



How does the GigaVUE handle MPLS frames?

Answer: The GigaVUE-MP™ and GigaVUE-420™ will forward all MPLS tagged Ethernet packets from input to output ports. However, the MPLS tags does impact their ability to perform packet filtering using the pre-defined filters such as IP addresses, application ports, etc, since they cannot automatically skip over the MPLS tags to filter on the layer 3 and higher parameters.

There is a workaround using user defined filters which provides a manual method to handle some types of packets with MPLS tags. A unicast MPLS packet has ethertype value of 0x8847 and a multicast MPLS packet has ethertype value of 0x8848. One can use the ethertype filter together with a user defined filter with the proper offset to filter on a MPLS packets with a particular IP address or any parameters that fits inside the two 16-byte (GV-420) or 4 byte (GV-MP) data fields of the user defined filter.

Gigamon FAQs

Offsets:

The value of the offset for the user defined filter is dependent on the number of MPLS labels inside the packet. Each MPLS label is four bytes. For a frame containing a single MPLS label, the IP Source Address will be at an offset of 30. For a frame containing two MPLS labels, the IP Source Address offset is increased by 4 bytes to 34, etc. The following diagrams shows a frame containing a single MPLS label in which the IP Source Address field begins at offset 30 (0x1E).

The image shows a Wireshark packet capture analysis. The packet list pane shows a packet of type GTP <T [TCP segment of a reassembled PDU] with source IP 10.176.13.22 and destination IP 10.184.13.14. The packet details pane shows the following structure:

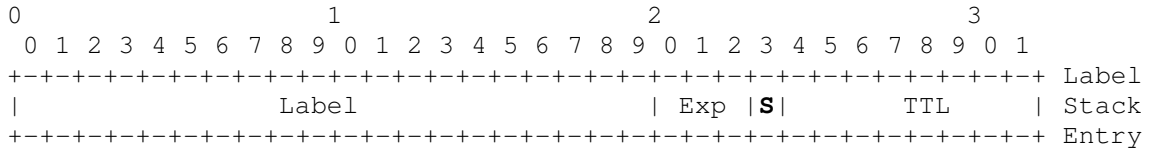
- Ethernet II, Src: Cisco_6e:bb:c0 (00:16:9c:6e:bb:c0), Dst: cisco_6d:26:40 (00:16:9c:6d:26:40)
- MultiProtocol Label Switching Header, Label: 764, Exp: 0, S: 1, TTL: 61
 - MPLS Label: 764
 - MPLS Experimental Bits: 0
 - MPLS Bottom Of Label Stack: 1
 - MPLS TTL: 61
- Internet Protocol, Src: 10.176.13.22 (10.176.13.22), Dst: 10.184.13.14 (10.184.13.14)
 - Version: 4
 - Header length: 20 bytes
 - Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
 - Total Length: 64
 - Identification: 0x768b (30347)
 - Flags: 0x04 (Don't Fragment)
 - Fragment offset: 0
 - Time to live: 62
 - Protocol: TCP (0x06)
 - Header checksum: 0x96a1 [correct]
 - Source: 10.176.13.22 (10.176.13.22)
 - Destination: 10.184.13.14 (10.184.13.14)
- Transmission Control Protocol, Src Port: ldap (389), Dst Port: 42173 (42173), Seq: 0, Ack: 0, Len: 0

The packet bytes pane shows the raw data in hexadecimal and ASCII. A red vertical line is drawn at offset 30 (decimal), which corresponds to the start of the IP source address field (0x1E). The source address is 10.176.13.22.

```
0000 00 16 9c 6d 26 40 00 16 9c 6e bb c0 88 47 00 2f ...m&@.. .n...G./
0010 c1 3d 45 00 00 40 76 8b 40 00 3e 06 96 a1 0a b0 .=E..@v. @.>...
0020 00 1e 0a b8 0d 0e 01 85 a4 bd 61 bd f9 be 6f 0e ..... .a...o.
0030 37 fe b0 10 05 a8 04 e4 00 00 01 01 08 0a 47 01 7..... .G.
0040 8f 38 17 46 22 45 01 01 05 0a 6f 0e 37 e3 6f 0e .8.F"E.. ..o.7.o.
0050 37 fe d4 54 84 5f ..... 7..T..
```

Gigamon FAQs

NOTE: According to RFC 3032, the last tag in an MPLS frame will have the **S bit** set to a one. This can also be filtered on to verify the position of the last tag in an MPLS frame.



Label: Label Value, 20 bits
 Exp: Experimental Use, 3 bits
S: Bottom of Stack, 1 bit
 TTL: Time to Live, 8 bits

For a frame with one MPLS label, the S bit is at offset 14 (0xE), bit 00-00-01-00

The image shows a Wireshark packet capture analysis of an MPLS label. The packet list pane shows a packet of type MultiProtocol Label Switching Header (mpls) with a label value of 764, experimental bits of 0, and the S bit set to 1. The packet details pane shows the MPLS label structure, including the S bit at offset 14. The packet bytes pane shows the raw data of the MPLS label, with the S bit highlighted at offset 14 (0xE).

No.	Time	Source	Destination	Protocol	Info
1	0.000000000	10.176.13.22	10.184.13.14	TCP	ldap > 42173 [ACK] Seq=0 Ack=0 win=1448 Len=0 TSV=1191284536 TSER=390472261 SLE=42949
2	0.000006900	10.176.13.19	10.184.13.14	LDAP	MsgId=1 Bind Result
3	0.000020600	216.220.218.21	10.130.156.40	GTP <T	[TCP segment of a reassembled PDU]
4	0.000027199	10.130.162.126	216.220.209.23	GTP <T	[TCP segment of a reassembled PDU]
5	0.000027800	10.172.219.230	66.135.39.78	GTP <T	4616 > http [ACK] Seq=0 Ack=0 win=41678 Len=0
6	0.000062300	10.110.233.163	216.220.209.23	GTP <T	2545 > 7375 [ACK] Seq=0 Ack=0 win=32720 Len=0 TSV=114944453 TSER=522444783
7	0.000071599	10.176.57.145	10.162.134.11	TCP	[TCP segment of a reassembled PDU]
8	0.000076199	10.176.13.17	10.184.13.14	LDAP	MsgId=1 Bind Result
9	0.000079500	216.155.165.50	10.174.177.138	GTP <T	R080 > 5547 [Ack] Seq=0 Ack=0 win=37640 Len=0

Frame 1 (86 bytes on wire, 86 bytes captured)
 Ethernet II, Src: Cisco_6e:bb:c0 (00:16:9c:6e:bb:c0), Dst: Cisco_6d:26:40 (00:16:9c:6d:26:40)
 MultiProtocol Label Switching Header, Label: 764, Exp: 0, S: 1, TTL: 61

MPLS Label: 764
 MPLS Experimental Bits: 0
 MPLS Bottom Of Label Stack: 1
 MPLS TTL: 61

Internet Protocol, Src: 10.176.13.22 (10.176.13.22), Dst: 10.184.13.14 (10.184.13.14)
 Version: 4
 Header length: 20 bytes
 Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
 Total Length: 64
 Identification: 0x768b (30347)
 Flags: 0x04 (Don't Fragment)
 Fragment offset: 0
 Time to live: 62
 Protocol: TCP (0x06)
 Header checksum: 0x96a1 [connect]

Offset 14 (decimal)

0000 00 16 9c 6d 26 40 00 16 9c 6e bb c0 88 47 00 2f ...m&@...n...G
 0010 45 00 00 40 76 8b 40 00 3e 06 96 a1 0a b0 ...E..@.>.....
 0020 0d 16 0a b8 0d 0e 01 85 a4 bd 61 bd f9 be 6f 0ea...o.
 0030 37 fe b0 10 05 a8 04 e4 00 00 01 01 08 0a 47 01 7.....G.
 0040 8f 38 17 46 22 45 01 01 05 0a 6f 0e 37 e3 6f 0e .8.F"E...o.7.o.
 0050 37 fe d4 54 84 5f 7..T..

S bit

MultiProtocol Label Switching Header (mpls), 4 bytes | P: 8174 D: 8174 M: 0

Gigamon FAQs

Notes on User Defined Filters:

Note 1: In the GigaVUE-420 the UDA offset value is defined in **decimal** and must align on a four byte boundary. The offset boundary begins at byte 2 and may increment as 6, 10, 14, etc, up to byte 110. Therefore the UDA can examine up to the first 126 bytes in a packet.

Note 2: In the GigaVUE-MP the UDA offset value is defined in **hexadecimal** and must align on a four byte boundary. The offset boundary begins at byte 0 and may increment as 4, 8, 12, etc, up to decimal value of 72. The UDF can examine up to the first 76 bytes in a packet.

Note 3: A difference between the User Defined Attributes for the GigaVUE-420 and the User Defined Filters for the GigaVUE-MP is the way the offsets are defined. In the GigaVUE-420 the two UDA offsets are global for all the User Defined Filters. In the GigaVUE-MP each User Defined Filter has its own User Defined Offset associated with that filter.

Note 4: In the GigaVUE-MP the UDF data field is 4 bytes long and in the GigaVUE-420 the UDA data field is 16 bytes long.

Gigamon FAQs

Example #1A: Here is an example of a mapping script for the GigaVUE-420 which will filter and forward traffic to and from IP network 10.98.0.0 for frames which have one MPLS label. The uda1 filter will be used to filter on the S bit and the uda2 filter will be used to filter on the IP Source and Destination Addresses.

```
config uda uda1_offset 14
```

```
config uda uda2_offset 30
```

```
config map type st alias mpls1
```

```
#unicast MPLS and IP source network 10.98.0.0
```

```
config map-rule mpls1 rule ethertype 8847 uda1_data 0000100-00000000-00000000-00000000 uda1_mask 0000100-00000000-00000000-00000000 uda2_data 0a620000-00000000-00000000-00000000 uda2_mask ffff0000-00000000-00000000-00000000 tool 1
```

```
#unicast MPLS and IP destination network 10.98.0.0
```

```
config map-rule mpls1 rule ethertype 8847 uda1_data 0000100-00000000-00000000-00000000 uda1_mask 0000100-00000000-00000000-00000000 uda2_data 00000000-0a620000-00000000-00000000 uda2_mask 00000000-ffff0000-00000000-00000000 tool 1
```

```
#multicast MPLS and IP source network 10.98.0.0
```

```
config map-rule mpls1 rule ethertype 8848 uda1_data 0000100-00000000-00000000-00000000 uda1_mask 0000100-00000000-00000000-00000000 uda2_data 0a620000-00000000-00000000-00000000 uda2_mask ffff0000-00000000-00000000-00000000 tool 1
```

```
#multicast MPLS and IP destination network 10.98.0.0
```

```
config map-rule mpls1 rule ethertype 8848 uda1_data 0000100-00000000-00000000-00000000 uda1_mask 0000100-00000000-00000000-00000000 uda2_data 00000000-0a620000-00000000-00000000 uda2_mask 00000000-ffff0000-00000000-00000000 tool 1
```

```
config mapping net 4 map mpls1
```

Gigamon FAQs

Example #1B: Here is the same example of a mapping script for the GigaVUE-MP which will filter and forward traffic to and from IP network 10.98.0.0 for frames which have one MPLS label. The uda1 filter will be used to filter on the S bit and the uda2 filter will be used to filter on the IP Source and Destination Addresses.

config map type st alias mpls1

#unicast MPLS and IP source network 10.98.0.0

config map-rule mpls1 rule ethertype 8847 offset1 10 data1 01000000 mask1 01000000 offset2 1c data2 00000a62 mask2 0000ffff tool 4

#unicast MPLS and IP destination network 10.98.0.0

config map-rule mpls1 rule ethertype 8847 offset1 10 data1 01000000 mask1 01000000 offset2 20 data2 000a6200 mask2 00ffff00 tool 4

#multicast MPLS and IP source network 10.98.0.0

config map-rule mpls1 rule ethertype 8848 offset1 10 data1 01000000 mask1 01000000 offset2 1c data2 00000a62 mask2 0000ffff tool 4

#multicast MPLS and IP destination network 10.98.0.0

config map-rule mpls1 rule ethertype 8848 offset1 10 data1 01000000 mask1 01000000 offset2 20 data2 000a6200 mask2 00ffff00 tool 4

config mapping net 8 map mpls1

Technical support contacts:

Email: support@gigamon.com

Phone: 408-263-2024.



Gigamon FAQs
