Displaying and Capturing Live Traffic on an Interface

This chapter describes how to display, capture, copy, and erase packet files. It contains the following sections:

Understanding Packet Display and Capture

You can display or capture live traffic from an interface and have the live traffic or a previously captured file put directly on the screen. Storage is available for one local file only, subsequent capture requests overwrites an existing file. The size of the storage file varies depending on the platform. A message may be displayed if the maximum file size is reached before the requested packet count is captured.

Note Capturing live traffic off the interface does not disrupt any of the functionality of the sensor.

Caution Changing the interface configuration results in abnormal termination of any **packet** command running on that interface.

Caution Executing the **packet display** or **capture** command causes significant performance degradation.

Note On IPS sensors with multiple processors (for example, the IPS 4260 and IPS 4270-20), packets may be captured out of order in the IP logs and by the **packet** command. Because the packets are not processed using a single processor, the packets can become out of sync when received from multiple processors.

Displaying Live Traffic on an Interface

Use the **packet display** *interface_name* [**snaplen** *length*] [**count** *count*] [**verbose**] [**expression** *expression*] command to display live traffic from an interface directly on your screen. Use the **packet display iplog** *id* [**verbose**] [**expression** *expression*] to display iplogs.

Note   To terminate the live display, press **Ctrl-C**.

The following options apply:

- *interface_name*—Interface name, interface type (GigabitEthernet, FastEthernet, Management) followed by slot/port. You can only use an interface name that exists in the system.

- **snaplen**—(Optional) Maximum number of bytes captured for each packet. The valid range is 68 to 1600. The default is 0. A value of 0 means use the required length to catch whole packets.

- **count**—(Optional) Maximum number of packets to capture. The valid range is 1 to 10000.

   Note   If you do not specify this option, the capture terminates after the maximum file size is captured.

- **verbose**—(Optional) Displays the protocol tree for each packet rather than a one-line summary.

- **expression**—Packet-display filter expression. This expression is passed directly to TCPDUMP and must meet the TCPDUMP expression syntax.

   Note   The expression syntax is described in the TCPDUMP man page.

   Note   When using the **expression** option when monitoring packets with VLAN headers, the expression does not match properly unless **vlan and** is added to the beginning of the expression. For example, **packet display iplog 926299444 verbose expression icmp** Will NOT show ICMP packets; **packet display iplog 926299444 verbose expression vlan and icmp** WILL show ICMP packets. It is often necessary to use **expression vlan and** on the AIP SSC-5, IDSM2, and IPS appliance interfaces connected to trunk ports.

- **file-info**—Displays information about the stored packet file.

   File-info displays the following information:

   Captured by: *user:id*, Cmd: *cliCmd*

   Start: yyyy/mm/dd hh:mm:ss zone, End: yyyy/mm/dd hh:mm:ss zone or in-progress

   Where

   user = username of user initiating capture

   id = CLI ID of the user

   cliCmd = command entered to perform the capture

Caution   Executing the **packet display** command causes significant performance degradation.

To configure the sensor to display live traffic from an interface on the screen, follow these steps:

Step 1   Log in to the sensor using an account with administrator or operator privileges.

Step 2   Display the live traffic on the interface you are interested in, for example, GigabitEthernet0/1.

```
sensor# packet display GigabitEthernet0/1
Warning: This command will cause significant performance degradation
tcpdump: listening on ge0_1, link-type EN10MB (Ethernet), capture size 65535 bytes
03:43:05.691883 IP (tos 0x10, ttl  64, id 55460, offset 0, flags [DF], length: 100)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 4233955485:4233955533(48) ack
1495691730 win 8576 <nop,nop,timestamp 44085169 226014949>
03:43:05.691975 IP (tos 0x10, ttl  64, id 55461, offset 0, flags [DF], length: 164)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 48:160(112) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.691998 IP (tos 0x10, ttl  64, id 53735, offset 0, flags [DF], length: 52)
10.89.147.50.41805 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 48 win 11704
<nop,nop,timestamp 226014949 44085169>
03:43:05.693165 IP (tos 0x10, ttl  64, id 53736, offset 0, flags [DF], length: 52)
10.89.147.50.41805 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 160 win 11704
<nop,nop,timestamp 226014949 44085169>
03:43:05.693351 IP (tos 0x10, ttl  64, id 55462, offset 0, flags [DF], length: 316)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 160:424(264) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.693493 IP (tos 0x10, ttl  64, id 55463, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 424:664(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.693612 IP (tos 0x10, ttl  64, id 55464, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 664:904(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.693628 IP (tos 0x10, ttl  64, id 53737, offset 0, flags [DF], length: 52)
10.89.147.50.41805 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 424 win 11704
<nop,nop,timestamp 226014949 44085169>
03:43:05.693654 IP (tos 0x10, ttl  64, id 53738, offset 0, flags [DF], length: 52)
10.89.147.50.41805 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 664 win 11704
<nop,nop,timestamp 226014949 44085169>
03:43:05.693926 IP (tos 0x10, ttl  64, id 55465, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 904:1144(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.694043 IP (tos 0x10, ttl  64, id 55466, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 1144:1384(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.694163 IP (tos 0x10, ttl  64, id 55467, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 1384:1624(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014949>
03:43:05.694209 IP (tos 0x10, ttl  64, id 53739, offset 0, flags [DF], length: 52)
10.89.147.50.41805 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 1384 win 11704
<nop,nop,timestamp 226014950 44085169>
03:43:05.694283 IP (tos 0x10, ttl  64, id 55468, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 1624:1864(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014950>
03:43:05.694402 IP (tos 0x10, ttl  64, id 55469, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 1864:2104(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014950>
03:43:05.694521 IP (tos 0x10, ttl  64, id 55470, offset 0, flags [DF], length: 292)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 2104:2344(240) ack 1 win 8576
<nop,nop,timestamp 44085169 226014950>
03:43:05.694690 IP (tos 0x10, ttl  64, id 53740, offset 0, flags [DF], length: 52)
10.89.147.50.41805 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 2344 win 11704
<nop,nop,timestamp 226014950 44085169>
03:43:05.694808 IP (tos 0x10, ttl  64, id 55471, offset 0, flags [DF], length: 300)
10.89.147.31.22 > 10.89.147.50.41805: P [tcp sum ok] 2344:2592(248) ack 1 win 8576
<nop,nop,timestamp 44085169 226014950>
```

Step 3 Use the **expression** option to limit what you display, for example, only TCP packets.

> ✎
>
> **Note** As described in the TCPDUMP man page, the protocol identifiers tcp, udp, and icmp are also keywords and must be escaped by using two back slashes (\\).

```
sensor# packet display GigabitEthernet0/1 verbose expression ip proto \\tcp
Warning: This command will cause significant performance degradation
tcpdump: listening on ge0_1, link-type EN10MB (Ethernet), capture size 65535 bytes
03:42:02.509738 IP (tos 0x10, ttl  64, id 27743, offset 0, flags [DF], length: 88)
10.89.147.31.22 > 64.101.182.54.47039: P [tcp sum ok] 3449098782:3449098830(48) ack
3009767154 win 8704
03:42:02.509834 IP (tos 0x10, ttl  64, id 27744, offset 0, flags [DF], length: 152)
10.89.147.31.22 > 64.101.182.54.47039: P [tcp sum ok] 48:160(112) ack 1 win 8704
03:42:02.510248 IP (tos 0x0, ttl 252, id 55922, offset 0, flags [none], length: 40)
64.101.182.54.47039 > 10.89.147.31.22: . [tcp sum ok] 1:1(0) ack 160 win 8760
03:42:02.511262 IP (tos 0x10, ttl  64, id 27745, offset 0, flags [DF], length: 264)
10.89.147.31.22 > 64.101.182.54.47039: P [tcp sum ok] 160:384(224) ack 1 win 8704
03:42:02.511408 IP (tos 0x10, ttl  64, id 27746, offset 0, flags [DF], length: 248)
10.89.147.31.22 > 64.101.182.54.47039: P [tcp sum ok] 384:592(208) ack 1 win 8704
03:42:02.511545 IP (tos 0x10, ttl  64, id 27747, offset 0, flags [DF], length: 240)
10.89.147.31.22 > 64.101.182.54.47039: P [tcp sum ok] 592:792(200) ack 1 win 8704
```

Step 4 Display information about the packet file.

```
sensor# packet display file-info
Captured by: cisco:25579, Cmd: packet capture GigabitEthernet0/1
Start: 2003/02/03 02:56:48 UTC, End: 2003/02/03 02:56:51 UTC
sensor#
```

Capturing Live Traffic on an Interface

Use the **packet capture** *interface_name* [**snaplen** *length*] [**count** *count*] [**expression** *expression*] command to capture live traffic on an interface.

Only one user can use the **packet capture** command at a time. A second user request results in an error message containing information about the user currently executing the capture.

> ⚠
>
> **Caution** Executing the **packet capture** command causes significant performance degradation.

The **packet capture** command captures the libpcap output into a local file.

Use the **packet display packet-file** [**verbose**] [**expression** *expression*] command to view the local file.

Use the **packet display file-info** to display information about the local file, if any.

The following options apply:

- *interface_name*—Logical interface name. You can only use an interface name that exists in the system.

- **snaplen**—Maximum number of bytes captured for each packet (optional). The valid range is 68 to 1600. The default is 0.

- **count**—Maximum number of packets to capture (optional). The valid range is 1 to 10000.

    ✎
    Note   If you do not specify this option, the capture terminates after the maximum file size is captured.

- **expression**—Packet-capture filter expression. This expression is passed directly to TCPDUMP and must meet the TCPDUMP expression syntax.
- **file-info**—Displays information about the stored packet file.

    File-info displays the following information:

    Captured by: *user:id*, Cmd: *cliCmd*

    Start: yyyy/mm/dd hh:mm:ss zone, End: yyyy/mm/dd hh:mm:ss zone or in-progress

    Where

    user = username of user initiating capture

    id = CLI ID of the user

    cliCmd = command entered to perform the capture

- **verbose**—Displays the protocol tree for each packet rather than a one-line summary. This parameter is optional.

To configure the sensor to capture live traffic on an interface, follow these steps:

Step 1   Log in to the sensor using an account with administrator or operator privileges.

Step 2   Capture the live traffic on the interface you are interested in, for example, GigabitEthernet0/1.

```
sensor# packet capture GigabitEthernet0/1
Warning: This command will cause significant performance degradation
tcpdump: WARNING: ge0_1: no IPv4 address assigned
tcpdump: listening on ge0_1, link-type EN10MB (Ethernet), capture size 65535 bytes
125 packets captured
126 packets received by filter
0 packets dropped by kernel
```

Step 3   View the captured packet file.

```
sensor# packet display packet-file
reading from file /usr/cids/idsRoot/var/packet-file, link-type EN10MB (Ethernet)
03:03:13.216768 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:13.232881 IP 64.101.182.244.1978 > 10.89.130.108.23: . ack 3266153791 win
64328
03:03:13.232895 IP 10.89.130.108.23 > 64.101.182.244.1978: P 1:157(156) ack 0 wi
n 5840
03:03:13.433136 IP 64.101.182.244.1978 > 10.89.130.108.23: . ack 157 win 65535
03:03:13.518335 IP 10.89.130.134.42342 > 255.255.255.255.42342: UDP, length: 76
03:03:15.218814 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:15.546866 IP 64.101.182.244.1978 > 10.89.130.108.23: P 0:2(2) ack 157 win
65535
03:03:15.546923 IP 10.89.130.108.23 > 64.101.182.244.1978: P 157:159(2) ack 2 wi
n 5840
03:03:15.736377 IP 64.101.182.244.1978 > 10.89.130.108.23: . ack 159 win 65533
03:03:17.219612 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:19.218535 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
```

```
03:03:19.843658 IP 64.101.182.143.3262 > 10.89.130.23.445: P 3749577803:37495778
56(53) ack 3040953472 win 64407
03:03:20.174835 IP 161.44.55.250.1720 > 10.89.130.60.445: S 3147454533:314745453
3(0) win 65520 <mss 1260,nop,nop,sackOK>
03:03:21.219958 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:21.508907 IP 161.44.55.250.1809 > 10.89.130.61.445: S 3152179859:315217985
9(0) win 65520 <mss 1260,nop,nop,sackOK>
03:03:23.221004 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:23.688099 IP 161.44.55.250.1975 > 10.89.130.63.445: S 3160484670:316048467
0(0) win 65520 <mss 1260,nop,nop,sackOK>
03:03:25.219054 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:25.846552 IP 172.20.12.10.2984 > 10.89.130.127.445: S 1345848756:134584875
6(0) win 64240 <mss 1460,nop,nop,sackOK>
03:03:26.195342 IP 161.44.55.250.2178 > 10.89.130.65.445: S 3170518052:317051805
2(0) win 65520 <mss 1260,nop,nop,sackOK>
03:03:27.222725 802.1d config TOP_CHANGE 8000.00:04:9a:66:35:01.8025 root 8000.0
0:04:6d:f9:e8:82 pathcost 8 age 2 max 20 hello 2 fdelay 15
03:03:27.299178 IP 161.44.55.250.2269 > 10.89.130.66.445: S 3174717959:317471795
9(0) win 65520 <mss 1260,nop,nop,sackOK>
03:03:27.308798 arp who-has 161.44.55.250 tell 10.89.130.66
03:03:28.383028 IP 161.44.55.250.2349 > 10.89.130.67.445: S 3178636061:317863606
1(0) win 65520 <mss 1260,nop,nop,sackOK>
--MORE--
```

Step 4   View any information about the packet file.

```
sensor# packet display file-info
Captured by: cisco:8874, Cmd: packet capture GigabitEthernet0/1
Start: 2003/01/07 00:12:50 UTC, End: 2003/01/07 00:15:30 UTC
sensor#
```

Copying the Packet File

Use the **copy packet-file** *destination_url* command to copy the packet file to an FTP or SCP server for saving or further analysis with another tool, such as Wireshark or TCPDUMP.

The following options apply:

- **packet-file**—Locally stored libpcap file that you captured using the **packet capture** command.

- *destination_url*—The location of the destination file to be copied. It can be a URL or a keyword.

> ✎
>
> **Note**   The exact format of the source and destination URLs varies according to the file.

   – ftp:—Destination URL for an FTP network server. The syntax for this prefix is:

   ftp:[//[username@] location]/relativeDirectory]/filename

   ftp:[//[username@]location]//absoluteDirectory]/filename

- scp:—Destination URL for the SCP network server. The syntax for this prefix is:

  scp:[//[username@] location]/relativeDirectory]/filename

  scp:[//[username@] location]//absoluteDirectory]/filename

✎

Note   When you use FTP or SCP protocol, you are prompted for a password.

To copy packets files to an FTP or SCP server, follow these steps:

Step 1   Log in to the CLI using an account with administrator privileges.

Step 2   Copy the packet-file to an FTP or SCP server.

```
sensor# copy packet-file scp://jbrown@64.101.182.20/work/
Password: *********
packet-file                 100% 1670     0.0KB/s    00:00
sensor#
```

Step 3   View the packet file with Wireshark or TCPDUMP.

Erasing the Packet File

Use the **erase packet-file** command to erase the packet file. There is only one packet file. It is 16 MB and is over-written each time you use the **packet capture** command.

To erase the packet file, follow these steps:

Step 1   Display information about the current captured packet file.

```
sensor# packet display file-info
Captured by: cisco:1514, Cmd: packet capture GigabitEthernet0/1
Start: 2005/02/15 03:55:00 CST, End: 2005/02/15 03:55:05 CST
sensor#
```

Step 2   Erase the packet file.

```
sensor# erase packet-file
sensor#
```

Step 3   Verify that you have erased the packet file.

```
sensor# packet display file-info
No packet-file available.
sensor#
```