Unified Debug Condition to Match IPv4 and IPv6 Traffic Over MPLS

Table 1: Feature History

Feature Name	Release Information	Description
Unified Debug Condition To Match IPv4 and IPv6 Over MPLS	Cisco IOS XE Catalyst SD-WAN Release 17.11.1a	This feature introduces a debug condition to identify and resolve issues related to matching IPv4 and IPv6 traffic over an MPLS network.

- Information About the Unified Debug Condition, on page 1
- Restrictions of the Unified Debug Condition, on page 2
- Use Cases for the Unified Debug Condition, on page 2
- Debug to Match IPv4 and IPv6 Traffic Over MPLS Using the CLI, on page 2
- Verify the Unified Debug Condition to Match IPv4 and IPv6 Traffic Over MPLS, on page 4

Information About the Unified Debug Condition

The Cisco IOS XE Catalyst SD-WAN devices support the ability to add a debug condition for IPv4 and IPv6 traffic over MPLS packets. You can specify a filter condition to select the overlay IP address and optionally, the underlay MPLS label with a stack depth. Use the unified debug condition to troubleshoot MPLS networks by identifying specific packets that match certain filtering criteria and troubleshoot any issues related to the MPLS traffic to ensure that your network runs smoothly. The matching of IPv4 and IPv6 over MPLS is a three step process:

1. Debugging
2. Specifying the filtering conditions
3. Applying the filer conditions to the devices

In Cisco IOS XE Catalyst SD-WAN devices, the MPLS label is used to represent an IP VRF and is distributed by OMP protocol.

Restrictions of the Unified Debug Condition

- The debug condition for IPv4 and IPv6 over an MPLS network is supported only using the device CLI.
- Cisco SD-WAN Manager doesn't display the filtered debugging results as part of the packet trace debugging output. For more information see, Packet Trace.
- You can't enable the debug condition to match IPv4 traffic over MPLS and IPv6 traffic over MPLS at the same time.
- You can match only one MPLS label with either IPv4 or IPv6 traffic.
- The number of configurable Feature Invocation Array (FIA) entries per array is 31.

Use Cases for the Unified Debug Condition

The following are the use cases for matching IPv4 and IPv6 traffic over an MPLS network using a debug condition:

- Debugging conditions can help troubleshoot connectivity or performance issues in the network. By matching specific IPv4 and IPv6 traffic over the MPLS network, administrators can isolate the traffic that is causing the issue and analyze the behavior in more detail.
- Debugging conditions can also be useful for implementing QoS policies on the network. By matching specific IPv4 and IPv6 traffic over the MPLS network, administrators can apply different QoS policies to different types of traffic based on their characteristics, such as bandwidth requirements, latency sensitivity, or priority. For more information see, Cisco SD-WAN Forwarding and QoS Configuration Guide.

Debug to Match IPv4 and IPv6 Traffic Over MPLS Using the CLI

Use the **debug platform condition mpls match-inner** command to match IPv4 and IPv6 traffic over MPLS using various filtering conditions such as *match-inner ipv4*, *match-inner ipv6*, and *allow-no-label*. Specify the MPLS label information, inner IPv4 and IPv6 address based on the debugging requirement.

1. Debug the MPLS network.

   ```
   debug platform condition mpls
   ```

2. Specify the debugging conditions as per your requirement.
   - Use the following condition to debug IPv4 traffic over an MPLS network without specifying the MPLS label:

     ```
     match-inner ipv4
     ```

   - Use the following condition to debug IPv6 traffic over an MPLS network without specifying the MPLS label:

     ```
     match-inner ipv6
     ```

- Use **allow-no-label** condition to match IPv4 or IPv6 packets irrespective of the MPLS labels being encapsulated or not. Use the allow-no-label condition when you want the decapsulated router traffic from the MPLS network to be transmitted as IPv4 or IPv6 packets:

  match-inner ipv4 *{ipv4-source-prefix | any | host | payload-offset | protocol}{ipv4-destination-prefix | any | host}* **{application | both | ingress | egress}[bidirection] [allow-no-label]**

  or

  match-inner ipv6 *{ipv6-source-prefix | any | host | payload-offset | protocol}{ipv4-destination-prefix | any | host}* **{application | both | ingress | egress}[bidirection] [allow-no-label]**

- Specify the MPLS label and depth to filter IPv4 packets flowing through a particular MPLS interface:

  [interface *interface-name interface-number*] **mpls** *depth-of-mpls-label* **match-inner ipv4** *{ipv4-source-prefix | any | host | payload-offset | protocol}{ipv4-destination-prefix | any | host}* **{application | both | ingress | egress}[bidirection] [allow-no-label]**

- Specify the MPLS label and depth to filter IPv6 packets flowing through a particular MPLS interface:

  [interface *interface-name interface-number*] **mpls** *depth-of-mpls-label* **match-inner ipv6** *{ipv6-source-prefix | any | host | payload-offset | protocol}{ipv6-destination-prefix | any | host}* **{application | both | ingress | egress}[bidirection] [allow-no-label]**

3. Exit the privileged EXEC mode:

   exit

Examples

The following example shows how to use the debug condition **debug platform condition mpls** command to enable matching of IPv4 or IPv6 traffic over MPLS networks :

```
Device# debug platform condition mpls match-inner ipv4
Device# debug platform condition mpls match-inner ipv4 any any
Device# debug platform condition mpls match-inner ipv4 any any both
Device# debug platform condition mpls match-inner ipv4 any any both allow-no-label
```

For more information see, debug platform condition mpls command page.

The following example shows how to use the debug condition **debug platform condition interface mpls** command to enable matching of IPv4 or IPv4 traffic over MPLS networks for a specific interface:

```
Device# debug platform condition interface
Device# debug platform condition interface Loopback 3 mpls
Device# debug platform condition interface Loopback 3 mpls match-inner ipv6 host
2001:db8:3333:4444:5555:6666:7777:8888
Device# debug platform condition interface Loopback 3 mpls match-inner ipv6 host
2001:db8:3333:4444:5555:6666:7777:8888 any both
Device# debug platform condition interface Loopback 3 mpls match-inner ipv6 host
2001:db8:3333:4444:5555:6666:7777:8888 any both allow-no-label
```

Verify the Unified Debug Condition to Match IPv4 and IPv6 Traffic Over MPLS

Verify the Debug State

The following is a sample output from the **show platform conditions** command:

```
Device# show platform conditions

Conditional Debug Global State: Start

Conditions
     Direction
-------------------------------------------------------------------------------------------|---------
All Interfaces                          & MPLS [ALL LABEL] & IPV4 Filter [ALL PROTO] [host
 10.20.24.17] [host 10.20.25.16] both bi



Feature Condition       Type                      Value
----------------------|-----------------------|-------------------------------------------------------



Feature      Type           Submode
                                Level
---------|---------|-------------------------------------------------------------------------|-------
```

In this output, **Conditional Debug Global State: Start** indicates that the debugging is enabled. You can verify the debug filter configuration as well.

Packet Trace Statistics

The following is a sample output from the **show platform packet-trace statistics** command:

```
Device# show platform packet-trace statistics
Packets Summary
  Matched  2
  Traced   2
Packets Received
  Ingress  0
  Inject   0
Packets Processed
  Forward  0
  Punt     0
  Drop     0
  Consume  0

          PKT_DIR_IN
             Dropped       Consumed       Forwarded
INFRA            0             0             0
TCP              0             0             0
UDP              0             0             0
IP               0             0             0
IPV6             0             0             0
ARP              0             0             0
```

```
          PKT_DIR_OUT
              Dropped       Consumed       Forwarded
INFRA               0              0               0
TCP                 0              0               0
UDP                 0              0               0
IP                  0              0               0
IPV6                0              0               0
ARP                 0              0               0
```

In this output, **Matched** and **Traced** indicates the number of matched and traced packets.

Decode the IPv4 and IPv6 Matching over MPLS

The following is a sample output from the **show platform packet-trace** *packet 0* **decode** command:

```
Device# show platform packet-trace packet 0 decode
Packet: 0 CBUG ID: 39872
Summary
Input : GigabitEthernet5
Output : GigabitEthernet1
State : FWD
Timestamp
Start : 10090556741529 ns (12/02/2022 05:54:03.730220 UTC)
Stop : 10090556747391 ns (12/02/2022 05:54:03.730226 UTC)
Path Trace
Feature: MPLS(Output)
Input : GigabitEthernet5
Output : Tunnel1
Label Stack Entry[1]: 0x003eb17f
StackEnd:YES, TTL:127, EXP:0, Label:1003, is SDWAN:YES
SDWAN Proto: IPV4, SDWAN dst_vpn: 1
Feature: MPLS_OUTPUT_L2_REWRITE
Entry : Output - 0x81323e6c
Input : GigabitEthernet5
Output : Tunnel1
Lapsed time : 239 ns
Feature: DEBUG_COND_MAC_EGRESS
Entry : Output - 0x81499d88
Input : GigabitEthernet5
Output : Tunnel1
Lapsed time : 33 ns
Feature: MPLS_OUTPUT_FRAG
Entry : Output - 0x814cdb3c
Input : GigabitEthernet5
Output : Tunnel1
Lapsed time : 152 ns
Feature: SDWAN_LOSS_PROTECT_TX
Entry : Output - 0x814d962c
Input : GigabitEthernet5
Output : Tunnel1
Lapsed time : 15 ns
Feature: MPLS_SDWAN_TUNNEL_OUTPUT_FINAL
Entry : Output - 0x814d60cc
Input : GigabitEthernet1
Output : Tunnel1
Lapsed time : 157 ns
Feature: SDWAN_TUNNEL_PRE_CHK_LKUP
Entry : Output - 0x814d911c
Input : GigabitEthernet1
Output : Tunnel1
Lapsed time : 19 ns
Feature: SDWAN_TUNNEL_PRE_QOS_OUTPUT
Entry : Output - 0x814d956c
Input : GigabitEthernet1
```

```
Output : Tunnel1
Lapsed time : 70 ns
Feature: SDWAN_TUNNEL_OUTPUT_UNIFY_FNF_FINAL
Entry : Output - 0x814aaa68
Input : GigabitEthernet1
Output : Tunnel1
Lapsed time : 95 ns
Feature: IPV4_OUTPUT_IPSEC_SDWAN_FEATURE
Entry : Output - 0x814c7480
Input : GigabitEthernet1
Output : Tunnel1
Lapsed time : 101 ns
Feature: IPV4_OUTPUT_IPSEC_CLASSIFY
Entry : Output - 0x814c7438
Input : GigabitEthernet1
Output : Tunnel1
Lapsed time : 151 ns
Feature: IPV4_IPSEC_FEATURE_RETURN
Entry : Output - 0x814c7478
Input : GigabitEthernet1
Output : Tunnel1
Lapsed time : 35 ns
Feature: IPV4_TUNNEL_PRE_GOTO_OUTPUT
Entry : Output - 0x814d60c4
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 461 ns
Feature: CBUG_OUTPUT_FIA
Entry : Output - 0x81499d68
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 35 ns
Feature: IPV4_VFR_REFRAG
Entry : Output - 0x814c894c
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 33 ns
Feature: DEBUG_COND_APPLICATION_OUT_CLR_TXT
Entry : Output - 0x81499d74
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 11 ns
Feature: IPV4_OUTPUT_L2_REWRITE
Entry : Output - 0x81323e50
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 53 ns
Feature: DEBUG_COND_MAC_EGRESS
Entry : Output - 0x81499d88
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 34 ns
Feature: DEBUG_COND_APPLICATION_OUT
Entry : Output - 0x81499d78
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 11 ns
Feature: IPV4_OUTPUT_FRAG
Entry : Output - 0x814c78e8
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 44 ns
Feature: IPV4_OUTPUT_DROP_POLICY
Entry : Output - 0x814d16b8
```

```
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 247 ns
Feature: IPV4_OUTPUT_SDWAN_FNF_FINAL
Entry : Output - 0x814aaa4c
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 74 ns
Feature: DEBUG_COND_OUTPUT_PKT
Entry : Output - 0x81499d8c
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 24 ns
Feature: MARMOT_SPA_D_TRANSMIT_PKT
Entry : Output - 0x814df374
Input : GigabitEthernet1
Output : GigabitEthernet1
Lapsed time : 780 ns
Packet Copy In
003eb17f 4500002a 0eb70000 7f11ed0a 10000001 30000001 a2c42710 00160000
801296c3 9baf96f9 9b56c437 0aca
Unable to decode layer 2 trying to skip to layer 3
MPLS
Label Stack Entry[1]
TTL : 127
Label : 1003
EXP : 0
StackEnd : YES
SDWAN : YES
SDWAN Label : 1003
SDWAN Proto : IPV4
IPv4
Version : 4
Header Length : 5
ToS : 0x00
Total Length : 42
Identifier : 0x0eb7
IP Flags : 0x0
Frag Offset : 0
TTL : 127
Protocol : 17 (UDP)
Header Checksum : 0xed0a
Source Address : 10.0.0.1
Destination Address : 10.0.0.1
UDP
Source Port : 41668
Destination Port : 10000
Length : 22
Checksum : 0x0000
Decode halted - unsupported udp port number
Packet Copy Out
52540095 dbed5254 007ffb83 08004500 0046ab1a 4000ff2f 9d4d0a01 0f0f0a01
10100000 8847003e b17f4500 002a0eb7 00007f11 ed0a1000 00013000 0001a2c4
27100016 00008012 96c39baf 96f99b56 c4370aca
ARPA
Destination MAC : 5254.0095.dbed
Source MAC : 5254.007f.fb83
Type : 0x0800 (IPV4)
IPv4
Version : 4
Header Length : 5
ToS : 0x00
Total Length : 70
Identifier : 0xab1a
```

```
IP Flags : 0x2 (Don't fragment)
Frag Offset : 0
TTL : 255
Protocol : 47 (GRE)
Header Checksum : 0x9d4d
Source Address : 10.0.0.1
Destination Address : 10.0.0.1
GRE ver 0
Optional Fields : None
Strict Source Route : NO
Recursion Control : 0
Flags : 0x00
Protocol : 0x8847 (MPLS)
MPLS
Label Stack Entry[1]
TTL : 127
Label : 1003
EXP : 0
StackEnd : YES
SDWAN : YES
SDWAN Label : 1003
SDWAN Proto : IPV4
IPv4
Version : 4
Header Length : 5
ToS : 0x00
Total Length : 42
Identifier : 0x0eb7
IP Flags : 0x0
Frag Offset : 0
TTL : 127
Protocol : 17 (UDP)
Header Checksum : 0xed0a
Source Address : 10.255.255.255
Destination Address : 10.255.255.254
UDP
Source Port : 41668
Destination Port : 10000
Length : 22
Checksum : 0x0000
Decode halted - unsupported udp port number
```

In this example, **Source Address** and **Destination Address** indicate that the debugging condition is successful. The **MPLS** section displays the SD-WAN labels specified.