Packet Trace

Table 1: Feature History

Feature Name	Release Information	Description
Bidirectional Support for Packet Tracing	Cisco IOS XE Release 17.8.1a Cisco SD-WAN Release 20.8.1 Cisco vManage Release 20.8.1	This feature provides a detailed understanding of how data packets are processed by the edge devices in both the directions. Bidirectional debugging can help you to diagnose issues and troubleshoot them more efficiently.

Information About Packet Trace

The Packet Trace feature enables you to debug packet loss on edge devices and to inspect any forwarding behavior of traffic flows on the devices in the network. You can configure packet tracer with various conditions based on which the flow of the packets is segregated and is captured for tracing. This helps you to diagnose issues and troubleshoot them more efficiently.

Packet tracer includes 2048 bytes of internal memory that is used to copy path data. This memory is overwritten during circular mode of tracing.

The Packet Trace feature provides three levels of inspection for packets—accounting, summary, and path data. Each level provides a detailed view of packet processing at the cost of some packet-processing capability. However, packet trace limits the inspection of packets that match the **debug platform condition** statements, and is a viable option even under heavy-traffic situations in customer environments.

From Cisco IOS XE Release 17.8.1a, bidirectional support is added on the edge devices for a conditional debugging match filter. Conditional debugging allows you to filter out some of the debugging information on the edge device. You can check the debugging information that matches a certain interface, MAC address, or username.

Table 2: Packet Trace Levels

Packet Trace Level	Description
Accounting	Packet trace accounting provides a count of packets that enter and leave the network processor. Packet trace accounting is a lightweight performance activity, and runs continuously until it is disabled.
Summary	At the summary level of packet trace, data is collected for a finite number of packets. Packet trace summary tracks the input and output interfaces, the final packet state, the consumed packet state and punt, drop, or inject packets, if any. Collecting summary data adds to additional performance compared to normal packet processing, and can help to isolate a troublesome interface.
Path data	Packet trace path data level provides the greatest level of detail in packet trace. Data is collected for a finite number of packets. Packet trace path data captures data, including a conditional debugging ID that is useful to correlate with feature debugs, a timestamp, and also feature-specific path-trace data. Path data also has two optional capabilities—packet copy and Feature Invocation Array (FIA) trace. The packet copy option enables you to copy input and output packets at various layers of the packet (layer 2, layer 3, or layer 4). The FIA trace option tracks every feature entry invoked during packet processing and helps you to know what is happening during packet processing. **Note** Collecting path data consumes more packet-processing resources, and the optional capabilities incrementally affect packet performance. We recommend that you use path-data level in a limited way or in situations where packet performance change is acceptable.

Usage Guidelines for Configuring Packet Trace

Consider the following best practices while configuring the Packet Trace:

- Use of ingress conditions when using the packet trace is recommended for a more comprehensive view of packets.
- Packet trace configuration requires data plane memory. On systems where data plane memory is constrained, carefully consider how you will select the packet trace values. A close approximation of the amount of memory consumed by packet trace is provided by the following equation:

memory required = (statistics overhead) + (number of packets) * (summary size + data size + packet copy size).

When the Packet Trace feature is enabled, a small, fixed amount of memory is allocated for statistics. Similarly, when per-packet data is captured, a small, fixed amount of memory is required for each packet for summary data. However, as shown by the equation, you can significantly influence the amount of memory consumed by the number of packets you select to trace, and whether you collect path data and copies of packets.

Note The amount of memory consumed by the packet trace feature is affected by the packet trace configuration. You should carefully select the size of per-packet path data and copy buffers and the number of packets to be traced in order to avoid interrupting other router services.

Limitations

- Only IP packets are supported. L2 (ARP) packets, bridge packets, fragmented packets, and multicast packets are not supported.
- IPv6 is not supported.
- Packet duplication is not supported.
- Any packet that goes through resubmission (for example, IPsec or GRE encrypted packets) and matches the configured filters in both the inner packet (decrypted packet) as well as the outer packet (encrypted packet) will have individual trace entries. To use the packet tracer more efficiently, you should configure as many filters as possible with the available information to debug the issue.

Configure Packet Trace

Use the **debug platform packet-trace** command to configure a packet tracer on edge devices with various conditions such as bidirectional, VPN, circular, destination IP, source IP, interface, start, stop, logging, and clear.

Configure Packet Trace on Cisco IOS XE SD-WAN devices

1. Enable packet trace for the traffic and specify the maximum number of packets:

   ```
   Device# debug platform packet-trace packet [number of traced packets]
   ```

2. Specify the matching criteria for tracing packets. Matching criteria provides the ability to filter by protocol, IP address and subnet mask, interface, and direction:

   ```
   Device# debug platform condition [interface interface name] {match ipv4|ipv6|mac src dst}  {both|ingress|egress} [bidirectional]
   ```

3. Enable the specified matching criteria and start packet tracing:

   ```
   Device# debug platform condition start
   ```

4. Deactivate the condition and stop packet tracing:

   ```
   Device# debug platform condition stop
   ```

5. Exit the privileged EXEC mode:

```
exit
```

Configure Packet Trace on Cisco vEdge devices

The following example shows how to configure conditions for packet tracing:

```
Device# debug packet-trace condition source-ip 10.1.1.1
Device# debug packet-trace condition vpn-id 0
Device# debug packet-trace condition interface ge0/1
Device# debug packet-trace condition stop
```

For more information, see debug packet-trace condition command page.

Monitor Packet Trace

Packet trace configuration is based on the AND operation of the specified conditions, with the packets matching all the configured conditions being traced.

Monitoring Packet Trace on Cisco vEdge devices

Use the **show packet-trace statistics** command on Cisco vEdge devices to view the summary of all the packets matching the specified condition.

The following example displays all the conditions that are configured for packet tracing:

```
Device# show debugs
debugs packet-trace condition source-ip 10.1.1.1
debugs packet-trace condition vpn-id 0
debugs packet-trace condition interface ge0/1
debugs packet-trace condition state Stopped
```

Use the **show packet-trace statistics** command on Cisco vEdge devices to view the summary of all the packets matching the specified condition.

The following example displays a packet trace statistics for the specified interface, in this case, ge 0:

```
Device# show packet-trace statistics source-interface ge0_0
packet-trace statistics 0
source-ip 10.1.15.13
source-port 0
destination-ip 224.0.0.5
destination-port 0
source-interface ge0_0
destination-interface loop0.0
decision PUNT
duration 40
```

For more information, see show packet-tracer command page.

Detailed Packet View:

The following is a sample output of the **show packet-trace details** command, which is displayed for the specified trace ID 10:

```
Device# show packet-trace details 10
```

```
=============================================================================================
Pkt-id     src_ip(ingress_if)     dest_ip(egress_if)     Duration     Decision
=============================================================================================
10         10.1.15.15:0 (ge0_0)   192.168.255.5:0 (ge0_0)     15 us          PUNT
INGRESS_PKT:
01 00 5e 00 00 05 52 54 00 6b 4b fa 08 00 45 c0 00 44 f8 60 00 00 01 59 c7 2b 0a 01 0f 0f
```

```
e0
00 00 05 02 01 00 30 ac 10 ff 0f 00 00 00 33 8d 1b 00 00 00 00 00 00 00 00 00 ff ff ff
00 00 0a 02 00 00 00 00 28 0a 01 0f 0d 00 00 00 00 ac 10 ff 0d 00 00 00 00 00 00 00 00
00 00 00 00 00
EGRESS_PKT:
01 00 5e 00 00 05 52 54 00 6b 4b fa 08 00 45 c0 00 44 f8 60 00 00 01 59 c7 2b 0a 01 0f 0f
e0
00 00 05 02 01 00 30 ac 10 ff 0f 00 00 00 33 8d 1b 00 00 00 00 00 00 00 00 00 ff ff ff
00 00 0a 02 00 00 00 00 28 0a 01 0f 0d 00 00 00 00 ac 10 ff 0d 00 00 00 00 00 00 00 00
00 00 00 00 00
Feature Data
-----------------------------------
TOUCH : fp_proc_packet
-----------------------------------
TOUCH : fp_proc_packet2
-----------------------------------
TOUCH : fp_send_to_host
-----------------------------------
FP_TRACE_FEAT_PUNT_INFO:
icmp_type : 0
icmp_code : 0
qos : 7
-----------------------------------
TOUCH : fp_hw_x86_pkt_free
```

Use the **show packet-trace details** command to view detailed information for the specified trace ID. The detailed packet view output displays three sections - summary data section, packet dump section, and featured data section.

Monitoring Packet Trace on Cisco IOS XE SD-WAN Devices

Summary View:

Use the **show platform packet-trace summary** command on Cisco IOS XE SD-WAN devices to view the summary of all the packets matching the specified condition.

The following example displays a packet trace summary on Cisco IOS XE SD-WAN devices:

```
Device# show platform packet-trace summary

Pkt   Input                     Output                    State  Reason
0     INJ.12                    Gi2                       FWD
1     Gi2                       internal0/0/rp:0          PUNT   5
2     INJ.1                     Gi2                       FWD
3     INJ.1                     Gi2                       FWD
4     Gi2                       internal0/0/rp:0          PUNT   5
5     Gi2                       internal0/0/rp:0          PUNT   5
6     INJ.1                     Gi2                       FWD
7     INJ.1                     Gi2                       FWD
8     Gi2                       internal0/0/rp:0          PUNT   5
9     Gi2                       internal0/0/rp:0          PUNT   5
10    Gi2                       internal0/0/rp:0          PUNT   5
11    INJ.1                     Gi2                       FWD
12    Gi2                       internal0/0/rp:0          PUNT   5
13    INJ.1                     Gi2                       FWD
14    INJ.1                     Gi2                       FWD
```

Detailed Packet View:

The following is a sample output of the **show packet trace details** command on Cisco IOS XE SD-WAN devices, which is displayed for the specified trace ID 10:

```
Device# show platform packet-trace packet 10

Packet: 10          CBUG ID: 116
Summary
  Input     : GigabitEthernet2
  Output    : internal0/0/rp:0
  State     : PUNT 5   (CLNS IS-IS Control)
  Timestamp
    Start   : 2427641145361169 ns (02/23/2022 00:14:58.869057 UTC)
    Stop    : 2427641145374580 ns (02/23/2022 00:14:58.869071 UTC)
Path Trace
  Feature: DEBUG_COND_INPUT_PKT_EXT
    Entry       : Input - 0x813e9f60
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 176 ns
  Feature: LAYER2_INPUT_LOOKUP_PROCESS_EXT
    Entry       : Input - 0x81419e2c
    Input       : GigabitEthernet2
    Output      : internal0/0/rp:0
    Lapsed time : 896 ns
  Feature: LAYER2_INPUT_GOTO_OUTPUT_FEATURE_EXT
    Entry       : Input - 0x813ed9e8
    Input       : GigabitEthernet2
    Output      : internal0/0/rp:0
    Lapsed time : 553 ns
  Feature: LAYER2_OUTPUT_QOS_EXT
    Entry       : Output - 0x81420930
    Input       : GigabitEthernet2
    Output      : internal0/0/rp:0
    Lapsed time : 748 ns
  Feature: LAYER2_OUTPUT_DROP_POLICY_EXT
    Entry       : Output - 0x8142092c
    Input       : GigabitEthernet2
    Output      : internal0/0/rp:0
    Lapsed time : 947 ns
  Feature: INTERNAL_TRANSMIT_PKT_EXT
    Entry       : Output - 0x813eaa6c
    Input       : GigabitEthernet2
    Output      : internal0/0/rp:0
    Lapsed time : 6575 ns
Packet Copy In
  0180c200 00140050 569a8a44 0062fefe 03831b01 00120100 00005f04 af200020
  00200000 00000000 0a0aee03 01040349 00018101 cc890b61 6c706861 2d637372
  2d31020c 000a0000 00200020 00200001 84046400 00018018 0a808080 0a000a00
  ffffff00 00808080 64000001 ffffffff
Packet Copy Out
  01010000 00110070 00b80028 200a0000 00000000 00000006 00000000 80010500
  02065900 00000001 01010000 000e003c 00000000 00000074 03f50000 00000005
  00000000 80010700 0180c200 00140050 569a8a44 0062fefe 03831b01 00120100
  00005f04 af200020 00200000 00000000 0a0aee03 01040349 00018101 cc890b61
  6c706861 2d637372 2d31020c 000a0000 00200020 00200001 84046400 00018018
  0a808080 0a000a00 ffffff00 00808080 64000001 ffffffff

IOSd Path Flow: Packet: 10    CBUG ID: 116
  Feature: INFRA
  Pkt Direction: IN
    Packet Rcvd From DATAPLANE
```

Use the **show platform packet-trace summary** command to view detailed information for the specified trace ID. The detailed packet view output displays three sections—summary data section, packet dump section, and featured data section.

- Summary data section: Displays packet trace ID, ingress interface, egress interface, and the forward decision taken for the packet to traverse across the device information for the specified trace ID.
- Packet dump section: Displays ingress and egress packet information. Only the first 96 bytes of packet header details are displayed.

Note The complete packet dump is not displayed because of tracer-memory limitations.

- Feature data section: Displays forwarding plane features that generate feature-specific tracing data and provides feature data decodes. These features provide debugging information to packet tracer, such as forward result, drop reason, and other behavior.

Configuration Examples for Packet Trace

The following example shows how to configure and monitor the conditions for packet tracing:

```
Device# debug platform packet-trace packet 2048
Device# debug platform condition ingress
Device# debug platform condition start
Device# debug platform condition stop
Device# show platform packet-trace summary
Pkt Input Output State Reason
0 Gi0/0/2.3060 Gi0/0/2.3060 DROP 402
1 internal0/0/rp:0 internal0/0/rp:0 PUNT 21 2 internal0/0/recycle:0 Gi0/0/2.3060 FWD
```