Host Services and Applications

A host service or application is a router feature that

- checks network connectivity and the route a packet follows to reach a destination,
- maps a host name to an IP address or an IP address to a host name, and
- transfers files between routers and UNIX workstations.

Prerequisites

Install the relevant optional RPM package before using host services or applications.

HTTP clients

A HTTP client is a network utility that

- transfers files and data from HTTP servers to devices over a network using the HTTP protocol,
- enables configuration of connection, security, and transport parameters via **http client** command, and
- and supports advanced features such as SSL/TLS options, source interface selection, and protocol versioning.

Configurable parameters for HTTP clients

HTTP client is available by default. You can configure http client settings or view and modify the existing settings. To configure the settings, use the **http client** command in XR configuration mode.

```
RP/0/RP0/CPU0:ios#configure
RP/0/RP0/CPU0:ios(config)#http client ?
connection          Configure HTTP Client connection
response            How long HTTP Client waits for a response from the server
                    for a request message before giving up
secure-verify-host  Verify that if server certificate is for the server it is known as
secure-verify-peer  Verify authenticity of the peer's certificate
source-interface    Specify interface for source address
ssl                 SSL configuration to be used for HTTPS requests
tcp-window-scale    Set tcp window-scale factor for High Latency links
```

```
version             HTTP Version to be used in HTTP requests
vrf                 Name of vrf
```

Table 1: Commands used to configure HTTP client settings

Parameters	Description
connection	Configure HTTP Client connection by using either retry or timeout options.
response	How long HTTP Client waits for a response from the server for a request message before giving up.
secure-verify-host	Verify host in peer's certificate. To disable verifying this, you can use the command **http client secure-verify-host disable**
secure-verify-peer	Verify authenticity of the peer's certificate.
source-interface	Specifies the interface for source address for all outgoing HTTP connections. You can enter either an ipv4 or ipv6 address or both.
ssl version	SSL version (configuration) to be used for HTTPS requests.
tcp-window-scale scale	Set tcp window-scale factor for high latency links.
version version	HTTP version to be used in HTTP requests. • 1.0 - HTTP1.0 will be used for all HTTP requests. • 1.1 - HTTP1.1 will be used for all HTTP requests. • default libcurl - will use HTTP version automatically.
vrf name	Name of vrf.

This example shows how to set the tcp window-scale to 8.

```
RP/0/RP0/CPU0:ios(config)#http client tcp-window-scale 8
```

This example shows how to set the HTTP version to 1.0.

```
RP/0/RP0/CPU0:ios(config)#http client version 1.0
```

Note HTTP client uses libcurl version 7.30

TCP

A Transmission Control Protocol (TCP) is a transport layer protocol that

- establishes reliable, connection-oriented communication between devices,
- manages sequencing and acknowledgment of data packets to ensure correct delivery, and
- multiplexes multiple application data streams over a single network connection.

TCP dump file converter

A TCP dump file converter is a network analysis tool that

- converts IOS XR binary dump files into user-friendly formats, such as PCAP or text,
- enables easier viewing and analysis of packet traces during incidents like Non-Stop Routing (NSR) disablement or session flaps, and
- assists network engineers in identifying and resolving issues by making critical packet information accessible.

Additional reference information

When NSR is disabled or a session flap occurs on the system, the TCP process running on an NCS system automatically stores the latest 200 packet traces in binary format within a temporary folder. These traces include details about configured routing protocols and overall network traffic.

Methods to convert and view binary packet trace files

There are two primary methods to convert and view binary packet trace files:

- You can use the **show tcp dump-file <binary filename>** command to view each binary file in text format manually. For more information, see View binary files in text format manually, on page 3.

  This process is time-consuming because you must view each file individually, one after the other.

- You can convert all stored packet traces in binary files into PCAP, text, or both using the **tcp dump-file convert** command. For more information, see Convert binary files to readable format using TCP dump file converter, on page 5.

  This active approach greatly improves the efficiency and ease of packet analysis during network troubleshooting.

Limitations and restrictions for TCP dump file converter

These limitations and restrictions apply to the TCP dump file converter:

- The system stores only the most recent 200 message exchanges. These occur right before session termination, when NSR is disabled, or during a session flap.
- You can view only one binary file in text format using the **show tcp dump-file <binary filename>** command.
- TCP dump files are generated by default for BGP, MSDP, MPLS LDP, and SSH.

View binary files in text format manually

This task enables manual examination of packet trace binary files by displaying their contents in text format, which supports troubleshooting without conversion tools.

This task is useful when you need to check packet traces directly on the device and do not have access to TCP dump file conversion utilities.

Follow these steps to view packet trace binary files in text format:

Procedure

Step 1 Use the **show tcp dump-file list all** command to view the list of packet traces in binary files stored in the tcpdump folder.

Example:

```
RP/0/RP0/CPU0:ios# show tcp dump-file list all
total 1176
-rw-r--r-- 1 root root 5927 Nov 22 12:42 31_0_0_126.179.20966.cl.1700656933
-rw-r--r-- 1 root root 5892 Nov 22 12:42 31_0_0_127.179.35234.cl.1700656933
-rw-r--r-- 1 root root 6148 Nov 22 12:42 31_0_0_149.179.54939.cl.1700656933
-rw-r--r-- 1 root root 5894 Nov 22 12:42 31_0_0_155.179.18134.cl.1700656933
-rw-r--r-- 1 root root 6063 Nov 22 12:42 31_0_0_156.179.25445.cl.1700656933
-rw-r--r-- 1 root root 5860 Nov 22 12:42 31_0_0_161.179.30859.cl.1700656933
-rw-r--r-- 1 root root 5832 Nov 22 12:42 31_0_0_173.179.36935.cl.1700656933
-rw-r--r-- 1 root root 5906 Nov 22 12:42 31_0_0_190.179.25642.cl.1700656933
```

Step 2 Use the **show tcp dump-file <binary filename>** command to view each packet trace binary file in text format.

Example:

```
RP/0/RP0/CPU0:ios# show tcp dump-file 10_106_0_73.179.34849.cl.1707424077 location 0/RP0/CPU0
Filename: 10_106_0_73.179.34849.cl.1707424077

================================================================
Connection state is CLOSED, I/O status: 0, socket status: 103
PCB 0x00007f86bc05e3b8, SO 0x7f86bc05e648, TCPCB 0x7f86bc0c3718, vrfid 0x60000000,
Pak Prio: Medium, TOS: 192, TTL: 1, Hash index: 1593
Local host: 10.106.0.72, Local port: 179 (Local App PID: 11354)
Foreign host: 10.106.0.73, Foreign port: 34849
(Local App PID/instance/SPL_APP_ID: 11354/1/0)

Current send queue size in bytes: 0 (max 0)
Current receive queue size in bytes: 0 (max 0)  mis-ordered: 0 bytes
Current receive queue size in packets: 0 (max 0)

Timer          Starts    Wakeups         Next(msec)
Retrans        103448          8               0
SendWnd             0          0               0
TimeWait            1          0               0
AckHold        106815     106545               0
KeepAlive           1          0               0
PmtuAger            0          0               0
GiveUp              0          0               0
Throttle            0          0               0
FirstSyn            0          0               0

   iss: 161240548   snduna: 163206936   sndnxt: 163206936
sndmax: 163206936   sndwnd: 63104       sndcwnd: 18120
   irs: 3691232436  rcvnxt: 3693473072  rcvwnd: 26099   rcvadv: 3693499171
```

This sample displays only a portion of the actual output. The full output provides additional packet trace details.

You can examine the text details of packet trace binary files directly using the command-line interface without converting them to other formats.

Convert binary files to readable format using TCP dump file converter

This task enables you to convert TCP dump packet trace files into readable formats (PCAP and text) for analysis and review.

You may need to analyze network packet traces captured in binary format, which are not readable as-is. Converting them to PCAP or text allows you to use analysis tools or view them manually in the CLI.

Follow these steps to convert TCP dump packet traces to PCAP and text formats:

Procedure

Step 1 Use the **tcp dump-file convert all-formats all** command to convert the dump packet traces in binary files into PCAP and text formats.

Example:

```
RP/0/RP0/CPU0:ios# tcp dump-file convert all-formats all
ascii file is saved at :
/harddisk:/decoded_dumpfiles/text_tcpdump_peer_all_node0_RP0_CPU0_2024_3_19_10_8_53.462070.txt
pcap file is saved at :
/harddisk:/decoded_dumpfiles/pcap_tcpdump_peer_all_node0_RP0_CPU0_2024_3_19_10_8_40.154838.pcap
[OK]
```

By default, the system stores the converted files in the decoded_dumpfiles folder on the hard disk.

Using the **location node-id** and **file <file path>** keywords, you can save the converted TCP dump file to your desired location.

For example, **tcp dump-file convert all-formats all location** *0/RP0/CPU0* **file** */harddisk:/demo2*.

```
RP/0/RP0/CPU0:ios# tcp dump-file convert all-formats all location 0/RP0/CPU0 file /harddisk:/demo2
ascii file is saved at : /harddisk:/demo2.txt
pcap file is saved at : /harddisk:/demo2.pcap
[OK]
```

Step 2 Use the **run cat <text file path>** command to view the converted text file in the CLI.

Example:

```
RP/0/RP0/CPU0:ios# run cat
/harddisk:/decoded_dumpfiles/text_tcpdump_peer_all_node0_RP0_CPU0_2024_3_19_10_8_53.462070.txt
Filename: 2024_3_19_10_8_53.462070

============================================================
Connection state is CLOSED, I/O status: 0, socket status: 103
PCB 0x0000000000f47a80, SO 0xf476d0, TCPCB 0xf6a370, vrfid 0x60000000,
Pak Prio: Medium, TOS: 192, TTL: 255, Hash index: 563
Local host: 14:11:11::1, Local port: 47743 (Local App PID: 19579)
Foreign host: 14:11:11::2, Foreign port: 179
(Local App PID/instance/SPL_APP_ID: 19579/1/0)

Current send queue size in bytes: 0 (max 0)
Current receive queue size in bytes: 0 (max 0)  mis-ordered: 0 bytes
Current receive queue size in packets: 0 (max 0)

Timer          Starts    Wakeups         Next(msec)
Retrans            70          2                  0
SendWnd             0          0                  0
TimeWait            2          0                  0
AckHold            66         61                  0
```

```
KeepAlive           1          0               0
PmtuAger            0          0               0
GiveUp              0          0               0
Throttle            0          0               0
FirstSyn            1          1               0

   iss: 3113104891  snduna: 3113106213  sndnxt: 3113106213
sndmax: 3113106213  sndwnd: 31523       sndcwnd: 2832
   irs: 4250126727  rcvnxt: 4250128049  rcvwnd: 31448   rcvadv: 4250159497
```

This sample displays only a portion of the actual output. The full output provides additional details.

Step 3 Use the **scp** command to copy the converted packet traces from the system to your local computer and view the converted PCAP file.

The system converts TCP dump packet traces into readable PCAP and text formats. The files are stored at your specified location and are accessible for further analysis from the CLI or on your local machine.