Packet Trace

Table 1: Feature History

Feature Name	Release Information	Description
Bidirectional Support for Packet Tracing	Cisco IOS XE Catalyst SD-WAN 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.
Packet Trace Improvements	Cisco IOS XE Catalyst SD-WAN Release 17.11.1a Cisco vManage Release 20.11.1	This feature offers the following enhancements to packet trace: • A new command **show platform packet-trace fia-statistics**, available on Cisco IOS XE Catalyst SD-WAN devices, displays Feature Invocation Array (FIA) statistics in a packet trace. In FIA statistics, you can find data about a packet trace's feature count, the average processing time, the minimum processing time, and the maximum processing time. • View label information for the Multiprotocol Label Switching (MPLS) feature in a packet trace.

- Information About Packet Trace, on page 2
- Configure Packet Trace, on page 3
- Monitor Packet Trace, on page 4
- Configuration Examples for Packet Trace, on page 9

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 Catalyst SD-WAN 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 Catalyst 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 MPLS output label trace. A MPLS output label trace is included in debug path to reduce the impact on performance.

```
Device# debug platform hardware qfp active feature cef-mpls datapath mpls all
```

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

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

5. Deactivate the condition and stop packet tracing:

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

6. 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.

Monitor 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 10.4.0.5
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)   12.168.255.5:0 (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 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 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 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 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.

Monitor Packet Trace on Cisco IOS XE Catalyst SD-WAN Devices

Summary View

Use the **show platform packet-trace summary** command on Cisco IOS XE Catalyst 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 Catalyst 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 platform packet-trace packet 0** command on Cisco IOS XE Catalyst SD-WAN devices:

```
Device# show platform packet-trace packet 0

Packet: 0           CBUG ID: 4321
Summary
  Input     : GigabitEthernet2
  Output    : GigabitEthernet3
  State     : FWD
  Timestamp
    Start   : 1124044721695603 ns (09/20/2022 01:47:28.531049 UTC)
    Stop    : 1124044722142898 ns (09/20/2022 01:47:28.531497 UTC)
Path Trace
  Feature: IPV4(Input)
    Input       : GigabitEthernet2
    Output      : <unknown>
    Source      : 10.10.10.10
    Destination : 20.20.20.20
    Protocol    : 1 (ICMP)
  Feature: DEBUG_COND_INPUT_PKT
    Entry       : Input - 0x814670b0
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 600 ns
  Feature: IPV4_INPUT_DST_LOOKUP_ISSUE
    Entry       : Input - 0x81494d2c
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 1709 ns
  Feature: IPV4_INPUT_ARL_SANITY
    Entry       : Input - 0x814690e0
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 1274 ns
  Feature: IPV4_INPUT_DST_LOOKUP_CONSUME
    Entry       : Input - 0x81494d28
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 269 ns
  Feature: IPV4_INPUT_FOR_US_MARTIAN
    Entry       : Input - 0x81494d34
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 384 ns
  Feature: DEBUG_COND_APPLICATION_IN
```

```
  Entry       : Input - 0x814670a0
  Input       : GigabitEthernet2
  Output      : <unknown>
  Lapsed time : 107 ns
Feature: DEBUG_COND_APPLICATION_IN_CLR_TXT
  Entry       : Input - 0x8146709c
  Input       : GigabitEthernet2
  Output      : <unknown>
  Lapsed time : 36 ns
Feature: IPV4_INPUT_LOOKUP_PROCESS
  Entry       : Input - 0x81494d40
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 38331 ns
Feature: IPV4_INPUT_IPOPTIONS_PROCESS
  Entry       : Input - 0x81495258
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 259 ns
Feature: IPV4_INPUT_GOTO_OUTPUT_FEATURE
  Entry       : Input - 0x8146ab58
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 9485 ns
Feature: IPV4_VFR_REFRAG
  Entry       : Output - 0x81495c6c
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 520 ns
Feature: IPV6_VFR_REFRAG
  Entry       : Output - 0x81496600
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 296 ns
Feature: MPLS(Output)
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Label Stack Entry[1]: 0x03e850fe
    StackEnd:NO, TTL:254, EXP:0, Label:16005, is SDWAN:NO
  Label Stack Entry[2]: 0x000121fe
    StackEnd:YES, TTL:254, EXP:0, Label:18, is SDWAN:NO
Feature: MPLS_OUTPUT_ADD_LABEL
  Entry       : Output - 0x8145e130
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 29790 ns
Feature: MPLS_OUTPUT_L2_REWRITE
  Entry       : Output - 0x812f4724
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 23041 ns
Feature: MPLS_OUTPUT_FRAG
  Entry       : Output - 0x8149ae5c
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 785 ns
Feature: MPLS_OUTPUT_DROP_POLICY
  Entry       : Output - 0x8149ebdc
  Input       : GigabitEthernet2
  Output      : GigabitEthernet3
  Lapsed time : 14697 ns
Feature: MARMOT_SPA_D_TRANSMIT_PKT
  Entry       : Output - 0x814ac56c
  Input       : GigabitEthernet2
```

```
    Output      : GigabitEthernet3
    Lapsed time : 45662 ns
Packet Copy In
  00505683 d54f0050 56830863 08004500 00641018 0000ff01 6f450a0a 0a0a1414
  14140800 3839001c 00000000 00005b3a eabaabcd abcdabcd abcdabcd abcdabcd
Packet Copy Out
  00505683 d4900050 5683429a 884703e8 50fe0001 21fe4500 00641018 0000fe01
  70450a0a 0a0a1414 14140800 3839001c 00000000 00005b3a eabaabcd abcdabcd
```

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.

View FIA Statistics

Minimum supported releases: Cisco vManage Release 20.11.1 and Cisco IOS XE Catalyst SD-WAN Release 17.11.1a

Use the **show platform packet-trace fia-statistics** command on Cisco IOS XE Catalyst SD-WAN devices to view to FIA statistics. FIA statistics provides details about the number of features, and the time details–minimum time, maximum time, and average time about a feature.

The following example displays FIA statistics on Cisco IOS XE Catalyst SD-WAN devices:

```
Device# show platform packet-trace fia-statistics

Feature                                        Count  Min(ns)   Max(ns)   Avg(ns)
-------------------------------------------------------------------------------
INTERNAL_TRANSMIT_PKT_EXT                         66     4720     28400     13333
MARMOT_SPA_D_TRANSMIT_PKT_EXT                     16     4560     16920     11955
L2_SVI_OUTPUT_BRIDGE_EXT                           1     3640      3640      3640
INTERNAL_INPUT_GOTO_OUTPUT_FEATURE_EXT            16     1680      3880      2755
IPV4_INPUT_LOOKUP_PROCESS_EXT                      1     2720      2720      2720
IPV4_OUTPUT_L2_REWRITE_EXT                         1     2240      2240      2240
IPV4_OUTPUT_DROP_POLICY_EXT                        4     1040      2880      2050
IPV4_INTERNAL_DST_LOOKUP_CONSUME_EXT               1     1960      1960      1960
SSLVPN_INJECT_TX_MSG_EXT                          15      600      2440      1746
IPV4_INTERNAL_FOR_US_EXT                           1     1560      1560      1560
LAYER2_OUTPUT_QOS_EXT                             63      280      2480      1537
LAYER2_OUTPUT_DROP_POLICY_EXT                     78      120      3120      1525
LAYER2_INPUT_LOOKUP_PROCESS_EXT                   15      280      2240      1312
UPDATE_ICMP_PKT_EXT                                1     1280      1280      1280
DEBUG_COND_MAC_EGRESS_EXT                          3      840      1160       973
IPV4_INTERNAL_INPUT_SRC_LOOKUP_CONSUME_EXT         1      960       960       960
IPV4_PREF_TX_IF_SELECT_EXT                         1      800       800       800
```

```
DEBUG_COND_OUTPUT_PKT_EXT                        66       80      1640       707
IPV4_INTERNAL_ARL_SANITY_EXT                      3      240       960       666
IPV4_INTERNAL_INPUT_SRC_LOOKUP_ISSUE_EXT          1      640       640       640
IPV4_VFR_REFRAG_EXT                               5      320       920       640
EVC_EFP_VLAN_TAG_ATTACH_EXT                      15       80      1040       629
L2_SVI_OUTPUT_GOTO_OUTPUT_FEATURE_EXT             1      520       520       520
LAYER2_VLAN_INJECT_EXT                           15      120       760       504
L2_ES_OUTPUT_PRE_TX_EXT                          16        0      1000       502
DEBUG_COND_APPLICATION_IN_EXT                     1      480       480       480
DEBUG_COND_APPLICATION_OUT_CLR_TXT_EXT            3       80       720       426
DEBUG_COND_INPUT_PKT_EXT                         16       80       880       417
IPV4_OUTPUT_FRAG_EXT                              1      360       360       360
DEBUG_COND_APPLICATION_IN_CLR_TXT_EXT             1      320       320       320
DEBUG_COND_APPLICATION_OUT_EXT                    3      240       280       266
LFTS_INJECT_PKT_EXT                              16       40       480       250
LAYER2_BRIDGE_INJECT_EXT                         15       40       560       234
```

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
```