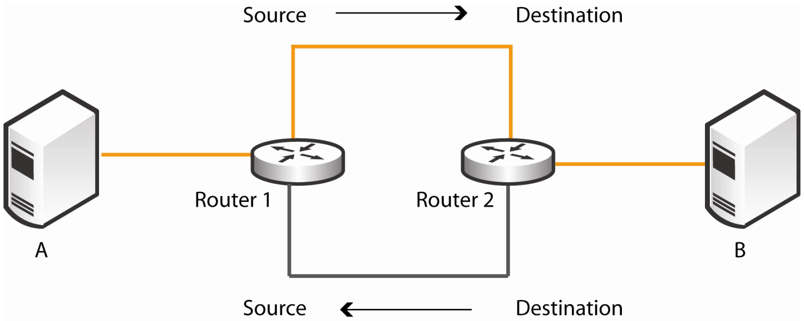
Understanding and Troubleshooting Asymmetric Routing

riverbed

Think fast.™

Understanding Asymmetric Routing

Asymmetric routing is when a packet takes one path to the destination and takes another path when returning to the source. For example, review the following diagram.

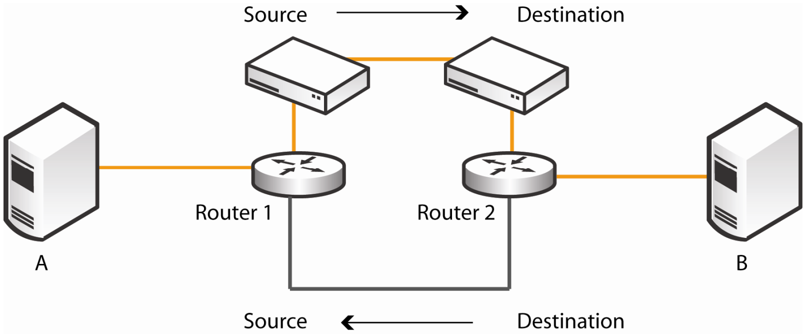


Packets from A to B take one route and packets from B to A take another route. This is not a problem for regular TCP connections because TCP does not care what route a packets takes; it just cares whether or not the packets make it from source to destination.

Asymmetric routing is common within most networks; the larger the network, the more likely there is asymmetric routing in the network.

Asymmetric routing is an undesirable situation for many network devices including, firewalls, VPNs, and Steelhead appliances. These devices all rely on seeing every packet to function properly.

For this network example, add two Steelheads.



In this scenario, packets traveling from A to B would be intercepted by the Steelhead appliances. However, packets traveling from B to A would miss both Steelheads. Before the Riverbed Optimization System (RiOS) Version 3.0, this situation would cause TCP connections to break. To account for this, Riverbed developed an Asymmetric Routing Detection feature for the 3.0 release.

Understanding the Asymmetric Routing Detection Feature

The Riverbed Asymmetric Routing Detection feature enables Steelhead appliances to detect the presence of asymmetry within the network. Asymmetry is detected by the client-side Steelhead.

Once detected, the Steelhead will pass through asymmetric traffic allowing the TCP connections to continue to work. One caveat is that the first TCP connection for a pair of addresses can be dropped. This is due to the fact that during the detection process the Steelhead appliances have no way of knowing that the connection is asymmetric. However, once detected, an entry is placed in the asymmetric routing table and any subsequent connections from that IP pair will be passed through. This pass through will continue to happen until the timer expires on the table entry or the appliance detects that asymmetry is no longer present.

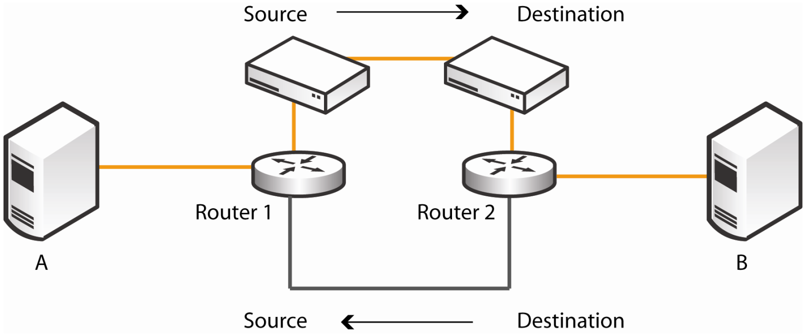
There are several asymmetry scenarios that can occur and each scenario has a different packet sequence that allows the Steelhead to identify it. These scenarios are Complete Asymmetry, Server-Side Asymmetry, Client-Side Asymmetry, and Multi SYN Retransmit. These scenarios are described in detail in the following sections.

Asymmetric Scenarios

The following sections describe the various types of network asymmetry.

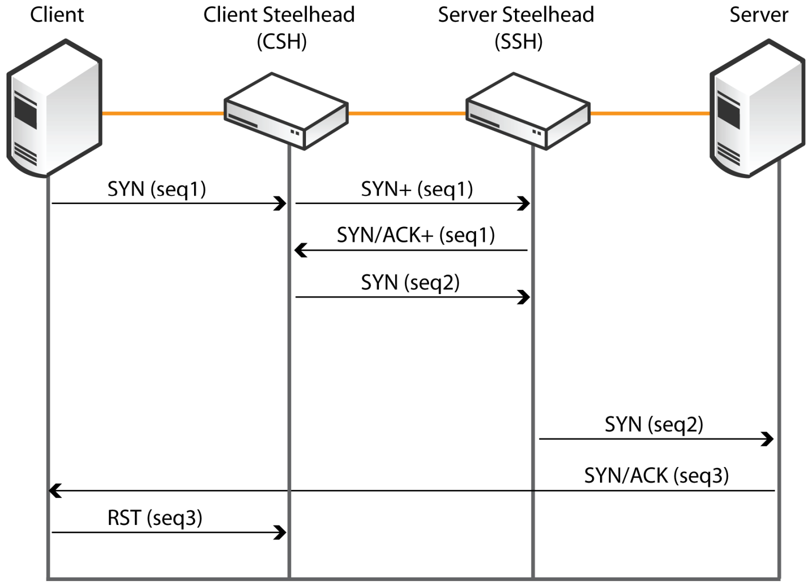
Complete Asymmetry

In the complete asymmetry scenario, packets traverse both Steelheads going from client to server but bypass both Steelheads on the return path.



Packet Map

The following is a detailed description of the packets that allow the client-side Steelhead to detect the asymmetry and pass through the connection.



In this packet map, the Client sends a SYN packet to the Server. The packet is intercepted by the CSH (Client-side Steelhead) which then forwards a SYN+ frame to the server and the SYN+ is intercepted by the SSH (Server-side Steelhead) which immediately responds with a SYN/ACK+ frame. This triggers the creation of the inner channel, where the SYN, SYN/ACK, and ACK frame are exchanged between Steelheads with the second set of sequence numbers. Once the inner channel is created, the SSH forwards a SYN frame with the third set of sequence numbers and the server responds with a SYN/ACK. However, since there is another route back to the client this packet bypasses both Steelheads. Thus, the client receives an invalid SYN/ACK and resets the connection. When the CSH sees this reset with an invalid sequence number on it, this packet triggers the asymmetric routing detection and all subsequent packets are passed through. The Asymmetric Routing (AR) table is updated with an entry for this IP address pair and the alarm is triggered.

Log Entry and Reason Code

The following log entry appears when asymmetry is detected in a complete asymmetry scenario.

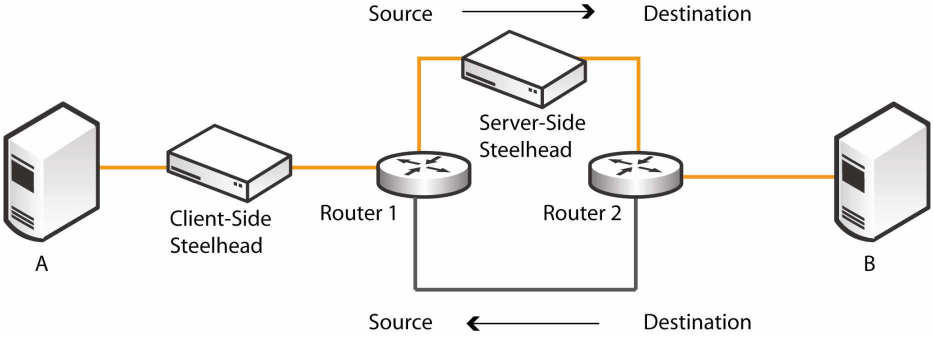
- **Log Entry:** `Sep 5 11:16:38 gen-sh102 kernel: [intercept.WARN] asymmetric routing between 10.11.111.19 and 10.11.25.23 detected (bad RST)`

The following reason code appears for the asymmetric routing table entry.

- **Reason Code in AR Table:** bad-RST

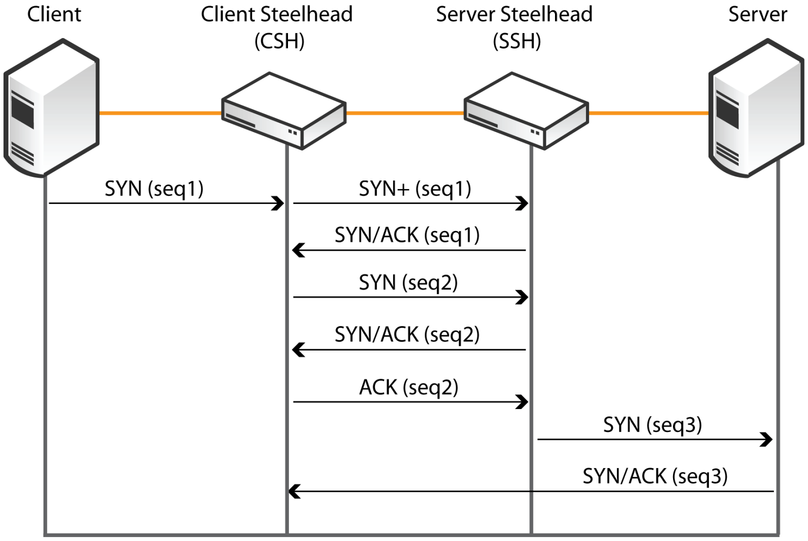
Server-Side Asymmetry

In the server-side asymmetry scenario, packets traverse both Steelheads going from client to server but bypass the server-side Steelhead on the return path.



Packet Map

The following is a detailed description of the packets that allow the client-side Steelhead to detect the asymmetry and pass through the connection.



In this packet map, the Client sends a SYN packet to the Server this packet is intercepted by the CSH which forwards a SYN+ frame to the server. The SYN+ is intercepted by the SSH which immediately responds with a SYN/ACK+ frame. This triggers the creation of the inner channel. A SYN, SYN/ACK, and ACK frame are exchanged between Steelheads with the second set of sequence numbers. Once the inner channel is created, the SSH forwards a SYN frame with the third set of sequence numbers and the server responds with a SYN/ACK. However, since there is a route back to the client that bypasses the SSH, the packet arrives at the CSH directly. The CSH sees this as an invalid SYN/ACK. When the CSH sees this invalid SYN/ACK, it triggers the asymmetric routing detection. All subsequent packets are passed through. The AR table is updated with an entry for this IP address pair and the alarm is triggered.

Log Entry and Reason Code

The following log entry appears when asymmetry is detected in a server-side asymmetry scenario.

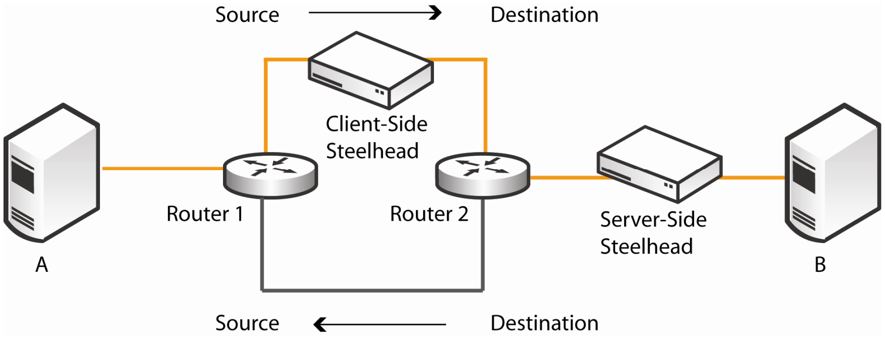
- **Log Entry:** `Sep 7 16:17:25 gen-sh102 kernel: [intercept.WARN] asymmetric routing between 10.11.25.23:5001 and 10.11.111.19:33261 detected (invalid SYN/ACK)`

The following reason code appears for the asymmetric routing table entry.

- **Reason Code in AR Table:** invalid-SYN/ACK

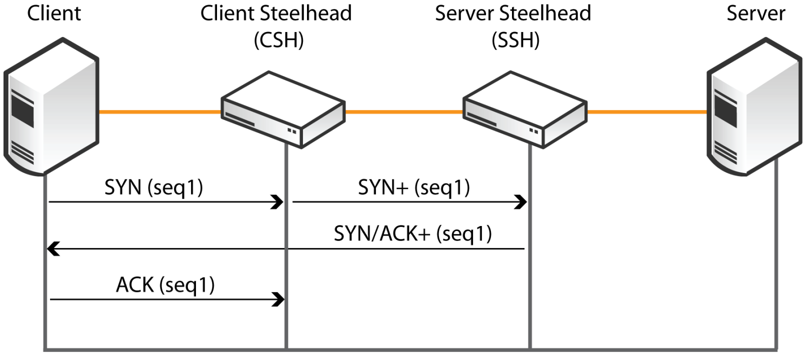
Client-Side Asymmetry

In the client-side asymmetry scenario, packets traverse both Steelheads going from client to server but bypass the client-side on the return path.



Packet Map

The following is a detailed description of the packets that allow the client-side Steelhead to detect the asymmetry and pass through the connection.



In this packet map the Client sends a SYN packet to the Server. This packet is intercepted by the CSH which forwards a SYN+ frame on to the server. The SYN+ is intercepted by the SSH which immediately responds with a SYN/ACK+ frame. Since there is a route that bypasses the CSH, the SYN/ACK+ goes directly to the Client. The client will ACK this packet and the CSH will see an ACK for a connection for which it did not receive a SYN/ACK+. When the CSH sees this ACK, it triggers the asymmetric routing detection. All subsequent packets are passed through. The AR table is updated with an entry for this IP address pair and the alarm is triggered.

Log Entry and Reason Code

The following log entry appears when asymmetry is detected in a client-side asymmetry scenario.

- **Log Entry:** `Sep 7 16:41:45 gen-sh102 kernel: [intercept.WARN] asymmetric routing between 10.11.111.19:33262 and 10.11.25.23:5001 detected (no SYN/ACK)`

The following reason code appears for the asymmetric routing table entry.

- **Reason Code in AR Table:** no-SYN/ACK

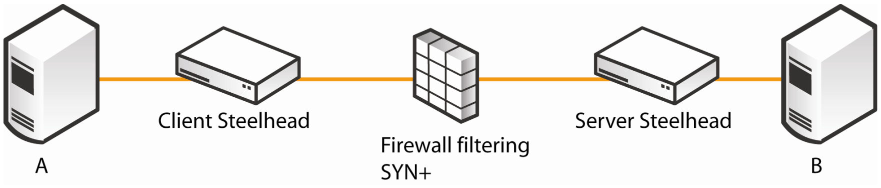
Multi SYN Retransmit

Multi SYN retransmit can happen for several reasons, such as the server is not reachable, probe packets are being filtered, and so on. There are two scenarios that display in the Asymmetric Routing table, “probe-filtered (not AR)” and SYN-rexmit (confirmed AR).

The first scenario (probe-filtered), occurs when the CSH sends out multiple SYN+ frames and does not get a response. At that point, an entry is placed in the table with a short timeout (the default is 5 seconds). The CSH then sends out a normal SYN packet and if it receives a SYN/ACK in response from the server, it marks this connection in the table as “probe-filtered (not AR)” and increases the timer to 5 minutes. There is no alarm or email generated for this scenario as it is not a true asymmetric situation. The probe filtered scenario typically happens when there is no server-side Steelhead, as would be the case if a customer put a Steelhead at the edge of their network and didn’t add pass-through rules for the Internet.

The second scenario occurs when the CSH receives multiple SYN retransmits from a client and does not see a SYN/ACK packet from the destination server. At that point, an entry is placed in the table with a short timeout (the default is 5 seconds). The CSH then receives an ACK for that connection which implies that it missed the SYN/ACK. This will cause asymmetry to be detected and the entry in the table will change to “SYN-rexmit (confirmed AR)” and the timer will be increased to 24 hours.

Probe Filtered

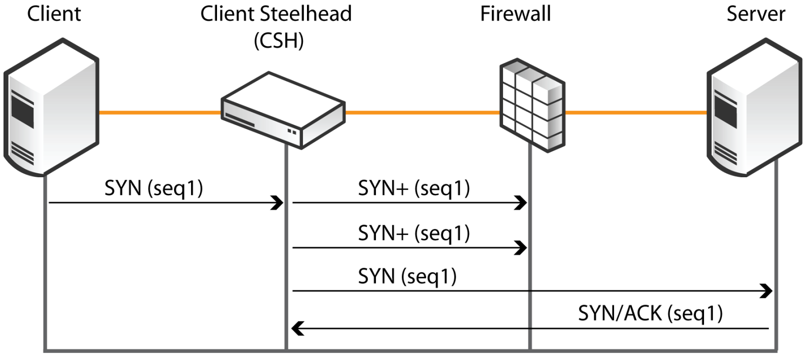


SYN Rexmit (confirmed-AR)

Any number of configurations can result in this situation so there is no one diagram that describes this scenario—any of the previous diagrams could apply.

Packet Map - Probe Filtered

The following is one example of the packet map for the probe-filtered scenario.



In this packet map, the Client sends a SYN to the Server. This packet is intercepted by the CSH which forwards a SYN+ packet. This packet is filtered by the firewall between the Steelheads. The CSH will send another SYN+ and then send a regular SYN packet. When the CSH receives a SYN/ACK to the regular SYN packet, it marks this as probe filtered (not asymmetric routing) and sets the timeout to 300 seconds within the table. Since this is not a true asymmetric routing condition, no alarm is raised and no emails are generated. An entry is placed in the log.

Log Entry and Reason Code

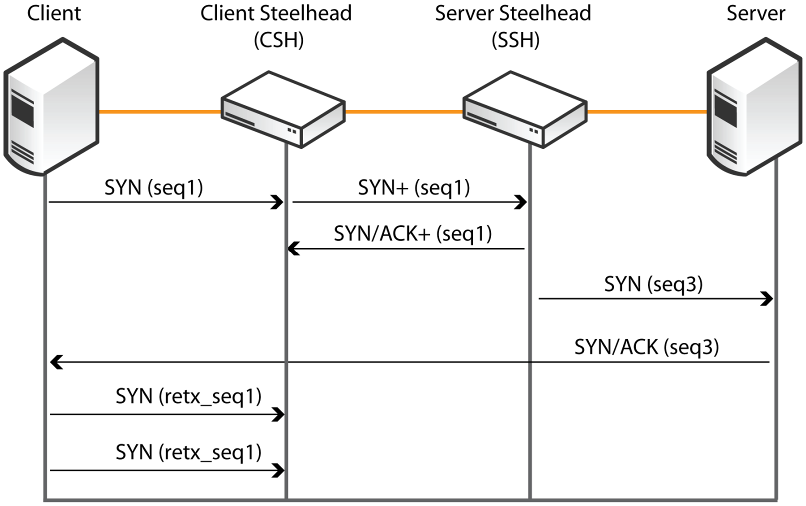
The following log entry appears when asymmetry is detected in a client-side asymmetry scenario.

- **Log Entry:** `Sep 13 20:59:16 gen-sh102 kernel: [intercept.WARN] it appears as though probes from 10.11.111.19 to 10.11.25.23 are being filtered. Passing through connections between these two hosts.`

The following reason code appears for the asymmetric routing table entry.

- **Reason Code in AR Table:** probe-filtered(not-AR)

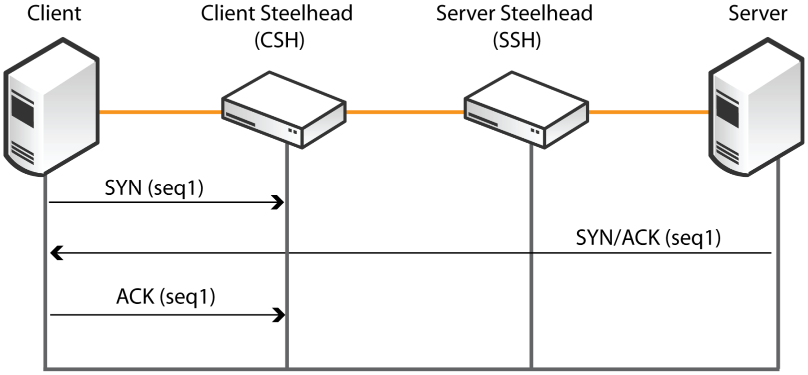
SYN Rexmit (confirmed-AR)



In this packet map, the Client sends a SYN packet to the Server this packet is intercepted by the CSH. The CSH then forwards on a SYN+ packet. The inner channel is established and a SYN with the third set of sequence numbers is sent to the server. The SYN/ACK bypasses the Steelheads and arrives at the Client. For some reason, possibly a firewall, that packet is ignored. The Client continues to send SYN packets which cause the CSH to add an entry in the table with a short timeout. When the Client sees the ACK for a connection that it did not see a SYN/ACK for, it updates the timer to 24 hours and changes the reason code to “SYN-rexmit (confirmed AR).” Again this is just one scenario that may cause this detection to occur. At this point, the alarm is raised and an email is generated as this is a true AR condition.

The SYN/ACK(seq2) is dropped because the client is running a personal firewall.

An AR table entry is added and a second connection is attempted.



Log Entry and Reason Code

The following log entry appears when asymmetry is detected in a client-side asymmetry scenario.

- **Log Entry:**
  ```
  Sep 13 21:28:34 gen-sh102 kernel:
  [intercept.WARN] asymmetric routing still exists for
  10.11.111.19 10.11.25.23 SYN-rexmit 4 (no SYN/ACK)
  ```

The following reason code appears for the asymmetric routing table entry.

- **Reason Code in AR Table:** SYN-rexmit(confirmed-AR)

Asymmetric Routing Commands

You can use the following commands to detect and analyze asymmetric routing.

- **show in-path asym-route-tab**

  This command shows the asymmetric route table. The table contains any asymmetric routes that currently exist. It will have the source IP, destination IP, reason code, and timeout.

  This is also viewable through the Management Console. In 4.1 and earlier, go to Setup >Advanced Networking >Asymmetric Routing. In 5.0 and later, go to Configure > Networking > Asymmetric Routing.

  Sample output:

  ```
  gen-sh102 (config) # sho in-path asym-route-tab

  Format: [IP 1]    [IP 2]         [reason]    [timeout(s)]

  10.11.111.19      10.11.25.23    no-SYNACK   770
  ```

- **show in-path ar-circbuf**

  This command shows the asymmetric route circular buffer. The buffer contains all the asymmetric routes that have been detected. This is a circular buffer and wraps after period of time. One thing to note is the reason in the circular buffer can be set to artable-match if a new TCP connection is created for a pair of IP addresses that already have an AR table entry. This reason code only exists in the circular buffer. The buffer is set up with Source IP:Source Port, Destination IP:Destination Port, and reason code.

  Sample output:

  ```
  gen-sh102 (config) # sho in-path ar-circbuf

  Format: [IP 1]:[port 1]  [IP 2]:[port 2]  [reason]

  10.11.111.19:33280   10.11.25.23:5001 artable-match

  10.11.111.19:33278   10.11.25.23:5001 no-SYNACK

  10.11.111.19:33277   10.11.25.23:5001 SYN-rexmit

  10.11.111.19:33271   10.11.25.23:5001 artable-match

  10.11.111.19:33270   10.11.25.23:5001 SYN-rexmit
  ```

Configuration Commands

You can use the following commands to configure asymmetric routing.

- **in-path asymmetric routing detection enable**

  This command enables asymmetric route detection. To disable this feature, use the "No" counterpart to this command. Asymmetric route detection is enabled by default. If you disable this feature, it will effectively break any asymmetrically routed TCP connections. No logging, alarms, or emails will be created when this is disabled, thus it is not recommended that this feature be disabled.

  The configuration of this feature is also available through the Management Console. In 4.1 and earlier, go to Setup >Advanced Networking >Asymmetric Routing. In 5.0 and later, go to Configure > Networking > Asymmetric Routing. There is a check box to enable and disable this feature.

- **in-path asymmetric routing pass-through enable**

  This command enables and disables the pass-through feature for asymmetric routing. To disable this feature, use the "No" counterpart to this command. If this feature is disabled, asymmetrically routed TCP connections are still detected and a warning message is logged, however, the connection is not passed-through and no alarm or email is sent. This would be used if a customer wants to ensure no connections are passed-through the Steelheads but still wants logging when asymmetric routes occur.

  The configuration of this feature is also available through the Management Console. In 4.1 and earlier, go to Setup >Advanced Networking >Asymmetric Routing. In 5.0 and later, go to Configure > Networking > Asymmetric Routing. There is a check box to enable and disable this feature.

- **in-path asym-route-tab flush**

  This command clears all routes from the asymmetric routing table. This is also available through the Web interface. In 4.1 and earlier, go to Setup >Advanced Networking >Asymmetric Routing. In 5.0 and later, go to Configure > Networking > Asymmetric Routing. Check the box next to every route in the table and click the **Remove Selected Entries** button.

- **in-path asym-route-tab remove** <*address-pair* >

  This command clears a single route from the asymmetric routing table. It requires the specification of an address pair that exists in the table, for example 1.1.1.1-2.2.2.2.

  This is also available through the Web interface. In 4.1 and earlier, go to Setup >Advanced Networking >Asymmetric Routing. In 5.0 and later, go to Configure > Networking > Asymmetric Routing. Select the check box next to the route that you desire to delete. Click the Remove Selected Entries button.

Troubleshooting Tools

TCP Dump

You can use TCP Dump on the client-side Steelhead to verify the packet sequence that is causing the asymmetric route detection. You can take traces on the LAN and WAN ports of the Steelhead and, based on the packet maps, look for the packet sequence that is expected for the type of warning message that was in the log. The easiest way to do this is to clear the route, and start the traces. Stop the traces immediately following the triggering of the alarm. Then, filter the trace utilizing the addresses that where discovered in the AR table. To get a narrower scope for the filter, you can utilize the TCP port pair that appears in the AR circular buffer.

Here are sample **tcpdump** commands.

This command catches all packets on the WAN interface, sourced from or destined to 10.0.0.1, and with a source/destination TCP port of 80.

```
tcpdump -i wan0_0 host 10.0.0.1 port 80
```

To filter just SYN, SYN/ACK, and reset packets, you can use this filter. Remember, this will not show you ACK packets but it can be useful if the link is saturated with traffic and the traces are filling quickly. The following command is using the **-i** parameter to specify the interface and the **-w** parameter to write to a file:

```
tcpdump -i wan1_0 'tcp[tcpflags] & (tcp-syn|tcp-fin|tcp-rst)
= 0' -w lookingforasymwan
```

Use the following **tcpdump** command to write to three files continuously with a size of 10 MB each. As with the previous command, this one uses the **-i** parameter to specify the interface and the **-w** parameter to write to a file. In the command the **-C** option gives the size of each file in megabytes. The **-W** option specifies the number of files you want to create. When one file reaches 10 MB, it starts writing the next file, when it gets to the third file it wraps back to the first file. A number is appended to the name given for the file. The **/var/tmp** directory is a good place to write these files.

```
tcpdump -i lan0_0 -s 300 -C 10 -W 3 –w /var/tmp/lookingforasym
```

Trace Route

Traceroute is a handy tool to discover what path a packet is taking from client to server and from server to client. Simply access the client and perform a **traceroute** command with the IP address of the server. Then, from the server perform a **traceroute** with the IP address of the client. The command is slightly different based on the operating system. Here are some samples of the command.

- **trace 10.11.x.x** (Cisco router)
  This is the variation for a Cisco router. This may be needed if the system administrator does not have access to the client. They can use this on the router nearest to the client. Obviously, this is the Cisco version, however, most routers have a trace functionality.

- **traceroute x.x.x.x** (Linux machine)
  This is the variation for a Linux box.
- **tracert x.x.x.x** (Windows machine)
  This is the variation for the Windows operating system.

Sample output:

- Client's Address: 10.1.0.2
- Server's Address: 10.0.0.4

```
client# traceroute 10.0.0.4
Type escape sequence to abort.
Tracing the route to 10.0.0.4
1 10.1.0.1 4 msec 0 msec 4 msec
2 10.0.0.2 4 msec 4 msec 0 msec
3 10.0.0.3 4 msec 4 msec 0 msec
4 10.0.0.4 4 msec 4 msec 0 msec

server# traceroute 10.1.0.2
Type escape sequence to abort.
Tracing the route to 10.1.0.2
1 10.0.0.6 4 msec 0 msec 4 msec
2 10.0.0.5 4 msec 4 msec 0 msec
3 10.1.0.1 4 msec 4 msec 0 msec
4 10.1.0.2 4 msec 4 msec 0 msec
```

The command prints a list of the routers that were used, ending with the device that was the target of the trace route. You are looking to see if the packets take a different route from client to server then from server to client. As you can see from the example, the packet took a different route when tracing from client to server and then tracing from server to client. One thing to note here is that asymmetry between Steelheads is acceptable; asymmetry that causes a Steelhead to be bypassed should be corrected.

Determining Routes

If the trace route does not give you enough information, most routers can determine what interface or path the packet will take based on a destination. Cisco provides a command in their routers to show what interface the packet would be sent out to reach the destination address.

Example output:

```
tr3640#sho ip route 10.11.25.23

Routing entry for 10.11.25.0/24

Known via "static", distance 1, metric 0

Routing Descriptor Blocks:

* 10.11.62.101, via FastEthernet0/1.147

Route metric is 0, traffic share count is 1
```

If the packet is going to be sent out through an interface you did not expect, review the configuration to determine how the router is learning the route to the destination IP. This may be due to a static route or learned route.

Review

- Asymmetry is common in most networks.
- Asymmetry is more common in larger networks.
- Asymmetry is not a problem for TCP because TCP does not care about the path the packet takes.
- There are four asymmetry scenarios: Complete Asymmetry, Server-Side Asymmetry, Client-Side Asymmetry, and Multi SYN Retransmit.
- Asymmetry is an undesirable situation for many network devices (such as firewalls, VPNs, IDS, and Steelheads).
- Starting in RiOS 3.0, the Steelhead software includes a new feature that detects asymmetry and passes traffic through.
- The Steelhead is not causing the problem, it is detecting it.
- If the Asymmetric Detection feature is disabled in the Steelhead, asymmetrically routed TCP connections will break.
- If the asymmetry is not fixed, the TCP connections that are asymmetrically routed will not be optimized.