the troubleshooting parts.

Chapter 1. Introduction to WAN Optimization

1.1. Index

This chapter describes the background of WAN optimization in generic terms, from a vendor neutral point of view.

- Why WAN optimization is needed: Where are the delays?
- The three different optimization methods: The network transport layer, the packet payload and the application protocol.
- Effects of WAN optimization on the networks, servers and clients.

1.2. Why WAN optimization?

WAN optimization is a set of techniques that address various issues experienced in a Wide Area Network (WAN):

- Link related issues, caused by physical limitations.
- Protocol related issues.

1.2.1. Link related issues

A network link has two characteristics:

- The link speed, which defines the speed of the data through the physical layer.
- The link delay, which defines how long it takes for a bit to reach the other side of the link.

1.2.1.1. Historical changes in link speed

The link speed is the number of bits per second which can be transferred over a link. In general, the speed of the LAN is much faster than the speed of the WAN.

Around 1995, the speed of a WAN link was measured in multitudes of 64 kbps and a LAN link was still working on a 10 Mbps shared coax cable. It wasn't for another five years in 2000 that an E1 link at 2 Mbps became the standard speed for WAN links and that a 100 Mbps Ethernet cables towards a LAN switch became available for the desktop. Five years later in 2005 the general speed for a WAN link was 10 Mbps and the network switch was connected to the desktop at gigabit Ethernet speed (1000 Mbps).

This LAN/WAN speed difference made it in the past necessary to have services such as file servers, mail servers and web proxy servers, on the local LAN on the remote location (the branch). However, the speed of the WAN link in 2005 is now fast enough to consolidate them into a data center.

Consolidation of these remote servers into data centers made the management of the machines easier: Reduction of the number of machines by consolidating then, better control over the environment the machines are operating in, easier implementation of fail-over scenarios because of the availability of high bandwidth needed for replication, and a simplified management of the services running. However, the bandwidth towards the client was reduced from gigabit speeds back to pre-2000 speeds of 10 Mbps.

1.2.1.1.1. Serialization delay

The link-speed defines the serialization delay, the time it takes for a packet to be converted from set of bytes in the memory of a host (computer, router, or switch) into a string of bits on the wire.

For example, to forward one packet of 1500 bytes through a router with a WAN interface of 1 Mbps, it will take 12 milliseconds (1500 bytes * 8 bits per byte / 1 000 000 bits per second = 12 ms).

If WAN optimization reduces that packet of 1500 bytes to a packet of 100 bytes, this serialization delay will be reduced to 0.8 milliseconds.

Figure 1.1. Serialization delay

```
^ |
| |                                  _____
H |                       __________/    |
o |                      /|        |     |
p |                     / |        |     |
  |                    /  |        |     |
d |                   /   |        |     |
e |                  /    |        |     |
l |                 /     |        |     |
a |      __________/      |        |     |
y |     / |        |      |        |     |
  +----------------------------------------------- network hop distance
  client  WAN router      WAN router     server
```

1.2.1.2. Link delay

On longer distances, the speed the data travels through the medium comes into play. For example, the speed of light in a fiber cable, which is about 5 microseconds per kilometer (1.5 / 300 000 km/s, where 1.5 is the refraction index for fiber{SOURCE:Wikipedia Latency (engineering)}). For a fiber cable with a length of 1000 kilometers this would add 5 ms for the signal to reach the other side.

1.2.2. Network protocols

The TCP network protocol has had lots of improvements over the last years, but not all TCP stacks support all features. By terminating the TCP session locally on the LAN and setting up a new one which does support all the new features, the WAN optimizer can make these TCP sessions faster over the WAN.

Some protocols are smart, like the connection-oriented TCP protocol, which makes sure that all the data sent by the sending application is presented to the receiver application in such a way that nothing is missing and that it is in the right order.

1.2.3. Application protocol related issues

Application protocols are the layer in which the client and the server talk to each other.

Application protocols can be unintentionally implemented with a special environment in mind. For example results from database queries can be requested either in bulk or one by one. The first implementation works great over a WAN but needs more memory on the client to store all the answers, while the second implementation works bad over a WAN but doesn't need as much memory. Serialization delay is here the cause of the problem for a WAN deployment.

1.3. Three different optimization methods

WAN optimization attacks the issues mentioned previously in three different ways:

- Transport layers optimization.
- Data reduction.
- Latency optimization.

1.3.1. Transport layers optimization

These days, the most popular transport layers are Ethernet, IP and TCP.

Figure 1.2. A typical Ethernet frame

```
.---------------------------------------------------------------------.
| Ethernet | IP | TCP | Payload                                       |
'---------------------------------------------------------------------'
                      <------------------------------ 1460 bytes ----->
                <------------------------------------ 1480 bytes ----->
           <----------------------------------------- 1500 bytes ----->
```

- Ethernet is the local network layer to the next hop in the network. The default maximum payload for an Ethernet frame size is 1500 bytes.
- IP is the layer which takes care of the routing of the packet in the network.
- TCP is the layer which takes care of the data exchanged between the client and server and which makes sure that the data is fed to the receiving application in the same order as the sending application has delivered it. It takes care of the retransmission of lost packets and time-outs, if there is a problem it backs off a little bit and allows the network to recover.

  Standard TCP has the following features and limitations:

  - The TCP Sliding Window method, which makes it possible for the sender to keep sending data until the limit for maximum number of bytes in flight as specified by the receiver is reached. In the original TCP design this was limited to 64 kilobytes{SOURCE:RFC 793}.

    ### Figure 1.3. Bytes in-flight over time.

    ```
    B I |        +--+--+--+--+--+
    y n |        |  |  |  |  |  |
    t f |        |  |  |  |  |  |
    e l |     +--+  |  |  |  |  |
    s i |     |  |  |  |  |  |  |
      g |  +--|  |  |  |  |  |  |
      h |  |  |  |  |  |  |  |  |
      t +-------------------------- ---> time
                   ^
                    \__ First acknowledgement received
    ```

  - The original TCP Acknowledge method specifies that the receiver can only acknowledge packets it has received. It cannot tell the sender that it has missed packets but has to wait for a timeout in the acknowledgement from the server. Then the server has to retransmit all the unacknowledged packets.

    ### Figure 1.4. Retransmission due to a time out in the acknowledgement

    ```
    |
    |  +--+--+--+  +--+--+         +--+--+--+
    |  |  |  |  |  |  |  |         |  |  |  |
    |  |N |N |N |  |N |N |         |N |N |N |
    |  |- |- |- |  |+ |+ |         |  |+ |+ |
    |  |3 |2 |1 |  |1 |2 |         |  |1 |2 |
    |  |  |  |  |  |  |  |         |  |  |  |
    |  |  |  |  |  |  |  |         |  |  |  |
    +----------------------------------------- ---> time
                  ^      ^         ^__ ACK timeout
                   \      \___________ TCP Window full
                    \_________________ Lost packet
    ```

  - The TCP Window Size uses a Slow Start mechanism which starts with a Window Size of 2 * Segment Size{SOURCE:RFC 793} and increases it with one Segment per received acknowledgment packet.

In case of packet loss it will half the TCP Window Size and slowly increase the Window Size again per received acknowledgment packet.

Figure 1.5. Retransmission due to time out

```
Window Size
|
|                       +--+--+                       +--+
|                    +--+  |  |                    +--+  |
|                 +--+  |  |  |                 +--+  |  |
|              +--+  |  |  |  |              +--+  |  |  |
|           +--+  |  |  |  |  |     +--+--+--+  |  |  |  |
|  +--+--+--+  |  |  |  |  |  |     |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |
+----------------------------------------------------------- ---> time
             ^          ^     ^    ^        ^____________ First ACK
              \          \     \    \____________________ Retransmission
               \          \     \________________________ Lost packet
                \          \_____________________________ TCP Window maximum size
                 \_______________________________________ First ACK
```

Transport layer optimization can add the following features for the traffic between two Steelhead appliances:

- For Ethernet there is not much which can be improved here, unless there is full control of all the WAN devices between the clients and servers, in which case Jumbo frames can be used.

  The payload of Ethernet Jumbo frames can be increased up to 9000 bytes {SOURCE: Wikipedia Jumbo Frame}. This means the IP payload, instead of having to split a stream in pieces of 1480 bytes, can be split in pieces of 8980 bytes. For the routers in the path this means that they have to process only one sixth of the number of packets. However on the TCP level the improvement is not comparable since the maximum number of outstanding bytes is still limited to the TCP Window size.

- On the IP level there is no optimization possible.

- On the TCP level there are several enhancements to improve the traffic flow:

  - In newer TCP stacks a feature called TCP Window Scaling has been introduced which increases the 64 kilobyte window size limit by multiplying it with a power of two value {SOURCE:RFC 1323}. This means that the number of bytes in-flight can be much larger.

  - TCP Fast Retransmission {SOURCE:RFC 2581} is a method for the receiver to signal, by sending three empty duplicate ACKs, that a certain packet has not been received but that later packets have been received. This will reduce the time for the start of the retransmission.

  - TCP Selective ACK {SOURCE:RFC 2018} is a method to inform the sender that further packets have already been received, implicitly telling which packets have been lost by giving the already received TCP sequence numbers. This will reduce the time for the start of the retransmission and the amount of packets retransmitted.

  - On the sender side, the use of TCP Timestamps {SOURCE:RFC 2018} can be used to determine the Round Trip Time (RTT) towards the receiver and thus the expected time that an acknowledgment for a packet should arrive. This will give the sender an indication of a lost packet before the official timer expires.

  - If it is known that the quality of the network is not optimal due to packet loss, or a high latency, then the TCP Window might never be at the maximum size and the performance will be sub-optimal. The sender can start with a large TCP Window and decrease the TCP Window by only one Segment Size instead of a halving it to improve the performance over such kind of networks.

  The use of TCP Selective ACK and TCP Timestamps is negotiated during the setup of the TCP session.

1.3.2. Data reduction

Once the transport layer is optimized, the next step is to reduce the payload sent over it. There are several methods to reduce the size of the data: Compression and dictionaries.

- Compression is the process of encoding data to use less bytes than the original data took.

  As an example, the string *AAAAAA* could be compressed to *6"A"(NUL)*, reducing it from six bytes to three. The string *ABABAB* could be compressed to *3"AB"(NUL)*, reducing it from six bytes to four bytes. And the string *ABCDEF* can only be compressed to *1"ABCDEF"(NUL)*, increasing it from six bytes to eight bytes.

  Certain data can be compressed very well, like HTML data and source code files. Other data cannot be compressed because it is too unique or compressed already.

- Dictionaries are lists of patterns that known to both Steelhead appliances, so that instead of having to send a long pattern it will send a reference to that data. For example if the sender receives the pattern *AAAAABBBBB* and both sides have a reference for *AAAAA* and *BBBBB* in their dictionary, the sender can send the labels of these references, the receiver will look them up in the dictionary and forwards the referenced data on the wire.

  These dictionaries are learnt over time by looking at the traffic going through the Steelhead appliances. Patterns which are used often will end up in the front of the dictionary. When the dictionary is full, patterns which were used only once or haven't been used for a long time get removed from the dictionary and space to store new learned patterns is created.

  When a data stream contains patterns which are never seen before, the quality of the sending of the data is considered *a cold transfer*. When a data stream contains patterns which have been seen before, the quality of the sending of data is considered *a warm transfer*.

  Unlike compression which doesn't like compressed data, dictionaries do not mind that the data is already compressed. However, this only works if the data is an object retrieved from a remote server, it doesn't work if the data-stream is compressed interactive traffic.

  Storing patterns from encrypted data streams in the dictionary is also a bad thing, because by definition this encrypted data is unique. For example, the first time the string *AAAAA* is encrypted it shows up as *ABCDE*, the second time it is encrypted as *AEDBC*, the time after that it is encrypted as *BEBAC"*. Instead of learning a repetitive pattern, the dictionary will be polluted with patterns which never will be seen again.

- Encryption integration

  As mentioned previously, patterns of encrypted data streams should not be stored in the dictionary. In trusted environments, WAN optimizers might be able to proxy and perform decryption and re-encryption on behalf of the sender. This way they are able to decrypt the data, optimize it over the WAN and then to encrypt it again to the receiver.

1.3.3. Application latency optimization

Application latency optimization improves the performance of the protocol on top of the transport layer. The gain is when the protocol is predictable or is chatty, meaning that many little requests and responses are exchanged. This is fine for the zero millisecond delay to a server on the local LAN, but very expensive if it has to go over the WAN. To be able to do latency optimization, a deep understanding of the protocol and the client behaviour is required.

Transaction prediction is part of latency optimization. It is a technique to predict what the next request from the client is, based on the past and current requests. For example, on a network file system, if a client opens a file and asks for the first block of a file, there is a huge chance it will ask for the second and the third block of that file too. If however the client opens a file, seeks to a random position, reads 16 bytes, seeks to another position, reads 32 bytes, seeks to another position and reads 24 bytes, that behaviour is not predictable.

Various latency optimization techniques include:

- Towards network file systems; the basic operations towards remote file systems include the reading and writing of files and scanning through directories.

  - Read-ahead: When a client opens a file and reads the first block, there is a huge chance that it will after that ask for the second block and the third block. The latency optimization sees the request for the first block, but asks the server for the first, second and third block and returns the first block to the client. When the client asks for the second block, the latency optimization already has this block available and the request doesn't have to go over the WAN anymore.

  - Write-behind: When the client creates a new file and writes the first block to it, the local latency optimization can tell the client that the block was written after which the client can prepare and start to write the next block. In the meantime the latency optimization sends the data to the server.

  - Directory caching: When a client asks the first entry in a directory on a server, there is a huge chance it will also ask for the next directory entries. The latency optimization will ask the server for all directory entries, send them to the client side latency optimization which then can answer the consecutive requests from the client without having to send the requests to the remote server.

  - Pre-population: This is a technique in which new files on network file system are transferred over the WAN so that the Steelhead appliances know about the patterns and when the user asks for them they can be warm transferred.

  Data integrity is very important in network file systems, the client should not get the impression that the data has been written to disk while the data is still in transit. There are several commands which shouldn't touch local latency optimization, for example the open and close commands: It is the server which is saying Yes or No on the success of that command.

- Towards mail servers; the basic operations towards mail-servers are to open a mailbox, retrieve the contents of a message, retrieve attachments for a message, general mailbox maintenance, to move messages between folders and to deliver new messages.

  - Attachment warming: When the mail client polls for new email and retrieves a list of new messages, the latency optimization can check the contents to see if there are any attachments on these messages. If there are attachments, they can be downloaded by latency optimization so that the patterns are known in the dictionary. When the user later reads the message and wants to see the attachment, the patterns are already known in the dictionary and the attachment gets retrieved in "a warm transfer".

  - Attachment write behind: This works just like the write behind feature of the network file systems.

- Towards web servers; Web servers receive a request and return an object. This object can contain references to other objects, which in turn need to be requested.

  - Caching of objects: If the returned object is a static object which is valid for a long time, for example an audio file or an image, the latency optimization uses the meta data of this object to check how long the object is valid for. If another client asks for the same object and the object is still valid, the latency optimization can serve it to the client without having to get it from the server again.

  - Prefetching of static objects: When the object requested can contain references, the latency optimization can parse the object and retrieve the referenced objects in it before the client asks for them.

1.4. Effects of WAN optimization on other equipment

WAN optimization will remove the WAN bottleneck, but that means that suddenly other bottlenecks can appear:

- The amount of traffic on the WAN side of the Steelhead appliance will decrease, but on the other side the amount of traffic on the LAN side will increase.

Without WAN optimization, the maximum amount of traffic on the LAN side is the same as the maximum amount of traffic on the WAN side. With WAN optimization, the maximum amount of traffic on the LAN is suddenly not limited to the maximum amount of traffic on the WAN, it will be a multitude of it.

For example, if the WAN link is 10 Mbps and the optimized factor is 15x, then the amount of traffic on the LAN side will be 150 Mbps. If the speed of the Ethernet segment on the LAN side is 100 Mbps, then the LAN side is suddenly the bottleneck.

Figure 1.6. LAN/WAN bandwidth comparisons

```
       Without WAN optimization
B p s |
y e e |+x+x+x+x+x+x+x+x+x+x+x+x+x+x+x+   + is incoming WAN
t r c |                                 x is outgoing LAN
e   o |
s   n |
    d +------------------------------
                             time ->

       With WAN optimization
B p s |xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
y e e |                                 + is incoming WAN
t r c |                                 x is outgoing LAN
e   o |++++++++++++++++++++++++++++++
s   n |
    d +------------------------------
                             time ->
```

- The traffic pattern will change: There will be higher bursts of traffic on the client-side LAN, but they will be shorter.

Figure 1.7. Traffic pattern on the client-side LAN side.

```
  Without WAN optimization    With WAN optimization

  LAN traffic out             LAN traffic out
  |                           | ___
  |                           ||...|
  | _________                 ||...|
  ||.........|                ||...|
0 +-----------              0 +-------------
       time ->                      time ->
```

- The CPU utilization on routers and switches behind the Steelhead appliances will change: The amount of packets and data transferred will stay the same over time, but the number of packets per time-period will increase.
- The CPU utilization on routers and switches between the Steelhead appliances will change: The number of packets transferred will decrease and the payload will be smaller. Due to the possibility to send more data, the number of packets will increase.
- On file servers, the I/O pattern to hard disks will change because the speed with which files are transferred over the network is faster.

Figure 1.8. Disk load on the server.

```
  Without WAN optimization    With WAN optimization

  Disk load                   Disk load
  |                           | ___
  |                           ||...|
  | _________                 ||...|
  ||.........|                ||...|
0 +-----------              0 +-------------
       time ->                      time ->
```

- Latency optimization will not only cause the data to be retrieved faster from the servers, it will also increase the amount of traffic in cases where transaction prediction failed.

Figure 1.9. Traffic pattern on the server-side LAN side.

```
  Without WAN optimization    With WAN optimization

  LAN traffic in              LAN traffic in
  |                           | ______
  |                           ||......|
  | _________                 ||......|
  ||.........|                ||......|
0 +-----------              0 +-------------
       time ->                      time ->
```

Chapter 2. The Riverbed Approach to WAN Optimization

2.1. Index

This chapter describes how Riverbed has implemented its WAN optimization products.

- The design of the Steelhead appliances, physical and logical.
- The startup of the Steelhead appliance, the processes running on it.
- The configuration of the Steelhead appliance, routing of the packets, interface settings, license keys.
- The design of an optimized TCP session, the setup of an optimized TCP session.
- In-path rules, peering rules, WAN visibility.
- Different hardware models.

2.2. Appliance Setup

2.2.1. Technical overview

Riverbed has two types of technologies for WAN optimization:

- The Steelhead technology, which performs the WAN optimization.

  Note that the Steelhead Mobile, Cloud Steelhead, Steelhead Cloud Accelerator and Virtual Steelhead all have the same technology as the Steelhead appliance but are implemented in different scenarios:

  - Steelhead Mobile runs on desktop and laptop computers running the Microsoft Windows and Apple OS X operating system and optimizes traffic from the single computer it is running on.
  - Cloud Steelhead runs in cloud environments such as Amazon EC2 and optimizes traffic towards hosts in the cloud.
  - Steelhead Cloud Accelerator runs in the Akamai network and optimizes traffic towards Software-as-a-Service applications like Microsoft Office365 and Salesforce.
  - Virtual Steelhead runs as a virtual machine under VMware ESXi.

- The Interceptor technology, which acts as a redirector of traffic towards Steelhead appliances.

  Unlike other traffic redirection protocols like PBR or WCCP which are agnostic of the device it is forwarding to, the Interceptor only works with Steelhead appliances and takes advantage of the knowledge of the status reported by the Steelhead appliances.

There are two related appliances developed by Riverbed, which do not perform any WAN optimization but are used to manage and monitor the Steelhead appliances in the network:

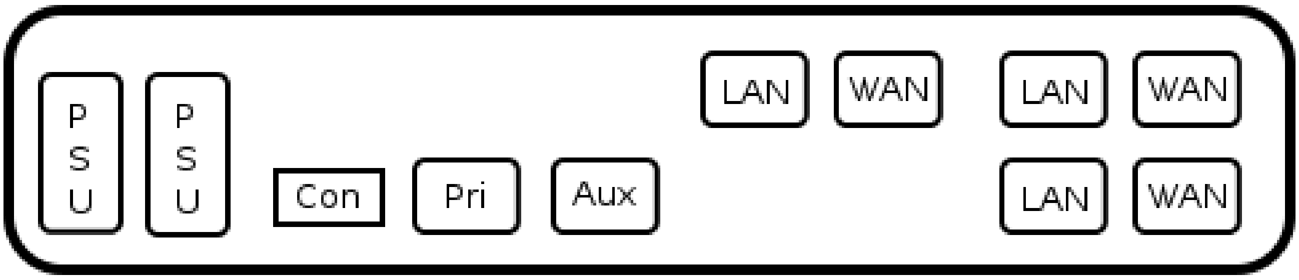
- The Steelhead Mobile Controller (SMC), a configuration and reporting appliance to manage Steelhead Mobile deployments.
- The Central Management Console (CMC), a configuration and reporting appliance to manage configurations and monitor the behaviour of Steelhead appliances, Interceptor appliances and Steelhead Mobile Controllers.

2.2.2. Design of the Steelhead appliance

2.2.2.1. The physical design

From the outside, a Steelhead appliance has the following typical features:

Figure 2.1. Back of a Steelhead appliance



- A serial console port, to be used to do the initial configuration of the device and a way to access the device in case it isn't reachable anymore via the IP addresses configured on the primary interface. Its speed is 9600 bps, 8 data bits and one stop bit.
- Two network interfaces, primary and auxiliary. These are the base interfaces used for management. The primary interface is normally the management port, which should be used to access the device via the network. The auxiliary interface is commonly used for data store synchronization but can also be used for management.
- One or more by-pass cards, each with a pair of network interfaces labeled LAN and WAN. These are the interfaces which are connected to the WAN router and the LAN switch. The by-pass cards can be integrated on the chassis or installed via additional PCI cards.

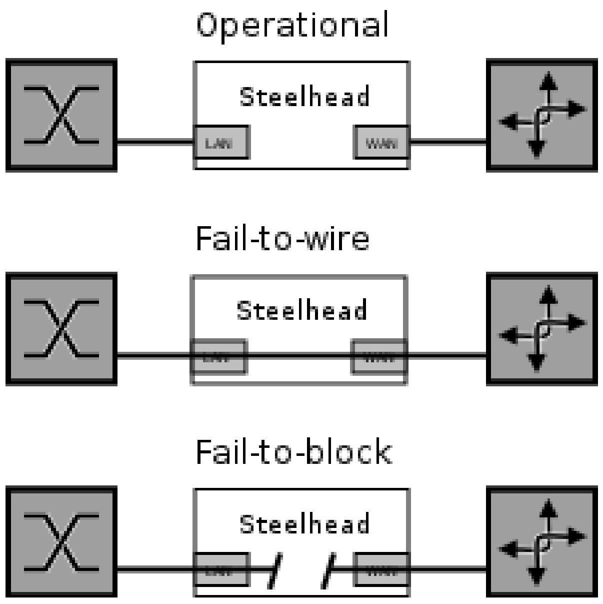
Depending on the model, the following features may be available:

- LED alarm lights on the front. They show the operational status of the device.
- Hot-swappable hard disks on the front.
- One or more hot-swappable power supplies.

On the inside there are several other important features:

- The machine has two hardware watchdogs.
  - The hardware watchdog, which will reboot the appliance if a user-land process has not been communicated with the hardware watchdog for 30 seconds. This prevents the Steelhead appliance to hang indefinitely and gives it a chance to recover.
  - The network watchdog, which will put the by-pass card into fail-to-wire or fail-to-block mode if the optimization service has not communicated with the network watchdog for 7 seconds or more. This will remove the Steelhead appliance from the path in the network if there are problems with the optimization service.
- If the machine is in normal operation, the LAN and WAN ports of the by-pass card are in operational mode and the optimization service is intercepting the packets. If the appliance gets powered off, the by-pass card either goes in fail-to-wire or fail-to-block mode. In the former mode, the LAN and WAN port are directly cross-connected with each other so that the LAN switch and WAN router are on Ethernet level directly connected to each other. In the latter mode, the LAN and WAN port will be disconnected from each other and no Ethernet link will be established.

Figure 2.2. Ethernet links in normal operation, fail-to-wire and fail-to-block.



2.2.2.2. The logical design

- As mentioned before, the primary and auxiliary interfaces are for the management of the port. Despite being two different interfaces, they cannot have an IP address on the same IP subnet as they share the same IP routing table.

- Each physical LAN and WAN interface is combined into a virtual in-path interface in the configuration of the Steelhead appliance. The IP subnet on the in-path interface can overlap with the IP subnet of the primary and auxiliary interfaces and with the other in-path interfaces. Each in-path interface has its own routing table and must have an IP address and a default gateway configured before optimization is possible via this in-path interface.

- Each Steelhead appliance has a data store to store its dictionary and every data store has a unique data store ID to identify the data store with other Steelhead appliances in the network. The exceptions for these are Steelhead appliances which are in data store synchronization cluster, where the data store ID of the slave is the same value as the master.

- Some Steelhead appliances have only one hard disk, others have multiple hard disks in a RAID0, RAID1 or RAID10 setup so that for the operation system it looks like one giant hard disk. The operating system, the optimization service and the data store are all located on various partitions on this one giant hard disk.

  Devices with a Fault Tolerant Segstore (FTS) have their operating system and optimization service separated from the data store: These devices have a normal RAID1 implementation for the operating system and the optimization service, while the data store is spread over the remaining disks. The contents of the data store are not protected for disk-failure but will be lost when one of the disks of the FTS fails. The advantage of the FTS is that there is no RAID overhead, the full capacity of the disks installed in the Steelhead appliance is used and there is no RAID rebuild overhead when a disk gets replaced.

- The Steelhead appliance has flash memory storage where it boots from and where it keeps a copy of the configuration. It allows models without the RAID1 or RAID10 capability to rebuild themselves in the field if the hard disk gets replaced.

- The Steelhead appliance has two partitions which it can boot into, one with the current version of RiOS and one with the version installed prior to the last upgrade. When a console is connected during boot-up of the appliance, you will have the option to select the partition to boot from.

2.2.3. Start-up of the Steelhead appliance

When a serial console is attached to the serial port of the Steelhead appliance, you can follow the start-up sequence of the Steelhead appliance.

- The Power On Self Test (POST) of the machine, during whic hthe BIOS is accessible and the RAID controller gets initialized.
- The boot-loader, which selects the RiOS version to boot from.
- The start-up of the Linux kernel and loading of various device drivers.
- The mounting, and if needed the checking, of file systems.
- If a software upgrade or downgrade needs to be done, this happens now.
- Other upgrades such as a network card firmware and RAID Kit installation will also occur at this stage.
- Start of the process *pm*. This process is the RiOS Process Manager which controls all the RiOS specific processes, such as the statistics process, the optimization service, the management process, and the SNMP service related processes to name a few.
- From now on you will be able to login to the appliance via the network and via serial console.

Once the Process Manager is started, the configuration of the network interfaces will be applied, but the by-pass card will not go out of fail-to-wire or fail-to-block yet: This happens only when the optimization service has completed the initialization and has become operational.

2.2.4. The processes on the Steelhead appliance

Besides the optimization service, there are other processes to support the operation of the Steelhead appliance. The following lists contain the processes known on a Steelhead appliance running RiOS version 7.0, but some processes are only run when certain features are enabled. This list can be obtained with the commands `show pm process ?` on the CLI of the Steelhead appliance.

2.2.4.1. Optimization service related processes

This is a list of the most important process on the Steelhead appliance:

- **acp**, the Akamai tunnel process.
- **alarmd**, the Alarm manager
- **cli**, the CLI of the Steelhead appliance.
- **cmcfc**, the CMC auto-registration.
- **httpd**, the web server for the GUI.
- **mgmtd**, the central management process.
- **pm**, the Process Manager which keeps track of all Steelhead related processes.
- **qosd**, the interface towards the QoS service.
- **rcud**, related to CIFS pre-population.
- **rgp**, CMC management related process.

- **rpgd**, CMC management related process.
- **rspd**, RSP watchdog
- **shark**, for Cascade Pilot integration
- **sched**, the job scheduler.
- **sport**, the optimization service.
- **statsd**, the statistics collection process.
- **virt_wrapperd**, RSP related.
- **wdt**, the user-land process to keep the watchdog happy.
- **webasd**, related to the web server for the GUI.
- **winbind**, for Active Directory integration.

2.3. Basic configuration of a Steelhead appliance

2.3.1. Host configuration

The Steelhead appliance needs to know its hostname, DNS domain name(s), DNS server(s) and NTP server(s).

The hostname is used to create a self-signed SSL certificate for HTTPS access to GUI of the Steelhead appliance, displayed in the list of the Connected Appliances overview on other Steelhead appliances, used in the SSL certificate for Secure Peering, used during the joining of the Steelhead appliance to an Active Directory domain and used on the login page of the GUI and on the CMC in the appliances list.

The domain names, DNS servers and NTP servers will be important when the Steelhead appliance performs CIFS Pre-population and signed CIFS and encrypted MAPI latency optimization as this involves integration into the Active Directory infrastructure.

Using valid NTP servers is a good practice even without Active Directory integration, as it makes sure that the logging and statistics match with other Steelhead appliances. By default the devices are configured with the NTP server of Riverbed and NTP servers from the pool.ntp.org project. If the Steelhead appliances cannot communicate towards to the public Internet, make sure that they use the internal NTP servers of your network.

2.3.2. Management interface configuration

The physical management network interfaces of the Steelhead appliance are the primary and the auxiliary network interfaces. As a general habit, you should access the Steelhead appliances via the IP address defined on the primary interface.

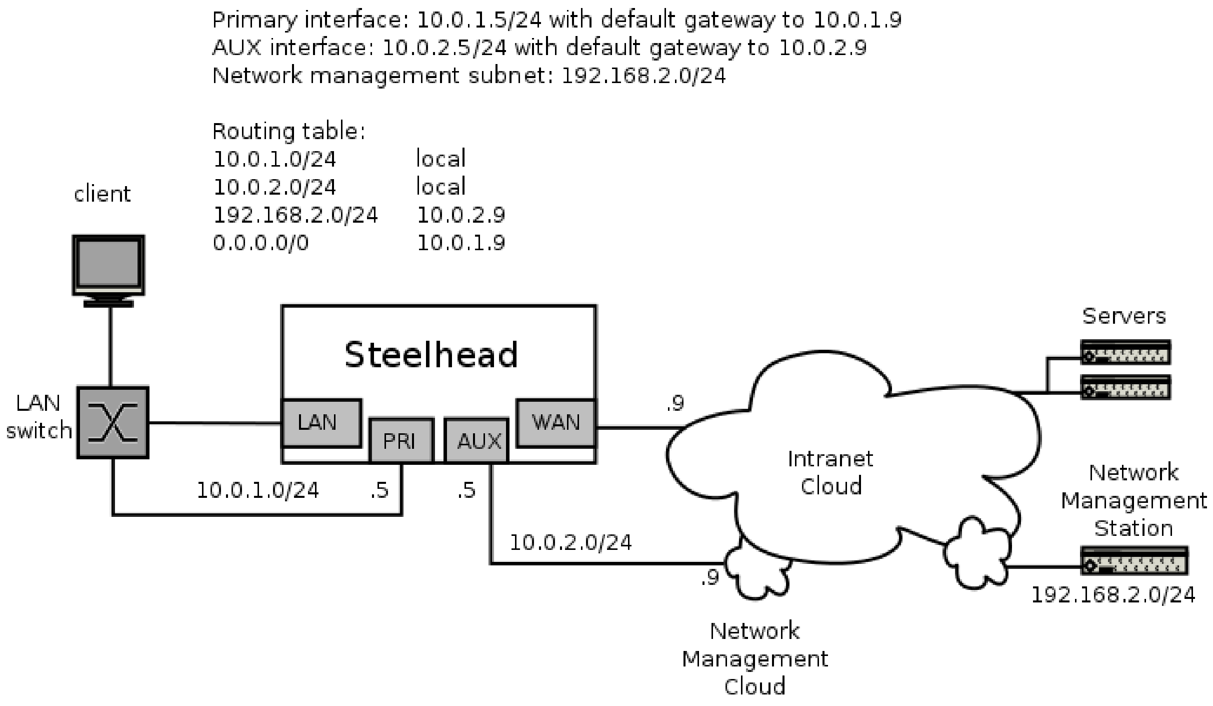
The primary interface needs an IP address, subnet mask and default gateway. The auxiliary interface should only be needed to be configured if Data Store Synchronization is enabled or if a separate private network management network is used.

Both the primary interface and the auxiliary interface share the same routing table and it is not possible to use addresses from the same IP subnet on the two interfaces.

When the Steelhead appliance is initiating traffic, for example to send out SNMP traps or when it is downloading software, it uses the IP address of the primary interface as the source IP address.

To force management packets out via the auxiliary interface, the destination IP subnet should have a gateway IP address defined on the auxiliary interface IP subnet:

Figure 2.3. Management via both the primary and auxiliary interfaces



2.3.3. In-path interface configuration

The in-path interfaces behave partly like a network bridge and partly like a router:

- All traffic not going to be optimized is bridged through from the LAN port to the WAN port and vice versa without changing it, not even decreasing the TTL in the header of the IP packet.
- All traffic to be optimized will follow the routing configuration and routing decisions as defined on the in-path interface and learned via Simplified Routing.

Each active in-path interface needs an unique IP address, subnet mask and default gateway. Optionally it can have a VLAN tag ID configured on which the IP subnet is defined. Explicit routing entries can be defined for the traffic terminating on or initiated from the in-path interface.

The routing table of each in-path interface is separate and each in-path interface can have an IP address in subnets on the others IP subnets.

2.3.3.1. Simplified Routing

Simplified Routing is a feature which can be used to overcome the administrative overhead of having to configure all the IP subnets behind the LAN switch or LAN router or the multiple gateways on the WAN side.

By default, the routing table on an in-path interface has only the default gateway. This works fine if the IP subnet defined on the in-path interface is the same IP subnet as all the hosts behind the Steelhead appliance. If however there are multiple IP subnets behind the Steelhead appliance, either because the IP subnet is defined on the WAN router or because the IP subnet is defined on the LAN router, then all traffic for those subnets will be send to the default gateway before being send to the right IP subnet, traveling via the WAN interface back through the Steelhead appliance before it reaches the LAN router it had to end up in the beginning.

This is further explained in the IP Routing Related Issues section in the Operation Related Chapter.

2.3.4. Interface speeds

All interfaces can have either auto-negotiation defined or have a fixed speed and duplex setting defined. These days, general interoperability is not a reason to use a fixed speed and duplex anymore and auto-negotiation should be used everywhere unless not possible. This is especially important for gigabit speed interfaces, which is explained in a later section.

2.3.4.1. Fixed speed and duplex settings

Keep in mind that when a fixed speed and duplex setting is used that the LAN and the WAN interface and the devices connected to them should all be at the same speed and duplex settings.

If the interface speed and duplex settings of the LAN and WAN interfaces are different, no Ethernet link will be established if the Steelhead appliance is rebooted or turned off and all networks behind the Steelhead appliance will be unreachable.

2.3.4.2. Gigabit NICs with fixed speed

10 Mbps Ethernet and 100 Mbps Fast Ethernet network cards can be configured to a fixed speed and duplex settings which bypasses the negotiation phase of setting up of the Ethernet link between the devices.

There is no such feature as a fixed speed and duplex settings for the 1000 Mbps gigabit and faster NICs, the 802.3ab specification[SOURCE IEEE 802.3ab] stated that auto-negotiation is a requirement.

During the negotiation phase, each network card advertises the speeds it is capable of. With these "fixed speed" gigabit configurations, it will only advertise the 1000 Mbps speed and not the 100 Mbps and 10 Mbps speeds, thus forcing the device on the other side to accept the 1000 Mbps speed.

2.3.5. License keys

Riverbed uses license keys to determine which feature-sets are available on the Steelhead appliances in your network. The following licenses are the most common feature sets:

- Three license keys are included by default with every Steelhead appliance: The BASE, the CIFS and the MAPI licenses. Without these three licenses the Steelhead appliance won't be able to perform its basic optimization services.
- The SSL license allows optimization of SSL encrypted traffic and SSL tunneling for the secure inner channel feature. It can be obtained for without charge from Riverbed for Steelhead appliances installed in countries without USA export restrictions against them.
- The RSP license, which can be used to enable the RSP virtualization system.
- Model base and configuration upgrade licenses. With the development of the xx50 series models, the upgrade from a certain model to a higher capacity configuration can be done via a license key without the need for a hardware swap.

Licenses are only valid for the Steelhead appliance with the serial number they are ordered for. When a Steelhead appliance needs to be replaced via an RMA, replacement licenses will be submitted during the RMA process.

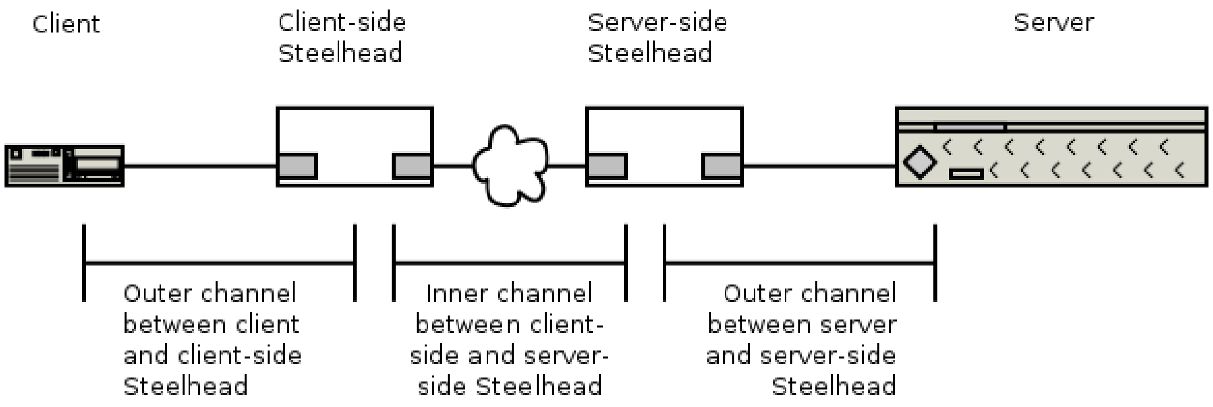
The list of license keys for a Steelhead appliance can be found on the Riverbed Support website under the assets list.

2.4. The layout of an optimized TCP session

The Riverbed implementation of an optimized TCP session is best described as a TCP tunnel: The internal channel between the two Steelhead appliances is transported via a TCP session. Of the optimized TCP session, only the TCP payload gets optimized, the details of the transport layer itself (the TCP header, IP header and Ethernet frame header) does not get forwarded.

Because of that design, there will be three different TCP sessions on the path between the client and the server, each which look similar but with their own characteristics and behaviour.

Figure 2.4. Three independent TCP sessions


Client
Client-side Steelhead
Server-side Steelhead
Server
Outer channel between client and client-side Steelhead
Inner channel between client-side and server-side Steelhead
Outer channel between server and server-side Steelhead

2.4.1. Three different TCP sessions

The first TCP session is between the client and the client-side Steelhead appliance. It will be a fast connection with a low latency, most likely to just behind the LAN switch where the client is located. This TCP session is setup by the client and will have the TCP capabilities that the client supports, for example if the client doesn't support TCP Window Scaling, then this TCP session will have a sliding window with a maximum size of 64 Kb. This should not be too much of a problem, since the latency on this part of the path is close to zero. This TCP session is called the client-side outer channel.

This TCP session has the following characteristics:

- The IP addresses are the IP addresses of the client and server.
- The TCP ports are the TCP ports used by the client and the server[*].
- The TCP Sequence numbers used are specific for the TCP session between the client and the client-side Steelhead appliance.
- It only supports the TCP features that the client offers.

The second TCP session is between the two Steelhead appliances. In general it will be a slow connection with a high latency going over the WAN links. The Steelhead appliances will use all possible TCP features they share, like TCP Window Scaling, TCP Fast Retransmission, TCP Selective ACK, TCP Timestamps and Packet Loss behaviour. This TCP session is called the inner channel.

This TCP session has the following characteristics (by default):

- The IP addresses are the IP addresses of the in-path interface of the Steelhead appliances.[**]
- The TCP ports used on the server-side Steelhead appliance is 7800, or 7810 for Fixed Target configurations.[**]
- The TCP Sequence numbers used are specific for the TCP session between the two Steelhead appliances.
- It supports the TCP features that the Steelhead appliances support.

The third TCP session is between the server-side Steelhead appliance and the server. It will be a fast connection with a low latency, most likely to just after the LAN switch where the server is located. The Steelhead appliance will offer all possible TCP features to the server, which might or might not support them. This TCP session is called the server-side outer channel.

This TCP session has normally the following characteristics:

- The IP addresses are the IP addresses of the client and server.

- The TCP ports are the TCP ports used by the client and the server[*].
- The TCP Sequence numbers are specific for the TCP session between the server-side Steelhead appliance and the server.
- It supports the TCP features that the server supports.

Because of these different characteristics on TCP level, especially the difference in TCP sequence numbers, optimized TCP sessions will not recover when the optimization service on one of the Steelhead appliance is restarted or traffic is rerouted to a path without a Steelhead appliance in place.

[*] The exception on this is the with MAPI latency optimization, there the destination TCP port of 7830 is used between the client and client-side Steelhead appliance.

[**] In the default Correct Addressing WAN Visibility.

2.5. The setup of an optimized TCP session.

To setup a new optimized TCP session, the Steelhead appliance at the client-side needs to know if there is another Steelhead in the path and where to setup the inner channel to.

The setup of a TCP session happens with the three way handshake[SOURCE:RFC 793] where the client sends a TCP packet with the SYN flag set in the TCP header, the server replies with a TCP packet with the SYN and ACK flags set in the TCP header and the client replies with a TCP packet with the ACK flag set in the TCP header.

Figure 2.5. The setup of a normal TCP session without Steelhead appliances

```
State                 Client                   Server    Time
Send TCP SYN              S >-----
                                  '-----                    |
Receive TCP SYN                         '----->             |
                                                            |
Send TCP SYN/ACK                         -----< SA          |
                                   -----'                   |
Receive TCP SYN/ACK         <-----'                         v

Send TCP ACK              A >-----
                                  '-----
Receive TCP ACK                         '----->

Data can be exchanged       >----------------->
forwards and backwards      <-----------------<

                            >-- initiated by
                            --> received by
```

With a pair of Steelhead appliances in the path, they will see this setup of the TCP session:

Figure 2.6. The setup of a normal TCP session seen by Steelhead appliances

```
Line State                 Client   CSH   SSH   Server    Time
  1  Send TCP SYN              S >->
  2                                    -->                   |
  3  Receive TCP SYN                         -->             |
  4                                                          |
  5  Send TCP SYN/ACK                        <-< SA          |
  6                                    <--                   |
  7  Receive TCP SYN/ACK         <--                         v
  8
  9  Send TCP ACK              A >->
 10                                    -->
 11  Receive TCP ACK                         -->

 12  Data can be exchanged       >--   ---   -->
 13  forwards and backwards      <--   ---   --<

                            >-- initiated by
                            --> received by
```

When the Steelhead appliance sees a TCP packet with the SYN flag set coming in on the LAN side, it will know that this is a new TCP session initiated by a local client. The Steelhead appliance will check the In-path Rules table to determine what should happen with the TCP session:

- It can be passed through, the Steelhead appliance does not intercept it.
- It can be optimized to another Steelhead appliance configured by a Fixed Target rule. This happens unconditionally, the other Steelhead appliances cannot ignore this.
- It can be tagged for optimization via the auto-discovery protocol: The Steelhead appliance adds an auto-discovery probe in the TCP options part of the TCP header so any other Steelhead appliance in the path will know that this TCP session can become an optimized TCP session. The other Steelhead appliance will check its Peering Rules table to determine if it wants to optimize with this Steelhead appliance or for this TCP session.

A TCP SYN packet without the auto-discovery probe is called a *naked SYN*. A TCP SYN packet with the auto-discovery probe is called a *SYN+*. A TCP SYN/ACK with the auto-discovery probe is called a *SYN/ACK+*.

If a naked SYN is seen on the WAN side of the Steelhead appliance, it will ignore it and mark the TCP session as pass-through Note that this is only valid for true in-path designs. For virtual in-path designs where the traffic is forwarded via WCCP, PBR or Interceptors, all traffic is received via the WAN interface and thus it will accept and process a naked SYN packet on the WAN side.

2.5.1. The Out-of-Band Splice

When two Steelhead appliances see each other for the first time, either via auto-discovery or via a fixed target rule, they will start with the setup an Out-of-Band (OOB) Splice. This is a control TCP session between the two Steelhead appliances, used to test the connectivity towards between the two Steelhead appliances.

Figure 2.7. Setup of a new OOB Splice

```
SH sport[24798]: [splice/oob.INFO] 1 {- -} New OOB Splice created at 10.0.1.6 for peer 192 \
    .168.1.6
SH sport[24798]: [mgmt.INFO] - {- -} mgmtd notified that remote peer 192.168.1.6 was disco \
    vered
SH sport[24798]: [splice/oob.INFO] 1 {- -} Establishing OOB Splice from 10.0.1.6:0 to 192. \
    168.1.6:7800
SH mgmtd[3766]: [mgmtd.INFO]: EVENT:  /rbt/sport/peer/event/added
SH sport[24798]: [splice/oob.INFO] 1 {- -} OOB Splice connected from 10.0.1.6:40271 to 192 \
    .168.1.6:7800
SH sport[24798]: [splice/oob.INFO] 1 {- -} Negotiated sport protocol version: 8 for peer:  \
    192.168.1.6:7800
```

It has the following behaviour:

- After the setup of the OOB Splice, the two Steelhead appliances will exchange system information and a list of capabilities. This information contains the hostname, RiOS version and IP addresses. This capabilities list will contain the latency optimizations supported and help the two Steelhead appliances to match the level of these capabilities.
- When a Steelhead appliance goes into admission control or the optimization service gets restarted, the OOB Splice will be closed.

  ### Figure 2.8. Termination of an OOB Splice due to admission control

  ```
  SH sport[6693]: [splice/oob.NOTICE] 8 {- -} Received disconnect cause="Admission control"  \
      laddr=10.0.1.6:7800 raddr=192.168.1.6:44264
  SH sport[6693]: [splice/oob.NOTICE] 8 {- -} Lost OOB Splice between laddr=10.0.1.6:7800 an \
      d raddr=192.168.1.6:7800
  ```

  The first line is the message that the other side goes into Admission Control. The second line is that the OOB Splice is about to be terminated.

- The OOB Splice has a TCP Keep Alive interval of 20 seconds. This means that if there is a network routing issue towards the other side, within one minute the Steelhead appliances will know that the other side is unreachable. Except for these TCP keep-alive packets there is no other data being transferred during normal operations.

2.5.2. The Connection Pool

Once the OOB Splice is setup, the optimization service will setup a Connection Pool towards the remote Steelhead appliance. This is a pool of 20 TCP sessions which can later be used to convert to inner channels. Every time one of these TCP sessions is used to create an inner channel, a replacement TCP session for the pool will be setup. The TCP Keep Alive interval is very large set to 20 minutes.

The Connection Pool sessions is maintained as long as the OOB Splice is established. If the OOB Splice gets terminated, the TCP sessions in the unused Connection Pool will be terminated too.

Figure 2.9. Termination of the Connection Pool because of the loss of the OOB Splice

```
SH sport[6693]: [splice/oob.NOTICE] 8 {- -} Lost OOB Splice between laddr=10.0.1.6:7800 an \
    d raddr=192.168.1.6:7800
SH sport[6693]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
SH sport[6693]: [spawnsplice.NOTICE] - {- -} (from 192.168.1.6:23210) End of stream readin \
    g splice hdr info. Peer maybe down.
```

2.5.3. In-path rules

The default set of in-path rules consist of three pass-through ("don't try to optimize this traffic") statements and a default auto-discovery rule:

- Pass-through of encrypted or secure traffic.
- Pass-through of interactive traffic.
- Pass-through of Riverbed protocols related to the optimization process.
- Auto-discovery of everything else.

Figure 2.10. In-path rules on a Steelhead appliance.

```
SH # show in-path rules
 Rule Type P O L N W K VLAN Source Addr        Dest Addr          Port
----- ---- - - - - - - ---- ------------------ ------------------ --------------
    1 pass - - - - - - all  all                all                Secure
    2 pass - - - - - - all  all                all                Interactive
    3 pass - - - - - - all  all                all                RBT-Proto
  def auto N F F A C N all  all                all                all

3 user-defined rule(s)

(P) Preoptimization Policy: O=Oracle-Forms S=SSL +=Oracle-Forms-over-SSL N=None
(O) Optimization Policy:    F=Full S=SDR-only C=Compression-only M=SDR-M N=None
(L) Latency Optimizations:  F=Full H=HTTP-only O=Outlook-anywhere N=None
(N) Neural Framing:         A=Always D=Dynamic T=TCP hints N=Never
(W) WAN Visibility Mode:    C=Correct-Addressing
                            P=Port-Transparency
                            F=Full-Transparency
                            R=Full-Transparency w/Reset
(K) Auto Kickoff:           Y=Enabled
                            N=Disabled
```

In-path rules can be used to prevent optimization towards certain machines or services. They can also be used to set specific options for specific optimized TCP session with regarding to WAN visibility, data reduction behaviour, pre-optimization features and so on.

The in-path rules are parsed in order of definition. Global pass-through, auto-discovery and fixed target in-path rules should be added after the default in-path rules. Specific auto-discovery in-path rules should be added to the end of the list of in-path rules. This way ensures that you prevent optimization before you start allowing it.

Figure 2.11. Additional fixed target in-path rule on a Steelhead appliance.

```
SH # show in-path rules
 Rule Type P O L N W K VLAN Source Addr        Dest Addr          Port
----- ---- - - - - - - ---- ----------------- ----------------- -------------
    1 pass - - - - - - all  all               all               Secure
    2 pass - - - - - - all  all               all               Interactive
    3 pass - - - - - - all  all               all               RBT-Proto
    4 fixe N F F A - N all  all               192.168.1.1/32    all
                               Target: 192.168.1.6       7810
  def auto N F F A C N all  all               all               all
```

2.5.3.1. Fixed Target in-path rules

Fixed Target rules specify the remote Steelhead appliance to be used for optimization of a certain set of traffic. It is an administrative task of the order of O(N!) if you would do it for every Steelhead appliance in the network.

Fixed Target rules are often used in environments where auto-discovery isn't possible because other devices in the network are stripping the auto-discovery probe or where the auto-discovery process fails because of policies on firewalls and NAT gateways.

Figure 2.12. The setup of an optimized TCP session using Fixed Target rules.

```
Line State                 Client   CSH   SSH   Server    Time
  1  Send TCP SYN              S >->
  2  Setup inner channel              >->                 |
       via Connection Pool
( 3  Setup of new TCP               S >->           )     |
( 4    session for the                <-< SA        )     |
( 5    Connection Pool              A >->           )     |
  6  Forward TCP SYN                      S >->           v
  7  Receive TCP SYN/ACK                    <-< SA
  8  Send TCP ACK                         A >->
  9                                   <-<
 10  Receive TCP SYN/ACK         <-< SA
 11  Send TCP ACK              A >->

 12  Data can be exchanged       >->   >->   >->
 13  forwards and backwards      <-<   <-<   <-<

                            >-- initiated by
                            --> received by
```

2.5.3.2. Auto-Discovery in-path rules

When a client-side Steelhead appliance adds an auto-discovery probe the TCP header of a TCP SYN packet, it will inform any other Steelhead appliances in the path that this TCP session can be optimized. The server-side Steelhead appliance will reply with a SYN/ACK+ packet containing its IP address and TCP port number. When the client-side Steelhead appliance sees the SYN/ACK+ packet, it will setup an inner channel towards the server-side Steelhead appliance.

Figure 2.13. The setup of an optimized TCP session using the default auto-discovery process

```
Line State                 Client   CSH   SSH   Server    Time
  1  Send TCP SYN              S >->
  2  Forward SYN+                   S+ >->                  |
  3  Receive SYN/ACK+                 <-< SA+               |
  4  Setup inner channel              >->                   |
( 5  Setup of new TCP               S >->           )     |
( 6    session for the                <-< SA        )     |
( 7    Connection Pool              A >->           )     |
  8  Forward TCP SYN                        S >->           v
  9  Receive TCP SYN/ACK                      <-< SA
 10  Send TCP ACK                           A >->
 11  Server side TCP is setup          <-<
 12  Receive TCP SYN/ACK         <-< SA
 13  Send TCP ACK              A >->

 14  Data can be exchanged       >->   >->   >->
 15  forwards and backwards      <-<   <-<   <-<

                            >-- initiated by
                            --> received by
```

In the default auto-discovery process the inner channel towards the first Steelhead appliances seen gets setup before the outer channel to the server is setup.

Figure 2.14. The setup of an optimized TCP session using the enhanced auto-discovery process

```
Line State                 Client   CSH   SSH   Server    Time
  1  Send TCP SYN              S >->
  2  Forward SYN+                   S+ >->                  |
  3  Tell CSH that a SH was seen      <-< SA*               |
  4  Forward SYN+                         S+ >->            |
  5  Receive TCP SYN/ACK                      <-< SA        |
  6  Receive SYN/ACK+                 <-< SA+               |
  7  Setup inner channel              >->                   |
( 8  Setup of new TCP               S >->           )     |
( 9    session for the                <-< SA        )     |
(10    Connection Pool              A >->           )     |
 11  Send TCP ACK                           A >->           v
 12  Server side TCP is setup          <-<
 13  Receive TCP SYN/ACK         <-< SA
 14  Send TCP ACK              A >->

 15  Data can be exchanged       >->   >->   >->
 16  forwards and backwards      <-<   <-<   <-<

                            >-- initiated by
                            --> received by
```

In the enhanced auto-discovery process the outer channel towards the server is determined before the inner channel is setup, making it possible to auto-discover the Steelhead appliance furthest in the network.

2.5.4. Peering rules

Peering rules define what the Steelhead appliance should do when it receives a SYN+ packet. It can chose to accept the auto-discovery attempt and reply with a SYN/ACK+ or to ignore it and pass it on.

Figure 2.15. Peering rules on a Steelhead appliance.

```
SSH (config) # show in-path peering rules
Rule  Type Source             Destination        Port   Peer             SSL-Cap
----- ---- ------------------ ------------------ ------ ---------------- -------
1     pass all                all                all    all              in-cap
      desc: Default rule to passthrough connections destined to SSL servers
2     auto all                all                443    all              cap
      desc: Default rule to attempt to optimize connections destined to port 443 as SSL
def   auto all                all                all    all              no-chk
--------------------------------------------------------------------------------
2 user added rule(s)
```

Peering rules cannot be used to block optimized TCP sessions from Fixed Target in-path rules coming from other Steelhead appliances.

2.6. WAN Visibility

As described earlier, an inner channel between two Steelhead appliances is a TCP session with the IP addresses of the in-path interfaces of the Steelhead appliances and the destination TCP port of 7800.

This means that your WAN devices don't know about the different traffic types anymore: All traffic is seen on TCP port 7800. As a result, QoS based on port number or IP subnet will fail and NetFlow collectors will show everything coming from only two hosts.

To overcome these issues, there are two additional WAN visibility methods:

- Port Transparency: Instead of using destination TCP port 7800, the inner channel uses the TCP port numbers of the original TCP session between the client and the server.
- Full Transparency: Like Port Transparency, but it also uses the IP addresses of the TCP session between the client and the server.

Port Transparency and Full Transparency work by adding a TCP option in the TCP packet header of the inner channel which identifies the IP addresses and TCP port numbers of the inner channel to the two Steelhead appliances. For Full Transparency, as long as the IP routing in the network makes sure that the traffic towards the IP subnets goes through the respective Steelhead appliances, this method will work fine.

Figure 2.16. Tcpdump output showing the Transparency TCP options

```
11:54:02.740843 IP 10.0.1.100.43802 > 192.168.1.1.445: Flags [S], seq 2130333957, win 5840 \
    , options [mss 1460,sackOK,TS val 284297166 ecr 0,nop,wscale 2,rvbd-trans Transp sSH:1 \
    0.0.1.6:54608 dSH:192.168.1.6:7800 00000a000106c0a80106d5501e78], length 0
11:54:02.872968 IP 192.168.1.1.445 > 10.0.1.100.43802: Flags [S.], seq 1657645820, ack 213 \
    0333958, win 5792, options [mss 1460,sackOK,TS val 284639662 ecr 284297166,nop,wscale  \
    2,rvbd-trans Transp sSH:192.168.1.6:7800 dSH:10.0.1.6:54608 0000c0a801060a0001061e78d5 \
    50], length 0
```

2.7. The optimization protocol

The protocol spoken between two Steelhead appliances on the inner channel consists of nine different commands: DEF, ESC, REF, REQ, REP, ACK, EOF, INF, LOP.

- Create a new DEFinition:

  With the DEF command the sending Steelhead appliance will tell the receiving Steelhead appliance to learn a new reference. The two Steelhead appliances will store the label and the frame in their data store and both now can use the label to send references later.

- ESCape this data:

  With the ESC command the sending Steelhead appliance will tell the receiving Steelhead appliance to forward this data to the client or server but not to store it in the data store. This can happen with data which is too small (less than 16 characters) or when the data reduction policy for an in-path rule is set to compression only or to no data reduction at all.

- Use this REFerence:

  With the REF command the sending Steelhead appliance tells the receiving Steelhead appliance to use the frame from an earlier learned reference.

- REQuest a reference:

  When a REF command comes in and the receiving Steelhead appliance cannot find the label in its data store, it will request the frame from the sending Steelhead appliance with this command.

- REPeat a reference:

  When a request for a reference gets answered by the original sending Steelhead appliance it will happen via a REP command.

- ACKnowledge an earlier command:

  With the ACK command an earlier DEF, ESC or REP statement gets acknowledged so the original sending Steelhead appliance knows that it has been processed properly.

- EOF: Outer channel was terminated:

  With the EOF (End Of File) command, the sending Steelhead appliance tells that the outer channel has been properly terminated.

- INF: INFormation packet:

  With the INF packet the receiving Steelhead appliance tells the sender about possible failures (Duplicate references, duplicate requests) and changes in the SDR-A optimization policy.

- LOP: Lower Overhead Protocol

  Like the ESC command, but with a much smaller protocol overhead (three bytes instead of 15 bytes).

2.8. Hardware models

There have been several generations of hardware, the xx10 series, the xx20 series, the xx50 series, the CX series and the EX series. In each generation there are several different models, from low end desktop units to high end data center rack mounted models.

2.8.1. Steelhead xx10 series models

The xx10 series models are obsoleted and are not discussed here. These models can run up to and including RiOS version 5.0.

2.8.2. Steelhead xx20 series models

The xx20 series models consist of a set of desktop models, 1RU rack models and 3RU rack models. To upgrade between models, except for the 50, 100, 200 and 300 models, the whole device must be replaced.

These models can run up to and including RiOS version 6.5.

2.8.2.1. Model 50, 100, 200, 300

The 50, 100, 200 and 300 models are desktop units. They have a single in-path interface pair and a single hard disk.

These models only differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.1. Model 50, 100, 200, 300

	Model 50	Model 100	Model 200	Model 300
WAN Capacity	256 Kbps	1 Mbps	1 Mbps	2 Mbps
TCP Connections	250	25	110	165
Data store size	35 Gb	35 Gb	35 Gb	35 Gb
Memory	512 Mb	512 Mb	512 Mb	512 Mb
On-board in-path interfaces	1	1	1	1
Expansion slots	0	0	0	0

2.8.2.2. Model 520, 1020, 1520, 2020

The 520, 1020, 1520 and 2020 models are 1RU units. They have two on-board in-path interface pairs and can have one extra bypass card for additional in-path interfaces.

Table 2.2. Model 520, 1020, 1520, 2020

	Model 520	Model 1020	Model 1520	Model 2020
WAN Capacity	1 Mbps	2 Mbps	4 Mbps	10 Mbps
TCP Connections	330	700	1 100	2 000
Data store size	80 Gb	80 Gb	80 Gb	150 Gb
Memory	2 Gb	2 Gb	2 Gb	4 Gb
On-board in-path interfaces	0	0	0	0
Expansion slots	1	1	1	1

2.8.2.3. Model 3020, 3520, 5520, 6020, 6120

The 3020, 3520, 5520, 6020 and 6120 models are 3RU units. They have two on-board in-path interface pairs and can have up to three extra bypass cards.

Table 2.3. Model 3020, 3520, 5520, 6020, 6120

	Model 3020	Model 3520	Model 5520	Model 6020	Model 6120
WAN Capacity	20 Mbps	45 Mbps	155 Mbps	310 Mbps	310 Mbps
TCP Connections	3 500	6 000	15 000	40 000	4 000
Data store size	250 Gb	512 Gb	700 Gb	1.4 Tb	3.4 Tb
Memory	4 Gb	6 Gb	8 Gb	16 Gb	16 Gb
On-board in-path interfaces	0	0	0	0	0
Expansion slots	3	3	3	3	3

2.8.3. Steelhead xx50 series models

Unlike the xx20 series models, where a hardware upgrade between similar models involved a replacement of the hardware, most of the xx50 series models can perform a configuration upgrade by using a license key.

2.8.3.1. Models 150, 250 and 550

The 150, 250 and 550 models are desktop units. They have a single in-path interface pair and a single hard disk.

The 150 model comes only in the M configuration.

Table 2.4. Model 150

	Model 150M
WAN Capacity	1 Mbps
TCP Connections	20
Data store size	40 Gb
Memory	1 Gb
On-board in-path interfaces	1
Expansion slots	0

The 250 model comes in L, M and H configurations which only differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.5. Model 250

	Model 250L	Model 250M	Model 250H
WAN Capacity	1 Mbps	1 Mbps	2 Mbps
TCP Connections	40	125	200
Data store size	40 Gb	40 Gb	40 Gb
Memory	1 Gb	1 Gb	1 Gb
On-board in-path interfaces	1	1	1
Expansion slots	0	0	0

The 550 model comes in M and H configurations which only differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.6. Model 550

	Model 550M	Model 550H
WAN Capacity	2 Mbps	4 Mbps
TCP Connections	300	600
Data store size	80 Gb	80 Gb
Memory	2 Gb	2 Gb
On-board in-path interfaces	1	1
Expansion slots	0	0

2.8.3.2. Models 1050 and 2050

The 1050 and 2050 models are 1RU units. They have two on-board in-path interface pairs and can have one extra in-path card.

The 1050 model comes in U, L, M and H configurations and with an optional RAID10 feature.

- The 1050U model has the same WAN and TCP capacity as the model 550H, but is capable of running 64-bit RSP instances.
- The 1050L and 1050M models have a single hard disk and have 2 Gb of memory.
- The 1050H model has two hard disks in a RAID0 configuration and has 4 Gb of memory.

- The 1050LR, 1050MR and 1050HR models contain four hard disks in a RAID10 configuration.

An upgrade towards a 1050H model requires the installation of extra memory, one extra hard disk and requires the data store to be re-initialized.

An upgrade towards a RAID configuration requires the installation of extra hard disks and will clear the data store.

Table 2.7. Model 1050

	Model 1050U	Model 1050L / 1050LR	Model 1050M / 1050MR	Model 1050H / 1050HR
WAN Capacity	4 Mbps	8 Mbps	10 Mbps	20 Mbps
TCP Connections	600	800	1 200	2 300
Data store size	80 Gb	100 Gb	100 Gb	200 Gb
Memory	2 Gb	2 Gb	2 Gb	4 Gb
On-board in-path interfaces	1	2	2	2
Expansion slots	0	1	1	1

The 2050 model comes in L, M and H configurations which only differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.8. Model 2050

	Model 2050L	Model 2050M	Model 2050H
WAN Capacity	45 Mbps	45 Mbps	45 Mbps
TCP Connections	2 500	4 000	6 000
Data store size	400 Gb	400 Gb	400 Gb
Memory	6 Gb	6 Gb	6 Gb
On-board in-path interfaces	2	2	2
Expansion slots	1	1	1

2.8.3.3. Models 5050 and 6050

The 5050 and 6050 models are 3RU units. They have two on-board in-path interface pairs and can have four extra in-path cards.

The 5050 model comes in L, M and H configurations which differ in the optimized WAN capacity, the number of optimized TCP sessions and the size of the data store.

An upgrade towards a 5050H model requires the installation of extra hard disks and will clear the data store.

Table 2.9. Model 5050

	Model 5050L	Model 5050M	Model 5050H
WAN Capacity	90 Mbps	90 Mbps	155 Mbps
TCP Connections	7 500	10 000	18 000
Data store size	600 Gb	600 Gb	800 Gb
Memory	8 Gb	8 Gb	8 Gb
On-board in-path interfaces	2	2	2

Expansion slots	4	4	4

The 6050 model comes in only one configuration.

Table 2.10. Model 6050

	Model 6050
WAN Capacity	310 Mbps
TCP Connections	50 000
Data store size	3.5 Tb
Memory	24 Gb
On-board in-path interfaces	2
Expansion slots	4

2.8.3.4. Model 7050

The 7050 model is a 3RU unit.

The 7050 model comes in L and M configurations which differ in number of optimized TCP sessions and in data store size. The storage of the data store in the 7050 model are solid-state disks instead of hard disks.

An upgrade towards a 7050M model requires the installation of extra memory, extra hard disks and but, because of the use of the FTS, will not require the re-initialization of the data store.

Table 2.11. Model 7050

	Model 7050L	Model 7050M
WAN Capacity	1 Gbps	1 Gbps
TCP Connections	75 000	100 000
Data store size	2.2 Tb	4.4 Tb
Memory	24 Gb	32 Gb
On-board in-path interfaces	2	2
Expansion slots	4	4

2.8.4. Steelhead CX series models

The CX series models are the "classic series", one of the successors of the xx50 series models. They are "classic" because they purely do WAN optimization and don't have the virtualization capabilities of the RSP/VSP features. Also the PFS capability has been removed from them.

2.8.4.1. Model CX255

The CX255 model is a desktop unit. It has only one in-path interface pairs and a single hard disk. There is no AUX port and the serial port is an RJ45 port.

The CX255 model comes in U, L, M and H configurations which differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.12. Model CX255

	Model CX255U	Model CX255L	Model CX255M	Model CX255H
WAN Capacity	2 Mbps	6 Mbps	6 Mbps	6 Mbps
TCP Connections	50	75	150	230

Data store size	50 Gb	50 Gb	50 Gb	50 Gb
Memory	2 Gb	2 Gb	2 Gb	2Gb
On-board in-path interfaces	1	1	1	1
Expansion slots	0	0	0	0

2.8.4.2. Model CX555 and CX755

The CX555 and CX755 models are desktop units. They have two in-path interface pairs and a dual hard disk.

The CX555 model comes in L, M and H configurations which differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.13. Model CX555

	Model CX555L	CX555M	Model CX555H
WAN Capacity	6 Mbps	10 Mbps	10 Mbps
TCP Connections	250	400	650
Data store size	80 Gb	80 Gb	80 Gb
Memory	2 Gb	2 Gb	2 Gb
On-board in-path interfaces	2	2	2
Expansion slots	0	0	0

The CX755 model comes in L, M and H configurations which differ in the optimized WAN capacity, the number of optimized TCP sessions and size of the data store.

The CX755H model comes with a single hard disk and a single solid-state disk, the latter used for the data store.

An upgrade towards a CX755M model does not require a change in hardware.

Table 2.14. Model CX755

	Model CX755L	Model CX755M	Model CX755H
WAN Capacity	10 Mbps	10 Mbps	20 Mbps
TCP Connections	900	1 500	2 300
Data store size	100 Gb	100 Gb	160 Gb
Memory	2 Gb	2 Gb	4 Gb
On-board in-path interfaces	2	2	2
Expansion slots	0	0	0

2.8.4.3. Model CX570 and CX770

The CX570 and CX770 models are desktop units. They have two in-path interface pairs and a hard disk and solid-state disk, the latter used for the data store.

The CX570 model comes in L, M and H configurations which differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.15. Model CX570

	Model CX570L	CX570M	Model CX570H

WAN Capacity	6 Mbps	10 Mbps	10 Mbps
TCP Connections	250	400	650
Data store size	70 Gb	70 Gb	70 Gb
Memory	2 Gb	2 Gb	2 Gb
On-board in-path interfaces	2	2	2
Expansion slots	0	0	0

The CX770 model comes in L, M and H configurations which differ in the optimized WAN capacity and the number of optimized TCP sessions.

Table 2.16. Model CX770

	Model CX770L	Model CX770M	Model CX770H
WAN Capacity	10 Mbps	10 Mbps	20 Mbps
TCP Connections	900	1 500	2 300
Data store size	150 Gb	150 Gb	150 Gb
Memory	4 Gb	4 Gb	4 Gb
On-board in-path interfaces	2	2	2
Expansion slots	0	0	0

2.8.4.4. Model CX1555

The CX1555 model is an 1RU units.

The CX1555 model comes in L, M and H configurations which differ in the number of optimized TCP sessions and size of the data store.

The CX1555L and CX1555M models have four hard disks in a RAID10 configuration, the CX1555H model has two hard disks in a RAID1 configuration and two SSD disks in an FTS configuration.

An upgrade towards a CX1555M model does not require a change in hardware, the upgrade towards a CX1555H model requires a replacement of two hard disks with two SSD disks.

Table 2.17. Model CX1555

	Model CX1555L	Model CX1555M	Model CX1555H
WAN Capacity	50 Mbps	50 Mbps	100 Mbps
TCP Connections	3 000	4 500	6 000
Data store size	400 Gb	400 Gb	320 Gb
Memory	8 Gb	8 Gb	8 Gb
On-board in-path interfaces	2	2	2
Expansion slots	2	2	2

2.8.4.5. Model CX5055 and CX7055

The CX5055 and CX7055 models are 2RU units.

Like the 7050 model, they use SSD disks in a Fault Tolerant Segstore method for the data store and use two hard disks for the operating system.

The CX5055 model comes in M and H configurations which differ in the optimized WAN capacity and the number of optimized TCP sessions.

An upgrade towards a CX5055H model does not require a change in hardware.

Table 2.18. Model CX5055

	Model CX5055M	Model CX5055H
WAN Capacity	200 Mbps	400 Mbps
TCP Connections	14 000	25 000
Data store size	640 Gb	640 Gb
Memory	16 Gb	16 GB
On-board in-path interfaces	2	2
Expansion slots	4	4

The CX7055 model comes in L, M and H configurations which differ in the optimized WAN capacity, the number of optimized TCP sessions, the amount of memory and the size of the data store.

The CX7055L model does not have a hardware compression card, the CX7055M and CX7055H models do have a hardware compression card.

The CX7055 models cannot be upgraded in the field, it requires a replacement of the unit.

Table 2.19. Model CX7055

	Model CX7055L	Model CX7055M	Model CX7055H
WAN Capacity	622 Mbps	1000 Mbps	1500 Mbps
TCP Connections	75 000	100 000	150 000
Data store size	1.6 Tb	2.4 Tb	4.8 Tb
Memory	32 Gb	48 Gb	64 Gb
On-board in-path interfaces	2	2	2
Expansion slots	4	4	4

2.8.5. Steelhead EX series models

The EX series models are the "edge series", one of the successors of the xx50 series. Unlike the CX series models they are capable of doing the virtualization with VSP and are ready to work with the Granite product. Like the CX series models, they don't have the PFS capabilities.

2.8.5.1. Models EX560, EX760 and EX1160

The EX560, EX760 and EX1160 models are 1RU units. They have two on-board in-path interfaces. They all use solid state disks for their data stores.

The EX560 model comes in G, L, M and H configurations. The EX560G model is only for use with the Granite product.

An upgrade towards an EX560L, EX560M and EX560H model does not require a change in hardware.

Table 2.20. Model EX560

	Model EX560L	Model EX560M	Model EX560H
WAN Capacity	6 Mbps	10 Mbps	10 Mbps

TCP Connections	250	400	650
Data store size	40 Gb	70 Gb	70 Gb
Memory	8 Gb	8 Gb	8 Gb
On-board in-path interfaces	2	2	2
Expansion slots	2	2	2

The EX760 model comes in L, M and H configurations.

An upgrade towards an EX760M and EX760H model does not require a change in hardware.

Table 2.21. Model EX760

	Model EX760L	Model EX760M	Model EX760H
WAN Capacity	10 Mbps	10 Mbps	20 Mbps
TCP Connections	900	1 500	2 300
Data store size	150 Gb	150 Gb	150 Gb
Memory	16 Gb	16 Gb	16 Gb
On-board in-path interfaces	2	2	2
Expansion slots	2	2	2

The EX1160 model comes in G, L, M, H and VH configurations. The EX1160G model is only for use with the Granite product.

An upgrade towards an EX1160L, EX1160M and EX1160H model does not require a change in hardware.

Table 2.22. Model EX1160

	Model EX1160L	Model EX1160M	Model EX1160H	Model EX1160VH
WAN Capacity	10 Mbps	10 Mbps	20 Mbps	50 / 100 Mbps
TCP Connections	900	1 500	2 3000	6 000
Data store size	140 Gb	140 Gb	140 Gb	280 Gb
Memory	20 Gb	20 Gb	20 Gb	24 Gb
On-board in-path interfaces	2	2	2	2
Expansion slots	2	2	2	2

The WAN capacity of an EX1160VH can be increased to 100 Mbps via a license key.

2.8.5.2. Model EX1260

The EX1260 model is a 2RU chassis.

The EX1260 model comes in G, L, M, H and VH configurations. The EX1260G model is only for use with the Granite product.

An upgrade towards an EX1260L, EX1260M and EX1260H model does not require a change in hardware.

Table 2.23. Model EX1260

	Model EX1260L	Model EX1260M	Model EX1260H	Model EX1260VH

WAN Capacity	10 Mbps	10 Mbps	20 Mbps	50 / 100 Mbps
TCP Connections	900	1 1 500	2 300	6 000
Data store size	140 Gb	140 Gb	140 Gb	280 Gb
On-board in-path interfaces	2	2	2	2
Expansion slots	4	4	4	4

The memory field has been removed from here because of the way the model EX1260 can be configured.

The WAN capacity of an EX1260VH can be increased to 100 Mbps via a license key.

2.8.5.3. Model EX1360

The EX1360 model is a 2RU model.

The EX1360 model comes in the G, L and M configurations. The EX1360G model is only for use with the Granite product.

An upgrade towards an EX1360L and EX1360M model does not require a change in hardware.

Table 2.24. Model EX1360

	Model EX1360L	Model EX1360M
WAN Capacity	50 Mbps	100 Mbps
TCP Connections	4 500	6 000
Data store size	320 Gb	320 Gb
Memory	24 Gb	24 Gb
On-board in-path interfaces	2	2
Expansion slots	4	4

2.8.6. Steelhead DX series models

The DX series models are the "data center series", for specific data-center to data-center environments. They have a memory-only data store and only limited latency optimization features.

The DX series models will only optimize against other DX series models.

2.8.6.1. Model DX8000

The DX8000 model is a 2RU unit. They have one on-board in-path interface.

Table 2.25. Model DX8000

	Model DX8000
WAN Capacity	no limit
TCP Connections	10 500
Data store size	memory only
Memory	128 Gb
On-board in-path interfaces	1
Expansion slots	-

2.8.7. Interceptor models

The two hardware models are the 9200 and 9350 model. The old 9200 model had a single hard disk and supported up to and including Interceptor software version 3.0.x. The new 9350 model has two hard disks in a RAID1 configuration.

2.8.8. Steelhead Mobile models

The two hardware models are the 8500 and 8650 model. The old 8500 model had a single hard disk and supported up to and including SMC software version 3.1.x. The new 8650 model has two hard disks in a RAID1 configuration.

The VM based SMC model has two virtual disks.

2.8.9. Central Management Console models

The two hardware models are the 8000 and 8150 model. The old 8000 model had a single hard disk and supported up to and including CMC software version 6.5.x. The new 8150 model has two hard disks in a RAID1 configuration.

The VM based CMC 8001 model has three virtual disks, two for the operating system and the CMC software and one for the data.

Chapter 3. The Command Line and Tools

3.1. Index

This chapter describes how to access and deal with the Command Line Interface (CLI) and the various troubleshooting tools available on the Steelhead appliances, how to use them and how to interpret the results.

The following tools are described:

- **tcpdump**, the network packet capture tool.
- **tcpdump-x**, a wrapper around tcpdump.
- **ping** and **ping6**, to determine the reachability of a remote host.
- **traceroute** and **traceroute6**, to determine the path to a remote host.
- **telnet**, to setup a TCP connection.
- **tproxytrace**, to detect Steelhead appliances in the network.
- **Nettest**, the internal network testing tools.
- **ssl-connect**, to setup an SSL encrypted TCP connection.

3.1.1. Introduction

The Steelhead appliance comes with a large set of built-in tools to investigate network related issues interfering with the setup and the operation of optimized TCP sessions. Knowing how to use them and what to expect in the output will reduce the time needed to finalize installations and identify operational issues.

Note: most of the tools use Linux specific options. Some of the tools use a different naming for the options on other operating systems such as FreeBSD and Microsoft Windows. See the "man" pages or the help section of these other operating systems to find the right options.

3.1.2. Network diagram

Most of the scenarios, logging and traces are obtained from the following network:

Figure 3.1. The network diagram

```
                              .-,(  ),-.
                           .-(          )-.
                     .--- (       WAN      ) ------.
                     |     '-(          ).-'       |
                     |         '-.( ).-'           |
                     |                             |
                     v                             v
                __________                     __________
               [_...__...*]                   [_...__...*]
                  Router                         Router
                    |                              |
                    |                              |
                    v                              v

                __________                     __________
               [    W     ]                   [    W     ]
               [    L  P  ]                   [    L  P  ]
               [_...__...*]                   [_...__...*]
                   SSH |                          CSH |
                    |  |                           |  |
 ________           |  |                           |  |
|==|=====|          v  v                           v  v
|  |     |      __________                     __________
|  |     |     [_...__...*]                   [_...__...*]
|  |     |        Switch                         Switch
|  |     |          |                              |          _  __
|  |====*|          |                              |         |=|[__]
|__|_____|<---------'                              '-------->|_|\::\
  Server                                                     Client

Client:             10.0.1.1        d4:9a:20:c2:52:0e
C-SWI:              10.0.1.8
CSH PRI:            10.0.1.5        00:0e:b6:42:f8:98
CSH 0_0 LAN:        10.0.1.6        00:0e:b6:8c:74:9c
CSH 0_0 WAN:        10.0.1.6        00:0e:b6:8c:74:9b
C-RTR-lan:          10.0.1.9        00:0d:b9:17:28:de
C-RTR-wan           172.16.1.1
                    30 ms
N-RTR-1-wan:        172.16.1.2
N-RTR-1-wan:        172.16.2.1
                    60 ms
N-RTR-2-wan:        172.16.2.2
N-RTR-2-wan:        172.16.3.1
                    40 ms
S-RTR-wan:          172.16.3.2
S-RTR-lan           192.168.1.9     00:0d:b9:17:28:dd
SSH 0_0 WAN:        192.168.1.6     00:0e:b6:8c:7a:ed
SSH 0_0 LAN:        192.168.1.6     00:0e:b6:8c:7a:ee
SSH PRI:            192.168.1.5     00:0e:b6:31:45:e0
S-SWI:              192.168.1.8
Server:             192.168.1.1     00:21:70:3e:6b:7f
```

3.2. Dealing with the Command Line Interface

3.2.1. Accessing the CLI via the serial console

The serial console of the Steelhead appliance can be used to access the CLI if the device does not have network connectivity yet. The serial console assumes that the screen size of the terminal is 80x25 characters. If the screen size of the terminal is different, then the output to the console will become garbled when the pager is used, for example during the display of the configuration with the commands `show running` and `show log`.

3.2.2. Accessing the CLI via SSH

The Command Line Interface of the Steelhead appliance can be accessed via a Secure Shell application, SSH. The best known SSH application for OS X, Unix and Linux hosts is OpenSSH, the best known for Microsoft Windows is PuTTY, available from www.chiark.greenend.org.uk/~sgtatham/putty/.

Figure 3.2. Setup of an SSH session to the CLI of the Steelhead appliance

```
$ ssh admin@10.0.1.5
The authenticity of host '10.0.1.5 (10.0.1.5)' can't be established.
ECDSA key fingerprint is 7f:e2:c5:18:db:17:08:91:2d:44:e9:7a:b4:97:f7:92.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.1.5' (ECDSA) to the list of known hosts.
Riverbed Steelhead
Password:
SH > en
SH # cli session auto-logout 0
SH # configure terminal
SH (config) #
```

After being logged in the CLI session is a limited command mode, identified by the '>' in the prompt. Use the command `enable` to elevate to the more powerful enable mode, identified by the '#' in the prompt. To be able to make changes in the configuration mode, use the command `configure terminal` and the prompt will have the string '(config)' in it. Use the command `exit` to go one level lower.

The first line complains that the SSH application hasn't seen the SSH fingerprint of this host before. This is normal for the first time an SSH connection is made to this host. If the device has been replaced via an RMA, then the replacement will have a different SSH fingerprint and SSH will complain about it:

Figure 3.3. Setup of an SSH session to the CLI of a replaced Steelhead appliance

```
$ ssh admin@10.0.1.5
Riverbed Steelhead
Password:
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
79:8f:b4:7e:4b:3c:4a:fb:ad:e6:f6:fe:56:c1:50:2c.
Please contact your system administrator.
Add correct host key in /usr/home/edwin/.ssh/known_hosts to get rid of this message.
Offending RSA key in /usr/home/edwin/.ssh/known_hosts:50
RSA host key for 10.0.1.5 has changed and you have requested strict checking.
Host key verification failed.
```

If this happens, confirm that the device was replaced.

To overcome this issue with OpenSSH, remove the line in the file mentioned. With PuTTY a dialog box will be shown with the option to replace the key.

3.2.3. Accessing the file system via SCP

Secure Copy, or SCP, is a file transfer layer on top of SSH. It can be used to upload RiOS software images, RSP images and RSP packages and to download system dumps, process dumps and tcpdump traces.

The OpenSSH software includes the scp binary. The PuTTY website distributes a SCP program called "pscp.exe".

The following limitations are implemented on the Steelhead appliance:

- You can only upload to:
- /images for RiOS images.
- /rsp/images for RSP images.
- /rsp/packages for RSP packages.
- You can only download from:
- /sysdumps for system dumps.

- /snapshots for process dumps.
- /tcpdumps for tcpdumps captures.
- /log for log files.

Note that only SCP is permitted. SFTP is not supported, neither is FTP over SSL.

Figure 3.4. Download of a system dump via SCP

```
$ scp admin@10.0.1.5:/sysdumps/sysdump-CSH-20120612-123456.tgz .
Riverbed Steelhead
Password:
sysdump-CSH-2012-123456.tgz                        100% 34Mb   400KB/s  00:00
```

Figure 3.5. Attempt to access a non-permitted directory

```
$ scp admin@10.0.1.5:/var/opt/tms/image-history .
Riverbed Steelhead
Password:
Riverbed ssh: ssh remote command is not allowed.
```

3.2.4. Surviving the 'less' pager

The pager used on the Steelhead appliance to view multiple pages of output is based on the 'less' pager, known from Unix and Linux environments. If the last character on the screen is a "`:`" or "`(END)`" then you are in this pager.

Here is a list of keys and what their function is:

Table 3.1. The less pager cheat sheet

Key	Functionality
q	Quit the pager (very important)
Enter or j or arrow-down	Scroll one line down
k or y or arrow-up	Scroll one line up
Space	Scroll one page down
b	Scroll one page backwards
control-G	Show information of the file, size and location
control-L	Redraw the screen, in case it got garbled
/	Search forward
?	Search backward
n	Perform the last search again, forwards or backwards
g	Scroll to the beginning of the file
G	Scroll to the end of the file

The search string is a regular expression, so some magic can be performed with it.

Normally, the less pager starts at the beginning of the text to display. However, when viewing a log files the less pager will start at the end. Use the 'b' key to go backwards!

3.2.5. Control keys on the CLI

The following combination of control keys are available to navigate on the CLI:

Table 3.2. Control characters on the CLI

^U	Delete everything from the cursor to the beginning of the line

^W	Delete the word left of the cursor
^H	Delete the character in front of the cursor
^D	Delete the character under the cursor. If no characters are left, exit the CLI session.
^Y	Paste the previously deleted string
^L	Clear the screen and write the current entered command at the top
^A	Go to the beginning of the line
^E	Go to the end of the line
^C	Abort the current command
^P or arrow up	Show the previous command executed
^N or arrow down	Show the next command (if ^P has been used)
^S	Stop scrolling on the the CLI
^Q	Continue scrolling on the CLI
^V	Interpret the next key literally

3.2.6. Backspace doesn't work

When the backspace is pressed and a `^?` shows up, try control-H to delete the character.

When the backspace is pressed and a `^H` shows up, try control-backspace to delete the character.

3.3. Tcpdump, the network packet capture tool

Being able to see the traffic that goes in and out a network card is great value to determine what occurs on network level. Tcpdump, the standard network packet capture tool, is included on the Steelhead appliances. It uses the pcap library for the capturing of the packets on the NIC.

The place where tcpdump captures the data from the network stack is just before the sending or just after the receiving of the data to and from the network card. This makes sure that the capture happens with the data as closely as it is on the network cable as possible.

Tcpdump supports both IPv4 and IPv6 addresses.

3.3.1. Tcpdump options

- The option `-n` will prevent DNS name resolution of the IP addresses and services detected in the captured packet.
- The option `-s` specifies the number of bytes to read from the captured packets.
- The option `-i` will select the interface to capture on.
- The option `-c` specifies the limit on the number of packets captured.
- The option `-w` specifies the filename to write the capture to.
- The option `-v` will print how many packets are captured and written to disk.
- The option `-le` will print the link layer information of the captured packet.
- The option `-X` (capital X) will also print contents of the captured packet in hex and character dump format.

3.3.2. Usage hints for tcpdump

- Always use the `-n` option to disable name resolution otherwise it will slow down the display of packets and possible dropping of captured packets.
- Don't capture on the in-path interfaces, always capture on the corresponding LAN and WAN interfaces.
- Don't capture on the *primary* interface, always capture on the *prihw* interface.
- Use `-c` to limit the number of packets captured to prevent them from scrolling of the screen.
- When writing captured packets to disk, use the `-v` option to see the number of packets written.
- Use correct filters to capture only the packets you are interested in.
- The default filter is based on the standard Ethernet frame headers. Use the `vlan` primitive to switch to the 802.1Q VLAN Ethernet frame format.
- Use a snaplength of 1600 for a full payload capture.

3.3.3. Auto-discovery and Full transparency / Port Transparency options

The normal tcpdump program will not be able to decode the Riverbed TCP Options and will display them as a hex-string:

Figure 3.6. The Riverbed specific TCP options not decoded in tcpdump before RiOS 6.1.4 and 6.5.2

```
11:12:47.668647 IP 10.0.1.1.52175 > 192.168.1.1.80: Flags [S], seq 4101063156, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 693134564 ecr 0,sackOK,nop,nop,opt-76:01 \
    010a0001060005,opt-76:0c01,nop,eol], length 0
11:12:47.801418 IP 192.168.1.1.80 > 10.0.1.1.52175: Flags [S.], seq 20020520, ack 41010631 \
    57, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 693134564 ecr 0,sackOK,no \
    p,nop,opt-76:0c01,nop,nop,nop,eol], length 0
11:12:47.801454 IP 192.168.1.1.80 > 10.0.1.1.52175: Flags [S.], seq 20020520, ack 41010631 \
    57, win 65535, options [mss 1460,nop,wscale 3,TS val 693134564 ecr 0,sackOK,opt-76:111 \
    10a000106c0a801061e78,opt-76:0e3d,nop,eol], length 0
```

With the release of RiOS 6.1.4 and 6.5.2, the tcpdump program included on the Steelhead appliances will decode the auto-discovery and the WAN Visibility Transparency TCP options:

Figure 3.7. The Riverbed specific TCP options for auto-discovery decoded in tcpdump after RiOS 6.1.4 and 6.5.2

```
11:12:47.668647 IP 10.0.1.1.52175 > 192.168.1.1.80: Flags [S], seq 4101063156, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 693134564 ecr 0,sackOK,nop,nop,rvbd-prob \
    e AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
11:12:47.801418 IP 192.168.1.1.80 > 10.0.1.1.52175: Flags [S.], seq 20020520, ack 41010631 \
    57, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 693134564 ecr 0,sackOK,no \
    p,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
11:12:47.801454 IP 192.168.1.1.80 > 10.0.1.1.52175: Flags [S.], seq 20020520, ack 41010631 \
    57, win 65535, options [mss 1460,nop,wscale 3,TS val 693134564 ecr 0,sackOK,rvbd-probe \
     AD CSH:10.0.1.6 SSH:192.168.1.6:7800 11110a000106c0a801061e78,rvbd-probe EAD 0e3d,nop \
    ,eol], length 0
```

For the Auto-Discovery process, it will show the IP addresses of the CSH (Client-side Steelhead appliance) and the SSH (Server-side Steelhead appliance).

- The keyword `rvbd-probe` tells that the option is TCP option 76 for the auto-discovery process.
- The keyword `AD` tells that the option is part of the auto-discovery process.
- The keyword `EAD` tells that the option is part of the enhanced auto-discovery process.

- The keyword `CSH` gives the IP address of the client-side Steelhead appliance.
- The keyword `SSH` gives the IP address and TCP port of the server-side Steelhead appliance.

For the WAN Visibility Transparency feature, it will show the IP addresses and TCP port numbers of the inner channel.

Figure 3.8. The Riverbed specific TCP options for WAN Visibility Transparency feature decoded in tcpdump after RiOS 6.1.4 and 6.5.2

```
12:02:03.274738 IP 10.0.1.1.3789 > 192.168.1.1.80: Flags [P.], seq 1:11, ack 7, win 4592,  \
    options [nop,nop,TS val 958790087 ecr 1176840218,rvbd-trans Transp sSH:10.0.1.6:42455  \
    dSH:192.168.1.1:7800 00000a000106c0a80106a5d71e78], length 10
12:02:03.288821 IP 192.168.1.1.80 > 10.0.1.1.3789: Flags [.], seq 7, ack 11, win 23, optio \
    ns [nop,nop,TS val 1176840247 ecr 958790087,rvbd-trans Transp sSH:192.168.1.6:7800 dSH \
    :10.0.1.6:42455 0000c0a801060a0001061e78a5d7], length 0
```

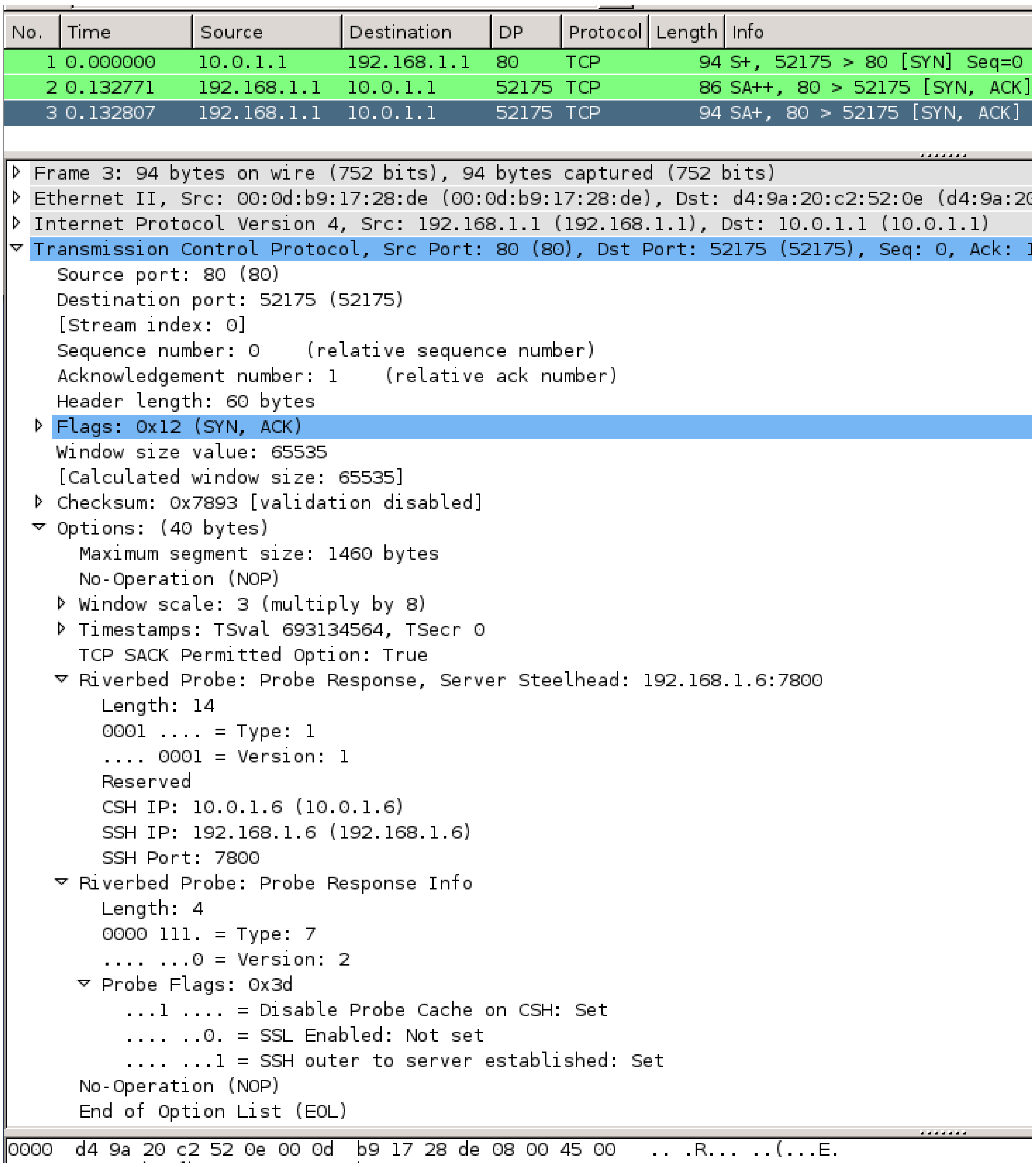
- The keyword `rvbd-trans` tells that the option is TCP option 78 for the WAN Visibility Full Transparency feature.
- The keyword `Transp` tells that the standard transparency options are used.
- The keyword `sSH` gives the IP address and TCP port of the sending Steelhead appliance.
- The keyword `dSH` gives the IP address and TCP port of the destination Steelhead appliance.

With the release of Wireshark version 1.6.0, it can decode the auto-discovery and the transparency TCP options.

Figure 3.9. The Riverbed specific TCP options decoded in tshark

```
$ tshark -O tcp -V -nr autodiscovery.cap
Frame 1: 94 bytes on wire (752 bits), 94 bytes captured (752 bits)
Ethernet II, Src: d4:9a:20:c2:52:0e (d4:9a:20:c2:52:0e), Dst: 00:0d:b9:17:28:de (00:0d:b9: \
    17:28:de)
Internet Protocol Version 4, Src: 10.0.1.1 (10.0.1.1), Dst: 192.168.1.1 (192.168.1.1)
Transmission Control Protocol, Src Port: 52175 (52175), Dst Port: 80 (80), Seq: 0, Len: 0
    Source port: 52175 (52175)
    Destination port: 80 (80)
    [Stream index: 0]
    Sequence number: 0    (relative sequence number)
    Header length: 60 bytes
    Flags: 0x02 (SYN)
    Window size value: 65535
    [Calculated window size: 65535]
    Checksum: 0xe76d [validation disabled]
    Options: (40 bytes)
        Maximum segment size: 1460 bytes
        No-Operation (NOP)
        Window scale: 3 (multiply by 8)
        No-Operation (NOP)
        No-Operation (NOP)
        Timestamps: TSval 693134564, TSecr 0
        TCP SACK Permitted Option: True
        No-Operation (NOP)
        No-Operation (NOP)
        Riverbed Probe: Probe Query, CSH IP: 10.0.1.6
            Length: 10
            0000 .... = Type: 0
            .... 0001 = Version: 1
            Reserved
            CSH IP: 10.0.1.6 (10.0.1.6)
            Application Version: 5
        Riverbed Probe: Probe Query Info
            Length: 4
            0000 110. = Type: 6
            .... ...0 = Version: 2
            Probe Flags: 0x01
                .... .0.. = Not CFE: Not set
                .... ...1 = Last Notify: Set
        No-Operation (NOP)
        End of Option List (EOL)

Frame 2: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
Ethernet II, Src: 00:0d:b9:17:28:de (00:0d:b9:17:28:de), Dst: d4:9a:20:c2:52:0e (d4:9a:20: \
    c2:52:0e)
Internet Protocol Version 4, Src: 192.168.1.1 (192.168.1.1), Dst: 10.0.1.1 (10.0.1.1)
Transmission Control Protocol, Src Port: 80 (80), Dst Port: 52175 (52175), Seq: 0, Ack: 1, \
     Len: 0
    Source port: 80 (80)
    Destination port: 52175 (52175)
    [Stream index: 0]
    Sequence number: 0    (relative sequence number)
    Acknowledgement number: 1    (relative ack number)
    Header length: 52 bytes
    Flags: 0x12 (SYN, ACK)
    Window size value: 65535
    [Calculated window size: 65535]
    Checksum: 0xe020 [validation disabled]
    Options: (32 bytes)
        Maximum segment size: 1460 bytes
        No-Operation (NOP)
        Window scale: 3 (multiply by 8)
        No-Operation (NOP)
        No-Operation (NOP)
        Timestamps: TSval 693134564, TSecr 0
        TCP SACK Permitted Option: True
        No-Operation (NOP)
        No-Operation (NOP)
        Riverbed Probe: Probe Query Info
            Length: 4
            0000 110. = Type: 6
            .... ...0 = Version: 2
            Probe Flags: 0x01
                .... .0.. = Not CFE: Not set
                .... ...1 = Last Notify: Set
        No-Operation (NOP)
        No-Operation (NOP)
        No-Operation (NOP)
        End of Option List (EOL)
    [SEQ/ACK analysis]
        [This is an ACK to the segment in frame: 1]
        [The RTT to ACK the segment was: 0.132771000 seconds]

Frame 3: 94 bytes on wire (752 bits), 94 bytes captured (752 bits)
Ethernet II, Src: 00:0d:b9:17:28:de (00:0d:b9:17:28:de), Dst: d4:9a:20:c2:52:0e (d4:9a:20: \
    c2:52:0e)
Internet Protocol Version 4, Src: 192.168.1.1 (192.168.1.1), Dst: 10.0.1.1 (10.0.1.1)
Transmission Control Protocol, Src Port: 80 (80), Dst Port: 52175 (52175), Seq: 0, Ack: 1, \
     Len: 0
    Source port: 80 (80)
    Destination port: 52175 (52175)
    [Stream index: 0]
    Sequence number: 0    (relative sequence number)
    Acknowledgement number: 1    (relative ack number)
    Header length: 60 bytes
    Flags: 0x12 (SYN, ACK)
    Window size value: 65535
    [Calculated window size: 65535]
    Checksum: 0x7893 [validation disabled]
    Options: (40 bytes)
        Maximum segment size: 1460 bytes
        No-Operation (NOP)
        Window scale: 3 (multiply by 8)
        Timestamps: TSval 693134564, TSecr 0
        TCP SACK Permitted Option: True
        Riverbed Probe: Probe Response, Server Steelhead: 192.168.1.6:7800
            Length: 14
            0001 .... = Type: 1
            .... 0001 = Version: 1
            Reserved
            CSH IP: 10.0.1.6 (10.0.1.6)
            SSH IP: 192.168.1.6 (192.168.1.6)
            SSH Port: 7800
        Riverbed Probe: Probe Response Info
            Length: 4
            0000 111. = Type: 7
            .... ...0 = Version: 2
            Probe Flags: 0x3d
                ...1 .... = Disable Probe Cache on CSH: Set
                .... ..0. = SSL Enabled: Not set
                .... ...1 = SSH outer to server established: Set
        No-Operation (NOP)
        End of Option List (EOL)
```

Figure 3.10. The Riverbed specific TCP options decoded in Wireshark



```
No.  Time      Source       Destination  DP     Protocol Length Info
  1 0.000000   10.0.1.1     192.168.1.1  80     TCP          94 S+, 52175 > 80 [SYN] Seq=0
  2 0.132771   192.168.1.1  10.0.1.1     52175  TCP          86 SA++, 80 > 52175 [SYN, ACK]
  3 0.132807   192.168.1.1  10.0.1.1     52175  TCP          94 SA+, 80 > 52175 [SYN, ACK]

Frame 3: 94 bytes on wire (752 bits), 94 bytes captured (752 bits)
Ethernet II, Src: 00:0d:b9:17:28:de (00:0d:b9:17:28:de), Dst: d4:9a:20:c2:52:0e (d4:9a:2
Internet Protocol Version 4, Src: 192.168.1.1 (192.168.1.1), Dst: 10.0.1.1 (10.0.1.1)
Transmission Control Protocol, Src Port: 80 (80), Dst Port: 52175 (52175), Seq: 0, Ack: 1
  Source port: 80 (80)
  Destination port: 52175 (52175)
  [Stream index: 0]
  Sequence number: 0    (relative sequence number)
  Acknowledgement number: 1    (relative ack number)
  Header length: 60 bytes
  Flags: 0x12 (SYN, ACK)
  Window size value: 65535
  [Calculated window size: 65535]
  Checksum: 0x7893 [validation disabled]
  Options: (40 bytes)
    Maximum segment size: 1460 bytes
    No-Operation (NOP)
    Window scale: 3 (multiply by 8)
    Timestamps: TSval 693134564, TSecr 0
    TCP SACK Permitted Option: True
    Riverbed Probe: Probe Response, Server Steelhead: 192.168.1.6:7800
      Length: 14
      0001 .... = Type: 1
      .... 0001 = Version: 1
      Reserved
      CSH IP: 10.0.1.6 (10.0.1.6)
      SSH IP: 192.168.1.6 (192.168.1.6)
      SSH Port: 7800
    Riverbed Probe: Probe Response Info
      Length: 4
      0000 111. = Type: 7
      .... ...0 = Version: 2
      Probe Flags: 0x3d
        ...1 .... = Disable Probe Cache on CSH: Set
        .... ..0. = SSL Enabled: Not set
        .... ...1 = SSH outer to server established: Set
    No-Operation (NOP)
    End of Option List (EOL)

0000  d4 9a 20 c2 52 0e 00 0d  b9 17 28 de 08 00 45 00   .. .R... ..(...E.
```

3.3.4. Pcap filters

There are several basic pcap primitives to be understood before tcpdump can be used effectively:

- The primitive `host <IP address>` filters the packets to match this source or destination IP address.
- The primitive `net <IP subnet>` filters the packets to match this source or destination IP subnet.
- The primitive `port <port number>` filters the packets to match this source or destination TCP or UDP port numbers.
- The primitive `ip` and `ip6` filters explicitly for IPv4 and IPv6 addresses.
- The primitive `icmp` filters for only ICMP packets, the primitive `tcp` filters for only TCP packets, the primitive `udp` filters for only UDP packets.
- The primitive `vlan` is not so much a filter but more a change in the offset in the filter: From this part of the filter, the offset for the IP packet is indexed to the size of a 802.1Q VLAN tagged Ethernet frame.

The various primitives can be merged with a boolean "and" and "or". But keep in mind that, unlike boolean logic, the "and" and "or" have the same precedence. The precedence can be changed by using round brackets around a part of the filter.

For example, if the filter needs to capture all traffic from host H1 on port P1 or from host H2 on port P2, then `host H1 and port P1 or host H2 and port P2` gets interpreted as `((((host H1) and port P1) or host H2) and port P2)`. However, what was needed was: `(host H1 and port P1) or (host H2 and port P2)`.

Always use round brackets to prioritize parts of the filter!

3.3.5. Examples of tcpdump scenarios

3.3.5.1. Display the first ten packets from or to one host

The first example is capturing the first 10 packets to host 10.0.1.1.

Figure 3.11. Display the first ten packets from and to a certain host

```
SH # tcpdump -ni lan0_0 -c 10 'host 10.0.1.1'
tcpdump: WARNING: lan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
12:07:07.298829 ARP, Request who-has 10.0.1.6 tell 10.0.1.1, length 46
12:07:07.298872 ARP, Reply 10.0.1.6 is-at 00:0e:b6:8c:74:9c, length 28
12:07:08.194021 IP 10.0.1.1.52902 > 202.83.176.1.53: 53837 [1au] A? www.mavetju.org. (44)
12:07:08.234151 IP 202.83.176.1.53 > 10.0.1.1.52902: 53837*- 1/2/3 A 192.168.1.1 (141)
12:07:09.464621 IP 10.0.1.1.42825 > 192.168.1.1.80: Flags [S], seq 3872240793, win 65535,  \
    options [mss 1460,nop,wscale 6,sackOK,TS val 5351449 ecr 0], length 0
12:07:09.598157 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [S.], seq 2971872340, ack 387224 \
    0794, win 5792, options [mss 1460,sackOK,TS val 285084142 ecr 5351449,nop,wscale 2], l \
    ength 0
12:07:09.602342 IP 10.0.1.1.42825 > 192.168.1.1.80: Flags [.], seq 1, ack 1, win 1040, opt \
    ions [nop,nop,TS val 5351588 ecr 285084142], length 0
12:07:09.602375 IP 10.0.1.1.42825 > 192.168.1.1.80: Flags [P.], seq 1:188, ack 1, win 1040 \
    , options [nop,nop,TS val 5351589 ecr 285084142], length 187
12:07:09.602393 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [.], seq 1, ack 188, win 1716, o \
    ptions [nop,nop,TS val 285084146 ecr 5351589], length 0
12:07:09.875867 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [P.], seq 1:1151, ack 188, win 1 \
    716, options [nop,nop,TS val 285084420 ecr 5351589], length 1150
10 packets captured
14 packets received by filter
0 packets dropped by kernel
```

In the example above, 14 packets were captured from the network and 10 matched the filter. Zero packets could not be processed in time by the pcap filter.

The packets are: an Ethernet broadcast ARP request and reply, a DNS request and answer and then a TCP SYN, SYN/ACK and ACK to setup the TCP session and some data towards a web server.

3.3.5.2. Display the first ten packets between two hosts

For troubleshooting, capturing traffic between two hosts is required. Here we filter between the hosts 10.0.1.1 and 192.168.1.1.

Figure 3.12. Display the first ten packets between hosts 10.0.1.1 and 192.168.1.1

```
SH # tcpdump -ni lan0_0 -c 10 'host 10.0.1.1 and host 192.168.1.1'
tcpdump: WARNING: lan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
12:07:09.464621 IP 10.0.1.1.42825 > 192.168.1.1.80: Flags [S], seq 3872240793, win 65535,  \
    options [mss 1460,nop,wscale 6,sackOK,TS val 5351449 ecr 0], length 0
12:07:09.598157 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [S.], seq 2971872340, ack 387224 \
    0794, win 5792, options [mss 1460,sackOK,TS val 285084142 ecr 5351449,nop,wscale 2], l \
    ength 0
12:07:09.602342 IP 10.0.1.1.42825 > 192.168.1.1.80: Flags [.], seq 1, ack 1, win 1040, opt \
    ions [nop,nop,TS val 5351588 ecr 285084142], length 0
12:07:09.602375 IP 10.0.1.1.42825 > 192.168.1.1.80: Flags [P.], seq 1:188, ack 1, win 1040 \
    , options [nop,nop,TS val 5351589 ecr 285084142], length 187
12:07:09.602393 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [.], seq 1, ack 188, win 1716, o \
    ptions [nop,nop,TS val 285084146 ecr 5351589], length 0
12:07:09.875867 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [P.], seq 1:1151, ack 188, win 1 \
    716, options [nop,nop,TS val 285084420 ecr 5351589], length 1150
12:07:09.979743 IP 10.0.1.100.42825 > 192.168.1.6.80: Flags [.], seq 188, ack 1151, win 10 \
    40, options [nop,nop,TS val 5351967 ecr 285084420], length 0
12:07:10.881288 IP 10.0.1.100.42825 > 192.168.1.6.80: Flags [P.], seq 188:420, ack 1151, w \
    in 1040, options [nop,nop,TS val 5352869 ecr 285084420], length 232
12:07:10.881392 IP 192.168.1.6.80 > 10.0.1.100.42825: Flags [.], seq 1151, ack 420, win 17 \
    16, options [nop,nop,TS val 285085425 ecr 5352869], length 0
12:07:11.106812 IP 192.168.1.6.80 > 10.0.1.100.42825: Flags [P.], seq 1151:1663, ack 420,  \
    win 1716, options [nop,nop,TS val 285085651 ecr 5352869], length 512
10 packets captured
18 packets received by filter
0 packets dropped by kernel
```

The packets shown are the setup of a HTTP TCP session and the exchange of the data.

3.3.5.3. Display packets on a VLAN trunk

The difference in length of a normal IEEE 802.3 Ethernet frame and a IEEE 802.1q VLAN Ethernet frame is four bytes:

Figure 3.13. A normal 802.3 Ethernet frame

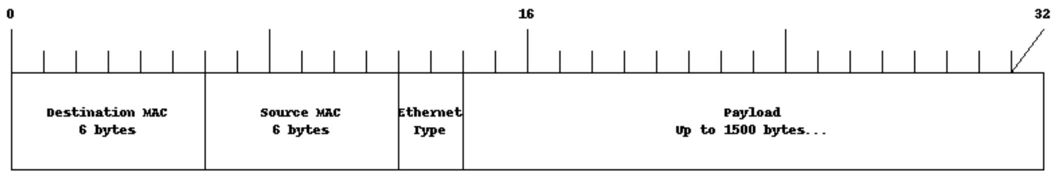
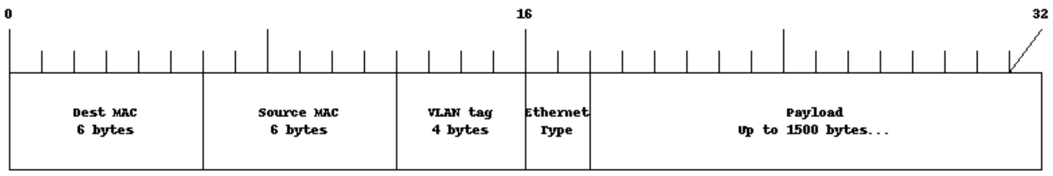


Figure 3.14. An 802.1Q VLAN Ethernet frame



The implementation of pcap capture library is not capable to automatically determine the size difference of the two frame types and it will assume the size of a normal 802.3 Ethernet frame. This means that the offset for the IP header, and thus the higher level protocols on top of IP, for 802.1q frames is incorrect and the filtering of packets on the specified fields will fail. To overcome this, the primitive "vlan" can be used, however it will cause the rest of the filter to assume that the Ethernet frame size is the size of a 802.1q Ethernet frame.

To filter for host 10.0.1.1 for normal 802.3 Ethernet frames: `host 10.0.1.1`. To filter for host 10.0.1.1 for 802.1q Ethernet frames: `(vlan and host 10.0.1.1)`. To filter for host 10.0.1.1 for normal 802.3 and 802.1q Ethernet frames: `host 10.0.1.1 or (vlan and host 10.0.1.1)`.

The default output of tcpdump doesn't show the VLAN tag ID of the 802.1q Ethernet frames, to show then use the option `-le` to display them.

Figure 3.15. Display the first ten packets between host 10.0.1.1 and host 192.168.1.1

```
SH # tcpdump -le -ni lan0_0 -c 10 'host 10.0.1.1 and host 192.168.1.1 or (vlan and host 10 \
    .0.1.1 and host 192.168.1.1)''
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
12:11:36.684544 00:21:70:3e:6b:7f > 00:0d:b9:17:28:de, ethertype 802.1Q (0x8100), length 9 \
    4: vlan 200, p 0, ethertype IPv4, IP 10.0.1.1.60035 > 192.168.1.1.https: S 1139103120: \
    1139103120(0) win 5840 <mss 1460,sackOK,timestamp 1864443320 0,nop,wscale 2>
12:11:36.685159 00:0d:b9:17:28:de > 00:21:70:3e:6b:7f, ethertype 802.1Q (0x8100), length 8 \
    2: vlan 200, p 0, ethertype IPv4, IP 192.168.1.1.https > 10.0.1.1.60035: S 2827780182: \
    2827780182(0) ack 1139103121 win 4380 <mss 1460,nop,wscale 0,nop,nop,timestamp 7927005 \
    29 1864443320,sackOK,eol>
12:11:36.685203 00:21:70:3e:6b:7f > 00:0d:b9:17:28:de, ethertype 802.1Q (0x8100), length 7 \
    0: vlan 200, p 0, ethertype IPv4, IP 10.0.1.1.60035 > 192.168.1.1.https: . ack 1 win 1 \
    460 <nop,nop,timestamp 1864443320 792700529>
12:11:37.025716 00:21:70:3e:6b:7f > 00:0d:b9:17:28:de, ethertype 802.1Q (0x8100), length 1 \
    36: vlan 200, p 0, ethertype IPv4, IP 10.0.1.1.60035 > 192.168.1.1.https: P 1:67(66) a \
    ck 1 win 1460 <nop,nop,timestamp 1864443661 792700529>
12:11:37.026275 00:0d:b9:17:28:de > 00:21:70:3e:6b:7f, ethertype 802.1Q (0x8100), length 1 \
    518: vlan 200, p 0, ethertype IPv4, IP 192.168.1.1.https > 10.0.1.1.60035: . 1:1449(14 \
    48) ack 67 win 4380 <nop,nop,timestamp 792700870 1864443661>
12:11:37.026284 00:0d:b9:17:28:de > 00:21:70:3e:6b:7f, ethertype 802.1Q (0x8100), length 1 \
    518: vlan 200, p 0, ethertype IPv4, IP 192.168.1.1.https > 10.0.1.1.60035: . 1449:2897 \
    (1448) ack 67 win 4380 <nop,nop,timestamp 792700870 1864443661>
12:11:37.026288 00:0d:b9:17:28:de > 00:21:70:3e:6b:7f, ethertype 802.1Q (0x8100), length 1 \
    518: vlan 200, p 0, ethertype IPv4, IP 192.168.1.1.https > 10.0.1.1.60035: P 2897:4345 \
    (1448) ack 67 win 4380 <nop,nop,timestamp 792700870 1864443661>
12:11:37.026314 00:21:70:3e:6b:7f > 00:0d:b9:17:28:de, ethertype 802.1Q (0x8100), length 7 \
    0: vlan 200, p 0, ethertype IPv4, IP 10.0.1.1.60035 > 192.168.1.1.https: . ack 1449 wi \
    n 2184 <nop,nop,timestamp 1864443661 792700870>
12:11:37.026327 00:21:70:3e:6b:7f > 00:0d:b9:17:28:de, ethertype 802.1Q (0x8100), length 7 \
    0: vlan 200, p 0, ethertype IPv4, IP 10.0.1.1.60035 > 192.168.1.1.https: . ack 2897 wi \
    n 2908 <nop,nop,timestamp 1864443661 792700870>
12:11:37.026336 00:21:70:3e:6b:7f > 00:0d:b9:17:28:de, ethertype 802.1Q (0x8100), length 7 \
    0: vlan 200, p 0, ethertype IPv4, IP 10.0.1.1.60035 > 192.168.1.1.https: . ack 4345 wi \
    n 3632 <nop,nop,timestamp 1864443661 792700870>
10 packets captured
19 packets received by filter
0 packets dropped by kernel
```

3.3.5.4. Display all ICMP traffic from and to a certain host

This can be used to determine the forward and backward path the traffic takes by pinging from the client.

Figure 3.16. Capture all traffic for all ICMP traffic from and to host 10.0.1.1

```
SH # tcpdump -ni lan0_0 -c 10 'host 10.0.1.1 and icmp'
tcpdump: WARNING: lan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
10:41:36.717861 IP 10.0.1.1 > 192.168.1.1: ICMP echo request, id 2722, seq 0, length 64
10:41:36.851611 IP 192.168.1.1 > 10.0.1.1: ICMP echo reply, id 2722, seq 0, length 64
10:41:37.714418 IP 10.0.1.1 > 192.168.1.1: ICMP echo request, id 2722, seq 1, length 64
10:41:37.847110 IP 192.168.1.1 > 10.0.1.1: ICMP echo reply, id 2722, seq 1, length 64
10:41:44.120470 IP 10.0.1.1 > 192.168.1.1: ICMP echo request, id 2978, seq 0, length 64
10:41:44.253989 IP 192.168.1.1 > 10.0.1.1: ICMP echo reply, id 2978, seq 0, length 64
10:41:45.121247 IP 10.0.1.1 > 192.168.1.1: ICMP echo request, id 2978, seq 1, length 64
10:41:45.254498 IP 192.168.1.1 > 10.0.1.1: ICMP echo reply, id 2978, seq 1, length 64
10:41:46.122364 IP 10.0.1.1 > 192.168.1.1: ICMP echo request, id 2978, seq 2, length 64
10:41:46.255055 IP 192.168.1.1 > 10.0.1.1: ICMP echo reply, id 2978, seq 2, length 64
10:41:47.123360 IP 10.0.1.1 > 192.168.1.1: ICMP echo request, id 2978, seq 3, length 64
10:41:47.255448 IP 192.168.1.1 > 10.0.1.1: ICMP echo reply, id 2978, seq 3, length 64
```

3.3.5.5. Display the first ten packets from and to certain host on TCP port 443

This can be used to capture the setup of a TCP session.

Figure 3.17. Capture all traffic from and to host 10.0.1.1 on port 443

```
SH # tcpdump -ni lan0_0 -c 10 'host 10.0.1.1 and port 443'
tcpdump: WARNING: lan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
10:42:56.184613 IP 10.0.1.1.57118 > 192.168.1.1.443: Flags [S], seq 1770219666, win 65535, \
     options [mss 1460,nop,wscale 1,nop,nop,TS val 2747475397 ecr 0,sackOK,eol], length 0
10:42:56.317961 IP 192.168.1.1.443 > 10.0.1.1.57118: Flags [S.], seq 2854931292, ack 17702 \
    19667, win 5792, options [mss 1460,sackOK,TS val 92454666 ecr 2747475397,nop,wscale 2] \
    , length 0
10:42:56.320460 IP 10.0.1.1.57118 > 192.168.1.1.443: Flags [.], seq 1, ack 1, win 33304, o \
    ptions [nop,nop,TS val 2747475537 ecr 92454666], length 0
10:42:56.320522 IP 10.0.1.1.57118 > 192.168.1.1.443: Flags [P.], seq 1:149, ack 1, win 333 \
    04, options [nop,nop,TS val 2747475537 ecr 92454666], length 148
10:42:56.454635 IP 192.168.1.1.443 > 10.0.1.1.57118: Flags [.], seq 1, ack 149, win 1448,  \
    options [nop,nop,TS val 92454803 ecr 2747475537], length 0
10:42:56.509572 IP 192.168.1.1.443 > 10.0.1.1.57118: Flags [.], seq 1:1449, ack 149, win 1 \
    448, options [nop,nop,TS val 92454858 ecr 2747475537], length 1448
10:42:56.514621 IP 192.168.1.1.443 > 10.0.1.1.57118: Flags [P.], seq 1449:2564, ack 149, w \
    in 1448, options [nop,nop,TS val 92454858 ecr 2747475537], length 1115
10:42:56.518535 IP 10.0.1.1.57118 > 192.168.1.1.443: Flags [.], seq 149, ack 2564, win 327 \
    46, options [nop,nop,TS val 2747475729 ecr 92454858], length 0
10:42:56.522466 IP 10.0.1.1.57118 > 192.168.1.1.443: Flags [P.], seq 149:347, ack 2564, wi \
    n 33304, options [nop,nop,TS val 2747475732 ecr 92454858], length 198
10:42:56.522477 IP 10.0.1.1.57118 > 192.168.1.1.443: Flags [P.], seq 347:384, ack 2564, wi \
    n 33304, options [nop,nop,TS val 2747475732 ecr 92454858], length 37
```

3.3.5.6. Display all Ethernet broadcast traffic

When the issue seems to be related to a mismatch of the VLAN tag, the best way to determine what goes wrong is to look at the Ethernet broadcast traffic.

Figure 3.18. Tcpdump filter for all Ethernet broadcast traffic

```
SH # tcpdump -le -ni lan0_0 -c 10 'ether broadcast or (vlan and ether broadcast)''
18:53:46.027193 00:0e:b6:42:f8:9c > ff:ff:ff:ff:ff:ff, ethertype 802.1Q (0x8100), length 4 \
    6: vlan 12, p 0, ethertype ARP, Request who-has 10.0.1.9 tell 10.0.1.6, length 28
18:53:46.123323 00:0d:b9:17:28:de > ff:ff:ff:ff:ff:ff, ethertype 802.1Q (0x8100), length 4 \
    6: vlan 3, p 0, ethertype ARP, Request who-has 10.0.1.1 tell 10.0.1.9, length 28
```

In this example, the Ethernet frames of host 10.0.1.6 are on VLAN 12, while the Ethernet frames of host 10.0.1.9 are on VLAN 3.

If no traffic shows up, try to force it by:

- Clearing the ARP table on the Steelhead appliance and on the router.
- From the router, ping an unused IP address on the LAN. This will force the router to keep sending out ARP requests and then, from the Steelhead appliance, ping an unused IP address on the LAN.

3.3.6. Tcpdump related issues

3.3.6.1. TCP Checksum errors

When running tcpdump, packets created by the Steelhead appliance and put on the wire might show up with a TCP checksum mismatch. This is caused by the capability of the network card to generate the TCP header checksum and the IP header checksum.

Figure 3.19. Incorrect TCP checksum messages because of TCP checksum offloading

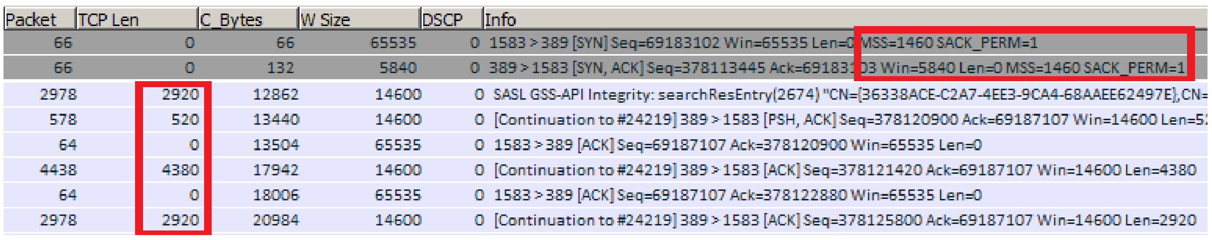
```
SH # tcpdump -lennv -s 1500 -i wan0_0
tcpdump: WARNING: wan0_0: no IPv4 address assigned
listening on wan0_0, link-type EN10MB (Ethernet), capture size 1500 bytes
19:47:09.324317 d4:9a:20:c2:52:0e > 00:0d:b9:17:28:de, ethertype IPv4 (0x0800), length 66: \
     (tos 0x0, ttl 127, id 6410, offset 0, flags [DF], proto TCP (6), length 52)
    10.0.1.1.58616 > 192.168.1.1.445: Flags [S], cksum 0x0e5a (correct), seq 3025755960, w \
    in 8192, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
19:47:09.342483 00:0d:b9:17:28:de > d4:9a:20:c2:52:0e, ethertype IPv4 (0x0800), length 66: \
     (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.1.1.445 > 10.0.1.1.58616: Flags [S.], cksum 0x4695 (incorrect -> 0xdc12), seq  \
    753995387, ack 3025755961, win 5840, options [mss 1460,nop,nop,sackOK,nop,wscale 2], l \
    ength 0
```

The second packet has an incorrect calculated TCP checksum: `cksum 0x4695 (incorrect -> 0xdc12)`. Since the MAC address identifies it as being created by this Steelhead appliance, it is safe to assume that the TCP checksum mismatch is caused by the TCP checksum offloading towards the network card and can be ignored.

3.3.6.2. TCP Segmentation Offloading

TCP Segmentation Offloading is a feature on the NICs which allows the NIC to present multiple TCP packets as one large one:

Figure 3.20. Wireshark showing larger than MSS size TCP packets



Packet	TCP Len	C_Bytes	W Size	DSCP	Info
66	0	66	65535	0	1583 > 389 [SYN] Seq=69183102 Win=65535 Len=0 MSS=1460 SACK_PERM=1
66	0	132	5840	0	389 > 1583 [SYN, ACK] Seq=378113445 Ack=69183103 Win=5840 Len=0 MSS=1460 SACK_PERM=1
2978	2920	12862	14600	0	SASL GSS-API Integrity: searchResEntry(2674) "CN={36338ACE-C2A7-4EE3-9CA4-68AAEE62497E},CN=
578	520	13440	14600	0	[Continuation to #24219] 389 > 1583 [PSH, ACK] Seq=378120900 Ack=69187107 Win=14600 Len=5
64	0	13504	65535	0	1583 > 389 [ACK] Seq=69187107 Ack=378120900 Win=65535 Len=0
4438	4380	17942	14600	0	[Continuation to #24219] 389 > 1583 [ACK] Seq=378121420 Ack=69187107 Win=14600 Len=4380
64	0	18006	65535	0	1583 > 389 [ACK] Seq=69187107 Ack=378122880 Win=65535 Len=0
2978	2920	20984	14600	0	[Continuation to #24219] 389 > 1583 [ACK] Seq=378125800 Ack=69187107 Win=14600 Len=2920

Despite that the MSS size is negotiated to 1460, the captured packets have a TCP length of 2920 and 4380 bytes in size. This is two times and three times the maximum size.

The impact of this is that a capture length of 1600 suddenly doesn't capture the whole packet and analysis by Wireshark and other tools have insufficient data to work with.

To disable this feature, use the command `no interface inpathX_X gso enable`. Use the command `show interface inpathX_X gso` to see the current settings:

Figure 3.21. Output of the command "show interface inpathX_X gso"

```
SH # show interfaces inpath0_0 gso
generic-segmentation-offload: on
TCP-segmentation-offload: on
```

3.3.6.3. WCCP / PBR captures

In a WCCP or PBR environment, traffic which is not supposed to be optimized can still be forwarded to the Steelhead appliance.

The traces of pass-through traffic will show double packets for pass-through traffic: First the incoming packet towards the Steelhead appliance, then the outgoing packet towards the default gateway. On Ethernet level the payload is the same, the only difference is the MAC addresses.

However, captures taken of this pass-through traffic will show that the MAC addresses of the two packets are the same: This is because the way the data is handled in the network buffers. Wireshark will show these packets as duplicate packets. This is a dissector issue for this redirected traffic, not an issue with the traffic.

3.4. Tcpdump-x

Tcpdump-x is a wrapper around tcpdump. It has the advantage over the normal *tcpdump* command that the parameters are in a friendlier format and that it can capture on multiple interfaces at the same time.

It knows the following parameters:

- **all-interfaces**: Capture on all interfaces
- **interfaces <interface list>** Capture on the specified interfaces, comma separated.
- **capture-name <identifier>**: Use the specified *identifier* to identify the files stored when the capture is finished.
- **snaplength <length>**: Capture the first *length* bytes per frame. Use 1600 for a full packet.
- **custom <pcap filter>**: Use this pcap filter to limit the packets captured. It is fed to tcpdump so all the primitives available to tcpdump are possible here.
- **continuous**: Do not automatically terminate the capture after a certain time period
- **duration <number of seconds>**: Run the capture for *number of seconds* seconds.
- **file-size <size>**: Rotate the capture file when its size is *size* megabytes.
- **rotate-count <number>**: Store at maximum *number* capture files per interface.
- **stop <capture-name>**: Stop the capture specified by the previous *capture-name*.

In the following example, the first command will capture all packets on the lan0_0 and wan0_0 interfaces between host 10.0.1.1 and host 192.168.1.1. The second command captures all packets on the lan0_0 and wan0_0 interfaces for 30 seconds, maximum file size is 10 Mb and the maximum number of files is 12. The third command shows all running tcpdump-x captures. The fourth command terminates the first command and the fifth command shows all the tcpdump captures.

Figure 3.22. Examples for using tcpdump-x

```
SH # tcpdump-x interfaces lan0_0,wan0_0 capture-name test1 snaplength 1600 custom 'host 10 \
    .0.1.1 and host 192.168.1.1' continuous
SH # tcpdump-x interfaces lan0_0,wan0_0 capture-name test2 snaplength 1600 rotate-count 12 \
     duration 30 file-size 10
SH # show tcpdump-x
Name: test1
Start Time: 21:41:14
SH # tcpdump-x capture-name test1 stop
SH # show files tcpdump
SH_lan0_0_test1.cap0
SH_wan0_0_test1.cap0
SH_lan0_0_test2.cap0
SH_wan0_0_test2.cap0
```

Note that order of the options can be specific, so always use the *?* to see which options are available!

3.5. Ping

Ping is the most well-known command in the networking world to determine if a remote host is reachable. It works by sending *ICMP Echo Request* packets and displays any ICMP packet returned.

Besides the *ping* command, used for the IPv4 transport layer, there is also the *ping6* command, used for the IPv6 transport layer. The options used for the IPv4 version can also be used for the IPv6 version unless noted differently.

3.5.1. Ping options

The following options are available:

- The option `-c #` limits the number of ICMP packets send out.
- The option `-I <IP address>` specifies the source IPv4 address to use and thus the outgoing interface. To specify the IPv6 source address, use the `-I <interface>` option, the IPv6 address on the specified interface will be used there.
- The option `-s #` specifies the size of the ICMP packet payload.
- The option `-M do` disables IP packet fragmentation due to MTU size issues.

3.5.2. Ping examples

On the Steelhead appliance, the ping command uses by default the IP address of the primary interface as the source IP address of the packets:

Figure 3.23. Ping a host via the primary interface

```
SH # ping -c 3 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=0 ttl=59 time=290.0 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=59 time=260.0 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=59 time=260.0 ms

--- 192.168.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 260.0/280.0/290.0/10.000 ms, pipe 2
```

The option `-c 3` limits the number of ICMP packets send to three, otherwise it will keep sending *ICMP Echo Request* packets until the ping command is aborted via the control-C key combination. The next option is the IP address or the hostname of the destination host.

The output shows the IP address of the device which answered the *ICMP Echo Request* command, the sequence number of the *ICMP Echo Reply* packet received back, the TTL of the IP packet which can be used to identify the number of hops to the destination host is and the time it took for the answer to come back.

The `-I` option can be used to specify the IP address of an in-path interface to force the ICMP packet out via the corresponding LAN or WAN interfaces. It will follow the routing table of that interface.

Figure 3.24. Ping a host via the in-path interface

```
SH # ping -c 3 -I 10.0.1.6 192.168.1.1
PING 192.168.1.1 (192.168.1.1) from 10.0.1.6 : 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=0 ttl=59 time=290 ms
64 bytes from 192.168.1.1: icmp_seq=0 ttl=59 time=260 ms
64 bytes from 192.168.1.1: icmp_seq=0 ttl=59 time=260 ms

--- 192.168.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 260.0/290.0/280.0/10.000 ms, pipe 2
```

In this example, 10.0.1.6 was the in-path interface IP address.

3.5.3. Detecting failure scenarios

3.5.3.1. No answer from a host on the local subnet

When pinging a host on a local subnet and you get an *ICMP Destination Host Unreachable* message, it means that the ARP request for the destination host didn't get an answered. Most of the time it means that the host with that IP address is turned off.

Figure 3.25. IP address 10.0.1.4 is not in use on this local IP subnet

```
SH # ping -c 3 10.0.1.4
PING 10.0.1.4 (10.0.1.4) 56(84) bytes of data.
From 10.0.1.5 icmp_seq=0 Destination Host Unreachable
From 10.0.1.5 icmp_seq=0 Destination Host Unreachable
--- 10.0.1.4 ping statistics ---
3 packets transmitted, 0 received, +2 errors, 100% packet loss, time 0ms , pipe 2
```

Note that the *ICMP Destination Host Unreachable* message occurred because the ARP request for the IP address did not get answered. If the host was firewalled and would drop the ICMP traffic, the ARP request would have been answered so instead of the *ICMP Destination Host Unreachable* message there would have been no output at all.

3.5.3.2. No answer at all from a remote host

For a host on a remote subnet, no answers can mean one of the following issues:

- The IP address is not in use and the router for that subnet is not sending *ICMP Destination Host Unreachable* packets back.
- The destination host has been configured to not respond to *ICMP Echo Request packets.*
- The destination IP subnet is unreachable from the sending host.
- The source IP subnet is unreachable from the remote host.
- A firewall in the network has blocked either the *ICMP Echo Request* or the *ICMP Echo Reply* packet.

Figure 3.26. IP address 192.168.1.1 didn't reply.

```
SH # ping -c 3 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.

--- 192.168.1.1 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 5000ms, pipe 2
```

3.5.3.3. Serious packet loss

In every *ICMP Echo Request* message sent, the *icmp_seq* field contains the sequence number of the request. The returned *ICMP Echo Response* will contain the same sequence number and it can be used to determine which packets were lost:

Figure 3.27. ICMP packets 3 to 5 and 8 to 9 didn't get answered.

```
SH # ping -c 10 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=0 ttl=59 time=260.0 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=59 time=260.0 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=59 time=260.0 ms
64 bytes from 192.168.1.1: icmp_seq=6 ttl=59 time=260.0 ms
64 bytes from 192.168.1.1: icmp_seq=7 ttl=59 time=260.0 ms

--- 192.168.1.1 ping statistics ---
10 packets transmitted, 5 received, 50% packet loss, time 0ms
rtt min/avg/max/mdev = 260.0/260.0/260.0/0.000 ms, pipe 2
```

3.5.4. Path MTU detection

When an IP packet arrives at a router and the outgoing link has a maximum packet size which is smaller than the IP packet, it will break the IP packet in several smaller IP packets and send that over the wire. To prevent this breaking up from happening, the sender can set the *Don't Fragment* flag in the IP header. As a result, the router will not be able to forward the packet and will send an *ICMP Fragmentation Needed* packet back to the sender. This *ICMP Fragmentation Needed* packet contains the MTU size of the link. The sender should then retransmit the IP packet with a smaller payload.

The IP packets of the inner channel of an optimized TCP session have that *Don't Fragment* flag set. Therefore, routers on a link with a smaller MTU size than what is configured on the Steelhead in-path interface should send back an *ICMP Fragmentation Needed* packet. If the network for whatever reason doesn't deliver this to the Steelhead appliance, the packet will not be delivered, the inner channel will time-out and the optimized TCP session will break.

The size parameter on the ping command can be used to determine the maximum MTU size off the network.

The initial size value to check should be the MTU size of the interface on the Steelhead appliance minus 28 bytes. The 28 bytes are the IP header and the *ICMP Echo Request* header.

Figure 3.28. ICMP based MTU size detection - MTU of 1500 bytes

```
SH # ping -s 1472 -M do 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 1472(1500) bytes of data.
556 bytes from 10.0.1.9 (10.0.1.9): frag needed and DF set (MTU 1492)
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src         Dst
 4  5  00 d505 0058   0 0000  40  01 b4e1 10.0.1.5 192.168.1.1
```

This *ICMP Fragmentation Needed* packet shows the following information:

- The IP header contains 10.0.1.9, the IP address of the router which could not forward the IP packet.
- The ICMP payload contains 0058, the IP packet identifier of the IP packet which could not be forwarded.
- The ICMP payload contains 10.0.1.5 and 192.168.1.1, the source and destination IP address of the IP packet which was too big.
- 1492, the MTU size of the outgoing link.

If no *ICMP Fragment Needed* packet or *ICMP Echo Reply* is returned, that is an indication that the network is dropping the reply packet.

The next steps is to reduce the size value to a value until you receive *ICMP Echo Replies* instead of *ICMP Fragmentation Needed* packets. In this case where it is now known that the MTU size of the first link is 1492, the next value to try should be 1492 minus 28 is 1464.

Figure 3.29. ICMP based MTU size detection - MTU of 1492 bytes

```
SH # ping -s 1464 -M do 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 1464(1492) bytes of data.
1472 bytes from 192.168.1.1: icmp_seq=0 ttl=59 time=260.0 ms
```

Setting the MTU size of the in-path interface to 1492 will overcome the communication problem towards this remote Steelhead appliance.

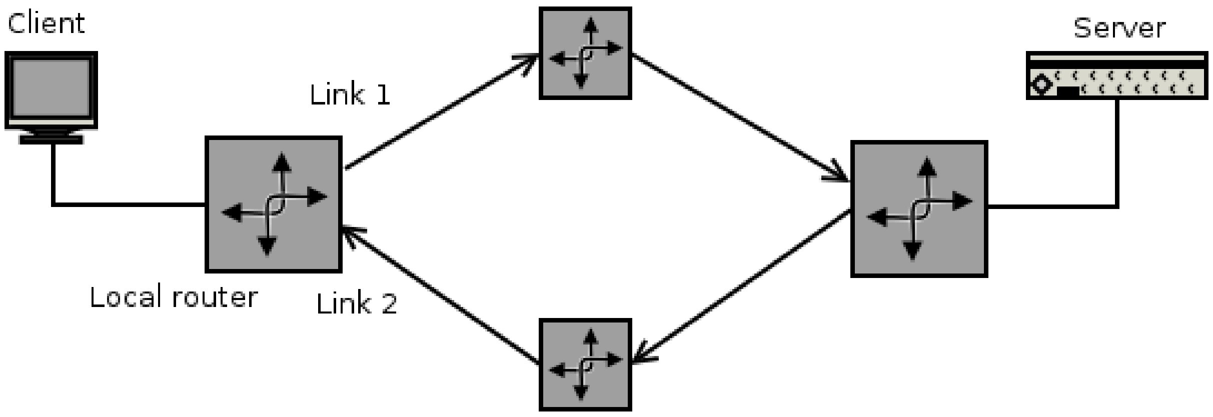
3.6. Traceroute

Traceroute is a basic command in the networking world to determine the path the traffic takes to a remote host. It sends a number of UDP packets with different source ports and with increasing TTL value. The source address of the returned *ICMP TTL Exceeded* replies will show the IP addresses of the path.

Besides the *traceroute* command, used for the IPv4 transport layer, there is also the *traceroute6* command, used for the IPv6 transport layer. The options used for the IPv4 version can also be used for the IPv6 version unless noted differently.

With the returned ICMP packets, the IP addresses on it are often the IP address of the outgoing interface on that router which might not be the IP address of the incoming interface of that router. So to confirm the path, the traceroute program should to be run on both the client and the server.

Figure 3.30. WAN router receives packet via a different path than it send it



3.6.1. Traceroute options

The following options are available for the traceroute command:

- The option `-n` prevents name-resolution of the IP addresses displayed.
- The option `-s <IP address>` changes the source IPv4 or IPv6 address to use for and thus the outgoing interface.
- The option `-S #` changes the size of the payload of the outgoing UDP packet.
- The option `--mtu` can be used to identify the maximum MTU size towards the remote host.

3.6.2. Traceroute examples

The traceroute command uses by default the IP address of the primary interface:

Figure 3.31. The command to traceroute to a remote host via the primary interface

```
SH # traceroute -n 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 172.16.2.2  90.657 ms  90.245 ms  90.342 ms
 4 172.16.3.2  130.657 ms  130.245 ms  130.342 ms
 5 192.168.1.1  130.857 ms  130.545 ms  130.642 ms
```

The `-s` option can be used to specify the IP address of an in-path interface:

Figure 3.32. Traceroute to a remote host via an in-path interface

```
SH # traceroute -n -s 10.0.1.5 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1) from 10.0.1.5, 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 172.16.2.2  90.657 ms  90.245 ms  90.342 ms
 4 172.16.3.2  130.657 ms  130.245 ms  130.342 ms
 5 192.168.1.1  130.857 ms  130.545 ms  130.642 ms
```

3.6.3. MTU Path Detection with traceroute

The ping command can be used to determine that there are MTU size issues in a network, the traceroute command can be used to determine where it is happening.

If the network is properly informing with *ICMP Fragmentation Needed* packets, the `--mtu` option can be used to see it:

Figure 3.33. Determining the MTU size with the --mtu option

```
CSH # traceroute -n --mtu 192.168.1.1
traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 65000 byte packets
 1 10.0.1.9  0.657 ms F=1422  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 172.16.2.2  90.657 ms  90.245 ms  90.342 ms
 4 172.16.3.2  130.657 ms  130.245 ms  130.342 ms
 5 192.168.1.1  130.857 ms  130.545 ms  130.642 ms
```

The detected maximum MTU size here is 1422 bytes.

If the network is not sending the *ICMP Fragmentation Needed* packets, the same method by increasing and decreasing the packet size as used with the *ping* command should be used.

First the path between towards the remote host needs to be determined:

Figure 3.34. Successful traceroute between two hosts

```
SH # traceroute -n 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 172.16.2.2  90.657 ms  90.245 ms  90.342 ms
 4 172.16.3.2  130.657 ms  130.245 ms  130.342 ms
 5 192.168.1.1  130.857 ms  130.545 ms  130.642 ms
```

Now the same trace but with a packet size larger than the MTU size of the link:

Figure 3.35. Failed traceroute due to MTU size issues

```
SH # traceroute -n -S 1480 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 1480 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 * * *
 4 * * *
 5 * * *
```

The router with the IP address of 172.16.1.2 did forward the packet, but the router with the IP address of 172.16.2.2 did not return any answers. Use a smaller option for the size to determine the right maximum MTU size.

3.6.4. Determine failure scenarios

3.6.4.1. The trace doesn't show data for certain hops.

In this example the traceroute command returned no data for hop 3:

Figure 3.36. Missing reply for hop three.

```
SH # traceroute -n 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 * * *
 4 172.16.3.2  130.657 ms  130.245 ms  130.342 ms
 5 192.168.1.1  130.857 ms  130.545 ms  130.642 ms
```

This could happen when for example:

- A firewall doesn't allow ICMP TTL Expired messages from the IP addresses on hop 3.
- The gateway in hop 3 is instructed not to send out ICMP TTL Expired messages.

As the end-to-end communication works, the missing hop may be ignored.

3.6.4.2. The trace doesn't show data after a certain hop.

The traceroute command could have returned only *'s after a certain hop:

Figure 3.37. Missing reply for hop three and following.

```
SH # traceroute -n 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 * * *
 4 * * *
 5 * * *
```

This can happen when:

- The network after hop 2 doesn't know the way back to the source IP subnet of the trace of 10.0.1.0/24.
- The network after hop 2 doesn't know the way to 192.168.1.1 but doesn't send back ICMP Network Unreachable. If it did, the output would have ended with:

  ### Figure 3.38. Network Unreachable response

  ```
   3  192.168.1.1  3000.123 ms !N  3000.231 ms !N  3000.349 ms !N
  ```

- A firewall between hop 2 and 3 is blocking the traffic.

3.6.4.3. The trace doesn't return anything for the last hop.

The traceroute command could have returned *'s for the latest hop:

Figure 3.39. Missing reply for the latest hop.

```
SH # traceroute -n 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 172.16.2.2  90.657 ms  90.245 ms  90.342 ms
 4 172.16.3.2  130.657 ms  130.245 ms  130.342 ms
 5 * * *
```

This could happen when for example:

- When the firewall on the remote host is configured not to send ICMP Port Unreachable.
- When the destination IP address is not configured on that IP subnet and the final router did not send back ICMP Host Unreachable. If it did, the output would have ended with:

Figure 3.40. Host Unreachable response

```
5  192.168.1.1  3000.123 ms  !H 3000.231 ms !H  3000.349 ms !H
```

3.6.4.4. The trace did contain multiple IP addresses for a hop

This happens when a hop is doing load balancing over multiple links. In that case the outgoing interface is pretty much random.

Figure 3.41. Hop 4 has multiple outgoing interfaces

```
SH # traceroute -n 192.168.1.1
Traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 38 byte packets
 1 10.0.1.9  0.657 ms  0.245 ms  0.342 ms
 2 172.16.1.2  30.657 ms  30.245 ms  30.342 ms
 3 172.16.2.2  90.657 ms  90.245 ms  90.342 ms
 4 172.16.3.2  130.657 ms
   172.16.3.10  130.245 ms
   172.16.3.2  130.342 ms
 5 192.168.1.1  130.857 ms  130.545 ms  130.642 ms
```

3.7. Telnet

Telnet is a basic command in the networking world to setup a TCP session to a service on a remote host. It completes the TCP handshake (SYN, SYN/ACK, ACK) so it can be used to emulate the setup of an TCP session expected to be optimized.

Once telnet has successfully setup its TCP session, the only way to get out of it is to either let the server terminate the TCP session (protocol timeout, protocol error, issuing the command to terminate the TCP session) or to press the `control-]` and to type `quit` command in the `telnet>` prompt.

Using the telnet client available under Microsoft Windows, the screen will be cleared when the TCP session is setup.

With the introduction of Microsoft Windows 7, the telnet client is not installed by default anymore. You can install it via Control Panel -> Programs and Features -> Enable Features and install the Telnet Client from there.

The reason the command-line telnet application should be used and not an integrated telnet client like in PuTTY or in SecureCRT is that they are very time-wasting to setup and hide information from you. For example you will have to enter the connection details in a dialog box, you will get dialog box on the screen which shows that a failure happened and you will not see that a TCP session has setup successfully until you get text send back to you. The telnet application on the command-line has an easy way to enter the session details, it will be clear when the TCP session has been setup, has an easy way to terminate the session and is clear when the TCP session has been terminated.

The telnet command available on the Steelhead appliance supports both IPv4 and IPv6 destinations, but it is not possible to explicitly choose for an IPv4 or IPv6 address when using a hostname to connect to.

3.7.1. Telnet options

The following options are available for the telnet command:

- The option `-b <IP address>` binds the source IPv4 or IPv6 address of TCP session to the destination IP address.

3.7.2. Telnet examples

The telnet command uses by default the IP address of the primary interface:

Figure 3.42. Successful TCP session setup via the primary interface

```
SH # telnet 192.168.1.1 139
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'
```

In this case the TCP session is setup between the hosts 10.0.1.5 and 192.168.1.1 on destination TCP port 139.

The `-b` option can be used to specify the IP address of the in-path interface.

Figure 3.43. Successful TCP session setup via the in-path interface

```
SH # telnet -b 10.0.1.6 192.168.1.1 139
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'
```

In this case the TCP session is setup between the hosts 10.0.1.6 and 192.168.1.1.

3.7.3. Telnet failure scenarios

3.7.3.1. No route to host

A *No route to host* error is shown when the remote host does not exist and the router on that subnet sends back an ICMP Host Unreachable message or when the a router in the network doesn't know where to forward the IP packet to and sends an ICMP Network Unreachable message to the client.

Figure 3.44. Telnet attempt to a host which does not exist

```
SH # telnet 192.168.1.1 139
Trying 192.168.1.1...
telnet: connect to address 192.168.1.1: No route to host
```

3.7.3.2. Connection timed out

This happens when no TCP SYN/ACK is received on the three TCP SYN packets send out by the client.

Figure 3.45. Telnet attempt to a host which does not exist

```
SH # telnet 192.168.1.1 139
Trying 192.168.1.1...
telnet: connect to address 192.168.1.1: Connection timed out
```

3.7.3.3. Connection refused

This happens when the TCP SYN packet is delivered to the remote host but when there is no service listening on that TCP port.

Figure 3.46. Telnet attempt to a host with no service listening on port 139

```
SH # telnet 192.168.1.1 139
Trying 192.168.1.1...
telnet: connect to address 192.168.1.1: Connection refused
```

3.8. Tproxytrace

Tproxytrace is a RiOS specific command to probe for Steelhead appliances in the path to a remote server. It sends a SYN+ packet, and any Steelhead appliance in the path that answers the auto-discovery probe will have their information displayed.

The tproxytrace command does not support IPv6 destinations.

3.8.1. Tproxytrace options

The following options are available for the tproxytrace command:

- The option `-i <interface-name>` specifies the interface which outgoing packets will use and thus which IP address will be used for them.

3.8.2. Tproxytrace examples

The tproxytrace command does not use the IP address of the primary interface by default. Use the `-i` option to specify the outgoing interface:

Figure 3.47. Tproxytrace to a host via the primary interface

```
SH # tproxytrace -i primary 192.168.1.1:139
Probe from 10.0.1.5 (primary) to 192.168.1.1:139
depth 1 proxy 10.0.1.6:7800
depth 2 proxy 192.168.1.1:7800
```

Figure 3.48. Tproxytrace to a host via the in-path interface

```
SH # tproxytrace -i inpath0_0 192.168.1.1:139
Probe from 10.0.1.6 (inpath0_0) to 192.168.1.1:139
depth 1 proxy 192.168.1.1:7800
```

3.8.3. Tproxytrace failure scenarios

3.8.3.1. The tproxytrace output doesn't return expected Steelhead appliances

There could be various reasons, including:

- The path the SYN+ takes doesn't have a Steelhead appliance in it.
- The Steelhead appliances in the path are configured to not peer with the source specified.
- The Steelhead appliances in the path cannot send the answer back to the source of the SYN+.
- The auto-discovery TCP option number used in the SYN+ is different than the one used on the Steelhead appliances in the network.
- The expected Steelhead appliances in the network are in admission control.

3.9. Nettest

The nettest command can be used to perform some of the tests described earlier automatic.

3.9.1. Peer reachability

The Peer Reachability test tries to set up a TCP session to port 7801 to a remote Steelhead appliance. It attempts to set them up from all configured IP addresses:

This test only supports IPv4 addresses.

Figure 3.49. Output of a peer reachability test

```
CSH (config) # nettest run peer-reach addr 192.168.1.6
Peer Reachability Test        Last Run: 2013/11/25 10:54:01
Passed

Address            Interface          Result
==================================================================
192.168.1.6        primary            Passed
192.168.1.6        aux                Passed
192.168.1.6        inpath0_0          Passed
```

The captures on the various interfaces look like this:

Figure 3.50. Capture of a peer reachability test

```
CSH # tcpdump -ni primary host 10.0.1.5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on primary, link-type EN10MB (Ethernet), capture size 300 bytes
11:12:09.331690 IP 10.0.1.5.65535 > 192.168.1.6.7801: Flags [S], seq 169835245, win 5840,  \
    options [rvbd-probe unkn-ver-type:1/3 31010a0001050005,nop,eol], length 0
11:12:09.634781 IP 192.168.1.6.7801 > 10.0.1.5.65535: Flags [S.], seq 20020520, ack 169835 \
    246, win 5840, options [rvbd-probe AD CSH:10.0.1.5 SSH:192.168.1.6:7800 11110a000105c0 \
    a801061e78,rvbd-probe EAD 0e3c,nop,eol], length 0
11:12:09.634888 IP 10.0.1.5.65535 > 192.168.1.6.7801: Flags [R], seq 169835246, win 0, len \
    gth 0
11:12:09.635405 IP 10.0.1.5.65535 > 192.168.1.6.7801: Flags [S], seq 169835245, win 5840,  \
    options [rvbd-probe unkn-ver-type:1/3 31020a0001050005,nop,eol], length 0
11:12:09.936657 IP 192.168.1.6.7801 > 10.0.1.5.65535: Flags [S.], seq 1081397572, ack 1698 \
    35246, win 5840, options [mss 1460], length 0
11:12:09.936697 IP 10.0.1.5.65535 > 192.168.1.6.7801: Flags [R], seq 169835246, win 0, len \
    gth 0
```

As can be seen, the client-side Steelhead appliance sends a SYN+, the server-side Steelhead appliance replies with a SYN/ACK+. Finally the client-side Steelhead appliance sends a TCP RST and the test is tried again.

3.9.2. Net gateway test

The Net Gateway test sends three pings the IP addresses of the local network gateways. It takes the gateways of the base interfaces routing table and the in-path interfaces routing table and tries to ping them.

In the examples below an extra destination was added to the two routing tables before the test was ran:

Figure 3.51. Output of a net gateway test

```
CSH (config) # show ip route
Destination       Mask              Gateway           Interface
1.2.3.4           255.255.255.255   10.0.1.67         primary
10.0.1.0          255.255.255.0     0.0.0.0           primary
default           0.0.0.0           10.0.1.9          primary

CSH (config) # show ip in-path route inpath0_0
Destination       Mask              Gateway
1.2.3.4           255.255.255.255   10.0.1.66
10.0.1.0          255.255.255.0     0.0.0.0
default           0.0.0.0           10.0.1.9

CSH (config) # nettest run net-gateway
Gateway Test                  Last Run: 2013/11/25 11:22:01
Passed

Interface          Address            Packet Loss        Result
==================================================================
Default            10.0.1.9           0%                 Passed
inpath0_0          10.0.1.66          100%               Failed
inpath0_0          10.0.1.9           0%                 Passed
Static             10.0.1.67          100%               Failed

CSH (config) # nettest run net-gateway ipv6
Gateway Test                  Last Run: 2013/06/03 17:18:35
Passed

Interface          Address            Packet Loss        Result
==================================================================
Default            2600:809:200:4ff:20e:b6ff:fe01:6070
                                      0%                 Passed
```

As expected, the ones to 10.0.1.66 and 10.0.1.67 didn't get answered.

3.9.3. Duplex test

The Duplex test works by sending a large amount of ICMP ping tests over an interface. In case of a speed or duplex mismatch, there will be packet loss on it.

Figure 3.52. Output of a duplex test

```
CSH (config) # nettest run duplex primary target 10.0.1.9
Duplex Test                   Last Run: 2013/11/25 11:42:16
Passed

Interface          Number of Errors   Result
==================================================================
primary            0                  Passed
```

CSH (config) # nettest run duplex primary ipv6-target fd57:1083::3 Duplex Test Last Run: 2013/11/25 11:42:16 Passed

Interface Number of Errors Result == primary 0 Passed

3.9.4. IP Port reachability test

The IP Port Reachability test sets up a TCP session from one of the base interfaces to the specified host.

Figure 3.53. Output of the IP Port reachability test

```
CSH (config) # nettest run ip-port-reach source primary addr 192.168.1.1 port 22
IP/Port Reachability Test     Last Run: 2013/11/25 11:54:17
Passed

Interface          Address            Protocol           Result
==================================================================
aux                192.168.1.1:22     Netcat             Passed

CSH (config) # nettest run ip-port-reach source primary ipv6-addr fd57:1083::3 port 22
IP/Port Reachability Test     Last Run: 2013/11/25 11:54:17
Passed

Interface          Address            Protocol           Result
==================================================================
aux                [fd57:1083::3]:22  Netcat             Passed
```

On the wire it looks like a normal TCP session, prefixed by some DNS resolution:

Figure 3.54. Capture of the IP Port reachability test

```
CSH # tcpdump -ni primary host 10.0.1.5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on primary, link-type EN10MB (Ethernet), capture size 300 bytes
11:57:42.526526 IP 10.0.1.5.32825 > 192.168.1.1.53: 18418+ PTR? 5.1.0.10.in-addr.arpa. (39 \
    )
11:57:42.827737 IP 192.168.1.1.53 > 10.0.1.5.32825: 18418 NXDomain* 0/1/0 (91)
11:57:42.828122 IP 10.0.1.5.32825 > 192.168.1.1.53: 11624+ PTR? 1.1.168.192.in-addr.arpa.  \
    (42)
11:57:43.127581 IP 192.168.1.1.53 > 10.0.1.5.32825: 11624 NXDomain* 0/1/0 (94)
11:57:43.128040 IP 10.0.1.5.33329 > 192.168.1.1.80: Flags [S], seq 3966492797, win 5840, o \
    ptions [mss 1460,sackOK,TS val 5332034 ecr 0,nop,wscale 2], length 0
11:57:43.427441 IP 192.168.1.1.80 > 10.0.1.5.33329: Flags [S.], seq 3407753711, ack 396649 \
    2798, win 65535, options [mss 1460,nop,wscale 6,sackOK,TS val 3117926755 ecr 5332034], \
     length 0
11:57:43.427490 IP 10.0.1.5.33329 > 192.168.1.1.80: Flags [.], seq 1, ack 1, win 1460, opt \
    ions [nop,nop,TS val 5332333 ecr 3117926755], length 0
11:57:43.427799 IP 10.0.1.5.33329 > 192.168.1.1.80: Flags [F.], seq 1, ack 1, win 1460, op \
    tions [nop,nop,TS val 5332334 ecr 3117926755], length 0
11:57:43.729324 IP 192.168.1.1.80 > 10.0.1.5.33329: Flags [.], seq 1, ack 2, win 1040, opt \
    ions [nop,nop,TS val 3117927055 ecr 5332334], length 0
11:57:43.731237 IP 192.168.1.1.80 > 10.0.1.5.33329: Flags [F.], seq 1, ack 2, win 1040, op \
    tions [nop,nop,TS val 3117927055 ecr 5332334], length 0
11:57:43.731262 IP 10.0.1.5.33329 > 192.168.1.1.80: Flags [.], seq 2, ack 2, win 1460, opt \
    ions [nop,nop,TS val 5332637 ecr 3117927055], length 0
```

3.10. SSL connect

Tunneling a protocol over SSL is a simple feast these days. However, troubleshooting it is a difficult task: An extra layer is added and instead of a plain text protocol you are suddenly stuck with a binary protocol.

Since RiOS 8.0, the command `ssl-connect` provides a method to setup an SSL encrypted TCP session towards a server. It uses the same certificate store as the Steelhead appliance, so it can be used to check if all intermediate certificates are available and valid.

The ssl-connect command supports IPv4 and IPv6 but doesn't recognize the IPv6 address as such, a hostname which resolves into an IPv6 address is required.

Figure 3.55. Usage of the ssl-connect command

```
CSH # ssl-connect 192.168.1.1:443
CONNECTED(00000003)
depth=2 C = US, O = "The Go Daddy Group, Inc.", OU = Go Daddy Class 2 Certification Author \
    ity
verify return:1
depth=1 C = US, ST = Arizona, L = Scottsdale, O = "GoDaddy.com, Inc.", OU = http://certifi \
    cates.godaddy.com/repository, CN = Go Daddy Secure Certification Authority, serialNumb \
    er = 07969287
verify return:1
depth=0 O = *.mavetju.org, OU = Domain Control Validated, CN = *.mavetju.org
verify return:1
---
Certificate chain
 0 s:/O=*.mavetju.org/OU=Domain Control Validated/CN=*.mavetju.org
   i:/C=US/ST=Arizona/L=Scottsdale/O=GoDaddy.com, Inc./OU=http://certificates.godaddy.com/ \
    repository/CN=Go Daddy Secure Certification Authority/serialNumber=07969287
 1 s:/C=US/ST=Arizona/L=Scottsdale/O=GoDaddy.com, Inc./OU=http://certificates.godaddy.com/ \
    repository/CN=Go Daddy Secure Certification Authority/serialNumber=07969287
   i:/C=US/O=The Go Daddy Group, Inc./OU=Go Daddy Class 2 Certification Authority
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIFUzCCBDugAwIBAgIHKyoFsgHhjjANBgkqhkiG9w0BAQUFADCByjELMAkGA1UE
BhMCVVMxEDAOBgNVBAgTB0FyaXpvbmExEzARBgNVBAcTClNjb3R0c2RhbGUxGjAY
[...]
8gVExh5WcDWWTRAB3IgV+puWx6rFblZ2WTjHKqfvvpfolaCy8+xMVwQI7BJyA6vF
SgMsfQ42zKMwIiHEHM8jdI5AMVM5VCQ=
-----END CERTIFICATE-----
subject=/O=*.mavetju.org/OU=Domain Control Validated/CN=*.mavetju.org
issuer=/C=US/ST=Arizona/L=Scottsdale/O=GoDaddy.com, Inc./OU=http://certificates.godaddy.co \
    m/repository/CN=Go Daddy Secure Certification Authority/serialNumber=07969287
---
No client certificate CA names sent
---
SSL handshake has read 2782 bytes and written 439 bytes
---
New, TLSv1/SSLv3, Cipher is AES256-SHA
Server public key is 2048 bit
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1
    Cipher    : AES256-SHA
    Session-ID: 1809B7245E7E42B9F2CAB82B54019B3E20F3923C16A2927A4D3996FAC4528CB5
    Session-ID-ctx:
    Master-Key: B3DCF9008DBB76B7F0151B723DFB8AFF530310720985BC6739E9E60E7EE64EB8F81B88368A \
    9EA8131D4DE3FBEB77BEF5
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    Start Time: 1359015123
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
---
GET / HTTP/1.0
Host: www.mavetju.org

HTTP/1.1 200 OK
Date: Thu, 24 Jan 2013 08:12:29 GMT
Server: Apache/2.2.3 (FreeBSD)
Content-Length: 239
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1
X-RBT-Optimized-By: CSH (RiOS 8.0.1) IK

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
[...]
</body></html>
closed
```

The result code under *Verify return code*: is how the ssl-client command interprets the certificates. This is a list of possible result codes, obtained of the man page of the OpenSSL verify(1) utility.

- 0 / Ok: The operation was successful.

- 2 / Unable to get issuer certificate: The issuer certificate of a looked up certificate could not be found. This normally means the list of trusted certificates is not complete.
- 4 / Unable to decrypt certificate's signature: The certificate signature could not be decrypted. This means that the actual signature value could not be determined rather than it not matching the expected value, this is only meaningful for RSA keys.
- 6 / Unable to decode issuer public key: The public key in the certificate SubjectPublicKeyInfo could not be read.
- 7 / Certificate signature failure: The signature of the certificate is invalid.
- 9 / Certificate is not yet valid: The certificate is not yet valid: the notBefore date is after the current time.
- 10 / Certificate has expired: The certificate has expired: that is the notAfter date is before the current time.
- 13 / Format error in certificate's notBefore field: The certificate notBefore field contains an invalid time.
- 14 / Format error in certificate's notAfter field: The certificate notAfter field contains an invalid time.
- 17 / Out of memory: An error occurred trying to allocate memory. This should never happen.
- 18 / Self signed certificate: The passed certificate is self-signed and the same certificate cannot be found in the list of trusted certificates.
- 19 / Self signed certificate in certificate chain: The certificate chain could be built up using the untrusted certificates but the root could not be found locally.
- 20 / Unable to get local issuer certificate: The issuer certificate could not be found: this occurs if the issuer certificate of an untrusted certificate cannot be found.
- 21 / Unable to verify the first certificate: No signatures could be verified because the chain contains only one certificate and it is not self-signed.
- 24 / Invalid CA certificate: A CA certificate is invalid. Either it is not a CA or its extensions are not consistent with the supplied purpose.
- 25 / Path length constraint exceeded: The basicConstraints pathlength parameter has been exceeded.
- 26 / Unsupported certificate purpose: The supplied certificate cannot be used for the specified purpose.
- 27 / Certificate not trusted: The root CA is not marked as trusted for the specified purpose.
- 28 / Certificate rejected: The root CA is marked to reject the specified purpose.
- 29 / Subject issuer mismatch: The current candidate issuer certificate was rejected because its subject name did not match the issuer name of the current certificate.
- 30 / Authority and subject key identifier mismatch: The current candidate issuer certificate was rejected because its subject key identifier was present and did not match the authority key identifier current certificate.
- 31 / Authority and issuer serial number mismatch: The current candidate issuer certificate was rejected because its issuer name and serial number was present and did not match the authority key identifier of the current certificate.
- 32 / Key usage does not include certificate signing: The current candidate issuer certificate was rejected because its keyUsage extension does not permit certificate signing.

- 24 / Invalid CA certificate: A CA certificate is invalid. Either it is not a CA or its extensions are not consistent with the supplied purpose.
- 25 / Path length constraint exceeded: The basicConstraints pathlength parameter has been exceeded.
- 26 / Unsupported certificate purpose: The supplied certificate cannot be used for the specified purpose.

- 27 / Certificate not trusted: The root CA is not marked as trusted for the specified purpose.
- 28 / Certificate rejected: The root CA is marked to reject the specified purpose.
- 29 / Subject issuer mismatch: The current candidate issuer certificate was rejected because its subject name did not match the issuer name of the current certificate.
- 30 / Authority and subject key identifier mismatch: The current candidate issuer certificate was rejected because its subject key identifier was present and did not match the authority key identifier current certificate.
- 31 / Authority and issuer serial number mismatch: The current candidate issuer certificate was rejected because its issuer name and serial number was present and did not match the authority key identifier of the current certificate.
- 32 / Key usage does not include certificate signing: The current candidate issuer certificate was rejected because its keyUsage extension does not permit certificate signing.

Chapter 4. Installation Related Issues

4.1. Index

This chapter describes possible issues encountered during the installation of a Steelhead appliance.

4.2. Installation steps for fail-to-wire scenarios

The bypass cards have two failure modes: Fail-to-wire and fail-to-block.

During the physical installation of a Steelhead appliance in a fail-to-wire scenario the following scenarios should be checked:

- When the Steelhead appliance is turned off, the LAN switch and WAN router should negotiate an Ethernet link and all traffic should flow through, unless fail-to-block has been configured.
- When the Steelhead appliance is turned on, configured and the optimization service is running, the LAN switch and WAN router should negotiate an Ethernet link towards the Steelhead LAN and WAN interfaces and all traffic should flow through.
- Now that the Steelhead appliance is working properly, check again that the fail-to-wire scenario works: Shutdown the Steelhead appliance with the command `reload halt`. The traffic should flow through.

Afterwards, every time a device on the WAN or on the LAN side of the Steelhead appliance is replaced or when the interface configuration of the Steelhead appliance or one of the connecting devices is changed, this fail-to-wire implementation should be checked again.

The following list is a good order to install new Steelhead appliances in an in-path environment:

- Without being connected to the network, turn on the Steelhead appliance and configure the IP addresses and routing details on the primary interface and in-path interfaces. In the in-path configuration, disable the Link State Propagation feature. Enable in-path support and enable optimization on the in-path interfaces.

Figure 4.1. Disable LSP during installation of a new Steelhead appliance

```
SH (config) # interface inpath0_0 ip address 10.0.1.6 /24
SH (config) # ip in-path-gateway inpath0_0 "10.0.1.9"
SH (config) # in-path enable
SH (config) # in-path interface inpath0_0 enable
SH (config) # no in-path lsp enable
```

- Save the configuration with the command `write memory` and turn off the Steelhead appliance with the command `reload halt`.
- Connect the cables to the LAN and WAN interfaces and confirm that the WAN router and LAN switch have an Ethernet link established.
- Turn on the Steelhead appliance and wait for the optimization service to be initialized and started.
- Confirm that the LAN and WAN interfaces on the Steelhead appliance have an Ethernet link established with the commands `show interface lan0_0 brief` and `show interface wan0_0 brief`.
- In the in-path configuration, enable the Link State Propagation feature with the configuration command `in-path lsp enable` and save the configuration with the command `write memory`.

4.3. Fail-to-Wire

After installing a Steelhead appliance, the in-path interfaces can be in one of the following two states:

- In a non-operational state, when the Steelhead appliance is turned off or when the in-path interface is not enabled.

  The failure configuration of the in-path interface can then either be fail-to-wire or fail-to-block:

  - In the fail-to-block configuration, where the by-pass card does not link the WAN and LAN interfaces together. Traffic from and to the LAN switch and WAN router is blocked.

    In the output of `show interfaces inpath0_0 configured` the failure mode is *Disconnect*.

  - In the fail-to-wire configuration, where the by-pass card does link the WAN and LAN interfaces together. The LAN switch and the WAN router are then directly connected with each other and will be able to negotiate an Ethernet link.

    In the output of `show interfaces inpath0_0 configured` the failure mode is *Bypass*.

- In an operational state, when the by-pass card has two Ethernet links are negotiated: One between the WAN interface and the WAN router and the LAN interface and the LAN switch. The by-pass card intercepts the packets from the LAN and WAN interface and let the network stack process them.

  In the output of `show interfaces inpath0_0 brief` the traffic status will be *Normal*.

Figure 4.2. Inpath0_0 is operational and set to fail-to-wire

```
SH # show interfaces inpath0_0 brief
Interface inpath0_0 state
   Up:                yes
   Interface type:    ethernet
   IP address:        10.0.0.6
   Netmask:           255.255.255.0
   IPv6 link-local address: fe80::20e:b6ff:fe90:352e/64
   MTU:               1500
   HW address:        00:0E:B6:90:35:2E
   Traffic status:    Normal
   HW blockable:      yes
SH # show interfaces inpath0_0 configured
Interface inpath0_0 configuration
   Enabled:           yes
   DHCP:              no
```

```
Dynamic DNS DHCP:   no
DHCPv6:             no
Dynamic DNS DHCPv6: no
IP address:         10.0.0.6
Netmask:            255.255.255.0
IPv6 address:
MTU:                1500
Failure mode:       Bypass
```

4.4. Cables

Connecting cables between Gigabit NICs is very simple: The auto-negotiation feature between two NICs takes care about the crossover cable versus straight through cable detection and this section can be ignored.

Connecting cables toward Fast Ethernet NICs is pretty tricky: Everything towards a switch needs to be a straight through cable, everything else needs to be a crossover cable.

When a by-pass card gets turned off or disabled and the failure mode is fail-to-wire, it will act as a crossover cable.

4.4.1. Cabling for different scenarios

4.4.1.1. Single Steelhead appliance located between switch and router

The WAN router is connected via a crossover cable, the LAN switch is connected via a straight through cable.

During fail-to-wire a straight through cable is required.

Table 4.1. Cabling for placement of Steelhead appliance between switch and router

	Installation	Fail to wire
WAN router to WAN interface	crossover	
LAN switch to LAN interface	straight through	
WAN router to LAN switch		C + C + S -> straight through

4.4.1.2. Single Steelhead appliance located between two routers

The WAN router and the LAN router are connected via crossover cables to the Steelhead appliance.

During fail-to-wire a crossover cable is required.

Table 4.2. Cabling for placement of Steelhead appliance between two routers

	Installation	Fail to wire
WAN router to WAN interface	crossover	
LAN router to LAN interface	crossover	
WAN router to LAN router		C + C + C -> crossover cable

4.4.1.3. Serial Steelhead cluster located between switch and router

The WAN router is connected via a crossover cable, the two Steelhead appliances are connected via a crossover cable and the LAN switch is connected via a straight through cable.

During fail-to-wire a crossover cable is required for the Steelhead appliance to the WAN router, a straight through cable is required between the two Steelhead appliances and the LAN switch and a straight through cable is required between the WAN router and the LAN switch.

Table 4.3. Cabling for placement of two Steelhead appliances between switch and router

	Installation	Fail to wire
WAN router to SH1 WAN interface	crossover	
SH1 LAN interface to SH2 WAN interface	crossover	
SH2 LAN interface to LAN switch	straight through	
WAN router to SH2 WAN interface		C + C + C -> crossover
SH1 LAN interface to LAN switch		C + C + S -> straight through
WAN router to LAN switch		C + C + C + C + S -> straight through

4.4.1.4. Serial Steelhead cluster located between two routers

The WAN router and the LAN router are connected via crossover cables to the Steelhead-appliances, the two Steelhead appliances are connected via a crossover cable with each other.

During fail-to-wire crossover cables are required everywhere.

Table 4.4. Cabling for placement of two Steelhead appliances between two routers

	Installation	Fail to wire
WAN router to SH1 WAN interface	crossover	
SH1 LAN interface to SH2 WAN interface	crossover	
SH2 LAN interface to LAN switch	crossover	
WAN router to SH2 WAN interface		C + C + C -> crossover
SH1 LAN interface to LAN switch		C + C + C -> crossover
WAN router to LAN switch		C + C + C + C + C -> crossover

4.4.2. 100 Mbps fixed speed and MDI-X

When the Ethernet link is configured to 100 Mbps, it can happen that the Ethernet link doesn't come up because of a wrong cable type. Forcing the NIC to be MDI-X can resolve this:

Figure 4.3. Configuring interface wan0_0 with MDI-x

```
SH (config) # interface lan0_0 force-mdi-x enable
% Interface must be set to 100/full before enabling force mdi-x.
SH (config) # interface lan0_0 speed 100
SH (config) # interface lan0_0 duplex full
SH (config) # interface lan0_0 force-mdi-x enable
SH (config) # show interfaces force-mdi-x
aux: false
inpath0_0: false
lan0_0: true
lo: false
primary: false
wan0_0: false
```

Afterwards, a check for the fail-to-wire scenario should be done when the Steelhead appliance is turned off.

4.4.3. Fiber by-pass card

When a Fiber by-pass card shows that it has detected an Ethernet link, it means that it has seen the light shone into the fiber string by the other side. So it is possible that the switch says it has detected a link, but that the Steelhead appliance says that it hasn't detected a link. The problem issue lies between the TX port of the switch and the RX port on the by-pass card.

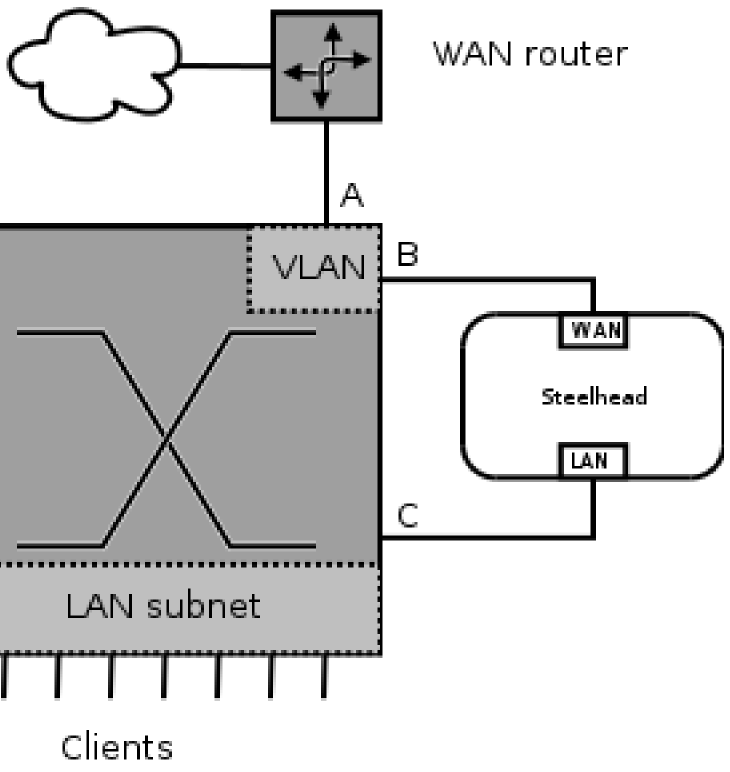
4.4.4. VLAN bridge to overcome slow fixed speed LAN interfaces

When a fixed speed and duplex setting is used on the LAN and the WAN interface and the devices connected to them should all be at the same speed and duplex settings. If the WAN bandwidth is more than about ten percent of the fixed speed, it might be worth to implement a VLAN bridge to overcome the fixed speed requirement on the Steelhead appliance LAN/WAN interfaces.

For example if the WAN bandwidth is 5 Mbps and the WAN router is set to 10 Mbps, the speed of the optimized traffic coming out of the LAN interface is capped to 10 Mbps. Suddenly the LAN interface would be the limiting factor.

Also, the WAN router could be managed by a third party which means changing the interfaces to auto-negotiate is not possible. The way to overcome this is by creating a VLAN bridge on the LAN switch:

Figure 4.4. VLAN bridge to overcome fixed speed/duplex settings



Port A should be set to the fixed speed and duplex setting of the WAN router. Port B and C should be set to auto-negotiation. The VLAN between port A and B should not have an IP address, it is just there to forward Ethernet packets.

This way, if the in-path interface goes into bypass, the LAN switch and WAN router are still able to talk to each other because the Ethernet link gets negotiated between port B and C.

4.5. Network Interface Card speed issues

4.5.1. Normal operation

The negotiated Ethernet links on the network interfaces should be stable and the interface error counters should not increase. The status and the counters can be seen with the command `show interfaces <interface name>`.

The counters in this output are cumulative since the start-up of the appliance and they can be reset to zero with the command `clear interfaces`.

Figure 4.5. The output of "show interfaces primary"

```
SH # show interfaces primary
Interface primary state
   Up:                 yes
   Interface type:     ethernet
   IP address:         10.17.6.119
   Netmask:            255.255.255.128
   IPv6 address:       2001:44b8:7bf1:a50:d69a:20ff:fec2:520e/64
   IPv6 auto-assigned: fe80::20e:b6ff:fe31:45e0/64
   Speed:              100Mb/s (auto)
   Duplex:             full (auto)
   MTU:                1500
   HW address:         00:0E:B6:31:45:E0

   RX bytes:           104060364
   RX packets:         660069
   RX mcast packets:   0
   RX discards:        0
   RX errors:          0
   RX overruns:        0
   RX frame:           0

   TX bytes:           3658160
   TX packets:         33089
   TX discards:        0
   TX errors:          1
   TX overruns:        0
   TX carrier:         1
   TX collisions:      0
```

The following fields are important:

- Up: Shows the administrative status of the network interface. *Yes* means that the interface is enabled.
- Speed: The value is the speed used by the network interface. If the value contains the string *(auto)*, then the speed is negotiated, otherwise it is a fixed speed.
- Duplex: The value is the duplex setting used by the network interface. If the value contains the string *(auto)*, then the duplex is negotiated, otherwise it is a fixed duplex setting.
- MTU: The MTU size defined on this network interface.
- RX bytes / RX packets / TX bytes / TX packets: The number of packets, sent and received, and the total number of bytes of data, sent and received.
- RX discards / RX errors / RX frame: Number of Ethernet frames which could not be processed because the Ethernet frame was corrupted.
- TX discards / TX errors / TX carrier / TX collisions: Number of Ethernet frames which could not be sent because of Ethernet layer related problems.
- RX overruns / TX overruns: Number of Ethernet frames which could not be processed because the NIC ran out of memory to store incoming frames or wasn't fast enough with the processing of frames the buffer.

4.5.2. Possible causes of TX and RX errors

4.5.2.1. Negotiation issues

The Ethernet specification states that if no Ethernet link can be negotiated that the network interface speed should be set to 10 Mbps and the duplex should be set to half-duplex.

This situation can happen when one side of the Ethernet link is set to a fixed duplex/speed and the other side is set to auto-negotiation.

Figure 4.6. Interface set for auto-negotiation but coming up as half-duplex

```
SH # show interfaces lan0_0
Interface lan0_0 state
   Up:                 yes
   Interface type:     ethernet
   Speed:              10Mb/s (auto)
   Duplex:             half (auto)
   MTU:                1500
   HW address:         00:0E:B6:8C:7A:EE
   Link:               yes
```

Alternatively when the NIC on the Steelhead appliance is set to a fixed speed and duplex but it receiving a lot of RX/TX errors in the counters:

Figure 4.7. Interface set for fixed speed but with a lot of RX/TX errors

```
SH # show interfaces lan0_0
Interface lan0_0 state
   Up:                 yes
   Interface type:     ethernet
   Speed:              100Mb/s
   Duplex:             full
   MTU:                1500
   HW address:         00:0E:B6:8C:7A:EE
   Link:               yes

   RX bytes:           104060364
   RX packets:         660069
   RX mcast packets:   0
   RX discards:        0
   RX errors:          12737
   RX overruns:        0
   RX frame:           12737

   TX bytes:           3658160
   TX packets:         33089
   TX discards:        0
   TX errors:          19244
   TX overruns:        0
   TX carrier:         6921
   TX collisions:      12323
```

These RX/TX counters are cumulative so make sure they are actually increasing before drawing conclusions about them.

4.5.2.2. Cable and switch port related issues

If the issue cannot be resolved by fixing the speed / duplex mismatch, consider a replacement of the cables.

If that doesn't overcome the problem, try a different port on the LAN switch or the WAN router and see if the problem migrates to the new port.

4.6. IP Subnet configuration related issues

To be able to communicate to other Steelhead appliances, each active in-path interface needs to have an IP address, subnet mask and a default gateway defined. If the IP address is defined in a tagged VLAN, also a VLAN tag ID should be defined.

The most common issues with regards to the configuration are that the in-path interface IP address is the wrong IP subnet, that the VLAN tag ID is not properly defined or that the default gateway is set wrong.

4.6.1. Wrong IP subnet for the in-path interface

The easiest way to determine if this is the case is to run tcpdump and check out the broadcast traffic like ARP requests, OSPF Hello packets and SMB traffic.

- The ARP requests will be for IP addresses which are not in the expected IP subnet.

  In this example the traffic from 192.168.1.0/24 while the in-path IP address is in 10.0.1.0/24:

  Figure 4.8. ARP requests are coming from IP subnet 192.168.2.0/24

```
SH # tcpdump -ni lan0_0 'arp or (vlan and arp)'
tcpdump: WARNING: lan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 65535 bytes
21:19:59.256938 ARP, Request who-has 192.168.1.1 tell 192.168.1.3 length 28
21:20:07.618390 ARP, Request who-has 192.168.1.254 tell 192.168.1.8 length 46
21:20:08.127260 ARP, Request who-has 192.168.1.13 tell 192.168.1.12 length 46
3 packets captured
2123 packets received by filter
0 packets dropped by kernel
```

- The SMB protocol uses IP broadcasts to announce services. These broadcasts should not leave the local IP subnet, so if you see them for IP addresses which are not on the expected IP subnet you know that there is something wrong.

  Figure 4.9. SMB broadcast packets are coming from IP subnet 192.168.2.0/24

```
SH # tcpdump -ni wan0_0 ether broadcast
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
21:35:19.514281 IP 192.168.2.55.42776 > 192.168.2.255.137: NBT UDP PACKET(137): QUERY; REQ \
    UEST; BROADCAST
21:35:21.407979 IP 192.168.2.55.42776 > 192.168.2.255.137: NBT UDP PACKET(137): QUERY; REQ \
    UEST; BROADCAST
21:35:21.678573 IP 192.168.2.55.42776 > 192.168.2.255.137: NBT UDP PACKET(137): QUERY; REQ \
    UEST; BROADCAST
```

- OSPF Hello packets are sent from the IP address of the router interface to a multicast address. If the IP address of the OSPF packet is not the IP address on the expected IP subnet you know that there is something wrong.

  Figure 4.10. OSPF broadcast and unicast packets and coming from IP subnet 192.168.170.0/24

```
23:20:13.384849 IP 192.168.170.8 > 224.0.0.5: OSPFv2, Hello, length 48
23:20:14.386980 IP 192.168.170.2 > 224.0.0.5: OSPFv2, Hello, length 48
23:20:14.414477 IP 192.168.170.8 > 192.168.170.2: OSPFv2, Database Description, length 32
23:20:14.415890 IP 192.168.170.2 > 192.168.170.8: OSPFv2, Database Description, length 32
23:20:14.418003 IP 192.168.170.2 > 192.168.170.8: OSPFv2, LS-Request, length 36
23:20:14.418258 IP 192.168.170.8 > 192.168.170.2: OSPFv2, LS-Request, length 108
23:20:14.418357 IP 192.168.170.8 > 224.0.0.5: OSPFv2, LS-Update, length 64
23:20:14.420459 IP 192.168.170.2 > 224.0.0.6: OSPFv2, LS-Update, length 292
```

To solve this issue:

- A visual confirmation of the ports of the devices connected to the LAN/WAN interfaces is required, with a follow-up in the configuration of these devices to check out which IP subnets are defined on them.
- A remote investigation by shutting down interfaces so Ethernet links go down on the connected devices, giving indication of the connected ports on the neighbouring devices.

4.6.2. In-path interface IP address is in the wrong VLAN

If the Ethernet frames generated by Steelhead appliance transmitted from the in-path interface are incorrectly tagged, the ARP requests coming out of it will never be answered or will be send out on the wrong VLAN.

First step: Ping the in-path interface IP address of the default gateway:

Figure 4.11. Ping the in-path interface IP address from the default gateway

```
RTR # ping
Protocol [ip]:
Target IP address: 10.0.1.6
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 10.0.1.9
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/3 ms

RTR # ping ip 10.0.1.6 source 10.0.1.9
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/3 ms
```

If the in-path interface is using the wrong VLAN, this will fail completely.

The output of tcpdump will show only ARP requests:

Figure 4.12. Tcpdump running for a ping from the Steelhead appliance

```
SH # tcpdump -leni wan0_0 'host 10.0.1.9 and icmp or (vlan and host 10.0.1.9 and icmp))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
18:53:46.123323 00:0d:b9:17:28:de > ff:ff:ff:ff:ff:ff, ethertype 802.1Q (0x8100), length 4 \
    6: vlan 3, p 0, ethertype ARP, Request who-has 10.0.1.6 tell 10.0.1.9, length 28
18:53:47.027193 00:0e:b6:42:f8:9c > ff:ff:ff:ff:ff:ff, ethertype 802.1Q (0x8100), length 4 \
    6: vlan 12, p 0, ethertype ARP, Request who-has 10.0.1.9 tell 10.0.1.6, length 28
```

So the ARP requests from the Steelhead appliance are on VLAN 12, while the ARP requests from the router are on VLAN 3.

4.6.3. Duplicate IP addresses on in-path interfaces

If an in-path interface has been disabled in the configuration, any IP address in the configuration will still be configured on the virtual in-path interface.

This means that if the same IP address has been configured on the inpath0_0 and inpath0_1 but inpath0_1 has been disabled, the two in-path interfaces will still get the same IP address configured. Despite that the in-path interfaces are independent, can be on the same IP subnet and have their own routing table, a duplicate IP address is not supported.

The solution would be to remove the IP address of the disabled in-path interfaces and a restart of the optimization service:

Figure 4.13. Remove the IP address of inpath0_1 via the CLI

```
SH (config) # no interface inpath0_1 ip address
SH (config) # restart
```

4.6.4. Duplicate IP addresses on the IP subnet

If an in-path interface detects that it has been configured with an IP address already in use on the IP subnet, it will complain about it in the logs:

Figure 4.14. Duplicate IP addresses on the in-path interfaces

```
kernel: [IP.ERR] inpath0_0: IP address 192.168.1.5 conflicts with 00:1c:c4:ee:3f:90
```

4.6.5. Default gateway unreachable

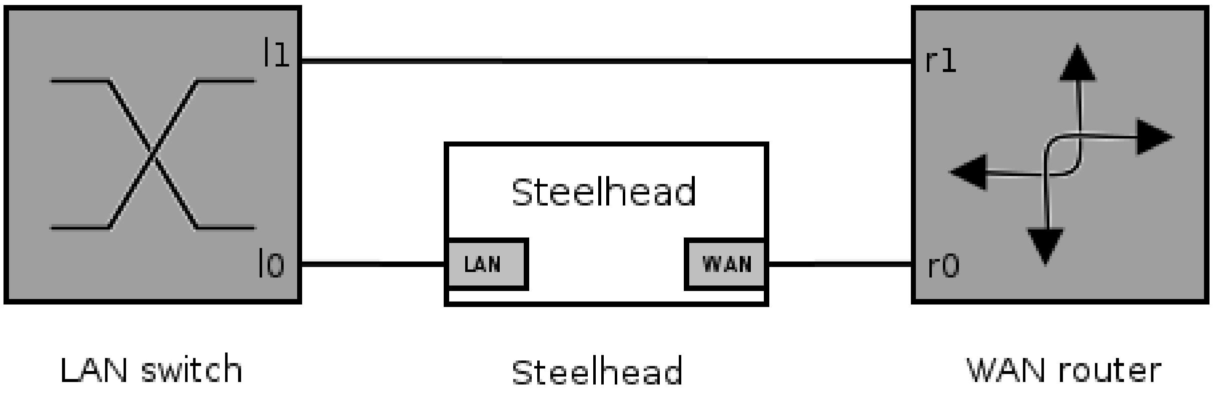
In the end, when everything is configured correctly, it just could be that the default gateway is unreachable. Ping any other IP addresses on the IP subnet and see if they work would be the next steps.

4.7. Wrong location

It could also be that the Steelhead appliance is connected to a wrong device in the network. One possibility is described in the previous section, where the IP subnet configured on the in-path interface is wrong, possibly because the LAN/WAN interfaces are connected to the wrong switch.

But there is another possibility, where the Steelhead appliance is properly put between the WAN router and LAN switch, but there is also a normal cable directly connected between the WAN router and LAN switch.

Figure 4.15. Steelhead appliance by-passed with a cable.



There are four possibilities:

- The LAN switch has the MAC address of the default gateway determined via interface *l0* and the WAN router has the MAC addresses of the hosts behind the LAN switch determined via the interface *r0*.

  As expected, all traffic goes through the Steelhead appliance and will be optimized.

- The LAN switch has the MAC address of the default gateway determined via interface *l0* and the WAN router has the MAC addresses of the hosts behind the LAN switch determined via the interface *r1*.

  The outgoing traffic will be marked for optimization, but the auto-discovery process will fail catastrophically because of the client-side asymmetry. The auto-discovery on the incoming traffic will fail safely because the auto-discovery probe will not be seen by the Steelhead appliance.

- The LAN switch has the MAC address of the default gateway determined via interface l1 and the WAN router has the MAC addresses of the hosts behind the LAN switch determined via the interface *r0*.

  The outgoing traffic will not be marked for optimization because it is not going through the Steelhead appliance. The auto-discovery of the incoming traffic will fail catastrophically because of the server-side asymmetry.

- The LAN switch has the MAC address of the default gateway determined via interface *l1* and the WAN router has the MAC addresses of the hosts behind the LAN switch determined via the interface *r1*.

  The outgoing traffic will not be marked for optimization because it is not going through the Steelhead appliance. The auto-discovery on the incoming traffic will fail safely because the auto-discovery probe will not be seen by the Steelhead appliance.

4.7.1. What can cause this?

This scenario can be caused in two ways:

- The Steelhead appliance got installed and the technician didn't take out the cable.
- The Steelhead appliance got installed, everything was fine and due to some reason later a site-technician sees that there is no cable between the LAN switch and WAN router and he thinks that this is a bad idea because he is only aware of scenarios without WAN optimizers and puts it back in.

These issues can be present but only manifest when the Steelhead appliance, WAN router, or LAN switch gets reset, and the devices need to re-discover the path to the MAC addresses of the default gateway, and that path bypasses the Steelhead appliance.

4.7.2. How to determine

4.7.2.1. With tcpdump

On a remote machine, start to ping a device behind the LAN switch. On the Steelhead appliance, run the tcpdump commands:

Figure 4.16. Run these tcpdump commands

```
SH # tcpdump -ni lan0 'icmp or (vlan and icmp)'
SH # tcpdump -ni wan0 'icmp or (vlan and icmp)'
```

If the traffic is going via the normal cable, you should only see the *ICMP Echo Reply* packets, the *ICMP Echo Request* packets or nothing at all.

4.7.2.2. On the LAN switch

To do this, you first need the MAC addresses of the LAN and WAN interfaces on the Steelhead appliance and the MAC address of the default gateway on the WAN router.

On the LAN switch, issue the command to see the ARP table. Search for the MAC address of the default gateway in the list and note down the outgoing port. Then search for the MAC addresses of the LAN and WAN interfaces in the list and note down the outgoing ports. These outgoing ports should be the same if the traffic goes via the Steelhead appliance.

4.7.2.3. On the WAN router

To do this, you first need the MAC addresses of the LAN and WAN interfaces on the Steelhead appliance and the MAC address of a device behind the LAN switch.

On the WAN router, issue the command to see the ARP table. Search for the MAC address of the LAN device in the list and note down the outgoing port. Then search for the MAC addresses of the LAN and WAN interfaces in the list and note down the outgoing ports. These outgoing ports should be the same if the traffic goes via the Steelhead appliance.

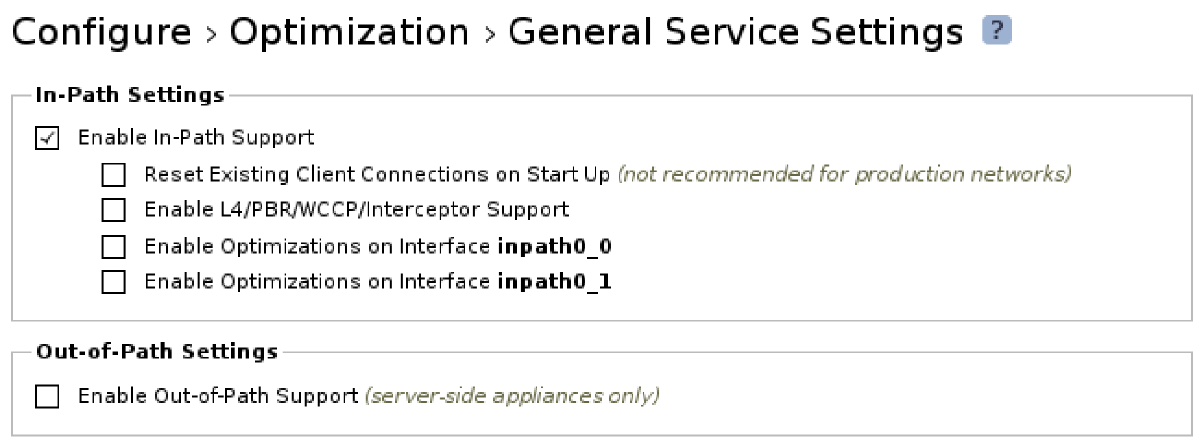
4.8. In-path support on interface not enabled

When a new in-path interface is activated in an existing Steelhead appliance and no optimization via that interface is happening, check out the following pre-requisites on the Steelhead appliance:

- The IP address, subnet mask, default gateway and VLAN tag ID for the in-path interface has been configured under Configure -> Networking -> In-path Interfaces.

- In-path optimization has been enabled for that in-path interface under Configure -> Optimization -> General.

Figure 4.17. No optimization support for inpath0_1



Configure › Optimization › General Service Settings

In-Path Settings

☑ Enable In-Path Support

☐ Reset Existing Client Connections on Start Up *(not recommended for production networks)*

☐ Enable L4/PBR/WCCP/Interceptor Support

☐ Enable Optimizations on Interface **inpath0_0**

☐ Enable Optimizations on Interface **inpath0_1**

Out-of-Path Settings

☐ Enable Out-of-Path Support *(server-side appliances only)*

On the CLI, check that the in-path interface is enabled with the `show in-path` command:

Figure 4.18. Enabled in-path interfaces for optimization

```
SH # show in-path
Enabled: yes
Kickoff: no
L4/PBR/WCCP/Interceptor: no
Maintain Multipath: no
Main Interface: inpath0_0
Asymmetric Routing Detection: yes
Asymmetric Routing Pass Through: yes

Optimizations Enabled On:
  inpath0_0

VLAN Tag IDs:
  inpath0_0: 0
```

4.9. Port security

Port security is a range of features on switches and routers to make sure that certain security policies are upheld.

The maximum number of MAC addresses policy can interfere with the operation of the Steelhead appliance: Normally a link between a switch and a router only has two devices on it. With an in-path device in it, that will increase and thus the number of MAC addresses seen by the router or switch increases.

On a Cisco switch or router, the status of the port security can be seen with the command `show port-security`.

Figure 4.19. Port security show commands

```
Switch# show port-security
Secure Port  MaxSecureAddr  CurrentAddr  SecurityViolation  Security Action
(Count)       (Count)          (Count)
---------------------------------------------------------------------------
      Gi0/1             1           1                 0          Protect
---------------------------------------------------------------------------
Total Addresses in System (excluding one mac per port)     : 0
Max Addresses limit in System (excluding one mac per port) : 1024

Switch#sh port-security interface gi0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Protect
Aging Time                 : 5 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address        : 00a0.1234.5678
Security Violation Count   : 0
```

Port-security can be disabled for a particular interface using command `no switchport port-security`. The number of MAC addresses allowed on an interface can be changed with the command `switchport port-security maximum <number>`.

Figure 4.20. Port security configuration commands

```
switch (config) # interface gig 0/1
switch (config-if) # no switchport port-security
switch (config-if) # switchport port-security maximum 3
```

4.10. Link State Propagation (LSP)

Link State Propagation is a feature on which an interface in an in-path interface pair will follow the state of the Ethernet link of the opposite interface.

Without Steelhead appliances, the WAN and LAN routers are directly connected. When the interface on the LAN router goes down (cable disconnected, router rebooting) then the interface on the WAN router will lose the Ethernet link. The routing process will be immediately informed about this and inform the rest of the routing cloud that the networks learned from the LAN router are unreachable.

With a Steelhead appliance between the two routers, when the interface on the LAN router goes down, it is only the LAN interface on the in-path interface which notices it. Since the WAN router will be unaware of this, the routing process on the WAN router will not get alerted and has to wait for a timeout on protocol level. With Link State Propagation enabled, when the LAN interface loses its Ethernet link, the in-path interface will also bring down the WAN interface and thus the interface on the WAN router will lose its Ethernet link too. Now the routing process on the WAN router will immediately be informed that path to the LAN router is down instead of having to wait for routing protocol timeouts.

4.10.1. How to determine which interface went down first

In this example the output of the `show interface lan0_0 brief` and `show interface wan0_0 brief` commands show that the lan0_0 went down (*Up: yes*) and that wan0_0 is brought down (*Up: no*).

Figure 4.21. Compare the "Up" status to see which interface was brought down.

```
SH # show in-path lsp
Link State Propagation Enabled: yes

SH # show interface lan0_0 brief
Interface lan0_0 state
   Up:                  yes
   Interface type:      ethernet
   Speed:               UNKNOWN
   Duplex:              UNKNOWN
   MTU:                 1500
   HW address:          00:0E:B6:8C:7C:42
   Link:                no

SH # show interface wan0_0 brief
Interface wan0_0 state
   Up:                  no
   Interface type:      ethernet
   Speed:               UNKNOWN
   Duplex:              UNKNOWN
   MTU:                 1500
   HW address:          00:0E:B6:8C:7C:41
   Link:                no
```

The LSP feature still works when you have multiple Steelhead appliances in a serial cluster.

4.10.2. LSP during installation

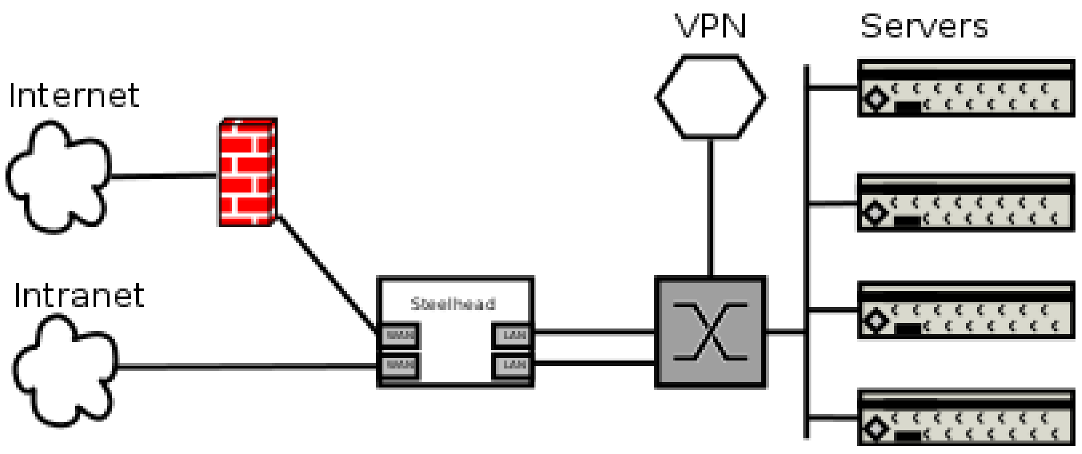
During installation a lot of swapping of cables might happen until everything is working properly. It could be easier to disable LSP at this time with the command `no in-path lsp enable` so that the persons on-site have direct visual feedback about link states when cables are plugged in. When all the cables are plugged in and all the Ethernet links are established, then LSP should be enabled again with the command `in-path lsp enable`.

4.11. VPN Concentrators

Steelhead appliances are not capable of optimizing any encrypted traffic between VPN concentrators, but it is of course capable of optimizing traffic coming out of the VPN concentrator.

A common problem is that the Steelhead appliances are setup on the network-side of the network, while the VPN concentrators are setup on the server-side of the network.

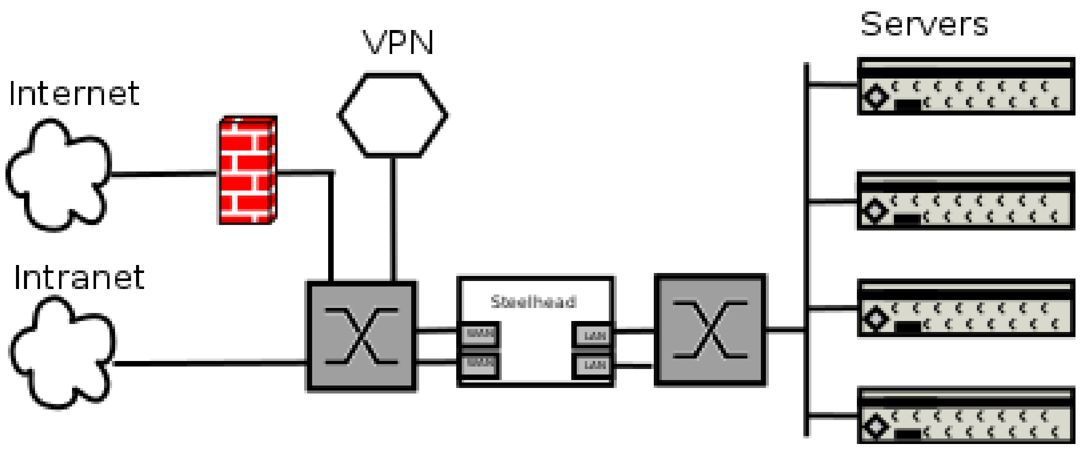
Figure 4.22. Wrong location of the VPN concentrator



In this case, the traffic from the hosts behind a remote VPN gateway towards the servers will not be optimized because the non-VPN traffic towards the servers does not touch the Steelhead appliance. Traffic towards the servers across the WAN will be optimized though.

The correct location for the VPN concentrator should be between the Steelhead appliance and the firewall.

Figure 4.23. Correct location of the VPN concentrator



4.12. Licenses

As discussed earlier, there are licenses for various features. The licenses for the Steelhead appliances can be found on the Riverbed Support website, except for the SSL license.

There are several reasons why the Steelhead appliance would need to be reconfigured with the licenses:

- The Steelhead appliances lost its configuration when it got factory reset because of relocation or software downgrade.
- The original license was a temporary evaluation license which has expired and the purchased license was not delivered to the networking group responsible for the Steelhead appliances.
- The device got replaced in an RMA and the licenses need to be swapped.
- Somebody got too enthousiastic on the license management page in the GUI and deleted the current licenses.

If the assets can't be found on the Riverbed Support website, please contact the Riverbed TAC for assistance. If the license required is an SSL license, then the application forms on the Riverbed Support website should be followed.

The following licenses are available on Steelhead appliances:

- BASE license, for example "LK1-SH10BASE-0000-...". This license is to allow the Steelhead appliance to perform SDR services like data reduction.
- CIFS license, for example "LK1-SH10CIFS-0000-...". This license is to allow the Steelhead appliance to perform CIFS latency optimization.
- MAPI license, for example "LK1-SH10EXCH-0000-...". This license is to allow the Steelhead appliance to perform MAPI latency optimization.
- SSL license, for example "LK1-SH40SSL-0000...". This license is to allow SSL pre-optimization and SSL secure peering.
- RSP license, for example "LK1-SH50RSP-0000...". This license is to allow the Riverbed Services Platform feature for RiOS 5.0 to be enabled.
- RSP license, for example "LK1-SH55RSPM-0000...". This license is to allow the Riverbed Services Platform feature on the xx20 and xx50 series models for RiOS 5.5 and later to be enabled.
- Print Package License, for example "LK1-SH55PKGPRNT-0000-...". This license is to allow a single instance of the RSP Print Package for RiOS 5.5 and later to be enabled.

- WAN Bandwidth licenses, for example "LK1-SH70BWLIMIT#100000-0000-...". This license is to allow the optimizable WAN bandwidth for the EX1160VH and EX1260VH models to be increased from 50 Mbps to 100 Mbps.
- Professional Services license, for example "LK1-PROFSRV-0000...". This license is to allow access to the underlying operating system without the challenge/response authorization.
- High Speed TCP license, for example "LK1-SH20HTCP-0000...". This license is to enable the High Speed TCP feature. It has been obsoleted.
- Hardware model upgrade, for example "LK1-SH20HWUP-0000...". This license is for the xx20 series models and allows an upgrade towards a higher capacity model.
- Machine specification license, for example "LK1-MSPEC2050M-0000...". This license is to specify the base model of the xx50 series model. In this case the device is a 2050M model.
- Machine specification configuration upgrade, for example "LK1-MSPECHWUP1050H-0000...". This license is for the xx50 series models and allows an upgrade towards a higher capacity configuration.
- Granite license, for example "LK1-GRANITE-000...". This license is to enable the Granite feature.
- SCPS license, for example "LK1-SH55SCPS-0000...". This license is to enable the Space Communication Protocol Specifications specific features.
- FIPS license, for example "LK1-FIPS-0000...". This license is to enable the Steelhead appliance to run the FIPS specific software.

The following licenses are available on Steelhead Mobile Controllers:

- Base license, for example "LK1-SMCVE#00087F85-0000...". This license is to allow the SMC VE to work.
- Virtual SMC license, for example "LK1-SMCVBASE#VA6MR00012334-0000...". This license is to allow the Virtual SMC to work.
- Additional client license, for example "LK1-SMCEL#190+4E12346D-0000-...". This license allows extra Steelhead Mobile Clients to be active. The string *190* in the license is the extra capacity in hex.
- Standard CIFS license for Steelhead Mobile Clients, for example "LK1-SMCCIFS-0000-...". This license allows the Steelhead Mobile Client to perform CIFS latency optimization.
- Standard MAPI license for Steelhead Mobile Clients, for example "LK1-SMCMAPI-0000-...". This license allows the Steelhead Mobile Client to perform MAPI latency optimization.
- SSL license, for example "LK1-SMCSSL-0000...". This license is to allow SSL pre-optimization and SSL secure peering.

The following licenses are available on Central Management Console:

- Base license, for example "LK1-CMC10BASE-0000". This license is to allow the CMC appliance to manage the devices.
- Base license for the CMC VE, for example "LK1-CMCVEBASE#V58GW000D1234-0000-". This license is to allow the CMC VE appliance to manage the devices. The string *V58GW000D1234* is the serial number of the CMC VE.
- Additional devices license, for example "LK1-CMCVE#32+50CA2E1D-0000-". This license increases the number of devices allowed to be managed. The string *32* is the extra number of devices.
- Additional devices license, for example "LK1-CMC10S0050-0000". This license increases the number of devices allowed to be managed. The string *0050* is the extra number of devices.
- Increased devices license, for example "LK1-CMCIL#122+51535ABE-0000-". The string *122* is the hex value of the extra number of devices.

The following licenses are available on the Interceptor appliance:

- The Interceptor base license, for example "LK1-IB10BASE-0000-...". This license is to allow the Interceptor to redirect traffic.

4.12.1. Licenses and tokens

In the xx20 series models, there were no licenses to upgrade between models. In the xx50 series models, there are licenses to upgrade between the different configurations inside the same model, but the appliances coming out of the manufacturing plants still had the distinction between if it was a 1050L, 1050M or 1050H model.

In the newer CX and EX series, the appliances coming out of the manufacturing plants will be the basic models like a 755 or 760 model, but not with licenses to make it a 755L or 755H configuration.

4.12.1.1. Licenses and tokens on the CX and EX series

When obtaining a new CX or EX series model, the configuration first needs to be activated. This can be done by going to the Riverbed Licensing website at https://licensing.riverbed.com/. You will be asked for your name, email address, phone number and device serial number. Next you will be asked what the base level is you want to make for it, for example an L or an M or an H model.

Once activated, the licenses are generated and available for download when the Steelhead appliance is powered up:

Figure 4.24. Automatic fetching of licenses at the startup of a new CX series model

```
SH mgmtd[5156]: [mgmtd.INFO]: Attempting automatic license retrieval
SH mgmtd[5156]: [mgmtd.INFO]: Next automatic license retrieval attempt in 81002 seconds
SH mgmtd[5156]: [mgmtd.NOTICE]: Installing new license 'LK1-MSPECCX755H-0000-0000-1-72D2-F \
    FCA-9FBC'
SH mgmtd[5156]: [mgmtd.INFO]: Starting database commit
```

After that the optimization server gets started and the Steelhead appliance will start optimization new TCP sessions.

Of course there are various network related problems which can occur.

- DNS lookup failures for the licensing server: Can the DNS server resolve external hosts?
- Unable to connect to the licensing server: Does the traffic need to go through a web proxy?

Figure 4.25. Failure of automatic fetching because of a DNS related failure

```
SH mgmtd[5156]: [mgmtd.INFO]: Attempting automatic license retrieval
SH mgmtd[5156]: [mgmtd.WARNING]: Automatic licensing server api.licensing.riverbed.com is  \
    unreachable: Couldn't resolve host name
SH mgmtd[5156]: [mgmtd.INFO]: Next automatic license retrieval attempt in 300 seconds
```

Once access to the licensing server has been fixed, the command `license autolicense fetch` can be used to force an update of the licenses.

Figure 4.26. Manual update of the licenses with "license autolicense fetch"

```
SH cli[9572]: [cli.INFO]: user admin: Executing command: license autolicense fetch
SH cli[9572]: [cli.INFO]: user admin: Command license autolicense fetch authorized
SH mgmtd[5156]: [mgmtd.INFO]: Attempting automatic license retrieval
SH mgmtd[5156]: [mgmtd.INFO]: Next automatic license retrieval attempt in 81571 seconds
SH mgmtd[5156]: [mgmtd.INFO]: Skipping already installed license 'LK1-MSPECCX755H-0000-000 \
    0-1-72D2-FFCA-9FBC'
SH mgmtd[5156]: [mgmtd.INFO]: Skipping already installed license 'LK1-MSPECCX755M-0000-000 \
    0-1-6799-DA37-1EF7'
SH mgmtd[5156]: [mgmtd.INFO]: Skipping already installed license 'LK1-SH10BASE-0000-0000-1 \
    -E32A-DC6A-E50F'
SH mgmtd[5156]: [mgmtd.INFO]: Skipping already installed license 'LK1-SH10CIFS-0000-0000-1 \
    -8BD2-4668-8EDA'
SH mgmtd[5156]: [mgmtd.NOTICE]: Installing new license 'LK1-SH10EXCH-0000-0000-1-211C-36AC \
    -455C'
```

4.12.1.1.1. SSL licenses

The SSL licenses will be distributed in this way too, but they are only generated once a day and will not be available immediately.

4.12.1.2. Licenses and tokens on the virtual appliances

When purchasing a virtual appliance, you will get a License Request Token. Once the virtual machine has been created and the appliance software has been installed in the virtual machine, you will enter this License Request Token in the GUI and it is converted to a license Request String. This License Request String contains some unique information about the virtual machine like the UID and the MAC address. The License Request String can be redeemed on the Riverbed Licensing website and the licenses will be generated. After that the license can be installed manually or automatically fetched by the appliance.

4.13. LAN and WAN cable switched

The processing of new TCP sessions depends of which interface the naked SYN (TCP SYN packet without an auto-discovery probe) gets received. If the naked SYN comes in via the WAN interface, then the Steelhead appliance will not process the packet but just forward it, marking the pass-through session as *SYN on WAN side*. Only if the naked SYN comes in via the LAN interface, the Steelhead appliance will check what to do with it via the In-path Rules and, if allowed, perform auto-discovery for it.

If the Steelhead appliance is wrongly cabled, with the LAN interface to the WAN router and the WAN interface to the LAN switch, no traffic coming from the LAN will be optimized.

There are two ways to determine this issue is happening: A visual inspection and a tcpdump with ping approach.

4.13.1. Visual check

The instructions are very simple: Walk to the Steelhead appliance, find the cable connected to the LAN interface and follow it all the way to the next device. If it is the router you expect to be on the WAN interface, then the cable is swapped.

4.13.2. Tcpdump and ping check

This will require three CLI sessions on the Steelhead appliance.

In the first CLI session, run tcpdump on the LAN interface. In the second CLI session, run tcpdump on the WAN interface. In the third CLI session, ping the IP address of a host which traffic doesn't get optimized.

Figure 4.27. Three CLI sessions for troubleshooting

```
SH # tcpdump -ni lan0_0 '(icmp and host 10.0.1.1) or (vlan and icmp and host 10.0.1.1)'
```

```
SH # tcpdump -ni wan0_0 '(icmp and host 10.0.1.1) or (vlan and icmp and host 10.0.1.1)'
```

```
SH # ping -I 10.0.1.6 10.0.1.1
```

We expect this ICMP Echo Request traffic to show up on the tcpdump output of the lan0_0 interface. In the scenario where the LAN and WAN cables are swapped, the *ICMP Echo Request* and *ICMP Echo Reply* packets show up on the WAN interface.

4.13.2.1. ICMP Echo Request and Echo Reply show up on the WAN interface.

When both the *ICMP Echo Request* and *ICMP Echo Reply* packets are seen up on the WAN interface, which indicates that the host is located behind the WAN interface of the Steelhead appliance.

Figure 4.28. ICMP Echo Request and Echo Reply show up on the WAN interface.

```
SH # ping -I 10.0.1.6 10.0.1.1
PING 10.0.1.6 (10.0.1.6) from 10.0.1.6 : 56(84) bytes of data.
64 bytes from 10.0.1.1: icmp_seq=0 ttl=63 time=2 ms
64 bytes from 10.0.1.1: icmp_seq=1 ttl=63 time=1 ms
64 bytes from 10.0.1.1: icmp_seq=2 ttl=63 time=1 ms

SH # tcpdump -ni lan0_0 '(icmp and host 10.0.1.1) or (vlan and icmp and host 10.0.1.1)'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 96 bytes

SH # tcpdump -ni wan0_0 '(icmp and host 10.0.1.1) or (vlan and icmp and host 10.0.1.1)'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
08:22:05.703095 IP 10.0.1.6 > 10.0.1.1: ICMP echo request, id 9738, seq 0, length 64
08:22:05.719544 IP 10.0.1.1 > 10.0.1.6: ICMP echo reply, id 9738, seq 0, length 64
08:22:06.728721 IP 10.0.1.6 > 10.0.1.1: ICMP echo request, id 9738, seq 1, length 64
08:22:06.736897 IP 10.0.1.1 > 10.0.1.6: ICMP echo reply, id 9738, seq 1, length 64
08:22:07.752379 IP 10.0.1.6 > 10.0.1.1: ICMP echo request, id 9738, seq 2, length 64
08:22:07.762682 IP 10.0.1.1 > 10.0.1.6: ICMP echo reply, id 9738, seq 2, length 64
```

In this case it is clear that the host is on the WAN side of the Steelhead appliance and therefore the naked SYN packet never gets processed.

4.14. After an RMA

When a full chassis gets replaced in the RMA process, some minor changes and updates might be required.

4.14.1. The replacement device doesn't boot

There are several kind of "not booting": No power, no output to the console, not completing the boot process.

4.14.1.1. No power

When the power cables get connected to the replacement chassis and the device gets turned on but no lights or fans come on:

- If the device is a 1RU or 3RU chassis, reseat the power supplies and hard disks.
- If that doesn't resolve the issue, install the power supplies of the old chassis in the replacement chassis.
- If that doesn't resolve the issue, open the chassis and press down all the connectors and reseat the memory.

If the replacement chassis doesn't power up at the end, call the Riverbed TAC to arrange a Support DOA RMA.

4.14.1.2. Not completing the boot process

This is when the device boots and initializes the kernel, as can be seen on the serial console, but it never ends up at the login prompt:

- Reseat the hard disks.
- Reseat the network cards.

If this doesn't resolve it, capture the full boot log and contact the Riverbed TAC to determine where it goes wrong here.

4.14.2. The extra memory or in-path card did not get pre-installed

The chassis kept in the depots are the basic models, they don't have extra kits like the RSP memory or extra in-path cards installed.

So when a chassis with RSP memory installed has to be replaced, two packages will get delivered: One big box with the chassis, and one small box with the RSP memory. This memory needs to be installed before the device gets installed and powered up.

The *Hardware Maintenance Guide* document on the Riverbed Support website has clear instructions on how to open the chassis and where the additional parts need to placed. The Multimedia menu in the Knowledge Base section on the Riverbed Support website has videos with instructions on how to replace parts in various appliances.

4.14.3. The licenses are absent

When the licenses are absent, the device will not start the optimization service. An overview of the licenses can be found in the asset section on the Riverbed Support website. If the asset list does not have the device on it, please contact Riverbed TAC to request a copy of the licenses.

Because of legal restrictions, the SSL license cannot be re-issued by Riverbed TAC. Please go to the Riverbed Support website and apply for them there.

4.14.4. The RiOS software version is incorrect

The RiOS version on the replacement chassis is installed during manufacturing and might not be the one you want to run in your network. Please follow the RiOS upgrade instructions in the Downgrade and upgrade issues chapter to go to the RiOS version you want to run.

4.15. Best cabling for remote management

In the standard design, the primary interface is used for management of the Steelhead appliance: The GUI and CLI are accessible via it, the remote logging of system messages and SNMP traps are send out via it. Also, the traffic for CIFS pre-population and Windows Active Directory integration is send out via this interface.

To increase the manageability of the Steelhead appliance, especially the central ones in data centers, several steps can be taken:

- Connect the auxiliary interface to a device on the WAN side of the Steelhead appliance. In the main routing table, configure an entry for a remote network management host going out via the IP subnet defined on the auxiliary interface. This way if the primary interface is unreachable (due to a failure of the optimization service, a fail-to-block configuration or a problem with the negotiation of the Ethernet link on the in-path interface), the device is still reachable via the auxiliary interface.
- Connect the serial port to a terminal server or to a serial-over-TCP module and connect the Ethernet port of the terminal server or module to a device on the WAN side of the Steelhead appliance.

Figure 4.29. Cabling for remote management

```
                         Terminal
                          Server
    .-,(  ),-.         __________
 .-(          )-.     [_...__...o]
(       WAN      )        |   ^
 '-(          ).-'        |   |   Serial __________   PRI
     '-.( ).-'            |   '---------|          |-----------.
         ^                |     ________|          |           |
         |                v    v    AUX |          |           v
         |             __________       |          |      __________
         '------------[_...__...o]<-----|_...__...o|---->[_...__...o]
                                    WAN              LAN
                       WAN router         Steelhead       LAN router
```

Figure 4.30. Configuration for routing towards remote management system via the auxiliary interface

Chapter 5. Operation Related Issues

5.1. Index

This chapter describes various issues encountered during the operational of a Steelhead appliance in the network.

- Hardware related issues, like hard disk failures, power supply failures, fan related failures and memory related failures.
- Admission control.
- Auto-negotiation and speed/duplex issues.
- LAN speed is maxed out.
- Unexpected restarts of the optimization service.
- Unexpected reboots of the Steelhead appliance.
- Watchdogs on the Steelhead appliance.
- Downgrade and upgrade related issues.
- Time related issues.
- Firewalls in the path.
- Data Store Synchronization.
- Data store related errors.
- WCCP related issues.
- Asymmetric Routing.
- IP Routing related issues.
- Alarms and health status.
- Long lived TCP sessions.
- Order of the in-path rules.
- Network monitoring and network management systems.
- Web proxies and web caches.
- Security.
- The Riverbed Services Platform (RSP)
- Access related issues
- Partitions running out of space
- Memory usage
- LAN side traffic seen
- Service errors
- Can this application be optimized?
- Interceptor cluster

- Expiring SSL certificates

5.2. Hardware related issues

This section describes the possible hardware related issues during the operation of the Steelhead appliance.

5.2.1. Storage

There are three kinds of storage devices in the Steelhead appliances:

- USB flash disk with the boot-partition, the configuration and a copy of the latest installed RiOS software.
- One or more hard disk drives.
- One or more solid-state disks

From a troubleshooting point of view, the USB flash disk is not interesting.

5.2.1.1. Hard disks

The hard disk in a Steelhead appliance can get hammered a lot and can fail after time. That is why in the higher end Steelhead appliances their hard disks are mirrored in a RAID array, so that if one hard disk fails the data is still available on the mirror hard disk.

On the Steelhead appliance without a RAID array, the failure of a hard disk will cause initial slowness in the processing of optimized TCP sessions, and at a certain moment the hardware watchdog will timeout and restart the appliance and then the device will fail in the boot process.

In case of a hard disk failure there are different steps per model:

- The low-end desktop models of the xx20, xx50, CX and EX series models have a single hard disk or a dual non-RAID set of disks. The whole chassis needs to be replaced.
- The 1RU xx20 series models have a single hard disk. The whole chassis needs to be replaced.
- The 1RU xx50 series 1050U, 1050L and 1050M models with a single hard disk, but the appliance can rebuild itself during the startup from the flash partition. Only a hard disk replacement is needed.
- The 1RU xx50 series 1050H model has two hard disks but they are in a RAID0 array. The appliance can rebuild itself during the startup of the appliance and only a hard disk replacement is needed.

  If a system dump is not available but the serial console bootlogs are, search for the available disks in with the string `Attached SCSI disk`

  Figure 5.1. Grep for "Attached SCSI disk" in the bootlogs

  ```
  [~/sysdumps] edwin@t43>grep "Attached SCSI disk" dmesg-boot
  sd 0:0:0:0: [sda] Attached SCSI disk
  sd 1:0:0:0: [sdb] Attached SCSI disk
  sd 6:0:0:0: [sdc] Attached SCSI disk
  ```

  If `sd 1:0:0:0` shows up, that is disk 1 which means that disk 0 is missing.

- The 1RU xx50 series 1050UR, 1050LR, 1050MR, 1050HR models and CX1555 models have four hard disks in a RAID10 array. In case of a single hard disk failure only the hard disk needs to be replaced. In case of a multiple hard disk failure on the same RAID1 array, the chassis can rebuild itself during the startup from the flash partition.
- The 1RU xx50 series 2050 model, the 3RU xx50 series 5050 and 6050 models have multiple hard disks in a RAID10 array. In case of a single hard disk failure only the hard disk needs to be replaced. In case of a multiple hard disk failure on the same RAID1 array, the whole chassis needs to be replaced.

- The 7050, 5055 and 6055 models have the hard disks in a RAID1 configuration while the solid state disks are in a FTS configuration.

During the RMA process of a broken hard disk, the device is running in a degraded mode. If the appliance has Silver level or Gold level support contract, you might want to obtain spare hard disks so you can replace them faster and thus get the redundancy on the RAID array back faster.

5.2.1.2. Solid-state disks

The solid-state disks will suffer from Write Endurance: The number of write cycles to any block of flash is limited. The Steelhead appliance keeps track of the number of writes per block of flash and if it gets higher than the manufacturer specifications it will raise an alarm that the solid-state disk needs to be replaced.

5.2.1.3. RAID status and rebuilding

The operational status of the RAID array can be seen with the command `show raid diagram`.

Figure 5.2. Output of the command "show raid diagram" on a 2050 model

```
SH # show raid diagram
[  0 : online   ][  1 : online   ][  2 : online   ][  3 : online ]
```

When a hard disk fails, its status will go from *online* to *missing* or *degraded*. When the broken hard disk is replaced, the status will go to *rebuilding*. When the rebuilding has finished, then the status will go back to *online*.

In this following example there is a RAID10 array of 12 hard disks, with the RAID1 array of disk 2 and 3 being rebuild, the RAID1 array of disk 10 and 11 is degraded because of a broken disk and the RAID1 array of disk 8 and 9 still okay but disk 9 is currently degraded.

Figure 5.3. Output of the command "show raid diagram" on a model 5050 model

```
SH # show raid diagram
[  0 : online   ][  1 : online   ][  2 : online   ][  3 : rebuilding ]
[  4 : online   ][  5 : online   ][  6 : online   ][  7 : online     ]
[  8 : online   ][  9 : degraded ][ 10 : online   ][ 11 : missing    ]
[ 12 : missing  ][ 13 : missing  ][ 14 : missing  ][ 15 : missing    ]
```

The rebuild of a RAID array will take about 12 hours, depending on how busy the Steelhead appliance is. On the xx50, CX and EX series models, the progress of the rebuilding can be seen with the command `raid swraid mdstat`:

Figure 5.4. Output of the command "raid swraid mdstat" on a 5050 model

```
SH # raid swraid mdstat
Personalities : [linear] [raid0] [raid10] [raid6] [raid5]
md3 : active raid10 sdd6[12] sda6[0] sdl6[11] sdk6[10] sdj6[9] sdi6[8] sdh6[7] sdg6[6] sdf \
    6[5] sde6[4] sdc6[2] sdb6[1]
      146512128 blocks 64K chunks 2 near-copies [12/11] [UUU_UUUUUUUU]
        resync=DELAYED

md0 : active raid10 sdd5[12] sda5[0] sdl5[11] sdk5[10] sdj5[9] sdi5[8] sdh5[7] sdg5[6] sdf \
    5[5] sde5[4] sdc5[2] sdb5[1]
      781288704 blocks 64K chunks 2 near-copies [12/11] [UUU_UUUUUUUU]
        resync=DELAYED

md2 : active raid10 sdd2[3] sda2[0] sdl2[11] sdk2[10] sdj2[9] sdi2[8] sdh2[7] sdg2[6] sdf2 \
    [5] sde2[4] sdc2[2] sdb2[1]
      37784448 blocks 64K chunks 2 near-copies [12/12] [UUUUUUUUUUUU]

md1 : active raid10 sdd3[12] sda3[0] sdl3[11] sdk3[10] sdj3[9] sdi3[8] sdh3[7] sdg3[6] sdf \
    3[5] sde3[4] sdc3[2] sdb3[1]
      100678656 blocks 64K chunks 2 near-copies [12/11] [UUU_UUUUUUUU]
      [=>...................]  recovery =  5.4% (920064/16779776) finish=6.3min speed=4182 \
    1K/sec

unused devices: <none>
```

In this example, disk 3 was replaced. The re-syncing of RAID1 array md2 has been completed already while md1 is currently at 5.4% completion while completion is estimated in 6.3 minutes. md3 and md0 will be made redundant after this.

On machines with multiple hard disks in a RAID10 configuration, the machine can survive a multiple hard disk failure as long as the failures are not on the same RAID1 array.

If a disk is in the *degraded* state then it is not yet removed from the RAID array but it still can cause performance problems. To remove it from the RAID array use the command `raid swraid fail-disk`.

5.2.1.4. Indication of a failure of a disk

The failure of a disk is often indicated by a slower-than-normal performance of a Steelhead appliance: The RAID controller or RAID software has not yet marked the hard disk as broken, but it sees long delays in reading and writing with it.

In the system logs a failing disk can be seen with the log entries like:

Figure 5.5. A Failing disk in the system logs

```
SH kernel: ata4.00: exception Emask 0x0 SAct 0x1 SErr 0x0 action 0x0
SH kernel: ata4.00: (irq_stat 0x40000008)
SH kernel: ata4.00: cmd 60/3e:00:43:00:00/00:00:00:00:00/40 tag 0 cdb 0x0 data 31744 in
SH kernel:          res 41/40:3e:43:00:00/00:00:00:00:00/40 Emask 0x9 (media error)
SH kernel: ata4.00: configured for UDMA/133
SH kernel: SCSI error : <3 0 0 0> return code = 0x8000002
SH kernel: Info fld=0x4000000 (nonstd), Invalid sdd: sense = 72 11
SH kernel: end_request: I/O error, dev sdd, sector 67
SH kernel: Buffer I/O error on device sdd1, logical block 33
SH kernel: ata4: EH complete
SH kernel: SCSI device sdd: 490350672 512-byte hdwr sectors (251060 MB)
```

5.2.2. Power supplies

The Steelhead appliances with multiple power supplies are able to sustain the loss of one. An alarm will be raised if a power supply has failed.

5.2.3. Fans

Depending on the model, there are one or multiple fans in a Steelhead appliance. An alarm will be raised if a fan has failed.

Figure 5.6. Output of the command "show stats fan" on the 100 / 200 / 300 models

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       5869    2657    ok
```

Figure 5.7. Output of the command "show stats fan" on the 520 / 1020 / 1520 / 2020 models

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       3000    750     ok
2       3000    750     ok
```

Figure 5.8. Output of the command "show stats fan" on the 3020 / 3520 / 5520 / 6020 / 6120 models

```
SH # show stats fan
FanId   RPM     Min RPM Status
0       4963    712     ok
1       4821    712     ok
2       4963    712     ok
3       4821    712     ok
4       4963    712     ok
5       4963    712     ok
```

Figure 5.9. Output of the command "show stats fan" on the 150 / 250 / 550 / 555 / 755 models

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       3559    2500    ok
2       3573    2500    ok
3       3588    2500    ok
```

Figure 5.10. Output of the command "show stats fan" on the 1050 / 2050 models

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       8760    1080    ok
2       9240    1080    ok
3       9480    1080    ok
5       9120    1080    ok
7       9120    1080    ok
```

Figure 5.11. Output of the command "show stats fan" on the 5050 / 6050 / 7050 models

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       3720    1080    ok
2       3840    1080    ok
3       3720    1080    ok
4       4920    1080    ok
5       3720    1080    ok
6       3720    1080    ok
```

Figure 5.12. Output of the command "show stats fan" on the CX255 model

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       5133    164     ok
3       5192    164     ok
```

Figure 5.13. Output of the command "show stats fan" on the CX555 / CX755 model

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       4518    2500    ok
2       4397    2500    ok
3       4368    2500    ok
```

Figure 5.14. Output of the command "show stats fan" on the CX5055 / CX7055 model

```
SH # show stats fan
SnId   RPM     Min RPM Status
1       5520    1080    ok
2       6840    1080    ok
3       5520    1080    ok
4       6960    1080    ok
5       5400    1080    ok
6       6840    1080    ok
7       5520    1080    ok
8       6840    1080    ok
```

Figure 5.15. Output of the command "show stats fan" on the DX8000 model

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       5280    1080    ok
2       6840    1080    ok
3       5400    1080    ok
4       7080    1080    ok
5       5280    1080    ok
6       6720    1080    ok
7       5400    1080    ok
8       6720    1080    ok
```

Figure 5.16. Output of the command "show stats fan" on the EX560 / EX760 model

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       5520    1800    ok
2       5520    1800    ok
3       5520    1800    ok
```

Figure 5.17. Output of the command "show stats fan" on the EX1160 model

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       6000    1080    ok
3       5640    1080    ok
4       5040    1080    ok
5       5760    1080    ok
6       5160    1080    ok
7       5640    1080    ok
8       5160    1080    ok
9       5640    1080    ok
10      5160    1080    ok
```

Figure 5.18. Output of the command "show stats fan" on the EX1260 / EX1360 model

```
SH # show stats fan
FanId   RPM     Min RPM Status
1       5520    1080    ok
2       6840    1080    ok
3       5520    1080    ok
4       7080    1080    ok
5       5520    1080    ok
6       6840    1080    ok
7       5760    1080    ok
8       6840    1080    ok
```

5.2.4. Memory errors

The memory in the Steelhead appliances supports ECC, which means it can determine if there are problems inside the memory modules. An alarm will be raised if such a problem has been detected.

Figure 5.19. Output of the command "show stats ecc-ram"

```
SH # show stats ecc-ram
No ECC memory errors have been detected
```

5.3. Admission Control

Admission Control is a state of the optimization service where it will no longer intercept any new TCP sessions because of licensing limitations or internal problems.

When in admission control, except for MAPI based admission control, the Steelhead appliance will drop all the Out-of-Band splices and the TCP sessions of the Connection Pools to the connected Steelhead appliances. When the Steelhead appliance comes out of admission control, the OOB Splice and the Connection Pool will be setup again once a new optimized TCP session towards the remote Steelhead appliance will be setup.

The various values for admission control can be seen in the CLI with the command `show admission`:

Figure 5.20. Output of the command "show admission" for a model 560L

```
SH # show admission
Enable Admission Control Override Settings: no

  Override Settings:
    Connection Enable:   250
    Connection Cutoff:   260
    Memory Enable:       1300 MB
    Memory Cutoff:       1400 MB
    Low Memory Ratio:    96%

  Current Settings:
    Connection Enable:   250
    Connection Cutoff:   260
    Memory Enable:       1300 MB
    Memory Cutoff:       1400 MB
    Low Memory Ratio:    96%

  Current State:
    Connections:         2
    Memory:              995 MB
```

Or in the system dump with the following command:

Figure 5.21. Determining the admission control values via the system dump

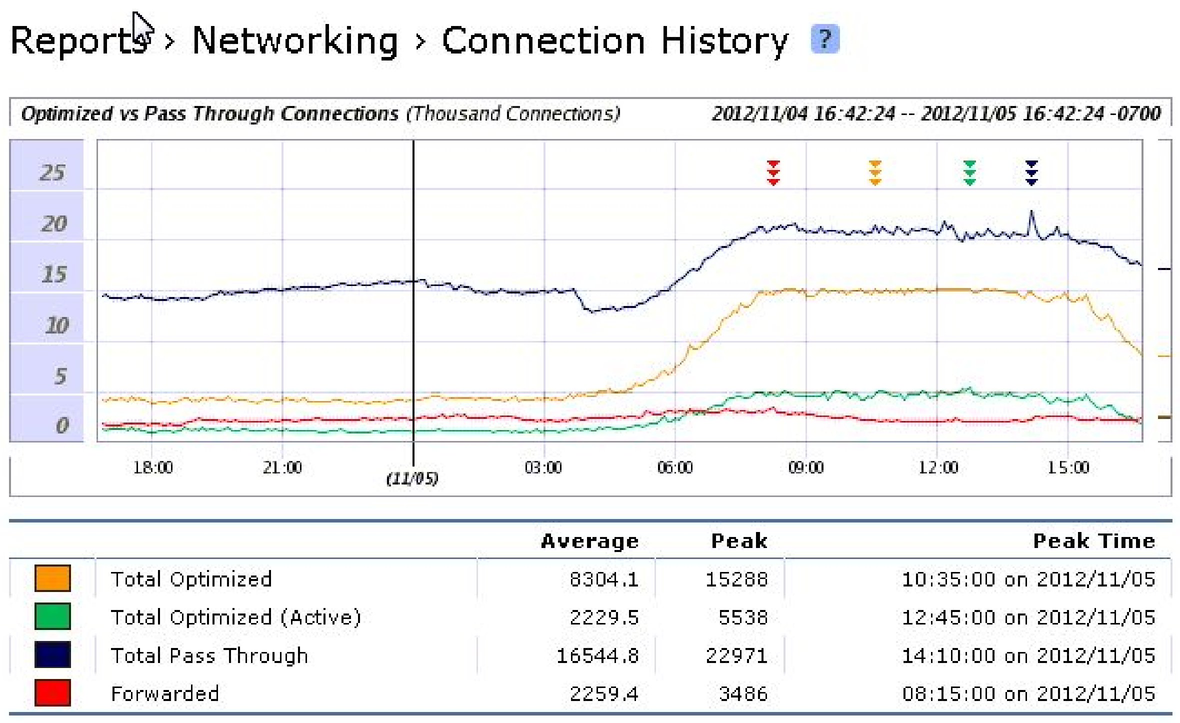
```
sh-4.1# grep /rbt/sport/admission/config/ active-brief.txt
/rbt/sport/admission/config/cpu_util/enable = false (bool)
/rbt/sport/admission/config/cutoff_conn = 260 (uint32)
/rbt/sport/admission/config/cutoff_mem = 1400 (uint32)
/rbt/sport/admission/config/enable_conn = 250 (uint32)
/rbt/sport/admission/config/enable_mem = 1300 (uint32)
/rbt/sport/admission/config/low_mem_ratio = 96 (uint32)
/rbt/sport/admission/config/mapi/enable = false (bool)
/rbt/sport/admission/config/mapi/threshold = 85 (uint32)
/rbt/sport/admission/config/override = false (bool)
/rbt/sport/admission/config/tcp_mem/enable = true (bool)
/rbt/sport/admission/config/unlock/licensed = false (bool)
/rbt/sport/admission/config/vm/dirty_background_ratio = 5 (uint8)
/rbt/sport/admission/config/vm/dirty_expire_centisecs = 800 (uint32)
/rbt/sport/admission/config/vm/dirty_ratio = 6 (uint8)
/rbt/sport/admission/config/vm/dirty_writeback_centisecs = 300 (uint32)
```

5.3.1. Connection-based admission control

Every Steelhead has a licensed maximum number of concurrent TCP connections, for example, the 250H model which has a maximum of concurrent 200 TCP sessions.

In the admission control algorithm, this value is used to define when the interception of new TCP sessions should be enabled again. Interception is disabled when this value plus some spare is reached. This spare amount is to dampen the flapping of the admission control status. Interception is enabled again when the number of optimized TCP drops below this value.

Figure 5.22. Connection-based admission control for a 250H model



		Average	Peak	Peak Time
🟧	Total Optimized	8304.1	15288	10:35:00 on 2012/11/05
🟩	Total Optimized (Active)	2229.5	5538	12:45:00 on 2012/11/05
🟦	Total Pass Through	16544.8	22971	14:10:00 on 2012/11/05
🟥	Forwarded	2259.4	3486	08:15:00 on 2012/11/05

The flat yellow line between 08:00 and 14:00 shows that the device was in connection-based admission control the entire time. Check the Current Connections report to determine which hosts uses most and if their traffic is legit.

5.3.1.1. Determining via the GUI

If the issue is still happening the data in the list of Current Connections can be used to find out if there is a host with an extraordinary large amount of optimized TCP sessions in the list.

5.3.1.2. Determining via the system dump

If a system dump is available, the file connection_table can be used to determine the host with the most TCP sessions:

Figure 5.23. Check the list of TCP sessions to find the host which uses the most

```
[~/systemdump-SH] edwin@t43>awk '{ print $1 }' < connection_table | sed -e 's/:.*//' | sor \
    t | uniq -c | sort -nr | head
 120   10.0.1.6
 112   10.0.1.1
   5   10.0.1.5
   3   10.0.1.8
```

The IP address 192.168.1.6 should be ignored since it is the IP address of an in-path interface. The host with IP address 192.168.1.1 is using 112 TCP sessions.

The next steps would be to find the TCP sessions for that host:

Figure 5.24. Get the list of TCP sessions for one specific host

```
[~/systemdump-SH] edwin@t43>grep 192.168.1.1 connection_table
10.0.1.1:6802 EAL_DEV_LAN 192.168.1.1:80 EAL_DEV_LOCAL CONN_ESTABLISHED 1338909287 0 NONE  \
    0 0 inpath0_0
10.0.1.1:6814 EAL_DEV_LAN 192.168.1.1:80 EAL_DEV_LOCAL CONN_CLOSED 1338909304 0 NONE 0 0 i \
    npath0_0
10.0.1.1:6832 EAL_DEV_LAN 192.168.1.1:80 EAL_DEV_LOCAL CONN_CLOSED 1338909308 0 NONE 0 0 i \
    npath0_0
10.0.1.1:6846 EAL_DEV_LAN 192.168.1.1:80 EAL_DEV_LOCAL CONN_ESTABLISHED 1338909329 0 NONE  \
    0 0 inpath0_0
10.0.1.1:6819 EAL_DEV_LAN 192.168.1.1:80 EAL_DEV_LOCAL CONN_CLOSED 1338909306 0 NONE 0 0 i \
    npath0_0
[...]
```

So for this host most of the traffic went to the host with IP address 192.168.1.1 on TCP port 80.

5.3.2. MAPI connection-based admission control

On the network level, a MAPI session uses multiple TCP sessions to communicate towards the Exchange server. All these TCP sessions should be optimized and go through the same Steelhead appliances, otherwise the latency optimization will not work properly and MAPI connections can be interrupted.

To prevent MAPI sessions from becoming partly optimized, the Steelhead appliance can be configured to stop performing latency optimization for new MAPI sessions when the Steelhead appliance is using 85% of the maximum number of concurrent TCP sessions. Non-MAPI traffic will still be optimized, new TCP sessions from current MAPI sessions will still be optimized, only new TCP sessions from new MAPI sessions will not be optimized anymore.

Figure 5.25. Output of the command "show admission internal"

```
SH (config) # admission control mapi threshold 85
SH (config) # show admission internal
    TCP Memory Enabled:      yes
    CPU Utilization Enabled: no
    MAPI Enabled:            no
    MAPI Threshold:          85
```

5.3.3. Memory-based admission control

During startup, the optimization process will allocate a large chunk of memory for its own internal memory pool. When new optimized TCP sessions get setup, memory from this pool will be allocated for its data-structures and buffers.

There are two situations where this memory-pool can run out of space and the optimization service will go in memory-based admission control:

- The optimization service is servicing so many TCP connections that it uses all the available memory. When the Steelhead appliance constantly enters and leaves memory-based admission control, this is most likely what is happening. If it happens for the first time, the first step would be to create a system dump and open a case with Riverbed TAC.

  The solution would be to reduce the amount of TCP sessions to be optimized via:

  - In-path rules to disable optimization for certain TCP sessions.
  - Additional hardware in a virtual in-path deployment cluster.
  - The increase the capacity of the Steelhead appliance, either by a license upgrade or by a model upgrade.

- The optimization service is allocating memory and never releasing it, this is called a memory-leak. If this happens, the first step would be to create a system dump and open a case with Riverbed TAC, but do not restart the optimization service yet. Unless the cause can be determined from the system dump provided, Riverbed TAC might want to obtain a process dump of the optimization service.

Once Riverbed TAC has all the required information, the immediate solution would be to restart the optimization service, the long term solution would be a software upgrade to a RiOS version where the memory-leak is resolved in.

5.3.4. TCP Memory Pressure

TCP Memory Pressure admission control happens when the optimization service is running out of memory allocated for network buffers.

Figure 5.26. Optimizaiton service going into TCP Memory Pressure based admission control

```
sport[9672]: [admission_control.NOTICE] - {- -} TCP Memory Pressure.
sport[9672]: [admission_control.WARN] - {- -} Pausing intercept: TCP Memory Pressure;
sport[9672]: [admission_control.NOTICE] - {- -} Memory Usage: 3298 Current Connections: 16 \
    55 TCP Memory Usage: 34869;
```

This can happen because of a slow client or server or because of a slow remote Steelhead appliance. The amount of memory used for every TCP session can be found in the file *sysinfo.txt* in the system of the device:

Figure 5.27. Non-zero receive queues and non-zero send queues

```
Output of 'netstat -a --numeric-hosts':"
[...]
Proto Recv-Q Send-Q Local Address           Foreign Address     State
tcp        0 108228 192.168.1.6:23456       192.168.1.1:543     ESTABLISHED
tcp   208322      0 192.168.1.6:7801        10.0.1.6:1040       ESTABLISHED
```

For the non-zero send queue, most likely there will be a large amount of TCP sessions towards a single host. Determining this host and finding out why it is so slow would be the next steps.

For the non-zero receive queue, a case with Riverbed TAC should be opened since this is a slowness towards the optimization service.

5.3.5. Optimized WAN bandwidth limitation

Technically speaking this is not an admission control related issue, the optimized WAN bandwidth limitation is a licensing related feature which will limit the bandwidth of the outgoing optimized traffic. This is not affecting pass-through traffic.

Different models have different optimized WAN bandwidth limitations. For example the 250M model has a limit of 1 Mbps, while the 250H model has a limit of 2 Mbps. Not all the models have capped optimized WAN bandwidth limitations, for example the 2050 model has a 45 Mbps optimized WAN bandwidth speed which is not capped but up to which the model is capable of running.

This limitation is implemented by a 200 milliseconds separated burst of traffic. This trace was taken for a transfer of a 1 Mb file via a 250M model via an optimized TCP session with only TCP optimization and no data reduction:

Figure 5.28. Wireshark I/O graph for a 1 Mbps optimized WAN bandwidth limit

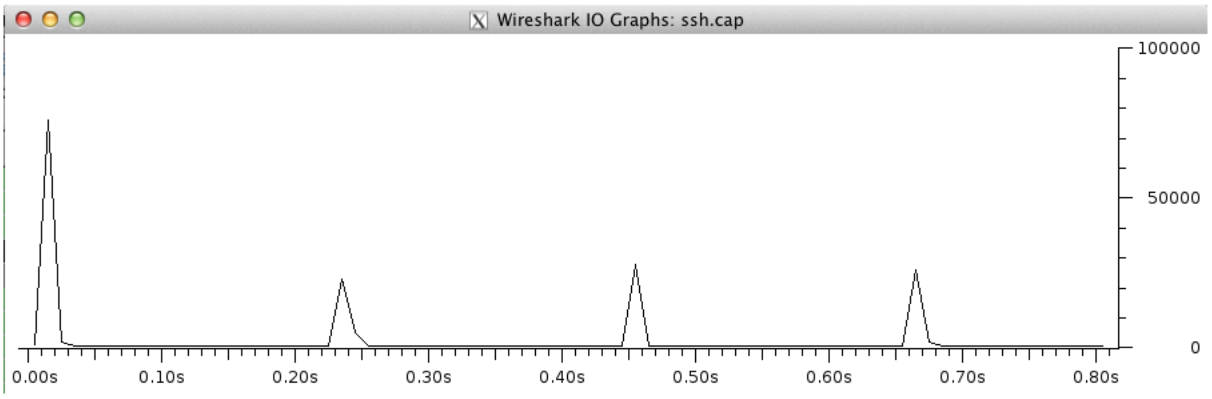
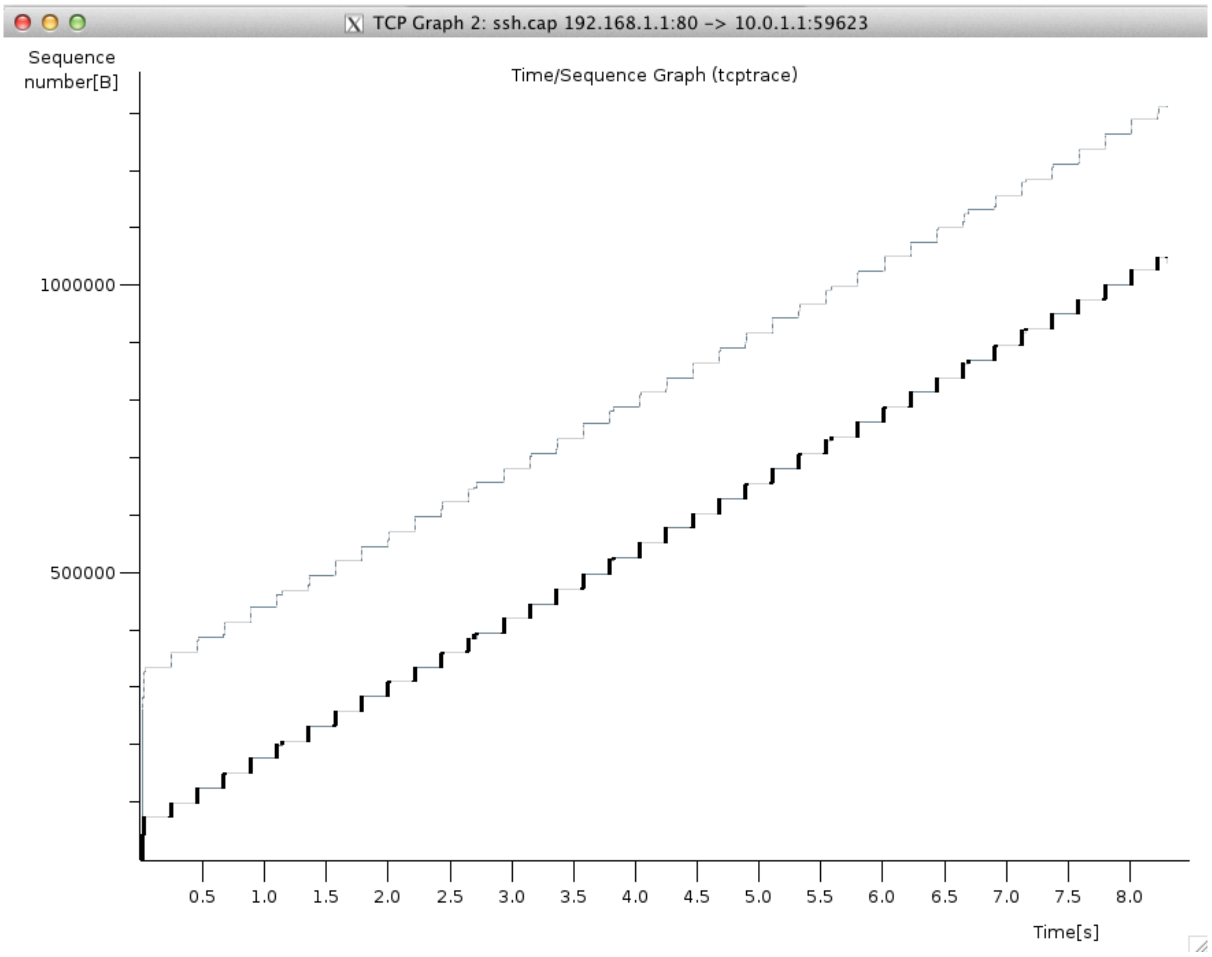
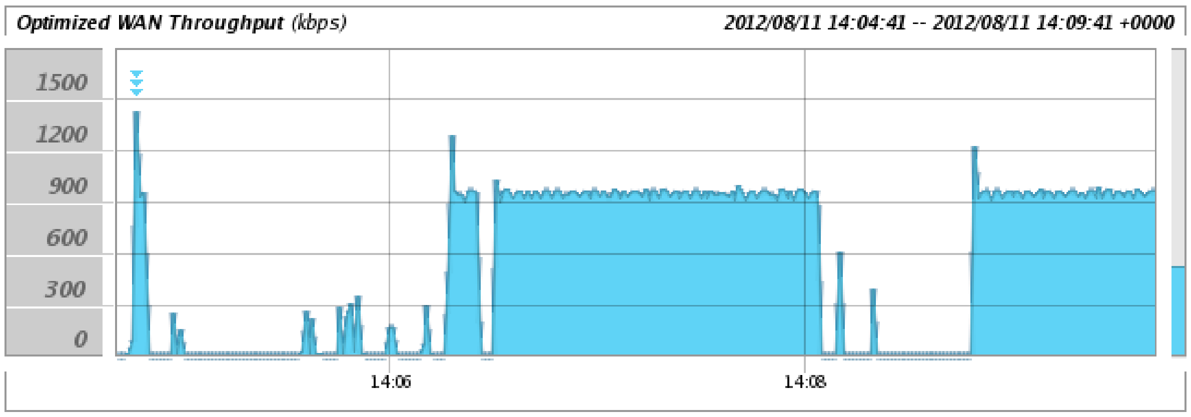


Figure 5.29. Wireshark TCP graph for a 1 Mbps optimized WAN bandwidth limit



It clearly shows the 200 ms interval during which no traffic is send.

Figure 5.30. Optimized Throughput graph for a 1 Mbps optimized WAN bandwidth limit



The Optimized Throughput graph is showing to be pegged at 1 Mbps.

5.4. Auto-negotiation and duplex issues

As discussed before, the interface speed and duplex settings for all Ethernet links on the path between the two Steelhead appliances should configured and negotiated correctly.

The symptoms for an incorrect configured or negotiation can be determined via various ways:

- Ping times to a remote site should be constant and without packet-loss.
- An unoptimized high-speed file transfer should have a consistent throughput.

5.4.1. Ping times

Under normal conditions, the response times for the ping command should be constant. With problems in the interface speed and duplex settings, this response time will fluctuate wildly because of constant problems on this link in the network.

Figure 5.31. Ping through a clean network

```
SH # ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=63 time=147.372 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=63 time=140.338 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=63 time=137.355 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=63 time=137.144 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=63 time=137.219 ms
```

Figure 5.32. Ping through a network with speed/duplex setting issues

```
SH # ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=63 time=303.304 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=63 time=253.271 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=63 time=312.912 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=63 time=264.712 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=63 time=248.469 ms
```

5.4.2. Throughput tests

This can be done on the Steelhead appliances but also on a client and server behind the Steelhead appliances.

These tests should only be done if the data stream is not affected by QoS.

Be aware that this determines the end-to-end packet-loss, not the packet-loss on individual links in the path.

5.4.2.1. Ping flood

A ping flood consists of the client sending out lots of ICMP Echo Request packets and measuring the amount of received ICMP Echo Response packets, making it possible to determine the packet-loss on a link in the network.

Figure 5.33. Ping flood on a relative clean path in the network

```
SH # ping -f -s 1400 -c 1000 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 1400(1428) bytes of data.

--- 192.168.1.1 ping statistics ---
1000 packets transmitted, 1000 received, 0% packet loss, time 13951ms
rtt min/avg/max/mdev = 131.142/133.056/134.008/0.561 ms, pipe 13, ipg/ewma 13.965/133.019  \
    ms
```

Figure 5.34. Ping flood on a path with duplex configuration or negotiation issues.

```
SH # ping -f -s 1400 -c 1000 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 1400(1428) bytes of data.
.......................................................................................... \
    ...................................................................................... \
    ...................................................................................... \
    .............................................
--- 192.168.1.1 ping statistics ---
1000 packets transmitted, 753 received, 24.7% packet loss, time 16701ms
rtt min/avg/max/mdev = 166.241/184.156/208.234/18.631 ms, pipe 13, ipg/ewma 16.701/160.190 \
     ms
```

5.5. LAN speed maxed out

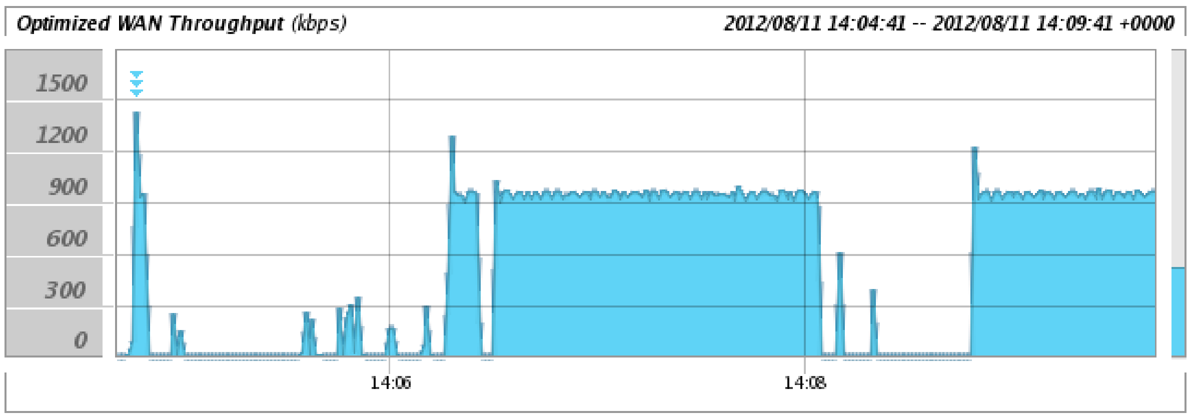
Without WAN optimization, the maximum bandwidth utilization between the WAN router and the LAN switch should not be more than the lowest bandwidth coming going through the WAN.

With WAN optimization, this balance doesn't exist anymore and the LAN interface can become a new bottleneck.

If your WAN bandwidth is 2 Mbps and LAN bandwidth is set to 10 Mbps, then there is a high probability that the post-optimized traffic will fill that 10 Mbps link.

If your WAN bandwidth is 10 Mbps and the LAN bandwidth is set to 100 Mbps, then there is a reasonable probability that the post-optimized traffic will fill that 100 Mbps link.

Figure 5.35. LAN bandwidth is more than 10 Mbps



If the reason for this 100 Mbps limit is a limitation of the switch, then the switch should be replaced.

If the reason for the 10 Mbps or 100 Mbps is because the interface on the WAN router owned by the service provider is set a fixed speed and duplex, then the service provider should be urged to reconfigure the interface on the WAN router from a fixed speed and duplex configuration to an auto-negotiation configuration. This way the LAN interface on the Steelhead appliance can negotiate a gigabit Ethernet link to the LAN switch while it can negotiate a lower speed on the Ethernet link between the WAN interface and the WAN router.

If the service provider cannot do this, a VLAN bridge can be implemented.

5.6. Watchdogs

As described earlier, the Steelhead appliance and the optimization service have several hardware watchdogs and watchdog processes:

- The hardware watchdog, which will reboot the appliance if the user-land process *wdt* has not been communicated with the hardware watchdog for 30 seconds. This prevents the Steelhead appliance to hang indefinitely and gives it a chance to recover.

  This process communicates with the hardware watchdog every six seconds. If it is too late, the following line will be printed:

  Figure 5.36. The wdt process is too slow

  ```
  SH wdt[11472]: [wdt.WARNING]: [lan1_0] watchdog poll late: 6265 with interval 6000
  ```

  The most likely cause is a problem with the hard disks.

- The network watchdog, which will put the by-pass card into fail-to-wire or fail-to-block mode if the optimization service hasn't communicated with the network watchdog for 7 seconds. This will remove the Steelhead appliance from the path in the network and the traffic flows again.

  The optimization service communicates with the network watchdog every second. If it is too late, the following lines will be printed:

  Figure 5.37. The optimization service is too slow

  ```
  SH sport[9367]: [ErWatchdog.NOTICE] - {- -} ether-relay watchdog poll late: 3491 period 10 \
      00
  SH sport[9367]: [null_watchdog.WARN] - {- -} null watchdog poll late: 3127 period 1000
  ```

  The most likely cause is a problem inside the optimization service and a case with Riverbed TAC should be opened.

- The software watchdog, a watchdog in the optimization service which checks if the threads inside the optimization service are still responsive enough or that they are blocked.

  Figure 5.38. The optimization service has hung threads

  ```
  SH sport[5348]: [eventthread/watch/worker/4.WARN] - {- -} watcher: One or more threads not \
       responding after at least 20s; unhealthy threads follow
  SH sport[5348]: [eventthread/watch/worker/4.WARN] - {- -} watcher:    EventThread(main)[LW \
      P 5348] 0x1c1a000 is not healthy
  ```

  The most likely cause is a problem inside the optimization service and a case with Riverbed TAC should be opened.

5.7. Unexpected restarts of the optimization service

There are two ways the optimization service gets aborted:

- Because of an assertion failure. In this situation the operation service encounters a failure because a pre-condition hasn't been met.
- Because of a process failure. In this situation the operation service encounters a failure because of an unknown condition.

In both situations, the Steelhead appliance will generate a process dump. When this happens, open a case with Riverbed TAC and provide them with a system dump. The process dump might be requested later by them.

5.7.1. Clean restart of the optimization service

When the optimization service gets restarted, either via the CLI or via the GUI, the logs will show that the optimization service gets cleanly shutdown:

Figure 5.39. Proper restart of the optimization service

```
SH pm[11256]: [pm.NOTICE]: Terminating process sport
SH sport[19125]: [signal.NOTICE] - {- -} /opt/rbt/bin/sport (pid 19125) received signal 2  \
    (SIGINT)
SH pm[11256]: [pm.NOTICE]: Output from sport: /opt/rbt/bin/sport (pid 19125) received sign \
    al 2 (SIGINT) marking bit
[...]
SH sport[19125]: [sport.NOTICE] - {- -} Sport is shutting down.
SH sport[19125]: [sport.NOTICE] - {- -} Sport has cleanly shutdown.
SH pm[11256]: [pm.NOTICE]: Launched sport with pid 19132
SH sport[19132]: [sport.NOTICE] - {- -} Sport initializing... (main thread pid 19132)
SH sport[19132]: [sport.NOTICE] - {- -} AsyncLogger initialized, backlog 1000, shutdown ti \
    meout 30
SH kernel:  <5>[intercept.NOTICE] Intercept paused since watchdog failed.
SH statsd[14594]: [statsd.NOTICE]: Alarm triggered for rising error for event bypass
[...]
SH sport[19132]: [sport.NOTICE] - {- -} Sport ready.
SH kernel: [intercept.NOTICE] Intercept unpaused , watchdog is normal.
SH wdt[14896]: [wdt.NOTICE]: Hardware watchdog change event.  Hardware Watchdog is now : e \
    nabled
SH statsd[14594]: [statsd.NOTICE]: Alarm triggered for rising clear for event bypass
```

5.7.2. Restart of the optimization service because of an assertion failure

An assertion failure happens when the optimization service experiences a pre-condition which wasn't met. This is put in the code by the developers because they know this is a situation which shouldn't happen and which will cause problems later on.

It can be seen in the logs by the string *received signal SIGABRT* in the log files together with the string *ASSERTION FAILED*. After that a stack trace is captured and the optimization service is restarted.

Figure 5.40. Optimization service restart because of an assertion failure

```
SH sport[20486]: [assert.CRIT] - {- -} ASSERTION FAILED (true == cur_req_->req_.done()) at \
     ../protocol/http/http.cc:3064
SH pm[11899]: [pm.ERR]: Output from sport: /opt/rbt/bin/sport (pid 20486) received signal  \
    6 (SIGABRT) dumping core
SH pm[11899]: [pm.ERR]: Output from sport: /opt/rbt/bin/sport STACK TRACE: 0x2a959c75b0:/l \
    ib64/tls/libpthread.so.0:@0x2a959bb000+0x0000c5b0 0x2a961a026d:/lib64/tls/libc.so.6:gs \
    ignal@0x2a96172000+0x0002e26d 0x2a961a1a6e:/lib64/tls/libc.so.6:abort@0x2a96172000+0x0 \
    002fa6e 0x0084b448:/opt/rbt/bin/sport:@0x400000+0x0044b448 0x011c8682:/opt/rbt/bin/spo \
    rt:_ZN10HttpServer8proc_reqEP7DataBufj13HttpTransType@0x400000+0x00dc8682 0x0117bac7:/ \
    opt/rbt/bin/sport:_ZN8HttpUnit19proc_http_turbo_msgEP7DataBufj@0x400000+0x00d7bac7 0x0 \
    11bfd55:/opt/rbt/bin/sport:_ZN10HttpServer9proc_dataEP7DataBufjb@0x400000+0x00dbfd55 0 \
    x0117b8cf:/opt/rbt/bin/sport:_ZN12HttpConsumer7consumeEP7DataBufj@0x400000+0x00d7b8cf  \
    0x009244b5:/opt/rbt/bin/sport:_ZN7Decoder5flushEv@0x400000+0x005244b5 0x00923a31:/opt/ \
    rbt/bin/sport:_ZN7Decoder25handle_refs_disk_callbackEv@0x400000+0x00523a31 0x00923eaa: \
    /opt/rbt/bin/sport:_ZN7Decoder12handle_eventE11EventSource9EventTypePvS2_@0x400000+0x0 \
    0523eaa 0x009d6e7a:/opt/rbt/bin/sport:_ZN20sp_lookup_b
SH pm[11899]: [pm.WARNING]: Output from sport:  (pid 20486) Force Flushing AsyncLogger, ti \
    mestamps may be incorrect
SH statsd[15655]: [statsd.NOTICE]: statsd/provider handle_session_close 68
SH pm[11899]: [pm.NOTICE]: Process sport (pid 20486) terminated from signal 6 (SIGABRT)
SH pm[11899]: [pm.WARNING]: Exit with code 1 from /sbin/fuser
SH pm[11899]: [pm.WARNING]: Core file found in sport's working directory
SH pm[11899]: [pm.NOTICE]: Waiting 1 seconds before relaunching sport
SH kernel: [intercept.NOTICE] Intercept paused since watchdog failed.
SH statsd[15655]: [statsd.NOTICE]: Alarm triggered for rising error for event bypass
SH sport[21850]: [sport.NOTICE] - {- -} Sport initializing... (main thread pid 21850)
SH pm[11899]: [pm.NOTICE]: Launched sport with pid 21850
```

5.7.3. Restart of the optimization service because of a process failure

A process failure happens when the optimization service experiences a condition which causes the operation system to terminate the optimization service.

It can be seen in the logs by the string *received signal SIGxxx* but without the string *ASSERTION FAILED*. After that a stack trace is captured and the optimization service is restarted.

Figure 5.41. Optimization service restart because of a process failure

```
SH pm[5996]: [pm.ERR]: Output from sport: /opt/rbt/bin/sport (pid 14917) received signal 1 \
    1 (SIGSEGV) dumping core
SH pm[5996]: [pm.ERR]: Output from sport: /opt/rbt/bin/sport STACK TRACE: 0x2a959c75b0:/li \
    b64/tls/libpthread.so.0:@0x2a959bb000+0x0000c5b0 0x012c36df:/opt/rbt/bin/sport:_ZN4Smb \
    212ClientParser22process_CreateResponseEPNS_14CreateResponseE@0x400000+0x00ec36df 0x01 \
    2aadce:/opt/rbt/bin/sport:_ZN4Smb215CompoundRequest17process_responsesEPNS_8Smb2ListI3 \
    PtrINS_12ResponseBaseEEEE@0x400000+0x00eaadce 0x0126b568:/opt/rbt/bin/sport:_ZN4Smb26P \
    arser17process_responsesERKNS_8Smb2ListI3PtrINS_12ResponseBaseEEEE@0x400000+0x00e6b568 \
     0x0126bb1f:/opt/rbt/bin/sport:_ZN4Smb212ClientParser18server_consume_smbEP7DataBufj@0 \
    x400000+0x00e6bb1f 0x00c9e5e9:/opt/rbt/bin/sport:_ZN4Cifs12CifsConsumer18process_data_ \
    queueEv@0x400000+0x0089e5e9 0x00c9e6ef:/opt/rbt/bin/sport:_ZN4Cifs12CifsConsumer15proc \
    ess_consumeEP7DataBufj@0x400000+0x0089e6ef 0x00c9d457:/opt/rbt/bin/sport:_ZNK5boost9fu \
    nction0IvEclEv@0x400000+0x0089d457 0x00c9be93:/opt/rbt/bin/sport:_ZN4Cifs10CifsParser9 \
    add_eventERKN5boost9function0IvEE@0x400000+0x0089be93
SH pm[5996]: [pm.WARNING]: Output from sport:  (pid 14917) Force Flushing AsyncLogger, tim \
    estamps may be incorrect
SH statsd[9264]: [statsd.NOTICE]: statsd/provider handle_session_close 4
SH pm[5996]: [pm.NOTICE]: Process sport (pid 14917) terminated from signal 11 (SIGSEGV)
SH pm[5996]: [pm.WARNING]: Exit with code 1 from /sbin/fuser
SH pm[5996]: [pm.WARNING]: Core file found in sport's working directory
SH pm[5996]: [pm.NOTICE]: Waiting 1 seconds before relaunching sport
SH mgmtd[5997]: [mgmtd.NOTICE]: Sending email to event recipients
SH kernel: [intercept.NOTICE] Intercept paused since watchdog failed.
SH statsd[9264]: [statsd.NOTICE]: Alarm triggered for rising error for event bypass
SH mgmtd[5997]: [mgmtd.NOTICE]: Sending email to event recipients
SH sport[11613]: [sport.NOTICE] - {- -} Sport initializing... (main thread pid 11613)
```

5.8. Unexpected reboots

Unexpected reboots can happen for three reasons: Kernel panics, power failures and hardware watchdog initiated reboots.

5.8.1. Kernel panics

The optimization service runs on top of a Linux kernel. When there is a problem in the kernel, it will panic and reboot the machine. Problems in the kernel can be either caused by software (for example a reference to a invalid block of memory or a divide by zero) or by hardware (errors in the memory-sticks). When this happens, the kernel will try to create a crash dump and reboot. During startup the startup scripts will detect these crash dumps and extract the necessary information from it.

Figure 5.42. Reboot caused by a kernel panic

```
localhost kernel: con_dump: restoring oops message, timestamp=1351785953 ---------
localhost kernel: divide error: 0000 [1] SMP
localhost kernel: CPU 0
localhost kernel: Pid: 0, comm: swapper Tainted: PF     2.6.9-34.EL-rbt-11902SMP
localhost kernel: RIP: 0010:[tcp_snack_new_ofo_skb+356/684] <ffffffff8045d302>{tcp_snack_n \
    ew_ofo_skb+356}
localhost kernel: RIP: 0010:[<ffffffff8045d302>] <ffffffff8045d302>{tcp_snack_new_ofo_skb+ \
    356}
localhost kernel: RSP: 0018:ffffffff806b4cc8  EFLAGS: 00010246
localhost kernel: RAX: 0000000000000551 RBX: 00000100755d5338 RCX: 0000000000000000
localhost kernel: RDX: 0000000000000000 RSI: 00000101286f7040 RDI: 00000100755d5338
localhost kernel: RBP: 00000100755d5000 R08: ffffffff01000000 R09: 00000101286f7040
localhost kernel: R10: 0000000000000000 R11: deaf1eed01000000 R12: 0000010074b5c034
localhost kernel: R13: 00000101286f7040 R14: 0000000000000295 R15: 0000000000000000
localhost kernel: FS:  0000000040401960(0000) GS:ffffffff80753e00(0000) knlGS:000000000000 \
    0000
localhost kernel: CS:  0010 DS: 0018 ES: 0018 CR0: 000000008005003b
localhost kernel: CR2: 0000002a98215000 CR3: 0000000000101000 CR4: 00000000000006e0
localhost kernel: Process swapper (pid: 0, threadinfo ffffffff80758000, task ffffffff805d9 \
    480)
localhost kernel: Stack: 00000100755d5338 ffffffff80460dc2 0000000000000001 00000100755d50 \
    00
localhost kernel:        0000000000000003 000001012a83c800 0000000000000000 000001012a83c8 \
    00
localhost kernel:        00000000ba098a8f ffffffff8046e25e
localhost kernel: Call Trace:<IRQ> <ffffffff80460dc2>{tcp_rcv_state_process+2471} <ffffff \
    f8046e25e>{tcp_child_process+51}
localhost kernel:        <ffffffff8046a72a>{tcp_v4_do_rcv+391} <ffffffff8046ad4a>{tcp_v4_r \
    cv+1449}
localhost kernel:        <ffffffff8044ad04>{ip_local_deliver_finish+248} <ffffffff8044ac0c \
    >{ip_local_deliver_finish+0}
localhost kernel:        <ffffffff8043e7a2>{nf_hook_slow+184} <ffffffff8044b007>{ip_local_ \
    deliver+552}
localhost kernel:        <ffffffff8044b20e>{ip_rcv_finish+512} <ffffffff8044b00e>{ip_rcv_f \
    inish+0}
localhost kernel:        <ffffffff8043e7a2>{nf_hook_slow+184} <ffffffff8044b6d7>{ip_rcv+11 \
    38}
localhost kernel:        <ffffffff80435528>{netif_receive_skb+590} <ffffffff804355eb>{proc \
    ess_backlog+137}
localhost kernel:        <ffffffff804356f5>{net_rx_action+129} <ffffffff8013ac80>{__do_sof \
    tirq+88}
localhost kernel:        <ffffffff8013ad29>{do_softirq+49} <ffffffff8011311f>{do_IRQ+328}
localhost kernel:        <ffffffff801107cb>{ret_from_intr+0}  <EOI> <ffffffff801
localhost kernel: con_dump: end of oops -----------------------------------
```

When this happens, please open a case with Riverbed TAC for follow-up.

In RiOS version 8.0 and later, there is the feature of a crash kernel available: When a kernel panic happens, instead of that the Steelhead appliance gets rebooted and the memory contents are lost, the Steelhead appliance will reboot into a second kernel and is able to collect all the memory contents for debugging later by Riverbed engineering.

Figure 5.43. Enabling the crash kernel feature

```
SH (config) # support show kexec
Kexec Mode Enabled: no
SH (config) # support kexec enable
You must reboot the appliance for your changes to take effect
SH (config) # write memory
SH (config) # reload
```

When this crash kernel feature is enabled, a kernel crash will take longer than normal. Afterwards a file named *kernel-crashdump-<timestamp>.tgz* will be avalable in the process dump directory.

5.8.2. Power failure caused reboots

Power failure is either when the power to the power supplies fails or when a power supply fails. On the xx20 series models without a hardware RAID controller and on the xx50, CX and EX series models, this kind of reboots can be identified by checking the SMART status of the hard disks. In case of a power off/on, the value of the *Current Power Cycle Count* attributes gets increased while in case of a reboot it doesn't get increased.

So by by searching for the lines *Current Power Cycle Count* and *shutdown_check* in the startup logs, the kind of reboot can be determined:

Figure 5.44. This device was rebooted because of loss of power.

```
localhost disk: Drive=sda, Serial= 080806DP1D10DGGHV06P, Current Power Cycle Count=68, Las \
    t Power Cycle Count=66
[...]
localhost shutdown_check: Checking for unexpected shutdown
localhost shutdown_check: Detected unexpected shutdown!
localhost shutdown_check:
localhost rc: Starting shutdown_check:  succeeded
```

In this case the *Current Power Cycle Count* was increased: This was a power related restart.

5.8.3. Hardware watchdog initiated reboots

The hardware watchdog is integrated on the motherboard and which will reboot the appliance if the process *wdt* has not been communicated with the hardware watchdog for 30 seconds.

This can happen when:

- The kernel hangs and doesn't allow the user-land to run. This is a bad situation and the reboot will give the machine a chance to recover, hopefully without getting into the same situation again.
- The kernel is waiting for a hard disk to complete a read or write operation and it takes too long. This can happen when a hard disk is in the process of failing but its controller hasn't given up yet.

  For non-RAID or non-FTS appliances, this might cause the Steelhead appliance to not come back since an essential hard disk has malfunctioned.

  For RAID-appliances, this might be the first step for the RAID controller, RAID software or FTS process to determine that a disk is broken.

  Figure 5.45. The watchdog interval was larger than expected.

  ```
  SH wdt[6805]: [wdt.WARNING]: watchdog poll late: 16415 with interval 1000
  ```

  In this example, the watchdog timer had been activated 16.415 seconds after the last time instead if the expected one second.

Unlike the previous section, where the line *Current Power Cycle Count* had a different value for the current and last, a hardware watchdog initiated reboot does not increase that value.

Figure 5.46. This device was rebooted by the hardware watchdog.

```
localhost disk: Drive=sda, Serial= 080806DP1D10DGGHV06P, Current Power Cycle Count=68, Las \
    t Power Cycle Count=68
[...]
localhost shutdown_check: Checking for unexpected shutdown
localhost shutdown_check: Detected unexpected shutdown!
localhost shutdown_check:
localhost rc: Starting shutdown_check:  succeeded
```

In the file *hardwareinfo.txt*, which is found in a system dump and discussed in the section Hardware Information, the hardware watchdog initiated reboots can be seen in the IPMI log:

Figure 5.47. Determining reboots based on the IPMI logs

```
Output of 'show_ipmi_info':

Motherboard '400-00100-01'

SEL Record ID          : 0005
 Record Type           : 02
 Timestamp             : 04/06/2011 15:56:54
 Generator ID          : 0020
 EvM Revision          : 04
 Sensor Type           : Critical Interrupt
 Sensor Number         : e7
 Event Type            : Sensor-specific Discrete
 Event Direction       : Assertion Event
 Event Data            : 03ffff
 Description           : Software NMI

SEL Record ID          : 0157
 Record Type           : 02
 Timestamp             : 04/06/2011 15:56:55
 Generator ID          : 0020
 EvM Revision          : 04
 Sensor Type           : Watchdog 2
 Sensor Number         : 81
 Event Type            : Sensor-specific Discrete
 Event Direction       : Assertion Event
 Event Data            : c104ff
 Description           : Hard reset
```

Since RiOS version 7.0 a software NMI handler is called before the hardware watchdog will reboot the device. This will be used to save a copy of the kernel stack which can be retrieved in the next startup of the device.

When the hardware watchdog resets the Steelhead appliance, please open a case with Riverbed TAC for investigation.

5.9. Downgrade and upgrade issues

The upgrade and downgrade process on Steelhead appliances is not always straightforward. The following restrictions and caveats are in place:

- You can only downgrade to RiOS versions which you have had installed on the Steelhead appliance. So if a certain Steelhead appliance comes installed with RiOS 5.0.6 and you have upgraded via 5.5.3, 6.0.1, 6.1.0 to 6.5.0, then you can only downgrade to these four versions.

  Use the command `show version history` to see the list of previous installed RiOS versions.

Figure 5.48. Output of "show version history": Upgrade to 6.1.5, then downgrade to 6.1.4

```
SH # show version history
Firstboot to rbt_sh 6.1.4 #92_4 2011-06-08 15:54:25 x86_64 root@palermo0:svn://svn/mgmt/br \
    anches/cook_92_fix_branch on Tue Dec  6 05:33:28 GMT 2011
Upgraded to rbt_sh 6.1.5 #104_8 2011-09-29 15:26:28 x86_64 root@athens:svn://svn/mgmt/bran \
    ches/cook_104_fix_branch on Wed Feb  1 23:30:34 GMT 2012
Downgraded to rbt_sh 6.1.4 #92_4 2011-06-08 15:54:25 x86_64 root@palermo0:svn://svn/mgmt/b \
    ranches/cook_92_fix_branch on Thu Feb  1 23:33:28 GMT 2012
```

- When a Steelhead appliance gets upgraded, the latest configuration file used on that software version will be saved. When a Steelhead appliance gets downgraded, it will restart from that saved configuration. So if network configurations like the IP address of the in-path interface have been changed, after the move to the current version, that IP address will be changed back to the old IP address.

- A Steelhead appliance can only be upgraded to a software version which is released later, time wise, than the current one. So an upgrade from 5.5.9 to 6.0.0 is not allowed because 5.5.9 had been released later than 6.0.0 had been released.

- In general, when upgrading a Steelhead appliance to a new branch, for example from 5.0.x to 6.1.x, all branches should be touched. So the Steelhead appliance needs to be upgraded to 5.5.x and 6.0.x before the upgrade to 6.1.x can be done.

  The *Software Upgrade Path Tool* on the Riverbed Support website will tell which intermediate software versions can be used to upgrade between two branches.

  There are several RiOS releases which are certified for direct upgrades, which mean that they can be upgraded to a different branch than the next one. To find out which ones, search the Riverbed Support KB system for "RiOS Direct Upgrade".

- When the Steelhead appliance is running RSP, the recommendation is to disable RSP before doing the upgrades and to install the correct version of RSP image at the end. After that RSP can be enabled again.

The steps in the upgrade process are the installation of the upgrade image on the spare boot partition and, after a reboot into the new boot partition, the upgrade of the current configuration into the new software is performed.

5.9.1. Upgrade via the CLI

To perform the RiOS upgrade from the CLI, issue the following commands. This example assumes that the image can be retrieved from the web server images.example.com.

Figure 5.49. Manual upgrade via the CLI, new image installed on partition 1

```
SH # show images
[...]
Last boot partition: 2
Next boot partition: 2
SH # image fetch http://images.example.com/rios/sh-5.5.3-i386.img
SH # image install sh-5.5.3-i386.img 1
SH # config term
SH (config) # image boot 1
SH (config) # write memory
SH (config) # reload
```

5.9.2. Force a downgrade

If a downgrade is not possible because the downgrade image has never been installed before, it will fail with the following error:

Figure 5.50. Downgrade was unsuccessful

```
localhost mgmtd[6353]: [mgmtd.NOTICE]: Downgrade fallback in progress.
localhost mgmtd[6353]: [mgmtd.NOTICE]: Downgrade fallback file /config/db/working-PROD-rbt \
    sh-VER-54b74c06 doesn't exist. Looking for the most recent file with version hash 54b7 \
    4c06.
localhost mgmtd[6353]: [mgmtd.ERR]: db & reg version hash mismatch, could not fallback eit \
    her. possibly due to module removed without upgrade rule
localhost mgmtd[6353]: [mgmtd.NOTICE]: Downgrade: could not find file for working with ver \
    sion hash 54b74c06 to downgrade to.
localhost mgmtd[6353]: [mgmtd.WARNING]: Configuration 'working' is damaged!
localhost mgmtd[6353]: [mgmtd.NOTICE]: mgmtd exiting at 2013/11/12 22:23:23 with code 1401 \
    0
localhost mdinit: failed.
localhost mdinit: Forcing reboot from fallback image 1
```

To force a downgrade to a RiOS version which has not been installed before, use the commands `image install <image> <partition> force` and `image boot <partition> force`. This will wipe the configuration including the license keys, so make sure to keep a copy of them. Also make sure to clear the data store afterwards, because a proper downgrade has not been performed.

The *force* in the `image install` command skips the check to see if the image to be installed is newer than the current image during the installation phase.

The *force* in the `image boot` command skips the check to see there is a saved configuration available for the new version during the downgrade phase.

5.9.3. New features after an upgrade

After an upgrade, new features are often disabled by default. Because the configuration displayed with the command `show running-config` doesn't show the configuration lines which still have their default values, the new features will often not show up. If during a later upgrade that feature becomes enabled by default, the configuration on the Steelhead appliance will show this feature to be disabled.

Table 5.1. Example of configuration changes for the Simplified Routing feature for the value "none"

RiOS version	Default value	Configured value	Displayed value
5.5	none	none	(nothing)
6.0	dest-only	none	in-path simplified routing "none"

Table 5.2. Example of configuration changes for the Simplified Routing feature for the value "dest-only"

RiOS version	Default value	Configured value	Displayed value
5.5	none	dest-only	in-path simplified routing "dest-only"
6.0	dest-only	dest-only	(nothing)

Table 5.3. Example of configuration changes for the Simplified Routing feature for the value "all"

RiOS version	Default value	Configured value	Displayed value
5.5	none	all	in-path simplified routing "all"
6.0	dest-only	all	in-path simplified routing "all"

5.9.4. Failures during the installation phase

An installation failure will be logged in the system logs:

Figure 5.51. Various upgrade failure scenarios

```
SH mgmtd[4229]: [mgmtd.WARNING]: Image install failed. -- The upgrade image provided is in \
    compatible with the architecture of this appliance's hardware.  Please provide an imag \
    e for the x86_64 architecture.
```

The following failures are possible:

- *The upgrade image provided is incompatible with this appliance*, which happens when a non-Steelhead image is being installed on the Steelhead appliance.
- *The upgrade image provided does not support this model* or *The upgrade image provided is incompatible with the architecture of this appliance's hardware. Please provide an image for the x86_64 architecture* or *The upgrade image provided is incompatible with the architecture of this appliance's hardware. Please provide an image for the i386 architecture*, which happens when the provided upgrade image has the wrong CPU architecture, 32-bit instead of 64-bit or the other way around.
- *The upgrade image provided is too old. Please use a newer image. See log for more details*, which happens when the upgrade image doesn't know about this model Steelhead appliance yet.
- *Unable to partition the disk to install image. Please retry, contact support if problem persists* or *Unable to create filesystem for upgrade image. Please retry, contact support if problem persists* or *Cannot find the image layout for machine. Please retry, contact support if problem persists* or *Partitions used to upgrade image are already mounted. Please unmount partitions manually before retrying* or *Could not get the upgrade image. Please check the url if using a remote image*, which happens when there is a problem during the installation with one of the partitions on the Steelhead appliance.
- *The upgrade image provided does not pass validation. Please check the to make sure the correct image is used for upgrade*, which happens when there is a problem with extracting the image. Use the command `show images checksum` to compare the MD5 checksum of the upgrade image with the MD5 checksums found on the Riverbed Support website.
- *Error mounting tmpfs to upgrade image. Please contact support*, which happens when the mounting of a installation partition fails.
- *No space left on disk to install image. Please contact support*, which happens when the target installation partition is full.

5.9.5. Upgrade failed, CLI doesn't become available anymore

If for a certain reason the upgrade was successful and the Steelhead appliance comes back but doesn't allow CLI or GUI access: This means that a back out of the upgrade cannot be performed.

The way out of this would be to hook up a serial console to the Steelhead appliance and reboot the device. After the BIOS, the Grub bootloader will show up with the text `Press any key to continue...`. After pressing a key, the two boot partitions are displayed. Use the *v* and the ^ to select the pre-upgrade boot image and press enter. After the downgrade, access to the Steelhead appliance should be available again and the Riverbed TAC should be involved to determine why after the upgrade the connectivity failed.

5.10. Time related issues - The NTP service.

Having all your network devices operate on a synchronized time is essential for reporting and troubleshooting. Further, to achieve Active Directory integration, the clock on the Steelhead appliance needs to be in sync with the clock on the Domain Controllers.

By default the Steelhead appliance has the NTP service enabled against a couple of NTP clocks on the public Internet. If the Steelhead appliances do not have access to the public Internet and the network does not have an internal NTP server, then the Steelhead appliances should sync to the clock on the Domain Controllers.

5.10.1. Configuration of the NTP service

In the configuration of the NTP service you specify a number of NTP servers.

Note that using the hostnames from the riverbed.pool.ntp.org pool of NTP servers returns geographically local IP addresses which might not match the IP addresses seen on other return statuses.

Figure 5.52. The standard NTP service configuration

```
SH # show running
[...]
no ntp disable
   ntp server ftp.riverbed.com enable
   ntp server ftp.riverbed.com version "4"
   ntp server 0.riverbed.pool.ntp.org enable
   ntp server 0.riverbed.pool.ntp.org version "4"
   ntp server 1.riverbed.pool.ntp.org enable
   ntp server 1.riverbed.pool.ntp.org version "4"
   ntp server 2.riverbed.pool.ntp.org enable
   ntp server 2.riverbed.pool.ntp.org version "4"
   ntp server 3.riverbed.pool.ntp.org enable
   ntp server 3.riverbed.pool.ntp.org version "4"
[...]

SH # show ntp
NTP enabled: yes
No NTP peers configured.
NTP server: 208.70.196.25 (version 4)   Enabled: yes
NTP server: 121.0.0.42 (version 4)      Enabled: yes
NTP server: 202.60.94.11 (version 4)    Enabled: yes
NTP server: 203.23.237.200 (version 4)  Enabled: yes
NTP server: 203.192.179.98 (version 4)  Enabled: yes
```

5.10.2. Determining the status of the NTP service

In RiOS version 6.5 and later the status of the NTP service can be seen with the command `show ntp status`:

Figure 5.53. Status of the NTP service with the command "show ntp status"

```
SH # show ntp status
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 ntp.tourism.wa. 130.95.179.80    2 u   13   64    1  1702.52  667.275   0.002
 ns.creativecont .STEP.          16 u  985   64    0    0.000    0.000   0.002
+cachens2.onqnet 209.81.9.7       2 u    6   64    1  772.591  218.753   0.002
+ds001.hostingsh 203.12.160.2     3 u  173   64  374  105.205   40.096  35.771
+wireless.org.au 17.83.253.7      3 u  173   64  374  111.738   12.724  35.467
*ftp.riverbed.co 203.161.12.165   3 u  172   64  374  114.232   -6.295  56.603
```

The NTP service determines the best NTP server based on the stratum (number of hops towards a reference clock), the delay (the network delay towards the server), the offset between the received time of the NTP server and the clock on the local machine and the network jitter.

A stratum of 16 means that the NTP server is unsynchronized.

Instead of IP addresses, the refid can also contain the name of the clock source, like *.GPS.* and *.CDMA.*.

The character in front of the remote clock explains how the NTP service feels about it: A * means that this NTP server is the chosen peer to sync from, a + means that this NTP server is a backup in case the chosen peer becomes unreachable.

5.10.3. Troubleshooting

5.10.3.1. Expected behaviour

Use the tcpdump tool to see if the NTP service is talking to the NTP servers and if they answer back. NTP traffic is UDP based and on both port 123 for the client and the server.

Figure 5.54. Tcpdump trace for NTP traffic

```
SH # tcpdump -ni primary port ntp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on primary, link-type EN10MB (Ethernet), capture size 96 bytes
08:43:03.149487 IP 119.12.133.239.123 > 202.158.218.239.123: NTPv4, Client, length 48
08:43:05.721339 IP 202.158.218.239.123 > 119.12.133.239.123: NTPv4, Server, length 48
08:43:14.632504 IP 119.12.133.239.123 > 192.189.54.17.123: NTPv4, Client, length 48
08:43:14.785459 IP 192.189.54.17.123 > 119.12.133.239.123: NTPv4, Server, length 48
08:43:15.631914 IP 119.12.133.239.123 > 119.148.74.66.123: NTPv4, Client, length 48
08:43:16.632304 IP 119.12.133.239.123 > 119.63.208.27.123: NTPv4, Client, length 48
08:43:16.632433 IP 119.12.133.239.123 > 192.189.54.17.123: NTPv4, Client, length 48
08:43:17.632670 IP 119.12.133.239.123 > 119.148.74.66.123: NTPv4, Client, length 48
08:43:18.440873 IP 119.63.208.27.123 > 119.12.133.239.123: NTPv4, Server, length 48
08:43:18.441874 IP 192.189.54.17.123 > 119.12.133.239.123: NTPv4, Server, length 48
08:43:18.441971 IP 119.148.74.66.123 > 119.12.133.239.123: NTPv4, Server, length 49
```

Both client and server traffic is seen here, which means that the NTP server is reachable and the Steelhead appliance is allowed to talk to its NTP service.

5.10.3.2. No answer from the NTP service

If the NTP server doesn't exist anymore or is configured to not respond to the queries, no answers will be returned:

Figure 5.55. Tcpdump trace for NTP traffic

```
SH # tcpdump -ni primary port ntp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on primary, link-type EN10MB (Ethernet), capture size 96 bytes
08:43:03.149487 IP 119.12.133.239.123 > 202.158.218.239.123: NTPv4, Client, length 48
08:43:18.494871 IP 119.12.133.239.123 > 202.158.218.239.123: NTPv4, Client, length 48
08:43:33.948712 IP 119.12.133.239.123 > 202.158.218.239.123: NTPv4, Client, length 48
08:43:48.487123 IP 119.12.133.239.123 > 202.158.218.239.123: NTPv4, Client, length 48
```

Only client traffic is seen here, which means that the NTP server is either unreachable or that the Steelhead appliance is not allowed to talk to this NTP server.

5.10.3.3. Time offset is too big

If the offset between the reference clock and the system clock is too big, more than 30 minutes, then the NTP service will refuse to synchronize and exit.

To overcome this problem, either disable the NTP service, set the clock of the Steelhead appliance with the command `clock set '<yyyy/mm/dd hh:mm:ss>'` to a value close to the reference clock and then enable the NTP service again. Or disable the NTP service, issue the command `ntpdate <NTP server>` and restart the NTP service.

Figure 5.56. Manual adjusting the date/time with the command "ntpdate"

```
SH (config) # show clock
Time: 12:12:20
Date: 2012/12/12
Zone: Australia Sydney
No active ntp peers
SH (config) # show ntp active-peers
No active ntp peers
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 static-96-226-1 139.78.135.14    2 u    2   64    1  209.560  3923152   0.000
 xen1.rack911.co 209.51.161.238   2 u    2   64    1  177.707  3923152   0.000
 lithium.constan 128.4.1.1        2 u    8   64    1  224.869  3923152   0.000
 240.140.8.72.in 64.147.116.229   2 u    7   64    1  172.970  3923152   0.000

     remote       conf  auth  key
===================================
 static-96-226-1  yes   none  none
 xen1.rack911.co  yes   none  none
 lithium.constan  yes   none  none
 240.140.8.72.in  yes   none  none

SH (config) # no ntp enable
SH (config) # ntpdate 0.riverbed.pool.ntp.org
26 Jan 21:59:13 ntpdate[12761]: step time server 208.68.36.196 offset 3923152.813162 sec
SH (config) # show clock
Time: 21:59:16
Date: 2013/01/26
Zone: Australia Sydney
No active ntp peers
SH (config) # ntp enable
SH (config) # show ntp active-peers
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
-shed.galexander 204.163.51.41    3 u    9   64   37  168.203   -3.754   0.433
+xen1.rack911.co 209.51.161.238   2 u    4   64   37  177.716    5.831   0.301
+ec2-50-16-231-1 209.51.161.238   2 u    9   64   37  218.954    3.933   0.577
*lttleman.deekay 130.207.244.240  2 u    9   64   37  223.387    6.205   1.380

     remote       conf  auth  key
===================================
 shed.galexander  yes   none  none
 xen1.rack911.co  yes   none  none
 ec2-50-16-231-1  yes   none  none
 lttleman.deekay  yes   none  none
```

At the beginning the offset is nearly 4 million seconds, or 45 days. After running the command `ntpdate 0.riverbed.pool.ntp.org` the clock is correct again and after a couple of minutes the NTP service shows up as synchronized again.

5.11. Firewalls in the path

From a network point of view, firewalls are routers with a stricter set of policies than normal routers. Based on their configured policies, they can interfere with the WAN optimization.

5.11.1. Stripping of TCP options

The TCP auto-discovery option and the TCP transparency option are not officially registered TCP options with the IANA [SOURCE http://www.iana.org/assignments/tcp-parameters/tcp-parameters.txt] and therefore firewalls can chose not to trust them by default and remove them from the TCP header.

When the firewall disallows the Auto-Discovery TCP options and it is stripped away from the SYN+ packet, the auto-discovery will fail but an unoptimized TCP session will be setup as normal.

When the firewall disallows the WAN visibility TCP options and it is stripped away by firewalls, the setup of the inner-channel will fail and after the third TCP SYN packet from the client an unoptimized TCP session will be setup as normal.

When the TCP SYN packet with the auto-discovery probe is blocked, after the third TCP SYN packet from the client an unoptimized TCP session will be setup as normal.

Figure 5.57. The TCP session goes into pass-through after the third SYN from the client

```
SH kernel: [intercept.NOTICE] nat_check: SYN packet for already natted connection 10.0.1.1 \
    :56923 -> 192.168.1.1:80 ==> 10.0.1.1:56923 -> 10.0.1.5:7801
```

5.11.2. Stateful firewalls

A firewall might check state of the TCP session. The auto-discovery of the optimized TCP session consists of a SYN+ and one or more SYN/ACK+s, but no final ACK is seen. The firewall might send a TCP RST packet after a timeout to reset the TCP connection on the client and server because of this missing final TCP ACK.

The sequence numbers of the TCP SYN packets seen during the auto-discovery and the setup of the inner channel with a Full Transparency WAN visibility differs: First the firewall sees the SYN+ packet with sequence number X, a moment later it sees a TCP SYN packet with the same IP addresses and TCP ports although with transparency options with sequence number Y. A firewall might block that packet or reset the TCP session. The way around this would be using the Full Transparency with Firewall Reset WAN visibility.

In this example, the auto-discovery happened in the first three packets and then the client-side Steelhead sends the TCP RST in packet 4. After that the inner channel gets setup in packet 5:

Figure 5.58. Setup of an inner channel with Full Transparency with Firewall Reset WAN visibility

```
22:37:34.613669 IP 10.0.1.1.23090 > 192.168.1.1.80: Flags [S], seq 1402966182, win 65535,  \
    options [mss 1460,nop,wscale 6,sackOK,TS val 3240829280 ecr 0,rvbd-probe AD CSH:10.0.1 \
    .6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
22:37:34.913093 IP 192.168.1.1.80 > 10.0.1.1.23090: Flags [S.], seq 20020520, ack 14029661 \
    83, win 65535, options [rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
22:37:34.918955 IP 192.168.1.1.80 > 10.0.1.1.23090: Flags [S.], seq 20020520, ack 14029661 \
    83, win 65535, options [rvbd-probe AD CSH:10.0.1.6 SSH:192.168.1.6:7800 11110a000106c0 \
    a801061e78,rvbd-probe EAD 0e3d,nop,eol], length 0
22:37:34.920476 IP 10.0.1.1.23090 > 192.168.1.1.80: Flags [R], seq 1402966183, win 5840, o \
    ptions [rvbd-trans Transp sSH:10.0.1.6:40296 dSH:192.168.1.6:7800 01000a000106c0a80106 \
    9d681e78,nop,nop,nop,eol], length 0
22:37:34.920582 IP 10.0.1.1.23090 > 192.168.1.1.80: Flags [S], seq 1335892636, win 5840, o \
    ptions [mss 1460,sackOK,TS val 3241303025 ecr 0,nop,wscale 2,rvbd-trans Transp sSH:10. \
    0.1.6:40296 dSH:192.168.1.6:7800 00000a000106c0a801069d681e78], length 0
22:37:35.223347 IP 192.168.1.1.80 > 10.0.1.1.23090: Flags [S.], seq 1345286799, ack 133589 \
    2637, win 5792, options [mss 1460,sackOK,TS val 3241350970 ecr 3241303025,nop,wscale 2 \
    ,rvbd-trans Transp sSH:192.168.1.6:7800 dSH:10.0.1.6:40296 0000c0a801060a0001061e789d6 \
    8], length 0
```

5.11.3. Deep Packet Inspection

This is a problem for inner channels with Port Transparency or Full Transparency WAN visibility. When a DPI engine determines the protocol based on the TCP port number of the server and expects a certain protocol to be followed but gets the content of the inner channel which doesn't make sense for it: It might block or reset the TCP connection.

5.11.4. Packet direction

Firewalls can have the policy that when an IP packet comes in via a certain network interface, it should not be forwarded back out via that network interface. If an in-path interface of a Steelhead appliance has its default gateway set to a firewall and the firewall enforces this policy and there are multiple IP subnets behind the Steelhead appliance, then the firewall will block traffic to these IP subnets. The way around this is to set the default gateway to a routing host on the LAN side and use Simplified Routing.

5.12. Data Store Synchronization

Data Store Synchronization is a technique which can be used for two Steelhead appliances to synchronize their data stores so that both Steelhead appliances have the same references known in their data store. This not a feature specific for serial clusters or high-availability clusters, but should also be used for parallel clusters and Interceptor / WCCP clusters.

Figure 5.59. Data Store Synchronization can be used in various designs

```
 .----------.               .----------.          .----------.
 |  Router  |               |  Router  |          |  Router  |
 '----------'               '----------'          '----------'
      |                        |    |                 |
      |                   .----'    '---.         .--------.
      |                   |             |         |   IC   |
 .----------.             |             |         '--------'
 |   SH 1   |---.   .-----------.  .-----------.      |       .--------.
 '----------'   |   |    SH 1   |  |   SH 2    |      |   .---|  SH 1  |-.
      |         |   '-----------'  '-----------'      |   |   '--------' |
      |         |         |   |      |  |             |   |   .--------. |
 .----------.   |         |   '------'  |             |   |---|  SH 2  |-'
 |   SH 2   |---'         |             |             |   |   '--------'
 '----------'             |             |             |   |
      |              .----------.  .----------.  .----------.
      |              |  Switch  |  |  Switch  |  |  Switch  |
 .----------.        '----------'  '----------'  '----------'
 |  Switch  |
 '----------'
Serial deployment       Parallel deployment      Virtual in-path deployment
```

This has two advantages:

- If one of the Steelhead appliances fails, the other Steelhead appliance has a warm data store and can serve optimized data without having to learn all references again.
- If traffic between two sites goes via a cluster of parallel Steelhead appliances and the traffic takes a different path for the incoming and for the outgoing traffic, both Steelhead appliances will know the references and can use the references without having to learn it again.

When a Data Store Synchronization cluster gets configured, one of the Steelhead appliances is made the master and other one is made the slave. This has nothing to do with the direction of the flow data but with which data store ID is used in the replicated data store.

When one of the Steelhead appliances fails, either the master or the slave, the replacement Steelhead appliance will always have to be configured to be the slave. The Steelhead appliance which didn't get replaced will always become the master.

When a Data Store Synchronization cluster gets unconfigured or the Data Store Synchronization cluster gets redeployed in the network, both the Steelhead appliances will need to have their data store synchronization feature disabled and their data stores cleared (via the command `restart clean` on the CLI). If this doesn't happen, both Steelhead appliances will show Service Alarm errors because they will see a Steelhead appliance in the network with the same data store ID and that is a situation which should not happen.

Data Store Synchronization clusters should be the same hardware and run the same RiOS version. If there are mismatches the synchronization service will throw an alarm and the status of the Steelhead appliance will become degraded.

When a Steelhead appliance in a Data Store Synchronization cluster comes back after a reboot or power-down, all new references learned are replicated from the other Steelhead appliance. In the log files this can be seen as:

Figure 5.60. Data Store Synchronization catching up after a restart

```
SH sport[123] [replicate/client.INFO] - {- -} Connected from: 40282 to: 7744
SH sport[123] [replicate/client.NOTICE] - {- -} Client Connected to 10.23.18.212:7744 tota \
    l_pages: 85258240 pages_in_use: 85258238
SH sport[123] [replicate/client.INFO] - {- -} Recvd header info with version: 3 store_id_: \
     482671
SH sport[123] [replicate/storesync.INFO] - {- -}  current store_id: 482671 remote store_id \
    : 482671
SH sport[123] [replicate/storesync.NOTICE] - {- -} Running with storeid 482671
SH sport[123] [replicate/client.NOTICE] - {- -} Requesting keepup start
SH sport[123] [replicate/client.NOTICE] - {- -} Requesting catchup start npages: 85258240
SH sport[123] [replicate/client.INFO] - {- -} [ping] Scheduling ping every 60 seconds.
SH sport[123] [replicate/sync_server.NOTICE] - {- -} Came to end of store.  Will end catch \
    up next time client request indexes.
SH sport[123] [replicate/sync_server.NOTICE] - {- -} requesting end to catchup cur_page_:  \
    85258240 npages_: 85258240
SH sport[123] [replicate/sync_server.NOTICE] - {- -} Done with catchup  Sent: 24573 pages  \

SH sport[123] [replicate/client.NOTICE] - {- -} client catchup is complete. catchup pages_ \
    written: 31720
```

5.13. Data store related errors

The Data store is the database with references to frames the Steelhead appliance has learned. There are two possible issues with regarding to data store:

- Data store identification, where the data store identifier is reused.
- Data store contents, with issues with regarding to references exchanged.

5.13.1. Data store identification issues

The Data store ID is a unique identifier to identify the data store on a Steelhead appliance. There is only one reason why a data store ID used on two Steelhead appliances could be the same and that is because they are part of a data store synchronization cluster: The backup node in the data store cluster will assume the data store ID of the master node.

Once a Steelhead is taken out of a data store synchronization cluster, its data store should be cleared before redeploying it. If this doesn't happen and the two Steelhead appliances detect each other in the field, they will complain about the duplicate data store ID.

Figure 5.61. Duplicate data store ID warnings from a former data store synchronization cluster node

```
SH sport[123] [splice/oob.NOTICE] 1 {- -} Got a iochannel for an existing oob to peer: 192 \
    .168.1.6
SH sport[123] [splice/oob.ALERT] 1 {- -} ALARM Peer sport id (478552) is the same as mine! \
     Closing connection to peer with store id (478552) remote address (192.168.1.6:7800).
SH sport[123] [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
```

If this happens, remove the obsolete data store synchronization configuration from the Steelhead appliance and restart the optimization service in the GUI with the option *Clear the data store* or with the CLI command `restart clean`.

Sometimes a data store synchronization cluster is deployed in a serial cluster, and it is possible that the first Steelhead appliance in the cluster will try to peer with the second Steelhead appliance in the cluster. This will show up as:

Figure 5.62. Duplicate data store ID warning in a data store synchronization cluster

```
SH sport[2967]: [sport/config_info.ALERT] - {- -} ALARM Peer (10.0.1.5) store id (685523)  \
    is the same as mine! Check for replicated datastore steelheads used as a peer.
```

If this happens, add peering rules on the two Steelhead appliances to prevent auto-discovery happening between the two of them:

Figure 5.63. Prevent optimization from the data store synchronization peer 192.168.1.7

```
SSH1 # show in-path peering rules
Rule Type C Source            Destination       Port   Peer           SSL-Cap
---- ---- - ----------------- ----------------- ------ -------------- -------
1    pass A all-ip            all-ip            all    192.168.1.7    no-chk
      desc: Prevent optimization from SSH2 (inpath0_0)
1    pass A all-ip            all-ip            all    all-ipv4       in-cap
      desc: Default rule to passthrough connections destined to currently bypassed SSL cli \
    ent-server pairs
2    auto A all-ip            all-ip            443    all-ipv4       cap
      desc: Default rule to auto-discover and attempt to optimize connections destined to  \
    port 443 as SSL
def  auto A all-ip            all-ip            all    all-ipv4       no-chk
---- ---- - ----------------- ----------------- ------ -------------- -------
3 user added rule(s)
(C) Cloud-accel mode:            A=Auto
                                 P=Passthru
```

5.13.1.1. Duplicate data store ID on Steelhead mobile clients.

Data store IDs in Steelhead Mobile Clients are generated by the PCs themselves when the Steelhead Mobile Client software is started for the first time. Before Steelhead Mobile version 3.1.3, the source for this Data store ID is the Security Identifier provided via the Windows operating system.

Although the Security Identifier is supposed to be unique, it is not. For normal operation of the Windows operating system and the various integrations into Active Directory, this is not an issue. For applications which use it as a source of uniqueness for the machine this is a problem.

The source of the problem of duplicate data store IDs on Steelhead mobile clients is caused by installation of new computers via cloning them from a master computer: The clones of this master machine have the same Security Identifier which will cause the creation of the same Data store ID on the Steelhead Mobile Clients.

Figure 5.64. Detection of duplicate labels

```
SH sport[123] [segpage.ERR] - {- -} Duplicate DEF {}25 hash 3792051065887642647 vs. 420433 \
    /96599509:1411#0{}115 hash 7975538566276211596, memcmp() -1
SH sport[123] [defunpacker.ALERT] - {- -} ALARM (clnt: 10.0.1.1:60663 serv: 192.168.1.1:53 \
     cfe: 10.0.1.1:2770 sfe: 192.168.1.5:7810) name maps to more than one segment has a st \
    eelhead been improperly installed and/or steelhead mobiles are sharing config files po \
    ssibly through copying or cloning?
```

In this case the *clnt* and the *cfe* are the same, which indicates that this is a Steelhead Mobile Client.

To overcome this issue, the following steps need to be taken:

- After the installation of the Steelhead Mobile Client on the master machine, remove the configuration file `C:\Documents and Settings\All Users\Application Data\Riverbed\Steelhead_Mobile\config\sport.xml`.
- After the cloning of the machine, run the program `newsid.exe` on the newly cloned machine. This will cause it to generate a new Security Identifier.
- Stop the Steelhead Mobile Client, issue the command `rbtdebug --new-sid` in the directory `C:\Program Files\Riverbed\Steelhead Mobile` and restart the Steelhead Mobile Client.
- Restart the Steelhead Mobile Client with the command `rbtsport -C` in the directory `C:\Program Files\Riverbed\Steelhead Mobile`.

Now the Steelhead Mobile Client is ready to be used.

Note that this issue is resolved in Steelhead Mobile Client version 3.1.3 and 3.2 and later where the Steelhead Mobile Client Data store ID gets determined by a different method than the Security Identifier.

5.13.2. Data store contents issues

If there are multiple Steelhead appliances in the network with the same Data store ID, data referencing will be hampered:

- The Steelhead appliance can have a reference with a label which the peer Steelhead appliance doesn't know about. The peer Steelhead appliance will request to have the definition send again and no error will be raised.
- The Steelhead appliance can generate a new reference with a label which the peer Steelhead appliance already has. Both Steelhead appliances will destroy the reference and raise an error.
- The Steelhead appliance can have a reference with a label which the peer Steelhead appliance has, but with a different checksum. Both Steelhead appliances will destroy the reference and an error will be raised.

In the log files it will show up like:

Figure 5.65. Detection of duplicate labels

```
SH sport[123] [segpage.ERR] - {- -} Duplicate DEF {}25 hash 3792051065887642647 vs. 420433 \
    /96599509:1411#0{}115 hash 7975538566276211596, memcmp() -1
SH sport[123] [defunpacker.ALERT] - {- -} ALARM (clnt: 10.0.1.1:60663 serv: 192.168.1.1:53 \
     cfe: 10.0.1.5:2770 sfe: 192.168.1.5:7810) name maps to more than one segment has a st \
    eelhead been improperly installed and/or steelhead mobiles are sharing config files po \
    ssibly through copying or cloning?
```

In this case the issue was happening between the Steelhead appliances with in-path IP addresses of 192.168.250.1 and 192.168.7.12. The TCP session it was optimizing had the IP addresses of 192.168.250.1 and 192.168.7.7.

5.13.2.1. False alerts about duplicate labels

When the data store on a Steelhead appliance gets cleared, the contents are only forgotten on that Steelhead appliance but the other devices in the network still have the references. In due time they will be removed by garbage collection methods in the optimization service, but for now they are there. If the Steelhead appliance with the empty data store generates a new label which still exist on another Steelhead appliance, the checksum of the new reference will most likely not match the checksum of the old reference. The Steelhead appliance which will receive the new definition will detect this duplicate definition and complain about it with the same kind of logs as above.

Keeping track of when data stores are cleared is a good thing to catch these false positives.

To overcome this alarm, use the command `service error reset`:

Figure 5.66. Use of the command "service error reset"

```
SH # service error reset
Please note that it may take a few seconds for the alarm to reset.
```

5.14. WCCP issues

WCCP is a protocol which is used for the redirection of packets from a WCCP router (read: the WAN router or LAN switch) to a set of WCCP caches (read: the Steelhead appliances).

Figure 5.67. A WCCP setup

```
     .-,(  ),-.
  .-(          )-.         .----------------------------.      .--------.
 (     network     )-------| WCCP router                |------| Server |
  '-(          ).-'        '-------------.--------------'  |   '--------'
      '-.( ).-'                          |                 |   .--------.
                                .--------'--------,        '---| Server |
                                |                 |        |   '--------'
                           .------------.  .------------.  |   .--------.
                           | WCCP cache |  | WCCP cache |  '---| Server |
                           '------------'  '------------'      '--------'
WCCP router: 192.168.2.1
WCCP cache: 192.168.2.2
WCCP cache: 192.168.2.3
```

5.14.1. The configuration of a WCCP router and WCCP cache

The WCCP router has one or more service-groups defined. A service-group is a set of IP addresses and TCP port numbers which are directed to a specific group of caches. By configuring multiple service-groups, different groups of WCCP caches can be used for the redirection of different kinds of traffic.

The WCCP cache has one or more service-groups and every service-group has one or more WCCP routers configured and various negotiable options like the encapsulation method and the assignment type.

5.14.2. Setup of operation of a WCCP cache

The negotiation of a WCCP cache with the WCCP router is initiated by the WCCP cache: It sends a WCCP HERE_I_AM packet to the WCCP routers, which should be answered with a WCCP I_SEE_YOU packet.

With tcpdump, this looks like:

Figure 5.68. Tcpdump of WCCP traffic

```
SH # tcpdump -ni wan0_0 port 2048
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
16:57:40.916187 IP 192.168.2.2.2048 > 192.168.2.1.2048: UDP, length 144
16:57:40.948061 IP 192.168.2.1.2048 > 192.168.2.2.2048: UDP, length 140
16:57:50.927298 IP 192.168.2.2.2048 > 192.168.2.1.2048: UDP, length 144
16:57:50.957309 IP 192.168.2.1.2048 > 192.168.2.2.2048: UDP, length 140
```

The UDP packet gets send from the in-path IP address of the Steelhead appliance to the WCCP router IP address. If the WCCP router is willing to accept the WCCP cache, it will send an UDP packet back.

5.14.2.1. No answer from the WCCP router

Figure 5.69. Tcpdump of WCCP traffic

```
SH # tcpdump -ni wan0_0 port 2048
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
16:57:40.916187 IP 192.168.2.2.2048 > 192.168.2.1.2048: UDP, length 144
16:57:50.927298 IP 192.168.2.2.2048 > 192.168.2.1.2048: UDP, length 144
```

Besides the obvious two reasons, that the IP address of the WCCP router is entered incorrectly on the WCCP cache or that the WCCP router is not configured correctly, this will happen when the WCCP options given by the WCCP cache are incompatible by the current operation of the WCCP router:

- The WCCP cache encapsulation and assignment options are not compatible with the capabilities of the WCCP router. For example the WCCP cache wants to use GRE forwarding, the WCCP router can only do L2 forwarding.

- This WCCP cache options are different from the options configured on the other WCCP caches in the same service group. In this scenario there will always be one WCCP cache joined to the WCCP router and the second one will not be successful.

5.14.2.2. Invalid answer from the WCCP router

It is possible that the IP address of the WCCP router configured on the WCCP cache is not the source IP address of the WCCP I_SEE_YOU packets returned and thus the packets are ignored by the WCCP cache.

Figure 5.70. Tcpdump of WCCP traffic

```
SH # tcpdump -ni wan0_0 port 2048
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
16:57:40.916187 IP 192.168.2.2.2048 > 172.16.3.2.2048: UDP, length 144
16:57:40.948061 IP 192.168.2.1.2048 > 192.168.2.2.2048: UDP, length 140
16:57:50.927298 IP 192.168.2.2.2048 > 172.16.3.2.2048: UDP, length 144
16:57:50.957309 IP 192.168.2.1.2048 > 192.168.2.2.2048: UDP, length 140
```

This could happen if the IP address of the WCCP router configured on the WCCP cache is an HSRP or a loopback address and the answers are coming from the IP address of the outgoing interface of the WCCP router.

5.14.3. Redirection of traffic towards a WCCP cache

Once the WCCP session is successfully negotiated, it will redirect packets as defined in the WCCP redirection list. This redirection list should be applied on both the incoming WAN interface and the incoming LAN interfaces.

The best practice is to use inbound redirection of the interfaces of the WCCP router: Packets redirected when they come in on the interface will have had less processing done on them, leaving the router free to do other important things like forwarding packets.

If the WCCP router and the WCCP cache are on the same IP subnet, L2 forwarding should be preferred over GRE encapsulation, to overcome possible IP fragmentation overhead.

When traffic or a certain kind of traffic does not get optimized, the easiest way to find out what happens is on the Steelhead appliance WAN interface with tcpdump:

- If the only traffic seen is traffic on UDP port 2048 and Ethernet broadcast traffic, then the WCCP router is not redirecting packets to the WCCP cache.

  Figure 5.71. Tcpdump shows that the WCCP router is not forwarding packets to the WCCP cache

  ```
  SH # tcpdump -ni wan0_0 -c 100
  tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
  listening on wan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
  16:57:40.916187 IP 192.168.2.2.2048 > 192.168.2.1.2048: UDP, length 144
  16:57:40.948061 IP 192.168.2.1.2048 > 192.168.2.2.2048: UDP, length 140
  16:57:42.123456 ARP, Request who-has 192.168.2.2 tell 192.168.2.5, length 46
  16:57:44.654321 ARP, Request who-has 192.168.2.2 tell 192.168.2.5, length 46
  16:57.48.932198 snap 0:0:c:20:0 CDP v2, ttl: 180s, checksum: 692 (unverified)
          Device-ID (0x01), length: 25 bytes: 'WCCP01.example.com'
  16:57:50.927298 IP 192.168.2.2.2048 > 192.168.2.1.2048: UDP, length 144
  16:57:50.957309 IP 192.168.2.1.2048 > 192.168.2.2.2048: UDP, length 140
  ```

  The next steps would be checking the path the traffic takes and the WCCP redirection access-lists.

- If the output shows traffic coming from one side, as in only WAN side traffic to port 80 or only LAN side traffic from port 80, then the redirect lists on either the LAN side VLANs or the WAN side interfaces is incorrect.

Figure 5.72. Tcpdump shows that the traffic is only forwarded from the server to the client

```
SH # tcpdump -ni wan0_0 port 80
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 96 bytes
12:07:09.598157 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [S.], seq 2971872340, ack 387224 \
    0794, win 5792, options [mss 1460,sackOK,TS val 285084142 ecr 5351449,nop,wsc
ale 2], length 0
12:07:09.602393 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [.], seq 1, ack 188, win 1716, o \
    ptions [nop,nop,TS val 285084146 ecr 5351589], length 0
12:07:09.875867 IP 192.168.1.1.80 > 10.0.1.1.42825: Flags [P.], seq 1:1151, ack 188, win 1 \
    716, options [nop,nop,TS val 285084420 ecr 5351589], length 1150
12:07:10.881392 IP 192.168.1.6.80 > 10.0.1.100.42825: Flags [.], seq 1151, ack 420, win 17 \
    16, options [nop,nop,TS val 285085425 ecr 5352869], length 0
12:07:11.106812 IP 192.168.1.6.80 > 10.0.1.100.42825: Flags [P.], seq 1151:1663, ack 420,  \
    win 1716, options [nop,nop,TS val 285085651 ecr 5352869], length 512
```

The next steps would be to confirm that the forward traffic is going via the WCCP router and to check the redirection access-lists.

5.15. Asymmetric Routing

As described in an earlier chapter, Asymmetric Routing is one of the reasons that auto-discovery for a TCP session expected to be optimized fails due to network routing related issues:

Figure 5.73. Different forms of asymmetric routing.

```
.--------.
|        |<----------------------------------------.
|        |-----------.     SYN/ACK+                |
|        |   ACK     v                             |
|        |         .-------------.          .-------------.
|        |         |             |          |             |         .--------.
|        |   SYN   |             |   SYN+   |             |   SYN+  |        |
| Client |-------->| Client-side |--------->| Server-side |-------->| Server |
|        |<--------| Steelhead   |<---------| Steelhead   |<--------|        |
|        | SYN/ACK |             | SYN/ACK+ |             | SYN/ACK '--------'
|        |         |             |          '-------------'            |  |
|        |         |             |                                     |  |
|        |         |             |                                     |  |
|        |         |             |<------------------------------------'  |
|        |         '-------------'                SYN/ACK                 |
|        |   ACK      ^                                                   |
|        |------------'                                                   |
|        |<---------------------------------------------------------------'
'--------'                      SYN/ACK
```

- At the top the SYN/ACK+ sent from the server-side Steelhead appliance bypasses the client-side Steelhead appliance, that is called Client Side Asymmetry.
- When the SYN/ACK sent from the server bypasses the server-side Steelhead appliance, that is called Server Side Asymmetry.
- When the SYN/ACK sent from the server bypasses both the server-side and client-side Steelhead appliance Steelhead appliance, that is called Complete Asymmetry.

The solution to resolve asymmetric routing issues is simple: Make sure that all traffic goes through the server-side and the client-side Steelhead appliance. This can be done by adding additional by-pass cards in the Steelhead appliance to cover all the links going out of a site, or by rolling out multiple Steelhead appliances per site and to use Connection Forwarding to inform the other Steelhead appliances that all traffic for this TCP session should be forwarded to this Steelhead appliance.

The last form of Asymmetric Routing is SYN Retransmit, which happens when the SYN+ packet does not get answered by either the server or another Steelhead appliance, but when a normal TCP SYN packet does get answered by the server.

When asymmetric traffic is detected, the IP addresses pair gets added to the asymmetric routing table and is passed-through for an interval to normally establish unoptimized TCP sessions.

Figure 5.74. Asymmetric routing table via the CLI

```
SH # show in-path asym-route-tab

[IP 1]  [IP 2]  [reason]  [timeout (sec)]  [created]  [last used]

10.0.1.1 192.168.1.1 no-SYNACK 86237 (07/03/12 12:34:51) (07/03/12 12:34:51)
10.0.1.1 192.168.1.1 invalid-SYNACK 55035 (07/03/12 18:46:04) (07/03/12 20:26:48)
10.0.1.1 192.168.1.1 bad-RST 86350 (07/17/12 11:14:29) (07/17/12 11:14:30)
10.0.1.1 192.168.1.1 probe-filtered(not-AR) 298 (07/03/12 11:57:39) (07/03/12 11:57:39)
```

5.15.1. Normal flow of packets

Figure 5.75. Normal setup of an optimized TCP session on the network map

```
 .--------.
 |        |         .-------------.          .-------------.
 |        |   SYN   |             |   SYN+   |             |   SYN+   .--------.
 | Client |-------->| Client-side |--------->| Server-side |-------- >| Server |
 |        | SYN/ACK | Steelhead   | SYN/ACK+ | Steelhead   | SYN/ACK  |        |
 |        |<---------             |<---------|             |<---------|        |
 |        |         |             |          |             |          '--------'
 |        |         |             |          '-------------'
 |        |         '-------------'
 '--------'
```

With normal auto-discovery, you should see the naked SYN packet on the LAN interface, the SYN+ packet on the WAN interfaces and the SYN/ACK+ on the WAN interfaces, the SYN+ again and a SYN/ACK on the server-side Steelhead appliance LAN interface, a SYN/ACK+ on the WAN interfaces, some traffic on port 7800 on the WAN interfaces and then a ACK on the LAN interfaces of the server-side Steelhead appliance and then a SYN/ACK and an ACK on the client-side Steelhead appliance LAN interface.

Figure 5.76. Auto-discovery for of an optimized TCP session - Flow of packets

```
      Client                CSH                 SSH                  Server
 1          ----- SYN ----->
 2                             ----- SYN+ ----->
 3                             <--- SYN/ACK+ ---
 4                                                 ----- SYN+ ----->
 5                                                 <--- SYN/ACK ----
 6                             <--- SYN/ACK+ ---
 7                             -- Setup Inner ->
 8                                                 ----- ACK ------>
 9          <--- SYN/ACK ---
10          ----- ACK ----->
```

On the wire, it will look like this:

Figure 5.77. Auto-discovery for an optimized TCP session - tcpdump

```
CSH LAN
12:50:33.020243 IP 10.0.1.1.61284 > 192.168.1.1.50: Flags [S], seq 1526534172, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3783247869 ecr 0,sackOK,eol], length 0
12:50:33.153146 IP 192.168.1.1.50 > 10.0.1.1.61284: Flags [S.], seq 4133019662, ack 152653 \
    4173, win 5792, options [mss 1460,sackOK,TS val 6324752 ecr 3783247869,nop,wscale 2],  \
    length 0
12:50:33.156482 IP 10.0.1.1.61284 > 192.168.1.1.50: Flags [.], seq 1, ack 1, win 65535, op \
    tions [nop,nop,TS val 3783247998 ecr 6324752], length 0

CSH WAN
12:50:33.020331 IP 10.0.1.1.61284 > 192.168.1.1.50: Flags [S], seq 1526534172, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3783247869 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
12:50:33.153021 IP 192.168.1.1.50 > 10.0.1.1.61284: Flags [S.], seq 20020520, ack 15265341 \
    73, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3783247869 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
12:50:33.153061 IP 192.168.1.1.50 > 10.0.1.1.61284: Flags [S.], seq 20020520, ack 15265341 \
    73, win 65535, options [mss 1460,nop,wscale 3,TS val 3783247869 ecr 0,sackOK,rvbd-prob \
    e AD CSH:10.0.1.6 SSH:192.168.1.6:7800 11110a000106c0a801061e78,rvbd-probe EAD 0e3d,no \
    p,eol], length 0

SSH WAN
12:33:06.883718 IP 10.0.1.1.61284 > 192.168.1.1.50: Flags [S], seq 1526534172, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3783247869 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
12:33:06.883823 IP 192.168.1.1.50 > 10.0.1.1.61284: Flags [S.], seq 20020520, ack 15265341 \
    73, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3783247869 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
12:33:06.884217 IP 192.168.1.1.50 > 10.0.1.1.61284: Flags [S.], seq 20020520, ack 15265341 \
    73, win 65535, options [mss 1460,nop,wscale 3,TS val 3783247869 ecr 0,sackOK,rvbd-prob \
    e AD CSH:10.0.1.6 SSH:192.168.1.6:7800 11110a000106c0a801061e78,rvbd-probe EAD 0e3d,no \
    p,eol], length 0

SSH LAN
12:33:06.884044 IP 10.0.1.1.61284 > 192.168.1.1.50: Flags [S], seq 2724579398, win 5840, o \
    ptions [mss 1460,sackOK,TS val 4294837098 ecr 0,nop,wscale 2,rvbd-probe AD CSH:10.0.1. \
    6 01010a0001060005,rvbd-probe EAD 0c05,nop,eol], length 0
12:33:06.884131 IP 192.168.1.1.50 > 10.0.1.1.61284: Flags [S.], seq 3757738477, ack 272457 \
    9399, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 1624809945 ecr 429483709 \
    8], length 0
12:33:06.884177 IP 10.0.1.1.61284 > 192.168.1.1.50: Flags [.], seq 1, ack 1, win 1460, opt \
    ions [nop,nop,TS val 4294837098 ecr 1624809945], length 0
```

5.15.2. How to identify the Client-side Asymmetry

Client-side Asymmetry happens when the return traffic from the server-side Steelhead appliance bypasses the client-side Steelhead appliance and goes directly back to the client.

Figure 5.78. Client-side Asymmetry on the network map

```
 .--------.
 |        |<--------------------------------------.
 |        |-----------.     SYN/ACK+              |
 |        |   ACK     v                           |
 |        |         .-------------.         .-------------.
 |        |   SYN   |             |   SYN+  |             |         .--------.
 | Client |-------->| Client-side |-------->| Server-side |         | Server |
 |        |         | Steelhead   |         | Steelhead   |         |        |
 |        |         |             |         |             |         '--------'
 |        |         |             |         '-------------'
 |        |         '-------------'
 '--------'
```

With Client-side Asymmetry, the packets seen are the naked SYN packet on the LAN interface, the SYN+ packet on the WAN interfaces, a SYN/ACK+ on the WAN interface of the server-side Steelhead appliance and an ACK on the LAN interface of the client-side Steelhead appliance.

Figure 5.79. Client-side Asymmetry - Flow of packets

```
      Client                CSH                 SSH                  Server
 1          ----- SYN ----->
 2                             ----- SYN+ ----->
 3          <---------------/ /-- SYN/ACK+ -----
 4          <---------------/ /-- SYN/ACK+ -----
 5          ----- ACK ---------------------------------------------->
```

On the wire, it will look like this:

Figure 5.80. Client-side Asymmetry on the wire

```
CSH LAN
12:42:20.243577 IP 10.0.1.1.61242 > 192.168.1.1.50: Flags [S], seq 2702983144, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3782775016 ecr 0,sackOK,eol], length 0
12:42:20.314239 IP 10.0.1.1.61242 > 192.168.1.1.50: Flags [.], seq 2702983145, ack 2002052 \
    1, win 65535, options [nop,nop,TS val 3782775083 ecr 3782775016], length 0

CSH WAN
12:42:20.243692 IP 10.0.1.1.61242 > 192.168.1.1.50: Flags [S], seq 2702983144, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3782775016 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
12:42:20.314349 IP 10.0.1.1.61242 > 192.168.1.1.50: Flags [.], seq 2702983145, ack 2002052 \
    1, win 65535, options [nop,nop,TS val 3782775083 ecr 3782775016], length 0

SSH WAN
12:24:53.637086 IP 10.0.1.1.61242 > 192.168.1.1.50: Flags [S], seq 2702983144, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3782775016 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
12:24:53.637205 IP 192.168.1.1.50 > 10.0.1.1.61242: Flags [S.], seq 20020520, ack 27029831 \
    45, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3782775016 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
12:24:53.637713 IP 192.168.1.1.50 > 10.0.1.1.61242: Flags [S.], seq 20020520, ack 27029831 \
    45, win 65535, options [mss 1460,nop,wscale 3,TS val 3782775016 ecr 0,sackOK,rvbd-prob \
    e AD CSH:10.0.1.6 SSH:192.168.1.6:7800 11110a000106c0a801061e78,rvbd-probe EAD 0e3d,no \
    p,eol], length 0
12:24:53.707892 IP 10.0.1.1.61242 > 192.168.1.1.50: Flags [.], seq 1, ack 1, win 65535, op \
    tions [nop,nop,TS val 3782775083 ecr 3782775016], length 0
```

Figure 5.81. Client-side Asymmetry in the logs

```
CSH kernel: [intercept.WARN] asymmetric routing between 10.0.1.1:61242 and 192.168.1.1:50  \
    detected (no SYN/ACK)
```

5.15.3. How to identify the Server-side Asymmetry

Service-side Asymmetry happens when the return traffic from the server bypasses the server-side Steelhead appliance and goes directly to the client-side Steelhead appliance.

Figure 5.82. Server-side Asymmetry on the network map

```
 .--------.
 |        |         .-------------.          .-------------.
 |        |   SYN   |             |   SYN+   |             |   SYN+  .--------.
 | Client |-------->| Client-side |--------->| Server-side |-------->| Server |
 |        |         | Steelhead   |<---------| Steelhead   |         |        |
 |        |         |             | SYN/ACK+ |             |         '--------'
 |        |         |             |          '-------------'            |
 |        |         |             |                                     |
 |        |         |             |                                     |
 |        |         |             |<------------------------------------'
 |        |         '-------------'                SYN/ACK
 '--------'
```

With Server-side Asymmetry, the packets seen are the naked SYN packet on the LAN interface, the SYN+ and SYN/ACK+ packets on the WAN interfaces, a SYN+ on the LAN interface of the server-side Steelhead appliance and then a SYN/ACK on the client-side Steelhead appliance WAN interface.

Figure 5.83. Server-side Asymmetry - Flow of packets

```
      Client                CSH                   SSH                 Server
 1          ----- SYN ----->
 2                             ----- SYN+ ------->
 3                             <---- SYN/ACK+ ----
 4                                                     ----- SYN+ ----->
 5                             <------------------/ /--- SYN/ACK -----
```

On the wire, it will look like this:

Figure 5.84. Server-side Asymmetry on the wire

```
CSH LAN
10:51:35.610009 IP 10.0.1.1.65116 > 192.168.1.1.50: Flags [S], seq 3361390561, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3862289659 ecr 0,sackOK,eol], length 0
10:51:35.768643 IP 192.168.1.1.50 > 10.0.1.1.65116: Flags [S.], seq 64094818, ack 45721004 \
    4, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 493782756 ecr 1312273], len \
    gth 0

CSH WAN
10:51:35.610117 IP 10.0.1.1.65116 > 192.168.1.1.50: Flags [S], seq 3361390561, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3862289659 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
10:51:35.766684 IP 192.168.1.1.50 > 10.0.1.1.65116: Flags [S.], seq 20020520, ack 33613905 \
    62, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3862289659 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
10:51:35.769341 IP 192.168.1.1.50 > 10.0.1.1.65116: Flags [S.], seq 64094818, ack 45721004 \
    4, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 493782756 ecr 1312273], len \
    gth 0

SSH WAN
10:33:56.895288 IP 10.0.1.1.65116 > 192.168.1.1.50: Flags [S], seq 3361390561, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3862289659 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
10:33:56.895407 IP 192.168.1.1.50 > 10.0.1.1.65116: Flags [S.], seq 20020520, ack 33613905 \
    62, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3862289659 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0

SSH LAN
10:33:56.896551 IP 10.0.1.1.65116 > 192.168.1.1.50: Flags [S], seq 457210043, win 5840, op \
    tions [mss 1460,sackOK,TS val 1312273 ecr 0,nop,wscale 2,rvbd-probe AD CSH:10.0.1.6 01 \
    010a0001060005,rvbd-probe EAD 0c05,nop,eol], length 0
```

Figure 5.85. Server-side Asymmetry in the logs

```
CSH kernel: [intercept.WARN] asymmetric routing between 192.168.1.1:50 and 10.0.1.1:65116  \
    detected (invalid SYN/ACK)
```

5.15.4. How to identify the Complete Asymmetry

Complete Asymmetry happens when the return traffic from the server bypasses both the Steelhead appliances and goes directly to the client.

Figure 5.86. Complete Asymmetry on the network map

```
 .--------.
 |        |         .-------------.          .-------------.
 |        |   SYN   |             |   SYN+   |             |   SYN+  .--------.
 | Client |-------->| Client-side |--------->| Server-side |-------->| Server |
 |        |         | Steelhead   |<---------| Steelhead   |         |        |
 |        |         |             | SYN/ACK+ |             |         '--------'
 |        |         |             |          '-------------'               |
 |        |         '-------------'                                        |
 |        |   RST      ^                                                   |
 |        |------------'                                                   |
 |        |<---------------------------------------------------------------'
 '--------'                     SYN/ACK
```

With Complete Asymmetry, the packets seen are the naked SYN packet on the LAN interface, the SYN+ packet on the WAN sides and the SYN/ACK+ on the WAN interface, a SYN+ on the server-side Steelhead appliance LAN interface and a RST on the client-side Steelhead appliance LAN interface.

Figure 5.87. Complete Asymmetry - Flow of packets

```
      Client                CSH                  SSH                 Server
 1          ----- SYN ----->
 2                             ----- SYN+ ------->
 3                             <---- SYN/ACK+ ----
 4                                                   ----- SYN+ ----->
 5          <---------------/ /------------------/ /--- SYN/ACK -----
 6          ----- RST --------------------------------------------->
```

On the wire, it will look like this:

Figure 5.88. Complete Asymmetry on the wire

```
CSH LAN
11:14:29.879623 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [S], seq 3888516971, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3863658656 ecr 0,sackOK,eol], length 0
11:14:29.953403 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [R], seq 1895386924, win 0, leng \
    th 0

CSH WAN
11:14:29.879773 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [S], seq 3888516971, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3863658656 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
11:14:29.953521 IP 192.168.1.1.50 > 10.0.1.1.65268: Flags [S.], seq 20020520, ack 38885169 \
    72, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3863658656 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
11:14:30.038356 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [R], seq 1895386924, win 0, leng \
    th 0

SSH WAN
10:56:51.168188 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [S], seq 3888516971, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3863658656 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
10:56:51.168305 IP 192.168.1.1.50 > 10.0.1.1.65268: Flags [S.], seq 20020520, ack 38885169 \
    72, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 3863658656 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
10:56:51.317550 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [R], seq 1895386924, win 0, leng \
    th 0

SSH LAN
10:56:51.170499 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [S], seq 1895386923, win 5840, o \
    ptions [mss 1460,sackOK,TS val 2686747 ecr 0,nop,wscale 2,rvbd-probe AD CSH:10.0.1.6 0 \
    1010a0001060005,rvbd-probe EAD 0c05,nop,eol], length 0
10:56:51.317623 IP 10.0.1.1.65268 > 192.168.1.1.50: Flags [R], seq 1895386924, win 0, leng \
    th 0
```

Figure 5.89. Complete Asymmetry in the logs

```
CSH kernel: [intercept.WARN] asymmetric routing between 10.0.1.1 and 192.168.1.1 detected  \
    (bad RST)
```

5.15.5. How to identify TCP SYN retransmit

SYN retransmit happens when there is no answer from the SYN+ packets but when the naked SYN is forwarded it is answered by a SYN/ACK.

Figure 5.90. Complete Asymmetry on the network map

```
 .--------.
 |        |         .-------------.          .-------------.
 |        |   SYN   |             | SYN+     |             |
 | Client |-------->| Client-side |---->...  | Server-side |
 |        |         | Steelhead   |--------->| Steelhead   |
 |        |         |             |   SYN    |             |
 |        |         |             |<---------|             |
 |        |         |             |  SYN/ACK '-------------'
 |        |         '-------------'
 '--------'
```

With TCP SYN retransmit, the packets seen are the naked SYN packet on the LAN interface, two SYN+ packets and a naked SYN packet on the WAN interface of the client-side Steelhead appliance WAN interface, only the naked SYN on the server-side Steelhead appliance WAN interface and a SYN/ACK on the client-side and server-side Steelhead appliances WAN interface.

Figure 5.91. TCP SYN retransmit - Flow of packets

```
      Client                CSH                   SSH                 Server
 1          ----- SYN ----->
 2                             ----- SYN+ --->...
 3                             ----- SYN+ --->...
 4                             ----- SYN ---------------------------->
 5          <------------------------------------------- SYN/ACK -----
```

On the wire, it will look like this:

Figure 5.92. TCP SYN retransmit on the wire

```
CSH LAN
CSH # tcpdump -ni lan0_0 port 50
tcpdump: WARNING: lan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
11:48:48.001687 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779692639 ecr 0,sackOK,eol], length 0
11:48:49.055780 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779693654 ecr 0,sackOK,eol], length 0
11:48:50.160832 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779694716 ecr 0,sackOK,eol], length 0
11:48:50.294530 IP 192.168.1.1.50 > 10.0.1.1.60961: Flags [S.], seq 3036249837, ack 116570 \
    4203, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 1496970988 ecr 377969471 \
    6], length 0
11:48:50.297878 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [.], seq 1, ack 1, win 65535, op \
    tions [nop,nop,TS val 3779694847 ecr 1496970988], length 0

CSH WAN
CSH # tcpdump -ni wan0_0 port 50
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
11:48:48.001814 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779692639 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
11:48:49.055894 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779693654 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
11:48:50.160937 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779694716 ecr 0,sackOK,eol], length 0
11:48:50.294456 IP 192.168.1.1.50 > 10.0.1.1.60961: Flags [S.], seq 3036249837, ack 116570 \
    4203, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 1496970988 ecr 377969471 \
    6], length 0
11:48:50.297927 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [.], seq 1, ack 1, win 65535, op \
    tions [nop,nop,TS val 3779694847 ecr 1496970988], length 0

SSH WAN
SSH # tcpdump -ni wan0_0 port 50
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
11:48:50.230826 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [S], seq 1165704202, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 3779694716 ecr 0,sackOK,eol], length 0
11:48:50.234423 IP 192.168.1.1.50 > 10.0.1.1.60961: Flags [S.], seq 3036249837, ack 116570 \
    4203, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 1496970988 ecr 377969471 \
    6], length 0
11:48:50.365821 IP 10.0.1.1.60961 > 192.168.1.1.50: Flags [.], seq 1, ack 1, win 65535, op \
    tions [nop,nop,TS val 3779694847 ecr 1496970988], length 0
```

Figure 5.93. TCP SYN retransmit in the logs

```
CSH kernel: [intercept.WARN] it appears as though probes from 10.0.1.1 to 192.168.1.1 are  \
    being filtered.  Passing through connections between these two hosts.
```

5.16. IP Routing Related Issues

In the simplest deployment scenario, an in-path deployment with the same IP subnet behind the Steelhead appliance LAN interface as on the in-path interface, the routing of packets coming from an in-path interface is simple: All delivery is either on the local IP subnet or to be sent to the default gateway of the IP subnet.

When there are multiple IP subnets defined on the LAN behind the Steelhead appliance or when the IP subnet on in-path interface is different from the IP subnets of the LANs, this approach of a single default gateway doesn't work anymore because traffic to the IP subnets which is not on the in-path interface will be sent to the default gateway on the WAN side of the Steelhead appliance instead of to the router on the LAN side of the Steelhead appliance. This causes that traffic to travel one extra hop, to be inspected by a firewall which might not like the traffic, to be applied to QoS rules defined on the WAN interface of the Steelhead appliance and so on.

Figure 5.94. Steelhead appliance with multiple LANs behind it

```
               192.168.0.0/24
 .------------.    ,----.    .------------.
 | WAN Router |----| SH |----| LAN Router |--- IP Subnet 1 / 192.168.1.0/24
 '------------'.9 W'----'L .8|            |
                             |            |--- IP Subnet 2 / 192.168.2.0/24
                             |            |
                             |            |--- IP Subnet 3 / 192.168.3.0/24
                             '------------'


WAN router MAC address: 00:0d:b9:17:28:dd
LAN router MAC address: 00:21:70:3e:6b:7f
Default gateway for Steelhead in-path interface: WAN router
```

With the in-path interface being on the same IP subnet as the rest of the hosts on the LAN side, in the auto-discovery phase there will be two or three packets seen on the WAN interface: one incoming SYN+ and one or two outgoing SYN/ACK+s, depending on if Enhanced Auto Discovery is enabled or not.

Figure 5.95. Tcpdump of traffic if the in-path IP address is on the same IP subnet as the hosts on the LAN

```
11:44:12.311915 00:0d:b9:17:28:dd > 00:21:70:3e:6b:7f, ethertype IPv4 (0x0800), length 94: \
     10.0.1.1.49850 > 192.168.1.1.50: Flags [S], seq 3346759466, win 65535, options [mss 1 \
    460,nop,wscale 1,nop,nop,TS val 2834406441 ecr 0,sackOK,nop,nop,rvbd-probe AD CSH:10.0 \
    .1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
11:44:12.312010 00:0e:b6:8c:7a:ed > 00:0d:b9:17:28:dd, ethertype IPv4 (0x0800), length 86: \
     192.168.1.1.50 > 10.0.1.1.49850: Flags [S.], seq 20020520, ack 3346759467, win 65535, \
     options [mss 1460,nop,wscale 1,nop,nop,TS val 2834406441 ecr 0,sackOK,nop,nop,rvbd-pr \
    obe EAD 0c01,nop,nop,nop,eol], length 0
11:44:12.312484 00:0e:b6:8c:7a:ed > 00:0d:b9:17:28:dd, ethertype IPv4 (0x0800), length 94: \
     192.168.1.1.50 > 10.0.1.1.49850: Flags [S.], seq 20020520, ack 3346759467, win 65535, \
     options [mss 1460,nop,wscale 1,TS val 2834406441 ecr 0,sackOK,rvbd-probe AD CSH:10.0. \
    1.6 SSH:192.168.1.6:7800 11110a000106c0a801061e78,rvbd-probe EAD 0e3d,nop,eol], length \
     0
```

With the in-path interface being on a different IP subnet than the rest of the hosts on the LAN side and the default gateway set to the WAN router, there will be seven packets seen on the WAN interface: One incoming SYN+, one outgoing SYN/ACK+, one forwarded SYN+ to the server seen twice, one final ACK to the server seen twice and one SYN/ACK+ back to the client-side Steelhead appliance.

Figure 5.96. Tcpdump of traffic on a different IP subnet behind the Steelhead appliance

```
11:44:25.629579 00:0d:b9:17:28:dd > 00:21:70:3e:6b:7f, ethertype IPv4 (0x0800), length 94: \
     10.0.1.1.49853 > 192.168.2.1.50: Flags [S], seq 3631007307, win 65535, options [mss 1 \
    460,nop,wscale 1,nop,nop,TS val 2834419715 ecr 0,sackOK,nop,nop,rvbd-probe AD CSH:10.0 \
    .1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
11:44:25.629695 00:0e:b6:8c:7a:ed > 00:0d:b9:17:28:dd, ethertype IPv4 (0x0800), length 86: \
     192.168.2.1.50 > 10.0.1.1.49853: Flags [S.], seq 20020520, ack 3631007308, win 65535, \
     options [mss 1460,nop,wscale 1,nop,nop,TS val 2834419715 ecr 0,sackOK,nop,nop,rvbd-pr \
    obe EAD 0c01,nop,nop,nop,eol], length 0
11:44:25.629966 00:0e:b6:8c:7a:ed > 00:0d:b9:17:28:dd, ethertype IPv4 (0x0800), length 90: \
     10.0.1.1.49853 > 192.168.2.1.50: Flags [S], seq 3498683517, win 5840, options [mss 14 \
    60,sackOK,TS val 11038597 ecr 0,nop,wscale 2,rvbd-probe AD CSH:10.0.1.6 01010a00010600 \
    05,rvbd-probe EAD 0c05,nop,eol], length 0
11:44:25.696347 00:0d:b9:17:28:dd > 00:21:70:3e:6b:7f, ethertype IPv4 (0x0800), length 90: \
     10.0.1.1.49853 > 192.168.2.1.50: Flags [S], seq 3498683517, win 5840, options [mss 14 \
    60,sackOK,TS val 11038597 ecr 0,nop,wscale 2,rvbd-probe AD CSH:10.0.1.6 01010a00010600 \
    05,rvbd-probe EAD 0c05,nop,eol], length 0
11:44:25.696686 00:0e:b6:8c:7a:ed > 00:0d:b9:17:28:dd, ethertype IPv4 (0x0800), length 66: \
     10.0.1.1.49853 > 192.168.2.1.50: Flags [.], seq 4162643507, ack 1520852933, win 1460, \
     options [nop,nop,TS val 11038663 ecr 2163923817], length 0
11:44:25.696790 00:0e:b6:8c:7a:ed > 00:0d:b9:17:28:dd, ethertype IPv4 (0x0800), length 94: \
     192.168.2.1.50 > 10.0.1.1.49853: Flags [S.], seq 20020520, ack 3631007308, win 65535, \
     options [mss 1460,nop,wscale 1,TS val 2834419715 ecr 0,sackOK,rvbd-probe AD CSH:10.0. \
    1.6 SSH:192.168.0.6:7800 11110a000106c0a800061e78,rvbd-probe EAD 0e3d,nop,eol], length \
     0
11:44:25.763141 00:0d:b9:17:28:dd > 00:21:70:3e:6b:7f, ethertype IPv4 (0x0800), length 66: \
     10.0.1.1.49853 > 192.168.2.1.50: Flags [.], seq 4162643507, ack 1520852933, win 1460, \
     options [nop,nop,TS val 11038663 ecr 2163923817], length 0
```

Checking the source and destination MAC addresses of the packets, you can see that the additional packets are the ones normally expected on the LAN side of the Steelhead appliance. However, because the default gateway is set to the WAN side router, currently they are first send back to the WAN router which then forwards them to the LAN router.

5.16.1. Static Routing Configuration

The first solution would be to configure the gateways to these IP subnets behind the Steelhead appliance in the routing table of the Steelhead in-path interface:

Figure 5.97. In-path interface gateways

```
SH (config) # ip in-path route inpath0_0 192.168.1.0 /24 192.168.0.8
SH (config) # ip in-path route inpath0_0 192.168.2.0 /24 192.168.0.8
SH (config) # ip in-path route inpath0_0 192.168.3.0 /24 192.168.0.8

SH # show ip in-path route inpath0_0 static
Destination       Mask              Gateway
default           0.0.0.0           192.168.0.8
192.168.1.0       255.255.255.0     192.168.0.8
192.168.2.0       255.255.255.0     192.168.0.8
192.168.3.0       255.255.255.0     192.168.0.8
```

This is an administrative overhead which needs to be synchronized every time a new IP subnet is defined behind the Steelhead appliance, but it will make sure that the in-path interface does know the path to the configured IP subnets.

5.16.2. Simplified Routing Configuration

Instead of having to configure all the IP subnets behind the LAN interfaces of the Steelhead appliances, the Steelhead appliance can learn the source and destination IP addresses of the packets received by associating it with the source and destination MAC addresses of the Ethernet frame.

This way it will build a list of destination IP addresses and the MAC address on the Ethernet segment they are reachable via. IP addresses not in the Simplified Routing table will be sent out via the normal IP routing table.

5.16.2.1. Learning from the sources, destinations, both or all

The Simplified Routing method can populate the IP address and MAC address combinations in the Simplified Routing table by looking at different kinds of data:

- Destination-only: The destination IP addresses and destination MAC addresses of optimized TCP sessions. This will populate the Simplified Routing table with the destination IP addresses based on the routing information to the destination IP addresses. This is the default since RiOS versions 6.0.
- Source and destination: Both the source and destination IP addresses of optimized TCP sessions.
- All: The source and destination IP addresses of all IP traffic, not only the optimized TCP sessions.

5.16.2.2. Examining the Simplified Routing table

Figure 5.98. Overview of the simplified routing table, pre RiOS 8.0

```
SH (config) # in-path simplified routing dest-only
SH # show in-path macmap-tables
    relay         ip_addr          mac_addr  vlan ref_count
inpath0_0        10.0.1.1 00:0d:b9:17:28:dd     0        1
inpath0_0     192.168.2.1 00:21:70:3e:6b:7f     0        1
```

Figure 5.99. Overview of the simplified routing table, RiOS 8.0 and later

```
SH # show in-path macmap-tables
    relay         ip_addr          mac_addr  vlan ref_count valid
inpath0_0        10.0.1.1 00:0d:b9:17:28:dd     0         0   Yes
inpath0_0     192.168.2.1 00:21:70:3e:6b:7f     0         0   Yes
```

So it knows that on inpath0_0 the host with IP address 10.0.1.1 is behind the gateway with MAC address d4:9a:20:c2:52:0e. But it still doesn't tell you if it is on the LAN or on the WAN side of the in-path interface, you need the system dump for that.

The file *macmap_tables* contains the table as above. The files *mactab_entries* in RiOS 5.5 and *pkttab_entries* in RiOS 6.0 and higher in the subdirectories of the `er/` directory contains the link between the MAC addresses of the gateways and the LAN or WAN side and the VLAN the gateway is in:

Figure 5.100. Matching the MAC addresses of the gateways with the LAN and WAN interfaces

```
[~/sysdumps-SH] edwin@t43>cat macmap_tables
inpath0_0        10.0.1.1 00:0d:b9:17:28:dd     0        1
inpath0_0     192.168.2.1 00:21:70:3e:6b:7f     0        1

[~/sysdumps-SH] edwin@t43>ls -al er/*/pkttab_entries
-rw-r--r--  1 edwin  edwin   681 Jun  6  2012 er/0/pkttab_entries
-rw-r--r--  1 edwin  edwin   664 Jun  6  2012 er/1/pkttab_entries
-rw-r--r--  1 edwin  edwin  5343 Jun  6  2012 er/2/pkttab_entries

[~/sysdump-SH] edwin@t43>cat er/0/pkttab_entries
ent/tot  ht[chain] p        addr           vlan   dev        type          hits  flaps p   \
     vlan_chg p   race
  1/  2   317[  0] 0  00:21:70:3e:6b:7f       0   lan0_0    EAL_DEV_LAN    0     0       0 \
         0     0     0
  2/  2   569[  0] 0  00:0d:b9:17:28:dd       0   wan0_0    EAL_DEV_WAN    509394     0    \
        0       0     0     0
```

So the host with IP address 10.0.1.1 is on inpath0_0 behind the gateway with MAC address 00:0d:b9:17:28:dd which is on the WAN side of the inpath0_0 interface.

5.16.2.3. What can go wrong?

While in theory this should be fool proof since it learns from the network there are a couple of scenarios to keep alert about.

5.16.2.3.1. Unlearning an entry from the macmap table.

Pre RiOS 7.0.4 and 8.0, the Simplified Routing table could only be cleared by rebooting the appliance. Once learned the entry could not be unlearned, only overwritten with new information.

With RiOS 7.0.4 and 8.0, the command `in-path simplified routing-entries flush <interface>` will mark all the macmap table entries for a specific in-path interface as invalid, causing them to be ignored until learned again:

Figure 5.101. Unlearning entries from the macmap tables

```
SH # in-path simplified routing-entries flush all
SH # show in-path macmap-tables
    relay         ip_addr          mac_addr  vlan ref_count valide_data   jiffies_update_l \
    ru
inpath0_0        10.0.1.1 00:0d:b9:17:28:dd     0         0    No          159168 59901600 \
    68
inpath0_0     192.168.2.1 00:21:70:3e:6b:7f     0         0    No          451650 59904525 \
    50
```

5.16.2.3.2. Interface bonding

Interface bonding consists of two interfaces on a server which listen to the same IP address. It has been the experience that they can be sending out packets with the MAC address of the non-master interface but not accepting packets send to it. Switching to the *Destination-Only* Simplified Routing setting will overcome this problem.

5.16.2.3.3. Connection Forwarding

In a Connection Forwarding deployment, always use the *Destination-Only* Simplified Routing setting.

5.16.2.3.4. Other in-path devices

Other in-path devices behind or in front of the Steelhead appliance can transparently generate the traffic, but will end up with the source MAC address of the in-path device, causing the Steelhead appliance to learn them. Use the *Destination-Only* Simplified Routing setting in this situation.

5.17. Alarms and health status

The Steelhead appliance has various alarms which indicate a problem with the Steelhead appliance in general or with the optimization service specifically.

To make sure you are informed about possible issues on the Steelhead appliances, alarms can be send out via several methods:

- SNMP Traps - Configure the SNMP trap servers under Configure -> System Settings -> SNMP Basic. You can use the CLI configuration command `snmp-server trap-test` to confirm that the SNMP trap gets delivered.
- Email alerts - Configure the email alerts under Configure -> System Settings -> Email. You can use the CLI command `email send-test` to confirm that the email delivery works.

There are six styles of lines for reporting alarm related issues:

- *Alarm triggered for rising error*: An error condition has been set because of the monitored value being too high. This is before RiOS 7.0.
- *Alarm triggered for rising clear*: An error condition caused by the monitored value being too high, has been cleared. This is before RiOS 7.0.
- *Alarm triggered for falling error*: An error condition has been set because of the monitored value being too low. This is before RiOS 7.0.
- *Alarm triggered for falling clear*: An error condition caused by the monitored value being too low, has been cleared. This is before RiOS 7.0.
- *Alarm 'X' clearing*: An error condition has been cleared. This is on RiOS 7.0 and higher.
- *Alarm 'X' triggering*: An error condition has been set. This is on RiOS 7.0 and higher.

The full output of the logging for alarms is:

Figure 5.102. Full output of a log entry for an alarm before RiOS 7.0

```
SH statsd[312]: [statsd.NOTICE]: Alarm triggered for rising error for event xxx
```

and

Figure 5.103. Full output of a log entry for an alarm on RiOS 7.0 and later

```
SH alarmd[29863]: [alarmd.NOTICE]: Alarm 'xxx' triggering
```

With the introduction of RiOS 7.0 there are aggregate alarms, which rise if one of the source alarms rises. The aggregation tree is currently:

Figure 5.104. Aggregate alarm tree for RiOS 8.0

```
health
 |\_ admission_control
 |     |\_ admission_conn
 |     |\_ admission_cpu
 |     |\_ admission_mapi
 |     |\_ admission_mem
 |      \_ admission_tcp
 |\_ arcount
 |\_ block_store
 |    \_ block_store:*
 |\_ bypass
```

```
|\_ connection_forwarding
|    |\_ cf_ipv6_incompatible_cluster
|    |\_ disconnected_sh_alert
|     \_ single_cf
|         |\_ cf_ack_timeout_aggr
|         |     \_ cf_ack_timeout:*
|         |\_ cf_conn_failure_aggr
|         |     \_ cf_conn_failure:*
|         |\_ cf_conn_lost_eos_aggr
|         |     \_ cf_conn_lost_eos:*
|         |\_ cf_conn_lost_err_aggr
|         |     \_ cf_conn_lost_err:*
|         |\_ cf_keepalive_timeout_aggr
|         |     \_ cf_keepalive_timeout:*
|         |\_ cf_latency_exceeded_aggr
|         |     \_ cf_latency_exceeded:*
|          \_ cf_read_info_timeout_aggr
|               \_ cf_read_info_timeout:*
|\_ cpu_util_indiv
|     \_ cpu:*:util
|\_ datastore
|    |\_ datastore_error
|    |\_ datastore_sync_error
|    |\_ disk_not_setup
|     \_ store_corruption
|\_ domain_join_error
|\_ duplex
|\_ flash_protection_failed
|\_ fs_mnt
|     \_ fs_mnt:*:full
|\_ granite-core
|     \_ granite-core:*
|\_ hardware
|    |\_ disk
|    |     \_ disk_error:*
|    |\_ fan_error
|    |\_ flash_error
|    |\_ ipmi
|    |\_ memory_error
|    |\_ other_hardware_error
|    |\_ power_supply
|    |\_ raid_disk_indiv
|    |     \_ disk:*:status
|     \_ ssd_wear
|          \_ ssd_wear_warning:*
|\_ high_availability
|     \_ high_availability:*
|\_ inbound_qos_wan_bw_err
|\_ iscsi
|     \_ iscsi:*
|\_ lan_wan_loop
|\_ licensing
|    |\_ appliance_unlicensed
|    |\_ autolicense_error
|    |\_ autolicense_info
|    |\_ license_expired
|     \_ license_expiring
|\_ link_duplex
|     \_ link_state:*:half_duplex
|\_ link_io_errors
|     \_ link_state:*:io_errors
|\_ linkstate
|     \_ link_state:*:link_error
|\_ lun
|     \_ lun:*
|\_ nfs_v2_v4
|\_ optimization_service
|    |\_ halt_error
|    |\_ optimization_general
|     \_ service_error
|\_ outbound_qos_wan_bw_err
|\_ paging
```

```
 |\_ path_selection_path_down
 |\_ pfs
 |    |\_ pfs_config
 |     \_ pfs_operation
 |\_ profile_switch_failed
 |\_ rsp
 |    |\_ rsp_general_alarm
 |    |\_ rsp_license_expired
 |    |\_ rsp_license_expiring
 |     \_ rsp_service
 |\_ secure_vault
 |    |\_ secure_vault_rekey_needed
 |    |\_ secure_vault_uninitialized
 |     \_ secure_vault_unlocked
 |\_ serial_cascade_misconfig
 |\_ smb_alert
 |\_ snapshot
 |     \_ snapshot:*
 |\_ ssl
 |    |\_ certs_expiring
 |    |\_ crl_error:*
 |    |\_ non_443_sslservers_detected_on_upgrade
 |     \_ ssl_peer_scep_auto_reenroll
 |\_ sticky_staging_dir
 |\_ sw_version_aggr
 |    |\_ mismatch_peer_aggr
 |    |     \_ mismatch_peer:*
 |     \_ sw_version_mismatch_aggr
 |           \_ sw_version:*
 |\_ system_detail_report
 |\_ temperature
 |    |\_ critical_temp
 |     \_ warning_temp
 |\_ uncommitted_data
  \_ vsp
      |\_ esxi_communication_failed
      |\_ esxi_disk_creation_failed
      |\_ esxi_initial_config_failed
      |\_ esxi_license
      |    |\_ esxi_license_expired
      |    |\_ esxi_license_expiring
      |     \_ esxi_license_is_trial
      |\_ esxi_memory_overcommitted
      |\_ esxi_not_set_up
      |\_ esxi_version_unsupported
      |\_ esxi_vswitch_mtu_unsupported
      |\_ virt_cpu_util_indiv
      |     \_ virt_cpu:*:util
      |\_ vsp_general_alarm
      |\_ vsp_service
      |\_ vsp_service_not_running
       \_ vsp_unsupported_vm_count
```

If an alarm in one of the end-nodes gets triggered, the nodes above it will get triggered too: So if the *warning_temp alarm* gets triggered, the *temperature alarm* will get triggered and the *health alarm* gets triggered.

The examples of log messages in this section use a shorter notation without the hostname, process name and severity.

5.17.1. General alarms

5.17.1.1. Generic health alarm

Figure 5.105. Generic health alarm

```
Alarm 'health' triggered
```

This is the top level node which gets triggered when any of the other alarms gets triggered.

5.17.1.2. Certificates related alarms

Figure 5.106. Certificates related alarms

```
Alarm 'certs_expiring' triggered
Alarm 'crl_error:SSL_CAs' triggered
Alarm 'crl_error:SSL_Peering_CAs' triggered
Alarm 'ssl' triggered
Alarm 'ssl_peer_scep_auto_enroll' triggered

Alarm triggered for rising error for event certs_expiring
Alarm triggered for rising error for event crl_error
Alarm triggered for rising error for event ssl_peer_scep_auto_enroll
```

The *certs_expiring alarm* gets triggered if any Root CA certificates or any SSL peering certificates or any SSL server certificates are going to expire in two months or already have expired.

Any expired Root CA certificates can be safely removed if they are not used by one of your SSL server certificates.

The expired SSL peering certificates need to be reissued on the Steelhead appliance itself.

The expired SSL server certificates need to be obtained again from the department which runs the SSL server and imported again on the Steelhead appliance.

Next steps: Check the expiry dates on the certificates in the Certificate Authority and the certificates in the SSL server list.

The *crl_error alarm* gets triggered when the LDAP server containing the Certificate Revocation List cannot be contacted.

Next steps: Check the connectivity and availability of the LDAP server containing the CRL list.

The *ssl_peer_scep_auto_enroll alarm* get triggered when the SCEP functionality, to enroll peering certificates, has encountered an error.

Next steps: Check the connectivity and the content of the LDAP server.

5.17.1.3. CPU load related alarms

Figure 5.107. CPU load related alarms

```
Alarm 'cpu_util_indiv' triggered
Alarm 'cpu:N:util' triggered

Alarm triggered for rising error for event cpu_util_indiv
```

These alarms get triggered when the usage on one of the CPU cores exceeds a certain threshold. The CPU usage metric are generally measured in four usage types: System, User, I/O Wait and Idle.

- System: The percentage of time the CPU spends in the kernel.
- User: The percentage of time the CPU spends in the user land (optimization service, the GUI, the CLI, the SNMP server etc).
- I/O Wait: The percentage of time the CPU is waiting for an I/O device to complete its actions. On the Steelhead appliance this is most likely to complete a read and write request to the hard disk.
- Idle: The percentage of time the CPU is waiting for interrupts to process.

Normally the CPU usage pattern should be more-or-less equal on all CPUs. It can be different if:

- In a data recovery scenario with few TCP sessions which are all handled on one single CPU. This will show as a high User usage on one of the CPUs and not on the others.

  This can be changed with the option Multi-Core Balancing at Configure -> Optimization -> Performance.

- PBR, WCCP and Interceptor redirected traffic is handled by a single thread, therefore it will be done on a single CPU. This will show up as a lot of System usage on one of the CPUs. For PBR and WCCP this can be resolved by only redirecting the traffic to be optimized and not the pass-through traffic.

  On a 10Gbps bypass card, this issue has been resolved since RiOS 8.0 by distributing this load over multiple CPUs.

- If there is a problem with one of the threads in the optimization service. This will show as a lot of user-land usage on one of the CPUs. Please open a case with Riverbed TAC to troubleshoot this.

The following reasons could be the reason for a generic high CPU utilization:

- If the CPU usage is mostly I/O Wait, then either one of the disks in the appliance has operational problems or there is a lot of encrypted traffic being optimized which causes a large amount of searching through the data store and a lot of writing of new segments. Inspecting the Traffic Summary and the SMART related part of the system dump would be the next steps.

- If the CPU usage is on average at 80-90% and the CPU pattern follows the traffic pattern, then it could just be that the machine is underpowered for the traffic load.

Next steps: Open a case with Riverbed TAC to determine the reason for the high CPU load.

5.17.1.4. License alarms

Figure 5.108. License alarms

```
Alarm 'appliance_unlicensed' triggered
Alarm 'license_expired' triggered
Alarm 'license_expiring' triggered
Alarm 'licensing' triggered
Alarm 'autolicense_error' triggered
Alarm 'autolicense_info' triggered

Alarm triggered for rising error for event license
Alarm triggered for rising error for event license
```

The appliance unlicensed alarm is raised when there is no MSPEC license on the xx55 and xx60 platforms.

One or more evaluation license keys on the Steelhead appliance are expired or about to expire.

Figure 5.109. RSP License alarms

```
Alarm 'rsp_license_expired' triggered
Alarm 'rsp_license_expiring' triggered

Alarm triggered for rising error for event rsp_license_expired
Alarm triggered for rising error for event rsp_license_expiring
```

These alarms get triggered when the evaluation RSP licenses are about to expire or already have expired.

When an RSP instance gets initially setup in an evaluation environment, it will get a time-limited RSP license. When the evaluation is finished and an RSP license is purchased and configured but the evaluation license which has not been removed will expire and this alarm is raised.

Next steps: Remove the temporary licenses, configure the proper licenses or stop the RSP service.

5.17.1.5. Disk alarms

Figure 5.110. Disk alarms

```
Alarm 'raid_disk_indiv' triggered
Alarm 'disk:X:status' triggered
Alarm 'disk' triggered
Alarm 'disk_error:X' triggered
Alarm 'ssd_wear' triggered
Alarm 'ssd_wear_warning:X' triggered

Alarm triggered for rising error for event disk_error
Alarm triggered for rising error for event disk_not_setup
Alarm triggered for rising error for event raid_error
Alarm triggered for rising error for event ssd_wear_warning
```

These alarms get triggered when a hard disk has high SMART error-rate or when a hard disk in a RAID array has failed. The *ssd_wear alarm* is related to the 5055, 7050 and 7055 models where the number of writes to a solid-state disk in the Fault Tolerant Segstore has exceeded the threshold.

Next steps: Contact Riverbed TAC for the replacement of the hard disk.

5.17.1.6. Domain joining alarm

Figure 5.111. Domain joining alarms

```
Alarm 'domain_join_error' triggered

Alarm triggered for rising error for event domain_join_error
```

This alarm gets triggered when the Steelhead appliance has been joined to the domain but the communication with one or more Domain Controllers has been interrupted.

Next steps: Check the settings for the domain and perform the domain join again.

5.17.1.7. Filesystem related alarm

Figure 5.112. File system alarms

```
Alarm 'fs_mnt' triggered
Alarm 'fs_mnt:X:full' triggered

Alarm triggered for rising error for event fs_mnt
```

This alarm gets triggered when the usage of one of the partitions are above a threshold or when it is completely filled up.

For the /var partition, there can be various reasons for this:

- Too many system dumps or process dumps have been created. You can remove them via the GUI under Reports -> Diagnostics -> System Dumps or in the CLI via the command `files debug ... delete` and `files snapshot ... delete`.
- A background tcpdump capture has captured too much data and has filled up the partition.
- The logging level has been set too high, most likely INFO level, and the rotation-interval of the log files has been changed.
- The rotation of the log files has failed. This can be determined by checking the dates on the earlier log files to spot a skip in time between them.
- For older RiOS versions: The log files of the neural framing algorithm are filling up the partition or the rotation of the log files has failed due to a race condition between two logrotate processes.

If the usage was 100%, then once the disk space has been reclaimed the best next step is to reboot the appliance so that all processes and all log files are recreated properly.

For the /proxy partition, in use by the RSP system, there can be various solutions for this:

- Remove old RSP installation images.
- Remove old RSP packages.
- Remove old RSP slots.

Note that the optimization service itself isn't affected by a full-disk situation since the data store is located on its own partition.

Next steps: If the space cannot be reclaimed via the removal of system dumps, process dumps or RSP related files, then contact Riverbed TAC for further investigation.

5.17.1.8. Paging alarms

Figure 5.113. Paging alarms

```
Alarm 'paging' triggered

Alarm triggered for rising error for event paging
```

This alarm gets triggered when there is excessive swapping happening.

Next steps: Do not reset the device and contact Riverbed TAC to analyse this issue.

5.17.1.9. RSP / VSP Alarms

Figure 5.114. RSP / VSP General alarms

```
Alarm 'rsp' triggered
Alarm 'rsp_general_alarm' triggered
Alarm 'rsp_service' triggered
Alarm 'virt_cpu_util_indiv' triggered
Alarm 'virt_cpu:*:util
Alarm 'vsp' triggered
Alarm 'vsp_general_alarm' triggered
Alarm 'vsp_service' triggered
Alarm 'vsp_service_not_running' triggered
Alarm 'vsp_unsupported_vm_count' triggered

Alarm triggered for rising error for event rsp_general_alarm
```

These alarms get triggered when the VSP or RSP service has experienced a problem.

For RSP on the xx20 and xx50 series models, most likely this will be an incompatibility between the RiOS version installed and the RSP version installed.

Next steps: Install the correct RSP version for this RiOS release.

For VSP on the EX series models, the issue could be related to communication between RiOS and the ESXi infrastructure.

Figure 5.115. ESXi specific alarms

```
Alarm 'esxi_communication_failed' triggered
Alarm 'esxi_disk_creation_failed' triggered
Alarm 'esxi_initial_config_failed' triggered
Alarm 'esxi_license' triggered
Alarm 'esxi_license_expired' triggered
Alarm 'esxi_license_expiring' triggered
Alarm 'esxi_license_is_trial' triggered
Alarm 'esxi_memory_overcommitted' triggered
Alarm 'esxi_not_set_up' triggered
Alarm 'esxi_version_unsupported' triggered
Alarm 'esxi_vswitch_mtu_unsupported' triggered
```

These alarms are related to the ESXi part of the EX appliances.

The *esxi_not_set_up alarm* happens when an EX appliance doesn't have the VSP service enabled yet.

Next step: Enable the VSP service in the GUI under Configure -> Virtualization -> Virtual Services Platform.

The *esxi_disk_creation_failed alarm* and the *esxi_initial_config_failed alarm* happen when the initial setup has failed.

Next steps: Contact the Riverbed TAC.

The *esxi_communication_failed alarm* happens when the communication towards the ESXi platform doesn't work anymore. Most likely reason is that the password has been changed in the ESXi infrastructure.

Next steps: Update the password in the VSP configuration.

The *esxi_memory_overcommitted alarm* happens when the memory required by the ESXi system is less than what is available.

Next steps: Reduce the memory requirements of the VMs in the ESXi system.

The *esxi_version_unsupported alarm* happens when the version of ESXi has changed due to patches.

Next steps: Back out to the original ESXi version.

The *esxi_vswitch_mtu_unsupported alarm* is raised when the vSwitch on the ESXi platform has an MTU size of more than 1500.

Next steps: Undo the MTU size changes on the vSwitch.

Figure 5.116. RSP License alarms

```
Alarm 'rsp_license_expired' triggered
Alarm 'rsp_license_expiring' triggered

Alarm triggered for rising error for event rsp_license_expired
Alarm triggered for rising error for event rsp_license_expiring
```

These alarms get triggered when the evaluation RSP licenses are about to expire or already have expired.

When an RSP instance gets initially setup in an evaluation environment, it will get a time-limited RSP license. When the evaluation is finished and an RSP license is purchased but not installed, the evaluation license will expire and the RSP service will not restart at the next restart.

Next steps: Remove the temporary licenses, configure the proper licenses or stop the RSP service.

5.17.1.10. Secure Vault alarm

Figure 5.117. Secure Vault alarms

```
Alarm 'secure_vault' triggered
Alarm 'secure_vault_rekey_needed' triggered
Alarm 'secure_vault_uninitialized' triggered
Alarm 'secure_vault_unlocked' triggered

Alarm triggered for falling error for event secure_vault_unlocked
```

The *secure vault* alarm is a general alarm when one of the other alarms get triggered.

The *secure vault unlocked* alarm gets triggered when the secure vault cannot be opened with the default password.

This happens when the password for the secure vault has been changed: After a restart of the Steelhead appliance the secure vault cannot be automatically opened anymore and this alarm gets triggered.

Next steps: Unlock the secure vault manually via the GUI of the Steelhead appliance or configure the correct secure vault password on the CMC.

5.17.2. Hardware related alarms

5.17.2.1. Fan alarms

Figure 5.118. Fan alarms

```
Alarm 'fan_error' triggered

Alarm triggered for rising error for event fan_error
```

This alarm gets triggered when one of the fans in the chassis has failed.

Next steps: Contact Riverbed TAC for the replacement of the fan.

5.17.2.2. Flash Error alarms

Figure 5.119. Flash Error alarms

```
Alarm 'flash_error' triggered
Alarm 'flash_protection_failed' triggered

Alarm triggered for rising error for event flash_error
```

The *flash_error alarm* gets triggered when the USB Flash Memory has become read-only or when it has become unavailable.

This is an issue with the xx50 series models, where the USB bus towards the Flash Memory becomes locked due to timing issues.

Checkout KB article S15568 to confirm that the device is running the right minimum RIOS version to best overcome this issue.

Next steps: Reboot the appliance as soon as possible to unlock the USB Flash Memory.

The *flash_protection_failed alarm* is raised when the backup of the USB Flash Memory could not have been completed.

Next steps: Confirm that there is enough free space on the */var* partition for the backup.

5.17.2.3. Generic hardware alarm

Figure 5.120. Generic Hardware alarms

```
Alarm 'hardware' triggered
Alarm 'other_hardware_error' triggered

Alarm triggered for rising error for event hardware_error
```

These alarms get triggered when there is a mismatch with the configuration of the Steelhead appliance:

- A faulty disk, insufficient memory or missing CPUs.
- An unqualified hard disk, memory stick or network card has been detected.

Next steps: Contact Riverbed TAC for investigation.

5.17.2.4. IPMI alarms

Figure 5.121. IPMI alarms

```
Alarm 'ipmi' triggered

Alarm triggered for rising error for event ipmi
```

This alarm gets triggered when the IPMI subsystem reports an issue:

- The chassis of the Steelhead appliance has been opened,
- The ECC memory has reported an error,
- A hard disk is failing or has failed,
- A power supply is failing or has failed.

Next steps: If the alarm is with regarding to the intrusion alarm, it can be reset from the GUI. If the issue is with regarding to failing hardware, then contact Riverbed TAC for a replacement.

5.17.2.5. Memory error alarms

Figure 5.122. Memory error alarms

```
Alarm 'memory_error' triggered
```

```
Alarm triggered for rising error for event memory_error
```

This alarm gets triggered when an ECC error on one of the memory sticks gets reported.

Next steps: Contact Riverbed TAC for identification and replacement.

5.17.2.6. Power supply alarms

Figure 5.123. Power supply alarms

```
Alarm 'power_supply' triggered
```

```
Alarm triggered for rising error for event power_supply
```

This alarm gets triggered when one of the power supplies in the chassis has failed.

Next steps: Confirm that the power source to the power supply is working properly, if not contact Riverbed TAC for a replacement.

5.17.2.7. SSL Hardware alarm

Figure 5.124. SSL Hardware alarm

```
Alarm triggered for rising error for event ssl_hardware
```

This alarm gets triggered when the SSL Offload hardware has encountered an error.

Next steps: Contact Riverbed TAC for investigation.

5.17.2.8. Temperature alarms

Figure 5.125. Temperature alarms

```
Alarm 'temperature' triggered
Alarm 'critical_temp' triggered
Alarm 'warning_temp' triggered

Alarm triggered for rising error for event warning_temp
Alarm triggered for rising error for event critical_temp
```

These alarms get triggered when the temperature of the Steelhead appliance is too high. Possible causes are fan-failures, high operating temperatures, high humidity, blocked air vents or dusty environment.

Next steps: Confirm that the environment temperature is not too high, that the airflow is sufficient, clear the chassis of excessive dust. If the issue keeps happening, contact Riverbed TAC for investigation.

5.17.3. Network related alarms

5.17.3.1. Asymmetric Routing alarms

Figure 5.126. Asymmetric routing alarms

```
Alarm 'arcount' triggered
```

```
Alarm triggered for rising error for event arcount
```

This alarm gets triggered if one or more instances of asymmetric routing have been detected. See the chapter IP Routing Related Issues on how to deal with this.

Next steps: If the cause of the alarm is real network asymmetry, then make sure that all paths in the network are covered with Steelhead appliances. If the cause of the alarm is SYN retransmission, then configure Fixed-Target in-path rules to overcome the auto-discovery issue, or configure pass-through in-path rules to prevent optimization towards that subnet.

5.17.3.2. Bypass alarms

Figure 5.127. Bypass alarms

```
Alarm 'bypass' triggered
```

```
Alarm triggered for rising error for event bypass
```

This alarm gets triggered when one or more in-path interfaces have gone into bypass, linking the WAN router and LAN interface together. This happens when the optimization service is stopped or when the network card watchdog has been activated.

Next steps: Restart the optimization service if the optimization service is stopped. Contact Riverbed TAC to investigate if the issue is related to the watchdog.

5.17.3.3. Duplex alarms

Figure 5.128. Duplex error

```
Alarm 'duplex' triggered
Alarm 'link_duplex' triggered
Alarm 'link_state:*:half_duplex' triggered
Alarm 'link_io_errors' triggered
Alarm 'link_state:*:io_errors' triggered
```

```
Alarm triggered for rising error for event duplex
```

These alarms gets triggered when there is a large amount of frame errors or carrier errors on one of the interfaces, often indicating a speed/duplex configuration mismatch.

Next steps: Check the speed and duplex settings on the interface reported. Consider a new Ethernet cable or a different port on the switch if the speed and duplex settings are the same.

5.17.3.4. Link state alarms

Figure 5.129. Link state alarms

```
Alarm 'linkstate' triggered
Alarm 'link_state:X:link_error' triggered
```

```
Alarm triggered for rising error for event linkstate
```

These alarms get triggered when one of the interfaces of the Steelhead appliance has lost its Ethernet link.

Next steps: Investigate the loss of Ethernet link.

5.17.4. Virtual Steelhead related alarms

Figure 5.130. Virtual Steelhead related alarms

```
Alarm 'disk_not_setup' triggered
Alarm 'lan_wan_loop' triggered
```

The *disk_not_setup alarm* gets triggered when the Virtual Steelhead appliance detects that the disk reserved for the data store is not yet provisioned or not of the right size.

Next steps: In the ESXi environment, check the properties of the virtual disks for the Virtual Steelhead.

The *lan_wan_loop alarm* gets triggered when the Virtual Steelhead appliance detects that the LAN and the WAN interface are configured to be on the same virtual switch.

Next steps: In the ESXi environment, put the LAN and the WAN interfaces in different virtual switches.

5.17.5. Optimization service alarms

5.17.5.1. Admission control alarms

Figure 5.131. Various admission control alarms

```
Alarm 'admission_control' rising
Alarm 'admission_conn' rising
Alarm 'admission_cpu' rising
Alarm 'admission_mapi' rising
Alarm 'admission_mem' rising
Alarm 'admission_tcp' rising

Alarm triggered for rising error for event admission_conn
Alarm triggered for rising error for event admission_cpu
Alarm triggered for rising error for event admission_mapi
Alarm triggered for rising error for event admission_mem
Alarm triggered for rising error for event admission_tcp
```

The following admission control alarms do exist:

- The generic admission control alarm, it is raised when one of the other alarms is raised.
- Connection based admission control, when the number of TCP sessions optimized has increased to above the number of licensed TCP sessions.

  Next steps: Check the list of TCP connections in the Current Connections overview and to reduce the number of optimized TCP sessions by applying in-path rules to pass-through traffic which does not get a high optimization factor. If the issue is structural consider an upgrade to a higher model.
- CPU load admission control, when the CPU load is too high and no new TCP sessions will be optimized.

  Next steps: Open a case with Riverbed TAC when this happens.
- MAPI based admission control, when the number of optimized TCP sessions exceeds a by default 85% threshold of the maximum number of licensed TCP sessions. This is to overcome the problems caused by the MAPI protocol use of multiple TCP sessions which all need to be optimized against the same client-side and server-side Steelhead appliances.

  Next steps: The same as with Connection based admission control.

- Memory based admission control, where the internal memory pool of the optimization service has exceeded a certain threshold.

  Next steps: Open a case with Riverbed TAC to investigate.

- TCP based admission control, where the TCP buffers in the kernel running on the Steelhead appliance is running out of memory.

  Next steps: In the sysinfo.txt in the system dump, check the netstat output and find out which hosts have the biggest send-queue values. That is the host with the slow network stack. Also, open a case with Riverbed TAC to investigate.

5.17.5.2. Connection Forwarding related alarms

Figure 5.132. Connection Forwarding alarms

```
Alarm 'cf_ipv6_incompatible_cluster' triggered
Alarm 'connection_forwarding' triggered
Alarm 'disconnected_sh_alert' triggered
Alarm 'single_cf' triggered
Alarm 'cf_ack_timeout_aggr' triggered
Alarm 'cf_ack_timeout:*' triggered
Alarm 'cf_conn_failure_aggr' triggered
Alarm 'cf_conn_failure:*' triggered
Alarm 'cf_conn_lost_eos_aggr' triggered
Alarm 'cf_conn_lost_eos:*' triggered
Alarm 'cf_conn_lost_err_aggr' triggered
Alarm 'cf_conn_lost_err:*' triggered
Alarm 'cf_keepalive_timeout_aggr' triggered
Alarm 'cf_keepalive_timeout:*' triggered
Alarm 'cf_latency_exceeded_aggr' triggered
Alarm 'cf_latency_exceeded:*' triggered
Alarm 'cf_read_info_timeout_aggr' triggered
Alarm 'cf_read_info_timeout:*' triggered

Alarm triggered for rising error for event cf_ack_timeout
Alarm triggered for rising error for event cf_conn_failure
Alarm triggered for rising error for event cf_conn_lost_eos
Alarm triggered for rising error for event cf_conn_lost_err
Alarm triggered for rising error for event cf_keepalive_timeout
Alarm triggered for rising error for event cf_latency_exceeded
Alarm triggered for rising error for event cf_read_info_timeout
```

The most common alarms with regarding to Connection Forwarding protocol are the *cf_conn_failure alarm* and the *cf_latency_exceeded alarm*.

The *cf_conn_failure alarm* happens when the Steelhead appliance is unable to setup of the TCP session for the Connection Forwarding protocol. This is most likely related to a network related issue between the two in-path interfaces or because the optimization service on the other Steelhead appliance is not operational.

The *cf_latency_exceeded alarm* happens when the responses in the Connection Forwarding sessions take too long to come back to the Steelhead appliance. This could be related to the network between the two in-path interfaces but also to the load on the neighbour Steelhead appliance.

Next steps: Check the network between the in-path interfaces.

The *disconnected_sh_alert alarm* gets triggered when a Connection Forwarding neighbour gets disconnected.

The *single_cf alarm* gets triggered when there are no neighbouring nodes in the Connection Forwarding cluster.

Next steps: Determine why the neighbours got disconnected.

The *cf_ipv6_incompatible_cluster alarm* gets triggered when one of the nodes in the cluster do not support optimization over IPv6 yet.

Next steps: Upgrade all nodes in the cluster to the right RiOS version.

5.17.5.3. Data store related alarms

Figure 5.133. Data store related alarms

```
Alarm 'datastore' triggered
Alarm 'datastore_error' triggered
Alarm 'datastore_sync_error' triggered
Alarm 'store_corruption' triggered

Alarm triggered for rising error for event datastore_error
Alarm triggered for rising error for event datastore_sync_error
Alarm triggered for rising error for event store_corruption
```

When the optimization service detects a corruption in the data store it try to recover from the issue and raise the *store_corruption alarm*. This alarm also can happen when encryption of the data store has been enabled or disabled but the restart of the optimization service didn't include the clearing of the data store.

Next steps: Restart the optimization service and clear the data store.

The *datastore_sync_error alarm* gets triggered when the Steelhead appliance is part of a data store synchronization cluster but its peer has become unreachable.

Next steps: Confirm the network between the two Steelhead appliances.

The *datastore_error alarm* gets triggered when the metadata in the data store cannot be initialized with the current settings. This alarm can happen when encryption of the data store has been changed or when the Extended Peering Table has been enabled or disabled but the restart of the optimization service didn't include the clearing of the data store.

Next steps: Restart the optimization service with the option to clear the data store.

5.17.5.4. Halt alarm

Figure 5.134. Halt alarms

```
Alarm 'halt_error' triggered

Alarm triggered for rising error for event halt_error
```

Next steps: Open a case with Riverbed TAC to investigate.

5.17.5.5. Mismatched Peer alarm

Figure 5.135. Mismatched Peer alarms

```
Alarm triggered for rising error for event mismatch_peer
```

Next steps: Open a case with Riverbed TAC to investigate.

5.17.5.6. NFS related alarm

Figure 5.136. NFS related alarms

```
Alarm 'nfs_v2_v4' triggered

Alarm triggered for rising error for event nfs_v2_v4
```

This alarm gets triggered when the NFS latency optimization has detected an NFS server which only uses the NFS version 2 or NFS version 4 protocols. The NFS latency optimization can only optimize NFS version 3 traffic.

Next steps: See if the NFS servers reported can be changed to support NFS version 3. If not, consider a pass-through rule for them.

5.17.5.7. Optimization Service alarm

Figure 5.137. Optimization Service alarms

```
Alarm 'service_error' triggered
Alarm 'optimization_service' triggered
Alarm 'optimization_general' triggered

Alarm triggered for rising error for event service_error
```

These alarms get triggered when the optimization service is not running or when an issue has been encountered which indicate a critical failure in the optimization protocol.

Next steps: Open a case with Riverbed TAC to investigate.

5.17.5.8. PFS related alarms

Figure 5.138. PFS related alarms

```
Alarm 'pfs' triggered
Alarm 'pfs_config' triggered
Alarm 'pfs_operation' triggered

Alarm triggered for rising error for event pfs_config
Alarm triggered for rising error for event pfs_operation
```

These alarms get triggered when there is an issue with the Proxy File Service.

Next steps: Open a case with Riverbed TAC to investigate.

5.17.5.9. Process dump related alarm

Figure 5.139. Process dump related alarms

```
Alarm 'sticky_staging_dir' triggered

Alarm triggered for rising error for event sticky_staging_dir
```

This alarm gets triggered when there is an issue creating new process dumps. Most of the time it correlates with a *fs_mnt alarm*.

Next steps: Open a case with Riverbed TAC to investigate.

5.17.5.10. Serial Cascade configuration alarm

Figure 5.140. Serial Cascade configuration alarms

```
Alarm 'serial_cascade_misconfig' triggered

Alarm triggered for rising error for event serial_cascade_misconfig
```

Next steps: Open a case with Riverbed TAC to investigate.

5.17.5.11. SMB alarm

Figure 5.141. SMB alarms

```
Alarm 'smb_alert' triggered

Alarm triggered for rising error for event smb_alert
```

This alarm gets triggered when there is an issue with the SMB Signing feature.

Next steps: Check the status of the computer object in the Active Directory service and the log files on the Domain Controllers. Check the status of the delegation user in the Active Directory service and the log files on the Domain Controllers. Check the connectivity between the Steelhead appliance primary interface and the Domain Controllers.

5.17.5.12. QoS related alarms

Figure 5.142. QoS alarms

```
Alarm 'inbound_qos_wan_bw_err' triggered
Alarm 'outbound_qos_wan_bw_err' triggered
```

These alarms get raised when the sum of the configured QoS bandwidth values exceed the configured WAN bandwidth speeds.

Next steps: Check the QoS settings page and confirm that the QoS bandwidth values are correct.

5.17.5.13. Software version alarm

Figure 5.143. Software version alarms

```
Alarm 'sw_version_aggr' triggered
Alarm 'mismatch_peer_aggr'
Alarm 'mismatch_peer:*'
Alarm 'sw_version_mismatch_aggr'
Alarm 'sw_version:*'

Alarm triggered for rising error for event sw-version
```

These alarms get triggered when an incompatibility with the RiOS software version of a remote Steelhead appliance has been detected.

Next steps: Upgrade the remote Steelhead appliance to a compatible RiOS version or configure Peering Rules to prevent optimization between the two Steelhead appliances.

5.17.5.14. SSL related alarm

Figure 5.144. SSL related alarms

```
Alarm 'non_443_ssl_servers_detected_on_upgrade' triggered

Alarm triggered for rising error for event non_443_ssl_servers_detected_on_upgrade
```

This alarm gets triggered when, after the upgrade to RiOS version 6.0 or later, an SSL server on a port different than port 443 has been detected.

In RiOS version 6.0 and later traffic on TCP port 443 is automatically assumed to be SSL encapsulated and when SSL optimization has been enabled, then no in-path rule is required to get that traffic optimized. For SSL traffic on a different TCP port than port 443 an in-path rule is still required.

Next steps: Configure the correct In-path Rules and Peering Rules to optimize this SSL traffic.

5.17.5.15. System Detail Report alarm

Figure 5.145. System Detail Report alarms

```
Alarm 'system_detail_report' triggered

Alarm triggered for rising error for event system_detail_report
```

This alarm gets triggered when an issue with the operation or configuration of the optimization service has been determined.

Next steps: Check the System Detail Report and attend to the issues reported.

5.17.6. Granite related alarms

Figure 5.146. Granite related alarms

```
Alarm 'block_store' triggered
Alarm 'block_store:*' triggered
Alarm 'profile_switch_failed' triggered
Alarm 'granite-core' triggered
Alarm 'granite-core:*' triggered
Alarm 'high_availability' triggered
Alarm 'high_availability:*' triggered
Alarm 'iscsi' triggered
Alarm 'iscsi:*' triggered
Alarm 'lun' triggered
Alarm 'lun:*' triggered
Alarm 'snapshot' triggered
Alarm 'snapshot:*' triggered
Alarm 'uncommitted_data' triggered
```

The *block_store alarm* gets triggered when there are issues with the Granite block store.

Next steps: Check the system logs and contact Riverbed TAC for investigation.

This alarm gets triggered when repartitioning the drives fails during the switching of the storage profile.

Next steps: Contact Riverbed TAC for investigation.

The *granite-core alarm* gets triggered when there are communication problems towards the Granite Core appliance.

Next steps: Confirm that there are no network related issues between the Granite Edge and the Granite Core.

The *high_availability alarm* gets triggered when there are communication problems towards a node in high availability cluster.

Next steps: Check connectivity with the HA peer.

The *iscsi alarm* gets triggered when the iSCSI initiator isn't accessible.

Next steps: Confirm the configuration of the iSCSI configuration.

The *lun alarm* gets triggered when a LUN isn't avaiable for the Granite Core.

Next step: Confirm the status of the LUN on the Data Center NAS.

The *snapshot alarm* gets triggered when a snapshot couldn't be completed or committed.

Next steps: Confirm the status of the Data Center NAS.

This alarm gets triggered when the Granite Edge has a large amount of uncommitted data in its blockstore.

Next steps: Confirm that the Granite Core and the NAS are working fine.

5.18. Long lived TCP session treatment

When the optimization service gets restarted or a Steelhead appliance gets rebooted, the current optimized TCP sessions will be terminated. During the initialization period of the optimization service new TCP sessions will be setup, either new ones or the replacement of the ones which were terminated earlier. When the initialization of the optimization service is completed and new TCP sessions will be optimized, these already established TCP sessions will be passed-through, showing up as pre-existing in the list of current connections.

This is fine for short lived TCP sessions such as HTTP, SMPT and POP3, which last only a short time. However, long lived TCP sessions like CIFS sessions, MAPI traffic, database replication and file server replication will stay unoptimized and suffer from a bad performance.

To overcome this, the optimization service needs to know which traffic needs to be optimized at all times. It will require regular interaction with the people who run the services to keep track of changes in their servers. It will need to be documented properly for operation of the network.

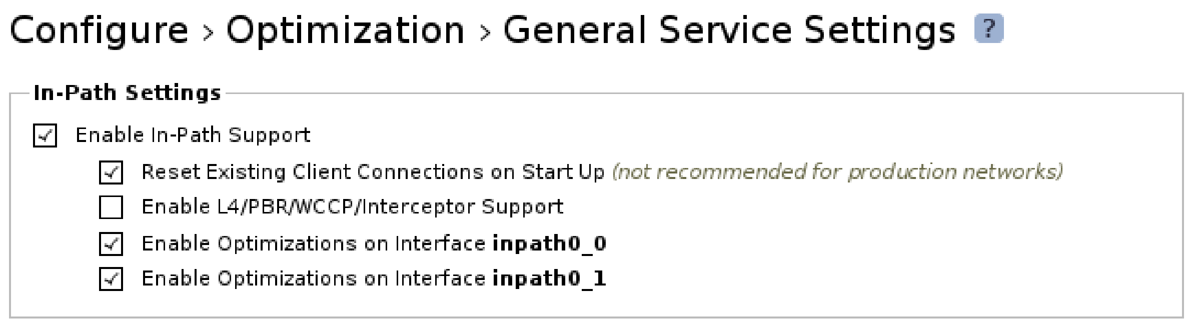
To restart these TCP sessions, there are three possible solutions from the Steelhead appliances perspective:

- Reset TCP connections at restart.
- Manual termination of TCP sessions.
- Auto kick-off of pre-existing TCP sessions via an in-path rule.

5.18.1. Reset TCP connections at restart

This option can be found under the General Optimization configuration: Reset Connections At Startup.

Figure 5.147. Reset Connections At Startup



Configure > Optimization > General Service Settings

In-Path Settings

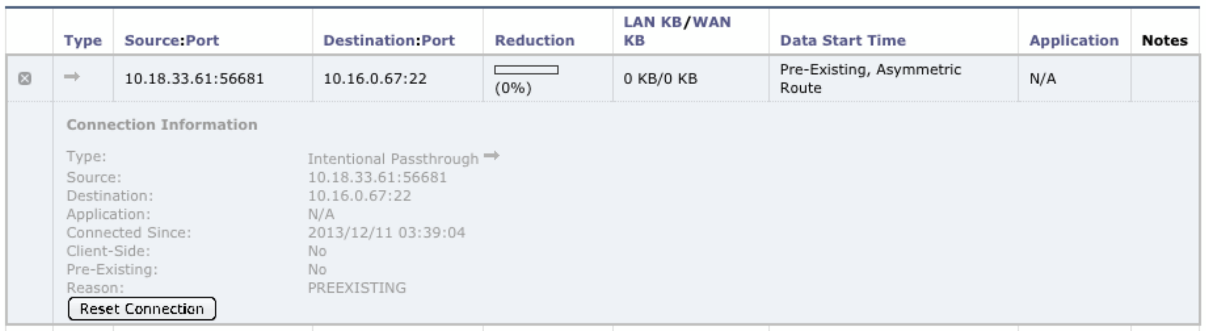
- [✓] Enable In-Path Support
  - [✓] Reset Existing Client Connections on Start Up *(not recommended for production networks)*
  - [ ] Enable L4/PBR/WCCP/Interceptor Support
  - [✓] Enable Optimizations on Interface **inpath0_0**
  - [✓] Enable Optimizations on Interface **inpath0_1**

It will cause all pre-existing TCP sessions to be reset when they are detected after the restart of the optimization service. This is obsoleted by the auto kick-off in-path rule.

5.18.2. Manual termination of TCP sessions

Under the details of the TCP session in the report of the Current Connections, there is an option to reset the connection.

Figure 5.148. Terminate Connection button in the GUI



	Type	Source:Port	Destination:Port	Reduction	LAN KB/WAN KB	Data Start Time	Application	Notes
⊠	→	10.18.33.61:56681	10.16.0.67:22	(0%)	0 KB/0 KB	Pre-Existing, Asymmetric Route	N/A	

Connection Information

Type:	Intentional Passthrough →
Source:	10.18.33.61:56681
Destination:	10.16.0.67:22
Application:	N/A
Connected Since:	2013/12/11 03:39:04
Client-Side:	No
Pre-Existing:	No
Reason:	PREEXISTING

In the CLI, the command to terminate the TCP session is `tcp connection send reset pass-reset`. It will require the IP addresses and the TCP port numbers of both the client and the server.

Figure 5.149. Terminate TCP session via the CLI

```
SH # show connections passthrough filter pre_existing
T  Source               Destination          App    Rdn Since
--------------------------------------------------------------------------------
PI 10.0.1.1         1025 192.168.1.1       139           pre_existing        c
--------------------------------------------------------------------------------
Pass Through (P):              1
PI = Passthrough Intentional
PU = Passthrough Unintentional
Forwarded (F):                 0
--------------------------------
Total:                         1
c = Client Steelhead
s = Server Steelhead

SH (config) # tcp connection send pass-reset source-addr 10.0.1.1 source-port 1025 dest-ad \
    dr 192.168.1.1 dest-port 139
```

5.18.3. In-path auto kick-off of pre-existing TCP sessions

The automatic way to terminate selected pre-existing TCP sessions after a restart of the optimization service can be done via an in-path rule on the client-side Steelhead appliance with the auto kick-off option enabled.

For example, to retry to optimize pre-existing CIFS sessions, which are normally long-lived, use the following in-path rule:

Figure 5.150. Example of auto kick-off in-path rule

```
SH (config) # in-path rule auto-discover auto-kickoff enable dstport 445 rulenum 4 descrip \
    tion "Always retry pre-existing CIFS connections"
```

Figure 5.151. Configuration of a Kick-off in-path rule

```
SH (config) # show in-path rules
 Rule Type P O L N W K VLAN Source Addr        Dest Addr          Port
----- ---- - - - - - - ---- ------------------ ------------------ --------------
    1 pass - - - - - - all  all                all                Secure
    2 pass - - - - - - all  all                all                Interactive
    3 pass - - - - - - all  all                all                RBT-Proto
    4 auto N F F A C Y all  all                all                445
      desc: Always retry pre-existing CIFS connections
  def auto N F F A C N all  all                all                all

4 user-defined rule(s)

(P) Preoptimization Policy: O=Oracle-Forms S=SSL +=Oracle-Forms-over-SSL N=None
(O) Optimization Policy:    F=Full S=SDR-only C=Compression-only M=SDR-M N=None
(L) Latency Optimizations:  F=Full H=HTTP-only O=Outlook-anywhere N=None
(N) Neural Framing:         A=Always D=Dynamic T=TCP hints N=Never
(W) WAN Visibility Mode:    C=Correct-Addressing
                            P=Port-Transparency
                            F=Full-Transparency
                            R=Full-Transparency w/Reset
(K) Auto Kickoff:           Y=Enabled
                            N=Disabled
```

Pre-existing TCP sessions on TCP port 445 will now be terminated the optimization service.

5.19. Order of the in-path rules

By default, there are three standard pass-through in-path rules on a Steelhead appliance and one default catch-all. The best way to add new in-path rules is:

- Any port specific rules go at the beginning.
- Any global rules go at the end.
- Any subnet specific rules go after the default rules but before the global rules.

These are the default in-path rules.

Figure 5.152. Default in-path rules

```
No Type         From To  Ports
 1 Pass-through Any  Any Secure
 2 Pass-through Any  Any Interactive
 3 Pass-through Any  Any RBT-proto
```

Any new port specific rules should be added before these three. For example to pass-through all traffic on TCP port 12345, add rule 1:

Figure 5.153. In-path rules: Add a new pass-through rule

```
No Type         From To  Ports
 1 Pass-through Any  Any 12345
 2 Pass-through Any  Any Secure
 3 Pass-through Any  Any Interactive
 4 Pass-through Any  Any RBT-proto
```

Any global "Use this WAN visibility" should be added to the end as rule 5:

Figure 5.154. In-path rules: Change the default WAN visibility

```
No Type            From To   Ports        WAN
 1 Pass-through    Any  Any  12345
 2 Pass-through    Any  Any  Secure
 3 Pass-through    Any  Any  Interactive
 4 Pass-through    Any  Any  RBT-proto
 5 Auto-Discovery  Any  Any  All          FT
```

Any specific optimization or auto-discovery features towards a specific IP subnet should be added after the standard pass-through rules as rule 6:

Figure 5.155. In-path rules: Add an all-ports auto-discovery rule

```
No Type            From To              Ports       WAN
 1 Pass-through    Any  Any             12345
 2 Pass-through    Any  Any             Secure
 3 Pass-through    Any  Any             Interactive
 4 Pass-through    Any  Any             RBT-proto
 5 Auto-Discovery  Any  192.168.1.0/24  All         FT,FW-RST
 6 Auto-Discovery  Any  Any             All         FT
```

Any specific optimization features towards a specific TCP port on a specific IP subnet or host could be added after the standard pass-through rules but can be in front of the pass-through rules, for example as rule 1:

Figure 5.156. In-path rules: Add an specific TCP port auto-discovery rule

```
SH # show in-path rules
No Type            From To              Ports       WAN        LatOpt
 1 Auto-Discovery  Any  192.168.1.1/32  1080        FT         HTTP
 2 Pass-through    Any  Any             12345
 3 Pass-through    Any  Any             Secure
 4 Pass-through    Any  Any             Interactive
 5 Pass-through    Any  Any             RBT-proto
 6 Auto-Discovery  Any  192.168.1.0/24  All         FT,FW-RST  Normal
 7 Auto-Discovery  Any  Any             All         FT         Normal
```

5.19.1. Specific in-path rules rules

5.19.1.1. MAPI latency optimization

Latency optimization of TCP sessions for the MAPI protocol gets determined by the traffic going to the port-mapper running on TCP port 135 on the Exchange server. To disable MAPI latency optimization via an in-path rule, use TCP port 135 in the in-path rule.

Figure 5.157. Disable MAPI optimization

```
No Type            From To              Ports       WAN        LatOpt
[...]
 5 Pass-through    Any  Any             135
```

Once a Exchange server has been detected, a hidden in-path rule gets added to the list which states that all traffic to that server needs to be Fixed Targeted to the Steelhead appliance in front of the Exchange server. This hidden in-path rule overrules any newly added in-path rules to not optimize traffic on TCP port 135.

This hidden in-path rule can be disabled with the CLI command `in-path probe-mapi-data`, which is enabled by default in RiOS versions 6.1.x and up to version 6.5.2.

5.19.1.2. Traffic on TCP port 443 is always SSL pre-optimized.

TCP port 443 is reserved for HTTPS traffic. The traffic for this port does always gets treated with a SSL pre-optimization policy, even if there is an in-path rule which doesn't have the SSL pre-optimization policy defined.

5.20. Network monitoring and network management systems

5.20.1. Network Management Systems

Network Management Systems and Network Monitoring Systems can be confused by the changes the Steelhead appliances make to the network. For example, the latency optimization might reply to requests much faster than the server would, giving the impression that the network and the server are fine.

Since there are two networks to be monitored now, an optimized and a non-optimized one, there should be two network management systems: One to monitor the services via the TCP sessions and one to monitor the services via the non-optimized TCP sessions. The unoptimized TCP sessions can be monitored by adding a pass-through in-path rule for traffic from that network management host on its local Steelhead appliance.

The two network management systems should give the same statuses for the remote machines or services, working, failure or unavailable. If there is a difference in the status, it means that there could be problem with a part of the network or with the services or with the optimization service.

5.20.2. Network Probing Systems

Network mappers and port scanners, like the nmap utility, work by sending out a large amount of TCP SYN packets to multiple ports on a single host or to a single port on multiple hosts, all during a short time. On an unoptimized network, this is cheap because the packets are small. On an optimized network, these TCP SYN packets will cause a lot of work to be done on the Steelhead appliances: Optimized TCP sessions will have to be setup and latency optimization will have to be initialized. As a result, the Steelhead appliance can generate alarms about connection-based admission control and high CPU.

OS Finger Printing will be incorrect because the TCP/IP stack the scanner is talking to is the one from the Steelhead appliance, not from the remote server.

The best practice is to exempt the traffic from port scanners and network mappers from being optimized via a pass-through in-path rule on the Steelhead appliance closest to it.

5.20.3. Spanning VLANs

The spanning feature on routers and switches allows all traffic coming into and going out via an interface or a VLAN to be copied and forwarded out via a physical interface or forwarded out via a spanning VLAN specifically defined for this kind of traffic.

In a network without Steelhead appliances, this forwarding over a VLAN work fine because there is often nothing in the path which worries about it except for some firewalls.

In a network with Steelhead appliances, if the traffic going over that spanning VLAN comes in touch with the Steelhead appliance it might confuse the Steelhead appliance in the following way:

- The simplified routing feature might learn the MAC address on the wrong interface and forward the packets to the wrong interface.
- The optimization service will count the packets double in its statistics in the list of Current Connections.
- If the TCP SYN packets are forwarded through the Steelhead appliance again the real second TCP SYN of the client will be seen as the third TCP SYN and the Steelhead appliance will put that TCP session into pass-through too early.

If the spanning VLAN has to go over the WAN, make sure it leaves the local network without it touching the Steelhead appliance.

Figure 5.158. The right location for a spanning VLAN on a switch

```
     .-,(  ),-.
  .-(          )-.
 (       WAN      )
  '-(          ).-'
      '-.( ).-'
          ^      __________    Steelhead      __________
          |     [          ]   _________     [          ]
          |     [          ]<--[_________]<---[          ]
          '-----[          ]                 [          ]
                [          ]<-----------------[          ]
                [_...__...o]    Spanning      [_...__....]
                   Router       VLAN vlan        Switch
```

5.21. Web proxies and web caches

Web proxies are servers in the network which proxy the traffic towards the web server by terminating the TCP session towards the client and establishing a TCP session towards the web server. The data retrieved on the session towards the web server is forwarded to the client.

Figure 5.159. The TCP sessions towards a web proxy

```
.--------.          .-----------.          .------------.
| client | <------> | web proxy | <------> | web server |
'--------'          '-----------'          '------------'
       TCP session from         TCP session from
      client to web proxy       web proxy to server
       on TCP port 8080          on TCP port 80
```

Web proxies can also have a caching functionality, where static objects requested by a remote web server are stored on the web proxy and served to the client if the web proxy knows that the object is still valid. This will save traffic on the link between the web proxy and the web server, but the latency is often still there.

For the Steelhead appliance, the issue is that the web proxy is serving both plain-text and encrypted traffic on TCP port 8080 which makes an optimal optimization on that TCP port impossible.

The details of which web proxy server to use are configured on the client's web browser and there are two approaches:

- Manual configuration in the clients web browsers settings, where different proxies can be configured for HTTP, HTTPS and FTP protocols.
- Automatic configuration of the clients web browsers settings by the use of a Proxy Auto-Configuration file, a JavaScript file which parses the URL about to be retrieved and returns the proxy to be used.

By selecting different TCP port numbers for different kinds of traffic, for example TCP port 8080 for HTTP traffic and TCP port 8443 for HTTPS traffic, it will be possible for the Steelhead appliances to only optimize the plain-text HTTP traffic on TCP port 8080 and pass-through the encrypted HTTPS traffic on TCP port 8443.

5.22. Security

With regarding to security on a Steelhead appliance, the following aspects can be considered:

- Access security, how to secure management access to the Steelhead appliance.
- Data security, how the data stored on the Steelhead appliance is secured.
- Network security, how data transmitted between Steelhead appliances is secured.

5.22.1. Access Security towards the Steelhead appliance

The management interfaces of the Steelhead appliance can be accessed via the CLI, either via SSH or via Telnet, and via the GUI, either via HTTP or via HTTPS. Further data can be retrieved from the device via the SNMP service.

5.22.1.1. Access Security via the Telnet service

Traffic transmitted via the Telnet protocol is transmitted in clear text. By default, the telnet service is not enabled on the Steelhead.

Figure 5.160. Telnet service is not enabled on this Steelhead appliance

```
SH (config) # no telnet-server enable
SH (config) # show telnet-server
Telnet server enabled:  no
```

5.22.1.2. Access Security via the SSH service

Traffic transmitted via the Secure Shell protocol is encrypted. By default the SSH service is enabled on the Steelhead appliance.

It can be used for interactive traffic or for batch transfers with the scp command, however the access to the device via scp is limited to specific directories.

By default, the SSH service accepts the SSH version 1 and SSH version 2 protocols, but it can be configured to use the SSH version 2 protocol only.

Not all SSH ciphers are enabled anymore in the later RiOS versions. PuTTY version 0.59 and before and SecureCRT are known to not to like this. See the command `show ssh server allow-ciphers` for the list of supported ciphers on the Steelhead appliance.

The SSH service by default listens for connections on all interfaces but it can be configured to listen only on specific interfaces.

As a bonus point the SSH service can be told to listen to a different TCP port. For Steelhead appliance configured with a public IP address and is directly accessible from the public Internet, this is a must to prevent dictionary attacks against the SSH service. However, Steelhead appliances managed by the CMC cannot use any other TCP port the default.

Figure 5.161. Secure SSH configuration

```
SH (config) # ssh server listen enable
SH (config) # ssh server listen interface primary
SH (config) # ssh server v2-only enable
SH (config) # ssh server port 8022
SH (config) # show ssh server allow-ciphers
SSH server allowed ciphers:
---------------------------
aes128-cbc
aes192-cbc
aes256-cbc
SH (config) # show ssh server
SH server enabled: yes
SSH server listen enabled: yes
SSH port: 8022
SSH v2 only: yes
   Listen Interfaces:
      Interface: primary
```

5.22.1.3. Access security via the HTTP and HTTPS services

Traffic transported via the HTTP protocol is transmitted in clear text, while traffic transported via the HTTPS protocol is encrypted. The HTTP and HTTPS services are both enabled by default.

The HTTP and HTTPS services by default listen for connections on all interfaces, but they can be configured to listen only on specific interfaces.

The HTTP and HTTPS services can be moved from the default TCP port 80 and TCP port 443 to different ports.

The SSL ciphers used for the HTTPS server can be configured with the standard Apache SSLCipherSuite statement options.

Figure 5.162. HTTP and HTTPS security configuration

```
SH (config) # no web http enable
SH (config) # web httpd listen enable
SH (config) # web httpd listen interface primary
SH (config) # web http port 8080
SH (config) # web https enable
SH (config) # web https port 8443
SH (config) # web ssl cipher ALL:!ADH:RC4+RSA:+HIGH:!MEDIUM:!LOW:!SSLv2:!EXPORT
SH (config) # show web
Web-based management console enabled: yes
   HTTP enabled: no
   HTTP port: 80
   HTTPS enabled: yes
   HTTPS port: 8443
   SOAP server enabled: no
   SOAP server port: 9001
   Configure Mode TRAP: yes
   Inactivity timeout: 120 minutes
   Session timeout: 60 minutes
   Session renewal threshold: 30 minutes
   Timeout during report auto-refresh: yes
   SSLv2 enabled: no
   SSLv3 enabled: yes
   TLSv1 enabled: yes
   Listen enabled: yes
   Listen Interfaces:
      Interface: primary
SH (config) # show web ssl cipher
   Apache SSL cipher string: ALL:!ADH:RC4+RSA:+HIGH:!MEDIUM:!LOW:!SSLv2:!EXPORT
```

5.22.1.4. Access security via the SNMP services

SNMP is an UDP based protocol. For the versions 1 and 2, the authentication is done via a shared secret known as the community string, which is transmitted in the clear. For version 3 the authentication is done via a challenge response mechanism.

The SNMP MIBs for the Riverbed appliances can be obtained from the Riverbed Support website under the documentation section or from the appliance itself under the Support tab in the GUI. There are several different MIB files, one for each appliance:

- RBT-MIB: The Riverbed specific MIB which links the individual appliance MIBs towards the enterprises MIB. This one is required for all other MIBs.
- STEELHEAD-MIB: The Steelhead appliance specific MIB.
- STEELHEAD-EX-MIB: The Steelhead EX appliance specific MIB.
- CMC-MIB: The CMC appliance specific MIB.
- CONTROLLER-MIB: The Steelhead Mobile Controller appliance specific MIB.
- INTERCEPTOR-MIB: The Interceptor appliance specific MIB.

The SNMP service on the Steelhead appliance can only be used to read data from the appliance, it cannot be used to write data to it or to enable features.

The SNMP service by default listens for connections on all interfaces, but it can be configured to listen only on specific interfaces.

Figure 5.163. SNMP security configuration

```
SH (config) # snmp-server listen enable
SH (config) # snmp-server listen interface primary
SH (config) # no snmp-server listen interface primary
SH (config) # snmp-server listen interface primary
SH (config) # show snmp
SNMP enabled: yes
System location:
System contact:
Engine ID: 0x8000430b80bfb2484edda91d
Read-only community:
Traps enabled: yes
Interface listen enabled: yes
Trap interface: primary
Persistent ifindex: no
Listen Interfaces:
   Interface: primary
No trap sinks configured.
```

5.22.1.4.1. SNMP v1

The SNMP version 1 implementation on the Steelhead appliance only supports one community string. If the community string is correct, then the Steelhead appliance will send the response. If the community string is incorrect, then the Steelhead appliance will not respond. There is no view on the data to limit the MIB tree that can be accessed via SNMP version 1.

Figure 5.164. SNMPv1 request for system.sysName.0

```
SH # tcpdump -ni primary -v -X -s 1600 port snmp
tcpdump: listening on primary, link-type EN10MB (Ethernet), capture size 1600 bytes
07:39:15.699362 IP (tos 0x0, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 71 \
    )
    10.0.1.1.39643 > 10.0.1.5.161:  { SNMPv1 { GetRequest(28) R=18892990  .1.3.6.1.2.1.1.5 \
    .0 } }
        0x0000:  4500 0047 0000 4000 3a11 103d 0a00 0101  E..G..@.:..=....
        0x0010:  0a00 0105 9adb 00a1 0033 88ab 3029 0201  .........3..0)..
        0x0020:  0004 0670 7562 6c69 63a0 1c02 0401 2048  ...public......H
        0x0030:  be02 0100 0201 0030 0e30 0c06 082b 0601  .......0.0...+..
        0x0040:  0201 0105 0005 00                        .......
07:39:15.699484 IP (tos 0x0, ttl 64, id 36, offset 0, flags [DF], proto UDP (17), length 7 \
    3)
    10.0.1.5.161 > 10.0.1.1.39643:  { SNMPv1 { GetResponse(30) R=18892990  .1.3.6.1.2.1.1. \
    5.0="SH" } }
        0x0000:  4500 0052 0024 4000 4011 0a0e 0a00 0005  E..R.$@.@.......
        0x0010:  0a00 0101 00a1 9adb 003e 30b9 3034 0201  .........>0.04..
        0x0020:  0004 0670 7562 6c69 63a2 2702 0401 2048  ...public.'....H
        0x0030:  be02 0100 0201 0030 1930 1706 082b 0601  .......0.0...+..
        0x0040:  0201 0105 0004 0253 48                   .......SH
```

As can be seen, the community string used here is *public*,

5.22.1.4.2. SNMP v3

The SNMP version 3 implementation on the Steelhead appliance supports multiple users and views. A user is a username and password combination, a view is a part of the MIB tree. Together they define what an SNMPv3 client can see.

In this example the username is "shtest" is allowed to see the view "system", which includes the MIB tree .1.3.6.1.2.1.1, which is the system MIB ".iso.org.dod.internet.mgmt.mib-2.system".

Figure 5.165. SNMPv3 configuration for the user "shtest" to the system MIB

```
snmp-server acl group shtest security-level noauth read-view system
snmp-server engine-ID "0x8000430b80bfb2484edda91d"
snmp-server group shtest security-model usm security-name shtest
snmp-server user shtest password encrypted 0x54e42e390606150a0dcb67fa5b6775ca auth-protoco \
    l MD5
snmp-server view system included ".1.3.6.1.2.1.1"
```

The SNMPv3 request will look like this on the wire:

Figure 5.166. SNMPv3 request for system.sysName.0

```
SH (config) # tcpdump -ni primary -v -X -s 1600 port snmp
tcpdump: listening on primary, link-type EN10MB (Ethernet), capture size 1600 bytes
12:25:01.539068 IP (tos 0x0, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 92 \
    )
    10.0.1.1.35409 > 10.0.1.5.161:  { SNMPv3 { F=r } { USM B=0 T=0 U= } { ScopedPDU E=  C= \
     { GetRequest(14) R=383165213  } } }
 0x0000:  4500 005c 0000 4000 3a11 1028 0a00 0101  E..\..@.:..(....
 0x0010:  0a00 0105 8a51 00a1 0048 23f9 303e 0201  .....Q...H#.0>..
 0x0020:  0330 1102 0424 9448 2702 0300 ffe3 0401  .0...$.H'.......
 0x0030:  0402 0103 0410 300e 0400 0201 0002 0100  ......0.........
 0x0040:  0400 0400 0400 3014 0400 0400 a00e 0204  ......0.........
 0x0050:  16d6 a31d 0201 0002 0100 3000            ..........0.
12:25:01.539478 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto UDP (17), length 13 \
    4)
    10.0.1.5.161 > 10.0.1.1.35409:  { SNMPv3 { F= } { USM B=32 T=631 U= } { ScopedPDU E= 0 \
    x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { Report(31) R=383165213  .1.3.6.1. \
    6.3.15.1.1.4.0=1 } } }
 0x0000:  4500 0086 0000 4000 4011 09fe 0a00 0105  E.....@.@.......
 0x0010:  0a00 0101 00a1 8a51 0072 30ed 3068 0201  .......Q.r0.0h..
 0x0020:  0330 1102 0424 9448 2702 0300 ffe3 0401  .0...$.H'.......
 0x0030:  0002 0103 041d 301b 040c 8000 430b 80bf  ......0.....C...
 0x0040:  b248 4edd a91d 0201 2002 0202 7704 0004  .HN.........w...
 0x0050:  0004 0030 3104 0c80 0043 0b80 bfb2 484e  ...01....C....HN
 0x0060:  dda9 1d04 00a8 1f02 0416 d6a3 1d02 0100  ................
 0x0070:  0201 0030 1130 0f06 0a2b 0601 0603 0f01  ...0.0...+......
 0x0080:  0104 0041 0101                           ...A..
12:25:01.710160 IP (tos 0x0, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 13 \
    7)
    10.0.1.1.35409 > 10.0.1.5.161:  { SNMPv3 { F=r } { USM B=32 T=631 U=shtest } { ScopedP \
    DU E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetRequest(28) R=383165214 \
      .1.3.6.1.2.1.1.5.0 } } }
 0x0000:  4500 0089 0000 4000 3a11 0ffb 0a00 0101  E.....@.:.......
 0x0010:  0a00 0105 8a51 00a1 0075 99c2 306b 0201  .....Q...u..0k..
 0x0020:  0330 1102 0424 9448 2802 0300 ffe3 0401  .0...$.H(.......
 0x0030:  0402 0103 0423 3021 040c 8000 430b 80bf  .....#0!....C...
 0x0040:  b248 4edd a91d 0201 2002 0202 7704 0673  .HN.........w..s
 0x0050:  6874 6573 7404 0004 0030 2e04 0c80 0043  htest....0.....C
 0x0060:  0b80 bfb2 484e dda9 1d04 00a0 1c02 0416  ....HN..........
 0x0070:  d6a3 1e02 0100 0201 0030 0e30 0c06 082b  .........0.0...+
 0x0080:  0601 0201 0105 0005 00                   .........
12:25:01.710314 IP (tos 0x0, ttl 64, id 1, offset 0, flags [DF], proto UDP (17), length 13 \
    9)
    10.0.1.5.161 > 10.0.1.1.35409:  { SNMPv3 { F= } { USM B=32 T=631 U=shtest } { ScopedPD \
    U E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetResponse(30) R=383165214 \
      .1.3.6.1.2.1.1.5.0="SH" } } }
 0x0000:  4500 0094 0001 4000 4011 09ef 0a00 0105  E.....@.@.......
 0x0010:  0a00 0101 00a1 8a51 0080 30fb 3076 0201  .......Q..0.0v..
 0x0020:  0330 1102 0424 9448 2802 0300 ffe3 0401  .0...$.H(.......
 0x0030:  0002 0103 0423 3021 040c 8000 430b 80bf  .....#0!....C...
 0x0040:  b248 4edd a91d 0201 2002 0202 7704 0673  .HN.........w..s
 0x0050:  6874 6573 7404 0004 0030 3904 0c80 0043  htest....09....C
 0x0060:  0b80 bfb2 484e dda9 1d04 00a2 2702 0416  ....HN......'...
 0x0070:  d6a3 1e02 0100 0201 0030 1930 1706 082b  .........0.0...+
 0x0080:  0601 0201 0105 0004 0b53 48              .........SH
```

The first two packets are a handshake between the SNMP client and the SNMP server, the next two packets are the actual request.

There are several failure situations here:

- Unknown user name
- No access to a MIB tree

5.22.1.4.2.1. Unknown user name

In case of an unknown username, the answer will be a Report instead of a GetResponse:

Figure 5.167. SNMPv3 request with an unknown username

```
11:24:52.269466 IP (tos 0x0, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 13 \
    7)
    10.0.1.1.43166 > 10.0.1.5.161:  { SNMPv3 { F=r } { USM B=26 T=275 U=shtest } { ScopedP \
    DU E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetRequest(28) R=794705738 \
      .1.3.6.1.2.1.1.5.0 } } }
11:24:52.269731 IP (tos 0x0, ttl 64, id 5, offset 0, flags [DF], proto UDP (17), length 14 \
    0)
    10.0.1.5.161 > 10.0.1.1.43166:  { SNMPv3 { F= } { USM B=26 T=276 U=shtest } { ScopedPD \
    U E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { Report(31) R=794705738  .1. \
    3.6.1.6.3.15.1.1.3.0=3 } } }
```

5.22.1.4.2.2. No access to a part of the MIB tree

The SNMPv3 configuration allows views to a part of the MIB tree. If the user is not configured to any part of the MIB tree, it will return an authorizationError:

Figure 5.168. SNMPv3 request for an unconfigured view

```
11:29:56.037463 IP (tos 0x0, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 13 \
    6)
    10.0.1.1.45142 > 10.0.1.5.161:  { SNMPv3 { F=r } { USM B=28 T=78 U=shtest } { ScopedPD \
    U E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetRequest(28) R=1322240049 \
      .1.3.6.1.2.1.1.5.0 } } }
11:29:56.037671 IP (tos 0x0, ttl 64, id 1, offset 0, flags [DF], proto UDP (17), length 13 \
    6)
    10.0.1.5.161 > 10.0.1.1.45142:  { SNMPv3 { F= } { USM B=28 T=78 U=shtest } { ScopedPDU \
     E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetResponse(28) R=1322240049 \
      authorizationError[errorIndex==0] .1.3.6.1.2.1.1.5.0= } } }
```

If the user is not allowed to see a certain part of the MIB tree, it will return a noSuchObject:

Figure 5.169. SNMPv3 request for a disallowed view

```
12:05:52.867922 IP (tos 0x0, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 13 \
    7)
    10.0.1.1.40284 > 10.0.1.5.161:  { SNMPv3 { F=r } { USM B=28 T=2234 U=shtest } { Scoped \
    PDU E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetRequest(28) R=11416833 \
    94  .1.3.6.1.2.1.2.1.0 } } }
12:05:52.868041 IP (tos 0x0, ttl 64, id 25, offset 0, flags [DF], proto UDP (17), length 1 \
    37)
    10.0.1.5.161 > 10.0.1.1.40284:  { SNMPv3 { F= } { USM B=28 T=2234 U=shtest } { ScopedP \
    DU E= 0x800x000x430x0B0x800xBF0xB20x480x4E0xDD0xA90x1D C= { GetResponse(28) R=11416833 \
    94  .1.3.6.1.2.1.2.1.0=[noSuchObject] } } }
```

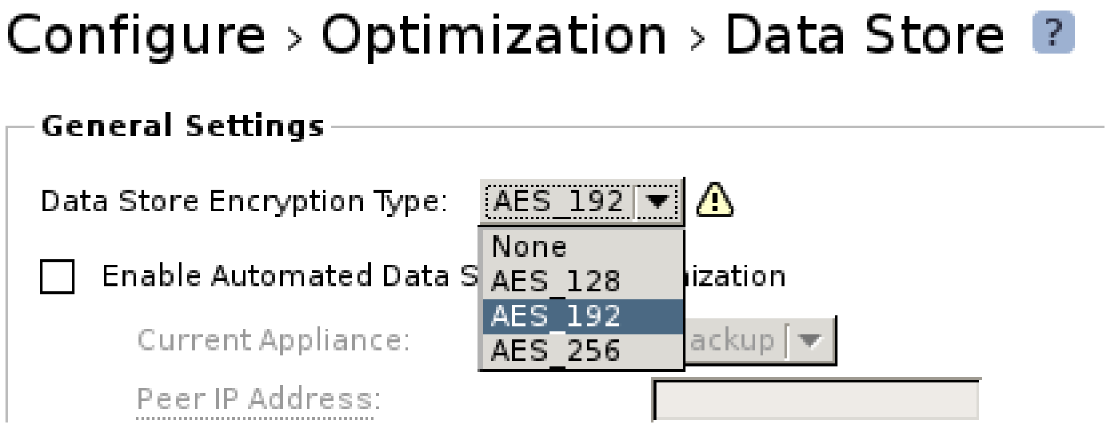
5.22.2. Data Security on the Steelhead appliance

With regarding to data security on the Steelhead appliance, there is data store encryption and there is the Secure Vault.

5.22.2.1. Data store encryption

The data store on the Steelhead appliance can be encrypted with the AES_128, AES_192 or AES_256 algorithms. The expected performance hit depends on workload and disk intensity, but about a 10% slower data store I/O should be expected.

Figure 5.170. Data store encryption level


Configure › Optimization › Data Store

General Settings

Data Store Encryption Type: AES_192

After a restart of the optimization service with the *Clear the data store* option enabled, the data store will be recreated and data will be written in an encrypted format:

Figure 5.171. Recreation of an encrypted data store

```
SH cli[10962]: [cli.INFO]: user admin: Executing command: restart clean
SH cli[10962]: [cli.INFO]: user admin: Command restart clean authorized
[...]
SH sport[11423]: [sport.INFO] - {- -} Stage 1 initialization starting...
SH sport[11423]: [segstore/DatastoreDiskOffset.INFO] - {- -} Initialized Special disk offs \
    et
[...]
SH sport[11423]: [segstore/peertable.INFO] - {- -} Peer table. Num peers : 4096 Groups : 1 \
    024 Pages : 9 Min Disconnect time : 86400
SH sport[11423]: [segstore.INFO] - {- -} Starting Persistent Store, config:  diskpolicy li \
    near cachepgs 1250000 hashbits 59 nblocks 4096 replacement_policy RVBDLRU w_behind 102 \
    4 r_ahead 1 w_queue_depth 128 encryption_type AES_128 clean_start 1
[...]
SH sport[11423]: [segstore/checker.NOTICE] - {- -} Creating a new data store. All existing \
     data will be erased.
[...]
SH sport[11423]: [sport.NOTICE] - {- -} Sport ready.
[...]
SH mgmtd[12237]: [mgmtd.INFO]: Exiting bypass mode
SH mgmtd[12237]: [mgmtd.INFO]: Traffic is now being optimized.
```

If some experimenting has been done with data store encryption and a data store has not been cleaned, the following message will be printed and the optimization service will not start:

Figure 5.172. After a move from AES_128 encryption to no encryption

```
SH sport[15692]: [segstore.INFO] - {- -} Starting Persistent Store, config:  diskpolicy li \
    near cachepgs 1250000 hashbits 59 nblocks 4096 replacement_policy RVBDLRU w_behind 102 \
    4 r_ahead 1 w_queue_depth 128 encryption_type NONE clean_start 0
[...]
SH sport[15692]: [segstore/drive/0 [/dev/md0].ERR] - {- -}   Encryption level mismatch.  C \
    urrently configured for: NONE but datastore is encrypted with: AES_128
SH sport[15692]: [mgmt.INFO] - {- -} Sending datastore alarm to true
SH sport[15692]: [segstore/drive/0 [/dev/md0].ERR] - {- -}  Segment store header invalid r \
    etry the dup header
SH sport[15692]: [segstore.ERR] - {- -} no good drives
SH sport[15692]: [segstore/drive/0 [/dev/md0].INFO] - {- -} destroying pending i/o stats
SH sport[15692]: [segstore.ALERT] - {- -} HALT Unable to initialize persistent store!
[...]
SH sport[15692]: [mgmt.WARN] - {- -} A corruption was found in the datastore. The datastor \
    e may not have been cleaned after the encryption type was changed.
SH sport[15692]: [mgmt.INFO] - {- -} Notifying management system that the datastore is cor \
    rupt.
SH sport[15692]: [mgmt.INFO] - {- -} Notifying management system that the service is stayi \
    ng down.
```

5.22.2.2. Secure Vault

The Secure Vault is an encrypted partition which contains the encrypted data store key and the SSL certificates and SSL private keys used for Secure Peering and SSL pre-optimization.

By default, the Secure Vault has been locked with a standard password which the Steelhead appliance can use to open the Secure Vault during its startup process. If this password is changed, the startup process cannot open the Secure Vault and the optimization service will not start until the Secure Vault is unlocked. The unlocking can be done either via the GUI or the CLI or automatically by the CMC.

Figure 5.173. Secure Vault is not unlocked yet

```
SH mgmtd[12258]: [mgmtd.INFO]: EVENT:  /rbt/discovery/event/sport_mgmt_disc_init_done
SH statsd[16376]: [statsd.NOTICE]: Alarm triggered for falling error for event secure_vaul \
    t_unlocked
SH mgmtd[12258]: [mgmtd.INFO]: EVENT:  /stats/event/alarm_activity
SH mgmtd[12258]: [mgmtd.INFO]: EVENT:  /rbt/health/event/activity
SH mgmtd[12258]: [mgmtd.INFO]: EVENT:  /stats/events/secure_vault_unlocked/falling/error
SH mgmtd[12258]: [mgmtd.INFO]: Secure vault must be unlocked
SH mgmtd[12258]: [mgmtd.INFO]: SSL acceleration and the secure datastore cannot be used un \
    til the secure vault has been unlocked.  To unlock the secure vault, please visit: htt \
    ps://SH/mgmt/gui?p=setupVault
SH mgmtd[12258]: [mgmtd.INFO]: EVENT:  /secure_vault/event/unlock_needed
```

5.22.3. Network security between Steelhead appliances

By default, the inner channel of the optimized TCP sessions is not encrypted. There are two ways to encrypt the inner channel: Via the IPsec Secure Peering feature, on which the IP packets with the inner channel gets transported via an IPsec tunnel over the WAN, or via the SSL Secure Peering feature, where the TCP session with the inner channel gets SSL encrypted.

5.22.3.1. IPsec Secure Peering

The configuration of the IPsec tunnels is symmetric, which means that the configuration on all the Steelhead appliances in the field which need secure peering between each other need to be in sync: Encryption policy, authentication policy and the shared secret. One incompatible mismatch between two IPsec peering configurations and all traffic is passed through unoptimized.

IPsec Secure Peering configuration is based on the IP addresses of the remote peer. If a renumbering of an in-path IP address happens or an additional in-path interface gets enabled, all peering Steelhead appliances need to be reconfigured to make sure that all optimized traffic towards this server stays encrypted.

5.22.3.1.1. Setup of an IPsec Secure Peering encrypted inner channel

Assume there is no IPsec tunnel setup towards the remote Steelhead appliance yet:

After the auto-discovery phase the inner channel gets setup: First the IPsec tunnel will be established towards the remote Steelhead appliance. Then the OOB Splice and Connection Pool will be setup over that IPsec tunnel. One of the TCP sessions of the inner channel will be converted into an inner channel and the inner channel is encrypted.

5.22.3.1.2. IPsec Secure Peering only enabled on one side

When the IPsec Secure Peering is configured on only one side of the inner channel, the client-side Steelhead appliance will try to setup an IPsec tunnel towards the server-side Steelhead appliance but fail. At the third TCP SYN packet from the client, the optimization service will pass-through the TCP session unoptimized.

Figure 5.174. IPsec negotiation fails between two Steelhead appliances because the other side isn't configured for IPsec Secure Peering

```
CSH racoon: 2012-06-16 10:10:45: INFO: IPSec-SA request for 192.168.1.6 queued due to no p \
    hase1 found.
CSH racoon: 2012-06-16 10:10:45: INFO: initiate new phase 1 negotiation: 10.0.1.6[500]<=>1 \
    92.168.1.6[500]
CSH racoon: 2012-06-16 10:10:45: INFO: begin Identity Protection mode.
CSH racoon: 2012-06-16 10:11:16: ERROR: phase2 negotiation failed due to time up waiting f \
    or phase1. ESP 192.168.1.6[500]->10.0.1.6[500]
CSH racoon: 2012-06-16 10:11:16: INFO: delete phase 2 handler.
CSH racoon: 2012-06-16 10:11:45: ERROR: phase1 negotiation failed due to time up. f2763262 \
    b03147b0:0000000000000000
```

Figure 5.175. Tcpdump shows that the IPsec service on the remote Steelhead appliance is not enabled

```
CSH # tcpdump -ni wan0_0 port 500 or icmp
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
10:29:37.907488 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident
10:29:38.040176 IP 192.168.1.6 > 10.0.1.6: ICMP 192.168.1.6 udp port 500 unreachable, leng \
    th 140
```

The *ICMP Port Unreachable* message shows that the IPsec service on the remote Steelhead appliance is not operational.

5.22.3.1.3. IPsec Secure Peering incompatible mismatch in parameters

The IPsec options can be depending on the model. For example the 50, 100, 200 and 300 models can only do encryption with the DES algorithm and cannot do encryption with the 3DES, AES or AES256 algorithms. When they setup an IPsec tunnel towards a Steelhead appliance which does not permit the low-level encryption algorithms, the setup of the IPsec tunnel will fail:

Figure 5.176. Incompatible options in the IPsec configuration

```
CSH sport[6467]: [splice/client.INFO] 914 {10.0.1.1:58923 192.168.1.1:25} init client 10.0 \
    .1.1:58923 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connect \
    ed: no
CSH racoon: 2012-06-16 10:34:45: INFO: IPsec-SA request for 192.168.1.6 queued due to no p \
    hase1 found.
CSH racoon: 2012-06-16 10:34:45: INFO: initiate new phase 1 negotiation: 10.0.1.6[500]<=>1 \
    92.168.1.6[500]
CSH racoon: 2012-06-16 10:34:45: INFO: begin Identity Protection mode.
CSH racoon: 2012-06-16 10:34:45: INFO: received Vendor ID: DPD
CSH racoon: 2012-06-16 10:34:46: INFO: ISAKMP-SA established 10.0.1.6[500]-192.168.1.6[500 \
    ] spi:59b96bb1d3f8513c:42bfd4b628b8d2ee
CSH racoon: 2012-06-16 10:34:46: INFO: initiate new phase 2 negotiation: 10.0.1.6[500]<=>1 \
    92.168.1.6[500]
CSH sport[6467]: [splice/client.ERR] 914 {10.0.1.1:58923 192.168.1.1:25} (clnt: 10.0.1.1:5 \
    8923 peer: 192.168.1.6:7800 serv: 192.168.1.1:25) Error connecting to peer: Resource t \
    emporarily unavailable. trpy: TRPY_NONE
CSH sport[6467]: [splice/client.INFO] 914 {10.0.1.1:58923 192.168.1.1:25} fini client 10.0 \
    .1.1:58923 server 192.168.1.1:25 cfe 10.0.1.6:0 sfe 192.168.1.6:7800 app TCP

SSH sport[7137]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:0 clnt: 10.0.1.1:58923 ser \
    v: 192.168.1.1:25) init
SSH racoon: 2012-06-16 10:18:56: INFO: respond new phase 1 negotiation: 192.168.1.6[500]<= \
    >10.0.1.6[500]
SSH racoon: 2012-06-16 10:18:56: INFO: begin Identity Protection mode.
SSH racoon: 2012-06-16 10:18:56: INFO: received Vendor ID: DPD
SSH racoon: 2012-06-16 10:18:57: INFO: ISAKMP-SA established 192.168.1.6[500]-10.0.1.6[500 \
    ] spi:59b96bb1d3f8513c:42bfd4b628b8d2ee
SSH sport[7137]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40302 clnt: 10.0.1.1:58923 \
     serv: 192.168.1.1:25) fini
SSH racoon: 2012-06-16 10:19:27: INFO: purging ISAKMP-SA spi=59b96bb1d3f8513c:42bfd4b628b8 \
    d2ee.
SSH racoon: 2012-06-16 10:19:27: INFO: purged ISAKMP-SA spi=59b96bb1d3f8513c:42bfd4b628b8d \
    2ee.
SSH racoon: 2012-06-16 10:19:27: ERROR: unknown Informational exchange received.
SSH racoon: 2012-06-16 10:19:28: INFO: ISAKMP-SA deleted 192.168.1.6[500]-10.0.1.6[500] sp \
    i:59b96bb1d3f8513c:42bfd4b628b8d2ee
SSH racoon: 2012-06-16 10:19:32: ERROR: unknown Informational exchange received.
SSH racoon: 2012-06-16 10:19:37: ERROR: unknown Informational exchange received.
SSH racoon: 2012-06-16 10:19:42: ERROR: unknown Informational exchange received.
SSH racoon: 2012-06-16 10:19:47: ERROR: unknown Informational exchange received.
SSH racoon: 2012-06-16 10:20:31: INFO: respond new phase 1 negotiation: 192.168.1.6[500]<= \
    >10.0.1.6[500]
SSH racoon: 2012-06-16 10:20:31: INFO: begin Identity Protection mode.
SSH racoon: 2012-06-16 10:20:31: INFO: received Vendor ID: DPD
SSH racoon: 2012-06-16 10:20:31: INFO: ISAKMP-SA established 192.168.1.6[500]-10.0.1.6[500 \
    ] spi:0ecac81771e26077:144dc771e6186531
```

```
SSH racoon: 2012-06-16 10:20:32: INFO: respond new phase 2 negotiation: 192.168.1.6[500]<= \
    >10.0.1.6[500]
SSH racoon: 2012-06-16 10:20:32: WARNING: trns_id mismatched: my:3DES peer:DES
SSH racoon: 2012-06-16 10:20:32: ERROR: not matched
SSH racoon: 2012-06-16 10:20:32: ERROR: no suitable policy found.
SSH racoon: 2012-06-16 10:20:32: ERROR: failed to pre-process packet.
```

Figure 5.177. Tcpdump shows that there are incompatible values in the configuration

```
CSH # tcpdump -ni wan0_0 port 500 or icmp
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
10:34:45.517507 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident
10:34:45.678227 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 1 R ident
10:34:45.706988 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident
10:34:45.849801 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 1 R ident
10:34:45.927267 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident[E]
10:34:46.209888 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 1 R ident[E]
10:34:46.209971 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 2/others R inf[E]
10:34:46.210702 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 2/others I inf[E]
10:34:51.342009 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 2/others R inf[E]
```

5.22.3.1.4. Incorrect shared secrets

When the IPsec shared secret doesn't match, the setup of the IPsec tunnel will fail.

Figure 5.178. Incorrect shared secret

```
CSH sport[6467]: [splice/client.INFO] 987 {10.0.1.1:65379 192.168.1.1:25} init client 10.0 \
    .1.1:65379 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connect \
    ed: no
CSH racoon: 2012-06-16 10:58:55: INFO: IPsec-SA request for 192.168.1.6 queued due to no p \
    hase1 found.
CSH racoon: 2012-06-16 10:58:55: INFO: initiate new phase 1 negotiation: 10.0.1.6[500]<=>1 \
    92.168.1.6[500]
CSH racoon: 2012-06-16 10:58:55: INFO: begin Identity Protection mode.
CSH racoon: 2012-06-16 10:58:56: INFO: received Vendor ID: DPD
CSH sport[6467]: [splice/client.ERR] 987 {10.0.1.1:65379 192.168.1.1:25} (clnt: 10.0.1.1:6 \
    5379 peer: 192.168.1.6:7800 serv: 192.168.1.1:25) Error connecting to peer: Resource t \
    emporarily unavailable. trpy: TRPY_NONE
CSH sport[6467]: [splice/client.INFO] 987 {10.0.1.1:65379 192.168.1.1:25} fini client 10.0 \
    .1.1:65379 server 192.168.1.1:25 cfe 10.0.1.6:0 sfe 192.168.1.6:7800 app TCP

SSH racoon: 2012-06-16 10:48:17: INFO: respond new phase 1 negotiation: 192.168.1.6[500]<= \
    >10.0.1.6[500]
SSH racoon: 2012-06-16 10:48:17: INFO: begin Identity Protection mode.
SSH racoon: 2012-06-16 10:48:17: INFO: received Vendor ID: DPD
```

Figure 5.179. Tcpdump shows that the IPsec negotiation fails because of an incorrect shared secret

```
CSH # tcpdump -ni wan0_0 port 500 or icmp
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
10:58:55.948616 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident
10:58:56.082075 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 1 R ident
10:58:56.093290 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident
10:58:56.237696 IP 192.168.1.6.500 > 10.0.1.6.500: isakmp: phase 1 R ident
10:58:56.249238 IP 10.0.1.6.500 > 192.168.1.6.500: isakmp: phase 1 I ident[E]
```

5.22.3.2. SSL Secure Peering

SSL Secure Peering is the encryption of the inner channel of an optimized TCP session via the Secure Sockets Layer protocol. Unlike IPsec Secure Peering where the IP packets of the inner channel got tunneled, with SSL Secure Peering it is only encrypting the TCP payload of the inner channel.

The advantages of this method are:

- The encryption starts after a TCP session of the Connection Pool has been converted into an inner channel. This makes it possible to use the Secure Peering feature for only certain protocols. This also makes it possible to fall-back to no encryption if the remote peer has not been configured properly.
- With IPsec Secure Peering all IP traffic from the Steelhead appliance towards the remote Steelhead appliance went through the IPsec tunnel, including ICMP traffic. With SSL Secure Peering only the optimized TCP session gets encrypted.
- With IPsec Secure Peering all traffic is seen as UDP traffic on UDP port 500, with SSL Secure Peering the inner channel is still visible on the WAN, making it possible to use a different WAN visibility to the TCP session and to apply QoS to the TCP session.
- With SSL Secure Peering, the dependency on knowing the IP addresses of the in-path interfaces of the remote Steelhead appliance is gone, the identification now happens on the SSL certificate of the remote Steelhead appliance.

The SSL Secure Peering can be enabled for SSL protocols only (via an in-path rule which has SSL pre-optimization enabled), for secure protocols (SSL, encrypted MAPI, encrypted Lotus Notes, encrypted Citrix) or for all protocols.

Before two Steelhead appliances setup a SSL session between them, they need to trust each other. This can be done via three methods:

- SSL Peering certificates from unknown Steelhead appliances are stored in the SSL Peering Gray list. If the unknown Steelhead appliance is considered to be trusted, you can move it to the SSL Peering White list and SSL Secure Peering can be performed towards that Steelhead appliance. If the unknown Steelhead appliance is considered not to be trusted, you can move it to the SSL Peering Black list and SSL Secure Peering will never be performed towards that Steelhead appliance. This is extensive manual labour of the order O(N!) but available to the Steelhead appliance without the need for external software. Using a CMC will make his much easier.
- The SSL Peering feature can be configured with a trusted CA certificate. As such, this Steelhead appliance will trust SSL peering certificates signed by this CA. This is an one time configuration per Steelhead appliance making it an issue of the order O(N) but an external certificate authority is required.

  There is also the Mobile Trust feature, which is the same approach but specifically for the SSL Peering CA certificates for Mobile Steelhead Mobile Controllers.
- Support for the Simple Certificate Enrollment Protocol, which allows the importing of the SSL Certificate generated by an automated Certificate Authority.

5.22.3.2.1. Setup of an SSL Secure Peering encrypted inner channel

The auto-discovery phase and the setup of the OOB Splice and Connection Pool are the same as a normal deployment. During the conversion from the TCP session of the Connection Pool to an inner channel, the request for an secure inner channel gets added. Once the SSL session has been setup, the inner channel data gets transported over it.

5.22.3.2.2. Unknown SSL peering certificates

In case the SSL peering certificate of a remote Steelhead appliance does not exist in the SSL Peering White list, the setup of the SSL session will fail with these messages:

Figure 5.180. Failure due to an unknown SSL peering certificate

```
CSH sport[13790]: [splice/client.INFO] 26 {10.0.1.1:15092 192.168.1.1:25} init client 10.0 \
    .1.1:15092 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connect \
    ed: yes
CSH sport[13790]: [splice/client.INFO] 26 {10.0.1.1:15092 192.168.1.1:25} Splice client si \
    de initializing: No protocol port = 25 protocol id = TCP(0) transport = TRANSPORT_ID_S \
    SLINNER
CSH sport[13790]: [splice/client.INFO] 26 {10.0.1.1:15092 192.168.1.1:25} Start flowing, l \
    port 40336, rport 7800, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT \
    _ID_SSLINNER, TPTOPT_NONE(0x0)
CSH sport[13790]: [ssl.WARN] - {- -} Failure in certificate verification:  error num: 18 e \
    rror string: self signed certificate depth: 0 subject name: /CN=Steelhead A17WT000628B \
    C/O=Riverbed Technology, Inc./L=San Francisco/ST=California/C=--
CSH sport[13790]: [keystore/0x9bdaa80.WARN] - {- -} Failure in certificate verification, s \
    ubject: "/CN=Steelhead A17WT000628BC/O=Riverbed Technology, Inc./L=San Francisco/ST=Ca \
    lifornia/C=--"
CSH sport[13790]: [keystore/0x9bdaa80.WARN] - {- -} Failed with error num: 18, error strin \
    g: "self signed certificate", depth: 0
CSH sport[13790]: [sslinnerchan/CliConnect.WARN] 26 {10.0.1.1:15092 192.168.1.1:25} The se \
    rver-side steelhead at ip address: 192.168.1.6 couldn't verify this steelhead's peerin \
    g certificate. Perhaps the peering trust list on server-side steelhead is misconfigure \
    d
CSH sport[13790]: [sslinnerchan/CliConnect.WARN] 26 {10.0.1.1:15092 192.168.1.1:25} SSL co \
    nnection to an untrusted Steelhead appliance at IP address: 192.168.1.6 while attempti \
    ng to optimize the end-to-end SSL connection between client: 10.0.1.1:15092 and server \
    : 192.
CSH sport[13790]: [sslinnerchan/CliConnect.WARN] 26 {10.0.1.1:15092 192.168.1.1:25} 168.1. \
    1:25
CSH sport[13790]: [mgmt.INFO] - {- -} mgmtd notified that ssl peer 192.168.1.6 failed hand \
    shake
CSH mgmtd[3768]: [mgmtd.INFO]: EVENT:  /rbt/sport/ssl/event/peer_info
CSH mgmtd[3768]: [mgmtd.NOTICE]: Peer 192.168.1.6 is created/updated in the gray list.

SSH sport[10600]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:0 clnt: 10.0.1.1:15092 se \
    rv: 192.168.1.1:25) init
SSH sport[10600]: [splice/server.INFO] 1 {- -} init cfe 10.0.1.6:40336 sfe 192.168.1.6:780 \
    0
SSH sport[10600]: [splice/server.INFO] 1 {- -}  sock 41 id 548627 client 10.0.1.1:15092 se \
    rver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[10600]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40268 clnt: 10.0.1.1:1509 \
    2 serv: 192.168.1.1:25) acquired
SSH sport[10600]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40268 clnt: 10.0.1.1:1509 \
    2 serv: 192.168.1.1:25) fini
SSH sport[10600]: [splice/server.INFO] 1 {- -} Splice server side initializing: No protoco \
    l port = 25 transport = TRANSPORT_ID_SSLINNER
SSH sport[10600]: [splice/server.INFO] 1 {10.0.1.1:15092 192.168.1.1:25} Start flowing, lp \
    ort 7800, rport 40336, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_SSLINNER, TPTOPT_NONE(0x0)
SSH sport[10600]: [ssl.WARN] - {- -} Failure in certificate verification:  error num: 18 e \
    rror string: self signed certificate depth: 0 subject name: /CN=Steelhead A41SU00085F1 \
    3/O=Riverbed Technology, Inc./L=San Francisco/ST=California/C=--
SSH sport[10600]: [keystore/0x9bda9a0.WARN] - {- -} Failure in certificate verification, s \
    ubject: "/CN=Steelhead A41SU00085F13/O=Riverbed Technology, Inc./L=San Francisco/ST=Ca \
    lifornia/C=--"
SSH sport[10600]: [keystore/0x9bda9a0.WARN] - {- -} Failed with error num: 18, error strin \
    g: "self signed certificate", depth: 0
SSH sport[10600]: [sslinnerchan/SrvAccept.WARN] 1 {10.0.1.1:15092 192.168.1.1:25} SSL conn \
    ection from an untrusted Steelhead appliance at IP address: 10.0.1.6 while attempting  \
    to optimize the end-to-end SSL connection between client: 10.0.1.1:15092 and server: 1 \
    92.168
SSH sport[10600]: [sslinnerchan/SrvAccept.WARN] 1 {10.0.1.1:15092 192.168.1.1:25} .1.1:25
SSH sport[10600]: [mgmt.INFO] - {- -} mgmtd notified that ssl peer 10.0.1.6 failed handsha \
    ke
SSH sport[10600]: [sslinnerchan/SrvAccept.WARN] 1 {10.0.1.1:15092 192.168.1.1:25} Informin \
    g the client-side steelhead about failure to verify its peering certificate, so that f \
    uture connections are temporarily bypassed.
SSH mgmtd[4019]: [mgmtd.INFO]: EVENT:  /rbt/sport/ssl/event/peer_info
SSH mgmtd[4019]: [mgmtd.NOTICE]: Peer 10.0.1.6 is created/updated in the gray list.
```

The list of entries in the Secure Peering Gray list can be found with the command `show secure-peering gray-lst-peers`. This output does only contain the IP address of the in-path interface and the hostname while the log files only contain the IP address of the in-path interface and the Steelhead appliance serial number.

Figure 5.181. Output of the command "show secure-peering gray-lst-peers"

```
CSH # show secure-peering gray-lst-peers
 IP               Hostname                                             Exp Date
 ---------------  -----------------------------------  ------------------------
 192.168.1.6      SSH                                  Nov 29 03:44:24 2012 GMT

SSH # show secure-peering gray-lst-peers
 IP               Hostname                                             Exp Date
 ---------------  -----------------------------------  ------------------------
 10.0.1.6         CSH                                  Jun 20 01:42:12 2013 GMT
```

Use the command `secure-peering gray-lst-peer <IP address> trust` to move an entry from the Secure Peering Gray list to the Secure Peering White list. After the move, new TCP connections should be optimized with a secure inner channel, this can be seen with the transport option set to TRANSPORT_ID_SSLINNER:

Figure 5.182. Optimized TCP session with an encrypted inner channel

```
CSH sport[25745]: [splice/client.INFO] 1 {10.0.1.1:44130 192.168.1.1:25} init client 10.0. \
    1.1:44130 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connecte \
    d: yes
CSH sport[25745]: [splice/client.INFO] 1 {10.0.1.1:44130 192.168.1.1:25} Splice client sid \
    e initializing: No protocol port = 25 protocol id = TCP(0) transport = TRANSPORT_ID_SS \
    LINNER
CSH sport[25745]: [splice/client.INFO] 1 {10.0.1.1:44130 192.168.1.1:25} Start flowing, lp \
    ort 40269, rport 7800, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_SSLINNER, TPTOPT_NONE(0x0)
```

5.22.3.2.3. Remote appliances without the SSL license

When a remote Steelhead appliance does not have a valid SSL license, the secure inner channel cannot be setup and the following message will be displayed in the logs:

Figure 5.183. Remove Steelhead appliance does not have a valid SSL license

```
SH sport[994]: [splice/client.WARN] 482649 {10.0.1.1:2800 192.168.1.1:135} SSL license is  \
    either absent or invalid on peer Steelhead appliance with IP address: 192.168.1.6
SH sport[994]: [splice/client.NOTICE] 482649 {10.0.1.1:2800 192.168.1.1:135} Together, loc \
    al and peer steelhead appliance at: 192.168.1.6:7800 request encryption of secure appl \
    ication traffic over WAN. And this connection between client: 10.0.1.1 and
SH sport[994]: [splice/client.NOTICE] 482649 {10.0.1.1:2800 192.168.1.1:135}  server: 192. \
    168.1.1.  201:135 belongs to a secure application protocol: EPM(4) But one (or both) o \
    f them is missing a valid enhanced cryptographic license.
```

5.23. The Riverbed Services Platform

The Riverbed Services Platform (RSP) is a VMware based solution to be able to run virtual machines on the xx20 and xx50 series models.

The Virtual Services Platform (VSP) is for the EX series models and discussed in a different chapter.

5.23.1. Design of the RSP feature

5.23.1.1. Prerequisites

To be able to run the RSP service on the Steelhead appliance, the following conditions need to be met:

- An RSP license needs to be installed. This will allow the RSP software to be installed and started.
- Extra RSP memory needs to be installed. This RSP memory is not used by the optimization service.

  For the 6050 model, this is a 4 Gb memory module but it will only use 2 Gb of it.

For the 1050 model, this is a 2 Gb memory module. However if more RSP memory is required it is possible to add another 2 Gb to it.

For the 150, 250 and 550 models, this is a 2 Gb memory module. However because of the design of the machine only 1.5 Gb of the memory is available for RSP.

For the other models, this is a 2 Gb memory module.

- The RSP image needs to be installed.

5.23.1.2. How it fits in the system

First some definitions:

- An RSP image is the software image of the RSP service as provided by Riverbed. Once installed it has become the RSP service.
- The RSP service is the virtualization software, based on VMware server.
- An RSP package is the virtual machine software image as generated by the RSP Package Generator. Once installed in the RSP service it can become a RSP slot.
- An RSP slot is a virtual machine, running on top of the RSP service.

The RSP service runs on top of RiOS. The RSP slots run on top of the RSP service.

Figure 5.184. RSP on RiOS

```
.------------------------------------------.
| RSP slot         | RSP slot      | ...    |
| (Windows server) | (Linux based) |        |
|------------------------------------------|
|        RSP service (VMware server)        |
|------------------------------------------|
|              RiOS (Linux)                 |
'------------------------------------------'
```

The networks to the RSP slots can be provided in several ways:

- The RSP slot can be configured to be on the network of the primary or auxiliary interface and will need to be configured to have an IP address of that IP subnet. This can be used for stand-alone services like an Active Directory Domain Controller or a local DNS/DHCP server.
- The RSP slot can be configured to be on the in-path interface on the LAN side, where it will receive the unoptimized traffic. This can be used for a transparent proxy or for an IPS service.
- The RSP slot can be configured to be on the in-path interface on the WAN side, where it will receive the optimized traffic. This can be used for a traffic shaper or the SCPS service.

5.23.2. Possible issues

5.23.2.1. Limitations

Because the RSP slot runs in the Steelhead appliance, there are several hardware components which are not available for the RSP slots:

- No floppy disk drives
- No CD-ROM drivers
- No audio

- No access to the USB ports

The RSP Package Generator checks for these issues and will generate a warning when they are defined in the virtual machine.

5.23.2.2. Upgrades

When the RiOS version is upgraded, make sure to check the release notes to see if the RSP image needs to be upgraded too. When the RSP image needs to be upgraded too, perform the following steps:

- Upload the new RiOS image.
- Upload the new RSP image.
- Install the new RiOS image.
- Disable the RSP service.
- Reboot the Steelhead appliance.
- Install the new RSP image.
- Enable the RSP service.

If there are multiple RiOS images to be installed, do not install the latest RSP image until the RiOS version required has been reached.

5.23.2.3. Excessive paging

Despite having 2 Gb of RSP memory available, this is not fully available to the RSP slots. The RSP service itself uses about 384 Mb, leaving 1664 Mb available for the RSP slots.

So if the Steelhead appliance has raised the paging alarm and RSP is enabled, make sure that the memory for the total memory in use on the RSP slots is less than 1664 Mb.

5.23.2.4. VLAN tagging on primary interface

VLAN trunking is not supported on the primary and auxiliary interfaces. So the virtual machine always needs to have an IP address from the IP subnets defined on the primary or auxiliary interfaces.

5.24. Access related problems

During the operation of the device it is possible that the GUI or CLI access is not working. It is important to notice various different styles of not working:

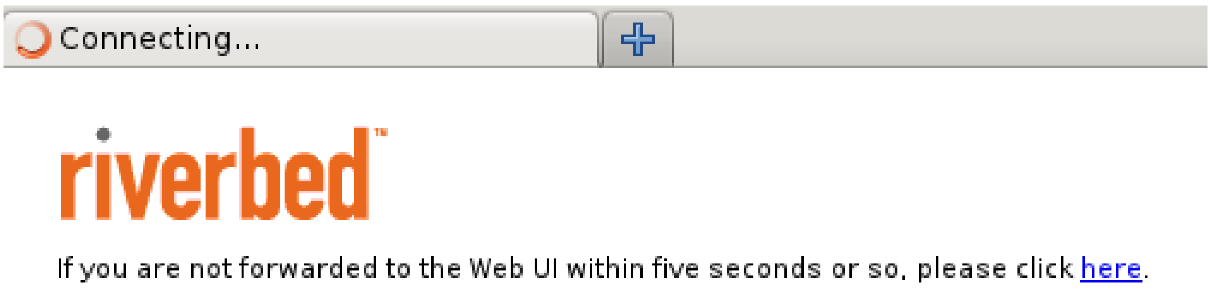
5.24.1. Unable to access the GUI

For the Steelhead appliance, the GUI consists of two pieces: A management application and a web server which acts as front-end of the management application.

When connecting to the root of the GUI, for example via the URL http://10.0.1.5/, the web server will answer the request for the root of the GUI with a redirection to /mgmt/, the path of the management application, in this example at http://10.0.1.5/mgmt/. At the root of the GUI there will be the login screen or the overview screen of the management application.

If the redirection page does not get loaded, then there is a problem with connecting to the web server. Check with ping to see if the primary interface of the Steelhead appliance is still reachable and with telnet to port 80 to see if the web server is running.

Figure 5.185. Steelhead GUI redirection screen



If you are not forwarded to the Web UI within five seconds or so, please click here.

With regards to the case where the redirection page does get loaded, but the management application does not load: This behaviour has been observed in the following situations:

- When the */var* partition is full. This can be checked with the command `support show disk`:

  Figure 5.186. Output of the command "support show disk"

  ```
  Filesystem          1024-blocks      Used Available Capacity Mounted on
  [...]
  /dev/sda3              16513448  16513448         0      100% /var
  [...]
  ```

  If this is the case, please call Riverbed TAC to help clean up the */var* partition.

- When the eUSB flash memory is locked for a long time and the machine isn't running the RiOS versions which deals with it. See KB article S15568 and S15587 for the full details.

- When the management application is failing: The next steps would be to connect to the CLI and issue the command `pm process webasd restart`.

- When there is something wrong with the optimized sessions towards that Steelhead appliance. Try using HTTPS instead of HTTP to overcome any optimization related issues.

5.24.2. Unable to connect to the CLI

There are several steps in the setup of the SSH session:

- (1) The command to start the SSH client.
- (2) The setup of the TCP session over which the SSH session gets setup.
- (3) The SSH server banner.
- (4) The host key exchange.
- (5) The DNS lookup of the client IP address by the server.
- (6) The login banner.
- (7) The password authentication.
- (8) The first output of the login shell.

Figure 5.187. An SSH session with verbose logging

```
1 edwin@t43>ssh -v admin@10.0.1.5
```

```
  OpenSSH_5.8p2_hpn13v11 FreeBSD-20110503, OpenSSL 0.9.8q 2 Dec 2010
  debug1: Reading configuration data /usr/home/edwin/.ssh/config
  debug1: Reading configuration data /etc/ssh/ssh_config
2 debug1: Connecting to 10.0.1.5 [10.0.1.5] port 22.
  debug1: Connection established.
  debug1: identity file /usr/home/edwin/.ssh/id_dsa type -1
  debug1: identity file /usr/home/edwin/.ssh/id_dsa-cert type -1
3 debug1: Remote protocol version 1.99, remote software version OpenSSH_5.2
  debug1: match: OpenSSH_5.2 pat OpenSSH*
  debug1: Remote is not HPN-aware
  debug1: Enabling compatibility mode for protocol 2.0
  debug1: Local version string SSH-2.0-OpenSSH_5.8p2_hpn13v11 FreeBSD-20110503
  debug1: SSH2_MSG_KEXINIT sent
  debug1: SSH2_MSG_KEXINIT received
  debug1: kex: server->client aes128-ctr hmac-md5 none
  debug1: kex: client->server aes128-ctr hmac-md5 none
  debug1: SSH2_MSG_KEX_DH_GEX_REQUEST(1024<1024<8192) sent
  debug1: expecting SSH2_MSG_KEX_DH_GEX_GROUP
  debug1: SSH2_MSG_KEX_DH_GEX_INIT sent
  debug1: expecting SSH2_MSG_KEX_DH_GEX_REPLY
4 debug1: Server host key: RSA ae:f3:6b:8c:6f:a0:e7:4d:01:1f:9c:cf:91:ad:14:e7
  The authenticity of host '10.0.1.5 (10.0.1.5)' can't be established.
  RSA key fingerprint is ae:f3:6b:8c:6f:a0:e7:4d:01:1f:9c:cf:91:ad:14:e7.
  Are you sure you want to continue connecting (yes/no)? yes
  Warning: Permanently added '10.0.1.5' (RSA) to the list of known hosts.
  debug1: ssh_rsa_verify: signature correct
  debug1: SSH2_MSG_NEWKEYS sent
  debug1: expecting SSH2_MSG_NEWKEYS
5 debug1: SSH2_MSG_NEWKEYS received
  debug1: Roaming not allowed by server
  debug1: SSH2_MSG_SERVICE_REQUEST sent
  debug1: SSH2_MSG_SERVICE_ACCEPT received
6 Riverbed Steelhead
  debug1: Authentications that can continue: publickey,password
  debug1: Next authentication method: publickey
  debug1: Trying private key: /usr/home/edwin/.ssh/id_rsa
  debug1: Trying private key: /usr/home/edwin/.ssh/id_dsa
  debug1: Trying private key: /usr/home/edwin/.ssh/id_ecdsa
  debug1: Next authentication method: password
7 admin@10.0.1.5's password:
  debug1: Authentication succeeded (password).
  Authenticated to 10.0.1.5 ([10.0.1.5]:22).
  debug1: HPN to Non-HPN Connection
  debug1: Final hpn_buffer_size = 2097152
  debug1: HPN Disabled: 0, HPN Buffer Size: 2097152
  debug1: channel 0: new [client-session]
  debug1: Enabled Dynamic Window Scaling

  debug1: Requesting no-more-sessions@openssh.com
  debug1: Entering interactive session.
8 Last login: Sun Jul 29 01:55:15 2012 from 10.0.1.1
```

5.24.2.1. SSH TCP session does not get setup

If the TCP session times out, then it could be either a routing issue, firewall issue or the Steelhead appliance is turned off.

If the SSH client comes back with *ssh: connect to host 10.0.1.5 port 22: Connection refused* then there is no SSH server running on that IP address.

The easiest way to check if the SSH server is running is to use telnet to setup a TCP session to the primary interface on port 22:

Figure 5.188. Telnet session to the SSH service

```
[~] edwin@t43>telnet 10.0.1.5 22
Trying 10.0.1.5...
Connected to 10.0.1.5.
Escape character is '^]'.
SSH-1.99-OpenSSH_5.2
```

If the TCP session is setup and there is a SSH banner, then the SSH service is working. If the SSH banner is not displayed, then the kernel on the Steelhead appliance is still working but the user-land is having problems.

5.24.2.2. SSH host key has changed

When a Steelhead appliance gets replaced in the network, the IP address or hostname will stay the same but the SSH host key will be different. OpenSSH will give a warning like this:

Figure 5.189. Setup of an SSH session to the CLI of a replaced Steelhead appliance

```
$ ssh admin@10.0.1.5
Riverbed Steelhead
Password:
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
e:f3:6b:8c:6f:a0:e7:4d:01:1f:9c:cf:91:ad:14:e7.
Please contact your system administrator.
Add correct host key in /usr/home/edwin/.ssh/known_hosts to get rid of this message.
Offending RSA key in /usr/home/edwin/.ssh/known_hosts:50
RSA host key for 10.0.1.5 has changed and you have requested strict checking.
Host key verification failed.
```

If this happens, confirm that the device was replaced.

To overcome this issue with the OpenSSH SSH client, remove the line in the file mentioned. With the PuTTY SSH client a dialog box will be shown with the option to replace the key.

5.24.2.3. SSH Reverse DNS timeout

If the SSH client has a delay of about 60 seconds before displaying the password prompt, then the issue is most likely related to reverse DNS lookup failure: Confirm that the DNS servers configured on the Steelhead appliance are correct and reachable.

5.24.2.4. SSH login shell not started at all.

If, after the authentication, the login shell is not started then there is most likely a disk failure on the Steelhead appliance.

5.24.2.5. Prompt only disappears after a very long delay

It can happen that the SSH client asks for the username and password, but doesn't get to the CLI. This has experienced when an appliance has a high system load, for example because of the generation of a lot of process dumps. The way out of that is to reboot the appliance in single user mode and remove all process dumps from */var/opt/tms/snapshots/.staging/*.

Next steps: Please contact the Riverbed TAC for assistance. Access to the serial console is required, as well as permission to reboot the Steelhead appliance.

5.24.2.6. Reduced CLI shell

During the login to the CLI, the login shell *cli* opens a communication channel to the process *mgmtd*. If this channel cannot be made, the prompt will show `CLI>` and only accept a certain subset of commands. Once the communication channel to the mgmtd process is available the shell will connect to it and all commands be available again.

If the process *mgmtd* does not come back, the next step would be to boot the appliance in single user mode and investigate why the process *mgmtd* didn't start.

5.25. Partitions running out of space

There are several partitions on the Steelhead appliance which are writable and can run out of space:

- */config*: This partition is part of the flash memory and is used for storing the configuration files.
- */proxy*: This partition is used for the RSP and PFS.
- */var*: This partition is used for logging, system dumps and process dumps.

When a partition is full, an alarm will be raised and the status of the device will become *Degraded*:

Figure 5.190. File system alarm

```
[statsd.NOTICE]: Alarm triggered for rising error for event fs_mnt
```

The command `support show disk` shows the status of the partitions:

Figure 5.191. Output of the command "support show disk"

```
SH # support show disk
Filesystem         1024-blocks      Used Available Capacity Mounted on
/dev/sdb7               418806     49904    347277      13% /config
/dev/sda3             16513448   2769100  12905408      18% /var
/dev/disk0p6          56765020  39510668  14370736      74% /proxy
```

The capacity shows the amount of usage which is used: 0% is empty, 100% is full.

5.25.1. The /config partition is full

The */config* partition is not manageable from the GUI or CLI and Riverbed TAC should be contacted to determine the cause and resolve the issue.

5.25.2. The /proxy partition is full

The */proxy* partition is used by both PFS and RSP.

If PFS is used and the partition is full, then the amount of data stored on the file server backed by the PFS share has exceeded the threshold.

If RSP is used there are several scenarios:

- Check if there are any RSP images, the RSP software or RSP backups which can be removed.

  Figure 5.192. Removing an obsolete RSP image

  ```
  SH # show rsp images
  image_rbt_rsp_6_0_0_n19.img
  image_rbt_rsp_6_5_0_n77.img
  SH # rsp image delete image_rbt_rsp_6_0_0_n19.img
  SH #
  ```

- Check if there are any RSP packages, the virtual machines, which can be removed.

  Figure 5.193. Removing an obsolete RSP package

  ```
  SH # show rsp packages
  FCI_WIN2K3_STD_X32_v3.0.pkg
  FCI_WIN2K3_STD_X32_v3.1.pkg
  SH # rsp package delete FCI_WIN2K3_STD_X32_v3.0.pkg
  SH #
  ```

- Check if there are any RSP backups which can be removed.

Figure 5.194. Removing an obsolete RSP backup

```
SH # show rsp backups
backup-slot3-20120801.tgz
backup-slot3-20120802.tgz
SH # rsp backups delete backup-slot3-20120801.tgz
SH #
```

5.25.3. The /var partition is full

The */var* partition is used for the logging of services running on the Steelhead appliance, to capture historical statistics of various parts of the Steelhead appliance, to store process dumps, system dumps and tcpdump captures and to keep track of process files.

Check the following things:

- Are there any tcpdump captures running? The background tcpdump capture files are not showing up in the GUI under Reports -> Diagnostics -> TCP Dumps but they still use the disk space:

Figure 5.195. Check for running background tcpdump captures

```
SH # show tcpdump-x
Name: ssh-cifs
Start Time: 10:48:17

SH # tcpdump-x capture-name ssh-cifs stop
SH #
```

- Are there any tcpdump capture files which can be removed?

Figure 5.196. Remove obsolete tcpdump capture files

```
SH # show files tcpdump
SH_primary_csh-cifs.cap
SH_lan0_0_csh-cifs.cap
SH_wan0_0_csh-cifs.cap
SH # file tcpdump delete SH_primary_csh-cifs.cap
SH #
```

- Are there any old process dumps which can be removed?

Figure 5.197. Remove obsolete process dumps

```
SH # show files process-dump
sysdump-SH-20120801-200301.tgz
sysdump-SH-20120801-200523.tgz
SH # file process-dump delete sysdump-SH-20120801-200301.tgz
SH #
```

- Are there any old system dumps which can be removed?

Figure 5.198. Remove obsolete system dumps

```
SH # show files debug-dump
SH-sport-20120801-200257.tar.gz
SH-sport-20120801-200519.tar.gz
SH # file debug-dump delete SH-sport-20120801-200257.tar.gz
SH #
```

- Neural framing statistics logs can be gathered but are never rotated. In the *sysinfo.txt* they can be found under the `Output of 'find /var/opt ( -name .running -prune ) -o -type f -ls -mount'`: section as */var/opt/rbt/neural-stats*. Contact the Riverbed TAC to remove them.

Figure 5.199. Neural Framing log files

```
Output of 'find /var/opt ( -name .running -prune ) -o -type f -ls -mount':
[...]
634172  564 ---S-wS--T  1 admin  root  72586 Nov  2  2007 /var/opt/rbt/neural-stats.22416. \
    0
634215   80 ---S-wS--T  1 admin  root  77441 Nov  2  2007 /var/opt/rbt/neural-stats.22416. \
    1
634216  116 ---S-wS--T  1 admin  root  11986 Nov  2  2007 /var/opt/rbt/neural-stats.22416. \
    2
634217 2312 ---S-wS--T  1 admin  root  60701 Nov  2  2007 /var/opt/rbt/neural-stats.22416. \
    3
[...]
640388   88 -rw-rws---  1 admin  root  84021 Jul 31  2008 /var/opt/rbt/neural-stats.6631.1 \
    159
640389  144 -rw-rws---  1 admin  root  40656 Jul 31  2008 /var/opt/rbt/neural-stats.6631.1 \
    160
640390   96 -rw-rws---  1 admin  root  92481 Jul 31  2008 /var/opt/rbt/neural-stats.6631.1 \
    161
```

If the issue isn't resolved after this, please contact Riverbed TAC for further investigation.

5.26. Memory usage on the Steelhead appliance

There are two kinds of memory on the Steelhead appliance: Real and virtual. Real memory comes from the DIMM memory sticks inside the appliance and is fast, virtual memory comes from an allocated slice on the hard disks and is slow. The process of moving memory from the real memory to the virtual and vice versa is called paging.

If paging is happening, it means that the processes needing its memory will have to wait before it is served and that other processes will start to page too. Thus: Active paging is a bad thing and a source of slowness.

Under normal circumstances, there should be no virtual memory in active use. The following exceptions will be there:

- On the 3RU xx50 series models like the 5050 and 6050 models, a part of the virtual memory allocated for the copy of the USB flash memory. However, it will not be actively used: No paging in and paging out should be happening except during times when there are write operations to the USB flash disk.
- On the 50, 100, 200 and 300 models the memory is very tight, under normal operations there is only 2-3 Mb of memory free. When logging in to the Steelhead appliance it will start to swap to rearrange the memory and this will cause paging.

The memory on the Steelhead appliance is used in several ways:

- Used by the optimization service.
- Used by the operating system.
- Used by the non-optimization processes.
- Used by the RSP service.

5.26.1. The optimization service

The optimization service takes up most of the memory on a Steelhead appliance. For example on a 1050L model with 2 Gb of memory, the optimization service aka the process *sport* will use 1.3 Gb of it.

When monitoring the memory usage on the Steelhead appliance via SNMP, you will see that the the memory usage on the Steelhead appliance is very high, often above 80%: This is normal.

During the startup of the optimization service, it will allocate a pool of memory which it will use to allocate internal memory structures used by the optimized TCP sessions. For example these structures are for the general TCP optimization, the CIFS read-ahead buffers and the objects for the HTTP optimization. The memory allocated by the optimization service is marked as "non-paging", which means that it will never be paged out to disk.

If coded properly, every object allocated will be released later. If not, a piece of memory will be lost. If this happens too often, this pool of memory will be exhausted and the optimization service will be entering memory based admission control. When this happens, please open a case with Riverbed TAC to investigate.

The only way out is to restart the optimization service, but don't do this until Riverbed TAC has the data required.

5.26.2. The operating system

The memory usage by the operating system is not directly identifiable but there are two usages which can be observed: The network buffers and the disk buffers.

If the operating system runs out of network buffers, the optimization service will enter TCP based admission control. This scenario is described in the *Admission Control* section.

If the operating system runs out of disk buffers, it will become slow at reading from and writing to disk. This can happen when a large amount of brand new, compressed or encrypted data comes through: The data store has to be searched for matching references, but they cannot be found. Furthermore the data store will be written to the whole time. This constant reading and writing will exhaust the disk buffers and cause slowness on the system.

5.26.3. The non-optimization processes

The non-optimization processes use the rest of the memory. Unless there is a memory related issue and paging is happening, they should all run inside the real memory. If however for some reason the Steelhead appliance is running out of memory and paging start happening, these are the processes which will be paged to disk.

5.26.4. The RSP service

As described in a previous section *The Riverbed Services Platform*, the RSP memory is used by the RSP software and the RSP slots.

5.27. LAN side traffic is seen on the Steelhead appliance

If the list of Current Connections shows TCP sessions, either optimized or pass-through, and they are coming from and going to the same IP subnet which is defined on the LAN side, then there is something wrong with either the switches or with the subnet mask definitions on the hosts.

5.27.1. IP subnet and Subnet masks mismatch

If the TCP session shows up for only one host on the network, checking the IP subnet and subnet mask on that host would be the first step to make sure that it matches the configured IP subnet and subnet mask as on the router.

Figure 5.200. Running ipconfig on a Windows machine to check the subnet mask

```
C:\>ipconfig

Ethernet adapter Local Area Connection:
    Connection-specific DNS Suffix . :
    IP Address . . . . . . . . . . . : 10.0.1.1
    Subnet . . . . . . . . . . . . . : 255.255.255.0
    Default Gateway. . . . . . . . . : 10.0.1.9

C:\>
```

Figure 5.201. Running ifconfig on a Unix machine to check the subnet mask

```
[~] edwin@t43>ifconfig bge0
bge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=8009b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM,LINKSTATE>
 inet 10.0.1.1 mask 255.255.255.0
        ether d4:9a:20:c2:52:0e
        inet6 fe80::216:41ff:fe53:6b26%bge0 prefixlen 64 scopeid 0x1
        nd6 options=23<PERFORMNUD,ACCEPT_RTADV,AUTO_LINKLOCAL>
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
```

5.27.2. Switch related issues

Normally a switch knows which MAC addresses are behind a physical port and it will forward the Ethernet frames to there: You will not see these packets on the Steelhead appliance.

If however the switch does not know where a MAC address is, it will forward it to out via all physical ports. The reason it doesn't know where a MAC address is can be for a couple of reasons:

- The host which has sent out that Ethernet frame has the MAC address for that IP address hard coded in its MAC table.
- The MAC table on the switch is full and it cannot learn any new MAC address to port mappings.

The way to troubleshoot this would be to take a short capture of a TCP session on the Steelhead with the link layer details:

Figure 5.202. Tcpdump capture for a LAN side TCP conversation

```
SH # tcpdump -ni lan0_0 -le -c 10 host 10.0.1.1 and host 10.0.1.5
08:12:56.776625 d4:9a:20:c2:52:0e > 00:0e:b6:42:f8:98, ethertype IPv4 (0x0800), length 82: \
     10.0.1.1.59623 > 10.0.1.5.80: Flags [.], ack 13, win 65535, options [nop,nop,TS val 1 \
    821865 ecr 1813898], length 0
08:12:56.777495 d4:9a:20:c2:52:0e > 00:0e:b6:42:f8:98, ethertype IPv4 (0x0800), length 82: \
     10.0.1.1.59623 > 10.0.1.5.80: Flags [.], ack 1445, win 65535, options [nop,nop,TS val \
     1821865 ecr 1813898], length 0
08:12:56.778494 d4:9a:20:c2:52:0e > 00:0e:b6:42:f8:98, ethertype IPv4 (0x0800), length 82: \
     10.0.1.1.59623 > 10.0.1.5.80: Flags [.], ack 2877, win 65535, options [nop,nop,TS val \
     1821865 ecr 1813898], length 0
08:12:56.778561 d4:9a:20:c2:52:0e > 00:0e:b6:42:f8:98, ethertype IPv4 (0x0800), length 82: \
     10.0.1.1.59623 > 10.0.1.5.80: Flags [.], ack 4309, win 65535, options [nop,nop,TS val \
     1821866 ecr 1813898], length 0
```

As can be seen, the MAC addresses are the ones expected. Also note that this traffic is only coming from one host, the one with IP address 10.0.1.5.

The next step would be check the switch if it knows where to send these MAC addresses to. On a Cisco switch this command is `show mac address-table`.

Figure 5.203. Check the MAC address table

```
SWITCH # show mac address-table
          Mac Address Table
-------------------------------------------
Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
[...]
d49a.20c2.520e
   1    d49a.20c2.520e    DYNAMIC     Fa0/3
[...]
SWITCH # show mac address-table | i d49a.20c2.520e
   1    d49a.20c2.520e    DYNAMIC     Fa0/3
SWITCH # show mac address-table | i 000e.b642.f898
SWITCH #
```

So it knows where to find the MAC address of d4:9a:20:c2:52:0e, but not where the MAC address of 00:0e:b6:42:f8:98 is located. And therefore the packet to 00:0e:b6:42:f8:98 gets forwarded to all ports.

The next step would be to find out why the switch doesn't know about the destination MAC address.

5.28. Service Error

Sometimes the optimization service determines that there is something wrong, either in the data store or in the communication towards a peer. It will then raise the service_error alarm, which is a non-fatal error for the optimization service but a fatal error for the TCP session it has been encountered on.

5.28.1. Data store related service errors

5.28.1.1. Steelhead Mobile Client Cloning

When a new computer gets rolled out and the installation of the software on it happens via cloning it from a master image, the data store id of the Steelhead Mobile Client software will be cloned too. As a result, the server-side Steelhead appliance will see duplicate labels and thus raise the service_error alarm:

Figure 5.204. Duplicate DEF and a Service error with a Steelhead Mobile Client

```
SH sport[1448]: [segpage.ERR] - {- -} Duplicate DEF {}136 hash 11237544087762729201 vs. 44 \
    0090/248488450:20#0{}176 hash 4723104384001475725, memcmp() 1
SH sport[1448]: [defunpacker.ALERT] - {- -} ALARM (clnt: 10.0.1.1:49358 serv: 192.168.1.1: \
    1352 cfe: 10.0.1.1:49349 sfe: 192.168.1.6:7800) name maps to more than one segment has \
     a steelhead been improperly installed and/or steelhead mobiles are sharing config fil \
    es p
SH sport[1448]: [defunpacker.ALERT] - {- -} ossibly through copying or cloning?
SH sport[1448]: [defunpacker.WARN] - {- -} (clnt: 10.0.1.1:49358 serv: 192.168.1.1:1352 cf \
    e: 10.0.1.1:49349 sfe: 192.168.1.6:7800) Killing page 0x2b8835a000 name: 440090/248488 \
    450:20 off 59732398 refs 2 cnt 50 flags CA--- tag 1954835526/12, segment #
SH sport[1448]: [defunpacker.WARN] - {- -} 544087762729201
SH sport[1448]: [segstore/kill_page.ERR] - {- -} Killing page name: 440090/248488450:20 of \
    f 59732398 refs 2 cnt 50 flags CA--- tag 1954835526/12 and dumping it to refd_pages.14 \
    48
SH statsd[28893]: [statsd.NOTICE]: Alarm triggered for rising error for event service_erro \
    r
```

The IP addresses of the *clnt* (client) and the *cfe* (client-side Steelhead) are the same, therefore it is a Steelhead Mobile Client causing this problem.

5.28.1.2. Data store clustering

Figure 5.205. Duplicate DEF and a Service error but not with a Steelhead Mobile Client

```
SH sport[26651]: [defunpacker.ALERT] - {- -} ALARM (clnt: 10.0.1.1:51748 serv: 192.168.1.1 \
    :80 cfe: 10.0.1.6:7801 sfe: 192.168.1.6:7800) name maps to more than one segment has a \
     steelhead been improperly installed and/or  steelhead mobiles are sharing conf
SH sport[26651]: [defunpacker.ALERT] - {- -} ig files possibly through copying or cloning? \
```

Here the IP addresses of the *clnt* (client) and the *cfe* (client-side Steelhead) are not the same, therefore the remote IP address belong to a real Steelhead appliance and not a Steelhead Mobile Client.

The most likely cause is that there has been an issue with hosts in a data store synchronization cluster and the data store wasn't cleared after the cluster got taken apart. Clearing the data store and a service restart with the command `restart clean` should resolve this issue.

5.28.1.3. Requested frame does not exist anymore

When the sending Steelhead appliance sends a reference and the receiving Steelhead appliance does not have that reference in its data store, the receiving Steelhead appliance will request that reference from the sending Steelhead appliance. If the sending Steelhead appliance does not have that reference it its data store anymore, it will throw this error:

Figure 5.206. Requested reference does not exist anymore

```
SH sport[32414]: [replypacker.ALERT] - {- -} ALARM ACK problem: REQd segment 380198/193427 \
    159:2600447491#16 absent
```

This can happen when the optimized TCP connection is running in the SDR-M optimization policy or the optimized TCP connection is a CIFS connection and the reference was only available in the CIFS cache part of the data store.

If this happens once, then just clear the alarm. If this happens more and more, then open a TAC case.

5.28.2. Communication related service errors

If there is a problem in the protocol spoken on the inner channel between the Steelhead appliances, a service error will be raised.

Since the protocol is spoken over TCP, the network stack will take care of TCP checksum issues and the data should be valid. When using the WAN visibility modes of port transparency or full transparency, there might be firewalls or IPS devices which check the protocol and "fix" irregularities in it, which can cause corruption on the inner channel.

Figure 5.207. A service error due to a checksum mismatch

```
SH sport[3612]: [sportpkt.ALERT] - {- -} ALARM decode_hdr: checksum mismatch, cmd 1, len 3 \
    2789
SH statsd[9205]: [statsd.NOTICE]: Alarm triggered for rising error for event service_error
```

To troubleshoot checksum mismatches would be to take traces in various places in the network between the two Steelhead appliances until the issue happens again and see if the corruption is already there. Then use the divide-and-conquer approach to find the device which is corrupting the content.

5.29. Can this application be optimized?

This question comes often to the Riverbed TAC and the answer is mostly "Most likely".

5.29.1. On individual TCP sessions

The Steelhead appliance optimizes in three ways: TCP optimization, Data Reduction and Latency Optimization.

The first one, TCP optimization, changes the characteristics of the TCP session, for example the TCP Window which allows more data outstanding over the WAN. If the client doesn't support these features in its TCP stack, the TCP optimization of the Steelhead appliance will improve the throughput.

The third one, latency optimization, only works if the application uses certain protocols like CIFS, NFS and HTTP. If the application is using these protocols to access remote data, then latency optimization on the Steelhead appliance will improve the throughput.

The second one, data reduction, works most likely unless the data stream is compressed or encrypted. If the data is compressed, then the configuration of the application should be checked to see if the compression can be disabled. If the data is encrypted via SSL, then the traffic can be optimized if the server SSL certificate and the server SSL private key are available. If these two items are not available, then the traffic cannot be optimized. If the data is not encrypted via SSL, then the traffic cannot be optimized.

If that application is already being optimized, the ratio of the traffic on the LAN and WAN side can be checked:

Figure 5.208. An encrypted or compressed TCP session

```
CSH # show connections opt filter 10.16.5.39 full
T  Source                Destination           App    Rdn Since
--------------------------------------------------------------------------------
O  10.0.1.1         2067 192.168.1.1      1039 TCP     0% 2013/03/20 11:45:47
                   Peer: 192.168.1.6      7800 Inner: 37024
            Outer Local: 10.0.1.6         7801   WAN: 2299KB
           Outer Remote: 10.0.1.1         2067   LAN: 2003KB
    WAN Visibility Mode: Correct Addressing
                  Flags: cfe,sdr,lz,te=0,pe=0
```

In this case the protocol is TCP (so no skewing because of a read-ahead feature in latency optimization) and the amount of bytes on the WAN side is larger than the amount of bytes on the LAN side, making the Reduction 0%. We found out that the application used compression on the wire. After the application configuration was changed the amount of bytes on the LAN side increased but due to the data reduction of the Steelhead appliance the amount of bytes on the WAN side decreased.

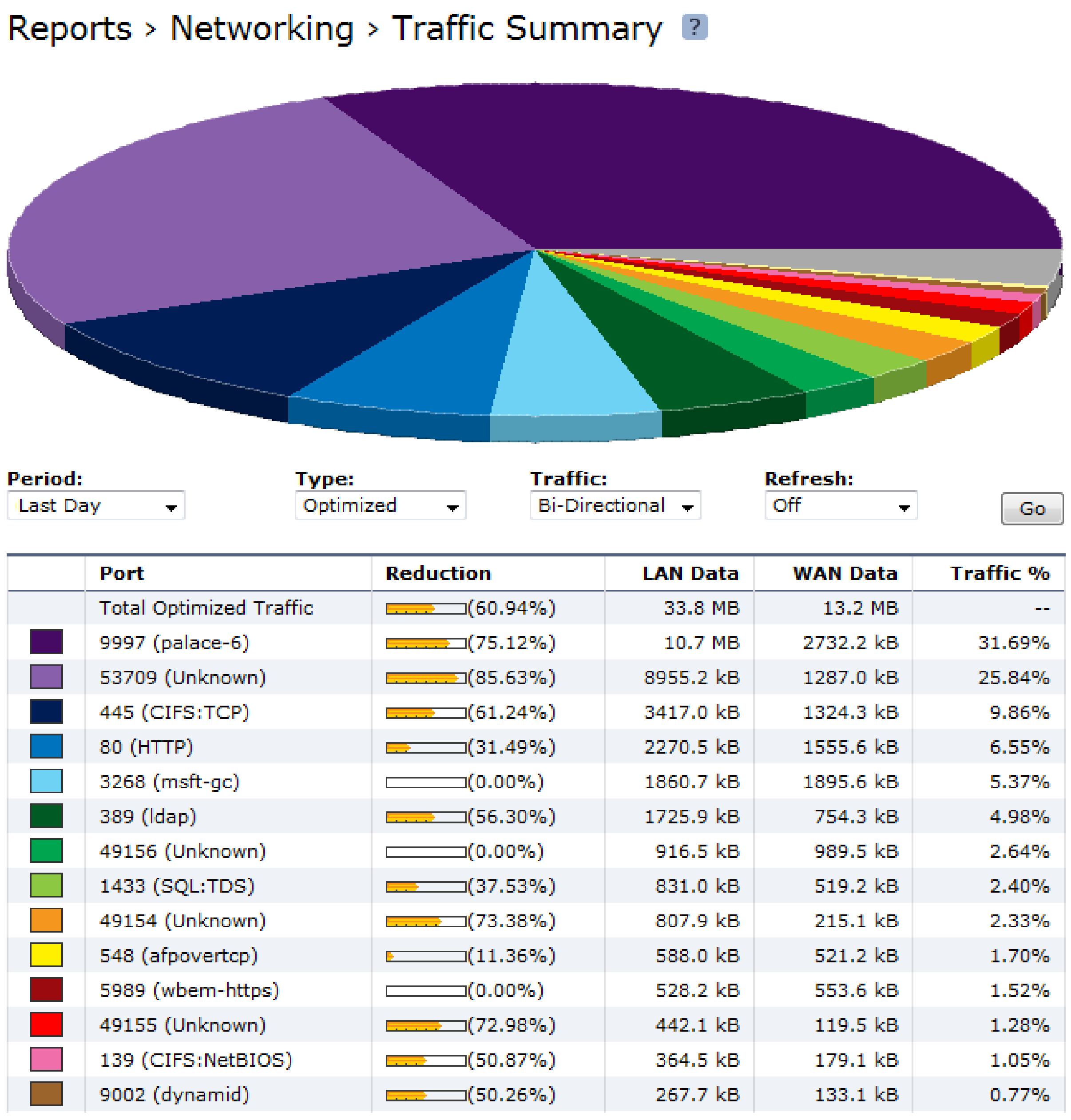
Figure 5.209. An formerly compressed TCP session

```
CSH # show connections opt filter 10.16.5.39 full
T  Source                Destination           App    Rdn Since
--------------------------------------------------------------------------------
O  10.0.1.1         2103 192.168.1.1      1039 TCP     9% 2013/03/20 11:57:12
                   Peer: 192.168.1.6      7800 Inner: 37041
            Outer Local: 10.0.1.6         7801   WAN: 599KB
           Outer Remote: 10.0.1.1         2103   LAN: 6412KB
    WAN Visibility Mode: Correct Addressing
                  Flags: cfe,sdr,lz,te=0,pe=0
```

5.29.2. Overview per protocol

In the Traffic Summary under Reports -> Networking -> Traffic Summary, it will be possible to determine the overal performance of a certain protocol:

Figure 5.210. The traffic summary report



Reports > Networking > Traffic Summary

Period: Last Day
Type: Optimized
Traffic: Bi-Directional
Refresh: Off

	Port	Reduction	LAN Data	WAN Data	Traffic %
	Total Optimized Traffic	(60.94%)	33.8 MB	13.2 MB	--
🟪	9997 (palace-6)	(75.12%)	10.7 MB	2732.2 kB	31.69%
🟪	53709 (Unknown)	(85.63%)	8955.2 kB	1287.0 kB	25.84%
🟦	445 (CIFS:TCP)	(61.24%)	3417.0 kB	1324.3 kB	9.86%
🟦	80 (HTTP)	(31.49%)	2270.5 kB	1555.6 kB	6.55%
🟦	3268 (msft-gc)	(0.00%)	1860.7 kB	1895.6 kB	5.37%
🟩	389 (ldap)	(56.30%)	1725.9 kB	754.3 kB	4.98%
🟩	49156 (Unknown)	(0.00%)	916.5 kB	989.5 kB	2.64%
🟩	1433 (SQL:TDS)	(37.53%)	831.0 kB	519.2 kB	2.40%
🟧	49154 (Unknown)	(73.38%)	807.9 kB	215.1 kB	2.33%
🟨	548 (afpovertcp)	(11.36%)	588.0 kB	521.2 kB	1.70%
🟥	5989 (wbem-https)	(0.00%)	528.2 kB	553.6 kB	1.52%
🟥	49155 (Unknown)	(72.98%)	442.1 kB	119.5 kB	1.28%
🟪	139 (CIFS:NetBIOS)	(50.87%)	364.5 kB	179.1 kB	1.05%
🟫	9002 (dynamid)	(50.26%)	267.7 kB	133.1 kB	0.77%

Here can be seen that traffic on TCP port 3268, 49156 and 5989 is not being reduced in size at all. The total percentage of traffic for these three ports is 9.53%, which is quite big. If the application settings cannot be modified to allow a better data reduction, disabling optimization for these ports via an in-path rule would prevent the data store to be filled with useless references and free up TCP connections.

5.30. Interceptor cluster related issues

The Interceptor appliance is a Riverbed specific appliance to redirects traffic to a cluster of Steelhead appliances. It does not take part in the optimization of the traffic itself, it only redirects.

An Interceptor cluster consists of two types on nodes: one or more Interceptors and one or more Steelhead appliances.

5.30.1. Network setup

While the Interceptor appliance is located inline in the traffic flow, the Steelhead appliances in the cluster are only connected with the WAN interface.

Figure 5.211. Network setup for the Interceptor cluster

```
 .----------.
 |  Router  |
 '----------'
      |
  .--------.
  |   IC   |
  '--------'
      |       .--------.
      |   .---|  SH 1  |
      |   |   '--------'
      |   |   .--------.
      |   |---|  SH 2  |
      |   |   '--------'
      |   |
   .----------.
   |  Switch  |
   '----------'
```

There are several traffic flows:

- Unoptimized traffic, which flows through the Interceptor appliance. This includes inner channels of optimized TCP sessions with the Correct Addressing or Port Transparency WAN Visibility methods which are not terminating on this Interceptor cluster.
- New traffic that could be optimized:
  - A SYN+ packet comes in via the WAN interface and gets GRE encapsulated forwarded to one of the Steelhead appliances in the cluster.
  - A naked SYN comes in via the LAN interface and gets GRE encapsulated forwarded to one of the Steelhead appliances in the cluster.
  - A naked SYN/ACK comes in via the LAN interface and gets GRE encapsulated forwarded to the correct Steelhead appliance.
  - A SYN/ACK+ comes in via the WAN interface and gets GRE encapsulated forwarded to the correct Steelhead appliance.
- Traffic from the server to the client, which gets received on the LAN side. The destination IP address in the IP header will be swapped for the IP address of the Steelhead appliance in-path interface and send to the correct Steelhead appliance.
- The inner channel of an optimized TCP session from the client-side Steelhead appliance to the server-side Steelhead appliance with the Full Transparency WAN visibility mode. This gets modified from Full Transparency mode to Correct Addressing mode and then send to the correct Steelhead appliance.

5.30.2. The cluster

The communication between the individual nodes is happening via the Connection Forwarding protocol via their in-path interfaces. At startup of the redirection service, the Interceptor appliance will contact all other nodes, Interceptor and Steelhead appliances, and exchange capability information which includes the IP addresses of the in-path interfaces and, for the Steelhead appliances, the capacity and health status.

Figure 5.212. Setup of connection forwarding session between Interceptor and Steelhead appliance

```
IC interecptor[25354]: [neigh/client/channel.INFO] - {- -} Establishing neighbor channel f \
    rom 192.168.1.12 to 192.168.1.6:7850
IC interecptor[25354]: [neigh/client/channel.INFO] - {- -} Neighbor channel to 192.168.1.6 \
    :7850 established.
```

If the neighbouring Steelhead appliance isn't reachable, it will time out:

Figure 5.213. Failure in the setup of connection forwarding session between Interceptor and Steelhead appliance

```
IC interceptor[4937]: [neigh/client/channel.INFO] - {- -} Establishing neighbor channel fr \
    om 192.168.1.12 192.168.1.6:7850
IC interceptor[4937]: [neigh/client/channel.WARN] - {- -} Connection failure: couldn't con \
    nect to neighbor 192.168.1.6:7850. Connection timed out
```

If the neighbouring Steelhead appliance optimization service isn't running, it will show a connection refused.

Figure 5.214. Failure in the setup of connection forwarding session between Interceptor and Steelhead appliance

```
IC interceptor[4937]: [neigh/client/channel.INFO] - {- -} Establishing neighbor channel fr \
    om 192.168.1.12 192.168.1.6:7850
IC interceptor[4937]: [neigh/client/channel.WARN] - {- -} Connection failure: couldn't con \
    nect to neighbor 192.168.1.6:7850. Connection refused
```

If the neighbour Steelhead appliance becomes unreachable, the session will timeout after three seconds.

Figure 5.215. Failure in the connection forwarding session between Interceptor and Steelhead appliance

```
IC interecptor[25354]: [neigh/client/channel.INFO] - {- -} Establishing neighbor channel f \
    rom 192.168.1.12 to 192.168.1.6:7850
IC interecptor[25354]: [neigh/client/channel.INFO] - {- -} Neighbor channel to 192.168.1.6 \
    :7850 established.
IC interecptor[25354]: [neigh/client/channel.WARN] - {- -} No response from neighbor 192.1 \
    68.1.6:7850. Count = 1
IC interecptor[25354]: [neigh/client/channel.WARN] - {- -} No response from neighbor 192.1 \
    68.1.6:7850. Count = 2
IC interecptor[25354]: [neigh/client/channel.WARN] - {- -} No response from neighbor 192.1 \
    68.1.6:7850. Count = 3
IC interecptor[25354]: [neigh/client/channel.WARN] - {- -} No response from neighbor 192.1 \
    68.1.6:7850. Neighbor is unreachable.
```

If the optimization service on the neighbour Steelhead appliance stops, the session will be terminated too:

Figure 5.216. Failure in the connection forwarding session between Interceptor and Steelhead appliance

```
IC interecptor[25354]: [neigh/client/channel.INFO] - {- -} Establishing neighbor channel f \
    rom 192.168.1.12 to 192.168.1.6:7850
IC interecptor[25354]: [neigh/client/channel.INFO] - {- -} Neighbor channel to 192.168.1.6 \
    :7850 established.
IC interceptor[25354]: [neigh/server/channel.NOTICE] - {- -} End of stream reading neighbo \
    r from 192.168.1.6:7850. Peer maybe down.
```

When a new optimizable TCP session gets seen, the Interceptor appliance will select one of the Steelhead appliances to which it will be redirected which uses the Connection Forwarding mechanism to inform all the other Interceptor and Steelhead appliances about it. If a new optimized TCP session gets setup to a Steelhead appliance directly, for example via a Fixed Target rule, it also will inform all the nodes in the Interceptor cluster about it.

5.30.2.1. Peering affinity

When a new optimizable TCP session gets seen by an Interceptor appliance, it needs to be forwarded to one of the Steelhead appliances in the cluster. The best data reduction can be obtained between two Steelhead appliances which have exchanged the most references. So the Interceptor cluster always tries to send the TCP session from a certain remote Steelhead appliance to the same Steelhead appliance in the cluster, to obtain the best data reduction.

5.30.2.2. Fair peering

With peering affinity, it would be possible for traffic going through an Interceptor cluster with multiple Steelhead appliances to end up on the same Steelhead appliance. With the Fair Peering feature the Interceptor cluster puts effort in the attempts to distribute the traffic from remote Steelhead appliances equally over the number of Steelhead appliances in the cluster.

5.30.2.3. Pressure Monitoring and Capacity Adjustments

With the Pressure Monitoring feature enabled, the Interceptor appliances keep track of the memory usage and the disk pressure on the Steelhead appliances. With the Capacity Adjustment feature, the Interceptor appliances will dynamically reduce the initially exchanged capacity of the Steelhead appliances if there is pressure on them. This will decrease the number of TCP sessions being redirected to that Steelhead appliance, giving it a chance to reduce the pressure. Once every hour the Interceptor appliance will decide if the pressure on the Steelhead appliances has been reduced far enough to increase the capacity again.

5.30.2.4. After an RMA

When a Steelhead appliance in an Interceptor cluster is replaced, its data store will be new to the network. Therefore, the peering affinity algorithm will not forward any TCP sessions to that new Steelhead appliance. Only when the Interceptor appliances consider the other Steelhead appliances in the cluster to be in admission control, either for real or via capacity adjustments, then new TCP sessions will be forwarded to the new Steelhead appliance.

5.31. Expiring SSL certificates

The SSL certificates used on the Steelhead appliances, for the GUI, the SSL pre-optimization and for the SSL Secure Peering, have an expiry date on them. Once the SSL certificate has expired it cannot be used anymore.

Only when SSL Optimization is enabled, the expiry of the SSL certificates is noticed in the health status of the device. The list of expiring SSL certificates can be found with the command `show protocol ssl expiring-certs`.

Figure 5.217. SSL certificates have expired

```
SH # show protocol ssl
Enabled: yes
Protocol Versions: SSLv3_or_TLSv1
SFE Mode: Advanced_Only
Mid Session SSL: no
No server certificates.
[...]

SH # show alarms triggered
Alarm ID:          certs_expiring
Alarm Description: SSL Certificates Expiring
Status:            error
-------------------------------------
Alarm ID:          health
Alarm Description: Appliance Health
Status:            error
-------------------------------------
Alarm ID:          ssl
Alarm Description: SSL
Status:            error
-------------------------------------

SH # show protocol ssl expiring-certs
Peering certificate is OK.

All server certificates are OK.
All server chain certificates are OK.

Expiring/Expired CA certificate(s):
  Akamai_Subordinate_3 (on May 11 23:59:00 2013 GMT)
  Autoridad_de_Certificacion_Firmaprofesional_CIF_A62634068 (on Oct 24 22:00:00
2013 GMT)
  Digisign_Server_ID_Enrich (on Jul 17 15:16:55 2012 GMT)
  GlobalSign_Organization (on Jan 27 11:00:00 2014 GMT)
  Google_Internet (on Jun  7 19:43:27 2013 GMT)
  Microsoft_Code_Signing_PCA (on Aug 25 07:00:00 2012 GMT)

All peering CA certificates are OK.

All peering white list certificates are OK.

All mobile trusts certificates are OK.
```

In the log files this will be shown as:

Figure 5.218. SSL alarm is being raised

```
SH alarmd[5547]: [alarmd.NOTICE]: Alarm 'certs_expiring' triggering
SH alarmd[5547]: [alarmd.INFO]: Propagating changes for 1 alarms
SH alarmd[5547]: [alarmd.NOTICE]: Alarm 'ssl' triggering
SH alarmd[5547]: [alarmd.NOTICE]: Alarm 'health' triggering
SH mgmtd[3544]: [mgmtd.INFO]: EVENT:  /alarm/event/alarm/certs_expiring/triggered
SH mgmtd[3544]: [mgmtd.INFO]: Expiring/Expired SSL certificate(s) detected.
SH mgmtd[3544]: [mgmtd.INFO]: Expiring/Expired SSL certificate(s) have been detected.  For \
     more information, please check these pages:  http://SH/mgmt/gui?p=setupServiceProtoco \
    lsSSLMain  http://SH/mgmt/gui?p=setupServiceProtocolsSSLPeering  http://SH/mgmt/gui?p= \
    setupServiceProtocolsSSLCAs or use the CLI command "show protocol ssl expiring-certs"
```

The expired SSL certificates can be removed via the GUI or CLI, depending on their roles:

- Peering certificates: Under Configure -> Optimization -> Secure Peering (SSL) you can regenerate the certificate. From the CLI: `secure-peering generate-cert rsa`
- Server certificates: Under Configure -> Optimization -> SSL Main Settings you can update or remove the certificate. From the CLI: `no protocol ssl server-cert name <name>`
- Server chain certificates: From the CLI: `no protocol ssl server-cert name <name> chain-cert <name>`
- CA certificates: Under Configure -> Optimization -> Certifcate Authorities you can remove the certificate. This list can be large, sorting it on Expiration Date will help. From the CLI: `no protocol ssl ca <name>`
- Peering CA certificates: Under Configure -> Optimization -> Secure Peering (SSL) under the Peering Trust you can remove the certificate. From the CLI: `no secure-peering trust ca <name>`
- Peering white list certificates: Under Under Configure -> Optimization -> Secure Peering (SSL) under the Self-Signed Peer White List you can remove the certificate. From the CLI: `no secure-peering white-lst-peer <name>`
- Mobile trusts certificates: Under Under Configure -> Optimization -> Secure Peering (SSL) under the Mobile Trust certificate. From the CLI: `no secure-peering trust mobile <name>`

5.32. High CPU related issues

When there is a high CPU alarm on the Steelhead appliance, there can be various reasons for this:

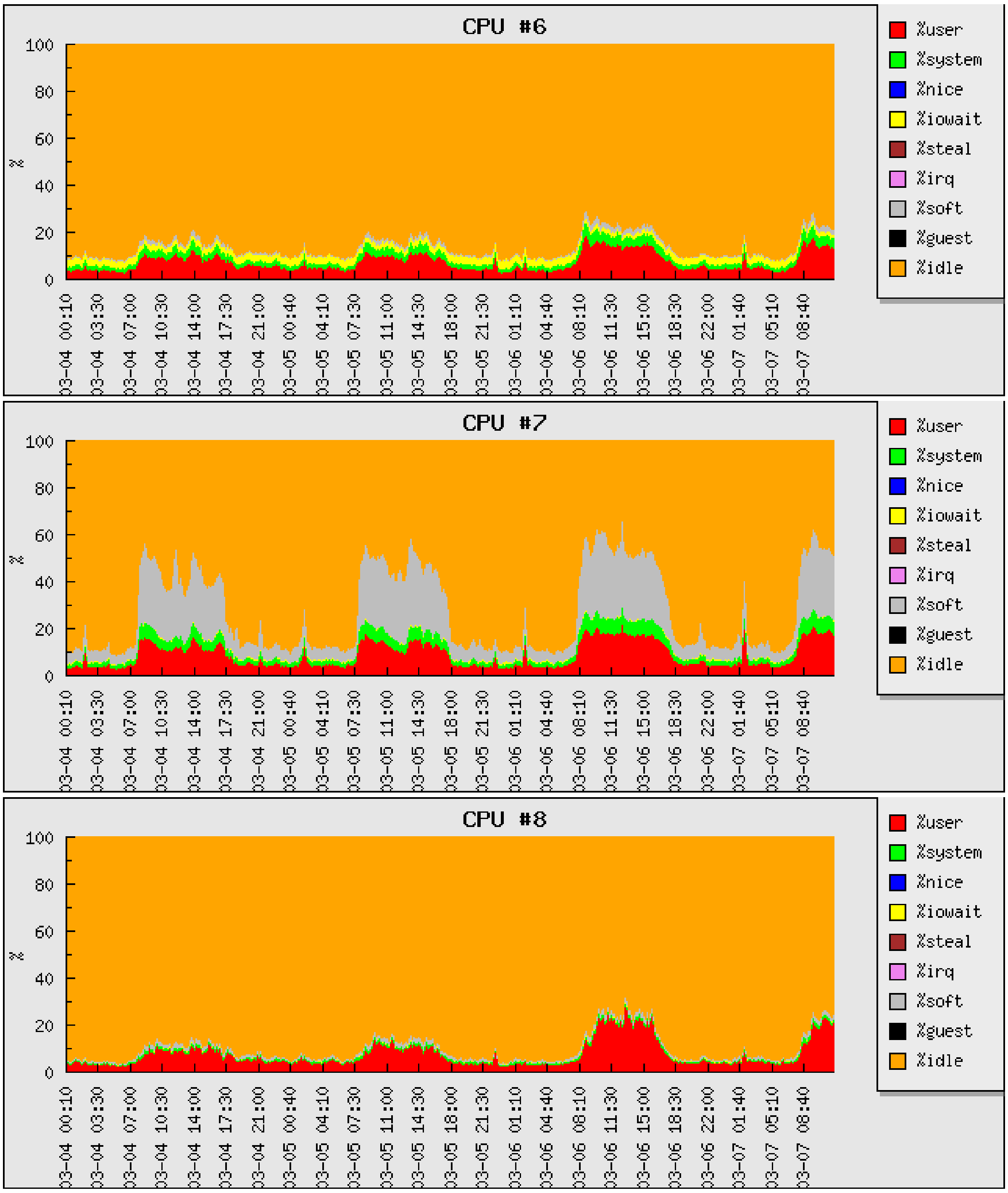
- High CPU due to traffic issues.
- High CPU due to RSP issues.
- High CPU due to spinning processes.

5.32.1. Traffic related CPU issues

If the CPU issues are related to the traffic, then the CPU load should follow the traffic pattern.

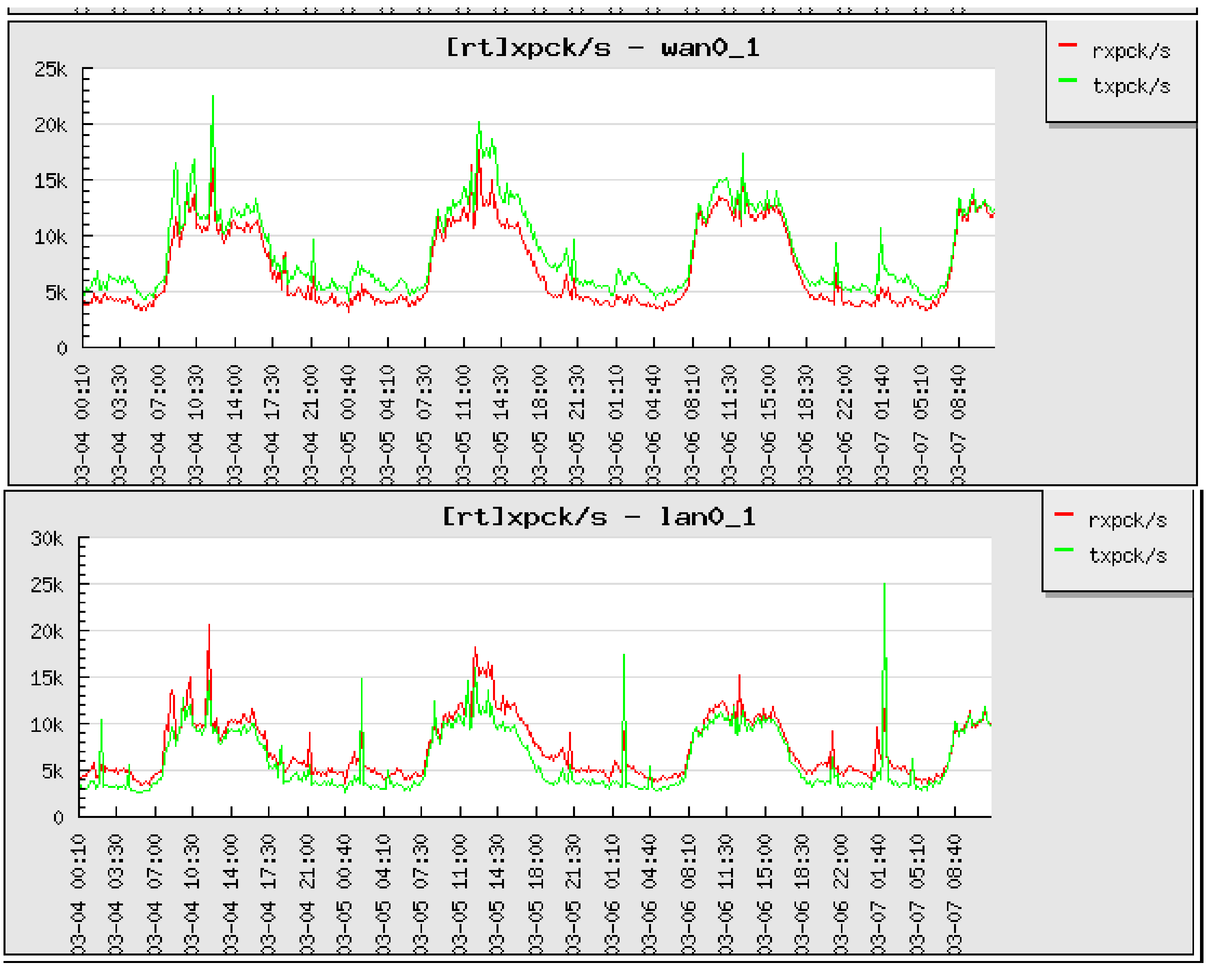
If the CPU load recorded in the System Accounting Records on this device show that CPU 0-6 and 8-11 have all more or less the same pattern and CPU 7 shows with a huge *%soft* value which might be related to the traffic:

Figure 5.219. CPU 7 shows a higher CPU load than the other ones



This is the traffic as in number of packets per second on the LAN side and on the WAN side as recorded in the System Accounting Records:

Figure 5.220. Number of packets going through the LAN and WAN side



So that confirms that the CPU load is following the traffic pattern, which means that the traffic is causing the CPU alarm.

5.32.1.1. NIC limitations and multi-queuing

For some of the NICs, the traffic is dealt with by the same CPU. This is fine for desktop and 1RU devices, however for the large 3RU devices this is often a bottleneck. Since RiOS 8.0 and higher, for some of the NICs the traffic can be distributed in multiple queues, spreading the load over multiple CPUs and thus preventing a single CPU to be overloaded.

If the NIC supports this feature can be seen in the section of `Output of 'cat /proc/interrupts'` the file *sysinfo.txt* in the system dump:

The first output shows that each NIC is dealt with by its own single CPU:

Figure 5.221. Distribution of interrupts for a NIC dealt with by a single CPU

```
            CPU0        CPU1        CPU2        CPU3
[...]
 48:     143          14     1217838          96   PCI-MSI-edge   wan0_0
 49:       3          88         108     1217838   PCI-MSI-edge   lan0_0
 50:      15           8     2573665          65   PCI-MSI-edge   wan0_1
 51:      54          99          82     8434328   PCI-MSI-edge   lan0_1
```

The second output shows that each NIC queue is spread over all CPUs:

Figure 5.222. Distribution of interrupts for a NIC dealt with by multiple CPUs

```
      CPU0       CPU1       CPU2       CPU3
[...]
 52:     4          0          0          0  PCI-MSI-edge  wan1_1
 53:    68  375249745  383988175  371645180  PCI-MSI-edge  wan1_1-rx-0
 54:    22  424257135  414943503  415290180  PCI-MSI-edge  wan1_1-rx-1
 55:    22  327132986  319572437  314607486  PCI-MSI-edge  wan1_1-rx-2
 56:    22  405144547  408061894  400918998  PCI-MSI-edge  wan1_1-rx-3
 57:    22  650505324  625281921  621856160  PCI-MSI-edge  wan1_1-tx-0
 58:  7663  540561142  529003647  513297250  PCI-MSI-edge  wan1_1-tx-1
 59:    22  668132358  654101138  626834780  PCI-MSI-edge  wan1_1-tx-2
 60:    27  640642031  624693149  605929490  PCI-MSI-edge  wan1_1-tx-3
```

The third output shows that each NIC queue is dealt with by its own single CPU:

Figure 5.223. Distrubution of interrupts for a NIC dealt with by multiple CPUs

```
      CPU0       CPU1       CPU2       CPU3
[...]
 88: 1379090814          0          0          0     PCI-MSI-edge      wan2_0-TxRx-0
 89:         97 1975362362          0          0     PCI-MSI-edge      wan2_0-TxRx-1
 90:         97          0 1616533530          0     PCI-MSI-edge      wan2_0-TxRx-2
 91:         97          0          0 1533039750     PCI-MSI-edge      wan2_0-TxRx-3
```

and in the output of the ethtool statistics:

Figure 5.224. Statistics show that there are multiple queues for TX and RX

```
Output of 'ethtool -S lan1_0':

NIC statistics:
     rx_packets: 63450051974
     tx_packets: 98718359686
     rx_bytes: 37546291563374
     tx_bytes: 37018865599885
[...]
     tx_queue_0_packets: 25155414255
     tx_queue_0_bytes: 9268449949519
     tx_queue_0_restart: 0
     tx_queue_1_packets: 23339319194
     tx_queue_1_bytes: 9115355264290
     tx_queue_1_restart: 0
     tx_queue_2_packets: 23978453361
     tx_queue_2_bytes: 8207159290199
     tx_queue_2_restart: 0
     tx_queue_3_packets: 26245172876
     tx_queue_3_bytes: 10032958161335
     tx_queue_3_restart: 0
     rx_queue_0_packets: 15784045932
     rx_queue_0_bytes: 9098239857902
     rx_queue_0_drops: 0
     rx_queue_0_csum_err: 0
     rx_queue_0_alloc_failed: 0
     rx_queue_1_packets: 15658010393
     rx_queue_1_bytes: 9148562150602
     rx_queue_1_drops: 0
     rx_queue_1_csum_err: 0
     rx_queue_1_alloc_failed: 0
     rx_queue_2_packets: 15809170269
     rx_queue_2_bytes: 9485746463221
     rx_queue_2_drops: 0
     rx_queue_2_csum_err: 0
     rx_queue_2_alloc_failed: 0
     rx_queue_3_packets: 16198806283
     rx_queue_3_bytes: 9559962031124
     rx_queue_3_drops: 0
     rx_queue_3_csum_err: 0
     rx_queue_3_alloc_failed: 0
```

If the right NICs is installed but the RiOS version is lower than 8.0, then upgrade to a newer RiOS version. If the right NIC is installed but not in use, then move the cables and reconfigure the IP addresses to the right in-path interface.

The full list of NICs supporting multiple queues is available in KB article S23409.

5.32.1.2. Bypass the traffic

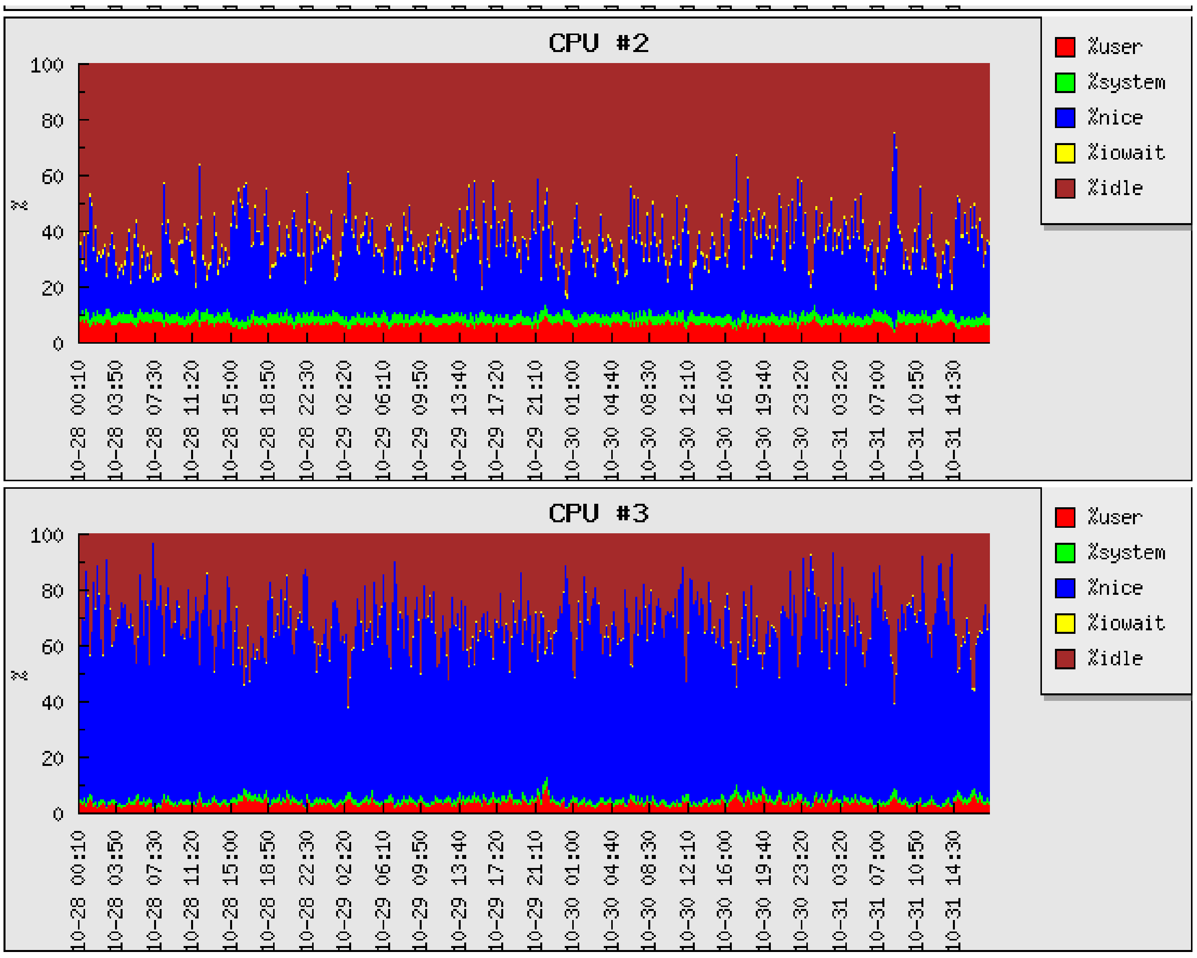
If the CPU is overloaded because of traffic, the most logical workaround would be to not send the traffic through the Steelhead appliance. There are several methods:

- If a large chunk of the traffic is not optimizable, for example UDP voice traffic towards a VLAN with VoIP phones, the traffic could be routed from the WAN router around the Steelhead appliance towards the LAN switch and vice-versa. This way the Steelhead appliance doesn't see it and thus not have to process it.
- If the NIC in the Steelhead appliance supports hardware by-pass, the traffic can be passed through on the NIC level instead of having to be dealt with in the kernel level on the Steelhead appliance.
- Change to a virtual in-path design where only the relevant traffic gets send to the Steelhead appliance. In case of WCCP and PBR deployment, use the right redirection access-lists to only forward optimizable traffic. In case of an Interceptor deployment, only optimizable traffic gets redirected to the Steelhead appliance.

5.32.2. RSP related CPU issues

In the System Accounting Records this could like look:

Figure 5.225. A lot of CPU usage is in the 'nice' category



The type is "Nice", which is indicates to RSP related process

The contents of the file *top_output* can show that the process causing the most load is the *vmware-vmx* process:

Figure 5.226. Single spinning CPU

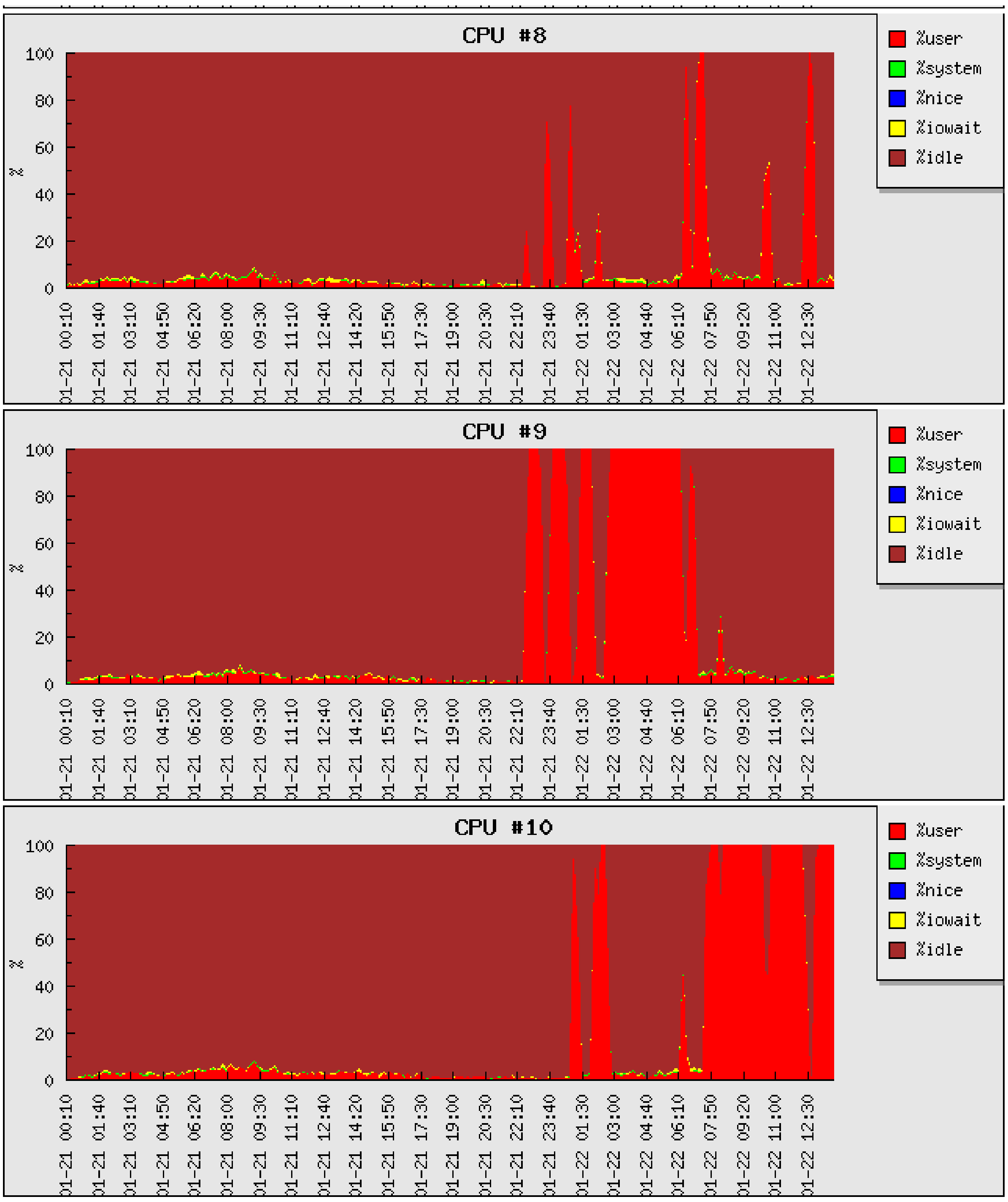
```
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 5436 admin     18   1 1065m 797m 539m R 98.7 10.0 169856:25 vmware-vmx
```

The next steps would be to determine what the RSP slot is doing.

5.32.3. Process related CPU issues

In the System Accounting Records this could like look:

Figure 5.227. A userland process is spinning on a CPU



What you see here is that since 21 January around 22:10, suddenly there were high CPU spikes on all CPUs. It can be on all CPUs, it can just be stuck on one CPU. The type is "User", which means that it is a userland process.

The contents of the file *top_output* can show that a single process uses 100% of the CPU time:

Figure 5.228. Single spinning CPU

```
  PID USER       PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1414 admin      25   0 81176  45m  41m R 100.9  0.1   1005:54 winbindd
```

That 100.9% CPU means that it is spinning in a loop, consuming a full CPU. The expected behaviour would be that the winbindd process will only be woken up when Active Directory integration related tasks are running, for example when a new encrypted MAPI session gets setup.

5.33. Central Management Console related issues

The CMC manages the configuration of the Steelhead appliances in the network and monitors the health of the Steelhead appliances, Steelhead Mobile Controllers, Interceptors and Whitewater appliances.

5.33.1. CMC connection to the Steelhead appliance

The CMC connects to a remote Steelhead appliance on TCP port 22 via the SSH protocol. It logs in and executes the command *rgp*.

Figure 5.229. CMC 'CMC' connects to a Steelhead appliance

```
SH sshd[9016]: Accepted password for admin from 192.168.1.102 port 59677 ssh2
SH sshd[9016]: pam_unix(sshd:session): session opened for user admin by (uid=0)
SH cli[9359]: [cli.NOTICE]: user admin: executing remote command: /opt/tms/bin/rgp CMC DA2 \
    HV0001234F
SH rgp[9359]: [rgp.NOTICE]: rgp starting at 2014/01/14 07:34:16
SH rgp[9359]: [rgp.INFO]: session 1: connected to server 'rgpd' as authenticated user 'adm \
    in' (uid 0, gid 0) authorization key 'admin', interactive 'false'
SH rgpd[8624]: [rgpd.INFO]: session 2: accepted client 'rgp-9359' for authenticated user ' \
    admin' (uid 0, gid 0) authorization key 'admin', interactive 'true'
SH rgpd[8624]: [rgpd.INFO]: Getting auth from new outer session
SH rgp[9359]: [rgp.NOTICE]: session 1: existing connection from
SH rgp[9359]: [rgp.INFO]: Sending first connect event
SH mgmtd[7342]: [mgmtd.INFO]: EVENT:  /cmc/event/connect
SH rgp[9359]: [rgp.INFO]: session 1: accepted client 'rbmd-4108' for authenticated user 'a \
    dmin' (uid 0, gid 0) authorization key 'admin', interactive 'true'
SH rgpd[8624]: [rgpd.INFO]: cli_init_pam(), cli_auth.c:91, build (null): Successfully star \
    ted pam session for the user admin from <unknown>
```

The CMC starts the process *rgp* and specifies the hostname of the CMC, *CMC* in this case, and the serial number of the Steelhead appliance.

To see if a Steelhead appliance is connected to a CMC and which CMC it is connected to, use the command `show cmc`.

Figure 5.230. Output of the command 'show cmc'

```
SH # show cmc
CMC auto-registration enabled:       yes
CMC auto-registration hostname:      riverbedcmc
Managed by CMC:                      yes
CMC hostname:                        CMC (192.168.1.102)
Auto configuration status:           Inactive
Last message sent to cmc:            Auto-registration
Time that message was sent:          Thu Jan  9 13:36:39 2014
```

5.33.2. CMC auto-registration

The Steelhead appliance can inform the CMC that it is up and running and that the CMC can connect to it. It tries to resolve the hostname *riverbedcmc* and sends an HTTP request to it on TCP port 443:

Figure 5.231. Auto-registration by the Steelhead appliance

```
17:51:06.614599 IP 10.0.1.5.14661 > 192.168.1.1.53: 2495+ A? riverbedcmc.example.org. (45)
17:51:06.614608 IP 10.0.1.5.14661 > 192.168.1.1.53: 2687+ AAAA? riverbedcmc.example.org. ( \
    45)
17:51:06.803037 IP 192.168.1.1.53 > 10.0.1.5.14661: 2495* 1/0/0 A 192.168.1.102 (61)
17:51:06.803275 IP 192.168.1.1.53 > 10.0.1.5.14661: 2687* 0/1/0 (96)
17:51:06.803368 IP 10.0.1.5.24332 > 192.168.1.102.443: Flags [S], seq 4235884934, win 5840 \
    , options [mss 1460,sackOK,TS val 2859884311 ecr 0,nop,wscale 2], length 0
17:51:06.993913 IP 192.168.1.102.443 > 10.0.1.5.24332: Flags [S.], seq 1834967271, ack 423 \
    5884935, win 5792, options [mss 1304,sackOK,TS val 1233057525 ecr 2859884311,nop,wscal \
    e 7], length 0
17:51:06.993930 IP 10.0.1.5.24332 > 192.168.1.102.443: Flags [.], seq 1, ack 1, win 1460,  \
    options [nop,nop,TS val 2859884501 ecr 1233057525], length 0
```

5.33.3. CMC Managed Appliance Alarms

The CMC will collect data from all Steelhead appliances, analyses this data and raise an alarm on the CMC while the Steelhead appliance itself is healthy. This report can be found in the GUI under Reports -> Appliance Diagnostics -> Appliance Details.

Examples of this alarms are:

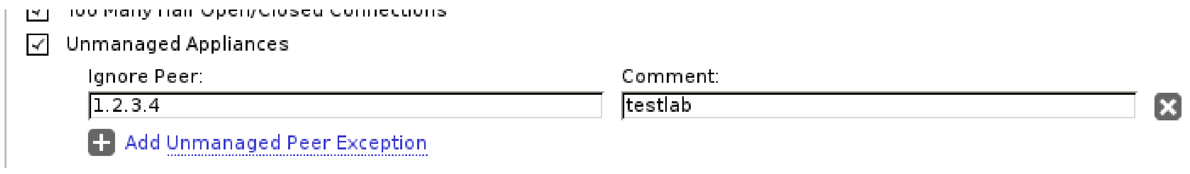
- Unmanaged appliances
- Time drift
- Duplex alarm
- Configuration change
- High usage and Connection Limit Warning
- Too many half-open or half-closed connections
- PFS and RSP
- Poll timeout

5.33.3.1. Unmanaged appliances

This alarm gets raised when the Steelhead appliance is peering with another Steelhead appliance this CMC is not managing.

The presence of an unknown peer Steelhead appliance which is not managed by the CMC can be caused by an oversight of the networking team or a problem in the network. If the origin of the unknown Steelhead appliance has been determined, it can either be added under control of the CMC or it can be added to a blacklist under Configure -> System Settings -> Alarms -> CMC Managed Appliance Alarms -> Unmanaged Appliances:

Figure 5.232. Add Steelhead appliances to the unmanaged peer whitelist.


Too Many Half Open/Closed Connections
✓ Unmanaged Appliances
Ignore Peer: 1.2.3.4
Comment: testlab

5.33.3.2. Time drift

This alarm gets raised when the time on the Steelhead appliance is significantly different from the time on the CMC. It might indicate a broken NTP configuration on the Steelhead appliance, which might interfere with the Windows Active Directory integration.

5.33.3.3. Duplex alarm

This alarm gets raised when there are frame errors detected on the Steelhead appliance. The next steps would be to inspect the Steelhead appliance interface NIC statistics and determine why this alarm got raised.

5.33.3.4. Configuration change

This alarm gets raised when the configuration on the Steelhead appliance has changed. Since the configuration is managed via the CMC, an uncontrolled change has happened and should be investigated.

5.33.3.5. High usage and Connection Limit Warning

These alarms get raised when the Steelhead appliance is close to Connection-based Admission Control.

5.33.3.6. Too many half-open or half-closed connections

This alarm gets raised when there are many half-opened or half-closed TCP connections on the Steelhead appliance. This can happen when there are problems in the auto-discovery process or during the setup of the optimized TCP session or in the teardown of the optimized TCP session.

5.33.3.7. PFS and RSP

This alarm gets raised when both RSP and PFS are configured on a Steelhead appliance. These two services are mutually exclusive.

5.33.3.8. Poll timeout

This alarm gets raised when the commands submitted by the CMC to the Steelhead appliance take a long time to obtain responses. The next step would be a health check of the Steelhead appliance.

Chapter 6. System Dump

6.1. Index

A system dump is a collection of historical data and a snapshot of the state of the Steelhead appliance.

It contains the following sections:

- Creation of the system dump and how to provide it to the Riverbed TAC.
- The current configuration of the device.
- The log files of the Steelhead appliance.
- Hardware information about the Steelhead appliance.
- System information about the Steelhead appliance.
- The alarms on a Steelhead appliance.
- Simplified routing.
- SMART data.
- Asymmetric routing.
- Image history.
- lsof.
- Memory dump of the optimization process.
- Out-of-memory profiling.
- CIFS pre-population.
- RSP related data.
- SDR statistics.
- Statistic files.
- SAR statistics.
- Active Directory information.
- Process dumps.

6.2. Creation of a system dump

To generate a system dump from the CLI, issue the commands `debug generate dump` (with the possible options of *stats* to include the statistics and *all-logs* to include all the log files). Then use the command `file debug <system dump> upload <URL>` to upload the system dump to a remote FTP server or SSH server. To see a list of all system dumps from the CLI, use the command `show files debug-dump`.

Figure 6.1. Create a system dump from the CLI

```
SH # debug generate dump
Systemdump sysdump-SH-20111010-163012.tgz generated
SH # show files debug-dump
sysdump-SH-20111010-163012.tgz
SH # file debug-dump sysdump-SH-20111010-163012.tgz upload ftp://192.168.1.1/incoming/sysd \
    ump-SH-20111010-163012.tgz
SH #
```

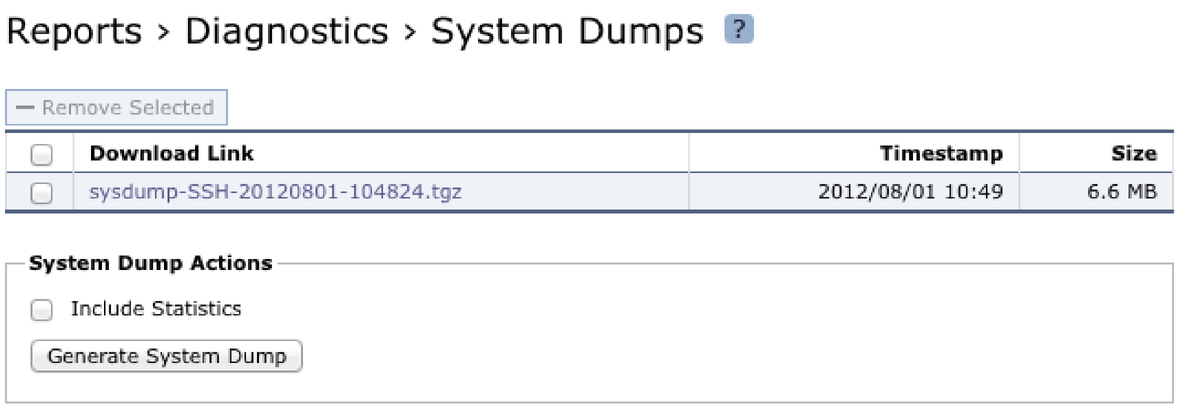
Since RiOS version 8.5, the system dump can be directly uploaded to the Riverbed FTP server if Steelhead appliance has direct access to the Internet:

Figure 6.2. Upload a system dump directly to the Riverbed FTP server

```
SH # file debug-dump sysdump-SH-20111010-163012.tgz case 123456
```

To generate a system dump from the GUI, go to Reports -> Diagnostics -> System dumps and click on the *Generate System Dump* button. When the system dump is completed you can download it to your computer.

Figure 6.3. Create a system dump from the GUI



	Download Link	Timestamp	Size
☐	sysdump-SSH-20120801-104824.tgz	2012/08/01 10:49	6.6 MB

If there is not enough disk space available or there are too many system dumps, the Steelhead appliance will complain:

Figure 6.4. Not being able to generate a system dump

```
SH # debug generate dump all-logs
% Insufficient disk space to perform sysdump.

SH # debug generate dump
% Limit on number of sysdumps on the system is 10,
please remove some before generating new ones
```

Use the commands in the section about *Partitions running out of space* to free up the disk space.

As can be seen from the name, the system dump is a gzip compressed tar-ball. The filename contains the hostname of the Steelhead appliance and the date and time the system dump was created.

Note that over time, new features are added to the system dump process. It could be possible that some files or sections described here are not available in earlier RiOS versions.

6.2.1. Dealing with a system dump

As the extension shows, the system dump is a gzip compressed tar-ball. The standard *tar* command can be used to extract it: `tar zxvf <system dump>`. Some of the files in it will still be compressed, use the *gunzip* command to decompress them: `gunzip sydump*/*.gz`.

6.3. Configuration of the system

This section describes the configuration of the Steelhead appliance as seen in the system dump.

6.3.1. The configuration of the optimization service

The configuration of the system, as seen by the output of the command `show running`, can be found in the file *active-running.txt*.

Figure 6.5. Contents of the active-running.txt file

```
##
## Network interface configuration
##
   interface inpath0_0 description ""
no interface inpath0_0 dhcp
no interface inpath0_0 dhcp dynamic-dns
no interface inpath0_0 force-mdi-x enable
   interface inpath0_0 ip address 10.17.6.100 /25
   interface inpath0_0 mtu "1500"
no interface inpath0_0 shutdown
   interface inpath0_0 speed "auto"
   interface inpath0_0 txqueuelen "100"
   interface lan0_0 description ""
```

The output of the command `show running full` can be found in the file *active-running-full.txt*.

There are more files which names start with *active*, like the *active.db*, *active.txt* and *active-brief.txt*. These are the configuration files which contain all of the knobs and buttons of the optimization service, but in an internal format.

Further there are files which names start with *local*, like the *local.txt* and *local-brief.txt*. These are also part of the configuration but with fields part of auto-configuration like the license keys.

6.3.2. The configuration of the various daemons

In the directory *output* are the configurations of the various daemons of the operating system, like the syslog daemon and the SNMP daemon:

Figure 6.6. Configuration of the syslog daemon and SNMP daemon

```
[~/CSH-20110530-163605] edwin@t43>cat output/syslog.conf
##
## This file was AUTOMATICALLY GENERATED.  DO NOT MODIFY.
## Any changes will be lost.
##
## Generated by md_syslog at 2011/05/24 16:07:00
##
*.notice                        -/var/log/messages
local1.*                        -/var/log/user_messages

[~/CSH-20110530-163605] edwin@t43>cat output/snmpd.conf
##
## This file was AUTOMATICALLY GENERATED.  DO NOT MODIFY.
## Any changes will be lost.
##
## Generated by md_snmp at 2011/05/24 16:07:11
##
rocommunity public
trapcommunity public
trapinterface primary
sysServices 78
engineID 0x8000430b805525428b4b8706c7
```

6.4. Log files

There are various system logs included in the system dump of the Steelhead appliance:

6.4.1. System logs

The messages files (messages, messages.1.gz, messages.2.gz etc) are the system logs which can also be seen on the CLI with the command `show log` and `show log files N`.

The first line in the system logs is the version of RiOS running on the Steelhead appliance at the time the system logs get rotated.

By default the logging level of the Steelhead appliances is set to *notice* level, but when investigating problems *info level logging* might be required.

Figure 6.7. Beginning of the file "messages"

```
BUILD_PROD_VERSION:              rbt_sh 6.5.0a #84_26 2011-04-04 20:34:46 i386 root@misko \
    lc:svn://svn/mgmt/branches/canary_84_fix_branch
May 28 00:00:00 CSH syslogd 1.4.1: restart.
May 28 00:00:21 CSH sport[6693]: [smbsfe.NOTICE] 15446 {10.0.1.1:53286 192.168.1.1:445} Cl \
    ient and Server negotiated SMB2 protocol SMB2.Turning OFF CIFS optimizations. However  \
    SDR optimizations will continue.
May 28 00:00:53 CSH sport[6693]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 10 \
    .45.5.3
May 28 00:00:53 CSH sport[6693]: [splice/oob.NOTICE] 10 {- -} Lost OOB Splice between ladd \
    r=10.33.0.91:7800 and raddr=10.45.5.3:7800
```

The format of the lines logged by the optimization service is:

Figure 6.8. Format of the log messages by the optimization service

```
<date> <time> <hostname> sport[process id] <functionality.loglevel> <splice-id> {<client I \
    P address:client TCP port> <server IP address:server TCP port>} <message>
```

The *date* only contains the month name and the date, not the year. The *date* and *time* do not contain the time zone configured on the Steelhead appliance. The loglevel, or severity, is the importance of the message shown, it can be info, notice, warn(ing), err(or) and crit(ical). The *splice-id* is a sequence number of the inner channel being setup. if no splice-id can be determined it will be replaced by a "-" (dash) and the IP address and TCP ports will be replaced by a dash too.

The *functionality* can be the part of the "sport" optimization service:

- splice/probe: Messages related to the auto-discovery of optimized TCP sessions.
- io/inner/prod: Messages related to the setup of an optimized TCP session.
- splice/oob: The Out-of-Band Splice between two Steelhead appliances.
- connect_pool: The list of TCP sessions part of the Connection Pool.
- mapi/*: MAPI latency optimization related messages.
- smbcfe: CIFS client-side latency optimization related messages.
- smbsfe: CIFS server-side latency optimization related messages.
- http/*: HTTP latency optimization related messages.

Or from other processes on the Steelhead appliances:

- kernel / intercept: Messages in related to the intercept kernel-module which tracks the setup of new optimized TCP sessions.
- mgmtd: Messages related to the management daemon.
- pm: Messages related to the process manager.

6.4.2. GUI logs

The GUI logs are logged in the files *web_access.log* and *web_error.log*, which are the normal Apache-style logging.

The file *web_access.log* contains the IP address the request came from, the date and time, the URL requested, the return result and the size of the payload returned.

Figure 6.9. Beginning of the file "web_access.log"

```
10.0.1.1 - - [30/May/2011:16:20:02 +1000] "GET /rollup-product.css?v=1001518176 HTTP/1.1"  \
    304 -
10.0.1.1 - - [30/May/2011:16:20:03 +1000] "POST /mgmt/xmldata?p=inpathRules HTTP/1.1" 200  \
    489
10.0.1.1 - - [30/May/2011:16:20:07 +1000] "GET /images/aet_edit_close.png HTTP/1.1" 200 17 \
    3
10.0.1.1 - - [30/May/2011:16:20:18 +1000] "POST /mgmt/xmldata?p=dynamicStatus HTTP/1.1" 20 \
    0 98
```

The file *web_error.log* contains the date and time and the reported issue.

Figure 6.10. Beginning of the file "web_error.log"

```
[Mon May 30 00:00:00 2011] [notice] Apache configured -- resuming normal operations
[Mon May 30 16:20:00 2011] [notice] Graceful restart requested, doing restart
[Mon May 30 16:20:00 2011] [notice] Apache configured -- resuming normal operations
```

6.4.3. Kernel log

The kernel log of the Steelhead appliance is recorded in the file *dmesg*. This kernel log is a circular buffer which gets populated from the moment the Steelhead appliance gets booted. If the system dump is taken soon after the reboot, it will contain the whole boot log.

Figure 6.11. Part of the file "dmesg"

```
Linux version 2.6.9-34.EL-rbt-7989SMP (root@miskolc.lab.nbttech.com) (gcc version 3.4.6 20 \
    060404 (Red Hat 3.4.6-10)) #2 SMP Fri Feb 4 18:54:45 PST 2011
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000000000 - 000000000009fc00 (usable)
 BIOS-e820: 000000000009fc00 - 00000000000a0000 (reserved)
 BIOS-e820: 00000000000e0000 - 0000000000100000 (reserved)
 BIOS-e820: 0000000000100000 - 00000000dffd0000 (usable)
```

The file *dmesg-boot*, available since RiOS version 8.5, contains a copy of the *dmesg* file just after the startup of the Steelhead appliance.

6.4.4. CLI history log

The file *cli_history_root* contains the commands entered on the CLI by the admin user. For example, this is the history of the upgrade of the Steelhead appliance:

Figure 6.12. Part of the file "cli_history_root"

```
 20110524155032 0
show run
 20110524155047 0
no interface primary shutdown
 20110524155500 0
image fetch *
 20110524160139 0
image install sh-i386-6.5.0a
 20110524160316 0
image install sh-i386-6.5.0a 1
 20110524160347 0
image boot 1
 20110524160349 0
wr m
 20110524160349 0
reload
```

The numbers are the year-month-day-hour-minute-second format.

There are lines which could contain a username or password. That part of the line has been removed, for example in the `image fetch` command.

There is a similar file for the monitor user called *cli_history_monitor*.

6.4.5. Optimization service memory log

Under normal operations, the Steelhead appliance is logging under the Notice priority. But a lot of interesting information is logged in the optimization service under the Info level.

The optimization service keeps the last 16 kilobytes of the messages logged under Info level and higher. When the optimization service logs an error with a priority of Error or higher (Error, Critical or Emergency), it will write this to the file `/var/log/memlog`. This way important information of the optimization service in case of an error in the optimization service can be caught even if specific Info level logging hasn't been enabled.

Figure 6.13. First ten lines of the file memlog.1

```
[Sep 14 15:26:25 19591 659750 /splice/client INFO] {57.200.61.29:1993 57.200.2.20:8014} St \
    art flowing, lport 38856, rport 7800, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_A \
    UTO, TRANSPORT_ID_NONE, TPTOPT_NONE(0x0), TRPY_FULL
[Sep 14 15:26:25 19591 -1 /http/cache INFO] {- -} decode decode DONE
[Sep 14 15:26:25 19591 659750 /splice/client INFO] {57.200.61.29:1993 57.200.2.20:8014} fi \
    ni client 57.200.61.29:1993 server 57.200.2.20:8014 cfe 57.200.157.219:38856 sfe 172.1 \
    9.18.152:7800 app TCP
[Sep 14 15:26:25 19591 -1 /http/cache INFO] {- -} decode decode DONE
[Sep 14 15:26:25 19591 659751 /splice/client INFO] {57.200.61.29:1994 57.200.2.20:8014} in \
    it client 57.200.61.29:1994 server 57.200.2.20:8014 cfe 57.200.157.219:7801 sfe 172.19 \
    .18.152:7800 client connected: yes
[Sep 14 15:26:25 19591 659751 /splice/client INFO] {57.200.61.29:1994 57.200.2.20:8014} tr \
    py: TRPY_FULL, csum 0 local: 57.200.61.29:1994 remote: 57.200.2.20:8014
[Sep 14 15:26:25 19591 -1 /http/cache INFO] {- -} decode decode DONE
[Sep 14 15:26:25 19591 -1 /http/cache INFO] {- -} decode decode DONE
[Sep 14 15:26:25 19591 -1 /http/cache INFO] {- -} decode decode DONE
[Sep 14 15:26:25 19591 659751 /splice/client INFO] {57.200.61.29:1994 57.200.2.20:8014} Sp \
    lice client side initializing: No protocol port = 8014 protocol id = TCP(0) transport  \
    = TRANSPORT_ID_NONE
```

6.4.6. Log file management

6.4.6.1. Period based log file rotation

By default the system logs are rotated once every day at midnight:

Figure 6.14. Output of the "show logging" command, daily rotation

```
SH # show logging
Local logging level: notice
Default remote logging level: notice
Number of archived log files to keep: 10
Log rotation frequency: daily
```

However, every twenty minutes the Steelhead appliance will check if the file size of the current log file is more than 1 Gb. If it is more than 1 Gb, it will force a rotation of it.

6.4.6.2. Size based log file rotation

The log file rotation can be set to a size based rotation scheme, for example when the 20 minute check interval is too small.

Figure 6.15. Output of the "show logging" command, rotation by size

```
SH (config) # logging files rotation criteria size 1000
SH (config) # show logging
Local logging level: notice
Default remote logging level: notice
Number of archived log files to keep: 10
Log rotation size threshold: 1000 megabytes
SH (config) # logging files rotation criteria frequency daily
```

6.4.6.3. Manual log file rotation

To force a log file rotation, use the command `logging files rotation force`.

To manually remove a set of log files, use the command `logging files delete oldest`.

Figure 6.16. Manually removing log files

```
SH # logging files delete oldest ?
<cr>             Delete the single oldest log file
<number>         Select the number of oldest log files to delete
SH # logging files delete oldest 3
SH #
```

You can download previous system logs via the GUI under Reports -> Diagnostics -> System Logs Download.

6.5. Hardware Information

The file *hardwareinfo.txt* contains information of the Steelhead appliance like the DMI table, the sensors on the motherboard, the PCI bus details, the hard disk and RAID status and IPMI information.

6.5.1. DMI and SMBIOS

The DMI (Desktop Management Interface) or SMBIOS (System Management BIOS) table is an overview of the devices of the Steelhead appliance like the motherboard, memory configuration, SATA controllers, USB controllers etc.

6.5.2. Sensors

The sensors section shows the state of the fan-speeds, line voltages and temperature sensors.

Figure 6.17. Output of the sensors section on the 100 / 200 / 300 models

```
w83627hf-isa-0290
Adapter: ISA adapter
VCore 1:   +1.22 V  (min =  +0.00 V, max =  +0.00 V)
VCore 2:   +1.04 V  (min =  +0.00 V, max =  +0.00 V)
+3.3V:     +3.28 V  (min =  +3.14 V, max =  +3.46 V)
+5V:       +4.89 V  (min =  +4.73 V, max =  +5.24 V)
+12V:     +11.31 V  (min = +10.82 V, max = +13.19 V)
-12V:      +1.13 V  (min = -13.18 V, max = -10.88 V)
-5V:       +2.54 V  (min =  -5.25 V, max =  -4.75 V)
V5SB:      +5.38 V  (min =  +4.73 V, max =  +5.24 V)
VBat:      +3.17 V  (min =  +2.40 V, max =  +3.60 V)
fan1:     5818 RPM  (min = 2657 RPM, div = 2)
fan2:        0 RPM  (min = 2657 RPM, div = 2)
fan3:        0 RPM  (min = 2657 RPM, div = 2)
temp1:       +52 C  (high =    -1 C, hyst =   -65 C)   sensor = thermistor
temp2:     +39.0 C  (high =  +120 C, hyst =  +115 C)   sensor = PII/Celeron diode
temp3:     -47.5 C  (high =  +120 C, hyst =  +115 C)   sensor = PII/Celeron diode
```

Figure 6.18. Output of the sensors section on the 520 / 1020 / 1520 / 2020 models

```
bmc-i2c-0-00
Adapter: IPMI adapter
in1:       +3.07 V  (min =  +2.64 V, max =  +0.00 V)
fan1:     3150 RPM  (min =  750 RPM)
fan2:     3150 RPM  (min =  750 RPM)
fan3:     1350 RPM  (min = 1425 RPM)
temp1:     +41.0 C  (high =  +125 C, hyst =  -127 C)
temp2:     +31.0 C  (high =   +53 C, hyst =  -127 C)
```

Figure 6.19. Output of the sensors section on the 3020 / 3520 / 5520 / 6020 / 6120 models

```
adm1026-i2c-1-2c
Adapter: SMBus2 AMD8111 adapter at cc00
```

```
in0:       +2.31 V  (min =  +0.00 V, max =  +2.99 V)
in1:       +1.59 V  (min =  +2.25 V, max =  +2.75 V)   ALARM
in2:       +2.62 V  (min =  +2.25 V, max =  +2.75 V)
in3:       +1.22 V  (min =  +2.25 V, max =  +2.75 V)   ALARM
in4:       +0.00 V  (min =  +0.00 V, max =  +2.99 V)   ALARM
in5:       +0.00 V  (min =  +0.00 V, max =  +2.99 V)   ALARM
in6:       +0.00 V  (min =  +0.00 V, max =  +2.49 V)   ALARM
in7:       +1.43 V  (min =  +1.02 V, max =  +1.68 V)
in8:       +0.00 V  (min =  +0.00 V, max =  +2.49 V)
in9:       +0.44 V  (min =  +0.00 V, max =  +0.70 V)
in10:      +3.30 V  (min =  +2.97 V, max =  +3.64 V)
in11:      +3.42 V  (min =  +0.00 V, max =  +4.42 V)
in12:      +3.35 V  (min =  +2.97 V, max =  +3.64 V)
in13:      +5.02 V  (min =  +4.50 V, max =  +5.49 V)
in14:      +1.43 V  (min =  +1.02 V, max =  +1.68 V)
in15:     +12.19 V  (min = +10.81 V, max = +13.19 V)
in16:     -11.95 V  (min = -13.18 V, max = -10.80 V)
fan0:     5113 RPM  (min =  712 RPM, div = 8)
fan1:     5113 RPM  (min =  712 RPM, div = 8)
fan2:     4963 RPM  (min =  712 RPM, div = 8)
fan3:     5113 RPM  (min =  712 RPM, div = 8)
fan4:     5113 RPM  (min =  712 RPM, div = 8)
fan5:     4963 RPM  (min =  712 RPM, div = 8)
fan6:        0 RPM  (min =  712 RPM, div = 8)   ALARM
fan7:        0 RPM  (min =  712 RPM, div = 8)   ALARM
temp1:       +34 C  (low  =    +0 C, high =   +80 C)
temp2:       +50 C  (low  =    +0 C, high =   +72 C)
temp3:       +45 C  (low  =    +0 C, high =   +72 C)

w83627hf-isa-0290
Adapter: ISA adapter
VCore 1:   +1.30 V  (min =  +0.00 V, max =  +0.00 V)
VCore 2:   +1.31 V  (min =  +0.00 V, max =  +0.00 V)
+3.3V:     +3.33 V  (min =  +3.14 V, max =  +3.47 V)
+5V:       +5.03 V  (min =  +4.76 V, max =  +5.24 V)
+12V:     +12.22 V  (min = +10.82 V, max = +13.19 V)
-12V:     -11.70 V  (min = -13.18 V, max = -10.80 V)
-5V:       -1.93 V  (min =  -5.25 V, max =  -4.75 V)
V5SB:      +5.32 V  (min =  +4.76 V, max =  +5.24 V)       ALARM
VBat:      +2.59 V  (min =  +2.40 V, max =  +3.60 V)
fan1:        0 RPM  (min = 2657 RPM, div = 2)
fan2:        0 RPM  (min = 2657 RPM, div = 2)
fan3:        0 RPM  (min = 2657 RPM, div = 2)
temp1:       +39 C  (high =  +120 C, hyst =  +115 C)   sensor = thermistor

temp2:     +43.0 C  (high =   +80 C, hyst =   +75 C)   sensor = thermistor

temp3:     +37.0 C  (high =   +80 C, hyst =   +75 C)   sensor = thermistor

ERROR: Can't get VID data!
alarms:
beep_enable:
          Sound alarm disabled
```

Figure 6.20. Output of the sensors section on the 150 / 250 / 550 models

```
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/cpu0_vid: "1500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/detach_state: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_input: "3624
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_min: "2500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_input: "3619
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_min: "2500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_input: "3673
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_min: "2500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_input: "0
```

```
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_input: "5026
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_max: "6641
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_input: "1192
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_max: "2988
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_input: "3321
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_max: "4383
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_input: "5016
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_max: "6641
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_input: "12188
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_max: "15938
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_input: "3309
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_max: "4383
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_input: "3221
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_max: "4383
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/name: "dme1737
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1: "103
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_channels_zone: "7
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_point1_pwm: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_point2_pwm: "255
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_pwm_min: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_enable: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_freq: "25000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_ramp_rate: "206
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2: "102
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_channels_zone: "7
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_point1_pwm: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_point2_pwm: "255
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_pwm_min: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_enable: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_freq: "25000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_ramp_rate: "206
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3: "102
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_channels_zone: "7
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_point1_pwm: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_point2_pwm: "255
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_pwm_min: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_enable: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_freq: "25000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_ramp_rate: "206
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_fault: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_input: "59000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_max: "127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_min: "-127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_offset: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_fault: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_input: "39000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_max: "127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_min: "-127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_offset: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_fault: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_input: "41000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_max: "127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_min: "-127000
```

```
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_offset: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/vrm: "14
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_channels_temp: "1
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point1_temp: "55000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point1_temp_hyst: "51000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point2_temp: "75000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point3_temp: "80000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_channels_temp: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point1_temp: "42000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point1_temp_hyst: "38000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point2_temp: "52000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point3_temp: "62000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_channels_temp: "4
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point1_temp: "47000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point1_temp_hyst: "43000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point2_temp: "57000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point3_temp: "67000
```

Figure 6.21. Output of the sensors section on the 1050 / 2050 models

```
bmc-i2c-1-00
Adapter: IPMI adapter
in1:       +1.11 V  (min =  +0.97 V, max =  +1.43 V)
in2:       +0.00 V  (min =  +0.97 V, max =  +1.43 V)
in3:       +1.14 V  (min =  +0.95 V, max =  +1.28 V)
in4:       +5.06 V  (min =  +4.63 V, max =  +5.33 V)
in5:       +1.47 V  (min =  +1.38 V, max =  +1.58 V)
in6:       +1.79 V  (min =  +1.66 V, max =  +1.90 V)
in7:      +11.81 V  (min = +11.14 V, max = +12.77 V)
in8:       +3.34 V  (min =  +3.06 V, max =  +3.52 V)
in9:       +4.87 V  (min =  +4.63 V, max =  +5.33 V)
fan1:     8760 RPM  (min = 1080 RPM)
fan2:     9360 RPM  (min = 1080 RPM)
fan3:     9480 RPM  (min = 1080 RPM)
fan4:        0 RPM  (min = 1080 RPM)
fan5:     9120 RPM  (min = 1080 RPM)
fan6:        0 RPM  (min = 1080 RPM)
fan7:     9120 RPM  (min = 1080 RPM)
fan8:        0 RPM  (min = 1080 RPM)
fan9:        0 RPM  (min = 1080 RPM)
fan10:       0 RPM  (min = 1080 RPM)
temp1:     +42.0 C  (high =   +95 C, hyst =   +10 C)
temp2:    +100.0 C  (high =   +95 C, hyst =   +10 C)
temp3:     +39.0 C  (high =   +55 C, hyst =  -123 C)
temp4:     +46.0 C  (high =   +55 C, hyst =  -123 C)
temp5:     +42.0 C  (high =   +57 C, hyst =  -123 C)
temp6:     +47.0 C  (high =   +57 C, hyst =  -123 C)
temp7:     +35.0 C  (high =   +57 C, hyst =  -123 C)
temp8:     +36.0 C  (high =   +57 C, hyst =  -123 C)
temp9:     +40.0 C  (high =   +57 C, hyst =  -123 C)
temp10:    +31.0 C  (high =   +57 C, hyst =  -123 C)
```

Figure 6.22. Output of the sensors section on the 5050 / 6050 / 7050 models

```
bmc-i2c-4-00
Adapter: IPMI adapter
in1:       +1.10 V  (min =  +0.91 V, max =  +1.50 V)
in2:       +0.00 V  (min =  +0.91 V, max =  +1.50 V)
in3:       +1.18 V  (min =  +0.95 V, max =  +1.28 V)
in4:       +1.81 V  (min =  +1.66 V, max =  +1.90 V)
in5:       +0.00 V  (min =  +1.66 V, max =  +1.90 V)
in6:       +1.24 V  (min =  +1.09 V, max =  +1.29 V)
in7:       +1.40 V  (min =  +1.25 V, max =  +1.48 V)
in8:       +1.51 V  (min =  +1.35 V, max =  +1.59 V)
in9:       +3.36 V  (min =  +3.06 V, max =  +3.52 V)
in10:      +4.92 V  (min =  +4.63 V, max =  +5.33 V)
fan1:     3000 RPM  (min = 1080 RPM)
fan2:     3240 RPM  (min = 1080 RPM)
fan3:     3120 RPM  (min = 1080 RPM)
fan4:     4920 RPM  (min = 1080 RPM)
fan5:     3000 RPM  (min = 1080 RPM)
fan6:     3000 RPM  (min = 1080 RPM)
```

```
fan7:     7800 RPM  (min =  540 RPM)
fan8:     6360 RPM  (min =  540 RPM)
fan9:     7560 RPM  (min =  540 RPM)
fan10:    6000 RPM  (min =  540 RPM)
temp1:     +50.0 C  (high =   +85 C, hyst =  -123 C)
temp2:    -128.0 C  (high =   +85 C, hyst =  -123 C)
temp3:     +40.0 C  (high =   +85 C, hyst =  -123 C)
temp4:     +39.0 C  (high =   +85 C, hyst =  -123 C)
temp5:     +36.0 C  (high =   +55 C, hyst =  -123 C)
temp6:     +38.0 C  (high =   +80 C, hyst =  -123 C)
temp7:     +37.0 C  (high =   +65 C, hyst =  -123 C)
temp8:     +36.0 C  (high =   +65 C, hyst =  -123 C)
temp9:     +33.0 C  (high =   +57 C, hyst =  -123 C)
temp10:    +33.0 C  (high =   +57 C, hyst =  -123 C)
```

Figure 6.23. Output of the sensors section on the 255 models

```
coretemp-isa-0000
Adapter: ISA adapter
Core 0:      +43.0 C  (high = +86.0 C, crit = +100.0 C)

nct7904-i2c-0-2d
Adapter: SMBus I801 adapter at f000
VSEN1:       +1.05 V  (min =  +0.00 V, max =  +4.09 V)
VSEN2:       +2.01 V  (min =  +0.00 V, max =  +4.09 V)
VSEN3:       +2.01 V  (min =  +0.00 V, max =  +4.09 V)
VSEN4:       +2.01 V  (min =  +0.00 V, max =  +4.09 V)
VSEN5:       +2.01 V  (min =  +0.00 V, max =  +4.09 V)
VSEN6:       +1.58 V  (min =  +0.00 V, max =  +4.09 V)
VSEN7:       +1.61 V  (min =  +0.00 V, max =  +4.09 V)
VSEN8:       +1.61 V  (min =  +0.00 V, max =  +4.09 V)
VSEN9:       +0.90 V  (min =  +0.00 V, max =  +4.09 V)
VSEN10:      +2.02 V  (min =  +0.00 V, max =  +4.09 V)
VSEN11:      +1.50 V  (min =  +0.00 V, max =  +4.09 V)
VSEN12:      +0.79 V  (min =  +0.00 V, max =  +4.09 V)
VSEN13:      +1.00 V  (min =  +0.00 V, max =  +4.09 V)
VSEN14:      +0.99 V  (min =  +0.00 V, max =  +4.09 V)
3VDD:        +3.31 V  (min =  +0.00 V, max = +12.28 V)
VBAT:        +3.28 V  (min =  +0.00 V, max = +12.28 V)
3VSB:        +3.32 V  (min =  +0.00 V, max = +12.28 V)
FAN1:       5192 RPM  (min =  164 RPM)
FAN2:          0 RPM  (min =  164 RPM)
FAN3:       5294 RPM  (min =  164 RPM)
FAN4:          0 RPM  (min =  164 RPM)
FAN5:          0 RPM  (min =  164 RPM)
FAN6:          0 RPM  (min =  164 RPM)
FAN7:          0 RPM  (min =  164 RPM)
FAN8:          0 RPM  (min =  164 RPM)
FAN9:          0 RPM  (min =  164 RPM)
FAN10:         0 RPM  (min =  164 RPM)
LTD:         +35.9 C  (high = +100.0 C, hyst = +95.0 C)  sensor = thermistor
DTS1:        +42.0 C  (high = +100.0 C, hyst = +95.0 C)  sensor = Intel PECI
DTS2:         +0.0 C  (high = +100.0 C, hyst = +95.0 C)  sensor = Intel PECI
DTS3:         +0.0 C  (high = +100.0 C, hyst = +95.0 C)  sensor = Intel PECI
DTS4:         +0.0 C  (high = +100.0 C, hyst = +95.0 C)  sensor = Intel PECI
```

Figure 6.24. Output of the sensors section on the 555 / 755 models

```
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/cpu0_vid: "1500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/detach_state: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_input: "3624
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_min: "2500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan1_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_input: "3619
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_min: "2500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan2_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_input: "3673
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_min: "2500
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan3_type: "2
```

```
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_input: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/fan4_type: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_input: "5026
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_max: "6641
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in0_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_input: "1192
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_max: "2988
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in1_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_input: "3321
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_max: "4383
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in2_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_input: "5016
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_max: "6641
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in3_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_input: "12188
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_max: "15938
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in4_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_input: "3309
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_max: "4383
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in5_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_input: "3221
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_max: "4383
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/in6_min: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/name: "dme1737
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1: "103
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_channels_zone: "7
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_point1_pwm: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_point2_pwm: "255
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_auto_pwm_min: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_enable: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_freq: "25000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm1_ramp_rate: "206
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2: "102
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_channels_zone: "7
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_point1_pwm: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_point2_pwm: "255
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_auto_pwm_min: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_enable: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_freq: "25000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm2_ramp_rate: "206
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3: "102
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_channels_zone: "7
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_point1_pwm: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_point2_pwm: "255
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_auto_pwm_min: "100
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_enable: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_freq: "25000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/pwm3_ramp_rate: "206
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_fault: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_input: "59000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_max: "127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_min: "-127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp1_offset: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_fault: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_input: "39000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_max: "127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_min: "-127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp2_offset: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_alarm: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_fault: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_input: "41000
```

```
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_max: "127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_min: "-127000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/temp3_offset: "0
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/vrm: "14
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_channels_temp: "1
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point1_temp: "55000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point1_temp_hyst: "51000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point2_temp: "75000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone1_auto_point3_temp: "80000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_channels_temp: "2
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point1_temp: "42000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point1_temp_hyst: "38000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point2_temp: "52000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone2_auto_point3_temp: "62000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_channels_temp: "4
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point1_temp: "47000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point1_temp_hyst: "43000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point2_temp: "57000
"/sys/devices/pci0000:00/0000:00:1f.3/i2c-0/0-002e/zone3_auto_point3_temp: "67000
```

For the CX5055 and CX7055 models, the output of the sensors is not available in the system dump yet up to RiOS version 8.5.

Figure 6.25. Output of the sensors section on the EX560 / EX760 models

```
bmc-i2c-1-00
Adapter: IPMI adapter
fan1:     4320 RPM  (min = 1800 RPM)
fan2:     4200 RPM  (min = 1800 RPM)
fan3:     3000 RPM  (min = 1800 RPM)
temp1:     -15.0 C  (high =  +127 C, hyst =  -127 C)
temp2:     +22.0 C  (high =   +47 C, hyst =  -127 C)
temp3:     +24.0 C  (high =   +95 C, hyst =  -127 C)
temp4:     +31.0 C  (high =    +0 C, hyst =  -127 C)
```

For the EX1160, EX1260 and EX1360 models, the output of the sensors is not available in the system dump yet up to RiOS version 8.5.

6.5.3. The PCI bus

The PCI bus shows the cards attached to the PCI bus, for example the various Ethernet NICs.

6.5.4. Hard disks

The hard disks are identified by mapping, hardware identification, RAID status, LED status and solid-state disk wear status.

6.5.4.1. Disks mapped to the chassis location

The chassis of a Steelhead appliance has several slots for the hard disks, which can be used fully or partly.

- The 1RU chassis of the 1050 and 2050 models can contain four hard disks, but for the 1050L and 1050M model there will be only one slot occupied and for the 1050H model there will only be two slots occupied. The 1050LR, 1050MR, 1050HR and 250 models have four disks.
- The chassis for the 5050 and 6050 models can contain 16 hard disks, but for the 5050L and 5050M models only eight hard disks are used while for the 5050H model only 12 hard disks are used.

Figure 6.26. Hard disk to chassis mapping for a 1050L and 1050M model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':
```

```
0:0:0:0 disk0 sda online
1:0:0:0 disk1 missing missing
2:0:0:0 disk2 missing missing
3:0:0:0 disk3 missing missing
6:0:0:0 flash0 sha online
```

Figure 6.27. Hard disk to chassis mapping for a 1050H model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

0:0:0:0 disk0 sda online
1:0:0:0 disk1 sdb online
2:0:0:0 disk2 missing missing
3:0:0:0 disk3 missing missing
6:0:0:0 flash0 sha online
```

Figure 6.28. Hard disk to chassis mapping for a 2050 model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

0:0:0:0 disk0 sda online
1:0:0:0 disk1 sdb online
2:0:0:0 disk2 sdc online
3:0:0:0 disk3 sdd online
6:0:0:0 flash0 sde online
```

Figure 6.29. Hard disk to chassis mapping for a 5050L and 5050M model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

1:0:0:0 flash0 sha online
0:0:42:0 disk0 sde online
0:0:43:0 disk1 sdf online
0:0:44:0 disk2 sdg online
0:0:36:0 disk3 sdb online
0:0:41:0 disk4 sdd online
0:0:38:0 disk5 sdc online
0:0:45:0 disk6 sdh online
0:0:35:0 disk7 sda online
unknown disk8 missing missing
unknown disk9 missing missing
unknown disk10 missing missing
unknown disk11 missing missing
unknown disk12 missing missing
unknown disk13 missing missing
unknown disk14 missing missing
unknown disk15 missing missing
```

Figure 6.30. Hard disk to chassis mapping for a 5050H model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

0:0:18:0 disk0 sda online
0:0:19:0 disk1 sdb online
0:0:20:0 disk2 sdc online
0:0:21:0 disk3 sdd online
0:0:22:0 disk4 sde online
0:0:23:0 disk5 sdf online
0:0:24:0 disk6 sdg online
0:0:25:0 disk7 sdh online
0:0:26:0 disk8 sdi online
0:0:27:0 disk9 sdj online
0:0:28:0 disk10 sdk online
0:0:29:0 disk11 sdl online
unknown disk12 missing missing
unknown disk13 missing missing
unknown disk14 missing missing
unknown disk15 missing missing
```

Figure 6.31. Hard disk to chassis mapping for a 6050 model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

1:0:0:0 flash0 sha online
0:0:18:0 disk0 sda online
0:0:19:0 disk1 sdb online
0:0:20:0 disk2 sdc online
0:0:21:0 disk3 sdd online
0:0:22:0 disk4 sde online
0:0:23:0 disk5 sdf online
0:0:31:0 disk6 sdn online
0:0:24:0 disk7 sdg online
0:0:25:0 disk8 sdh online
0:0:26:0 disk9 sdi online
0:0:27:0 disk10 sdj online
0:0:28:0 disk11 sdk online
0:0:29:0 disk12 sdl online
0:0:30:0 disk13 sdm online
0:0:32:0 disk14 sdo online
0:0:33:0 disk15 sdp online
```

Figure 6.32. Hard disk to chassis mapping for a 7050L model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

1:0:0:0 flash0 sha online
0:0:49:0 disk0 sdc online
0:0:61:0 disk1 sdg online
0:0:75:0 disk2 sdo online
0:0:74:0 disk3 sdn online
0:0:72:0 disk4 sdl online
0:0:51:0 disk5 sdd online
0:0:46:0 disk6 sda online
0:0:76:0 disk7 sdp online
0:0:73:0 disk8 sdm online
0:0:70:0 disk9 sdj online
0:0:52:0 disk10 sde online
0:0:47:0 disk11 sdb online
0:0:77:0 disk12 sdq online
0:0:71:0 disk13 sdk online
0:0:68:0 disk14 sdi online
0:0:54:0 disk15 sdf online
unknown disk16 missing missing
unknown disk17 missing missing
unknown disk18 missing missing
unknown disk19 missing missing
unknown disk20 missing missing
unknown disk21 missing missing
unknown disk22 missing missing
unknown disk23 missing missing
unknown disk24 missing missing
unknown disk25 missing missing
unknown disk26 missing missing
unknown disk27 missing missing
unknown disk28 missing missing
unknown disk29 missing missing
unknown disk30 missing missing
0:0:62:0 disk31 sdh online
```

Figure 6.33. Hard disk to chassis mapping for a 7050M model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':
```

```
0:0:49:0 disk0 sdd online
0:0:61:0 disk1 sdo online
0:0:75:0 disk2 sdac online
0:0:74:0 disk3 sdab online
0:0:72:0 disk4 sdz online
0:0:51:0 disk5 sdf online
0:0:46:0 disk6 sda online
0:0:76:0 disk7 sdad online
0:0:73:0 disk8 sdaa online
0:0:70:0 disk9 sdx online
0:0:52:0 disk10 sdg online
0:0:47:0 disk11 sdb online
0:0:77:0 disk12 sdae online
0:0:71:0 disk13 sdy online
0:0:68:0 disk14 sdv online
0:0:54:0 disk15 sdi online
0:0:48:0 disk16 sdc online
0:0:60:0 disk17 sdn online
0:0:69:0 disk18 sdw online
0:0:65:0 disk19 sds online
0:0:55:0 disk20 sdj online
0:0:50:0 disk21 sde online
0:0:59:0 disk22 sdm online
0:0:67:0 disk23 sdu online
0:0:64:0 disk24 sdr online
0:0:56:0 disk25 sdk online
0:0:53:0 disk26 sdh online
0:0:58:0 disk27 sdl online
0:0:66:0 disk28 sdt online
0:0:63:0 disk29 sdq online
unknown disk30 missing missing
0:0:62:0 disk31 sdp online
```

Figure 6.34. Hard disk to chassis mapping for a 1555L, 1555M and 1555H model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

0:0:0:0 disk0 sda online
1:0:0:0 disk1 sdb online
2:0:0:0 disk2 sdc online
3:0:0:0 disk3 sdd online
6:0:0:0 flash0 sha online
```

Figure 6.35. Hard disk to chassis mapping for a 5055M and 5055H model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

6:0:0:0 flash0 sda online
0:0:0:0 disk0 sdb online
0:0:1:0 disk1 sdc online
0:0:2:0 disk2 sdd online
0:0:3:0 disk3 sde online
0:0:4:0 disk4 sdf online
0:0:5:0 disk5 sdg online
0:0:6:0 disk6 sdh online
0:0:7:0 disk7 sdi online
0:0:8:0 disk8 sdj online
0:0:9:0 disk9 sdk online
unknown disk10 missing missing
unknown disk11 missing missing
unknown disk12 missing missing
unknown disk13 missing missing
unknown disk14 missing missing
unknown disk15 missing missing
unknown disk16 missing missing
unknown disk17 missing missing
unknown disk18 missing missing
unknown disk19 missing missing
unknown disk20 missing missing
unknown disk21 missing missing
unknown disk22 missing missing
unknown disk23 missing missing
```

Figure 6.36. Hard disk to chassis mapping for a 7055L model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

6:0:0:0 flash0 sdm online
0:0:0:0 disk0 sda online
0:0:1:0 disk1 sdb online
0:0:2:0 disk2 sdc online
0:0:3:0 disk3 sdd online
0:0:4:0 disk4 sde online
0:0:5:0 disk5 sdf online
0:0:6:0 disk6 sdg online
0:0:7:0 disk7 sdh online
0:0:8:0 disk8 sdi online
0:0:9:0 disk9 sdj online
0:0:10:0 disk10 sdk online
0:0:11:0 disk11 sdl online
unknown disk12 missing missing
unknown disk13 missing missing
unknown disk14 missing missing
unknown disk15 missing missing
unknown disk16 missing missing
unknown disk17 missing missing
unknown disk18 missing missing
unknown disk19 missing missing
unknown disk20 missing missing
unknown disk21 missing missing
unknown disk22 missing missing
unknown disk23 missing missing
```

Figure 6.37. Hard disk to chassis mapping for a 7055M model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':

6:0:0:0 flash0 sda online
0:0:0:0 disk0 sdb online
0:0:1:0 disk1 sdc online
0:0:2:0 disk2 sdd online
0:0:3:0 disk3 sde online
0:0:4:0 disk4 sdf online
0:0:5:0 disk5 sdg online
0:0:6:0 disk6 sdh online
0:0:7:0 disk7 sdi online
0:0:8:0 disk8 sdj online
0:0:9:0 disk9 sdk online
0:0:10:0 disk10 sdl online
0:0:11:0 disk11 sdm online
0:0:15:0 disk12 sdq online
0:0:16:0 disk13 sdr online
0:0:12:0 disk14 sdn online
0:0:13:0 disk15 sdo online
0:0:14:0 disk16 sdp online
unknown disk17 missing missing
unknown disk18 missing missing
unknown disk19 missing missing
unknown disk20 missing missing
unknown disk21 missing missing
unknown disk22 missing missing
unknown disk23 missing missing
```

Figure 6.38. Hard disk to chassis mapping for a 7055H model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':
```

```
6:0:0:0 flash0 sda online
0:0:0:0 disk0 sdb online
0:0:1:0 disk1 sdc online
0:0:2:0 disk2 sdd online
0:0:3:0 disk3 sde online
0:0:4:0 disk4 sdf online
0:0:5:0 disk5 sdg online
0:0:6:0 disk6 sdh online
0:0:7:0 disk7 sdi online
0:0:8:0 disk8 sdj online
0:0:9:0 disk9 sdk online
0:0:10:0 disk10 sdl online
0:0:11:0 disk11 sdm online
0:0:16:0 disk12 sdr online
0:0:17:0 disk13 sds online
0:0:12:0 disk14 sdn online
0:0:13:0 disk15 sdo online
0:0:14:0 disk16 sdp online
0:0:15:0 disk17 sdq online
unknown disk18 missing missing
unknown disk19 missing missing
unknown disk20 missing missing
unknown disk21 missing missing
unknown disk22 missing missing
unknown disk23 missing missing
```

Figure 6.39. Hard disk to chassis mapping for a DX8000 model

```
==================================================
Output of '/opt/hal/bin/hwtool.py -q disk=map':
6:0:0:0 flash0 sdc online
0:0:0:0 disk0 sda online
0:0:1:0 disk1 sdb online
unknown disk2 missing missing
unknown disk3 missing missing
unknown disk4 missing missing
unknown disk5 missing missing
unknown disk6 missing missing
unknown disk7 missing missing
unknown disk8 missing missing
unknown disk9 missing missing
unknown disk10 missing missing
unknown disk11 missing missing
unknown disk12 missing missing
unknown disk13 missing missing
unknown disk14 missing missing
unknown disk15 missing missing
unknown disk16 missing missing
unknown disk17 missing missing
unknown disk18 missing missing
unknown disk19 missing missing
unknown disk20 missing missing
unknown disk21 missing missing
unknown disk22 missing missing
unknown disk23 missing missing
```

6.5.4.2. Disk information

Every disk will have their model type displayed:

Figure 6.40. Various hard disk fields on a 2050 model

```
Output of '/opt/hal/bin/raid/rrdm_tool.py -d /disk':

Num Drives : 4
0       WDC WD2500YD-01N        490234752       mgmt    online
1       WDC WD2500YD-01N        490234752       mgmt    online
2       WDC WD2500YD-01N        490234752       mgmt    online
3       WDC WD2502ABYS-0        490350672       mgmt    online
```

And the front LED status:

Figure 6.41. Front LED status fields for a 2050 model

```
Output of '/opt/hal/bin/raid/rrdm_tool.py --get-led-status':

Disk 0 Fault LED is off
Disk 1 Fault LED is off
Disk 2 Fault LED is off
Disk 3 Fault LED is off
```

And the various fields of the physical disks:

Figure 6.42. Physical characteristics of the disks in a 2050 model

```
Output of '/opt/hal/bin/raid/rrdm_tool.py --get-physical':

        ----------------------------------------
        Physical Drive 0
        ----------------------------------------
        Status: online          Type: disk
        Product: WDC WD2500YD-01N               Capacity: 233 GB
        Serial: WD-WCANKM053027        Firmware: 10.02E01
        Licensed: True

        ----------------------------------------
        Physical Drive 1
        ----------------------------------------
        Status: online          Type: disk
        Product: WDC WD2500YD-01N               Capacity: 233 GB
        Serial: WD-WCANKM083657        Firmware: 10.02E01
        Licensed: True

        ----------------------------------------
        Physical Drive 2
        ----------------------------------------
        Status: online          Type: disk
        Product: WDC WD2500YD-01N               Capacity: 233 GB
        Serial: WD-WCANKM029761        Firmware: 10.02E01
        Licensed: True

        ----------------------------------------
        Physical Drive 3
        ----------------------------------------
        Status: online          Type: disk
        Product: WDC WD2502ABYS-0               Capacity: 233 GB
        Serial: WD-WCAT11168840        Firmware: 02.03B02
        Licensed: True
```

6.5.5. RAID configuration

The Steelhead appliances either use a single disk, a RAID0 configuration or a RAID10 configuration.

6.5.5.1. RAID0 configuration

For the RAID0 configuration in the 1050H model, only the data store and the /proxy partition are in a RAID0 configuration. The /var partition is not in a RAID1 configuration, it is only stored on the first hard disk:

Figure 6.43. RAID configuration for RAID0 on a 1050H model

```
Output of 'cat /proc/mdstat':

Personalities : [linear] [raid0] [raid10] [raid6] [raid5]
md1 : active linear sda6[0] sdb6[1]
      204812544 blocks 64k rounding

unused devices: <none>
```

The /var partition is on /dev/sda3, the /proxy partition is on /dev/md1.

Figure 6.44. Partition on a 1050H model

```
Filesystem         1024-blocks      Used Available Capacity Mounted on
/dev/sda3             16513448   4893416  10781092      32% /var
/dev/md1             201598176     32936 191324616       1% /proxy
```

6.5.5.2. RAID10 configuration

The capable 1050 models, the 2050, 5050, 6050 and CX1555 use RAID10 to provide redundancy. The 7050, CX5055, CX7055 and DX8000 models use RAID1 for redundancy on their hard disks, not for the solid state disks.

Figure 6.45. RAID configuration for RAID10 on a 2050 model

```
Output of 'cat /proc/mdstat':
Personalities : [linear] [raid0] [raid10] [raid6] [raid5]
md3 : active raid10 sda6[0] sdd6[3] sdc6[2] sdb6[1]
      73930880 blocks 64K chunks 2 near-copies [4/4] [UUUU]

md0 : active raid10 sda5[0] sdd5[3] sdc5[2] sdb5[1]
      390636288 blocks 64K chunks 2 near-copies [4/4] [UUUU]

md2 : active raid10 sda2[0] sdd2[3] sdc2[2] sdb2[1]
      6297344 blocks 64K chunks 2 near-copies [4/4] [UUUU]

md1 : active raid10 sda3[0] sdd3[3] sdc3[2] sdb3[1]
      16787712 blocks 64K chunks 2 near-copies [4/4] [UUUU]

unused devices: <none>
```

The */var* partition is on /dev/md1 and the */proxy* partition is on /dev/md3:

Figure 6.46. Partition on a 2050 model

```
Filesystem         1024-blocks      Used Available Capacity Mounted on
/dev/md1              16523904    611780  15072740       4% /var
/dev/md3              72768928  25591492  43480892      38% /proxy
```

6.5.6. IPMI

IPMI (Intelligent Platform Management Interface) is a standard for monitoring system hardware, like power supplies, fans, temperatures, watchdogs, voltage levels etc.

6.5.6.1. IPMI SEL records

In the *hardwareinfo.txt* file it keeps track of the events of various hardware pieces on the xx50 series models:

Figure 6.47. Various IPMI event records

```
SEL Record ID          : 0001
 Record Type           : 02
 Timestamp             : 02/02/2009 09:42:51
 Generator ID          : 0020
 EvM Revision          : 04
 Sensor Type           : Power Unit
 Sensor Number         : 84
 Event Type            : Generic Discrete
 Event Direction       : Assertion Event
 Event Data            : 01ffff
 Description           : Redundancy Lost

SEL Record ID          : 014b
```

```
 Record Type          : 02
 Timestamp            : 03/06/2009 14:43:10
 Generator ID         : 0020
 EvM Revision         : 04
 Sensor Type          : Temperature
 Sensor Number        : 7a
 Event Type           : Threshold
 Event Direction      : Assertion Event
 Event Data           : 593939
 Description          : Upper Critical going high

SEL Record ID         : 0157
 Record Type          : 02
 Timestamp            : 04/06/2011 15:56:55
 Generator ID         : 0020
 EvM Revision         : 04
 Sensor Type          : Watchdog 2
 Sensor Number        : 81
 Event Type           : Sensor-specific Discrete
 Event Direction      : Assertion Event
 Event Data           : c104ff
 Description          : Hard reset
```

The first record is an alert that one of the power supplies in the Steelhead appliance are disconnected or have failed. The second record is a temperature alarm and the third one was that the hardware watchdog has timed out and the machine has been reset.

Note that the time on the SEL records do not always match the system time. This issue is resolved in RiOS version 7.0 and higher.

The following sensor types are known:

- Power Supply or Power Unit: A power supply related event.
- FAN: A fan related event.
- Temperature: A temperature sensor related event.
- Watchdog 2: The hardware watchdog related event.
- NMI: The software interrupt related event.
- Event Logging Disabled: All SEL records have been erased. This is part of the manufacturing phase.
- Voltage CMOS Battery: The CMOS Battery is failing.

6.5.6.2. IPMI SDR list

IPMI also tracks the status of various sensors, which partly overlays with some of the output of the sensors section:

Figure 6.48. Output of the command "ipmitool sdr list" on the 1050 / 2050 models

```
Output of ipmitool sdr list:
CPU0 Vcore       | 1.19 Volts        | ok
CPU1 Vcore       | disabled          | ns
FSB Vtt          | 1.13 Volts        | ok
5V               | 5.02 Volts        | ok
1.5V             | 1.47 Volts        | ok
Vdimm            | 1.81 Volts        | ok
12V              | 11.81 Volts       | ok
3.3Vsb           | 3.34 Volts        | ok
5Vsb             | 4.85 Volts        | ok
CPU0 Below Tmax  | 45 degrees C      | ok
CPU1 Below Tmax  | disabled          | ns
SB Temp          | 37 degrees C      | ok
```

```
MEM Temp         | 40 degrees C      | ok
FAN1             | 8760 RPM          | ok
FAN3             | 9360 RPM          | ok
FAN5             | 7440 RPM          | ok
FAN6             | 9960 RPM          | ok
FAN7             | 7320 RPM          | ok
FAN8             | 9840 RPM          | ok
FAN9             | 9120 RPM          | ok
FAN10            | 0 RPM             | ok
FAN2             | 0 RPM             | ok
FAN4             | 0 RPM             | ok
LAN Temp         | 39 degrees C      | ok
MCH Temp         | 38 degrees C      | ok
HDD BP Temp1     | 33 degrees C      | ok
HDD BP Temp2     | 29 degrees C      | ok
PSU Temp         | 36 degrees C      | ok
SYS Temp         | 29 degrees C      | ok
PS0 Temp         | 32 degrees C      | ok
PS1 Temp         | disabled          | ns
PS0 FAN1         | 8280 RPM          | ok
PS0 FAN2         | 6540 RPM          | ok
PS1 FAN1         | disabled          | ns
PS1 FAN2         | disabled          | ns
PS0              | 0x01              | ok
PS1              | Not Readable      | ns
PWR_UNIT_RDN     | 0x02              | ok
```

If the PS FAN is 0 RPM while the PS Temp is non-zero, that means that the power cable is removed from the power supply.

Figure 6.49. PS0 is connected, PS1 is not connected

```
PS0 Temp         | 32 degrees C      | ok
PS1 Temp         | 38 degrees C      | ok
PS0 FAN1         | 8400 RPM          | ok
PS0 FAN2         | 6600 RPM          | ok
PS1 FAN1         | 0 RPM             | ok
PS1 FAN2         | 0 RPM             | ok
PS0              | 0x01              | ok
PS1              | 0x09              | ok
```

6.5.7. Riverbed approved hardware

Riverbed appliances only work with Riverbed approved hardware, like hard disks, memory and network cards. If unlicensed hardware is detected the Steelhead appliance will raise an alarm about it.

For example for a 2050L model, the licensing details will show up in the system dump as:

Figure 6.50. Licensed hardware in a 2050L model

```
Output of /opt/hal/bin/hwtool.py -q disk-license:
disk0 Licensed
disk1 Licensed
disk2 Licensed
disk3 Licensed
flash0 Licensed

Output of /opt/hal/bin/hwtool.py -q nic-license:
primary Licensed
aux Licensed
lan0_0 Licensed
lan0_1 Licensed
lan1_0 Licensed
lan1_1 Licensed
wan0_0 Licensed
wan0_1 Licensed
wan1_0 Licensed
wan1_1 Licensed

Output of /opt/hal/bin/hwtool.py -q memory:
Total Licensed Memory: 8 GB
Total Unlicensed Memory: 0 GB
DIMM0: 2048 Licensed
DIMM1: 2048 Licensed
DIMM2: 2048 Licensed
DIMM3: 2048 Licensed
DIMM4: No Module Installed
DIMM5: No Module Installed
DIMM6: No Module Installed
DIMM7: No Module Installed
Number of DIMMs: 8

Output of /opt/hal/bin/hwtool.py -q cpu:
Manufacturer: GenuineIntel
Number of cores: 4
Speed: 1995 MHz
```

6.5.8. The NICs and other PCI slots

The list of installed NICs and other PCI cards can be found in the *hwtool.py* section:

Figure 6.51. The NICs and other PCI cards

```
Output of '/opt/hal/bin/hwtool.py -q cli':

Hardware revision:
Mainboard:  Platform 3UABA Motherboard, 400-00300-10
Slot 0: .......... 4 Port Copper GigE Network Bypass Module, Integrated
Slot 1: .......... 2 Port LR Fiber 10 GigE PCI-E Network Bypass and Redirect Card, 410-003 \
    01-01
Slot 2: .......... 2 Port LR Fiber 10 GigE PCI-E Network Bypass and Redirect Card, 410-003 \
    01-01
Slot 5: .......... SDR Accelerator Card, 410-00303
```

There are several kinds of cards available:

- Copper-based bypass cards. They can be integrated on the motherboard or via extra PCI cards:

Figure 6.52. Copper-based bypass cards

```
2 Port Copper FastEthernet Network Bypass Module, Integrated
2 Port Copper GigE Network Bypass Card, CMP-00028
2 Port Copper GigE Network Bypass Module, Integrated
4 Port Copper GigE Network Bypass Module, Integrated
4 Port Copper GigE PCI-E 8 Lane, Low Profile, Network Bypass Card, 410-00103
4 Port Copper GigE PCI-E Ethernet Card, 410-GEOPS-02
4 Port Copper GigE PCI-E Network Bypass Card, 410-00044-01
4 Port Copper GigE PCI-E Network Bypass Card, 410-00047-01
4 Port Copper GigE PCI-X Network Bypass Card, CMP-00074
6 Port Copper GigE PCI-X Network Bypass Card, 150-00011
```

- Fiber-based bypass cards. They can be LR, SR, SX or LX.

Figure 6.53. Fiber-based bypass cards

```
2 Port Fiber (LR, SR) SFP+ 10 GigE PCI-E Server Adaptor Card, 410-00049-01
2 Port LR Fiber 10 GigE PCI-E gen-2 Network Bypass and Redirect Card, 410-00301-02
2 Port LR Fiber 10 GigE PCI-E Network Bypass and Redirect Card, 410-00301-01
2 Port LX Fiber GigE PCI-E Network Bypass Card, 410-00105
2 Port SR Fiber 10 GigE PCI-E gen-2 Network Bypass and Redirect Card, 410-00302-02
2 Port SR Fiber 10 GigE PCI-E Network Bypass and Redirect Card, 410-00302-01
2 Port SX Fiber GigE PCI-E Network Bypass Card, 410-00101
4 Port LX Fiber GigE PCI-E 8 Lane, Network Bypass Card, 410-00106
4 Port LX fiber GigE PCI-E Network Bypass Card, 410-00046-01
4 Port SX Fiber GigE PCI-E, half length, Network Bypass Card, 410-00102
4 Port SX fiber GigE PCI-E Network Bypass Card, 410-00045-01
4 Port SX Fiber GigE PCI-X Network Bypass Card, 150-00010
```

- Virtual bypass cards.

Figure 6.54. Virtual bypass cards

```
2 Port E1000 Pair Module, VM-E1000E
2 Port Virtual Inpath Pair Module, VM-X
```

- RAID controllers, for the xx20 series Steelhead appliances.

Figure 6.55. RAID controllers

```
LSI Logic / Symbios Logic MegaRAID SATA 300-8X RAID Controller, CMP-00127
```

- Video cards

Figure 6.56. Video cards

```
MSI VGA card, 410-00052-01
```

- SDR Accelerator cards, for the offload of LZ compression into hardware.

Figure 6.57. SDR Accelerator cards

```
SDR Accelerator Card 10X, 410-00055
SDR Accelerator Card, 410-00303
```

6.6. System Information

The file *sysinfo.txt* contains the system information of the Steelhead appliance. It contains the operating system information, hard disk utilization, ifconfig output, ECC counter details, system and in-path routing tables, the output of the arp command, the process list, the directory listing of /var/opt and /var/log, IP/TCP/UDP/ICMP networking statistics and NIC interface statistics.

6.6.1. Operating system and RiOS versions

The first section contains the hostname, serial number of the device, the RiOS version on the device, the original software version and the date and time and uptime of the device:

Figure 6.58. System Information for a 550M model

```
System information:

Hostname: CSH
Serial Number: J47NG00012345
Model Number: 550M
Version:  rbt_sh 6.5.2 #113_20 2011-10-04 15:39:37 i386 root@moscow:svn://svn/mgmt/branche \
    s/canary_113_fix_7_sedgman_branch
Manufactured version: 5.0.8
Date:     2011-12-16 17:42:23
Uptime:   70d 4h 4m 21s

Service Uptime [DD-HH:MM:SS]: 24-04:10:37
```

This device is running the 64-bit version of RiOS version 6.5.2. The i386 shows it is a 32-bit version, if it shows x86_64 then it is the 64-bit version.

The *Uptime* is the uptime of the Steelhead appliance, the *Service Uptime* is the uptime of the optimization service.

The next section contains the build details of the *sport* binary of this RiOS version. For RiOS EX, this can be used to determine the RiOS version used in this RiOS EX software:

Figure 6.59. System information for an EX1160L model

```
System information:

Hostname: CSH
Serial Number: F84YU00012345
Model Number: EX1160L
Disk Layout: vsp_granite
Version:  rbt_ex 1.0.5 #556_10 2013-05-07 17:11:29 x86_64 root@montreux:svn://svn/build/br \
    anches/malta-ex_556_fix_branch
Manufactured version: 1.0.1a
Date:     2013-12-18 16:28:47
Uptime:   12d 17h 54m 5s

Service Uptime [DD-HH:MM:SS]: 12-17:49:31

==================================================
Output of '/opt/rbt/bin/sport -v':

Version: rbt_sh 7.0.6 (malta/fix/556_10)
Options: configure --disable-debug --enable-optimize --enable-epoll --enable-nfs --with-na \
    t=DEVNBT --enable-linker-map --with-tmslib=/work/prebuilt-modules/built-mgmt/branches/ \
    malta_556_fix_branch/125493/centos-4.2/x86_64/rbt_sh-7.0.6/tmslib --with-vmwaretools=/ \
    mnt/builds/prebuilt-modules/built-vmware-tools/branches/malta_556_fix_branch/4752/cent \
    os-4.2/x86_64 --with-sysincl=/work/prebuilt-modules/built-kernel/branches/malta_556_fi \
    x_branch/16634/2.6.9-34.EL/centos-4.2/x86_64//smp/smp
Builder: root@montreux (Linux CentOS release 4.8 (Final))
Kernel : 2.6.9-34.EL-rbt-16251SMP
Time   : Tue May  7 16:24:05 PDT 2013
Input  : /work/flamebox/sport-build-19763/sport
Output : /work/flamebox/sport-build-19763/sport/build
MD5    : b576d0f26bea7fe622721e3adb2c4a88
```

After RiOS EX 3.0 the output of this command only shows the RiOS EX software version, not the corresponding RiOS software version.

The next table contains the full list of RiOS EX software and matching RiOS version. An up-to-date list can be found in KB article S20423.

Table 6.1. RiOS EX software and matching RiOS version

RiOS EX software	Corresponding RiOS version
1.0.1	7.0.2
1.0.2	7.0.3
1.0.4	7.0.5
1.0.5	7.0.6
2.0.0	8.0.0
2.0.1	8.0.1
2.0.2	8.0.1a
2.1.0	8.0.2
2.1.1	8.0.2
2.1.2	8.0.2
2.5.0	8.0.2d
2.5.1	8.0.2d
2.5.2	8.0.4
3.0.0	8.5.0

6.6.2. Hard disk utilization

The hard disk utilization part shows the usage of the various partitions on the Steelhead appliance. The only ones interesting are the */var* partition which stores the various log files and data files, the */config* partition which stores the systems configuration and the */proxy* partition which contains PFS and RSP data.

The data store partition is not visible here since its format is not known by the *df* utility.

Figure 6.60. Hard disk utilization overview

```
Output of 'df -Pka':

Filesystem         1024-blocks      Used Available Capacity Mounted on
rootfs                  516008    292116    197608      60% /
/proc                        0         0         0       -  /proc
none                   1815084       232   1814852       1% /dev
/dev/root               516008    292116    197608      60% /
none                   1815084       232   1814852       1% /dev
tmpfs                  1815084      7260   1807824       1% /lib/modules
/proc                        0         0         0       -  /proc
/proc/bus/usb                0         0         0       -  /proc/bus/usb
/sys                         0         0         0       -  /sys
none                         0         0         0       -  /dev/pts
/dev/sdb2               128716     11254    110816      10% /boot
/dev/sdb1               253611      4246    236271       2% /bootmgr
/dev/sdb7               418806     49904    347277      13% /config
/dev/sda3             16513448   2769100  12905408      18% /var
none                   1815084         0   1815084       0% /dev/shm
none                     16384         4     16380       1% /tmp
/dev/disk0p6          56765020  39510668  14370736      74% /proxy
encfs                   418806     49904    347277      13% /var/opt/rbt/decrypted
```

The */boot* mount point is sometimes on the second partition (*/dev/sdb2* on a 550 model) and sometimes on the third partition (*/dev/sdb3* on a 550 model), depending on which boot partition the Steelhead appliance is booted from. This is located on the USB Flash disk.

The */config* mount point is the partition with the configuration of the Steelhead appliance. It is located on the USB Flash disk.

The */var* mount point is the partition which is writable by the operating system. It contains:

- The log files are in */var/log*.
- RiOS images are uploaded to */var/opt/tms/images*.
- System dumps are created into */var/opt/tms/sysdumps*.
- Process dumps are created in */var/opt/tms/snapshots*.
- The Secure Vault is in */var/opt/rbt/decrypted*, which contains the SSL certificates and private keys and the encrypted data store password. It is not a physical file system but an encrypted file system which lies on */var/opt/rbt/encrypted*.

The */proxy* mount point is the partition which contains the Proxy File System data (PFS) or the Riverbed Services Platform data (RSP).

6.6.3. Ifconfig output

The ifconfig output shows some high level statistics of various network interfaces. The most important fields are the administrative status *UP*, the IP address *inet addr*, the hardware MAC address *HWaddr* and some general counters like *RX packets* and *TX packets*.

Figure 6.61. Ifconfig output

```
Output of 'ifconfig -a':

inpath0_0 Link encap:Ethernet  HWaddr 00:0E:B6:12:34:57
          inet addr:10.0.1.6  Bcast:10.0.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5703948 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5649152 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:1403364965 (1.3 GiB)  TX bytes:1171662453 (1.0 GiB)

lan0_0    Link encap:Ethernet  HWaddr 00:0E:B6:12:34:57
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4643691 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2781142 errors:2 dropped:0 overruns:0 carrier:2
          collisions:0 txqueuelen:100
          RX bytes:1040279574 (992.0 MiB)  TX bytes:601245915 (573.3 MiB)
          Memory:fdce0000-fdd00000

primary   Link encap:Ethernet  HWaddr 00:0E:B6:12:34:56
          inet addr:10.0.1.5  Bcast:10.0.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:38967 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1874049 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:3564489 (3.3 MiB)  TX bytes:472288871 (450.4 MiB)
          Memory:fd9e0000-fda00000

wan0_0    Link encap:Ethernet  HWaddr 00:0E:B6:12:34:58
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1060250 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2867967 errors:2 dropped:0 overruns:0 carrier:1
          collisions:0 txqueuelen:100
          RX bytes:466055536 (444.4 MiB)  TX bytes:627854299 (598.7 MiB)
          Memory:fdc80000-fdca0000
```

The following information can be seen in this overview:

- The virtual interface named *inpath0_0* has the IP address 10.0.1.6, the physical lan0_0 and wan0_0 interfaces do not have an IP address.
- The MAC address of the inpath0_0 interface is the MAC address of the lan0_0 interface.
- The MAC address of the lan0_0 interface is one lower than the MAC address of the wan0_0 interface.

- The administrative status of all interfaces is *UP*. In virtual in-path deployment the administrative status of the lan0_0 does not contain the string *UP*.
- The counters on the RX packets and TX packets are cumulative and should have zero errors. These counters can be zeroed out with the command `clear interfaces all`. In the example there have been two carrier errors on the lan0_0 interface, which could be because of loss of the Ethernet link with the connected device.
- The interfaces named *rios_wan0* and *rios_wan0* are used by the RSP system.
- The interface named *prihw* is the physical interface on which the primary interface is mapped.
- The interface named *rbtpipe* is used by the Steelhead Cloud Accelerator functionality.

6.6.4. Memory ECC counters

The memory used in the Steelhead appliances has Error Correcting Code capabilities, which means that if it finds a bit-error it will try to correct it: It will be marked as a *Correctable Error* if successful and as an *Uncorrectable Error* if unsuccessful.

Figure 6.62. ECC counter output

```
Output of 'show_ecc_status':

/sys/devices/system/edac/mc/mc0/csrow0/ce_count:0
/sys/devices/system/edac/mc/mc0/csrow0/ch0_ce_count:0
/sys/devices/system/edac/mc/mc0/csrow0/ue_count:0
/sys/devices/system/edac/mc/mc0/csrow3/ce_count:0
/sys/devices/system/edac/mc/mc0/csrow3/ch0_ce_count:0
/sys/devices/system/edac/mc/mc0/csrow3/ue_count:0
/sys/devices/system/edac/mc/mc0/csrow0/ch0_dimm_label:DIMM0
/sys/devices/system/edac/mc/mc0/csrow3/ch0_dimm_label:DIMM0
/sys/devices/system/edac/mc/mc0/csrow0/size_mb:1024
/sys/devices/system/edac/mc/mc0/csrow3/size_mb:1024
```

The output shows ce_count (the number of Correctable Errors), ue_count (the number of Uncorrectable Errors), the name of the memory stick (dimm_label) and the size of the memory stick (size_mb).

6.6.5. System and in-path routing tables

The Steelhead appliance has multiple routing tables: One for the primary and auxiliary interface and one for every in-path interface.

6.6.5.1. The system routing tables

The system routing table is used for management traffic send from the Steelhead appliance.

Figure 6.63. System routing table output

```
Output of 'route -n':

Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.1.0        0.0.0.0         255.255.255.0   U     0      0        0 primary
0.0.0.0         10.0.1.9        0.0.0.0         UG    1      0        0 primary

Output of 'route -Cn':

Kernel IP routing cache
Source       Destination     Gateway         Flags Metric Ref    Use Iface
10.0.1.7     192.168.1.98    10.0.1.9        0      0        2 inpath0_0
10.0.1.7     192.168.1.52    10.0.1.9        0      0        2 inpath0_0
10.0.1.7     192.168.1.175   10.0.1.9        0      0        2 inpath0_0
10.0.1.7     192.168.1.39    10.0.1.9        0      0        2 inpath0_0
[..]
```

6.6.5.2. The in-path routing tables

The in-path routing tables are the routing tables specifically for the in-path interfaces. For simple networks where there is only one IP subnet behind the Steelhead appliance, most of the time it will only contain the default gateway. For Steelhead appliances which use the simplified routing table it only needs to contain the default gateway.

Figure 6.64. In-path routing table output

```
Output of 'ip route show table proxytable0_0':

10.0.1.0/24 dev inpath0_0  scope link
default via 10.0.r.9 dev inpath0_0 onlink
```

6.6.6. The output of the arp command

The output of the ARP table contains the IP addresses of the devices on the IP subnets connected to both the primary and auxiliary interface and the in-path interfaces:

Figure 6.65. ARP table overview

```
Output of 'arp -na':

? (10.0.1.1) at 00:1B:0C:B0:34:38 [ether] on primary
? (10.0.1.9) at 00:0D:B9:17:28:DE [ether] on inpath0_0
? (10.0.1.1) at 00:1B:0C:B0:34:38 [ether] on inpath0_0
? (10.0.1.7) at <incomplete> on inpath0_0
```

The only MAC address known on the primary interface is the MAC address of the client with which we connect to the CLI of the Steelhead appliance. On the in-path interface there is the MAC address of the default gateway and the MAC address of the clients which have optimized TCP connections. The Steelhead appliance has tried to find out the MAC address of the host with IP address 10.0.1.7 but didn't get answer on its ARP request.

6.6.7. Output of netstat

In RiOS version 8.0 and below, the netstat section shows the open sockets on the Steelhead appliance and the state of the socket.

Figure 6.66. Output of the netstat command

```
Output of 'netstat -a --numeric-hosts':

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address    Foreign Address    State
tcp        0      0 10.0.1.6:7810    0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:8005   0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:904      0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:8009     0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:8333     0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:7821   0.0.0.0:*          LISTEN
tcp        0      0 10.0.1.6:http    0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:8307   0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:8308     0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:8086   0.0.0.0:*          LISTEN
tcp        0      0 10.0.1.6:ssh     0.0.0.0:*          LISTEN
tcp        0      0 10.0.1.6:7800    0.0.0.0:*          LISTEN
tcp        0      0 10.0.1.6:7801    0.0.0.0:*          LISTEN
tcp        0      0 10.0.1.6:https   0.0.0.0:*          LISTEN
tcp        0      0 0.0.0.0:8222     0.0.0.0:*          LISTEN
tcp        0      0 10.0.1.6:7801    192.168.1.6:19472  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:55588  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:55591  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:55596  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:58625  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:55597  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:55598  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:55599  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:39427  ESTABLISHED
tcp        0      0 10.0.1.6:7801    192.168.1.6:25651  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:62215  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:59687  ESTABLISHED
tcp        0      0 10.0.1.6:7800    192.168.1.6:57636  ESTABLISHED
tcp        0      0 127.0.0.1:33387  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33388  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33403  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33402  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33401  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33406  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33405  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33404  127.0.0.1:8086     CLOSE_WAIT
tcp        0      0 127.0.0.1:33392  127.0.0.1:8086     TIME_WAIT
tcp        0      0 127.0.0.1:33411  127.0.0.1:8086     FIN_WAIT2
```

The states here are:

- *LISTEN*, when the socket is opened by a process waiting for a new TCP session to be setup to it.
- *ESTABLISHED*, when a TCP session has been setup towards the service.
- *CLOSE_WAIT*, where the socket is still open on this computer despite having received a TCP FIN from the remote computer.
- *TIME_WAIT*, when a TCP session has been torn down properly and is now lingering to catch any leftover packets.
- *FIN_WAIT2*, when the socket has been closed but the remote computer has not send the FIN yet.

Normally the Recv-Q and Send-Q should be zero or close to zero. If the values in there are non-zero then it might be an indication of a slow client.

6.6.8. Output of ss

In RiOS version 8.5 and higher, the output of the tool *ss*, short for *socket statistics*, is included in the file *sysinfo.txt* in favour of the *netstat* command.

Figure 6.67. Output of the command "ss"

```
==================================================
Output of 'ss -meanoiA inet,unix':

Netid  State      Recv-Q Send-Q     Local Address:Port       Peer Address:Port
[...]
tcp    LISTEN     0      128      ::ffff:10.0.1.6:7801                 :::*      ino:56032 \
    08 sk:ffff8800703c01c0
         mem:(r0,w0,f0,t0)
tcp    LISTEN     0      128                  :::443                   :::*      ino:32080 \
     sk:ffff8800635faa80
         mem:(r0,w0,f0,t0)
[...]
tcp    ESTAB      0      0        ::ffff:10.0.1.6:7800    ::ffff:10.92.31.243:38880  timer \
    :(keepalive,4min58sec,0) ino:40312268 sk:ffff880065d8ca40
         mem:(r0,w0,f0,t0) ts sack reno wscale:8,2 rto:573 rtt:191/71.75 ato:40 cwnd:4 sen \
    d 242.6Kbps rcv_space:31856
```

After the header, the first socket listens on a specific IP address for connections on TCP port 7801, while the second socket listens on all IP addresses on TCP port 443.

The third socket shows an established connection with the following features and characteristics:

- *ts*: TCP Time stamps are used.
- *sack*: Selective ACKing can be used.

- *ecn*: Explicit congestion notification is used.
- *ecnseen*: At least one ECT packets was seen.
- *fastopen*: The SYN packet contained data.
- *reno*: TCP Reno will be used for network congestion diagnostics.
- *wscale:8,2*: TCP Window scaling is used with a factor 8 on the outgoing packets and factor 2 on the incoming packets.
- *ato:40*: ???
- *mss:1434*: The maximum segment size is 1434 bytes.
- *cwnd:4*: The congestion window is 4.
- *ssthresh:?*: The slow-start threshold.
- *send 242.6Kbps*: Calculated send-speed, based on sender congestion window, sender maximum segment size and round trip time.
- *unacked:2920*: Number of bytes not yet acknowledged.
- *retrans:1460*: Number of bytes retransmitted for this socket.
- *lost:1460*: Number of bytes lost for this socket.
- *sacked:4*: Number of frames retransmitted via SACK.
- *fackets:12*: Number of frames retransmitted because of a fast ACK.
- *reordering:123*: Number of frames which did not arrive in the right order.
- *rcv_rtt:1.3*: The receiver round trip time ???
- *rcv_space:31856*: The socket receive space, for incoming packets.
- *rto:573*: The TCP Retransmission timeout is 573ms.
- *rtt:191/71.75*: The round trip time ???
- *cwnd:4*: The congestion window is 4.
- *ssthresh:???*: The slow-start threshold.
- *qack:???*: ???
- *bidir:???*: ???

6.6.9. The process list

There are two different process lists available in the system dump. The first one is in the file *top_output.txt*, which is a snapshot of the process list made by the "top" program. The second one is the output of the "ps" program in the file *sysinfo.txt*.

6.6.9.1. Output of the top program

Top is a Unix program which gives a continuous updated overview of the CPU load, memory usage, process statistics and the processes.

Figure 6.68. Output of the top program

```
top - 17:47:03 up  1:53,  0 users,  load average: 0.58, 0.54, 0.48
Tasks:  96 total,   4 running,  92 sleeping,   0 stopped,   0 zombie
Cpu(s):  7.6% us,  3.5% sy,  0.0% ni, 76.9% id, 11.1% wa,  0.1% hi,  0.9% si
Mem:   2054092k total,  2037524k used,    16568k free,     3172k buffers
Swap:  2097136k total,      248k used,  2096888k free,  1557400k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 6693 admin     12  -3 1351m 1.3g 1.1g R   48 66.9   4:34.58 sport
 6601 admin     15   0 70056  34m 2764 S    4  1.7   2:30.85 statsd
 4466 admin     15   0  180m  36m 8940 S    1  1.8   0:53.26 mgmtd
```

The first part is split in five lines:

- The system time, uptime, number of users and the 1, 5 and 15 minute average load. The load average is the percentage of time when there are processes running or ready to run.
- The number of processes in total, the ones currently running, the ones waiting for data to run or for a timer to expire, the number of stopped processes and the number of zombie processes. The last two should be zero.
- The CPU overview shows the CPU utilization, averaged out from the number of cores in the system. There are seven states: User-land (non-kernel processes), system (kernel processes), niced user-land, idle, waiting for I/O, hardware interrupt handling and software interrupt handling.
- The real memory numbers: The total amount of memory, the amount used, the amount free and the amount used in various network and disk buffers.
- The swap memory numbers: The total amount of swap memory, the amount used, the amount free and the amount which is currently not used but can used for swap without having to reinitialize it.

6.6.9.2. Output of ps

This section shows the process list output as seen on Unix machines. It contains the process ID, the parent process ID, the amount of CPU it used, the memory utilization, the start time of the process and the process name.

Figure 6.69. The process lists.

```
Output of 'ps -Aww -o user,pid,ppid,pcpu,pri,nice,vsize,rss,majflt,tty,stat,wchan,start,bs \
    dtime,command':

USER       PID  PPID %CPU PRI  NI   VSZ  RSS MAJFLT TT       STAT WCHAN   STARTED   TIME C \
    OMMAND
admin        1     0  0.0  23   0  2016  608     14 ?        S    -        May 24   0:11 i \
    nit [3]
[...]
admin     5081     1  0.0  23   0  8724 5992      1 ?        Ss   -        May 24   0:04 / \
    opt/tms/bin/pm
admin     5082  5081  0.7  23   0 70124 63184     3 ?        Ss   -        May 24  64:57 / \
    opt/tms/bin/mgmtd
admin     7148  5081  0.0  23   0  6784 4320     21 ?        Ss   -        May 24   0:02 / \
    opt/tms/bin/snmpd -f -s -c /etc/snmpd.conf
admin     7149  5081  0.4  24   0 16728 9700      9 ?        Ss   -        May 24  38:39 / \
    opt/tms/bin/statsd
admin     7164  5081  0.0  23   0  6188 2780      5 ?        Ss   -        May 24   0:08 / \
    opt/tms/bin/cmcfc
admin     7177  5081  0.1  23   0  4972 2544      1 ?        Ss   -        May 24  11:19 / \
    opt/tms/bin/crld
admin     7186  5081  0.0  23   0  1540  600      1 ?        Ss   -        May 24   0:00 / \
    usr/sbin/crond -n
admin     7188  5081  0.0  23   0  5984 2936      9 ?        Ss   -        May 24   0:01 / \
    usr/sbin/httpd -D NO_DETACH -f /etc/opt/tms/output/httpd.conf
ntp       7193  5081  0.0  23   0  3760 1588      5 ?        Ss   -        May 24   0:01 / \
    usr/sbin/ntpd -n -u ntp -g
```

In this output you will see the earlier described processes running on the Steelhead appliance: pm, mgmtd, snmpd, statsd and so on.

6.6.10. The list of currently logged in users

This will show the users logged in and on which terminals they are logged in via:

Figure 6.70. Output of the "who" command

```
Output of 'who -a':

                        Sep 12 15:34               436 id=si    term=0 exit=0
           system boot  Sep 12 15:34
           run-level 3  Sep 12 15:34                   last=S
                        Sep 12 15:34              2797 id=l3    term=0 exit=0
LOGIN      tty1         Sep 12 15:34              8156 id=1
LOGIN      tty2         Sep 12 15:34              8157 id=2
LOGIN      ttyS0        Sep 14 15:49             24796 id=co
admin    + pts/0        Sep 14 15:46 00:02       16987 (57.200.1.39)
           pts/1        Sep 12 16:56             21448 id=ts/1  term=0 exit=0
LOGIN      ttyS0        Sep 14 13:06             22714 id=S0
```

6.6.11. The directory listing of /var/opt and /var/log

From a user-land perspective, there is only one partition writable on the Steelhead appliance and that is the */var* partition. It is used for storing the log files, uploading new RiOS images, statistics files, keeping state for processes and so on.

Figure 6.71. Contents of /var/log

```
Output of 'find /var/log -type f -ls':

985101  496 -rw-r--r--   1 admin    root       503664 May 23 23:50 /var/log/sa/sa23
985014  496 -rw-r--r--   1 admin    root       503664 May 24 23:50 /var/log/sa/sa24
985027  496 -rw-r--r--   1 admin    root       503664 May 22 23:50 /var/log/sa/sa22
984975  496 -rw-r--r--   1 admin    root       503664 May 25 23:50 /var/log/sa/sa25
984998 373836 -rw-r--r--   1 admin    root     382423665 May 30 17:46 /var/log/messages
985073    8 -rw-r--r--   1 admin    root         7020 May 30 17:46 /var/log/user_messages
985077    4 -rw-------   1 admin    root         1508 May  3 15:20 /var/log/web_access_log \
    .4.gz
985031 1608 -rw-rw-rw-   1 admin    root      1638508 May 29 00:00 /var/log/sdr.stats.2.gz
985051  268 -rw-r--r--   1 admin    root       270153 May 28 00:00 /var/log/messages.3.gz
985094    4 -rw-------   1 admin    root          101 May 29 00:00 /var/log/web_error_log. \
    2.gz
984999   16 -rw-------   1 admin    root        12392 May 30 17:00 /var/log/oom_profile.lo \
    g.3.gz
985089    4 -rw-------   1 admin    root          100 May 28 00:00 /var/log/web_error_log. \
    3.gz
985064    4 -rw-r--r--   1 admin    root          389 May 29 00:00 /var/log/user_messages. \
    2.gz
985084    4 -rw-------   1 admin    root         2629 May  5 00:00 /var/log/web_access_log \
    .2.gz
[...]
```

The files sa* contain the System Accounting Records, various log files named messages.*, log files for the web server named web*.

Figure 6.72. Contents of /var/opt

```
Output of 'find /var/opt ( -name .running -prune ) -o -type f -ls':

608197 104020 -rw-rw-rw-   1 admin    root     106404863 Jan 27  2010 /var/opt/tms/images/ \
    cmc.upgrade.tbz
607889 4336 -rw-------   1 admin    root      4426824 Apr 29  2009 /var/opt/tms/snapshots/ \
    CSH-webasd-20090409-090119.tar.gz
460290 5644 -rw-rw-rw-   1 admin    root      5764037 Apr  9  2009 /var/opt/tms/sysdumps/s \
    ysdump-CSH-20090409-085757.tgz
459922  528 -rw-rw-rw-   1 admin    root       532985 Apr  9  2009 /var/opt/tms/sysdumps/s \
    ysdump-CSH-20090409-090119.tgz
459990 27020 -rw-rw-rw-   1 admin    root     27634079 May 30 16:22 /var/opt/tms/sysdumps/ \
    sysdump-CSH-20110530-162201.tgz
```

Here you can see a RiOS image uploaded by the CMC, a process dump of the process *webasd* and three system dumps.

6.6.12. IP/TCP/UDP/ICMP networking statistics

This is the statistics output of the netstat command.

Figure 6.73. Output of the "netstat -s" command

```
Output of '/bin/netstat -s':

Ip:
    3313608 total packets received
    0 forwarded
    0 incoming packets discarded
    3312203 incoming packets delivered
    5123477 requests sent out
Icmp:
    4786 ICMP messages received
    2 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 56
        echo requests: 4730
    4786 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 56
        echo replies: 4730
Tcp:
    18875 active connections openings
    18098 passive connection openings
    4 failed connection attempts
    2118 connection resets received
    1506 connections established
    3301666 segments received
    3274784 segments send out
    2287 segments retransmited
    0 bad segments received.
    1536 resets sent
```

6.6.13. NIC interface settings and statistics.

This section shows more details on the Ethernet cards than the ifconfig output shows, for example the speeds the card is able to negotiate to, if it used auto-negotiation or not and the current negotiated speed.

6.6.13.1. Auto-negotiation and link speed

In this output, the NIC is configured for auto-negotiation. During the negotiation, it has advertised 10 Mbps, 100 Mbps and 1000 Mbps speeds and the negotiation came up with 1000 Mbps. Because of the gigabit speed, it was also able to negotiate the Ethernet pause feature, which will allow the NIC to alert the connected device to stop sending new Ethernet frames.

Figure 6.74. Ethtool section for an auto-negotiated Ethernet link

```
==================================================
Output of 'ethtool wan0_0':

Settings for wan0_0:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised auto-negotiation: Yes
        Speed: 1000Mb/s
        Duplex: Full
        MDI: mdi
        Port: Twisted Pair
        PHYAD: 1
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: d
        Wake-on: d
        Current message level: 0x00000001 (1)
        Link detected: yes

==================================================
```

In this output, the auto-negotiation is not enabled but its speed and duplex are set to 100 Mbps and Full Duplex:

Figure 6.75. Ethtool section for a fixed speed and duplex Ethernet link

```
==================================================
Output of 'ethtool lan0_0':

Settings for lan0_0:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  Not reported
        Advertised auto-negotiation: No
        Speed: 100Mb/s
        Duplex: Full
        MDI: mdi
        Port: Twisted Pair
        PHYAD: 1
        Transceiver: internal
        Auto-negotiation: off
        Supports Wake-on: d
        Wake-on: d
        Current message level: 0x00000007 (7)
        Link detected: yes

==================================================
```

In this example, the NIC didn't manage to negotiate an Ethernet link:

Figure 6.76. No Ethernet link negotiated

```
==================================================
Output of 'ethtool lan0_0':

Settings for lan0_0:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  Not reported
        Advertised auto-negotiation: No
        Speed: Unknown! (65535)
        Duplex: Unknown! (255)
        MDI: Unknown! (0)
        Port: Twisted Pair
        PHYAD: 1
        Transceiver: internal
        Auto-negotiation: off
        Supports Wake-on: d
        Wake-on: d
        Current message level: 0x00000007 (7)
        Link detected: no
```

```
==================================================
```

6.6.13.2. Ethernet flow control

The Ethernet flow control allows a receiver of Ethernet frames to indicate that the sender should pause sending temporary. It is possible that the flow control is allowed only in one direction.

Figure 6.77. Ethernet flow control

```
==================================================
Output of 'ethtool -a lan0_0':

Pause parameters for lan0_0:
Autonegotiate:  on
RX:             on
TX:             on

==================================================
```

6.6.13.3. Offloading features

Various offloading features like TCP checksum generation and TCP segmentation offloading.

Figure 6.78. TCP segmentation offloading

```
==================================================
Output of 'ethtool -a lan0_0':

Offload parameters for lan0_0:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: on
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: off
large-receive-offload: off
==================================================
```

6.7. Simplified Routing related files

The Simplified Routing tables keeps track of an IP address and the MAC address of the gateway via which the IP address is reachable. These tables are independent per in-path interface.

For RiOS version 6.0 and higher the files with the related information are *macmap_tables* and *er/*/pkttab_entries*.

6.7.1. Mapping IP addresses to MAC addresses

The *macmap_tables* file contains the mapping of IP address to MAC addresses:

Figure 6.79. The contents of the file macmap_tables

```
    relay         ip_addr          mac_addr  vlan ref_count
inpath0_0        10.0.1.1 d4:9a:20:c2:52:0e     0        0
inpath0_0        10.0.1.9 00:0d:b9:17:28:de     0        0
inpath0_0      172.16.3.1 00:0d:b9:17:28:de     0        0
inpath0_0     192.168.1.6 00:0d:b9:17:28:de     0        0
inpath0_0     192.168.1.1 00:0d:b9:17:28:de     0        0
[...]
```

This shows that on inpath0_0 the IP addresses in the 192.168.1.0/24 subnet can be found behind the device with the MAC address 00:0d:b9:17:28:de.

This data can be seen from the CLI with the command `show in-path macmap-tables`.

6.7.2. Mapping of MAC address to in-path interface LAN or WAN interfaces

The pkttab_entries file contains the mapping of the MAC addresses on the LAN or the WAN side of the in-path interface. The number in the path is the sequence number of the in-path interface.

Figure 6.80. The contents of the file er/0/pkttab_entries

```
ent/tot  ht[chain] p        addr           vlan   dev        type          hits  flaps p   \
     vlan_chg p   race
  1/  2    31[  0] 0  d4:9a:20:c2:52:0e       0   lan0_0    EAL_DEV_LAN    30    0     0   \
          0 0      0
  2/  2    31[  0] 0  00:26:88:1e:39:80       0   wan0_0    EAL_DEV_WAN    55    0     0   \
          0 0      0
```

This data cannot be seen from the CLI.

6.8. ATA controller SMART data

The file *smart.txt* contains the SMART data (Self-Monitoring, Analysis and Reporting Technology) of the ATA controller of the hard disks. This information is available for all hard disks except for the xx20 series 3RU models in which the hard disks are behind a hardware RAID controller.

The following fields are interesting from a possible failure point of view:

- Current Pending Sector Count: Number of sectors the hard disk controller thinks are unstable but not yet reallocated.
- Offline Uncorrectable: Number of sectors which have been attempted to reallocated but which failed. The data in there is lost.
- Raw Read Error Rate: The checksum of the data read from the platter was incorrect and error correction was required to restore the data.
- Seek Error Rate: The number of seek related issues, when the arm of the servo does not properly position the head on the right track.
- Reallocated Sector Count: The number of sectors which have been reallocated to a different sector on the disk.
- Reallocated Event Count: Number of times a sector has to be reallocated to a different location on the drive.

6.9. Asymmetric routing table

Note: See the chapter Asymmetric Routing in the *Operational Related Issues* section for the full explanation of the asymmetric routing feature.

The file *asymrouting_table* contains the current status of the asymmetric routing table:

Figure 6.81. Contents of the file "asymrouting_table"

```
10.20.53.121 10.21.1.201 SYN-rexmit 0 1344925735 1344925735
10.20.53.121 10.21.1.220 SYN-rexmit 4 1344925739 1344925739
10.8.6.20 10.8.32.54 bad-RST 54125 1344893467 1344900082
10.8.6.20 10.4.89.95 bad-RST 82927 1344922268 1344922277
10.8.6.20 10.12.64.20 bad-RST 54124 1344893466 1344922283
10.4.199.87 10.20.53.121 invalid-SYNACK 86170 1344925510 1344925510
10.20.53.121 10.21.1.212 SYN-rexmit 2 1344925737 1344925737
10.8.6.20 10.8.44.104 bad-RST 54124 1344893467 1344900082
10.20.53.121 10.21.1.200 SYN-rexmit 0 1344925735 1344925735
```

The format of the fields is:

- IP address client
- IP address server
- Asymmetric routing reason
- Timeout in seconds
- Date and time created, seconds since 1 January 1970
- Date and time last used, seconds since 1 January 1970

So for the line `10.8.6.20 10.4.89.95 bad-RST 82927 1344922268 1344922277`, the timeout was most likely 86400 seconds (one day), the time it was created was on 14 August 2012 at 05:31:08 and the last time a new TCP session got passed-through was nine seconds later on 05:31:17.

6.10. The Image History file

The file `image_history` contains the list of RiOS images which this Steelhead appliance has been booted into.

Figure 6.82. Contents of the file "image_history"

```
Firstboot to rbt_sh 5.5.9 #187_4 2010-10-14 18:48:05 x86_64 root@poznan:svn://svn/mgmt/bra \
    nches/guam_187_fix_branch on Thu Jun 30 21:51:32 GMT 2011
Upgraded to rbt_sh 6.1.3a #77_11 2011-03-13 20:11:47 x86_64 root@palermo0:svn://svn/mgmt/b \
    ranches/cook_77_fix_branch on Thu Jun 30 22:26:45 GMT 2011
Reverted to rbt_sh 5.5.9 #187_4 2010-10-14 18:48:05 x86_64 root@poznan:svn://svn/mgmt/bran \
    ches/guam_187_fix_branch on Tue Aug  2 12:42:39 GMT 2011
Reverted to rbt_sh 6.1.3a #77_11 2011-03-13 20:11:47 x86_64 root@palermo0:svn://svn/mgmt/b \
    ranches/cook_77_fix_branch on Thu Aug  4 17:38:40 GMT 2011
Upgraded to rbt_sh 6.5.1 #100_13 2011-05-24 14:19:19 x86_64 root@palermo0:svn://svn/mgmt/b \
    ranches/canary_100_fix_branch on Thu Aug  4 18:10:43 GMT 2011
```

This Steelhead appliance had RiOS version 5.5.9 installed when it came out of the box and got upgraded to RiOS version 6.1.3a half an hour later. Later it got downgraded to RiOS version 5.5.9 and then upgraded again to 6.1.3a and 6.5.1 at the same time.

6.11. The file "lsof"

The file `lsof.txt` contains the output of the command `lsof`, which is used to display the list of open file descriptors (files, pipes, sockets).

Figure 6.83. Output of the file "lsof.txt"

```
COMMAND     PID   USER   FD   TYPE   DEVICE     SIZE    NODE NAME
[...]
sshd       6198  admin  cwd    DIR     8,69     4096       2 /
sshd       6198  admin  rtd    DIR     8,69     4096       2 /
sshd       6198  admin  txt    REG     8,69   444313   33830 /usr/sbin/sshd
sshd       6198  admin  mem    REG     8,69   113224   39572 /lib64/ld-2.3.4.so
sshd       6198  admin  mem    REG     8,69    36144   39578 /lib64/libpam.so.0.77
sshd       6198  admin  mem    REG     8,69    17000   39624 /lib64/libdl-2.3.4.so
sshd       6198  admin  mem    REG     8,69  1565275    3019 /opt/rbt/lib/libcrypto.so.0.9 \
    .8
sshd       6198  admin  mem    REG     8,69    16280   39602 /lib64/libutil-2.3.4.so
sshd       6198  admin  mem    REG     8,69    77072   39560 /usr/lib64/libz.so.1.2.1.2
sshd       6198  admin  mem    REG     8,69   112176   39613 /lib64/libnsl-2.3.4.so
sshd       6198  admin  mem    REG     8,69    42672   39592 /lib64/libcrypt-2.3.4.so
sshd       6198  admin  mem    REG     8,69    90672   39579 /lib64/libresolv-2.3.4.so
sshd       6198  admin  mem    REG     8,69  1630336   39755 /lib64/tls/libc-2.3.4.so
sshd       6198  admin  mem    REG     8,69    63736   39586 /lib64/libaudit.so.0.0.0
sshd       6198  admin  mem    REG     8,69    58872   39582 /lib64/libnss_files-2.3.4.so
sshd       6198  admin  DEL    REG      0,6          2268647 /dev/zero
sshd       6198  admin  mem    REG     8,69    12888   39729 /lib64/security/pam_stack.so
sshd       6198  admin  mem    REG     8,69     8864   39721 /lib64/security/pam_nologin.s \
    o
sshd       6198  admin  mem    REG     8,69    20824   39706 /lib64/security/pam_limits.so
sshd       6198  admin  mem    REG     8,69    53104   39727 /lib64/security/pam_console.s \
    o
sshd       6198  admin  mem    REG     8,69   557176   39237 /usr/lib64/libglib-2.0.so.0.4 \
    00.7
sshd       6198  admin  mem    REG     8,69     4560   39731 /lib64/security/pam_deny.so
sshd       6198  admin  mem    REG     8,69    12624   39743 /lib64/security/pam_env.so
sshd       6198  admin  mem    REG     8,69    57662    2764 /opt/tms/lib/security/pam_rad \
    ius_auth.so
sshd       6198  admin  mem    REG     8,69   145059    2766 /opt/tms/lib/security/pam_uni \
    x.so
sshd       6198  admin  mem    REG     8,69    10692    2767 /opt/tms/lib/security/pam_fai \
    ldelay.so
sshd       6198  admin  mem    REG     8,69    24632   39609 /lib64/libnss_dns-2.3.4.so
sshd       6198  admin  DEL    REG      0,6          2269964 /dev/zero
sshd       6198  admin    0u   CHR      1,3             1102 /dev/null
sshd       6198  admin    1u   CHR      1,3             1102 /dev/null
sshd       6198  admin    2u   CHR      1,3             1102 /dev/null
sshd       6198  admin    3u  IPv4  2268635              TCP 10.0.1.5:ssh->10.0.1.1:40520  \
    (ESTABLISHED)
sshd       6198  admin    4r  FIFO      0,7          2269968 pipe
sshd       6198  admin    5w  FIFO      0,7          2269968 pipe
sshd       6198  admin    7w  FIFO      0,7          2269969 pipe
sshd       6198  admin    8r  FIFO      0,7          2269970 pipe
sshd       6198  admin   10r  FIFO      0,7          2269971 pipe
[...]
sshd      11011  admin    3u  IPv4    25118              TCP *:ssh (LISTEN)
[...]
```

In this example, the process *sshd* has the process ID *6198*, is running as the user *admin*, is the binary located in `/usr/sbin/sshd` has a couple of libraries loaded and has an ESTABLISHED TCP socket between 10.0.1.1:40520 and 10.0.1.5:22.

The process *sshd* with process ID 11011 has a TCP socket without an IP address and in the LISTEN mode, which means that it is waiting for new TCP sessions.

6.12. Memory dump of the process sport

The file *mem-dump* contains the dump of the meta data of the objects allocated in the memory pool of the optimization service.

6.12.1. The header

The header contains information about the memory available, the amount of memory used, the number of optimized TCP sessions and the state of the optimization service:

Figure 6.84. Header of the mem-dump file

```
AdmissionControl: Total System Memory = 7971, Current Usage = 4083 (3844 mmap, 236 non-hse \
    g), Slack Memory = 150, Connections = 522, State = 0

Peer Count: Online Peers = 13, Expired Peers = 0
```

When the current usage is of memory is above a certain value, the optimization service will go into memory based admission control. The values for this can be found in the value for the fields `/rbt/sport/admission/config/cutoff_mem` and `/rbt/sport/admission/config/enable_mem`:

Figure 6.85. Memory based admission based control values

```
[~/sysdump-SH] edwin@t43>grep /rbt/sport/admission/config/.*mem active-brief.txt
/rbt/sport/admission/config/cutoff_mem = 5200 (uint32)
/rbt/sport/admission/config/enable_mem = 5100 (uint32)
```

So in this case, the current usage is 4083 Mb, well below the cutoff value of 5200 Mb. Therefore there is no memory based admission control.

When the number of current connection usage is above a certain value, the optimization service will go into connection based admission control. The values for this can be found in the value for the fields `/rbt/sport/admission/config/cutoff_conn` and `/rbt/sport/admission/config/enable_conn`:

Figure 6.86. Connection based admission based control values

```
[~/sysdump-SH] edwin@t43>grep /rbt/sport/admission/config/.*conn active-brief.txt
/rbt/sport/admission/config/cutoff_conn = 2550 (uint32)
/rbt/sport/admission/config/enable_conn = 2500 (uint32)
```

So in the case, the current amount is 522, well below the cutoff value of 2550. Therefore there is no connection based admission control.

6.13. Out-of-memory profiling

The files *oom_profile.log** are populated every thirty seconds with the output of the commands `cat /proc/meminfo` and `ps`.

Figure 6.87. Contents of the file oom_profile.log

```
================================================================================
Fri Sep 14 15:40:28 CEST 2012
================================================================================

cat /proc/meminfo
MemTotal:      8162944 kB
MemFree:       2683204 kB
Buffers:         67880 kB
Cached:        4221328 kB
SwapCached:          0 kB
Active:        4292324 kB
Inactive:       944588 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:      8162944 kB
LowFree:       2683204 kB
SwapTotal:     6297336 kB
SwapFree:      6297336 kB
Dirty:           32504 kB
Writeback:           0 kB
Mapped:        4696692 kB
Slab:           178840 kB
Committed_AS:  4968392 kB
PageTables:      21556 kB
VmallocTotal: 536870911 kB
VmallocUsed:    280440 kB
VmallocChunk: 536588807 kB
HugePages_Total:     0
HugePages_Free:      0
Hugepagesize:     2048 kB

ps -Aww -o pid,ppid,pcpu,vsize,rss,majflt,tty,stat,wchan,command
  PID  PPID %CPU   VSZ  RSS MAJFLT TT       STAT WCHAN  COMMAND
    1     0  0.0  4768  628     14 ?        S    -      init [3]
    2     1  0.0     0    0      0 ?        S    migrat [migration/0]
    3     1  0.0     0    0      0 ?        SN   ksofti [ksoftirqd/0]
    4     1  0.0     0    0      0 ?        S    watchd [watchdog/0]
    5     1  0.0     0    0      0 ?        S    migrat [migration/1]
    6     1  0.0     0    0      0 ?        SN   ksofti [ksoftirqd/1]
    7     1  0.0     0    0      0 ?        S    watchd [watchdog/1]
    8     1  0.0     0    0      0 ?        S    migrat [migration/2]
    9     1  0.0     0    0      0 ?        SN   ksofti [ksoftirqd/2]
   10     1  0.0     0    0      0 ?        S    watchd [watchdog/2]
   11     1  0.0     0    0      0 ?        S    migrat [migration/3]
[...]
 8010     1  0.0 28716 7676      0 ?        Ss   -      /opt/tms/bin/pm
 8109     1  0.0  2396  320      0 ?        S<   -      /opt/tms/bin/csoftwatch
 8138     1  0.0  4188  588      3 ?        S<   wait   sh /sbin/softwatch.sh
 8156     1  0.0  2540  468      0 tty1     Ss+  -      /sbin/mingetty tty1
 8157     1  0.0  2540  468      0 tty2     Ss+  -      /sbin/mingetty tty2
10554     1  0.1 20696 3520      0 ?        Ss   fuse_d /usr/local/bin/encfs -S
/var/opt/rbt/encrypted/ /var/opt/rbt/decrypted
10907  8010  0.0  3640  660      0 ?        Ss   -      /usr/sbin/crond -n
10910  8010  0.0 17040 2008      7 ?        Ss   -      /usr/sbin/ntpd -n -u ntp
 -g
10912  8010  0.0 99480 28788    90 ?        Ssl  -      /opt/rbt/bin/rcud --logr
c /etc/rcud.logrc
11011  8010  0.0 15172 1876      3 ?        Ss   -      /usr/sbin/sshd -D
11012  8010  0.0 18536 8328      4 ?        S<Lsl -     /opt/tms/bin/wdt
11560  8010  0.0 38732 5176     33 ?        Ss   -      /usr/sbin/winbindd -F
17859 11560  0.0 38692 5128      0 ?        S    fcntl_ /usr/sbin/winbindd -F
19591  8010  4.2 4294084 4283876 0 ?        S<Lsl -     /opt/rbt/bin/sport /etc/opt/tms/ou \
    tput/configfile.xml --logrc /etc/sport.logrc --logfile /var/log/sport.log -t /etc/opt/ \
    tms/output/tds_config.xml
```

With running this every 30 seconds, memory usage related issues are tracked.

6.14. CIFS pre-population related data

The CIFS pre-population related configuration data is stored in the file `rcud.conf`. This data is in the XML format and contains a single <share> section per CIFS pre-population share:

Figure 6.88. The rcud.conf section for the share \\192.168.1.1\Test

```
  <share>
    <config       min_sync_freq_sec="60"
       sendbuf_size_byte="65536"
       recvbuf_size_byte="65536"
       get_share_size_freq_sec="10800"
       local_name="\\192.168.1.1\Test"
       server_name="192.168.1.1"
       server_path="\\192.168.1.1\Test"
       server_port="0"
       server_acct="mavetju\edwin"
       server_pwd="630434303e3cac08a322aa43346430394cc498f42be4e34d"
       mode="3"
       sync_freq_sec="300"
       sync_start_time="1299551270"
       full_sync_freq_sec="300"
       full_sync_start_time="1299551270"
       sharing="false"
       syncing="false"
       version_number="3"
       id="11"
       skip_init_copy="false"
/>
[...]
    <status       status="2"
       initial_copy_completion_time_sec="1299552948"
       initial_copy_completion_time_usec="8736"
       last_sync_status="true"
       last_sync_time_sec="1299553115"
       last_sync_time_usec="596340"
       last_full_sync_status="true"
       last_full_sync_time_sec="1299552948"
       last_full_sync_time_usec="8826"
       last_sync_begin_time_sec="30137660"
       last_sync_begin_time_usec="395507260"
       size_byte="0"
       acl_swapped="false"
/>
  </share>
```

The status of the CIFS pre-population can be found in the `pfs` directory. The names of the files are related to a share starts with that name, for example for the share with the name `\\192.168.1.1\Test`, the following files are generated:

Figure 6.89. Files for the share \\192.168.1.1\Test

```
-rw-r--r--   1 edwin  edwin      0 Mar  8  2011 \\192.168.1.1\Test.err_log
-rw-r--r--   1 edwin  edwin    290 Mar  8  2011 \\192.168.1.1\Test.initial-copy.status
-rw-r--r--   1 edwin  edwin   2153 Mar  8  2011 \\192.168.1.1\Test.last-sync.status
```

The synchronization status can be found in the files with the suffix `.initial-copy` and `last-sync.status`:

Figure 6.90. Contents of the files initial-copy.status and last-sync.status

```
Get initial copy started at Thu Sep  9 15:44:34 2010
Get initial copy completed at Thu Sep  9 15:44:43 2010
 Received 1 directories, 7 files, 0 deletions.  0 objects have error. (1101879 bytes were  \
    received in 7.330629 seconds at 1.202493 Mbps)

Full sync started at Thu Nov 25 15:45:48 2010
Full sync completed at Thu Nov 25 15:45:49 2010
 Sent 1 directories, 1 files, 0 deletions and 0 renames. 0 objects have error.
 (712321 bytes were sent in 3.213981 seconds)
===============================================
Full sync started at Thu Nov 25 16:51:50 2010
Full sync failed at Thu Nov 25 16:53:20 2010 connecting to server failed!
 Sent 0 directories, 0 files, 0 deletions and 0 renames. 0 objects have error.
 (0 bytes were sent in 0.000000 seconds)
```

6.15. RSP related data

The RSP related data in the system dump is stored in the `rsp/` directory. The most important information is related to the RSP slots.

6.15.1. RSP slots

The RSP slot related data is stored in the `rsp/slots` directory.

Figure 6.91. Files in the rsp/slots directory

```
[~/sysdumps/2050L] edwin@t43>ls -alR rsp/slots/
total 14
drwxr-xr-x  7 edwin  edwin   512 Sep 14 23:50 .
drwxr-xr-x  5 edwin  edwin  2048 Sep 14 23:50 ..
drwxr-xr-x  2 edwin  edwin   512 Sep 14 23:50 1
drwxr-xr-x  2 edwin  edwin   512 Sep 14 23:50 2
drwxr-xr-x  2 edwin  edwin   512 Sep 14 23:50 3
drwxr-xr-x  2 edwin  edwin   512 Sep 14 23:50 4
drwxr-xr-x  2 edwin  edwin   512 Sep 14 23:50 RiverbedSMC

rsp/slots/1:
total 4
drwxr-xr-x  2 edwin  edwin  512 Sep 14 23:50 .
drwxr-xr-x  7 edwin  edwin  512 Sep 14 23:50 ..

rsp/slots/2:
[...]

rsp/slots/RiverbedSMC:
total 18
drwxr-xr-x  2 edwin  edwin   512 Sep 14 23:50 .
drwxr-xr-x  7 edwin  edwin   512 Sep 14 23:50 ..
-rw-r--r--  1 edwin  edwin     5 Nov  9  2011 .enabled
-rw-r--r--  1 edwin  edwin     0 Aug 22  2011 .installed
-rw-r--r--  1 edwin  edwin     6 Apr 10 18:02 .priority
-rw-r--r--  1 edwin  edwin     0 Sep 29  2009 RiverbedSMC.vmsd
-rw-r--r--  1 edwin  edwin   266 Sep 29  2009 RiverbedSMC.vmxf
-rw-r--r--  1 edwin  edwin   698 Apr 10 18:02 rsp.conf
-rw-r--r--  1 edwin  edwin     0 Aug 22  2011 rsp.vmsd
-rw-r--r--  1 edwin  edwin  2895 Sep 13 02:09 rsp.vmx
-rw-r--r--  1 edwin  edwin   258 Aug 22  2011 rsp.vmxf
```

The file `.enabled` contains either "false" or "true", depending if an RSP slot is enabled. Of the file `rsp.vmx`, the only interesting field is *memsize* which defines how much memory is allocated for this RSP slot:

Figure 6.92. Memory allocation for the RSP slot RiverbedSMC

```
[~/sysdumps/2050L] edwin@t43>grep memsize rsp/slots/RiverbedSMC/rsp.vmx
memsize = "768"
```

The file `rsp.conf` contains the RSP Package Creator related configuration.

6.16. SAR statistics

The System Accounting Records contain statistics of the kernel running on the Steelhead appliance. The most basic ones are the CPU utilization, system load, NIC statistics and memory usage.

The Steelhead appliance keeps track of about a month of SAR data. The SAR data gets generated every day at 23:53 and during the generation of a system dump:

Figure 6.93. Overview of the SAR data files

```
-rw-r--r--  1 edwin  edwin  1281662 Sep 10 07:53 2012.09.09.23.53.sar
-rw-r--r--  1 edwin  edwin  1272770 Sep 11 07:53 2012.09.10.23.53.sar
-rw-r--r--  1 edwin  edwin  1281518 Sep 12 07:53 2012.09.11.23.53.sar
-rw-r--r--  1 edwin  edwin   872426 Sep 13 00:15 2012.09.12.16.15.sar
-rw-r--r--  1 edwin  edwin  1281519 Sep 13 07:53 2012.09.12.23.53.sar
```

You can use a visualization tool like kSar to graph the data. You can find it at http://sourceforge.net/projects/ksar/.

6.17. Active Directory Integration

For Active Directory integration, the data is stored in the `pfs/` directory. There are two important files: The joining of the Steelhead appliance to the domain and the communication towards the Domain Controllers.

Figure 6.94. Contents of the pfs directory

```
[~/sysdumps/CSH] edwin@t43>ls -al pfs
total 9836
drwxr-xr-x   2 edwin  edwin      512 Sep 18 19:47 .
drwxr-xr-x  19 edwin  edwin    10240 Sep 18 19:47 ..
-rw-r--r--   1 edwin  edwin   916073 Aug  4 22:51 log.wb-EXAMPLE
-rw-r--r--   1 edwin  edwin     1628 Aug 25 21:16 log.winbindd
-rw-r--r--   1 edwin  edwin        0 May 31  2010 log.winbindd-dc-connect
-rw-r--r--   1 edwin  edwin        0 Jul 21  2010 log.winbindd-locator
-rw-r--r--   1 edwin  edwin   310610 Sep  8  2010 net_ads.out
-rw-r--r--   1 edwin  edwin   175287 Aug 25 21:17 smbd.log
```

The file `net_ads.out` contains the logs for when the Steelhead appliance gets joined to the domain. The most important data it is at the end:

Figure 6.95. Contents of the net_ads.out

```
[2010/09/08 16:42:25,  1] libnet/libnet_join.c:libnet_Join(1920)
  libnet_Join:
      libnet_JoinCtx: struct libnet_JoinCtx
          out: struct libnet_JoinCtx
              account_name             : NULL
              netbios_domain_name      : 'EXAMPLE'
              dns_domain_name          : 'EXAMPLE.ORG'
              forest_name              : 'EXAMPLE.ORG'
              dn                       : 'CN=csh,CN=Computers,DC=EXAMPLE,DC=ORG'
              domain_sid               : *
                  domain_sid               : S-1-5-21-1298361332-912839128-1298391823
              modified_config          : 0x00 (0)
              error_string             : NULL
              domain_is_ad             : 0x01 (1)
              result                   : WERR_OK
[2010/09/08 16:42:25,  2] utils/net.c:main(849)
  return code = 0
Using short domain name -- EXAMPLE
Joined 'CSH' to realm 'EXAMPLE.ORG'
```

This shows that the Steelhead was successfully joined to the Active Directory domain.

The file `log.wb-EXAMPLE` contains the logs of the communication to the domain controller.

6.18. Process dumps

When a process on the Steelhead fails, the operating system will generate a core dump of it: A copy of all the memory of the process as it was in use at the moment it failed. The size of the core dump depends on the size of the process, for a core dump of the optimization service this can be hundreds of megabytes.

After the core dump is generated, the process *pm* will tell the process *mgmtd* to generate a system dump and merge the two dumps into a process dump. This process might take a while, during which the process does not get restarted.

It can be necessary in the lifetime of a case that Riverbed TAC requires a process dump of the optimization service. The way to generate one is to use the command `debug service clone dump start` or the command `pm process <PROCESS> send-signal SIGQUIT`.

The command `debug service clone dump start` will create a clone of the optimization service first and generates a process dump of the clone. This will cause the optimization and the normal traffic to continue.

Figure 6.96. Generating a process dump of the process sport by cloning it

```
SH # debug service clone dump start
[... some times passes ...]
SH # show files process-dump
SH-sport-20130102-123456.tar.gz
SH # file process-dump SH-sport-20130102-123456.tar.gz ftp://192.168.1.1/incoming/SH-sport \
    -20130102-123456.tar.gz
SH # show info
Current User:      admin

Status:            Healthy
Config:            working
Appliance Up Time: 12d 11h 20m 7s
Service Up Time:   12d 11h 18m 14s
Managed by CMC:    yes
Temperature (C):   44
```

The command `pm process <PROCESS> send-signal SIGQUIT` will terminate the optimization service and then generate a process dump of it. This will cause the optimized TCP sessions to be aborted and the optimization service to be restarted.

Figure 6.97. Generating a process dump of the process sport by restarting it

```
SH # pm process sport send-signal SIGQUIT
[... some times passes ...]
SH # show files process-dump
SH-sport-20130102-123456.tar.gz
SH # file process-dump SH-sport-20130102-123456.tar.gz ftp://192.168.1.1/incoming/SH-sport \
    -20130102-123456.tar.gz
SH # show info
Current User:      admin

Status:            Healthy
Config:            working
Appliance Up Time: 12d 11h 24m 23s
Service Up Time:   1m 31s
Managed by CMC:    yes
Temperature (C):   44
```

Chapter 7. Different Network Troubleshooting Scenarios

7.1. Index

In this chapter several failure scenarios and their troubleshooting approaches are described.

- Traffic is blocked when the Steelhead appliance goes in by-pass.
- Traffic is not optimized between two sites.
- Optimized TCP sessions which fail halfway.
- Slow networks.
- How to find the cause of connection-based admission control.

7.2. Traffic is blocked when the Steelhead appliance goes in by-pass.

Under normal operational circumstances when the Steelhead appliance is fully operational, the LAN and WAN interfaces each have an Ethernet links negotiated towards the devices they it connected to: From the LAN interface to the LAN switch and from the WAN interface to the WAN router.

When an in-path interface goes in by-pass, due to the termination of the optimization service or when the appliance gets turned off, it will connect the cables of the LAN switch and the WAN router directly to each other and these two device should negotiate an Ethernet link together.

Ethernet link negotiation failure could happen when:

- The cabling between the interfaces on the LAN switch and WAN router has become incompatible. Keep in mind that an in-path interface in by-pass mode acts like a crossover cable and that Fast Ethernet interfaces don't have the ability to detect the cable type. Also check the Cables section in the *Installation Related Issues* chapter.
- The configuration of the interfaces on the LAN switch and the WAN router has become incompatible. For example, one WAN interface is setup to 100 Mbps Full Duplex and the LAN interface is set to auto-negotiation on gigabit speeds only, then there will never an Ethernet link established.
- The in-path interface is by default configured for fail-to-wire but can be configured to be fail-to-block. Use the commands `show interfaces inpath0_0 configured` to determine the current state and the configuration command `no interface inpath0_0 fail-to-bypass enable` to go to fail-to-block.

Figure 7.1. In-path interface being configured for fail-to-block

```
SH (config) # show interfaces inpath0_0 configured
Interface inpath0_0 configuration
   Enabled:            no
   DHCP:               no
   Dynamic DNS DHCP:   no
   IP address:         10.0.1.6
   Netmask:            255.255.255.0
   MTU:                1500
   Failure mode:       Bypass
SH (config) # no interface inpath0_0 fail-to-bypass enable
SH (config) # show interfaces inpath0_0 configured
Interface inpath0_0 configuration
   Enabled:            no
   DHCP:               no
   Dynamic DNS DHCP:   no
   IP address:         10.0.1.6
   Netmask:            255.255.255.0
   MTU:                1500
   Failure mode:       Disconnect
```

7.3. Traffic is not optimized between two sites.

When traffic is not optimized between two sites, the following steps can be taken to determine the source of the problem:

- Check the health of the appliances.
- Setup a TCP session which is expected to be optimized.
- Determine if the client-side Steelhead appliance sees this TCP session.
- Determine if the server-side Steelhead appliance sees this TCP session.
- Determine the cause of the failure of setup of inner channel.

7.3.1. Health of the appliances

A Steelhead appliance will not optimize any new TCP sessions whilst the device is in Admission Control, like a TCP Connection Limit admission control or Memory based admission control. Use the command `show info` to confirm this and the command `show alarms triggered` to determine which alarms are triggered.

A Steelhead appliance will not optimize any new TCP sessions if the packet goes via an interface which is in bypass mode. Use tcpdump to confirm that the traffic from the client to the server goes via the Steelhead appliance.

A Steelhead appliance in a Connection Forwarding cluster will not optimize any new TCP sessions if the Connection Forwarding peer is unreachable and Allow Failure in the Connection Forwarding configuration is not enabled. Use the command `show in-path neighbour` to confirm this.

A Steelhead appliance will not optimize any new TCP sessions if the combination of the client IP address and server IP address is in the Asymmetric Routing table. Use the command `show in-path asym-route-tab` to and `in-path asym-route-tab flush` to deal with this.

Only once confirmed that both Steelhead appliances are in a healthy state, further troubleshooting can be started.

7.3.2. Setup a TCP session expected to be optimized

7.3.2.1. Obtain necessary data

First step is to determine the details of the TCP traffic that doesn't get optimized. The information required is the source-subnets where the clients are located (source-address) and the IP address of the server (dest-address) and the TCP port of the service (dest-port).

In this example the IP address of the client will be 10.0.1.1, the IP address of the server will be 192.168.1.1 and the TCP port is 543.

7.3.2.2. Obtain access to the client

The second step is to get access to a client. This would be easiest if there is access to a client via a remote access or remote desktop program:

- Access to Unix based clients can be achieved via SSH or Telnet. Once logged in on the client, use the Telnet command to setup a TCP session to the server.

- Access to Microsoft Windows based client can be achieved via the Microsoft Remote Desktop application or screen sharing applications programs like VNC or Goto Assist. Once access has been obtained, use the Shell command `telnet.exe` to setup a TCP session to the server.

  Note that Windows 7 does not install the telnet program by default, it can be installed via Control Panel -> Programs and Features -> Enable Windows Features.

- If no access can be obtained via real clients, consider accessing the local LAN switch, if it supports remote shells, or via an SSH session towards the local Steelhead appliance, if the primary interface of the Steelhead appliance is connected to the LAN switch. There use the telnet command to setup a TCP session to the server.

7.3.2.3. Use Telnet to setup a TCP session to the server

The syntax of the Telnet command is as follows:

Figure 7.2. Usage of the telnet command

```
$ telnet 192.168.1.1 543
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
```

The full usage of this command is described in the chapter Telnet section in the *The Command Line and Tools* chapter.

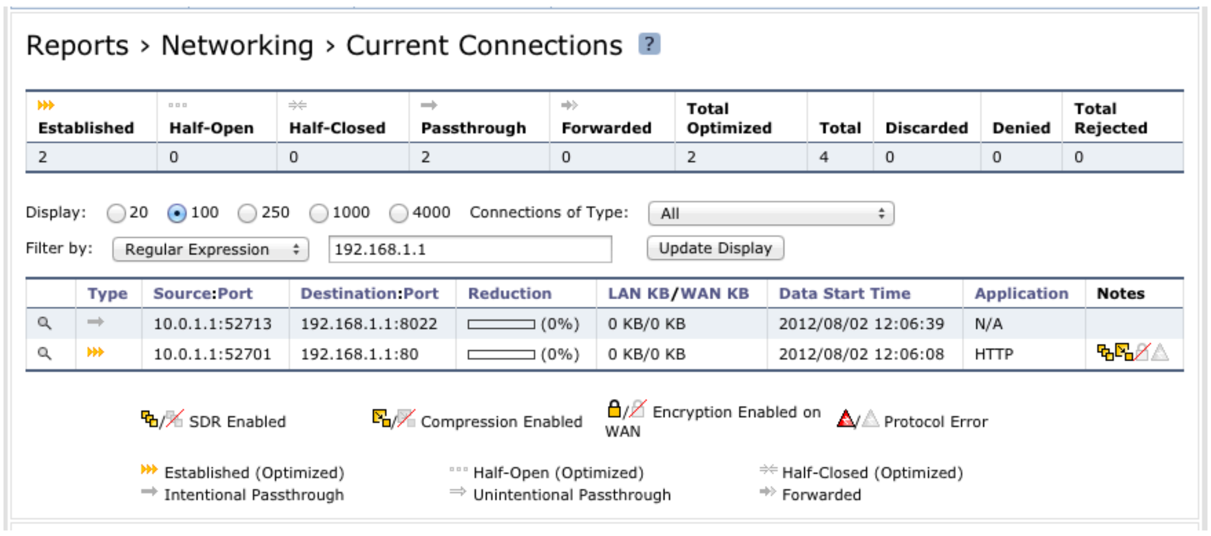
7.3.3. How the client-side Steelhead appliance sees this TCP session

The next step is to see if and how the client-side Steelhead appliance sees this TCP session.

7.3.3.1. More information on why the TCP session was not optimized

This can be found via the GUI under Reports -> Networking -> Current Connections. Under connection type, select *All* and under Filter enter the IP address of the client and search for the line with the IP addresses of the client and the server.

Figure 7.3. Reports -> Networking -> Current Connections



Reports › Networking › Current Connections ?

Established	Half-Open	Half-Closed	Passthrough	Forwarded	Total Optimized	Total	Discarded	Denied	Total Rejected
2	0	0	2	0	2	4	0	0	0

Display: 20 [✓ 100] 250 1000 4000 Connections of Type: All

Filter by: Regular Expression 192.168.1.1 Update Display

	Type	Source:Port	Destination:Port	Reduction	LAN KB/WAN KB	Data Start Time	Application	Notes
🔍	⇒	10.0.1.1:52713	192.168.1.1:8022	(0%)	0 KB/0 KB	2012/08/02 12:06:39	N/A	
🔍	⋙	10.0.1.1:52701	192.168.1.1:80	(0%)	0 KB/0 KB	2012/08/02 12:06:08	HTTP	

SDR Enabled · Compression Enabled · Encryption Enabled on WAN · Protocol Error

⋙ Established (Optimized) · Half-Open (Optimized) · Half-Closed (Optimized)
⇒ Intentional Passthrough · Unintentional Passthrough · Forwarded

On the CLI this can be done with the command `show connections all filter 10.0.1.1`.

Figure 7.4. Output of the "show connections all" command.

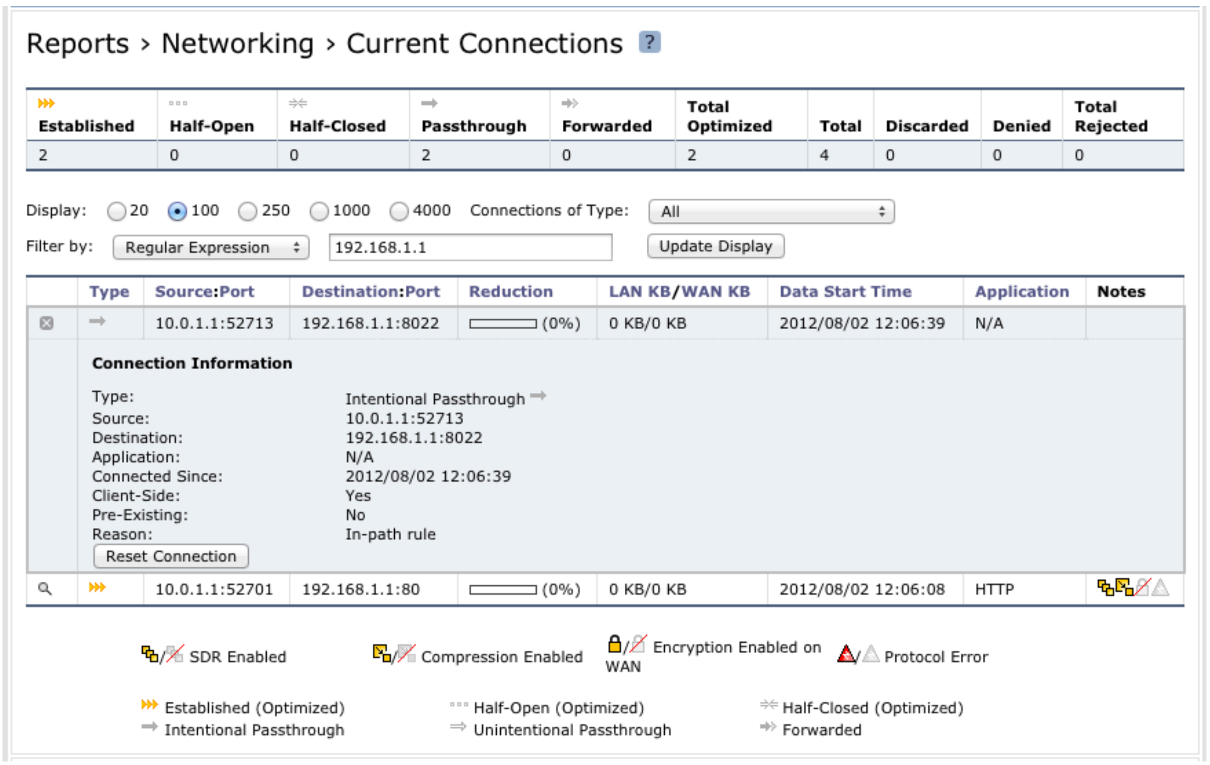
```
SH # show connections all
T  Source                    Destination          App    Rdn Since
--------------------------------------------------------------------------------
O  10.0.1.1         37780 192.168.1.1       543 TCP     0% 2012/07/29 01:54:35
PI 192.168.1.1      42342 10.0.1.1           22            2012/07/28 14:06:56 s
PU 10.0.1.1         51274 192.168.1.1       443            2012/07/29 01:43:22 c
PU 10.0.1.1         51490 192.168.1.1        80            2012/07/29 01:54:08 c
--------------------------------------------------------------------------------
Established Optimized (O):      1
Half-Opened Optimized (H):      0
Half-Closed Optimized (C):      0
Pass Through (P):               3
PI = Passthrough Intentional
PU = Passthrough Unintentional
```

If the client-side Steelhead appliance does not show this TCP session, then the TCP session followed a path between the client and the server which is not covered by this Steelhead appliance.

If the TCP session shows up as *192.168.1.1:543 10.0.1.1:1025*, then the first packet seen by the client-side Steelhead appliance is the TCP SYN-ACK, not the TCP SYN packet. In that case the path from the client to the server bypasses this Steelhead appliance and the path from the server to the client does go through this Steelhead appliance.

If the TCP session shows up as *10.0.1.1:1025 192.168.1.1:543*, then check the details of the TCP session. This can be done in the GUI by clicking on the magnifying glass in front of the TCP session.

Figure 7.5. TCP session details



Reports › Networking › Current Connections

Established	Half-Open	Half-Closed	Passthrough	Forwarded	Total Optimized	Total	Discarded	Denied	Total Rejected
2	0	0	2	0	2	4	0	0	0

Display: ○20 [● 100] ○250 ○1000 ○4000 Connections of Type: All

Filter by: Regular Expression 192.168.1.1 Update Display

	Type	Source:Port	Destination:Port	Reduction	LAN KB/WAN KB	Data Start Time	Application	Notes
⊠	➡	10.0.1.1:52713	192.168.1.1:8022	(0%)	0 KB/0 KB	2012/08/02 12:06:39	N/A	
🔍	⏩	10.0.1.1:52701	192.168.1.1:80	(0%)	0 KB/0 KB	2012/08/02 12:06:08	HTTP	

Connection Information

Type:	Intentional Passthrough ➡
Source:	10.0.1.1:52713
Destination:	192.168.1.1:8022
Application:	N/A
Connected Since:	2012/08/02 12:06:39
Client-Side:	Yes
Pre-Existing:	No
Reason:	In-path rule

Reset Connection

SDR Enabled; Compression Enabled; Encryption Enabled on WAN; Protocol Error

⏩ Established (Optimized); ··· Half-Open (Optimized); Half-Closed (Optimized); ➡ Intentional Passthrough; ⇨ Unintentional Passthrough; Forwarded

On the CLI this can be done with the command `show connections all filter 10.0.1.1 full`.

Figure 7.6. Output of the "show connections all full" command.

```
T  Source                  Destination           App    Rdn Since
--------------------------------------------------------------------------------
O  10.0.1.1        37780 192.168.1.1      543 TCP     0% 2012/07/29 01:54:35
                    Peer: 192.168.1.6    7800    Flags: cfe,sdr,lz,te=0,pe=0
             Outer Local: 10.0.1.6       7801    Inner: 19421
            Outer Remote: 10.0.1.6      37780   WAN/LAN: 0KB / 0KB
PI 192.168.1.1     42342 10.0.1.1          22            2012/07/28 14:06:56 s
PU 10.0.1.1        51274 192.168.1.1      443            2012/07/29 01:43:22 c
PU 10.0.1.1        51490 192.168.1.1       80            2012/07/29 01:54:08 c
```

7.3.3.2. Different kinds of pass-through TCP sessions

There are two kinds of pass-through TCP sessions:

- Intentional pass-through sessions, caused by in-path rules.
- Un-intentional pass-through session, caused by the failure to find or connect to another Steelhead appliance.

If the TCP session is intentional passed-through, then checking the in-path rules is a good first step.

7.3.3.3. Reasons for pass-through

The pass-through reasons contain the next clue on why the client-side Steelhead appliance couldn't setup an optimized TCP session:

- If the pass-through reason is *SYN on the WAN side*, then the naked SYN packet for this TCP session showed up at the WAN interface instead of at the LAN interface: The LAN and WAN cables are swapped.
- If the pass-through reason is *No more Steelheads*, then the SYN+ packet leaving the Steelhead appliance did not go through the server-side Steelhead appliance. Use the traceroute command to determine the path the IP packets take.

- If the pass-through reason is *Asymmetric Routing*, then the server-side Steelhead appliance saw the SYN+ packet and send the SYN/ACK+, but the client-side Steelhead appliance did not see the SYN/ACK+ but the result of the client receiving the SYN/ACK+.

  Check the Asymmetric Routing table to find out the reason for the pass-through of the TCP session.

- If the pass-through reason is *Connection is currently paused*, then the optimization service is in admission control and not optimizing new TCP sessions.

- There is no space in the TCP Options field for the auto-discovery probe. The maximum length of the TCP header is 64 bytes with 20 bytes used by fixed fields, leaving only 44 bytes for TCP Options. The standard TCP options like MSS size, Selective ACK, TCP Timestamps) leave enough space for the auto-discovery probe, but other extensions like auto-discovery probes from other vendors doesn't leave enough spare for the auto-discovery probe to be added.

  Figure 7.7. No room for new TCP options for the auto-discovery phase

  ```
  CSH [intercept.WARN] no room for option: 122.102.98.114:49809 -> 172.21.3.101:443
  ```

A full list of pass-through reasons can be found in the Steelhead Management Console and Users Guide and KB article S15377.

7.3.3.4. Does the auto-discovery probe get attached?

Next step is to confirm that the client-side Steelhead appliance is putting the auto-discovery probe on to the TCP SYN packet. For this two SSH sessions to the CLI are needed, one to trace traffic on the LAN side and one to trace traffic on the WAN side. If the TCP SYN packet comes in on a specific LAN interface, it will be sent out via the corresponding WAN interface.

7.3.3.4.1. Client-side Steelhead appliance does attach an auto-discovery probe.

In this example, the client-side Steelhead appliance does properly attach a TCP auto-discovery probe on the new TCP SYN packet.

Figure 7.8. Running tcpdump on the LAN side

```
CSH # tcpdump -ni lan0_0 '(host 192.168.1.1 and host 10.0.1.1 and port 543) or (vlan and ( \
    host 192.168.1.1 and host 10.0.1.1 and port 543))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
09:51:50.122355 IP 10.0.1.1.57198 > 192.168.1.1.543: Flags [S], seq 3205639252, win 65535, \
     options [mss 1460,nop,wscale 3,nop,nop,TS val 346474876 ecr 0,sackOK,eol], length 0
09:51:50.256198 IP 192.168.1.1.543 > 10.0.1.1.57198: Flags [S.], seq 3950051338, ack 32056 \
    39253, win 5792, options [mss 1460,sackOK,TS val 1644000 ecr 346474876,nop,wscale 2],  \
    length 0
09:51:50.259818 IP 10.0.1.1.57198 > 192.168.1.1.543: Flags [.], seq 1, ack 1, win 65535, o \
    ptions [nop,nop,TS val 346475011 ecr 1644000], length 0
```

Figure 7.9. Running tcpdump on the WAN side

```
CSH # tcpdump -ni wan0_0 '(host 192.168.1.1 and host 10.0.1.1 and port 543) or (vlan and ( \
    host 192.168.1.1 and host 10.0.1.1 and port 543))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
09:51:50.122430 IP 10.0.1.1.57198 > 192.168.1.1.543: Flags [S], seq 3205639252, win 65535, \
     options [mss 1460,nop,wscale 3,nop,nop,TS val 346474876 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
09:51:50.256076 IP 192.168.1.1.543 > 10.0.1.1.57198: Flags [S.], seq 20020520, ack 3205639 \
    253, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 346474876 ecr 0,sackOK,n \
    op,nop,rvbd-probe EAD 0c01,nop,nop,nop,eol], length 0
09:51:50.256109 IP 192.168.1.1.543 > 10.0.1.1.57198: Flags [S.], seq 20020520, ack 3205639 \
    253, win 65535, options [mss 1460,nop,wscale 3,TS val 346474876 ecr 0,sackOK,rvbd-prob \
    e AD CSH:10.0.1.6 SSH:192.168.1.6:7800 11110a000106c0a801061e78,rvbd-probe EAD 0e3d,no \
    p,eol], length 0
```

The TCP SYN packets with the *rvbd-probe AD* TCP option, or the SYN+, is seen here and shows that the client-side Steelhead has processed the new TCP session properly.

7.3.3.4.2. Client-side Steelhead appliance does not attach an auto-discovery probe.

In this example, the client-side Steelhead appliance does not attach a TCP auto-discovery probe on the new TCP SYN packet.

Figure 7.10. Running tcpdump on the LAN side

```
CSH # tcpdump -ni lan0_0 '(host 192.168.1.1 and host 10.0.1.1 and port 543) or (vlan and ( \
    host 192.168.1.1 and host 10.0.1.1 and port 543))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
09:55:09.968491 IP 10.0.1.1.57269 > 192.168.1.1.543: Flags [S], seq 587559488, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 346673686 ecr 0,sackOK,eol], length 0
09:55:10.102059 IP 192.168.1.1.543 > 10.0.1.1.57269: Flags [S.], seq 214939753, ack 587559 \
    489, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 2495570782 ecr 346673686] \
    , length 0
09:55:10.105859 IP 10.0.1.1.57269 > 192.168.1.1.543: Flags [.], seq 1, ack 1, win 65535, o \
    ptions [nop,nop,TS val 346673823 ecr 2495570782], length 0
```

Figure 7.11. Running tcpdump on the WAN side

```
CSH # tcpdump -ni wan0_0 '(host 192.168.1.1 and host 10.0.1.1 and port 543) or (vlan and ( \
    host 192.168.1.1 and host 10.0.1.1 and port 543))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
09:55:09.968576 IP 10.0.1.1.57269 > 192.168.1.1.543: Flags [S], seq 587559488, win 65535,  \
    options [mss 1460,nop,wscale 3,nop,nop,TS val 346673686 ecr 0,sackOK,eol], length 0
09:55:10.101988 IP 192.168.1.1.543 > 10.0.1.1.57269: Flags [S.], seq 214939753, ack 587559 \
    489, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 2495570782 ecr 346673686] \
    , length 0
09:55:10.105920 IP 10.0.1.1.57269 > 192.168.1.1.543: Flags [.], seq 1, ack 1, win 65535, o \
    ptions [nop,nop,TS val 346673823 ecr 2495570782], length 0
```

If there is no auto-discovery probe seen on the outgoing SYN packet, then the optimization service might not be enabled on that particular in-path interface, the appliance might be in admission control or there is asymmetric routing happening.

7.3.4. How the server-side Steelhead appliance sees this TCP session

The next step is to see what the server-side Steelhead appliance sees about this TCP session.

7.3.4.1. More information on why the TCP session was not optimized

The first step would be to check in the GUI on the Current Connections page for this TCP session, just like on the client-side Steelhead appliance.

If the pass-through reason is *SYN on the WAN side*, then the auto-discovery TCP option has been stripped on the path between the client-side Steelhead appliance and the server-side Steelhead appliance.

If the pass-through type is *Intentional pass-through*, then the auto-discovery probe is ignored because of a peering-rule.

7.3.4.2. Does the auto-discovery probe get detected?

Next step is to confirm that the server-side Steelhead appliance does see an auto-discovery probe in the SYN+ packet.

Figure 7.12. Running tcpdump on the client-side WAN side, seeing the SYN+ packet

```
CSH # tcpdump -ni wan0_0 '(host 192.168.1.1 and host 10.0.1.1 and port 543) or (vlan and ( \
    host 192.168.1.1 and host 10.0.1.1 and port 543))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
10:05:30.767917 IP 10.0.1.1.57489 > 192.168.1.1.543: Flags [S], seq 1098146093, win 65535, \
     options [mss 1460,nop,wscale 3,nop,nop,TS val 347290479 ecr 0,sackOK,nop,nop,rvbd-pro \
    be AD CSH:10.0.1.6 01010a0001060005,rvbd-probe EAD 0c01,nop,eol], length 0
10:05:30.900270 IP 192.168.1.1.543 > 10.0.1.1.57489: Flags [S.], seq 2704856677, ack 10981 \
    46094, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 1624055221 ecr 34729047 \
    9], length 0
10:05:30.903472 IP 10.0.1.1.57489 > 192.168.1.1.543: Flags [.], seq 1, ack 1, win 65535, o \
    ptions [nop,nop,TS val 347290613 ecr 1624055221], length 0
```

Figure 7.13. Running tcpdump on the server-side WAN side without the SYN+ packet

```
CSH # tcpdump -ni wan0_0 '(host 192.168.1.1 and host 10.0.1.1 and port 543) or (vlan and ( \
    host 192.168.1.1 and host 10.0.1.1 and port 543))'
tcpdump: WARNING: wan0_0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wan0_0, link-type EN10MB (Ethernet), capture size 300 bytes
09:47:29.245645 IP 10.0.1.1.57489 > 192.168.1.1.543: Flags [S], seq 1098146093, win 65535, \
     options [mss 1460,nop,wscale 3,nop,nop,TS val 347290479 ecr 0,sackOK,nop,nop,nop,nop, \
    nop,nop,nop,nop,nop,nop,nop,nop,nop,nop,nop,eol], length 0
09:47:29.245784 IP 192.168.1.1.543 > 10.0.1.1.57489: Flags [S.], seq 2704856677, ack 10981 \
    46094, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 1624055221 ecr 34729047 \
    9], length 0
09:47:29.382200 IP 10.0.1.1.57489 > 192.168.1.1.543: Flags [.], seq 1, ack 1, win 65535, o \
    ptions [nop,nop,TS val 347290613 ecr 1624055221], length 0
```

If a normal naked SYN packet gets seen on the server-side Steelhead appliance, although there was a SYN+ packet send by the client-side Steelhead appliance, then it is stripped somewhere in the path between the client-side Steelhead appliance and the server-side Steelhead appliance. Firewalls, intrusion detection and prevention systems (IDS/IPS) are well known for removing the probes.

If there is neither a SYN+ nor a naked SYN packet seen on the server-side Steelhead appliance, then the path of the traffic between the client and the server does not include the server-side Steelhead appliance. Use the traceroute program to determine the path of the traffic.

7.3.4.3. Return of SYN/ACK+ packet

When the server-side Steelhead appliances sees a SYN+ packet and it is willing to peer with it, it will return a SYN/ACK+ packet. This packet should arrive at the client-side Steelhead appliance. If it doesn't get seen at the client-side Steelhead appliance, then the path the traffic takes from the server-side Steelhead appliance to the client does not go via the client-side Steelhead appliance.

If both the Steelhead appliances see the auto-discovery probes, then the inner channel will be setup.

7.3.5. Cause of the failure of setup of inner channel

Once the two Steelhead appliance have detected each other via the auto-discovery probe in the TCP Options, the client-side Steelhead will setup an inner channel to the server-side Steelhead appliance.

7.3.5.1. With the Correct Addressing WAN Visibility mode

The behaviour in setup of an inner channel is different when the two appliances do or don't have a Out-of-Band Splice setup between them.

This can be determined via the list of Connected Appliances (available with the CLI command `show peers` and in the GUI at Reports -> Optimization -> Connected Appliances). If the peer in-path IP address shows up as Online, then an OOB Splice has been established already.

7.3.5.1.1. Out-of-Band Splice does not yet exist

When no Out-of-Band Splice exists, the optimization service will setup a new OOB TCP session between the IP address of the in-path interface on the client-side Steelhead appliance and the IP address of the in-path interface on the server-side Steelhead appliance on TCP port 7800.

On the client-side Steelhead appliance a TCP SYN packet will be sent, it should arrive on the server-side Steelhead appliance and be replied by a TCP SYN/ACK packet. The TCP SYN/ACK packet should arrive on the client-side Steelhead appliance and it should be replied with by a TCP ACK packet.

Note that the path the SYN+ packet from the client takes can be different from the path that the TCP SYN packet from the inner channel. This also means that the behaviour of various networking equipment, like routing decisions, the traffic shaper behaviour and the firewall policy, can be different.

Once the TCP session for the OOB splice is successfully setup, the Steelhead appliances will perform the OOB handshake during which they will exchange details and capabilities and the OOB Splice is completed. After that, unless one of the Steelhead appliances goes into Admission Control and will terminate the OOB Splice, there will be no more data exchanged over the OOB Splice.

If the OOB Splice does not get established, the inner channel for the optimized TCP session will also not be setup and the TCP session will be passed-through. Use the traceroute command to see if there is a difference in the path the traffic takes from the client to the server and in the path it takes for traffic from the IP address of the in-path interface of the client-side Steelhead appliance to the IP address of the in-path interface of the server-side Steelhead appliance.

The next step is the setup of a Connection Pool of 20 TCP sessions which can be converted later to inner channels. These are normal TCP sessions from the IP address of the in-path interface of the client-side Steelhead appliance to the IP address of the in-path interface of the server-side Steelhead appliance on TCP port 7800, but no data is send over it until later. If the OOB Splice got setup, then the setup of these TCP sessions shouldn't be a problem.

The final step is the conversion of one of the TCP sessions of the Connection Pool into an inner channel by sending the inner channel handshake over the TCP session and once acknowledged with a TCP ACK, the inner channel is established.

7.3.5.1.2. Out-of-Band Splice already exists

When the Out-of-Band Splice already exists, there will also be the Connection Pool, a pool of pre-setup TCP sessions between the in-path interfaces of the client-side and server-side Steelhead appliances available to be converted into inner channels for optimized TCP sessions.

The Connection Pool exists as long as the OOB Splice exists. The OOB Splice is a TCP session with a very short TCP keep-alive interval and thus should be the first to know that the server-side Steelhead is not reachable.

Before they start being used, these TCP sessions in the Connection Pool might have already been there for a long time, lying idle until a new TCP session gets optimized. When needed, the inner channel handshake gets send over one of the TCP sessions in the Connection Pool and once acknowledged with a TCP ACK, the TCP session is converted into an inner channel.

If the TCP ACK expected after the inner channel handshake does not show up at the client-side Steelhead appliance, then the inner channel might have been timed out in a session-table in a firewall.

If the reply to the inner channel handshake is a TCP RST packet, then there is possibly a firewall in the middle which has expired the TCP session.

7.3.5.2. With the Port Transparency WAN Visibility mode

With the Port Transparency WAN Visibility mode, the inner channel consists of a TCP session from the IP address of the in-path interface of the client-side Steelhead appliance to the IP address of the in-path interface of the server-side Steelhead appliance, but with the TCP port numbers of the original TCP session.

Because of the different TCP port numbers, it is not possible to setup a Connection Pool in advance and the setup of the TCP session for the inner channel does need to happen for every optimized TCP session with the Port Transparency WAN Visibility mode.

After the auto-discovery phase, the client-side Steelhead appliance will send a TCP SYN packet with the Transparency TCP Option in it. This Transparency TCP Option has the IP addresses of the in-path interfaces of the Steelhead appliances in it and the TCP port numbers used on the Steelhead appliances to identify the inner channel.

Just with the Correct Addressing WAN Visibility, the path that the TCP SYN packet with the Transparency TCP option takes can be different from the path that the SYN+ packet takes. This also means that the behaviour of various networking equipment, like the routing decisions, the traffic shaper behaviour and the firewall policy, can be different.

If this TCP SYN packet with the Transparency TCP Option does not reach the server-side Steelhead appliance, check the path the TCP SYN packet takes with the traceroute command.

If the path is the same, then check if a normal TCP session without the Transparency TCP Options towards the IP address of the server-side Steelhead appliance but to the server TCP port does arrive on the server-side Steelhead appliance. If it does arrive there, then there is something in the path which dropped the packet because of the Transparency TCP Option.

After the TCP SYN packet with the Transparency TCP Option arrives at the server-side Steelhead appliance, it will send a TCP SYN/ACK with the Transparency TCP Option back to the client-side Steelhead appliance which will send the final TCP ACK with the Transparency TCP Option and the TCP session of the inner channel is setup.

Once the TCP session of the inner channel is setup, the client-side Steelhead appliance will send the inner channel handshake over the TCP session and once acknowledged with a TCP ACK, the inner channel is established.

7.3.6. With the Full Transparency WAN Visibility mode

With the Full Transparency WAN Visibility mode, the inner channel consists of a TCP session from the IP address of the original TCP session and with the TCP port numbers of the original TCP session.

Also here it is not possible to setup a Connection Pool in advance and the setup of the TCP session for the inner channel does also need to happen for every optimized TCP session with the Full Transparency WAN Visibility mode.

After the auto-discovery phase, the client-side Steelhead appliance will send a TCP SYN packet with the Transparency TCP Option in it, just like with Port Transparency.

Unlike with the Correct Addressing and the Port Transparency WAN Visibility, the path that the TCP SYN packet with the Transparency TCP option takes is the same as the SYN+ packet takes. This means that firewall networking equipment might complain now: The previous TCP SYN packet had different TCP sequence numbers.

If this TCP SYN packet with the Full Transparency TCP Option does not reach the server-side Steelhead appliance, change the in-path rule from the Full Transparency WAN Visibility to the Full Transparency With Firewall Reset WAN Visibility. This will cause the Steelhead appliance to send a TCP RST before sending the TCP SYN packet with the Full Transparency TCP Options, attempting to clear any firewall state tables.

After the TCP SYN packet with the Transparency TCP Option arrives at the server-side Steelhead appliance, it will send a TCP SYN/ACK with the Transparency TCP Option back to the client-side Steelhead appliance which will send the final TCP ACK with the Transparency TCP Option and the TCP session of the inner channel is setup.

Once the TCP session of the inner channel is setup, the client-side Steelhead appliance will send the inner channel handshake over the TCP session and once acknowledged with a TCP ACK, the inner channel is established.

7.4. An optimized TCP Session gets reset or hangs

The final result is the same, the client and the server stop communicating with each other, but the cause is different:

An optimized TCP session hangs when traffic in one direction is blocked, an optimized TCP session gets reset when the client or the server gets a TCP RST packet.

7.4.1. Reset TCP sessions

A TCP session gets reset when it receives a TCP RST packet. TCP RST packets are generated when:

- A TCP SYN packet arrives at the server but there is no service listening on the destination TCP port. The client will report this as *Connection refused*.
- A TCP SYN/ACK packet arrives at the client but the client doesn't have a socket open on that TCP port. This can happen the program which has opened the socket has already closed it before the TCP SYN/ACK returns.
- A TCP packet arrives at a client or server but that one doesn't know about this TCP session anymore. This can happen when a long idle TCP session comes back alive and the machine on the other side has been restarted.
- A TCP packet arrives at a client or server but the TCP sequence numbers are invalid. This can happen in WAN optimization environments where the SYN/ACK+ bypasses both the client-side and server-side Steelhead appliances.
- A firewall sees traffic for a TCP session which is not in its TCP session table. This can happen when the TCP session comes back online after it has been idle for a long time and the TCP session has timed-out on the firewall.
- A firewall sees incorrect TCP sequence behaviour. This happens when traffic for a second TCP session with the same IP addresses and TCP port numbers gets seen. This can happen when using the Full Transparency WAN Visibility.
- A firewall detects a time-out on a state change in the TCP three-way handshake and resets the TCP session towards the client and the server to alert it that the TCP session could not be established. This can happen during the auto-discovery process when a firewall in the middle only sees the SYN+ and SYN/ACK+ and not the final TCP ACK.
- An Intrusion Prevention System detects a violation of the protocol on top of TCP and aborts the TCP session. This can happen when the Port Transparency or the Full Transparency WAN Visibility is used in a network with an IPS device between the Steelhead appliances.
- For an optimized TCP session, which consists of three TCP sessions, the inner channel between the two Steelhead appliances fails and therefore the two outer channels also get reset.

7.4.2. Hanging TCP sessions

A hanging TCP session happens when traffic being sent out from one side is blocked in the network and the TCP Window of the sender is filling up and stops sending further packets.

This can happen when:

- A device in the path is buffering the traffic while it waits for a network path to become available. This could happen when a VPN tunnel gets restarted or when a non-IP routing device is buffering the packets.
- A device in the path is blackholing the traffic.
- A device in the path cannot forward the IP packet because of MTU size issues and the ICMP Fragmentation Needed packets send by the blocking device do not arrive at the sender.
- An Intrusion Prevention System has detected a protocol violation on top of TCP and discards the packet. This can happen when the Port Transparency or the Full Transparency WAN Visibility is used in a network with an IPS between the Steelhead appliances.

This issue can be detected by tracing the TCP session at the sender and see that despite sending various TCP packets, there is no TCP acknowledgement coming from the other side.

7.5. Slow network

This part looks at the network being slow and what the causes can be.

7.5.1. Is the TCP session optimized?

In the list of Current Connections the TCP session should be shown as being optimized and not pass-through.

If it is pass-through, then why is it in pass-through? In the Current Connections reports, either from the GUI under Reports -> Networking -> Current Connections or from the CLI with the command `show connections all`, is shown if the TCP session is pass-through because of an intentional reason (That is an in-path rule which prevents the TCP session from being optimized) or because of an unintentional reason:

- SYN on WAN side: The naked SYN packet arrived on the WAN side of the device. This is normal on data-center Steelhead appliances for traffic which will terminate on the hosts in the data-center. If this TCP session is expected to be optimized, then check what is happening with the naked SYN and SYN+ packets during the auto-discovery phase.
- No Peer: The auto-discovery probe in the SYN+ did not get intercepted by another Steelhead appliance. If this TCP session is expected to be optimized, then check what is happening with the SYN+ packet during the auto-discovery phase.
- Pre-existing connection: The first packet for this TCP session seen by the optimization service was not the TCP handshake in the beginning of the stream but another packet halfway the stream. This can happen when the optimization service has been restarted. The way out is to either reset the TCP session from the GUI or CLI or to restart the application on the client.
- Asymmetric Routing: The source and destination IP addresses are in the asymmetric routing table.

A full list of them can be found at KB article S15377 - "What are the Reason Codes for pass-through connections in the Current Connections report?" available from the Riverbed Support website.

7.5.2. Is the Latency Optimization working?

In the list of Current Connections in the GUI the TCP session will show a red triangle if the latency optimization has failed and has reduced the optimization functionality back to bandwidth reduction only.

The following issues can occur:

- A CIFS session is requested to use SMBv2 and the Steelhead appliance is not able to perform latency optimization on this traffic (Available since RiOS version 6.5). A workaround can be to let the Steelhead appliance negotiate the SMB version back to SMB version 1.
- A CIFS session uses SMB Signing or a MAPI session encrypted MAPI and the server-side Steelhead appliance isn't configured to support this kind of data.
- A CIFS session uses SMB Signing or a MAPI session encrypted MAPI and the server-side Steelhead appliance isn't able to impersonate the user against the AD infrastructure. This issue is logged on the server-side Steelhead appliance and can be related to:
  - Inability to determine the Domain Controllers via DNS or to reach a certain Domain Controller.
  - In case the Steelhead appliance is configured to use to one specific Domain Controller, the Domain Controller can have been decommissioned or has out-of-sync data on it.

    - Steelhead appliance object in Active Directory has disappeared.
    - Delegate user in Active Directory has been locked out or has its password changed.
    - Absence of delegate user in Active Directory.
    - Absence of the CIFS server in the Delegate list on the delegate user in Active Directory.
    - Protocol issues between the Steelhead appliance and a Domain Controller (For example, Windows 2008R2 servers where not supported to delegate to until RiOS version 6.1)
    - Time skew between the server-side Steelhead appliance and the Domain Controllers.

- A signed CIFS session uses a system account to map a drive. Since no user account is used here, the optimization service cannot impersonate anything on this CIFS session. A workaround to reduce the side-effect of blacklisting this server would be to use a special CIFS server for shares mapped by computer accounts so that the blacklisting of this server doesn't affect the optimization of other CIFS shares. This can be overcome by using Kerberos authentication.
- A MAPI session uses a MAPI dialect which is not supported for latency optimization. Examples are:
    - Blackberry related traffic.
    - Exchange to Exchange traffic.
- An HTTP session is using an HTTP protocol which uses other methods than GET and PUT. An example of this is the DAV Filesystem which is layered on top of HTTP which is not supported until RiOS 8.5.

7.5.3. Is the slowness on the network?

At this moment, the TCP session is optimized and the latency optimization is working fine. The next step is to determine what goes slow and what goes fast and what the Steelhead appliance complains about.

7.5.3.1. CIFS related issues

The questions which need to be asked are:

- Is the slowness happening during the reading by an application? This could be a file server issue, like file locking or authentication / authorization.
- Is the slowness happening during the copying from the file server onto the computer? The expected behaviour here is a fast copy since there is no locking happening.
- Is the slowness happening during a write operation by an application and also on a re-save without any changes? The first one could be slow, the second one is expected to be fast.
- Is the slowness happening during a write operation from the computer to the file server? That should be fast, if the file had not changed since you copied it from the file server.

7.5.3.1.1. Response times

For CIFS latency optimization, we expect fast local acknowledgments on the SMB requests like closing files, disconnecting from a share and obtaining a directory listing.

In Wireshark you can see this under Statistics -> Service Response Times -> SMB.

Figure 7.14. Latency optimized SMB requests response times overview

```
=================================================================
SMB SRT Statistics:
Filter:
Commands                   Calls    Min SRT    Max SRT    Avg SRT
Close                          1   0.000212   0.000212   0.000212
Open AndX                      1   0.134227   0.134227   0.134227
Read AndX                     18   0.354067   0.887230   0.646874
Tree Disconnect                1   0.000169   0.000169   0.000169
Negotiate Protocol             1   0.133985   0.133985   0.133985
Session Setup AndX             2   0.133340   0.136641   0.134991
Tree Connect AndX              2   0.132323   0.134432   0.133378
Query Information Disk         1   0.222203   0.222203   0.222203

Transaction2 Commands      Calls    Min SRT    Max SRT    Avg SRT
FIND_FIRST2                    2   0.287443   0.560205   0.423824
FIND_NEXT2                     4   0.000094   0.009760   0.004776
QUERY_FILE_INFO                1   0.134086   0.134086   0.134086
GET_DFS_REFERRAL               1   0.133184   0.133184   0.133184

NT Transaction Commands    Calls    Min SRT    Max SRT    Avg SRT
=================================================================
```

If the response times of Close, Tree Disconnect and FIND_NEXT2 are not significantly lower, the CIFS latency optimization might be disabled or not working.

If it is only the response times of the write requests which are not significantly lower, then it might be that the optimization service has detected that the remote disk is near full capacity and has turned off its Write Behind feature for that CIFS session.

Figure 7.15. CIFS Write Behind has been disabled by the optimization service.

```
CSH sport[14140]: [smbcfe.NOTICE] 679459 {10.0.1.1:3346 192.168.1.1:445} Disk usage reache \
    d threshold (2048 MB free) on share \FOO\BAR, disabling write/setfile pre-acking
```

7.5.3.1.2. File formats

When saving a file via an application, the file format it uses is very important for the performance of the transmission of the file.

For example when the file format is uncompressed, a little change in the content will only affect a small of the file.

Take these two files which are nearly the same. Changing a single line in the text one generates a diff of 2 lines. After compression, the files are not only different in size, but as the output of bsdiff (a binary patch generator) shows, the contents of the file are also different enough to generate a patch which is larger than the original file.

Figure 7.16. Comparison of two binary files

```
[~/size] edwin@t43>ls -al gcc.1 gcc.2
-rw-r--r--  1 edwin  edwin  584775 Oct 21 16:58 gcc.1
-rw-r--r--  1 edwin  edwin  584775 Oct 21 16:58 gcc.2
[~/size] edwin@t43>wc gcc.1
   13118   79291  584775 gcc.1
[~/size] edwin@t43>diff gcc.1 gcc.2
134c134
< XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
---
> gcc \- GNU project C and C++ compiler

[~/size] edwin@t43>zip gcc.1.zip gcc.1; zip gcc.2.zip gcc.2
updating: gcc.1 (deflated 72%)
updating: gcc.2 (deflated 72%)
[~/size] edwin@t43>bsdiff gcc.1.zip gcc.2.zip bsdiff
[~/size] edwin@t43>ls -al gcc.1.zip gcc.2.zip bsdiff
-rw-r--r--  1 edwin  edwin  163096 Oct 24 07:32 bsdiff
-rw-r--r--  1 edwin  edwin  162215 Oct 24 07:22 gcc.1.zip
-rw-r--r--  1 edwin  edwin  162228 Oct 24 07:22 gcc.2.zip
```

Because of this huge change, the performance of a save inside the appliance will be like an initial cold transfer.

7.5.3.2. Database connections

Communication towards a database can be done in two ways:

- The client sends a request and asks for the complete data set. The result will be large blob of data which will have a great data-reduction and speed improvement on it.

- The client sends a request and asks for the first record, then the next record and the next record. Because of the constant exchange of single packet questions and answers with a small payload, the optimization level will be very small. To make things worse, various optimization features like Neural Framing will become a major delay for each packet.

  The best way to investigate if this stream can be optimized is:

  - Create an auto-discovery in-path rule which targets the traffic towards the IP address of the database server and TCP port. Perform the query and measure how long it takes.

  - Alter the in-path rule to disable Neural Framing. Perform the query and measure how long it takes.

  - Alter the in-path rule to become a pass-through rule. Perform the query and measure how long it takes.

  Depending on which one goes best, use that method.

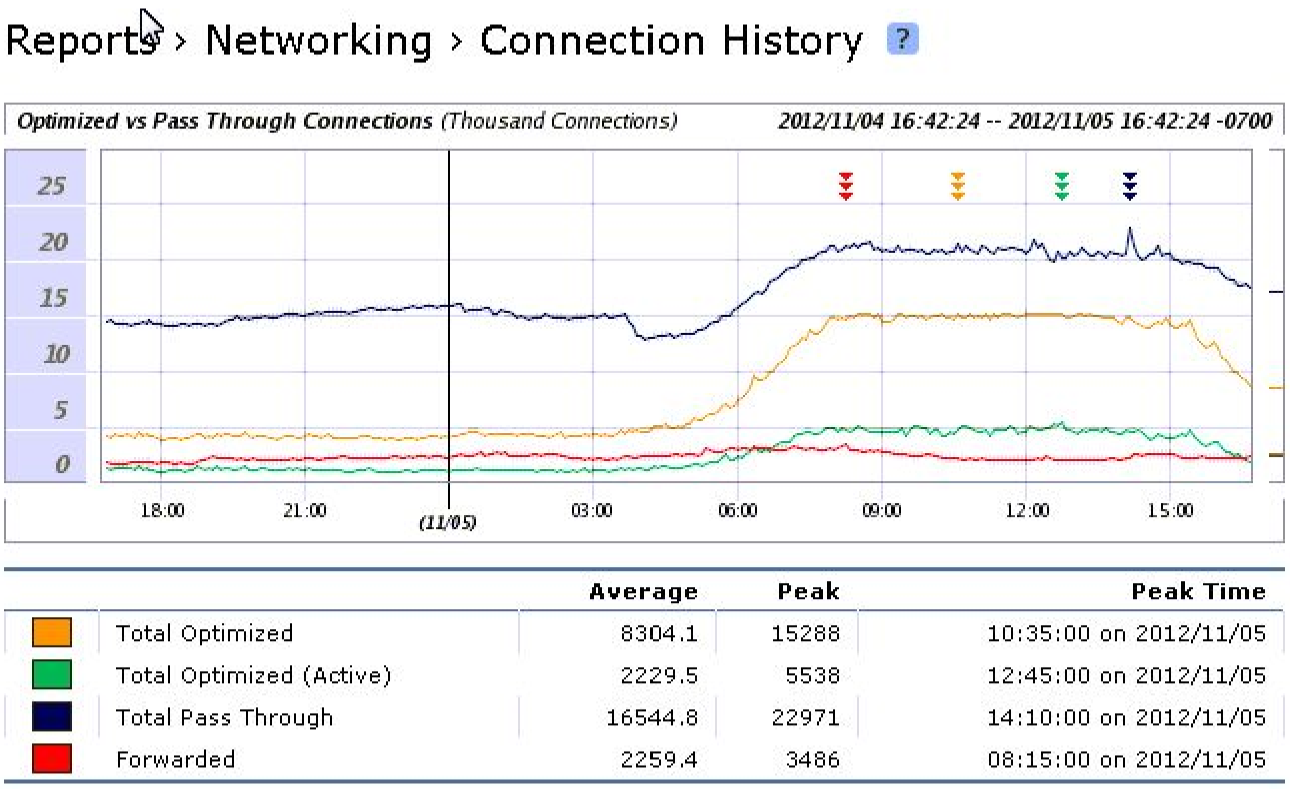
The best way forward here is to change the client application to perform the query in a large batch method instead of performing a large amount of small queries.

7.6. Connection-based Admission Control

As discussed earlier, there are several kinds of admission control. This chapter only focuses on connection-based admission control, where the number of optimized TCP sessions going through the Steelhead appliance is larger than the licensed amount.

The first symptom will be the alarm on the Steelhead appliance, delivered to you via email or an SNMP trap. In the reports section, the graph of the Connection History will show a flat line at the time the Steelhead appliance was in connection-based admission control:

Figure 7.17. Connection-based admission control for a 5520 model


Reports › Networking › Connection History
Optimized vs Pass Through Connections (Thousand Connections)
2012/11/04 16:42:24 -- 2012/11/05 16:42:24 -0700
25
20
15
10
5
0
18:00
21:00
(11/05)
03:00
06:00
09:00
12:00
15:00

		Average	Peak	Peak Time
🟧	Total Optimized	8304.1	15288	10:35:00 on 2012/11/05
🟩	Total Optimized (Active)	2229.5	5538	12:45:00 on 2012/11/05
■	Total Pass Through	16544.8	22971	14:10:00 on 2012/11/05
🟥	Forwarded	2259.4	3486	08:15:00 on 2012/11/05

At midnight, the number of optimized TCP sessions was below 15 000. At around 08:00, it has risen to 15 000 and the optimization service went into admission control. At 16:00, it has dropped back to below 15 000 and the optimization service came out of admission control.

There are several reasons for entering admission control:

- Sizing issues, caused by natural growth of the network or by a change in the protocols spoken.
- Malicious traffic, caused by clients which are consuming large amount of TCP sessions without any good reason.
- Failure to tear down TCP sessions, caused by clients or servers not properly taking down TCP sessions.

7.6.1. Sizing issues

In either of these two examples, the capacity of the current model is not sufficient anymore and should either be upgraded using a higher license or upgraded to a larger appliance if a license upgrade is not possible.

7.6.1.1. Natural growth

Natural growth is related to an increase of the number of clients behind the Steelhead appliance or to an increase in network-based applications.

7.6.1.2. Change in protocols spoken

A change in the protocols spoken is related to a migration from one protocol to another, like from POP3 to MAPI: POP3 uses short-lived TCP sessions so for 100 clients you can live with 20 concurrent TCP sessions for this, while MAPI uses at least 3 long-lived TCP sessions so for 100 clients you suddenly need about 300 concurrent TCP sessions already!

7.6.2. Malicious traffic

Use the Current Connections overview to see if there are obvious hosts to look at.

7.6.2.1. Port scanners

The behaviour of a TCP port scanner is that it sends out a large amount of TCP SYN packets and waits for any answers. When a Steelhead appliance sees these TCP SYN packets, it will try to setup optimized TCP sessions and can run into Connection-based admission control.

The way to find them is to run tcpdump and filter for packets with the TCP SYN flag set with the command `tcpdump -ni lan0_0 -c 100 'tcp[tcpflags] & tcp-syn != 0'`.

The best practice is to create an in-path rule on the local Steelhead appliance to not optimize any traffic from the host which is doing the port scanning.

7.6.2.2. How to determine the application

Once the offending host has been determined, the next step will be to identify the application which is sending the traffic.

On Linux and Unix systems, this can be done with the command `lsof`:

Figure 7.18. Use "lsof" to determine which application is owning a TCP session

```
$ lsof -n | grep 192.168.1.1:543
nc       5637 edwin     3u     IPv4   56433930    TCP  192.168.1.1:543->10.0.1.1:53036 (ES \
    TABLISHED)
```

In this case the application *nc* which is using the TCP session between 10.0.1.1:53036 and 192.168.1.1:543.

On Microsoft Windows, this can be done with the output of `netstat -no` to determine the Process ID:

Figure 7.19. Use "netstat -no" to determine the Process ID of the application owning a TCP session

```
C:\Users\edwin.MAVETJU>netstat -no

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  [...]
  TCP    10.17.6.60:51749       173.194.79.125:5222    ESTABLISHED     1908
  [...]
```

And the Windows Task Manager, add the PID to the output via View -> Select Columns, to determine the application:

Figure 7.20. Windows Task Manager with Process ID

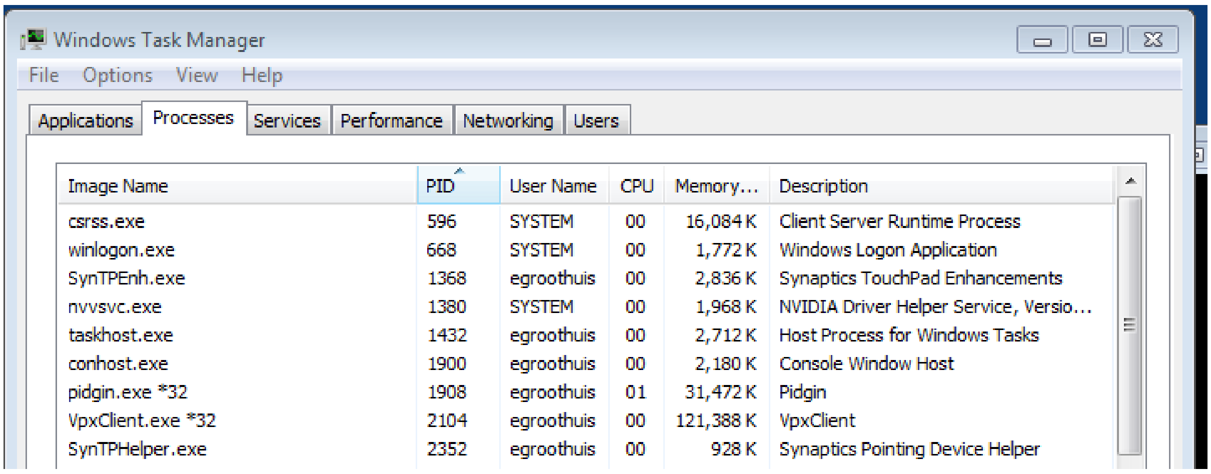

Windows Task Manager
File Options View Help
Applications Processes Services Performance Networking Users

Image Name	PID	User Name	CPU	Memory...	Description
csrss.exe	596	SYSTEM	00	16,084 K	Client Server Runtime Process
winlogon.exe	668	SYSTEM	00	1,772 K	Windows Logon Application
SynTPEnh.exe	1368	egroothuis	00	2,836 K	Synaptics TouchPad Enhancements
nvvsvc.exe	1380	SYSTEM	00	1,968 K	NVIDIA Driver Helper Service, Versio...
taskhost.exe	1432	egroothuis	00	2,712 K	Host Process for Windows Tasks
conhost.exe	1900	egroothuis	00	2,180 K	Console Window Host
pidgin.exe *32	1908	egroothuis	01	31,472 K	Pidgin
VpxClient.exe *32	2104	egroothuis	00	121,388 K	VpxClient
SynTPHelper.exe	2352	egroothuis	00	928 K	Synaptics Pointing Device Helper

In this case the application "pidgin.exe" which is using the TCP session between 10.0.1.1:51749 and 173.194.79.125:5222.

7.6.3. Failure to tear down TCP sessions

A TCP session starts with a SYN, SYN/ACK and ACK handshake and is terminated with a RST if things go wrong or with a FIN where both side have successfully received all the traffic.

Once the client or the server has sent a FIN, the TCP session is in a half-closed state. Once both the client and the server have sent a FIN, the TCP session is closed.

As long as an optimized TCP session is in the half-closed state, it counts towards the TCP session usage. If the FIN by the other side is never send, the TCP session on the Steelhead appliance will stay open indefinitely. In the Current Connection report, you will then see an abnormal large amount of half-closed TCP sessions. For example for a 250H model, it will show 20 optimized TCP sessions and 200 half-closed TCP sessions.

According to the RFC about TCP, the half-closed timer for TCP sessions should expire after 2 times MSL (Maximum Segment Lifetime, most common is 60 seconds). So to overcome this half-closed state issue, the following CLI command can be entered to close the half-closed TCP sessions after certain period: `protocol connection lan half-closed kill-timeout <number of seconds>`.

If a half-closed TCP session is about to be terminated due to this time-out, it will show up in the logs as:

Figure 7.21. A half-closed TCP is about to be terminated

```
SH sport[431]: [io/outer/cons.INFO] 5 {192.168.0.1:1025 10.0.1.1:80} Kicking off half-clos \
    ed connection
```

Chapter 8. Latency Optimization

8.1. Index

This chapter describes various aspects of the latency optimization features.

8.2. Introduction to Latency Optimization

Latency Optimization is a WAN optimization method where the optimization is happening on the application protocol transported on top of the TCP layer.

Latency Optimization is based on the behaviour seen between the client and the server, to enable the Steelhead appliance to act on behalf of the server or the client to improve the response time for the client.

Latency Optimization is often controlled by the client-side Steelhead appliance, the capabilities are limited to the features available on both the client-side and server-side Steelhead appliance.

8.2.1. Protocol improvements

8.2.1.1. File system related protocol improvements

For network file system related protocols, like CIFS, SMBv2, SMBv3 and NFS, the following improvements can be made:

- Directory content prefetching, where the full contents of a directory and its meta-data is pre-read by the client-side Steelhead appliance when the client asks for the first directory entry.
- Read-ahead on read operations, where the next blocks of a file are read in advance by the client-side Steelhead appliance when the client asks for the first block of the file.

- Early acknowledgement on write operations, where the client-side Steelhead will acknowledge the writing of a block once it has transmitted the request over the wire but before the file server has remotely acknowledged it.

8.2.1.2. FTP related protocol improvements

There is no real protocol improvements, the only thing the FTP latency optimization does do is keep track of the FTP data channel to make sure that the traffic statistics are marked as such.

8.2.1.3. MS-SQL related protocol improvements

This is only for the database access as provided by MS-Access 2000.

8.2.1.4. Email related protocol improvements

For email related protocols, like MAPI and Lotus Notes, the following improvements can be made:

- Data store warming on attachments, where the client-side Steelhead appliance will read the attachment in advance of the request of the client.
- Read-ahead on attachments, where the client-side Steelhead appliance will read the next block of the attachment in advance of the request of the client.
- Write-behind on attachments, where the client-side Steelhead appliance will acknowledge the writing of the block of an attachment on behalf of the server.

8.2.1.5. HTTP related protocol improvements

For HTTP related protocols, the following improvements can be made:

- Caching of objects where the client-side Steelhead appliance will keep static objects in its objects cache for as long as specified by the HTTP server.
- Pre-acking of non-changed objects, where the client-side Steelhead appliance will answer requests for static objects based as long as specified by the HTTP server.
- Prefetching of static objects based on the HTML code returned in the response from the server.

8.2.2. What can go wrong?

Unlike TCP Optimization where the protocol is well documented, vendor neutral and compatibility towards other TCP stacks is a priority, latency optimization is done on protocols which are vendor specific protocol, with client and server behaviour not always well known, and with compatibility layers towards older versions of the protocol.

Latency optimization is done for specific server and specific client actions and capabilities, based on the behaviour observed for certain client-side actions. This behaviour can differ between different versions of the protocol and even between different versions of the client and server software.

Moving away from the versions of the clients and servers supported by the version of RiOS running on the Steelhead appliances could cause problems in the latency optimization because of yet unsupported changes in the client and server behaviour.

Possible issues which can go wrong:

- New versions of the protocol which implement new features which are not compatible with the current protocol.
- New versions of the protocol which cannot fallback to earlier versions. For example, some SMB version 2 clients can fallback to the CIFS version 1 protocol.

- Servers which do not implement the protocol as expected, for example by returning statuses which could be considered valid for the client but are considered fatal for the latency optimization.

Before doing upgrades of the client and the server software for protocols on which latency optimization is performed, please make sure that the new protocols are supported by the software versions running on Steelhead appliances.

8.3. CIFS Latency Optimization

CIFS latency optimization works in the following ways:

- Directory index pre-fetching and directory meta-data subscription.
- File read-ahead and write-behind.

CIFS latency optimization does not work if:

- The client or server operating system is not supported. For example the AS/400 CIFS client is not supported as a client.
- CIFS traffic is signed and the Active Directory integration failed. For example Active Directory user delegation for the CIFS server being optimized to is not enabled in the Active Directory configuration. Another example is that the Windows 2008 R2 server wasn't supported for Active Directory integration until RiOS version 6.1.
- If the server-side Steelhead appliance cannot get a lock on a file. This can happen on CIFS file servers which have this file locking as a non-default option like the NetApp file server.
- If the SMB version required by the client is not supported by the Steelhead appliance. For example SMBv2 is supported only from RiOS version 6.5 and SMBv3 is supported since RiOS version 8.5.

The result will be that the interactive performance of the CIFS session will be like an unoptimized CIFS session and that during file-transfers only data-reduction will happening.

8.3.1. Setup of an optimized CIFS session

When a Windows client attaches to a CIFS share, it will setup two TCP sessions towards the server: One on port 445 and one on port 139. The one on port 139 will be terminated immediately if the one on port 445 sets up properly.

Figure 8.1. Example of a TCP session on port 139 being terminated immediately

```
CSH sport[6693]: [splice/probe.NOTICE] 0 {- -} (locl: 10.0.1.6:50499 clnt: 10.0.1.1:2686 s \
    erv: 192.168.1.1:139) No inner channel created for this probe splice
CSH sport[6693]: [clientsplice.ERR] - {clnt:10.0.1.1:2690 peer: 192.168.1.6:7800 serv:192. \
    168.1.1:139} End of stream reading connect result
```

The first line gets logged when the TCP RST packet was received by the client-side Steelhead appliance before the inner channel was setup, the second line gets logged when the TCP session gets closed by the server without having send or received any data over the TCP connection.

8.3.2. Characteristics of an optimized CIFS session:

Since the Steelhead appliance is a WAN optimizer and not a network file system, all file system related operations and retrieval of data are still coming from the remote file servers. Requests for the creation, opening, closing and removal of a file are still fully going to the remote file server and answered by the remote file server. The reading from and writing to files will be locally acknowledged by the Steelhead appliance and then forwarded to the remote file server.

To display the response times in Wireshark go to the Statistics -> Service Response Times -> SMB.

Figure 8.2. Wireshark Response time statistics for an unoptimized CIFS session

```
=================================================================
SMB SRT Statistics:
Filter:
Commands                   Calls    Min SRT    Max SRT    Avg SRT
Close                          1   0.133577   0.133577   0.133577
Open AndX                      1   0.133348   0.133348   0.133348
Read AndX                     18   0.368455   1.169644   0.902801
Tree Disconnect                2   0.133128   0.133643   0.133386
Negotiate Protocol             1   0.132288   0.132288   0.132288
Session Setup AndX             2   0.132696   0.135587   0.134142
Tree Connect AndX              2   0.133005   0.133278   0.133142
Query Information Disk         1   0.132691   0.132691   0.132691

Transaction2 Commands      Calls    Min SRT    Max SRT    Avg SRT
FIND_FIRST2                    2   0.136227   0.558457   0.347342
FIND_NEXT2                     4   0.133153   0.135975   0.134642
QUERY_FILE_INFO                1   0.133137   0.133137   0.133137
GET_DFS_REFERRAL               1   0.132695   0.132695   0.132695

NT Transaction Commands    Calls    Min SRT    Max SRT    Avg SRT
=================================================================
```

All SMB commands take at least the 130 ms of the RTT time.

Figure 8.3. Wireshark Response time statistics for an optimized CIFS session

```
=================================================================
SMB SRT Statistics:
Filter:
Commands                   Calls    Min SRT    Max SRT    Avg SRT
Close                          1   0.000212   0.000212   0.000212
Open AndX                      1   0.134227   0.134227   0.134227
Read AndX                     18   0.354067   0.887230   0.646874
Tree Disconnect                1   0.000169   0.000169   0.000169
Negotiate Protocol             1   0.133985   0.133985   0.133985
Session Setup AndX             2   0.133340   0.136641   0.134991
Tree Connect AndX              2   0.132323   0.134432   0.133378
Query Information Disk         1   0.222203   0.222203   0.222203

Transaction2 Commands      Calls    Min SRT    Max SRT    Avg SRT
FIND_FIRST2                    2   0.287443   0.560205   0.423824
FIND_NEXT2                     4   0.000094   0.009760   0.004776
QUERY_FILE_INFO                1   0.134086   0.134086   0.134086
GET_DFS_REFERRAL               1   0.133184   0.133184   0.133184

NT Transaction Commands    Calls    Min SRT    Max SRT    Avg SRT
=================================================================
```

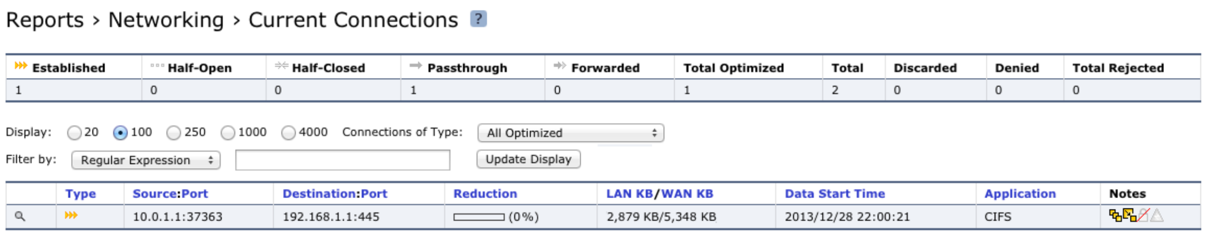
Locally handled commands like the Close and Tree Disconnect return immediately. Pre-fetched information like the Read AndX (18 * 0.646 seconds optimized versus 18 * 0.902 seconds unoptimized) and the FIND_NEXT2 commands (2 * 0.423 + 4 * 0.004 seconds optimized versus 2 * 0.347 + 4 * 0.134 seconds unoptimized) take much less time.

8.3.3. CIFS latency related issues

8.3.3.1. Skewed statistics due to read-ahead in the Windows Explorer

The preview feature in the Windows Explorer and the Mac OS X Finder only read parts of the file to get the meta-data of it, but the read-ahead functionality in the latency optimization service will read much more than ever will be requested by the client, therefore the statistics for that TCP option or for the that protocol might be skewed.

Figure 8.4. Skewed statistics on the client-side Steelhead appliance because of read-ahead in the Windows Explorer.


Reports › Networking › Current Connections

Established	Half-Open	Half-Closed	Passthrough	Forwarded	Total Optimized	Total	Discarded	Denied	Total Rejected
1	0	0	1	0	1	2	0	0	0

Display: ◯ 20 [✓ 100] ◯ 250 ◯ 1000 ◯ 4000 Connections of Type: All Optimized

Filter by: Regular Expression Update Display

	Type	Source:Port	Destination:Port	Reduction	LAN KB/WAN KB	Data Start Time	Application	Notes
		10.0.1.1:37363	192.168.1.1:445	(0%)	2,879 KB/5,348 KB	2013/12/28 22:00:21	CIFS	

This behaviour might be seen on the client-side Steelhead where the amount of WAN traffic is significant larger than the amount of LAN traffic, or when comparing the client-side and server-side LAN traffic where the amount of LAN traffic on the server-side is significantly larger than on the client-side.

Figure 8.5. Comparison of the client-side and server-side LAN traffic for CIFS latency optimization

```
XXX
```

8.3.3.2. CIFS read-ahead failure

The CIFS read-ahead feature works great when reading a file sequentially, but when the data is accessed randomly it will cause a large amount of unnecessary reads to the server and data transfers to the client-side Steelhead appliance. If the extensions of the files which are randomly accessed are known, it is possible to put them in a blacklist for the CIFS read-ahead feature:

Figure 8.6. Disable the CIFS read-ahead feature for files with the extension "myob"

```
CSH (config) # protocol cifs big-read-blklst add myob
You must restart the optimization service for your changes to take effect.
CSH (config) # show protocol cifs big-read-blklst
  myob
```

8.3.3.3. Slow file servers

When a request to a file server takes more than 500 milliseconds, the Steelhead appliance will log a message about this:

Figure 8.7. Heads up that the replies from the file servers are slow

```
CSH sport[27756]: [smbcfe.NOTICE] 379355 {10.0.1.1:49202 192.168.1.1:445} Request nt_creat \
    e_andx still waiting after 0.715137 sec, mid=2496 queue_cnt=1 queue_wait_state=1 enque \
    ue time=1347517794.630794
CSH sport[27756]: [smbcfe.NOTICE] 379355 {10.0.1.1:49202 192.168.1.1:445} Request nt_creat \
    e_andx still waiting after 1.715144 sec, mid=2496 queue_cnt=1 queue_wait_state=1 enque \
    ue time=1347517794.630794
CSH sport[27756]: [smbcfe.NOTICE] 379355 {10.0.1.1:49202 192.168.1.1:445} Request nt_creat \
    e_andx still waiting after 2.715170 sec, mid=2496 queue_cnt=1 queue_wait_state=1 enque \
    ue time=1347517794.630794
CSH sport[27756]: [smbcfe.NOTICE] 379355 {10.0.1.1:49202 192.168.1.1:445} Request nt_creat \
    e_andx still waiting after 4.715130 sec, mid=2496 queue_cnt=1 queue_wait_state=1 enque \
    ue time=1347517794.630794
```

The request types can be *nt_create_andx*, *nt_read_andx* and *transaction2*.

If this happens on the server-side Steelhead appliance, then check the load on the file server and any speed/duplex issues on the path towards the file server.

If this only happens on the client-side Steelhead appliance, then check the path towards the server-side Steelhead appliance.

8.3.3.4. Write-behind feature stops

The Write-behind feature stops if the CIFS share is more than 98% full and will be enabled again when the CIFS share is less than 92% full.

Figure 8.8. CIFS write-behind stopping because of a full disk.

```
CSH sport[14140]: [smbcfe.NOTICE] 679459 {10.0.1.1:3346 192.168.1.1:445} Disk usage reache \
    d threshold (2048 MB free) on share \FOO\BAR, disabling write/setfile pre-acking
```

8.3.3.5. Oplock failure

An oplock is a feature which deals with the opening of the same file by multiple clients. When one client has a lock on a file, any other clients cannot get a lock until the first client has released the lock. If the server-side Steelhead appliance deals with the lock, it will be able to do better read-ahead optimization.

If the server-side Steelhead appliance cannot obtain the oplock, it will not perform read-ahead on the file and the CIFS performance to the client will suffer from it:

Figure 8.9. Unable to obtain oplock on a file

```
CSH sport[22371]: [smbcfe.INFO] 2720339 {10.0.1.1:11847 192.168.1.1:445} process_create_an \
    dx_response file was not granted any oplock File: \Share\book.pdf Desired Access : 0x2 \
    0089 FID : 0x4012
```

Possible reasons are:

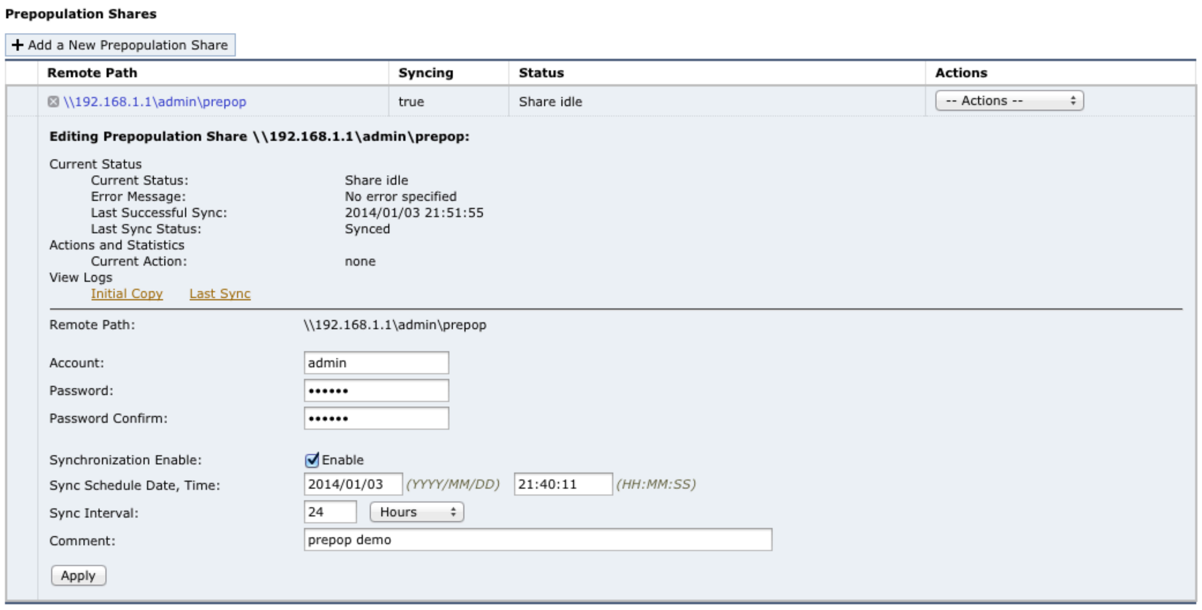
- The file server does not support oplocks. Please check the configuration of the file server.
- The file server was accessed by a remote client whose CIFS session did not get optimized or which CIFS session did not go through this Steelhead appliance.
- The file was opened on the console of the file server.

8.4. CIFS Pre-population

Strictly speaking CIFS pre-population is not a latency optimization feature but a data store warming feature. The feature overcomes the initial cold data transfer of new files read from a remote CIFS share by going through the remote CIFS share during the night and performing the initial cold transfer. As a result the next morning the data store is warm for the contents of the new files from the remote CIFS share and the users immediately experience the full optimization.

The configuration of the CIFS pre-population requires name of the remote CIFS share, the username and the password to access the remote CIFS share. All files and directories on that remote CIFS share should be readable by that user.

Figure 8.10. CIFS pre-population configuration



	Remote Path	Syncing	Status	Actions
	\\192.168.1.1\admin\prepop	true	Share idle	-- Actions --

Editing Prepopulation Share \\192.168.1.1\admin\prepop:

Current Status

Current Status:	Share idle
Error Message:	No error specified
Last Successful Sync:	2014/01/03 21:51:55
Last Sync Status:	Synced

Actions and Statistics

Current Action:	none

View Logs

Initial Copy Last Sync

Remote Path:	\\192.168.1.1\admin\prepop
Account:	admin
Password:	••••••
Password Confirm:	••••••
Synchronization Enable:	✓ Enable
Sync Schedule Date, Time:	2014/01/03 (YYYY/MM/DD) 21:40:11 (HH:MM:SS)
Sync Interval:	24 Hours
Comment:	prepop demo

Apply

The optimization service on the client-side Steelhead appliance needs to see the naked SYN of the CIFS session. The TCP SYN packet will leave the primary interface and should go arrive on the LAN interface where it will be processed as a normal TCP session to be optimized and go through the in-path rules.

When modifying the CIFS pre-population share from the CLI, the share needs to be entered like a Windows path with backslashes. Because the backslash has a different meaning on the CLI, it is required to escape the backslashes. This is the way to escape the name for the share `\\192.168.1.1\admin\prepop` is `\\\\192.168.1.1\\admin\\prepop`.

The CIFS pre-population process consists of three steps: The registration of the CIFS share, the initial syncing and the periodical syncing.

8.4.1. The registration of the CIFS share

During the registration the CIFS pre-population process will connect to the remote server, login and map the share.

Figure 8.11. Successful registration of the CIFS pre-population session

```
CSH rcud[5555]: [rcud/req/.NOTICE] - {- -} Received action: /rbt/rcu/action/share_config
CSH rcud[5555]: [rcud/main/.INFO] - {- -} Adding share:  Share : \\192.168.1.1\admin\prepo \
    p Server name : 192.168.1.1 Server path : \\192.168.1.1\admin\prepop Server port : 0 S \
    hare Mode : 3 Sync freq : 604800
CSH rcud[5555]: [rcud/main/.INFO] - {- -} Share parameters verified locally. Reply to mgmt \
    d
CSH rcud[5555]: [rcud/main/.INFO] - {- -} succ=0, msg=Share registration in progress ...
CSH rcud[5555]: [policy_manager.INFO] - {- -}  @ [126]: Saved policies
CSH rcud[5555]: [rcud/main/.INFO] - {- -} For Share : \\192.168.1.1\admin\prepop
CSH mgmtd[3544]: [mgmtd.NOTICE]: Share registration in progress ...
CSH webasd[6087]: [web.INFO]: web: Received return code 0, return message 'Share registrat \
    ion in progress ...\n' from gclSession pygs_handle_any_response
CSH rcud[5555]: [rcud/share/.INFO] - {- -} Share \\192.168.1.1\admin\prepop Enter dispatch \
    _cmd(): cmd=REGISTER_SHARE
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} dispatch_cmd() START CMD : REGISTER_SHARE Sha \
    re : \\192.168.1.1\admin\prepop
CSH sport[2946]: [splice/client.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} init client 10.0 \
    .1.5:47055 server 192.168.1.1:139 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connec \
    ted: yes
CSH sport[2946]: [splice/client.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Start flowing, l \
    port 40281, rport 7800, OPOL_NORMAL, NAGLE_NEVER, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_NONE, TPTOPT_NONE(0x0), TRPY_NONE
CSH rcud[5555]: [cifsclient.INFO] - {- -}  @ [320]: Connected to server 192.168.1.1
CSH webasd[6087]: [web.INFO]: web: User admin viewing setupPrepop page.
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[0] = PC NETWORK PROGRAM 1.0
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[1] = MICROSOFT NETWORKS 1.03
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[2] = MICROSOFT NETWORKS 3.0
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[3] = LANMAN1.0
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[4] = LM1.2X002
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[5] = DOS LANMAN2.1
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[6] = NT LANMAN 1.0
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[7] = NT LM 0.12
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[8] = Samba
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect[9] = RCU SMB 1
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} rcu dialect index = 9
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect in response before: 6
CSH sport[2946]: [cifs_neg.INFO] 14 {- -} Dialect in response after: 9
CSH sport[2946]: [smbmux_cfe/scons.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Creating pars \
    er type: SMB_PARSER
CSH sport[2946]: [smbmux_cfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Creating client-sid \
    e SMB parser
CSH sport[2946]: [smbcfe.INFO] 14 {- -} RTT time: sec:0, usec:305000
CSH sport[2946]: [smbcfe.INFO] 14 {- -} MAX metadata cache time: 2.000000 sec, data cache  \
    time 20.000000 sec
CSH sport[2946]: [smbcfe.INFO] 14 {- -} Idle FOI timeouts : 15250 No oplock timeout : 7625 \

CSH sport[2946]: [smbcfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Client Native OS Versio \
    n: bytes:  52 69 4f 53                                      RiOS
CSH sport[2946]: [smbcfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Client Native LAN Manag \
    er: bytes  52 42 54 2d 43 69 66 73   43 6c 69 65 6e 74       RBT-Cifs  Client
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH sport[2946]: [smbcfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Client Native OS Versio \
    n: bytes:  52 69 4f 53                                      RiOS
CSH sport[2946]: [smbcfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Client Native LAN Manag \
    er: bytes  52 42 54 2d 43 69 66 73   43 6c 69 65 6e 74       RBT-Cifs  Client
CSH sport[2946]: [smbcfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Server Native OS Versio \
    n: unix: Unix
CSH sport[2946]: [smbcfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Server Native LAN Manag \
    er: Samba: Samba
CSH sport[2946]: [smbsign_cfe.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} Shutting down sign \
    ing blade for this connection. Reason - Signing feature disabled on SFE.
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH sport[2946]: [smbcfe/ccons.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x52d2b30, qlen=0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed successfully  CMD : REGISTER_SHARE  \
    Share : \\192.168.1.1\admin\prepop
CSH sport[2946]: [smbcfe/scons.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x52d2c08, qlen=0
CSH sport[2946]: [splice/client.INFO] 14 {10.0.1.5:47055 192.168.1.1:139} fini client 10.0 \
    .1.5:47055 server 192.168.1.1:139 cfe 10.0.1.6:40281 sfe 192.168.1.6:7800 app CIFS
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share Registered
```

It shows up as a normal CIFS session on TCP port 139 or 445 and it should complete with a successful completion: `Completed successfully CMD : REGISTER_SHARE Share : \\192.168.1.1\admin\prepop`.

8.4.2. Failure of a registration

8.4.2.1. Invalid hostname in the remote share

The host portion of the remote share can be entered by IP addresses or by hostname, for example *\\192.168.1.1\admin* or *\\share.example.com\admin*.

If the hostname cannot be resolved, the logs show the error `Host not found (authoritative)`:

Figure 8.12. Registration failed due to an unknown hostname

```
CSH mgmtd[3544]: [mgmtd.NOTICE]: Share registration in progress ...
CSH webasd[6087]: [web.INFO]: web: Received return code 0, return message 'Share registrat \
    ion in progress ...\n' from gclSession pygs_handle_any_response
CSH rcud[5555]: [rcud/share/.INFO] - {- -} Share \\foo.example.com\test Enter dispatch_cmd \
    (): cmd=REGISTER_SHARE
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} dispatch_cmd() START CMD : REGISTER_SHARE Sha \
    re : \\foo.example.com\test
CSH webasd[6087]: [web.INFO]: web: User admin viewing setupPrepop page.
CSH rcud[5555]: [cifsclient.NOTICE] - {- -}  @ [233]: Host not found (authoritative)
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} kerberos authentication on port 139 Failed,  \
    trying NTLMSSP...
CSH rcud[5555]: [cifsclient.NOTICE] - {- -}  @ [233]: Host not found (authoritative)
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Logon to port: 139 Failed:
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Retrying alternate CIFS port...
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH rcud[5555]: [cifsclient.NOTICE] - {- -}  @ [233]: Host not found (authoritative)
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} kerberos authentication on port 445 Failed,  \
    trying NTLMSSP...
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH rcud[5555]: [cifsclient.NOTICE] - {- -}  @ [233]: Host not found (authoritative)
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Network error. Success, error no : 0 => erro \
    r_info.h:235
CSH rcud[5555]: [rcud/client/.WARN] - {- -} run_rbcp_cmd_v3 : Connecting to server failed. \
      Server : foo.example.com Share : \\foo.example.com\test
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} record_start_to_status_file(): Received unkno \
    wn command: 0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -}  got unknown cmd:0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed with Error   CMD : REGISTER_SHARE S \
    hare : \\foo.example.com\test
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share has error
```

8.4.2.2. Remote host doesn't respond

If the host doesn't exist or doesn't respond, the logs show the error: `Network error. Connection reset by peer, error no : 104`:

Figure 8.13. Registration failed due to an non-responding remote server

```
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH last message repeated 2 times
CSH sport[2946]: [segstore/peertable.INFO] - {- -} Attempting to write stat table to offse \
    t 1. Num pages: 9
CSH sport[2946]: [segstore/peertable.INFO] - {- -} Write stat table complete, error 0
CSH kernel: [intercept.WARN] Detected SYN retransmits for a PFSv3 connection.  Potential r \
    easons include the remote host or network being down, probe filtering, and asymmetric  \
    routing.
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH kernel: [intercept.WARN] Detected SYN retransmits for a PFSv3 connection.  Potential r \
    easons include the remote host or network being down, probe filtering, and asymmetric  \
    routing.
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH last message repeated 3 times
CSH rcud[5555]: [cifsclient.INFO] - {- -}  @ [320]: Connected to server 192.168.1.2
CSH rcud[5555]: [cifsclient.NOTICE] - {- -}  @ [53]: Send failed
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Logon to port: 139 Failed:
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Retrying alternate CIFS port...
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH kernel: [intercept.WARN] Detected SYN retransmits for a PFSv3 connection.  Potential r \
    easons include the remote host or network being down, probe filtering, and asymmetric  \
    routing.
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH kernel: [intercept.WARN] Detected SYN retransmits for a PFSv3 connection.  Potential r \
    easons include the remote host or network being down, probe filtering, and asymmetric  \
    routing.
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH last message repeated 4 times
CSH rcud[5555]: [cifsclient.INFO] - {- -}  @ [320]: Connected to server 192.168.1.2
CSH rcud[5555]: [cifsclient.NOTICE] - {- -}  @ [53]: Send failed
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Network error. Connection reset by peer, err \
    or no : 104
CSH rcud[5555]: [rcud/client/.WARN] - {- -} run_rbcp_cmd_v3 : Connecting to server failed. \
      Server : 192.168.1.2 Share : \\192.168.1.2\admin\prepop
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} record_start_to_status_file(): Received unkno \
    wn command: 0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -}  got unknown cmd:0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed with Error   CMD : REGISTER_SHARE S \
    hare : \\192.168.1.2\admin\prepop
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share has error
```

8.4.2.3. Authentication failed

If the username or password provided is incorrect, the logs will show `Session setup failed. Status code : NT_STATUS_LOGON_FAILURE`:

Figure 8.14. Registration failed due to a login failure

```
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Session setup failed. Status code : NT_STATU \
    S_LOGON_FAILURE => ntlm_session_impl.cc:131
CSH rcud[5555]: [rcud/client/.WARN] - {- -} run_rbcp_cmd_v3 : Connecting to server failed. \
      Server : 192.168.1.1 Share : \\192.168.1.1\prepop
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed with Error   CMD : REGISTER_SHARE S \
    hare : \\192.168.1.1\prepop
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share has error
```

8.4.2.4. Invalid path on remote server

If the path on the remote server does not exist, the logs will show `Failed to map share, because network path is unavailable`:

Figure 8.15. Registration failed due to a login failure

```
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Registration in progress...
CSH rcud[5555]: [rcud/client/.NOTICE] - {- -} Could not connect to share \\192.168.1.1\pre \
    pop. Tree connect failed. Status code : Failed to map share, because network path is u \
    navailable => tree_connect_impl.cc:90
CSH rcud[5555]: [rcud/client/.WARN] - {- -} run_rbcp_cmd_v3 : Connecting to server failed. \
      Server : 192.168.1.1 Share : \\192.168.1.1\prepop
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} record_start_to_status_file(): Received unkno \
    wn command: 0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -}  got unknown cmd:0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed with Error   CMD : REGISTER_SHARE S \
    hare : \\192.168.1.1\prepop
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share has error
```

8.4.3. The syncing of data

When the registration has completed successfully, the syncing needs to be enabled. From the CLI this can be done with the command `prepop share modify remote-path <share> syncing true`.

In this example, the remote share has two files in it: *\foo.doc* and *\bar\bar.pdf*.

Figure 8.16. A successful sync

```
CSH rcud[5555]: [crawler.INFO] - {- -} Get initial copy started at Fri 2014-Jan-03 21:29:0 \
    1
CSH rcud[5555]: [cifsfs.INFO] - {- -}  @ [199]: Base cifs sync directory = '\prepop'
CSH sport[2946]: [smbcfe/file.INFO] 17 {10.0.1.5:47717 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH sport[2946]: [smbcfe/file.INFO] 17 {10.0.1.5:47717 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \foo.doc
CSH last message repeated 4 times
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Get initial copy in progress s \
    ince Fri Jan  3 21:28:59 2014
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \foo.doc
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \bar\bar.pdf
CSH last message repeated 4 times
CSH rcud[5555]: [rcud/req/.NOTICE] - {- -} Received action: /rbt/rcu/action/share_config
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -}  Syncing changed for Share : \\192.168.1.1\ad \
    min\prepop Old syncing : 1 New syncing: 0
```

```
CSH rcud[5555]: [rcud/main/.INFO] - {- -} succ=0, msg=Share modify succeeded
CSH mgmtd[3544]: [mgmtd.NOTICE]: Share modify succeeded
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \bar\bar.pdf
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Get initial copy in progress s \
    ince Fri Jan  3 21:28:59 2014
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Get initial copy in progress s \
    ince Fri Jan  3 21:28:59 2014
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \bar\bar.pdf
CSH rcud[5555]: [crawler/sync_lists.INFO] - {- -}  @ [194]: Synced file : \foo.doc
CSH rcud[5555]: [crawler/sync_lists.INFO] - {- -}  @ [194]: Synced file : \bar\bar.pdf
CSH sport[2946]: [smbcfe/file.NOTICE] 17 {10.0.1.5:47717 192.168.1.1:139}  File::add_trans \
    2_findnext_data() last entries dont match : \
CSH rcud[5555]: [crawler.INFO] - {- -} Get initial copy completed successfully at Fri 2014 \
    -Jan-03 21:29:14
CSH rcud[5555]: [crawler.INFO] - {- -} Received 2 files, 0 objects have error. (34125972 b \
    ytes were received in 13.7194 seconds at 19.8993 Mbps)
CSH sport[2946]: [smbcfe/ccons.INFO] 17 {10.0.1.5:47717 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x52f0130, qlen=0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed successfully  CMD : GET_INITIAL_COP \
    Y Share : \\192.168.1.1\admin\prepop
CSH sport[2946]: [smbcfe/scons.INFO] 17 {10.0.1.5:47717 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x52f0208, qlen=0
CSH sport[2946]: [splice/client.INFO] 17 {10.0.1.5:47717 192.168.1.1:139} fini client 10.0 \
    .1.5:47717 server 192.168.1.1:139 cfe 10.0.1.6:40284 sfe 192.168.1.6:7800 app CIFS
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share idle
```

The *Initial Copy* log file shows it as:

Figure 8.17. Log file of the Initial Copy

```
Get initial copy started at Fri 2014-Jan-03 21:29:01
Synced file : \foo.doc
Synced file : \bar\bar.pdf
Get initial copy completed successfully at Fri 2014-Jan-03 21:29:14
Received 2 files, 0 objects have error. (34125972 bytes were received in 13.7194 seconds a \
    t 19.8993 Mbps)
```

8.4.4. Failure of syncing of data

8.4.4.1. Client-side Steelhead doesn't see the CIFS session

In previous RiOS versions, if the client-side Steelhead appliance did not see the CIFS pre-population session, the CIFS pre-population process would warn about this and abort.

With the new implementation this doesn't happen anymore. The best way to check is to initiate a manual sync and confirm that the TCP session is optimized. This can be done with the command `prepop share manual-sync remote-path <path>`

Figure 8.18. Manually starting a CIFS pre-population sync

```
CSH (config) # prepop share manual-sync remote-path \\\\192.168.1.1\\admin\\prepop
Share manual sync in progress...
```

This can be caused by:

- The cable of the primary interface is attached to the WAN router instead of to the LAN switch. If this is on design, consider connecting the auxiliary interface to the LAN switch and a routing statement to force traffic to the CIFS servers to go via the gateway on the auxiliary interface.
- In case of a parallel cluster, the traffic from the LAN switch does not go via this Steelhead appliance. The way around this could be to configure the data store synchronization and to enable the CIFS pre-population on the Steelhead appliance through which the naked SYN normally goes.
- In-path rules on the Steelhead appliance prevent optimization from this traffic. The way around this would be to allow optimization from the Steelhead appliance primary interface to the CIFS servers.

8.4.4.2. Unable to read the files

When the CIFS pre-population process cannot read a certain file or directory, it will mark it as failed.

In this example, the file *bar\bar.pdf* is not readable by the specified user.

Figure 8.19. Unable to read some of the files

```
CSH rcud[5555]: [crawler.INFO] - {- -} Get initial copy started at Fri 2014-Jan-03 21:23:5 \
    7
CSH rcud[5555]: [cifsfs.INFO] - {- -}  @ [199]: Base cifs sync directory = '\prepop'
CSH sport[2946]: [smbcfe/file.INFO] 16 {10.0.1.5:47574 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH sport[2946]: [smbcfe/file.INFO] 16 {10.0.1.5:47574 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH rcud[5555]: [sync_obj.ERR] - {- -}  @ [59]: Nt Create AndX failed. Status code : NT_ST \
    ATUS_ACCESS_DENIED => nt_create_andx_impl.cc:94. For \bar\bar.pdf
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \foo.doc
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Get initial copy in progress s \
    ince Fri Jan  3 21:23:55 2014
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \foo.doc
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Get initial copy in progress s \
    ince Fri Jan  3 21:23:55 2014
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \foo.doc
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \foo.doc
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \bar\bar.pdf
CSH rcud[5555]: [crawler/sync_lists.INFO] - {- -}  @ [194]: Synced file : \foo.doc
CSH rcud[5555]: [crawler/sync_lists.INFO] - {- -}  @ [194]: Synced file : \bar\bar.pdf
CSH rcud[5555]: [rcud/client/.ERR] - {- -} Sync error.
CSH rcud[5555]: [rcud/client/.ERR] - {- -} Failed to sync share \\192.168.1.1\admin.
CSH rcud[5555]: [crawler.INFO] - {- -} Get initial copy completed with error.
CSH rcud[5555]: [crawler.INFO] - {- -} Received 1 files, 1 objects have error. (17062986 b \
    ytes were received in 11.5794 seconds at 11.7885 Mbps)
CSH sport[2946]: [smbcfe.INFO] 16 {10.0.1.5:47574 192.168.1.1:139} clearing pending req:   \
    uid: 0x64 tid: 0x1 pid: 0x15b3 mid: 0x11f omid: 0x11f smid: 0x11f cmd: close subcmd: 0 \
     infolvl: 0 fid: 0x1cb5 status: 1 len 45 time: 1388744649.518695 sec
CSH sport[2946]: [smbcfe/ccons.INFO] 16 {10.0.1.5:47574 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x52f1530, qlen=0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed with Error   CMD : GET_INITIAL_COPY \
     Share : \\192.168.1.1\admin\prepop
```

The Initial Copy log shows:

Figure 8.20. Initial copy failed due to reading errors

```
Get initial copy started at Fri 2014-Jan-03 21:23:57
Synced file : \foo.doc
Error in syncing file \bar\bar.pdf
Get initial copy completed with error.
Received 1 files, 1 objects have error. (17062986 bytes were received in 11.5794 seconds a \
    t 11.7885 Mbps)
```

To prevent the failure of one or two files which couldn't be synchronized to be seen as a full failure of the Initial Sync, a failure ignore percentage can be configured:

Figure 8.21. CIFS pre-population failure settings

```
CSH (config) # show prepop settings
Maximum Mpx Count Percentage:    40
Ignore Percentage:    0
CSH (config) # prepop settings ignore-pct 10
Processed RCU configuration command.
CSH (config) # show prepop settings
Maximum Mpx Count Percentage:    40
Ignore Percentage:    10
```

Now up to 10% of the files to be synced can fail and the initial sync is still considered to be successful.

8.4.5. Periodical syncing

After the initial syncing, the periodical syncing checks for new files and updated files.

Figure 8.22. No new files found during the periodical sync

```
CSH rcud[5555]: [crawler.INFO] - {- -} Sync started at Fri 2014-Jan-03 21:33:15
CSH rcud[5555]: [cifsfs.INFO] - {- -}  @ [199]: Base cifs sync directory = '\prepop'
CSH sport[2946]: [smbcfe/file.INFO] 19 {10.0.1.5:47840 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH sport[2946]: [smbcfe/file.INFO] 19 {10.0.1.5:47840 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH sport[2946]: [smbcfe/file.NOTICE] 19 {10.0.1.5:47840 192.168.1.1:139}  File::add_trans \
    2_findnext_data() last entries dont match : \
CSH rcud[5555]: [crawler.INFO] - {- -} Sync completed successfully at Fri 2014-Jan-03 21:3 \
    3:16
CSH rcud[5555]: [crawler.INFO] - {- -} Received 0 files, 0 objects have error. (0 bytes we \
    re received in 1.26024 seconds at 0 Mbps)
CSH sport[2946]: [smbcfe/ccons.INFO] 19 {10.0.1.5:47840 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x5494530, qlen=0
CSH rcud[5555]: [rcud/share/.NOTICE] - {- -} Completed successfully  CMD : START_SYNC Shar \
    e : \\192.168.1.1\admin\prepop
CSH sport[2946]: [smbcfe/scons.INFO] 19 {10.0.1.5:47840 192.168.1.1:139} SMBConsumer::eof( \
    ) this=0x5494608, qlen=0
CSH sport[2946]: [splice/client.INFO] 19 {10.0.1.5:47840 192.168.1.1:139} fini client 10.0 \
    .1.5:47840 server 192.168.1.1:139 cfe 10.0.1.6:40286 sfe 192.168.1.6:7800 app CIFS
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : Share idle
```

The *Last Sync* log files show as:

Figure 8.23. The Last Sync log files shows no new files found

```
Sync started at Fri 2014-Jan-03 21:33:15
Sync completed successfully at Fri 2014-Jan-03 21:33:16
Received 0 files, 0 objects have error. (0 bytes were received in 1.26024 seconds at 0 Mbp \
    s)
```

If the timestamp of a remote file has changed or a new file has been detected, it will be transferred:

Figure 8.24. New files found during the periodical sync

```
CSH rcud[5555]: [crawler.INFO] - {- -} Sync started at Fri 2014-Jan-03 21:34:15
CSH rcud[5555]: [cifsfs.INFO] - {- -}  @ [199]: Base cifs sync directory = '\prepop'
CSH sport[2946]: [smbcfe/file.INFO] 20 {10.0.1.5:47863 192.168.1.1:139} process_trans2_fin \
    dfirst_response() has search attr : 23
CSH last message repeated 2 times
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \quux\quux.mbox
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \quux\quux.mbox
CSH rcud[5555]: [rcud/share/.INFO] - {- -}  Status string : START_SYNC in progress since F \
    ri Jan  3 21:34:12 2014
CSH rcud[5555]: [sync_file.INFO] - {- -}  @ [207]: Syncing file \quux\quux.mbox
CSH last message repeated 3 times
CSH rcud[5555]: [crawler/sync_lists.INFO] - {- -}  @ [194]: Synced file : \quux\quux.mbox
CSH sport[2946]: [smbcfe/file.NOTICE] 20 {10.0.1.5:47863 192.168.1.1:139}  File::add_trans \
    2_findnext_data() last entries dont match : \
CSH rcud[5555]: [crawler.INFO] - {- -} Sync completed successfully at Fri 2014-Jan-03 21:3 \
    4:23
CSH rcud[5555]: [crawler.INFO] - {- -} Received 1 files, 0 objects have error. (17062986 b \
    ytes were received in 7.97558 seconds at 17.1152 Mbps)
```

The *Last Sync* log files show as:

Figure 8.25. The Last Sync log files shows new or updated files found

```
Sync started at Fri 2014-Jan-03 21:34:15
Synced file : \quux\quux.mbox
Sync completed successfully at Fri 2014-Jan-03 21:34:23
Received 1 files, 0 objects have error. (17062986 bytes were received in 7.97558 seconds a \
    t 17.1152 Mbps)
```

The status of the CIFS pre-population process can be determined with the command `show prepop stats shares`:

Figure 8.26. Status of the pre-population shares during syncing and when the share is idle

```
CSH # show prepop stats shares
+=============================
| Prepopulation name: '\\192.168.1.1\admin\prepop'
|
| ----- Statistics -----
|   Receiving                : NONE
|   Files Received           : 1
|   Directories Received     : 0
|   Bytes Received           : 17062986
CSH # show prepop stats shares
+=============================
| Prepopulation name: '\\192.168.1.1\admin\prepop'
|
| ----- Statistics -----
|   No action in progress
```

8.5. NFS Latency Optimization

NFS latency optimization works in the following ways:

- Directory index pre-fetching.
- File read-ahead and write-behind.

NFS latency optimization does not work if:

- The NFS version spoken is not NFS version 3.

Figure 8.27. NFS optimization does not work for NFS version 4

```
SH sport[24148]: [SunRpc-CFE/SunRpcParser.WARN] - {10.0.1.1:1023 192.168.1.1:2049} NFSv4 \
   traffic detected between client 10.0.1.1 and server 192.168.1.1
SH sport[24148]: [SunRpc-CFE/SunRpcParser.WARN] - {10.0.1.1:1022 192.168.1.1:2049} NFSv4 \
   traffic detected between client 10.0.1.1 and server 192.168.1.1
statsd[23848]: [statsd.NOTICE]: Alarm triggered for rising error for event nfs_v2_v4
```

- If the NFS Authentication method is set to Kerberos authentication.

Figure 8.28. NFS optimization does not work when using Kerberos authentication

```
SH sport[24148]: [SunRpc-CFE/NfsV3Parser.NOTICE] - {10.0.1.1:1023 192.168.1.1:2049} SunR \
  pcParser: Client Call has an authenticationflavor which we do not support. Shutting La \
  tencyOptimization for this connection.
```

NFS Funkiness:

- There is a 15 second directory meta-data timer[SOURCE: NFS standard] which can show files which have been removed or not show files which have been created on other machines. This is NFS behaviour, not Steelhead appliance specific behaviour.

8.6. MAPI Latency Optimization

MAPI latency optimization works in the following ways:

- Attachment read-ahead and write-behind.
- MAPI pre-population for attachment warming.

MAPI latency optimization does not work if:

- MAPI traffic is encrypted and the Active Directory integration has not been setup or is temporary suspended.

8.6.1. Setup of an optimized MAPI session

The Outlook client sets up a TCP session on port 135 to the Exchange servers port mapper to find out on which TCP port the Exchange service is running. The client-side Steelhead appliance shows this session as EPM, Exchange Port Mapper. The server-side Steelhead appliance will receive the answer from the EPM service, rewrite the TCP port to port 7830 and forwards the answer to the client-side Steelhead appliance. Then the client-side Steelhead appliance will tell the Outlook client to use port 7830 and sets up a hidden in-path rule which states that all traffic towards the Exchange server on TCP port 7830 has to go to the current server-side Steelhead appliance.

The Outlook client will setup three TCP sessions to the Exchange service on port 7830, they are intercepted by the client-side Steelhead appliance and forwarded to the configured server-side Steelhead appliance. There they are forwarded to the Exchange server on the TCP port received in the answer from the Exchange Port Mapper.

By default three TCP sessions are setup, one will be added for every shared calendar.

The hidden in-path rule is for all new TCP sessions towards the Exchange server and is parsed before the normal in-path rules. To disable this in-path rule, use the CLI command `in-path probe-mapi-data` command. Use the command `show in-path probe-mapi-data` to see the current setting of this feature.

Figure 8.29. See the MAPI probing option

```
CSH (config) # in-path probe-mapi-data
You must restart the optimization service for your changes to take effect.
CSH (config) # show in-path probe-mapi-data
Probe MAPI connections to learn VLAN info: yes
```

8.7. Windows Active Directory integration

Integration into the Windows Active Directory infrastructure is needed to seamlessly integrate in the optimization of encrypted MAPI and signed CIFS sessions.

8.7.1. The prerequisites for Active Directory integration

Before the Steelhead appliances can do this integration, they need to be joined to the Active Directory domain. Before this can happen you need to take care of several things:

- The primary interface of the Steelhead appliance should be configured and have access to the network where the Domain Controllers are in.
- The hostname configured on the Steelhead appliance should be 15 characters or less.
- The hostname configured on the Steelhead appliance should not yet exist in the Active Directory configuration.
- One of the domain names configured on the Steelhead appliance should contain the Active Directory domain name.
- The DNS servers configured in the Steelhead appliance should be able to resolve the Active Directory domain name.
- In DNS, the IP address of the primary interface of the Steelhead appliance needs to be available as a PTR record:

Figure 8.30. 192.168.1.6 points to ssh-primary.example.org

```
[~] edwin@t43>dig -x 192.168.1.6

; <<>> DiG 9.8.1-P1 <<>> -x 192.168.1.6
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18322
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;6.1.168.192.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
6.1.168.192.in-addr.arpa. 3583  IN      PTR     ssh-primary.example.org.

;; AUTHORITY SECTION:
1.168.192.in-addr.arpa.  3583   IN      NS      ns0.example.org.

;; ADDITIONAL SECTION:
ns0.example.org.        3583    IN      A       192.168.1.1

;; Query time: 12 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Fri Oct 26 20:13:39 2012
;; MSG SIZE  rcvd: 142
```

- The clock on the Steelhead appliances needs to be in sync with the clock of the AD Domain Controller. If the Domain Controllers are synchronized against the same NTP server as the Steelhead appliance, then this will be fine. If the Domain Controllers are not synchronized against NTP servers, it will be best to configure the NTP servers on the Steelhead appliances to the Domain Controller:

Figure 8.31. NTP servers configured are the local Domain Controllers

```
ntp enable
ntp server 192.168.1.1 enable
ntp server 192.168.1.1 version "4"
```

- DNS should contain the LDAP SRV records for the domain to join. So if the domain is example.com, the following DNS SRV records should exist: _ldap._tcp.example.org.

Figure 8.32. DNS SRV records for _ldap._tcp.example.org

```
[~] edwin@t43>dig _ldap._tcp.example.org srv
; <<>> DiG 9.8.1-P1 <<>> _ldap._tcp.example.org srv
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5339
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; QUESTION SECTION:
;_ldap._tcp.example.org.         IN      SRV

;; ANSWER SECTION:
_ldap._tcp.example.org.      600 IN      SRV     0 100 389 dc.example.org.

;; ADDITIONAL SECTION:
dc.example.org.             3600 IN      A       192.168.1.1

;; Query time: 719 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Fri Oct 26 20:59:16 2012
;; MSG SIZE  rcvd: 2105
```

This shows that the domain controllers for the Active Directory domain example.org can be found at dc.example.org.

• Access to an Active Directory account with domain join privileges.

8.7.1.1. Possible issues during the domain join

Here are several examples of failures of the domain join.

8.7.1.1.1. Clock skew issues

If the clocks on the Steelhead appliance and on the domain controller are too far apart, the domain join will fail with the following error:

Figure 8.33. Domain join failed because of a clock skew between the Steelhead appliance and the Domain Controller

```
mgmtd[4232]: [mgmtd.NOTICE]: Join domain in progress...
rcud[5610]: [rcud/main/.INFO] - {- -} Waiting for join domain to finish ...
rcud[5610]: [rcud/main/.ERR] - {- -} Join domain failed. Failed to join domain: Error: Uns \
    pecified GSS failure. Minor code may provide more information : Clock skew too great
mgmtd[4232]: [mgmtd.ERR]: Domain configuration failed: 1 Join failed
```

8.7.1.1.2. DNS related issues

If the Active Directory domain cannot be found in DNS, the domain join will fail with the following error:

Figure 8.34. Domain join failed because of DNS related issues

```
mgmtd[4232]: [mgmtd.NOTICE]: Join domain in progress...
rcud[5610]: [rcud/main/.INFO] - {- -} Waiting for join domain to finish ...
rcud[4637]: [rcud/main/.ERR] - {- -} Failed to join domain: Error: Operations error. Possi \
    ble DNS misconfiguration.
mgmtd[4232]: [mgmtd.ERR]: Domain configuration failed: 1 Join failed
```

8.7.1.1.3. Active Directory object already exists

If the object for the Steelhead appliance already exist in Active Directory and the joining account does not have privileges to replace old objects, the domain join will fail with the following error:

Figure 8.35. Domain join failed because the object in Active Directory already exists

```
mgmtd[4232]: [mgmtd.NOTICE]: Join domain in progress...
rcud[5610]: [rcud/main/.INFO] - {- -} Waiting for join domain to finish ...
rcud[14701]: [rcud/main/.ERR] - {- -} Failed to join domain: Failed to set account flags f \
    or machine account (NT_STATUS_ACCESS_DENIED)
mgmtd[4232]: [mgmtd.ERR]: Domain configuration failed: 1 Join failed
```

8.8. MS-SQL Latency Optimization

The optimization of the MS-SQL protocol is only supports the MS-SQL 2000 protocol. The reason for this is because the encrypted login session for later versions of the MS-SQL protocol is not decodable and thus is the optimization service not able to identify the database to connect to.

Before enabling this feature please contact Riverbed Professional Services for an investigation in the improvements of this data.

Since RiOS 8.5 the MS-SQL latency optimization feature has been removed from the GUI.

8.9. FTP Latency Optimization

There is not really FTP latency optimization happening, the only reason it gets intercepted and processed is to get the FTP Data channel statistics correct and to get the synchronization between the data channel and command channel correctly.

8.9.1. How it works

In the lifetime of the FTP Command Channel, the Steelhead appliance checks for the PORT commands to see which TCP sessions are used for the FTP Data Channel. Once known, the server-side Steelhead appliance (for active FTP) or the client-side Steelhead appliance (for passive FTP) will setup a hidden fixed target in-path rule so that the traffic goes through the two Steelhead appliances and gets identified as FTP Data traffic.

From RiOS version 6.1 this hidden in-path rule does not get created by default anymore.

8.9.2. Data channel and command channel synchronization

According to FTP standard, the transfer of a file goes as follows:

- The client sends a PORT command over the command channel which informs the server to which IP address and TCP port it should connect to.
- The server sets up a TCP session for the data channel, transfers the data over it and closes the data channel.
- The server sends a 226 response code, which means that the file has been transferred.
- The client knows now that the whole file has been transferred.

With a Steelhead appliance in between, the server-side Steelhead appliance might still be encoding the data from the server when the server closes the data channel and sends the 226 response code to the client. That early 226 response code might confuse the client and makes it close the data channel while it is still receiving the data.

8.9.3. Where things can go wrong

8.9.3.1. FTP Data Channel not recognized as such

If Enhanced Auto Discovery is not enabled and there are multiple Steelhead appliances in a serial deployment at the FTP server side, it is possible that the Steelhead appliance with the termination of the optimized FTP Data channel is not the one which terminates the optimized FTP Command channel. That will prevent the traffic statistics to be correctly identified as FTP data.

Figure 8.36. FTP Data reported as unknown protocols

```
!! Graphics file XXX.png not found
XXX Lots of unknown data which should have been FTP Data
```

8.10. HTTP Latency Optimization

HTTP latency optimization works in various ways:

- HTTP meta-data caching.
- HTTP object prefetching.

HTTP latency optimization doesn't work when:

- When the HTTP request method is not POST nor GET, for example WEBDAV related requests.
- When the format of the requested URL or the referrer field is invalid.
- When there are multiple HTTP requests in a single TCP session and one of them fails, the rest of the HTTP requests won't be optimized.

8.10.1. How HTTP Latency Optimization work

8.10.1.1. HTTP Meta-Data Caching

A HTTP Reply header can contain the expected life-time of an object:

Figure 8.37. An HTTP Reply header

```
HTTP/1.0 200 OK
Content-Type: image/png
Last-Modified: Fri, 15 Jul 2011 23:21:33 GMT
Date: Mon, 14 Nov 2011 21:41:50 GMT
Expires: Mon, 15 Nov 2011 21:41:50 GMT
Cache-Control: public, max-age=86400
X-Content-Type-Options: nosniff
Server: example.mavetju.org
Content-Length: 35636
X-XSS-Protection: 1; mode=block
```

The following meta-data can be seen on this response:

- The modification time of the object is *Fri, 15 Jul 2011 23:21:33 GMT*.
- The date on the server is *Mon, 14 Nov 2011 21:41:50 GMT*.
- The object can be used in public caches until *Mon, 15 Nov 2011 21:41:50 GMT*. This is an old notation style.
- The object can be used in public caches for *86400 seconds* (or 1 day). This is the new notation style.

There are several ways for a web-browser to get an object:

- From its browser cache. From the last time the object was requested, the browser was told that it could use it for 86400 seconds.
- After the 86400 seconds, the object in the browser-cache has become stale and the web-browser will ask the web-server to send the object again if it has been changed since its *Last-Modified* date. If it has been changed, the web server will send the new object. If it has not been changed, it will respond with a HTTP 304 response and a new *Cache-Control* line which states that the stale object can be used for another 86400 seconds. This is a low-bandwidth browser-cache mechanism for refreshing objects.

  This answer can either go all the way to the web-server or can be answered by a local web-cache or a server-side web-cache.

  Figure 8.38. A 304 response code

  ```
  GET /foo.png HTTP/1.1
  Host: example.mavetju.org
  User-Agent: Mozilla/5.0 (X11; FreeBSD i386; rv:22.0) Gecko/20100101 Firefox/22.0
  Referer: http://example.mavetju.org/
  Connection: keep-alive
  If-Modified-Since: Mon, 10 Mar 2008 18:47:44 GMT

  HTTP/1.1 304 Not Modified
  Date: Sun, 29 Sep 2013 17:06:08 GMT
  Server: Apache/2.2.3 (FreeBSD)
  Connection: close
  Expires: Sun, 29 Sep 2013 21:08:31 GMT
  Cache-Control: public, max-age=86400
  ```

- The web-browser can ask the web-server to send the object forcing it to come from the web-server without allowing it to be served from any caches.

Figure 8.39. Network with Web-browsers, web-caches and web-servers

```
browser -- SH -- WAN -- SH -- cache -- WAN -- SH -- server cache -- server
```

When the web-browser asks for an object and the Steelhead appliance optimizes the HTTP session, it can keep track of the meta-data. If the client asks for an object refresh via an *If-Modified-Since* and the Steelhead appliance has already learned from a request via another client that the object hasn't changed and a new expiration time has been given, the Steelhead appliance can answer the web-browser that the object is still valid and provide a new expiration time. That doesn't save only bandwidth but also reduces the round trip time.

8.10.1.2. HTTP Object Prefetching

If the object requested by the web-server is of the type text/html, it can contain embedded objects like images, style sheet and JavaScript files.

If the Steelhead appliance parses the returned text/html object and fetches the embedded objects from the web-server before the web-browser has a chance to request them, then the Steelhead appliance can serve the objects to the web-browser locally instead of the web-browser having to go all the way to the web-server for it.

8.10.2. Troubleshooting HTTP latency optimization

8.10.2.1. Response of the first request takes too long

Unlike proxy servers where the TCP session is terminated on the proxy server, an optimized TCP session is setup to the remote server first before the latency optimization kicks in. Thus the first request will take longer to setup, especially if the Full Transparency WAN visibility is used for the inner channel.

8.10.2.2. Invalid URL or referrer field

The most common issue is that a requested URL or given referrer contains a space. While technically this is a violation of the syntax, the space should be replaced by a %20, a lot of scripts or web-browsers send it, while web-servers and proxies have been taught to deal with the issue.

Figure 8.40. An invalid URL error messages due to spaces in the URL

```
CSH sport[24250]: [http/client.INFO] 3277788 {10.0.1.1:24750 192.168.1.1:80} Request parse \
     error, code = HTTP_ERR_HEADER_LINE GET /foo /bar.png HTTP:/1.1\013\010Referrer: http: \
    //example.mavetju.org/\013\010Host: example.mavetju.org\013\010
```

By default, the Steelhead appliances will throw an error and not perform HTTP latency on this TCP session anymore, but with the command `protocol http space-in-uri enable` it will allow this and process the request anyway.

Figure 8.41. Output of the command "show protocol http"

```
CSH (config) # protocol http space-in-uri enable
You must restart the optimization service for your changes to take effect.
CSH (config) # show protocol http internal
[..]
Allow Parsing Space in URI:          yes
```

8.10.2.3. What kind of HTTP latency optimization is performed?

When performing the HTTP latency optimization, the client-side Steelhead will add a line in the HTTP response header:

Figure 8.42. X-RBT-Optimized header

```
X-RBT-Optimized-By: CSH (RiOS 6.1.5b) PT
```

The first part of the string is the name of the client-side Steelhead performing the latency optimization, the middle part contains the RiOS version running on the Steelhead appliance and the third part contains the optimization scheme code.

Table 8.1. HTTP Optimization Scheme Code

UL	URL Learning	The optimization service learns the order of URLs being requested for a certain base URL and will request the next ones once a base URL gets requested
PP	Parse-and-Prefetch	The optimization service is parsing the return HTML code for objects it can request in advance
PT	Object Prefetch Table (OPT)	Response is served from the local Steelhead HTTP cache
MC	Metadata Response	
RA	Reuse-Auth	
FN	Force-NTLM	
SA	Strip-Auth-hdr	
GR	Gratuitous-401	The client-side Steelhead appliance replied with a 401 or 403
SC	Strip-Compress	The header field "Accept-Encoding: gzip, deflate" has been removed
IC	Insert-Cookie	
IK	Insert-Keep-alive	Despite that the client didn't request it, the optimization service inserted a "Connection: Keep-Alive" in the request
IF	Inflight-Cache (Stream Splitting)	

8.10.2.4. What kind of internal decisions for HTTP optimization are made?

Further information about the internals of the HTTP optimization can be enabled with the command `protocol http x-debug enable`.

Figure 8.43. Enabling the HTTP debugging feature

```
CSH (config) # protocol http x-debug enable
CSH (config) # show protocol http internal
Insert Pre-Fetch X-Header:                    yes
Insert X-RBT-Debug header:                    yes
[...]
```

In the HTTP response, this will add a new line: `X-RBT-Debug`.

Figure 8.44. The X-RBT-Debug line

```
X-RBT-Debug: 0x81,SPIP,FPSF
```

The first field, *0x81*, is the optimization scheme and its value represents a bitmask which explains which features are enabled for the request:

Table 8.2. Optimization scheme bit mask

0x0001	URL Learning
0x0002	Parse and Prefetch

0x0004	Object Prefetch Table
0x0008	Reuse NTLM Authentication
0x0010	Force Negotation NTLM
0x0020	Strip authentication header
0x0040	Gratuitous 401
0x0080	Strip Compression
0x0100	Insert Cookie
0x0200	Insert keep-alive
0x010000	Frontpage Extensions
0x020000	WebDAV support
0x040000	FSS HTTP

So for the value of 0x81 the features used are *Strip Compression* and *URL Learning*.

This value is also seen in the logs for that TCP session:

Figure 8.45. HTTP Optimization scheme

```
CSH sport[16611]: [http/client.INFO] 6629561 {10.0.1.1:65310 192.168.1.1:80} codec flow co \
    ntrol is disabled
CSH sport[16611]: [http/client.INFO] 6629561 {10.0.1.1:65310 192.168.1.1:80} opt_scheme_ u \
    pdated to 129 from auto config table
```

In this example, the 129 decimal is 0x81 hexadecimal, relates to the same features as seen earlier.

The next fields are the actions and decisions of the HTTP optimization code. The four letters are part of a code:

The first letter is the status:

- **S** from Success
- **F** from Failure
- **I** from Information

The second letter is the category:

- **C** from Cache
- **P** from Prefetch
- **M** from Miscellaneous

The third and the fourth letter are the action and specific reason codes.

So the code *SPIP* would be a success from the prefetching functionality and the code *FPSF* would be a failure for the prefetching functionality.

Table 8.3. Cache related category

FCAA	Failed to cache because the content is authenticated but cache_without_auth is off
FCAE	Failed to cache because the extension is not present in the cache extension list

FCAI	Invalid request, e.g., absence of host header
FCAL	The body is too big to cache
FCAS	The remaining lifetime is too short to cache
FCAU	Not allowed to cache, e.g., cache-control header
FCS2	The current cache entry only has the header
FCSA	Failed to serve due to the absence of Accept-Encoding header
FCSC	The Steelhead appliance is not configured to serve full body cache
FCSD	Failed to serve from cache because of a different encoding
FCSE	Failed to serve from cache because the extension is not present in the cache extension list
FCSI	Invalid request, e.g., absence of host header
FCSM	Miscellaneous reason
FCSN	The object is not present in cache
FCSU	The object is not allowed to cache, e.g., cache-control header
FCSX	Failed to serve cached object because it has been expired
SCAN	The object is added to cache
SCAR	The object has been refreshed in the cache

Table 8.4. Prefetch related category

FPIA	Failed to initiate prefetching because authentication is required. OR, no response has been received through this connection so the Steelhead appliance is not yet aware if authentication is needed or not
FPID	Failed to add the URL to the APT database
FPIE	Didn't initiate prefetching because inner channel is being terminated
FPIH	Failed to initiate prefetching because a header is missing (most likely cookie header)
FPIM	Reached the maximum prefetch limit
FPLF	Failed to learn the object because APT is full
FPLH	Failed to learn the object due to the absence of the header
FPLM	Failed to learn the object because the Steelhead appliance is occupying its MAX MEM
FPSA	Authentication is required
FPSE	APT request was EOF'ed
FPSF	APT was not found for the object
FPSN	APT request was not sent
FPSR	Received error status code for APT request
IPGP	APT is generated via PnP
IPGU	APT is generated via URL Learning
IPLP	The object has been purged from UL APT tree

IPSC	The prefetched object is served through callback, i.e., the request was halted for the APT response to come.
SPIP	This object triggered prefetching
SPLN	The object was learned by UL
SPPP	The page is parsed for PnP

Table 8.5. Miscellaneous category

FM4S	Received a 401 despite that strip authentication took the strip header off
FMGS	Failed to store the 401 response because the body is shorter than specified
IMAA	Expecting an authenticaton header in the request but it was absent
IMAC	Stopped storing the body for processing because the Steelhead appliance is in a LOW MEM state
IMBR	This is a base request
IMIR	The Steelhead appliance inserted a referer header
SMGN	The 401/407 response is newly added
SMGR	The 401/407 response cache entry is refreshed

8.11. SSL Pre-optimization

Unlike the other protocols in this chapter, SSL pre-optimization is a transport layer feature: Instead of the protocol being encapsulated inside the TCP layer, the protocol is being encapsulated inside the SSL layer and then encapsulated inside the TCP layer.

Figure 8.46. SSL encapsulated HTTP traffic

```
                    .---------------.
                    | HTTP protocol |
  .---------------. |---------------|
  | HTTP protocol | | SSL protocol  |
  |---------------| |---------------|
  | TCP protocol  | | TCP protocol  |
  |---------------| |---------------|
  | IP protocol   | | IP protocol   |
  '---------------' '---------------'
```

SSL encapsulation is not limited to HTTP only. It is used for the encryption of many protocols like HTTP, IMAP, POP3 and SMTP.

8.11.1. Configuration on the Steelhead appliances

8.11.1.1. Obtain an SSL license

To be able to optimize SSL encrypted traffic, SSL licenses are required to enable this feature. You can request them for free from the Riverbed Support website.

8.11.1.2. Configuring SSL peering

This is described in the SSL Secure Peering section in the *Operational Related Issues* chapter.

8.11.1.3. Server Certificate and Server Private Key

To be able to optimize traffic towards an SSL server, you will need to obtain the SSL certificate and the SSL private key of the server.

In this scenario, the company Example Dot Com has a Root CA and an intermediate CA. To get this working the server-side Steelhead appliance needs to know about the CA certificate, the intermediate CA certificate and the server private key and certificate.

8.11.1.3.1. Import of the server private key and certificate with the issuer certificate missing.

If the issuer certificate of the server certificate is not known on the Steelhead appliance, then you need to obtain that issuer certificate too.

Figure 8.47. Missing issuer certificate during the import of the server key and certificate

```
webasd[7055]: [web.INFO]: web: User admin viewing setupServiceProtocolsSSLMain page.
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} getting SSL discovered servers information
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} getting SSL bypassed servers information
webasd[7055]: [web.NOTICE]: web: user admin: SSL ACTION: /rbt/sport/ssl/action/server_cert \
    s/add_import; PARAMETERS: name = , exportable = true
mgmtd[3958]: [mgmtd.INFO]: while importing www.example.com: code 20 at 0 depth lookup: una \
    ble to get local issuer certificate
mgmtd[3958]: [mgmtd.INFO]: /C=AU/ST=NSW/L=Sydney/O=Example Dot Com/OU=Webserver Department \
    /CN=www.example.com/emailAddress=www@example.com
mgmtd[3958]: [mgmtd.INFO]: EVENT:  /rbt/sport/ssl/event/change/backend_server
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} dyn config modify/add SSL server certificate [w \
    ww.example.com].
sport[7402]: [sslmodule.INFO] - {- -} Added a certificate with name "www.example.com"
mgmtd[3958]: [mgmtd.NOTICE]: Server certificate "www.example.com" added successfully.
mgmtd[3958]: [mgmtd.INFO]: verification failed: code 20 at 0 depth lookup: unable to get l \
    ocal issuer certificate
mgmtd[3958]: [mgmtd.INFO]: /C=AU/ST=NSW/L=Sydney/O=Example Dot Com/OU=Webserver Department \
    /CN=www.example.com/emailAddress=www@example.com
mgmtd[3958]: [mgmtd.NOTICE]: But the certificate did not pass verification, so if the actu \
    al backend server uses the same certificate, the appliance might not be able to connec \
    t to it. Additional Certificate Authorities might be needed.
mgmtd[3958]: [mgmtd.NOTICE]: To avoid potential verification problems at clients (eg, brow \
    ser pop-up warnings), additional/correct chain certificates might be needed.
webasd[7055]: [web.INFO]: web: Received return code 0, return message 'Server certificate  \
    "www.example.com" added successfully.\nBut the certificate did not pass verification,  \
    so if the actual backend server uses the same certificate, the appliance might not be  \
    able to connect to it. Additional Certificate Authorities might be needed.\nTo avoid p \
    otential verification problems at clients (eg, browser pop-up warnings), additional/co \
    rrect chain certificates might be needed.\n' from gclSession pygs_handle_any_response
webasd[7055]: [web.INFO]: web: User admin viewing setupServiceProtocolsSSLMain page.
```

To see which issuer certificate is missing, use the command `show protocol ssl server-cert`:

Figure 8.48. Output of the command "show protocol ssl server-cert"

```
SSH # show protocol ssl server-cert name www.example.com
Name: www.example.com
Exportable: yes

Certificate Details:
Issued To:
  Common Name:       www.example.com
  Organization:      Example Dot Com
  Locality:          Sydney
  State:             NSW
  Country:           AU
  Serial Number:     1 (0x1)
Issued By:
  Common Name:       Example Dot Com Intermediate Certificate
  Organization:      Example Dot Com
  State:             NSW
  Country:           AU
Validity:
  Issued On:         Aug  5 08:08:44 2012 GMT
  Expires On:        Aug  5 08:08:44 2013 GMT
Fingerprint:
  SHA1:              FB:C5:FF:07:81:08:7D:DF:A8:40:6B:68:15:03:47:63:89:F8:72:2C
Key:
  Type:              RSA
  Size (Bits):       4096

No chain certificates.
```

This shows that we don't have the certificate of the issuer *Example Dot Com Intermediate Certificate*.

8.11.1.3.2. Import of the server private key and certificate with the intermediate certificate available but the Root CA certificate missing

If the issuer certificate of the server certificate is known on the Steelhead appliance, but the CA Root certificate is not known, then you need to obtain that CA Root certificate too.

Figure 8.49. Missing Root CA certificate missing but intermediate certificate is there

```
webasd[7055]: [web.INFO]: web: User admin viewing setupServiceProtocolsSSLCAs page.
webasd[7055]: [web.NOTICE]: web: user admin: SSL ACTION: /rbt/sport/ssl/action/server_cert \
    s/add_import; PARAMETERS: name = , exportable = true
mgmtd[3958]: [mgmtd.INFO]: while importing www.example.com: code 2 at 1 depth lookup: unab \
    le to get issuer certificate
mgmtd[3958]: [mgmtd.INFO]: /C=AU/ST=NSW/L=Sydney/O=Example Dot Com/CN=Example Dot Com Inte \
    rmediate Certificate/emailAddress=intca@example.com
mgmtd[3958]: [mgmtd.INFO]: EVENT:  /rbt/sport/ssl/event/change/backend_server
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} dyn config modify/add SSL server certificate [w \
    ww.example.com].
sport[7402]: [sslmodule.INFO] - {- -} Added a certificate with name "www.example.com"
mgmtd[3958]: [mgmtd.NOTICE]: Server certificate "www.example.com" added successfully.
mgmtd[3958]: [mgmtd.INFO]: verification failed: code 2 at 1 depth lookup: unable to get is \
    suer certificate
mgmtd[3958]: [mgmtd.INFO]: /C=AU/ST=NSW/L=Sydney/O=Example Dot Com/CN=Example Dot Com Inte \
    rmediate Certificate/emailAddress=intca@example.com
mgmtd[3958]: [mgmtd.NOTICE]: But the certificate did not pass verification, so if the actu \
    al backend server uses the same certificate, the appliance might not be able to connec \
    t to it. Additional Certificate Authorities might be needed.
mgmtd[3958]: [mgmtd.NOTICE]: Certificate Authority "Example_Dot_Com_Intermediate_CA" autom \
    atically added to the certificate chain. To avoid potential verification problems at c \
    lients (eg, browser pop-up warnings), additional/correct chain certificates might be n \
    eeded.
webasd[7055]: [web.INFO]: web: Received return code 0, return message 'Server certificate  \
    "www.example.com" added successfully.\nBut the certificate did not pass verification,  \
    so if the actual backend server uses the same certificate, the appliance might not be  \
    able to connect to it. Additional Certificate Authorities might be needed.\nCertificat \
    e Authority "Example_Dot_Com_Intermediate_CA" automatically added to the certificate c \
    hain.\nTo avoid potential verification problems at clients (eg, browser pop-up warning \
    s), additional/correct chain certificates might be needed.\n' from gclSession pygs_han \
    dle_any_response
webasd[7055]: [web.INFO]: web: User admin viewing setupServiceProtocolsSSLMain page.
```

The difference here is that the string `Certificate Authority "Example_Dot_Com_Intermediate_CA" automatically added to the certificate chain` has been added, which shows that the issuer of the server certificate is known.

Again, the command `show protocol ssl server-cert` can be used to see which one is expected:

Figure 8.50. Output of the command "show protocol ssl server-cert" for a chained certificate

```
SSH # show protocol ssl server-cert name www.example.com
Name: www.example.com
Exportable: yes

Certificate Details:
Issued To:
  Common Name:       www.example.com
  Organization:      Example Dot Com
  Locality:          Sydney
  State:             NSW
  Country:           AU
  Serial Number:     1 (0x1)
Issued By:
  Common Name:       Example Dot Com Intermediate Certificate
  Organization:      Example Dot Com
  State:             NSW
  Country:           AU
Validity:
  Issued On:         Aug  5 08:08:44 2012 GMT
  Expires On:        Aug  5 08:08:44 2013 GMT
Fingerprint:
  SHA1:              FB:C5:FF:07:81:08:7D:DF:A8:40:6B:68:15:03:47:63:89:F8:72:2C
Key:
  Type:              RSA
  Size (Bits):       4096

Chain certificates:
  Name  (Issued To)
  DF095A93A56EA5A63C9286673A84AB22  (Example Dot Com Intermediate Certificate)

SSH # show protocol ssl server-cert name www.example.com chain-cert DF095A93A56EA5A63C9286 \
    673A84AB22 certificate
Issued To:
  Common Name:       Example Dot Com Intermediate Certificate
  Organization:      Example Dot Com
  State:             NSW
  Country:           AU
  Serial Number:     1 (0x1)
Issued By:
  Common Name:       Example Dot Com Root Certificate
  Organization:      Example Dot Com
  Locality:          Sydney
  State:             NSW
  Country:           AU
Validity:
  Issued On:         Aug  5 08:01:14 2012 GMT
  Expires On:        Aug  5 08:01:14 2013 GMT
Fingerprint:
  SHA1:              C3:D8:46:D6:A3:5B:F7:7D:1E:2E:C4:E9:DC:89:1D:AD:AF:4E:95:2F
Key:
  Type:              RSA
  Size (Bits):       4096
```

This shows that we don't have the certificate of the issuer *Example Dot Com Root Certificate*.

8.11.1.3.3. Import of the server private key and certificate with the intermediate certificate and Root CA certificate available

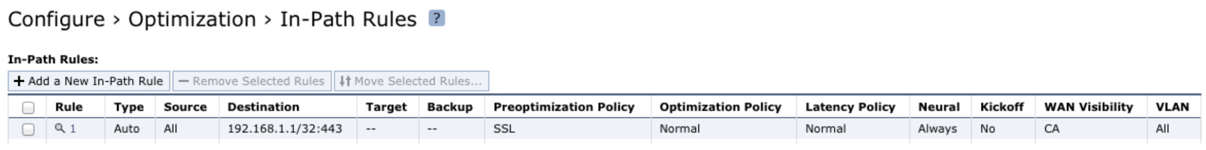
Figure 8.51. Successful import of the server certificate

```
webasd[7055]: [web.NOTICE]: web: user admin: SSL ACTION: /rbt/sport/ssl/action/server_cert \
    s/add_import; PARAMETERS: name = , exportable = true
mgmtd[3958]: [mgmtd.INFO]: EVENT:  /rbt/sport/ssl/event/change/backend_server
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} dyn config modify/add SSL server certificate [w \
    ww.example.com].
sport[7402]: [sslmodule.INFO] - {- -} Added a certificate with name "www.example.com"
mgmtd[3958]: [mgmtd.NOTICE]: Server certificate "www.example.com" added successfully.
mgmtd[3958]: [mgmtd.NOTICE]: Certificate Authority "DF095A93A56EA5A63C9286673A84AB22" auto \
    matically added to the certificate chain.
webasd[7055]: [web.INFO]: web: Received return code 0, return message 'Server certificate  \
    "www.example.com" added successfully.\nCertificate Authority "DF095A93A56EA5A63C928667 \
    3A84AB22" automatically added to the certificate chain.\n' from gclSession pygs_handle \
    _any_response
webasd[7055]: [web.INFO]: web: User admin viewing setupServiceProtocolsSSLMain page.
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} getting SSL discovered servers information
sport[7402]: [sport/mgmt/ssl.INFO] - {- -} getting SSL bypassed servers information
```

8.11.1.4. Configure an in-path rule with SSL pre-optimization

In the GUI of the client-side Steelhead appliance, add an in-path rule towards the SSL server with *SSL* as the *Pre-optimization Policy*:

Figure 8.52. SSL Pre-optimization in-path rule



▢	Rule	Type	Source	Destination	Target	Backup	Preoptimization Policy	Optimization Policy	Latency Policy	Neural	Kickoff	WAN Visibility	VLAN
▢	1	Auto	All	192.168.1.1/32:443	--	--	SSL	Normal	Normal	Always	No	CA	All

No change is needed on the server-side Steelhead appliance, as the peering rules will take care of it.

8.11.2. Test the optimization of the SSL pre-optimization

8.11.2.1. Test the optimization of the SSL session

If the SSL pre-optimization is working, after the setup of the TCP session data reduction should be seen and the server should show up as discovered in the SSL discovery list:

Figure 8.53. SSL discovery list

```
SSH # show protocol ssl backend disc-table
Discovered servers:
   # Server Name                IP              Port  Certificate Name
---- ------------------------- --------------- ----- -------------------------
   1 www.example.com           192.168.1.1     443   www.example.com
```

8.12. SCA Latency Optimization

SCA latency optimization works in various ways to improve performance times of cloud based Software-as-a-Service (SaaS) applications like Microsoft Office 365 and Salesforce.

The overall SCA service is delivered in conjunction with the Akamai CDN (Content Delivery Network). SCA combines the Riverbed WAN optimization technology (RiOS) with the Akamai Internet optimization technology (SureRoute) for accelerating SaaS platform performance. SCA uses Akamai SureRoute to provide a reliable transport across the fastest path through thousands of servers in many countries while dynamically adding RiOS instance at points nearest to the SaaS service provider.

SCA optimization is achieved using normal TCP, SDR and HTTP latency optimization techniques.

There are two methods of deployment at the customer premises.

- Direct branch Internet deployment: The branch offices have their own Internet connectivity.
- Back-hauled Internet deployment: The branch offices connect to the Internet over an internal WAN or a VPN connection to a centralized data center location that controls all Internet access.

8.12.1. How SCA Latency Optimizations work.

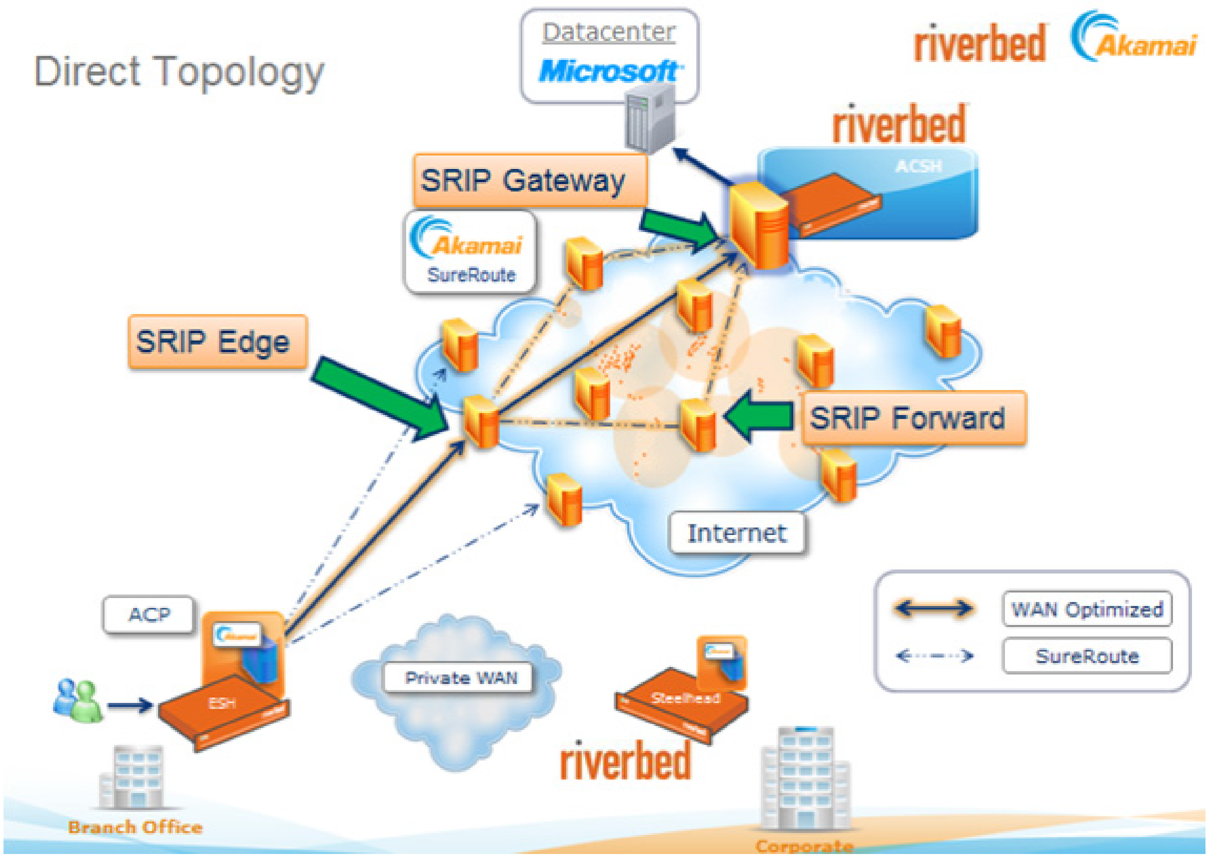
SCA optimization uses the same HTTP and TCP optimization features to deliver performance enhancement. See the HTTP Latency Optimization section in the *Latency Optimization* chapter for those details.

8.12.2. SCA components

- SaaS Platform: The platform that uses Software as a Service, such as Salesforce or Microsoft Office 365.
- Akamai Intelligent Platform: The Akamai distributed network of over 100 000 servers deployed in over 1 000 locations world-wide across the public Internet. The platform hosts Riverbed Steelhead technology and provides Internet-based optimization for Enterprise SaaS traffic.
- Akamai SureRoute Optimization: Akamai SureRoute Optimization uses a suite of technologies to provide fast and reliable delivery between the Akamai Edge Servers. Route optimization examines multiple paths across the Internet to find the fastest path and route past any failures; the Enhanced Akamai Protocol overcomes the inefficiencies of TCP to provide the highest throughput and fastest recovery; and Packet Redundancy enables you to recreate any lost data without having the client or server to retransmit.
- Akamai Edge Server: The Akamai Edge Server in the Akamai Intelligent Platform closest to the end-user is dynamically and intelligently selected (regardless of whether the end-user location has direct Internet access or the data is back-hauled to the Internet gateway at the data center). The Akamai Edge Server closest to the SaaS provider's data center runs the RiOS instances and acts as a peer to the registered ESH.
- Enterprise Data Center Steelhead (DCSH): The Steelhead appliance located in the customer data center close to the customer Internet egress point. It contains the Akamai Cloud Proxy (ACP) feature.
- Enterprise Branch Steelhead (BSH): The Steelhead appliance located in the customer branch office that intercepts any connections destined for the SaaS platform to be accelerated. The Enterprise Branch Steelhead can host the Akamai Cloud Proxy (ACP) feature, but does not require it. ACP is a software component that grants the Steelhead appliance access to the Akamai Intelligent Platform. The registered ESH accelerates application performance and data transfer over the private WAN and the Internet, overcoming bandwidth and geographical limitations.
- Riverbed Cloud Portal: An always on, always available Web portal that enables you to log on to deploy and manage Riverbed software, ESHs and SCA in the cloud. The Riverbed Cloud Portal manages registration and deregistration of SCA; it also provides the status of SCA.

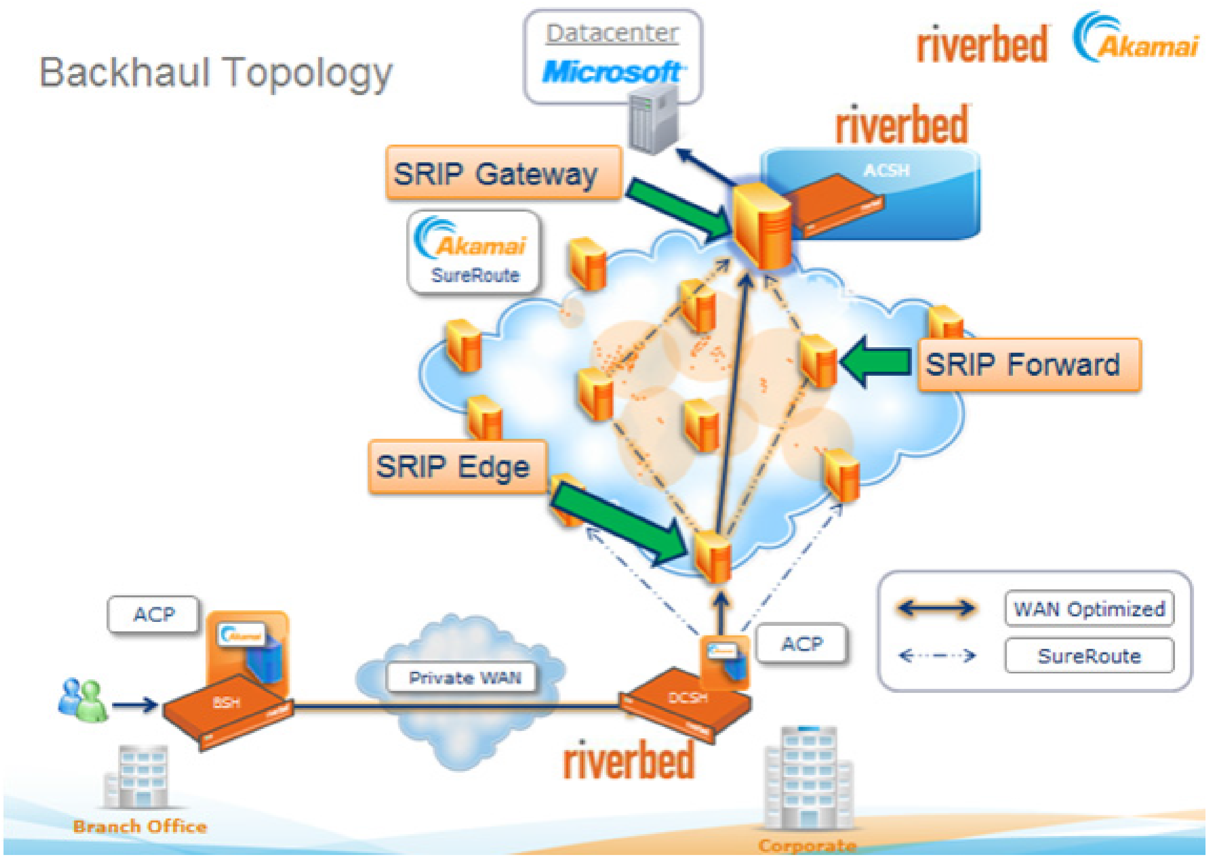
8.12.3. Setup of an optimized SCA connection Direct Branch Deployment

Figure 8.54. SCA connection Direct Branch Deployment



8.12.4. Setup of an optimized SCA connection Back-Hauled Deployment

Figure 8.55. SCA connection Back-Hauled Deployment



8.12.5. Troubleshooting SCA latency optimization

8.12.5.1. Mandatory components for a working SCA solution to operate

The following steps need to be taken to get a working SCA solution:

- The Steelhead appliance has a valid (not expired) SSL certificate.
- The Steelhead appliance has a valid SSL license installed.
- SSL optimization is enabled on the Steelhead appliance.
- Port 443 is removed from the "Secure" ports label list under Configure -> Networking -> Port Labels.
- The Steelhead appliance's primary interface is connected.
- The Host Settings is configured with valid DNS server IP addresses and these DNS servers must be able to resolve named from the public Internet, for example salesforce.com.
- The Steelhead appliance has NTP configured and is synced, also the correct time zone needs to be configured on the Steelhead appliance.

- An account on the Riverbed Cloud Portal is required.
- The Steelhead appliance needs to be appropriately registered on the Cloud Portal and a valid SCA license needs to be configured.
- Any firewall between the Steelhead appliance and the public Internet must allow for outbound access from the primary interface IP address to ports 80 and 443. If using external DNS or NTP servers, also the UDP/TCP ports 53 and UDP port 123 need to have outbound access. Ensure that the stateful feature is enabled for UDP packets.
- Any firewall between the Steelhead appliance and the public Internet should allow outbound UDP port 9545 from all in-path interfaces on the Steelhead appliance. Ensure that the stateful feature is enabled on the firewall in order to allow for returning UDP packets.
- You need to have an existing account with your SaaS provider.

8.12.5.2. Validate SCA subscription and acceleration service is turned on

In the Riverbed Cloud Portal, click Cloud Accelerator and select SaaS Platforms -> Office 365 | Salesforce | Google Apps.

- Check that the Start Time and the End Time are valid.
- The Active columns should display *true* and the Terminated column should display *false*.
- The Acceleration Service should be *ON*.

8.12.5.3. The ESH should communicate with the Riverbed Cloud Portal

Check the ESH is communicating with the Riverbed Cloud Portal using the following steps:

- In the Riverbed Cloud Portal, click Cloud Accelerator and then select Enterprise Steelhead Appliances. Verify that the serial number of the ESH is listed in either Pending, Granted or Denied sections.
- Check the output of the command `show service cloud-accel`. The Reason field indicates the current status of the ESH. If the reason is *Disabled by administrative action* and Enabled is *Yes* then the ESH might have been denied service by the Riverbed Cloud Portal, or de-registered from the ESH. You must register the ESH again.

  Figure 8.56. Output of the command 'show service cloud-accel'

  ```
  ESH # show service cloud-accel
  Enabled: Yes
  Status: Unregistered
  Reason: Disabled by administrative action (Tue Aug 21
  Portal: cloudportal.riverbed.com:443 (HTTPS)
  Redirection: Enabled
  Port: 9545
  State: Active
  Spill-over Policy: Disabled
  ESH #
  ```

- If the reason is *Disabled by administrative action* and Enabled is *No* then the ESH might have been denied service by the Riverbed Cloud Portal, or de-registered from the ESH. Also, the SCA service was disabled on the ESH.

Figure 8.57. Output of the command 'show service cloud-accel'

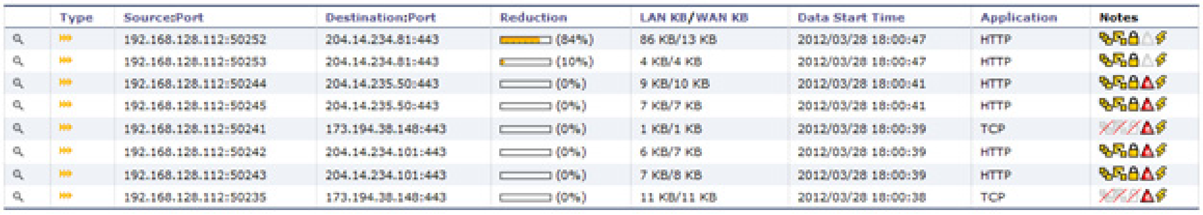
```
ESH # show service cloud-accel
Enabled: No
Status: Unregistered
Reason: Disabled by administrative action (Tue Aug 21
Portal: cloudportal.riverbed.com:443 (HTTPS)
Redirection: Enabled
Port: 9545
State: Inactive
Spill-over Policy: Disabled
ESH #
```

- If the reason is Appliance is *Pending Service* then you must grant access to the ESH on the Riverbed Cloud Portal. If the reason is *Couldn't resolve host name* then check the DNS settings on the ESH. SCA uses the Steelhead appliance DNS settings. When you change DNS settings, remember to disable and re-enable the cloud acceleration service on the ESH.

8.12.5.4. Some connections result in protocol errors

Protocol errors for some of the connections is expected behavior. This is because the system does not optimize every single SSL connection. For example, in the following figure below, connections to www.salesfore.com (204.14.235.50) and login.salesforce.com (204.14.234.101) are not optimized (not SSL decrypted). However, connections to na2.salesforce.com (204.14.234.81) are optimized.

Figure 8.58. SCA passed through connections.



	Type	Source:Port	Destination:Port	Reduction	LAN KB/WAN KB	Data Start Time	Application	Notes
		192.168.128.112:50252	204.14.234.81:443	(84%)	86 KB/13 KB	2012/03/28 18:00:47	HTTP	
		192.168.128.112:50253	204.14.234.81:443	(10%)	4 KB/4 KB	2012/03/28 18:00:47	HTTP	
		192.168.128.112:50244	204.14.235.50:443	(0%)	9 KB/10 KB	2012/03/28 18:00:41	HTTP	
		192.168.128.112:50245	204.14.235.50:443	(0%)	7 KB/7 KB	2012/03/28 18:00:41	HTTP	
		192.168.128.112:50241	173.194.38.148:443	(0%)	1 KB/1 KB	2012/03/28 18:00:39	TCP	
		192.168.128.112:50242	204.14.234.101:443	(0%)	6 KB/7 KB	2012/03/28 18:00:39	HTTP	
		192.168.128.112:50243	204.14.234.101:443	(0%)	7 KB/8 KB	2012/03/28 18:00:39	HTTP	
		192.168.128.112:50235	173.194.38.148:443	(0%)	11 KB/11 KB	2012/03/28 18:00:38	TCP	

8.12.5.5. SCA blocked by SaaS whitelist

Salesforce has a whitelist security feature. Customers can permit/deny source IP ranges to their particular Salesforce instances. For the SCA feature to work successfully, specific Akamai IP ranges must be permitted to the Salesforce instances. Please see KB S18309 for further information.

8.12.5.6. Is the traffic redirected to the Akamai SRIP network?

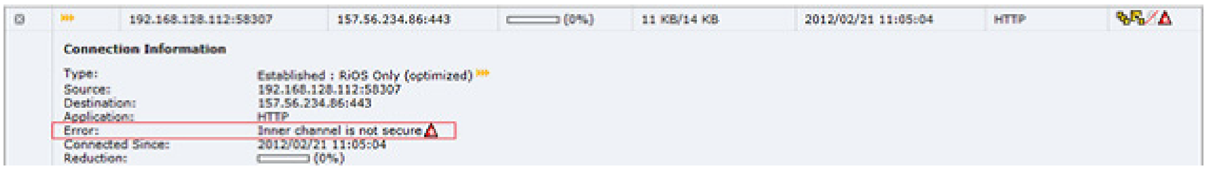
From a computer behind the client-side Steelhead appliance, initiate the traceroute command to a SaaS provider server and observe the IP address of the first hop. If the first hop is **NOT** the IP address of your default gateway, then the packets are redirected into the SRIP network.

Figure 8.59. The first hop found by traceroute should be the SRIP-Edge host

```
Default gateway is 192.168.128.1
C:\>tracert ch1prd0410.outlook.com
Tracing route to ch1prd0410.outlook.com [157.56.244.182]
over a maximum of 30 hops:
1 23 ms 38 ms 19 ms 58.27.86.183 << [SRIP-Edge]
2 305 ms 236 ms 234 ms 198.63.231.204 << [SRIP-Gateway]
3 235 ms 235 ms 237 ms be-5.r05.chcgil09.us.bb.gin.ntt.net [131.103.136.1]
4 235 ms 265 ms 237 ms 0.xe-10-2-0.BR3.CHI13.ALTER.NET [204.255.168.69]
5 234 ms 239 ms 236 ms 0.ae3.XL4.CHI13.ALTER.NET [152.63.66.77]
6 237 ms 235 ms 236 ms TenGigE0-5-2-0.GW2.CHI13.ALTER.NET [152.63.67.106]
7 268 ms 415 ms 417 ms microsoft-gw.customer.alter.net [63.84.96.94]
8 240 ms 238 ms 239 ms xe-3-0-1-0.ch1-16c-1a.ntwk.msn.net [207.46.46.153]
9 236 ms 235 ms 238 ms xe-5-0-0-0.ch1-96c-1b.ntwk.msn.net [207.46.46.125]
```

8.12.5.7. The error message 'Inner channel is not secure' appears

Figure 8.60. SCA inner channel is not secure.



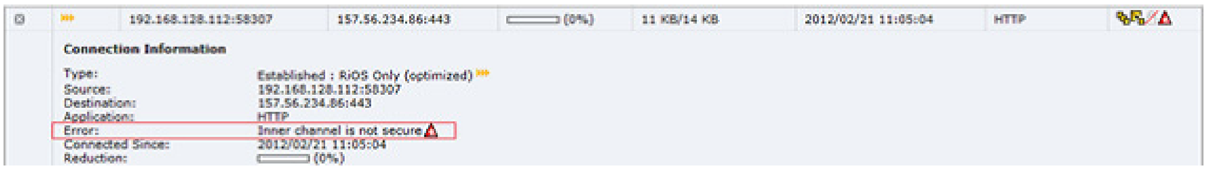
```
192.168.128.112:58307   157.56.234.86:443   (0%)   11 KB/14 KB   2012/02/21 11:05:04   HTTP

Connection Information

Type:              Established : RiOS Only (optimized)
Source:            192.168.128.112:58307
Destination:       157.56.234.86:443
Application:       HTTP
Error:             Inner channel is not secure
Connected Since:   2012/02/21 11:05:04
Reduction:         (0%)
```

If the connection details display the error message *Inner channel is not secure* then there is an issue with peering between the Steelhead appliance and the cloud. Check the validity of the peering certificate and ensure that the Peering Trust list contains the appropriate CA.

8.12.5.8. SSL security certificates error messages in the browser

When accessing the SaaS provider's website, the browser displays an error message about the SSL security certificate.

Figure 8.61. SCA invalid CA certificate installed in client browser.



```
192.168.128.112:58307   157.56.234.86:443   (0%)   11 KB/14 KB   2012/02/21 11:05:04   HTTP

Connection Information

Type:              Established : RiOS Only (optimized)
Source:            192.168.128.112:58307
Destination:       157.56.234.86:443
Application:       HTTP
Error:             Inner channel is not secure
Connected Since:   2012/02/21 11:05:04
Reduction:         (0%)
```

Ensure that the CA (CA that signed the Proxy Certificate) root certificate is installed correctly on your computer. This could be either a Customer-hosted CA or Cloud-hosted CA depending on the configuration for the SCA service in the cloud portal. The customer has a choice, by default it is cloud hosted.

Chapter 9. Logging

9.1. Index

This chapter describes on how to read the log files generated during the normal operation of the Steelhead appliance.

9.2. Logging format

The Steelhead appliance uses the standard Unix syslog format with a timestamp, a hostname, the process name and the process ID, the facility, the priority and the line to be logged:

Figure 9.1. Example log lines

```
Dec 12 16:21:59 CSH mgmtd[5082]: [mgmtd.NOTICE]: Configuration changed by user admin
Dec 12 16:26:27 CSH sport[24798]: [splice/client.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} i \
    nit client 10.0.1.1:60832 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 \
     client connected: yes
May 30 17:46:09 SSH webasd[6763]: [web.NOTICE]: web: Attempt to Authenticate admin
May 30 17:46:10 SSH last message repeated 2 times
```

The timestamp is the English three letter abbreviation for the month, the day of the month and the 24 hour time.

The hostname is the hostname of the machine the line is logged on.

If the process name is *sport*, that is the optimization service. The list of the process names can be found in the The processes on the Steelhead appliance section in the *Riverbed Approach to WAN Optimization* chapter. The number behind the process name is the Unix process ID.

The facility is the internal part of the various services. For the process *sport* that might be:

- splice: The setup and termination of TCP sessions used for optimization.
- mapi: MAPI latency optimization.
- smbcfe: CIFS latency optimization where this Steelhead appliance is the client-side Steelhead appliance.
- smbsfe: CIFS latency optimization where this Steelhead appliance is the server-side Steelhead appliance.

Some facilities can be further specified, for example:

- splice/oob: The Out-of-Band Splice related messages.
- splice/probe: The auto-discovery part of the setup of optimized TCP sessions.

The priorities are: Debug, Info, Notice, Warning, Error, Critical, Alert, and Emergency. Under normal operation, the logging level should be set to Notice. Under troubleshooting operation with Riverbed TAC, the logging level could be set to Info. Setting it to Warning or higher will reduce the ability of Riverbed TAC to properly analyze what was going on on the Steelhead appliance and should not be used. Do not use Debug.

It is possible to increase the logging level for a single part of the optimization service with the command `logging filter`, for example to have the HTTP latency optimization service set to Info level logging while the rest is on the default Notice level, use the following command: `logging filter http level info`.

Figure 9.2. Change the logging level of a single part of the optimization level

```
SH (config) # logging filter http level info
SH (config) # show logging
Local logging level: notice
Default remote logging level: notice
No remote syslog receivers configured.
Number of archived log files to keep: 10
Log rotation frequency: daily
SH (config) # show logging filter
Local logging level: notice
Process  Description                   Level
-------- ----------------------------- --------
http     HTTP Optimization             info
```

The rest of the line is the message being logged. For optimized TCP connections they start with the string in the format `<splicenumber> {<client IP address>:<client TCP port> <server IP address>:<server TCP port>}`. This string can be used to match multiple lines to a single optimized TCP session.

9.3. TCP Optimization

This section is about the basic setup of any optimized TCP session.

These messages are logged at INFO level, which means that they are normally not visible in the logs.

9.3.1. A single optimized TCP session via Auto-discovery

This is the setup of a TCP session between the client and the server on TCP port 25. The default values are used at the in-path rules, nothing fancy. Despite this is the first example, assumed is that the Out-of-Band Splice has already been setup.

9.3.1.1. With Enhanced Auto Discovery enabled

9.3.1.1.1. On the client-side Steelhead appliance

Figure 9.3. Setup of an optimized TCP session from the client-side Steelhead appliance, Enhanced Auto Discovery and no Enhanced Auto Discovery

```
CSH sport[24798]: [splice/client.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} init client 10.0. \
    1.1:60832 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connecte \
    d: yes
CSH sport[24798]: [splice/client.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} Splice client sid \
    e initializing: No protocol port = 25 protocol id = TCP(0) transport = TRANSPORT_ID_NO \
    NE
CSH sport[24798]: [splice/client.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} Start flowing, lp \
    ort 40268, rport 7800, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_NONE, TPTOPT_NONE(0x0)

CSH sport[24798]: [splice/client.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} fini client 10.0. \
    1.1:60832 server 192.168.1.1:25 cfe 10.0.1.6:40268 sfe 192.168.1.6:7800 app TCP
```

The client-side of the optimized TCP session can be determined by the logging facility: *splice/client*. The first line is printed when the SYN/ACK+ has been returned from the server-side Steelhead appliance.

There are four lines logged:

The first line is the start of the optimized TCP session, *init*. It determines the IP addresses and TCP ports of the client, the server, the client-side Steelhead appliance *cfe* and the server-side Steelhead appliance *sfe*. The TCP port of the client-side Steelhead appliance is port 7801, which is an internal port.

The words *client connected: yes* shows that the OOB Splice has already been setup.

The second line determines the latency optimization, plain TCP in this case *protocol id = TCP(0)*, and the pre-optimization parameters.

- TRANSPORT_ID_NONE: Nothing special on the inner channel.
- TRANSPORT_ID_SSLINNER: The inner channel is SSL encrypted.

The third line shows the TCP ports of the inner channel and the parameters on the outer channels.

The words *lport 40268* are from the TCP port of the inner channel on the client-side Steelhead appliance, the words *rport 7800* are from the TCP port of the inner channel on the server-side Steelhead appliance.

The next fields are the Optimization Policy, the Neural Framing, the Pre-optimization Policy, the Latency Policy, the inner channel Transport layer Policies and the Transport Optimization.

Optimization Policy:

- OPOL_NORMAL: Perform full optimization with SDR and compression.
- OPOL_SDR_ONLY: Perform SDR Only, do not perform compression.
- OPOL_COMPR_ONLY: Perform only compression, do not perform SDR.
- OPOL_NONE: Do not perform SDR nor compression.
- OPOL_IN_MEM_ONLY: Use SDR-M: Perform full optimization with SDR and compression but keep the references in memory only, do not write them to disk.

Neural Framing Mode:

- NAGLE_NEVER: Always process the TCP payload packet by packet.
- NAGLE_ALWAYS: Use the Nagle algorithm to decide when to process the TCP payload data.
- NAGLE_TCPHINTS: Process the TCP payload data when a TCP packet with the PUSH flag has arrived.
- NAGLE_NEURAL: Let the neural framing algorithm decide when to process the TCP payload data.

The Pre-optimization Policy:

- PREOPT_NONE: No pre-optimization necessary.
- PREOPT_SSL: The data on the outer channels is SSL encrypted.
- PREOPT_JINIT: The data on the outer channels is jinitiator encapsulated.
- PREOPT_JINIT_SSL: The data on the outer channels is jinitiator over SSL encrypted.

The Latency Optimization Policy:

- LATOPT_AUTO: The Latency Optimization is determined on the destination TCP port.
- LATOPT_HTTP: Use HTTP Latency Optimization.
- LATOPT_NONE: Do not use any Latency Optimization.
- LATOPT_RPCH: Use RPC-over-HTTP Latency for Outlook Anywhere.

The inner channel Transport layer Policies:

- TRANSPORT_ID_NONE: No transport policy, the inner channel is not encrypted.
- TRANSPORT_ID_SSLINNER: The inner channel is SSL encrypted.

Transport Optimization:

- TPTOPT_NONE: Do not perform a transport optimization.
- SCPS_IN_TE_W: Act as a SCPS initiator terminator on the WAN side.
- SCPS_IN_TE_L: Act as a SCPS initiator terminator on the LAN side.
- SCPS_TE_W: Act as a SCPS terminator on the WAN side.
- SCPS_TE_L: Act as a SCPS terminator on the LAN side.
- SCPS_IN_W: Act as a SCPS initiator on the WAN side.
- SCPS_IN_L: Act as a SCPS initiator on the LAN side.
- TPTOPT_PROXY: Do not use SCPS, only act as a TCP proxy.

The fourth line is the termination of the optimized TCP session, *fin*. It shows the IP addresses and TCP ports of the client, server, client-side Steelhead appliance and the server-side Steelhead appliance.

9.3.1.1.2. On the server-side Steelhead appliance

With Enhanced Auto Discovery the TCP session towards the server is setup before the inner channel is created.

Figure 9.4. Setup of an optimized TCP session from the server-side Steelhead appliance, Enhanced Auto Discovery

```
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:0 clnt: 10.0.1.1:60832 se \
    rv: 192.168.1.1:25) init
SSH sport[24255]: [splice/server.INFO] 2 {- -} init cfe 10.0.1.6:40268 sfe 192.168.1.6:780 \
    0
SSH sport[24255]: [splice/server.INFO] 2 {- -}  sock 40 id 548627 client 10.0.1.1:60832 se \
    rver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40269 clnt: 10.0.1.1:6083 \
    2 serv: 192.168.1.1:25) acquired
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40269 clnt: 10.0.1.1:6083 \
    2 serv: 192.168.1.1:25) fini
SSH sport[24255]: [splice/server.INFO] 2 {- -} Splice server side initializing: No protoco \
    l port = 25 transport = TRANSPORT_ID_NONE
SSH sport[24255]: [splice/server.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} Start flowing, lp \
    ort 7800, rport 40268, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_NONE, TPTOPT_NONE(0x0)

SSH sport[24255]: [splice/server.INFO] 2 {10.0.1.1:60832 192.168.1.1:25} fini client 10.0. \
    1.1:60832 server 192.168.1.1:25 cfe 10.0.1.6:40268 sfe 192.168.1.6:7800 app TCP
```

The server-side of the optimized TCP session can be determined by the logging facility: *splice/probe* and *splice/server*. The first line is printed when a SYN+ packet is seen.

The first line identifies the auto discovery probe in the SYN+ packet: It determines the in-path interface IP address the probe came in on and the IP addresses and TCP ports of the client and server. Note that this line was not shown on the client-side Steelhead appliance.

The second line is printed when the TCP session with the server has been setup. It contains the IP addresses and TCP port numbers of the inner channel to be setup.

The third line contains the IP addresses and TCP port numbers of the client and the server and the WAN visibility mode:

WAN Visibility:

- TRPY_NONE: Correct Addressing
- TRPY_PORT: Port Transparency
- TRPY_FULL: Full Transparency

Line four and five are internal housekeeping of the auto-discovery phase.

Line six contains the protocol port and the Transport Policies.

Line seven shows the TCP ports of the inner channel and the parameters on the outer channels.

- The *lport* is the TCP port of the inner channel on the client-side Steelhead appliance.
- The *rport* is the TCP port of the inner channel on the server-side Steelhead appliance.

The next fields are the Optimization Policy, the Neural Framing, the Pre-optimization Policy, the Latency Policy, the Transport layer Policies and the Transport Optimization.

Line eight shows the end of the optimized TCP session again.

9.3.1.2. With Enhanced Auto Discovery not enabled

Without Enhanced Auto Discovery the inner channel is created before the TCP session towards the server is setup. The logging on the client-side Steelhead appliance is the same as with Enhanced Auto Discovery enabled.

Figure 9.5. Setup of an optimized TCP session from the server-side Steelhead appliance, No Enhanced Auto Discovery

```
SSH sport[3073]: [splice/server.INFO] 2 {- -} init cfe 10.0.1.6:40268 sfe 192.168.1.6:7800 \

SSH sport[3073]: [splice/server.INFO] 2 {- -}  sock 39 id 548627 client 10.0.1.1:52776 ser \
    ver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[3073]: [splice/server.INFO] 2 {- -} Splice server side initializing: No protocol \
     port = 25 transport = TRANSPORT_ID_NONE
SSH sport[3073]: [splice/server.INFO] 2 {10.0.1.1:52776 192.168.1.1:25} Start flowing, lpo \
    rt 7800, rport 40268, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_I \
    D_NONE, TPTOPT_NONE(0x0)

SSH sport[3073]: [splice/server.INFO] 2 {10.0.1.1:52776 192.168.1.1:25} fini client 10.0.1 \
    .1:52776 server 192.168.1.1:25 cfe 10.0.1.6:40268 sfe 192.168.1.6:7800 app TCP
```

As can be seen here, the *splice/probe* messages do not appear because without Enhanced Auto Discovery the inner channel is setup towards the first Steelhead appliance in the network and therefore no further detection is necessary performed.

9.3.2. Optimized TCP session and an Out-of-Band Splice

When two Steelhead appliances setup an optimized TCP session between two in-path interfaces for the first time, they will setup the Out-of-Band Splice first and then the optimized TCP session. Note that the setup of the TCP sessions for the Connection Pool is not logged in the system logs.

9.3.2.1. With Enhanced Auto Discovery enabled.

9.3.2.1.1. On the client-side Steelhead appliance

Figure 9.6. Setup of an OOB Splice from the client-side Steelhead appliance, Enhanced Auto Discovery

```
CSH sport[24798]: [splice/client.INFO] 1 {10.0.1.1:47616 192.168.1.1:25} init client 10.0. \
    1.1:47616 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connecte \
    d: no
CSH sport[24798]: [splice/oob.INFO] 1 {- -} New OOB Splice created at 10.0.1.6 for peer 19 \
    2.168.1.6
CSH sport[24798]: [mgmt.INFO] - {- -} mgmtd notified that remote peer 192.168.1.6 was disc \
    overed
CSH sport[24798]: [splice/oob.INFO] 1 {- -} Establishing OOB Splice from 10.0.1.6:0 to 192 \
    .168.1.6:7800
CSH mgmtd[3766]: [mgmtd.INFO]: EVENT:  /rbt/sport/peer/event/added
CSH sport[24798]: [splice/oob.INFO] 1 {- -} OOB Splice connected from 10.0.1.6:40271 to 19 \
    2.168.1.6:7800
CSH sport[24798]: [splice/oob.INFO] 1 {- -} Negotiated sport protocol version: 8 for peer: \
     192.168.1.6:7800
CSH sport[24798]: [splice/client.INFO] 1 {10.0.1.1:47616 192.168.1.1:25} Splice client sid \
    e initializing: No protocol port = 25 protocol id = TCP(0) transport = TRANSPORT_ID_NO \
    NE
CSH sport[24798]: [splice/client.INFO] 1 {10.0.1.1:47616 192.168.1.1:25} Start flowing, lp \
    ort 40269, rport 7800, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_NONE, TPTOPT_NONE(0x0)

CSH sport[24798]: [splice/client.INFO] 1 {10.0.1.1:47616 192.168.1.1:25} fini client 10.0. \
    1.1:47616 server 192.168.1.1:25 cfe 10.0.1.6:40269 sfe 192.168.1.6:7800 app TCP
```

Line one has the words *client connected: no*, which means that no OOB Splice has been setup yet.

Line two to seven are related to the setup of the OOB Splice.

Line eight and nine are the setup of the inner channel of the optimized TCP session.

9.3.2.1.2. On the server-side Steelhead appliance

Figure 9.7. Setup of an OOB Splice from the server-side Steelhead appliance, Enhanced Auto Discovery

```
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:0 clnt: 10.0.1.1:47616 se \
    rv: 192.168.1.1:25) init
SSH sport[24255]: [splice/oob.INFO] 1 {- -} New OOB Splice created at 192.168.1.6 for peer \
     10.0.1.6
SSH sport[24255]: [mgmt.INFO] - {- -} mgmtd notified that remote peer 10.0.1.6 was discove \
    red
SSH sport[24255]: [splice/oob.INFO] 1 {- -} Negotiated sport protocol version: 8 for peer: \
     10.0.1.6:0
SSH mgmtd[4018]: [mgmtd.INFO]: EVENT:  /rbt/sport/peer/event/added
SSH sport[24255]: [splice/server.INFO] 1 {- -} init cfe 10.0.1.6:40269 sfe 192.168.1.6:780 \
    0
SSH sport[24255]: [splice/server.INFO] 1 {- -}  sock 41 id 548627 client 10.0.1.1:47616 se \
    rver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40268 clnt: 10.0.1.1:4761 \
    6 serv: 192.168.1.1:25) acquired
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40268 clnt: 10.0.1.1:4761 \
    6 serv: 192.168.1.1:25) fini
SSH sport[24255]: [splice/server.INFO] 1 {- -} Splice server side initializing: No protoco \
    l port = 25 transport = TRANSPORT_ID_NONE
SSH sport[24255]: [splice/server.INFO] 1 {10.0.1.1:47616 192.168.1.1:25} Start flowing, lp \
    ort 7800, rport 40269, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_ \
    ID_NONE, TPTOPT_NONE(0x0)

SSH sport[24255]: [splice/server.INFO] 1 {10.0.1.1:47616 192.168.1.1:25} fini client 10.0. \
    1.1:47616 server 192.168.1.1:25 cfe 10.0.1.6:40269 sfe 192.168.1.6:7800 app TCP
```

Line two to five are for the setup of the inner channel.

9.3.2.2. With Enhanced Auto Discovery not enabled

On the server-side Steelhead appliance the logging is similar to the server-side Steelhead appliance with Enhanced Auto Discovery enabled, but without the *splice/probe* messages:

Figure 9.8. Setup of an OOB Splice from the server-side Steelhead appliance, no Enhanced Auto Discovery

```
SSH sport[3073]: [splice/oob.INFO] 1 {- -} New OOB Splice created at 192.168.1.6 for peer  \
    10.0.1.6
SSH sport[3073]: [mgmt.INFO] - {- -} mgmtd notified that remote peer 10.0.1.6 was discover \
    ed
SSH sport[3073]: [splice/oob.INFO] 1 {- -} Negotiated sport protocol version: 8 for peer:  \
    10.0.1.6:0
SSH mgmtd[4018]: [mgmtd.INFO]: EVENT:  /rbt/sport/peer/event/added
SSH sport[3073]: [splice/server.INFO] 1 {- -} init cfe 10.0.1.6:40269 sfe 192.168.1.6:7800 \

SSH sport[3073]: [splice/server.INFO] 1 {- -}  sock 40 id 548627 client 10.0.1.1:34669 ser \
    ver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[3073]: [splice/server.INFO] 1 {- -} Splice server side initializing: No protocol \
     port = 25 transport = TRANSPORT_ID_NONE
SSH sport[3073]: [splice/server.INFO] 1 {10.0.1.1:34669 192.168.1.1:25} Start flowing, lpo \
    rt 7800, rport 40269, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_I \
    D_NONE, TPTOPT_NONE(0x0)

SSH sport[3073]: [splice/server.INFO] 1 {10.0.1.1:34669 192.168.1.1:25} fini client 10.0.1 \
    .1:34669 server 192.168.1.1:25 cfe 10.0.1.6:40269 sfe 192.168.1.6:7800 app TCP
```

9.3.3. Failures

9.3.3.1. Cannot setup Out-of-Band Splice

Before the inner channel of an optimized TCP session can be setup, the Out-of-Band Splice must have been setup. If the OOB Splice cannot be setup, then the optimized TCP session will not be setup.

Figure 9.9. Cannot setup OOB Splice

```
CSH sport[4564]: [splice/client.INFO] 1 {10.0.1.1:56604 192.168.1.1:25} init client 10.0.1 \
    .1:56604 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connected \
    : no
CSH sport[4564]: [splice/client.INFO] 1 {10.0.1.1:56604 192.168.1.1:25} fini client 10.0.1 \
    .1:56604 server 192.168.1.1:25 cfe 10.0.1.6:0 sfe 192.168.1.6:7800 app TCP
CSH kernel: [intercept.INFO] Peer 192.168.1.6:7800 is unreachable or incompatible. New con \
    nections going through that peer will be passed through temporarily.
CSH kernel: [intercept.INFO] Peer 192.168.1.6:7800 is unreachable or incompatible. New con \
    nections going to server 192.168.1.1:25 through that peer using local inpath 10.0.1.6  \
    will be passed through for at least 293 seconds.
CSH sport[4564]: [connect_pool.WARN] - {- -} Error connecting to peer: Connection timed ou \
    t
```

9.3.3.2. No service is listening

With Enhanced Auto Discovery the outer channel towards the server is setup before the inner channel is established. Therefore this issue is only logged on the server-side Steelhead appliance:

Figure 9.10. Server-side Steelhead appliance, no service listening on TCP port 25, Enhanced Auto Discovery

```
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:0 clnt: 10.0.1.1:22778 se \
    rv: 192.168.1.1:25) init
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40427 clnt: 10.0.1.1:2277 \
    8 serv: 192.168.1.1:25) Error connecting to server: Connection refused
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40427 clnt: 10.0.1.1:2277 \
    8 serv: 192.168.1.1:25) fini
```

Without Enhanced Auto Discovery the inner channel is setup first before being checked if the service is actually reachable and available. Therefore both the client-side and the server-side Steelhead appliance will log about it:

Figure 9.11. Client-side Steelhead appliance, no service listening on TCP port 25, no Enhanced Auto Discovery

```
CSH sport[4511]: [splice/client.INFO] 3 {10.0.1.1:41527 192.168.1.1:25} init client 10.0.1 \
    .1:41527 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connected \
    : yes
CSH sport[4511]: [splice/client.INFO] 3 {10.0.1.1:41527 192.168.1.1:25} Splice client side \
     initializing: No protocol port = 25 protocol id = TCP(0) transport = TRANSPORT_ID_NON \
    E
CSH sport[4511]: [splice/client.INFO] 3 {10.0.1.1:41527 192.168.1.1:25} Start flowing, lpo \
    rt 40270, rport 7800, OPOL_NORMAL, NAGLE_ALWAYS, PREOPT_NONE, LATOPT_AUTO, TRANSPORT_I \
    D_NONE, TPTOPT_NONE(0x0)
CSH sport[4511]: [decoder.INFO] 3 {10.0.1.1:41527 192.168.1.1:25} Warning: Inner channel d \
    own prematurely, peer probably down; requesting shutdown
CSH sport[4511]: [splice/client.INFO] 3 {10.0.1.1:41527 192.168.1.1:25} fini client 10.0.1 \
    .1:41527 server 192.168.1.1:25 cfe 10.0.1.6:40270 sfe 192.168.1.6:7800 app TCP
```

Figure 9.12. Server-side Steelhead appliance, no service listening on TCP port 25, no Enhanced Auto Discovery

```
SSH sport[3073]: [splice/server.INFO] 3 {- -} init cfe 10.0.1.6:40270 sfe 192.168.1.6:7800 \

SSH sport[3073]: [splice/server.INFO] 3 {- -}  sock 41 id 548627 client 10.0.1.1:41527 ser \
    ver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[3073]: [splice/server.INFO] 3 {- -} (clnt: 10.0.1.1:41527 peer: 10.0.1.6:40270 s \
    erv: 192.168.1.1:25) Error connecting to server: Connection refused
SSH sport[3073]: [splice/server.INFO] 3 {- -} fini client 10.0.1.1:41527 server 192.168.1. \
    1:25 cfe 10.0.1.6:40270 sfe 192.168.1.6:7800 app TCP
```

9.3.3.3. No host on the subnet

With Enhanced Auto Discovery the outer channel towards the server is tried to be setup before the inner channel is established. Since all failure is happening on the server-side Steelhead appliance, the client-side Steelhead appliance will not show any logging for this TCP session:

Figure 9.13. Server-side Steelhead appliance, no host with IP address 192.168.1.1, Enhanced Auto Discovery

```
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:0 clnt: 10.0.1.1:65396 se \
    rv: 192.168.1.1:25) init
SSH sport[24255]: [splice/probe.INFO] 0 {- -} (locl: 192.168.1.6:40428 clnt: 10.0.1.1:6539 \
    6 serv: 192.168.1.1:25) fini
```

Without Enhanced Auto Discovery the inner channel is setup first before being checked if the service is actually reachable and available. Therefor there is also logging for this TCP session on the client-side Steelhead appliance.

Figure 9.14. Client-side Steelhead appliance, no host with IP address 192.168.1.1, no Enhanced Auto Discovery

```
CSH sport[4511]: [splice/client.INFO] 4 {10.0.1.1:31909 192.168.1.1:25} init client 10.0.1 \
    .1:31909 server 192.168.1.1:25 cfe 10.0.1.6:7801 sfe 192.168.1.6:7800 client connected \
    : no
CSH sport[4511]: [splice/client.INFO] 4 {10.0.1.1:31909 192.168.1.1:25} Splice client side \
     initializing: No protocol port = 25 protocol id = TCP(0) transport = TRANSPORT_ID_NON \
    E
CSH sport[4511]: [splice/client.INFO] 4 {10.0.1.1:31909 192.168.1.1:25} fini client 10.0.1 \
    .1:31909 server 192.168.1.1:25 cfe 10.0.1.6:40272 sfe 192.168.1.6:7800 app TCP
CSH kernel: [intercept.INFO] Peer 192.168.1.6:7800 is unreachable or incompatible. New con \
    nections going through that peer will be passed through temporarily.
CSH kernel: [intercept.INFO] Peer 192.168.1.6:7800 is unreachable or incompatible. New con \
    nections going to server 192.168.1.1:25 through that peer using local inpath 10.0.1.6  \
    will be passed through for at least 293 seconds.
```

Figure 9.15. Server-side Steelhead appliance, no host with IP address 192.168.1.1, no Enhanced Auto Discovery

```
SSH sport[3073]: [splice/server.INFO] 4 {- -} init cfe 10.0.1.6:40272 sfe 192.168.1.6:7800 \

SSH sport[3073]: [splice/server.INFO] 4 {- -}  sock 43 id 548627 client 10.0.1.1:31909 ser \
    ver 192.168.1.1:25 remote inner port 7800 trpy TRPY_NONE
SSH sport[3073]: [splice/server.INFO] 4 {- -} fini client 10.0.1.1:31909 server 192.168.1. \
    1:25 cfe 10.0.1.6:40272 sfe 192.168.1.6:7800 app TCP
```

9.3.3.4. Third SYN received before inner channel got setup

To setup a TCP session the client will send a maximum of three SYN packets. When the first SYN packet is intercepted by the Steelhead appliance and an optimized TCP session can be setup for it, it will put the TCP session in a NAT table and send a SYN+. If the second SYN packet is intercepted, it will just ignore it because an optimized is already being setup. If the third SYN packet is received and the optimized TCP session is not yet setup, it will just pass the SYN packet through and mark the TCP session as pass-through.

Figure 9.16. Third SYN received before inner channel got setup

```
SH kernel: [intercept.NOTICE] nat_check: SYN packet for already natted connection 10.0.1.1 \
    :52181 -> 10.0.1.6:5486 ==> 10.0.1.1:52181 -> 192.168.1.6:7801
```

9.3.3.5. TCP session closed before the inner channel could have been setup

Some applications, for example the Windows SMB protocol, setup multiple TCP sessions at once and close the others when one of them gets setup successfully. It is possible that the inner channel for them hasn't been setup yet, and the following message is logged:

Figure 9.17. TCP session closed before the inner channel could have been setup

```
CSH sport[6693]: [splice/probe.NOTICE] 0 {- -} (locl: 10.0.1.6:50499 clnt: 10.0.1.1:2686 s \
    erv: 192.168.1.1:139) No inner channel created for this probe splice
```

9.4. OOB Splice and Connection Pool

This section is about the setup and termination of the Out-of-Band splice and the TCP sessions from the Connection Pool.

9.4.1. OOB Splice lost due to admission control

When a Steelhead appliance goes into admission control, it will send the OOB disconnect message *Admission Control* to the connected Steelhead appliances and terminate the TCP sessions of the connection pool.

Figure 9.18. Local Steelhead appliance into admission control

```
CSH statsd[7333]: [statsd.NOTICE]: Alarm triggered for rising error for event admission_co \
    nn
CSH sport[8298]: [admission_control.NOTICE] - {- -} Connection limit achieved. Total Conne \
    ctions 136, Branched Warmed Connections 0
CSH sport[8298]: [admission_control.WARN] - {- -} Pausing intercept: Connection limit achi \
    eved;
CSH sport[8298]: [admission_control.NOTICE] - {- -} Memory Usage: 296 Current Connections: \
     136 TCP Memory Usage: 29;
CSH sport[8298]: [splice/oob.NOTICE] 11 {- -} Resetting state for oob splice: laddr=10.0.1 \
    .6:43437 raddr=192.168.1.6:7800 cflag=0x8
CSH sport[8298]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
```

Figure 9.19. Remote Steelhead appliance into admission control

```
CSH sport[6693]: [splice/oob.NOTICE] 8 {- -} Received disconnect cause="Admission control" \
     laddr=10.0.1.6:7800 raddr=192.168.1.6:26539
CSH sport[6693]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
CSH sport[6693]: [splice/oob.NOTICE] 8 {- -} Lost OOB Splice between laddr=10.0.1.6:7800 a \
    nd raddr=192.168.1.6:7800
CSH sport[6693]: [spawnsplice.NOTICE] - {- -} (from 192.168.1.6:26607) End of stream readi \
    ng splice hdr info. Peer maybe down.
CSH sport[6693]: [spawnsplice.NOTICE] - {- -} (from 192.168.1.6:26582) End of stream readi \
    ng splice hdr info. Peer maybe down.
CSH sport[6693]: [spawnsplice.NOTICE] - {- -} (from 192.168.1.6:26578) End of stream readi \
    ng splice hdr info. Peer maybe down.
[...]
CSH sport[6693]: [spawnsplice.NOTICE] - {- -} (from 192.168.1.6:26545) End of stream readi \
    ng splice hdr info. Peer maybe down.
CSH sport[6693]: [spawnsplice.NOTICE] - {- -} (from 192.168.1.6:26556) End of stream readi \
    ng splice hdr info. Peer maybe down.
```

9.4.2. Inner channel setup failures

When an inner channel gets setup and fails, the existing OOB Splice gets torn down:

Figure 9.20. Remote Steelhead appliance failed to setup an inner channel

```
CSH sport[26658]: [splice/oob.NOTICE] 12 {- -} Received disconnect cause="TproxyClientSpli \
    ce connect failure" laddr=10.0.1.6:57962 raddr=192.168.1.6:7800
CSH sport[26658]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
CSH sport[26658]: [splice/oob.NOTICE] 12 {- -} Lost OOB Splice between laddr=10.0.1.6:5796 \
    2 and raddr=192.168.1.6:7800
```

9.4.3. High Availability switch

In a High Availability setup, where one Steelhead appliance is the primary and takes all connections and the second Steelhead appliance is the backup, the switch in roles shows like this:

Figure 9.21. Remote Steelhead becomes the backup in a High Availability cluster

```
CSH sport[26658]: [splice/oob.NOTICE] 12 {- -} Received disconnect cause="FailoverSplice r \
    eleased ownership" laddr=10.0.1.6:57962 raddr=192.168.1.6:7800
CSH sport[26658]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
CSH sport[26658]: [splice/oob.NOTICE] 12 {- -} Lost OOB Splice between laddr=10.0.1.6:5796 \
    2 and raddr=192.168.1.6:7800
```

9.4.4. Voluntary takedown of OOB Splice

The keep-alive packets going over the OOB Splice and the Connection Pool can be expensive with for example satellite links. It is possible for the OOB Splice to be taken down when there are no active optimized TCP sessions:

Figure 9.22. Local Steelhead appliance disconnects

```
CSH sport[15982]: [splice/oob.NOTICE] 1 {- -} Resetting state for oob splice: laddr=10.0.1 \
    .6:37964 raddr=192.168.1.6:7800 cflag=0x200
CSH sport[15982]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
CSH sport[15982]: [splice/OOBHandler.NOTICE] - {- -} Dropping oob because of no sessions t \
    o 192.168.1.6:7800
```

Figure 9.23. Local Steelhead appliance disconnects

```
CSH sport[15982]: [splice/oob.NOTICE] 1 {- -} Received disconnect cause="No connections to \
     Peer" laddr=10.0.1.6:37957 raddr=192.168.1.6:7800
CSH sport[15982]: [connect_pool.NOTICE] - {- -} Destroying pool to peer: 192.168.1.6
CSH sport[15982]: [splice/oob.NOTICE] 1 {- -} Lost OOB Splice between laddr=10.0.1.6:37957 \
     and raddr=192.168.1.6:7800
```

Chapter 10. Further help and support

This chapter contains the sources of further support.

- Riverbed Support and Riverbed Splash website.
- Documentation.
- Riverbed TAC

10.1. Riverbed websites

There are two websites to find help on issues: The Riverbed Support website and the Riverbed Splash website.

10.1.1. The Riverbed Support website

The Riverbed Support website can be found at http://support.riverbed.com/. It contains software images for all products, full documentation sets and a knowledge base system.

To get access to the support website, you need to create an account for which you need to have a Steelhead appliance with a valid support contract in your network, either purchased or via an evaluation implementation.

On the site you can see company specific details like:

- The assets assigned to your company with their support contract expiry date,
- The cases you have created with the Riverbed TAC,
- The license keys for your assets.

And general links to:

- The RiOS Software and Documentation download area. On it there is also a Software Upgrade Path which displays which intermediate software releases needs to be installed when upgrading between two software versions.
- The Knowledge base area, with various knowledge base articles which explains possible problem scenarios, third party software compatibility and explanation of logging and error messages.
- The Multimedia section, which has various videos on how to replace parts of the hardware of the system.

10.1.2. The Riverbed Splash website

The Riverbed Splash website is a community driven discussion forum and can be found at http://splash.riverbed.com/. It contains white papers, discussion forums and has an email alert service for software release announcements.

Unlike the official Riverbed Support website, everybody can sign up to this forum and join the discussions.

10.2. Documentation

There are several documents available from the Support website which will describe the basics, the capabilities and the features of the Steelhead appliances.

- The Management Console Users Guide, which describes the various parts of the GUI of the Steelhead appliance.
- The Deployment Guide, which describes various deployment scenarios and integration issues.
- The Hardware Maintenance Guide, which describes the steps for installing new hardware into a Steelhead appliance and replacing hardware into a Steelhead appliance received via an RMA.

10.3. Riverbed TAC

The Riverbed TAC provides technical support for operational issues related to Riverbed products in your network. There are six TAC centers globally distributed to provide follow-the-sun support: Sydney in Australia, Singapore, Hoofddorp in the Netherlands, Bracknell in the United Kingdom, New York in the eastern USA and Sunnyvale and San Francisco in the western USA.

They can be reached via email (mail to support@riverbed.com), via the Riverbed Support website (http://support.riverbed.com/cases/) or via the telephone (+1 415 247 7831).

10.3.1. Priority

There are four priority levels which are used by Riverbed TAC to find out the urgency of a case: P1 for critical issues, P2 for urgent issues, P3 for normal issues or questions and P4 for administrative issues.

For a case opened via email, it will always open as a P3. Cases opened via the website or via the phone can have their priority set during the entering process.

Do not open a P1 case via email or the website, always call in to get immediate support.

For non-P1 cases, the TAC engineer assigned to the case will most likely answer via email. If you want to be contacted via telephone to discuss, please state this and the telephone number you can be reached on in the initial email.

Sometimes cases are opened or picked up outside your normal business hours and are handled by a TAC engineer outside your region. If the situation sounds urgent and no contact could be made, the TAC engineer will re-queue the case to the appropriate region for follow up during your business hours. If this has not happened, feel free to call in the to TAC to request the case to be re-assigned to a local TAC engineer.

10.3.2. Before you open a case

To make life easier of the TAC engineer and for a quicker handling of the case, please consider the following issues:

10.3.2.1. Identify the systems involved

The systems involved in the issue can be either a single Steelhead appliance in case of a hardware problem. It can multiple Steelhead appliances on multiple sites of the network. It can include non-Riverbed devices, like Exchange servers or firewalls.

Make sure you are able to describe the network around the Steelhead appliances:

- What are the devices connected to the Steelhead appliance?
- Are there IDS/IPS/firewalls devices on the network between the client and server?
- Are there any load-balancers involved?
- Are the Steelhead appliances on this site a new implementation?
- Were there any changes to the network earlier?
- Were there any changes to the clients or servers earlier?

10.3.2.2. Prepare a system dump

Please create a system dump on the devices you are experiencing a problem on. If the issue is hardware related, a system dump of the device itself is sufficient. If the issue is not hardware related, then the system dumps of all the devices involved will most likely be required.

System dumps can be either:

- Emailed to the TAC engineer. There is a 50 Mb size limit on incoming emails.
- Attached to the case at the Riverbed Support website. There is a 100 Mb size limit on files uploaded via the Riverbed Support website.
- Uploaded to the Riverbed FTP server at ftp://ftp.riverbed.com/incoming/. Please prefix the file uploaded with the string "*case_NNNNNN_*" so the files can be easily identified. Make sure you use the FTP client Binary Mode to transfer the files. There is no size limit on this method.

  The */incoming* directory is write-only and the contents are hidden, which means that you can only upload files to it and not see the directory contents. If the upload gets aborted half-way then it is not possible to resume the upload. If this happens, change the filename and retry the upload again.

  Since RiOS version 8.5, the GUI and the CLI allow upload to the FTP server directly. If the network policies allow the Steelhead appliance to access the Riverbed FTP directly, please use that method.
- Be stored on a private dropbox system of your choice so that the TAC engineer can retrieve it from there.

If the system dump is too large to be emailed or uploaded in one go, consider splitting the system dump with WinZIP or WinRAR so that the pieces can be emailed or uploaded without running into the size or access restrictions.

10.3.2.3. Alarms

If you get an alarm from the Steelhead appliance via email, attach it to the case. Not all the information from these emails is available from the system dump.

10.3.2.4. RMA details

If the issue seems to be a hardware problem, please obtain the following details in case an RMA is required:

- The name, email address and telephone number of the person who will receive the replacement part.
- The address to send the replacement part to.
- If it is a full-chassis RMA, please state if the device has RSP features installed and the number of by-pass cards and their types installed.
- For Platinum and Gold Plus support RMAs, please state the date and time the delivery and replacement can be done (ASAP or a specific date/time) and any access restrictions to the site.
- For full chassis replacements in Platinum support RMAs, note that the field engineer swaps the device and only configures the device in such a way that the primary interface is reachable on the network. Software upgrades and further configuration will need to be done by the customer.

10.3.3. Life cycle of a case with Riverbed TAC

In general, once a case is submitted (*New*), it will be picked up by a TAC engineer (*Working*). During the investigation of the issue, it might be that the TAC engineer is waiting for further information like RMA details, a system dump or tcpdump traces (*Waiting Customer Information*). At a certain moment the problem has been identified and can be rectified (*Waiting Solution Verification*) and closure of the case (*Pending Closure*). Sometimes the problem needs a solution in the RiOS software itself to engineering (*Waiting for Engineering Fix, Waiting Software Release*) but at the end it will be all resolved (*Closed*).

There are other states, for example when a case is opened on a device without a valid support contract (*Waiting Entitlement*) or when an RMA has been issued (*Waiting Hardware Replacement*).

Appendix A. Jargon

- Admission Control (operational): The state for the optimization service which prevents it from the optimization of new TCP sessions based on TCP session limits, TCP bandwidth limits or memory utilization.
- Asymmetric Routing (networking): The situation where the traffic between client and server takes a different path compared with the traffic between server and client.
- Auto-Discovery (optimization): The method of discovering peer Steelhead appliances in the path between the client and the server.
- Auto-Discovery Probe (optimization): The TCP Option in the TCP SYN packet which is used in the Auto-Discovery method.
- By-pass card (hardware): A network card with one or more LAN and WAN interface pairs.
- Central Management Console: An application to manage the configuration of Steelhead appliances, to monitor the behaviour of Steelhead appliances, Interceptor application and Steelhead Mobile Controllers.
- CFE (networking): Client File Engine, internal name for Client-side Steelhead Appliance.
- CLI (operational): The Steelhead appliance command-line interface.
- Client-side Steelhead appliance (networking): The Steelhead appliance closest to the client setting up the TCP session.
- CMC (operational): See Central Management Console.
- Cold Transfer (optimization): An optimized file transfer with data which is not seen before by the Steelhead appliance and thus needs to be labeled and transferred to the other Steelhead appliance.
- Connection Forwarding (optimization): The method for two or more Steelhead appliances to exchange information about TCP sessions optimized by one of the Steelhead appliances.
- Connection Pool (optimization, networking): A set of pre-setup TCP sessions setup between two Steelhead appliance in-path interfaces which can be converted into Inner Channels.
- Correct Addressing (optimization, networking): The WAN visibility method of the inner channel where the IP addresses are the IP addresses of the Steelhead appliance in-path interfaces and the TCP port numbers used are TCP port numbers defined by the client-side and server-side Steelhead appliances.
- CSH (networking): See Client-side Steelhead appliance.
- Data store (optimization): The dictionary with the reference labels and frames.
- Data store ID (optimization): The identifier of a data store.
- Data Store Synchronization (optimization): The method for two Steelhead appliances to synchronize their data store so that learned references are not lost if one of the data stores vanishes.
- Fault Tolerant Segstore (hardware): A data store storage implementation which allows for failure of the underlying storage devices.
- Fixed Target (optimization): The method of setting up an inner channel by defining the IP address of the peer Steelhead appliance in the in-path rule.
- Frame (optimization): A string of characters.
- FTS (hardware): See Fault Tolerant Segstore.

- Full Transparency (optimization, networking): The WAN visibility method of the inner channel where the IP addresses and TCP port numbers used are the IP addresses and TCP port numbers of the client and server.
- GUI (operational): The Steelhead appliance web-interface.
- Inner channel (optimization, networking): The part of an optimized TCP session between two Steelhead appliances.
- In-Path Deployment (networking): A deployment method where the Steelhead appliance is located between two network devices.
- In-path Interface (operation, networking): A virtual interface consisting of a physical LAN and WAN interface on a by-pass card.
- In-Path Rules (optimization): The list of rules traversed to determine what to happen with a new TCP session seen by a naked SYN packet.
- Label (optimization): A unique identifier for a frame.
- Link State Protocol (networking): A method for an interface in a pair of LAN and WAN interfaces to follow the network link state of its peer.
- LSP (networking): See Link State Protocol.
- MFE (networking): Middle File Engine, internal name for Middle Steelhead Appliance.
- Middle Steelhead appliance (networking): Any Steelhead appliance in between the client-side Steelhead appliance and the server-side Steelhead appliance.
- MSH (networking): See Middle Steelhead appliance.
- Naked SYN (optimization): A TCP SYN packet without an auto-discovery probe attached to it.
- OOB Splice (optimization): See Out-of-Band splice.
- Optimization service (optimization): The process on the Steelhead appliance which does do the optimization of the network traffic.
- Out-of-Band Splice (optimization): A control TCP session between two in-path interface IP addresses.
- Out of Path Deployment (networking): A deployment method where the Steelhead appliance is not located in the path, virtual or non-virtual, of the traffic.
- Outer channel (optimization, networking): The part of an optimized TCP session between a Steelhead appliance and the client or the server.
- Peering Rule (optimization): A list of rules traversed to determine what to happen with a new TCP session identified by a SYN+ packet.
- Port Transparency (networking): The WAN visibility method of the inner channel where the IP addresses used are the IP addresses of the Steelhead appliance in-path interfaces and the TCP port numbers used are the TCP port numbers of the client and server.
- Reference (optimization): A frame in the data store, identified by a label.
- RiOS (optimization): The Riverbed Optimization System.
- RSP (virtualization): See Riverbed Services Platform.
- RSP image (virtualization): The RSP software as provided by Riverbed to be installed on the Steelhead appliance.
- RSP package (virtualization): A software image of the virtual machine as created by the RSP Package Generator.

- RSP Package Generator (virtualization): The tool to convert a VMware based virtual machine into an RSP package.
- RSP service (virtualization): The RSP software running on the Steelhead appliance.
- RSP slot (virtualization): The virtual machine on the RSP service.
- Riverbed Services Platform (virtualization): The virtualization platform on the Steelhead appliances.
- Scalable Data Referencing (optimization): The dictionary methods that the Steelhead appliance applies to optimize data.
- SDR (optimization): See Scalable Data Referencing.
- SDR-A (optimization): SDR Advanced, where the optimization service decided to use normal SDR or SDR-M based on the disk load on the Steelhead appliance.
- SDR-M (optimization): SDR Memory, where the optimization service only uses a memory based data store and not a disk based data store.
- Secure Sockets Layer (optimization): An encrypted transport layer.
- Secure vault (hardware): An encrypted disk partition on the Steelhead appliance.
- Segstore (optimization): See Data store.
- Server-side Steelhead appliance (networking): The Steelhead appliance closest to the server the TCP session is being setup to.
- SFE (networking): Server File Engine, internal name for Server-side Steelhead Appliance.
- SH (networking): Abbreviation for Steelhead appliance.
- SHM (optimization): The Steelhead Mobile software.
- Simplified Routing (networking, optimization): Routing decisions which are based on data learned from packets going through an in-path interface.
- SMC (hardware): The Steelhead Mobile Controller.
- SPort (optimization): See Optimization service.
- SSH (networking): See Server-side Steelhead appliance.
- SSL (optimization): See Secure Sockets Layer.
- SYN+ packet (optimization): A TCP SYN packet with an auto-discovery probe in it.
- SYN/ACK+ packet (optimization): A TCP SYN/ACK packet with an auto-discovery probe in it.
- Virtual In-Path Deployment (networking): A deployment method where the Steelhead appliance is not located in between two network devices but gets the interesting traffic forwarded by a WCCP router, via PBR or via a Interceptor appliance.
- WAN visibility (networking): The IP address and TCP port number used for the inner channel of the optimized TCP session.
- Warm Transfer (optimization): An optimized file transfer with data which has been seen before by the Steelhead appliance and thus only the references need to be send to the other Steelhead appliance.

Appendix B. Troubleshooting workflow for network performance related issues

This appendix describes the steps to be taken to isolate network performance related issues.

B.1. Step One: Does the TCP session get optimized?

If the optimized TCP session gets setup properly, it will be displayed in the list of Current Connections. If the TCP session doesn't show there, this chapter needs to be followed.

Most of the troubleshooting for this step can be done via the GUI or by checking the output of the *tcpdump* tool.

The setup of an optimized TCP session consists of three phases: The auto-discovery phase, the OOB channel and the setup of the inner and outer channels phase.

B.1.1. Phase A: Does the auto-discovery phase complete?

During the auto-discovery phase, the client-side Steelhead (CSH) should intercept the naked SYN from the client on the LAN interface, attach the auto-discovery probe and send the SYN+ out on the corresponding WAN interface.

Why was the naked SYN not seen?

- The traffic towards the server does not go via the CSH.
  - Are there other paths from the client's LAN switch to the WAN router.
  - Check the MAC tables on the various devices to confirm that the traffic towards the CSH in-path IP addresses and the gateway behind the CSH are going out via the same port.
  - Are there any other CSHs there?
- In case of a virtual in-path WCCP/PBR deployment, the WCCP/PBR router does not forward the traffic towards the CSH.
  - Check the redirection access-lists on the WCCP/PBR routers to confirm that the client/server IP addresses are matching.
  - Check that the incoming interface for the client traffic does have the redirection statements configured.
  - Check that the WCCP routers and the CSH can see each other.

Why did the auto-discovery probe not get attached?

- The optimization service is not running.
  - It can have crashed, is turned off, didn't want to start.
  - There is a connection-forwarding failure or fail-over mode.
- The naked SYN showed up on the WAN interface for in-path deployments.

  - LAN/WAN cable switched.
- The in-path interface is not enabled for optimization.
  - Check the settings under Configure -> Optimization -> General Settings.
- A pass-through in-path rule prevents it.
  - Check the in-path rules.
- The CSH is in connection-based admission control.
  - Confirm that there has been no admission control alarms.
- The client/server IP addresses are in the asymmetric routing table.
  - Check the asymmetric routing table and flush it if necessary.
- There is no space in the TCP header to store the auto-discovery probe.
  - Check the kernel logs this with the command `support show kernel-logs`.

The SYN+ should arrive on the server-side Steelhead (SSH).

Why does the SYN+ not get seen?

- Routing issues between CSH and SSH.
  - Run traceroute and confirm that the traffic goes through the SSH.
- A naked SYN gets seen!
  - Auto-discovery probe gets stripped by firewall.
  - Compare TCP headers.
  - Third SYN from the client? A Firewall doesn't like the auto-discovery probe and drops the SYN+.
- Traffic passed a NAT gateway which changes source IP address and maybe also stripped off the auto-discovery probe.
  - Try fixed-target rules

In case of Enhanced Auto-Discovery, the SSH will send a SYN/ACK++ to the CSH and forwards the SYN+ via the corresponding interface and waits for a SYN/ACK.

Why does the SYN/ACK++ not get seen on the SSH?

- The in-path interface gateway is set to the LAN side which doesn't send it back via this in-path interface.
  - Check in-path routing table.
  - Check the LAN side interface.
- In-path interface cannot resolve the gateway IP address.
  - Inspect ARP table.
  - Try to ping the default gateway from the in-path interface IP address.
- The optimization service is not running.

  - It can have crashed, is turned off, didn't want to start.
  - There is a connection-forwarding failure or fail-over mode.
- The in-path interface is not enabled for optimization.
  - Check the settings under Configure -> Optimization -> General Settings.
- A pass-through peering rule prevents it.
  - Check the peering rules.
- The SSH is in connection-based admission control.
  - Confirm that there has been no admission control alarms.

Why does the SYN/ACK++ not get seen by the CSH?

- Routing issues between the CSH and SSH.
  - Run traceroute and confirm that the traffic goes through the CSH.
- A naked SYN/ACK arrives.
  - Auto-discovery probe gets stripped by a firewall
  - Compare TCP headers.
- Asymmetric routing bypasses the CSH.
  - You should see a TCP RST from the client.

Why does the SYN+ not get seen leaving the SSH?

- No idea. Does this happen?

Why does the SYN/ACK not get seen?

- The server doesn't answer.
  - The server is off.
  - The traffic is blocked by firewall.
- The traffic from the server to the client does not go via the SSH.
  - This is a case of server-side asymmetry.

The SSH will send a SYN/ACK+ towards the CSH.

Why does the SYN/ACK+ not get send?

- The in-path interface gateway is set to the LAN side which doesn't send it back via the WAN interface.
  - Check in-path routing table.
  - Check the LAN interface.
- In-path interface cannot resolve the gateway IP address.
  - Inspect ARP table.
  - Try to ping the default gateway from the in-path interface IP address.

Why does the SYN/ACK+ not get received by the CSH?

- Asymmetric routing bypasses the CSH.
    - You should see a TCP RST from the client on the LAN side.
- In case of Enhanced Auto-Discovery, a firewall blocks the SYN/ACK+ because it has already seen it as the SYN/ACK++.
- A naked SYN/ACK arrives
    - Auto-discovery probe gets stripped by a firewall.
    - Compare TCP headers.

At the end of the auto-discovery phase, the CSH will not send a final ACK of the TCP handshake.

B.1.2. Phase B: Do the OOB channel and the Connection Pool get setup?

After the auto-discovery phase, the CSH will setup an OOB channel to the SSH. This is a TCP session from the CSH in-path interface to the SSH in-path interface on TCP port 7800.

Why does the OOB TCP session not get setup?

- Routing issues between the CSH and SSH.
    - Run ping and traceroute to see check the paths.
    - Telnet from CSH in-path IP address to SSH in-path IP address on TCP port 7800.
    - Run tcpdump to see which part doesn't work.
- Timeout in the setup of the OOB channel.
    - Traffic towards SSH is blocked.

After the OOB channel is setup, the CSH will setup the Connection Pool to the CSH. This is a set of 20 TCP sessions from the CSH in-path interface to the SSH in-path interface on TCP port 7800.

Why do the Connection pool TCP sessions not get setup?

- The CSH in-path rules state that the inner channel WAN visibility is Port of Full Transparency.
- Routing issues between the CSH and SSH.
    - Run ping and traceroute to see check the paths.
    - Telnet from CSH in-path IP address to SSH in-path IP address on TCP port 7800.
    - Run tcpdump to see which part doesn't work.
- Timeout of the setup of the TCP Connections.
    - Traffic towards SSH is blocked.

B.1.3. Phase C: Does the inner channel get established?

After the auto-discovery phase, the CSH will setup an Inner Channel.

For the Correct Addressing WAN visibility, this will happen by the conversion of a TCP session in the Connection Pool into an inner channel.

Why does the conversion from TCP session of the Connection Pool into an inner channel fail?

- Setup packet doesn't arrive at SSH.
  - The TCP sessions could have timed out by a firewall in the network. Reduce TCP Keep-Alive interval for the inner channels.

For the Port Transparency WAN visibility, this will be a normal TCP session from the CSH in-path interface IP address to the SSH in-path interface IP address, but with the TCP port of original naked SYN.

Why does the inner channel not get setup?

- Routing issues between the CSH and SSH.
  - Run ping and traceroute to see check the paths.
  - Take traces to see which part doesn't get setup.
- Traffic towards SSH is blocked.
  - Timeout of the setup of the inner channel.
- A naked SYN arrives at the SSH.
  - Transparency options get stripped by a firewall.

For the Full Transparency WAN visibility, this will be a TCP session from the original client IP address to the original server IP address, but with the TCP port of original naked SYN.

Why does the inner channel not get setup?

- A firewall doesn't allow the "old" SYN with different TCP sequence numbers.
  - Use Full Transparency with Firewall Reset to reset the state tables on the firewall.

Once the TCP session is setup, the CSH will send the splice setup message and the inner channel is setup.

B.2. Step Two: Does TCP optimization and data reduction work?

B.2.1. Issue one: The inner channel doesn't stay up.

As long as the client and the server don't close the TCP session, the inner channel should stay up.

- The TCP connection times out.
  - The inner channel doesn't support fragmentation. If there is a gateway in the path which doesn't support for the MTU size of the in-path interface, we expect to see an ICMP Fragmentation Required packet with the right MTU size. If that packet doesn't get delivered, the TCP session will time out.
- The TCP session gets reset.
  - A firewall in between the CSH and SSH has noted that the final ACK of the setup of the original TCP session didn't arrive. After a timeout the firewall will send a TCP RST to the client and server, intercepted by the CSH and SSH which will terminate the TCP session. This will affect long-lived TCP sessions most.

- With the Full Transparency or Port Transparency WAN visibility, an IPS/IDS will find out that the protocol spoken on the inner channel isn't the protocol expected and it will terminate the TCP session.

B.2.2. Issue two: Traffic is not suited for optimization.

The traffic must be suitable for optimization.

- Bad data reduction.
  - Encrypted traffic. If it is SSL encrypted it could be optimized with SSL pre-optimization. If anything else, no luck. Check the documentation of the application to see if this can be disabled.
  - Compressed traffic. Check the documentation of the application if this can be disabled.
- Bad traffic scenario.
  - If the traffic between the client and server is not a stream of lots of traffic in burst but more a ping-pong, the neural framing algorithm will cause extra delay here.

B.2.3. Issue three: Network issues

If the network isn't happy, the optimized traffic won't perform well neither.

- Speed / duplex mismatch.
- Congestion.
  - QoS not enabled.
  - QoS bandwidth misconfigured.
- Large Fat Network.
  - WAN link doesn't get filled. Use HS-TCP or MX-TCP.
- WAN bandwidth admission control.

B.2.4. Issue four: Steelhead related issues

And it could just be that the SH is not performing well.

- CPU load, paging.
- Broken disk.
- High disk utilization: SDR Cost / SDR Latency / Disk Load 3.
- QoS configuration wrong.
- Optimization service not happy.

B.3. Step Three: Does latency optimization work

If the latency optimization doesn't work, then it's all depending on the protocol being spoken:

SMB latency optimization:

- Failed oplock acquisition.
- Write-behind cancellation.
- Unsupported SMB version.
- Unsupported SMB server.
- If the data is signed and the Active Directory integration failed.
- Just an inefficient approach: Copy a roaming profile consisting of a lot of directories and small files instead of copying one large file and decompressing it locally.

MAPI latency optimization:

- If the data is encrypted and the Active Directory integration failed.
- For MAPI-OA, is the SSL optimization working?

HTTP latency optimization:

- Latency optimization discontinues for the rest of the requests when one fails.
- If Full Transparency is used, every new HTTP session needs to be auto-discovered and inner channel being setup before the requests are forwarded.
- + ?

NFS latency optimization:

- Unsupported NFS version.
- Unsupported authentication encryption mechanism.

MS-SQL:

- Unsupported MS-SQL version.

Lotus Notes:

- The traffic is encrypted and the latency optimization is not configured for it.

SSL pre-optimization:

- The maximum number of optimized TCP sessions being SSL encrypted is about 40-50% of the hardware specified amount of TCP sessions the machine is capable of.