Configure QoS over Tunnel GRE

Contents

Introduction

This document describes how to configure and troubleshoot QoS over tunnel GRE in Nexus 9300 (EX-FX-GX) model.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- QoS
- Tunnel GRE
- Nexus 9000

Components Used

The information in this document is based on these software and hardware versions:

- Hardware: N9K-C9336C-FX2
- Version: 9.3(8)

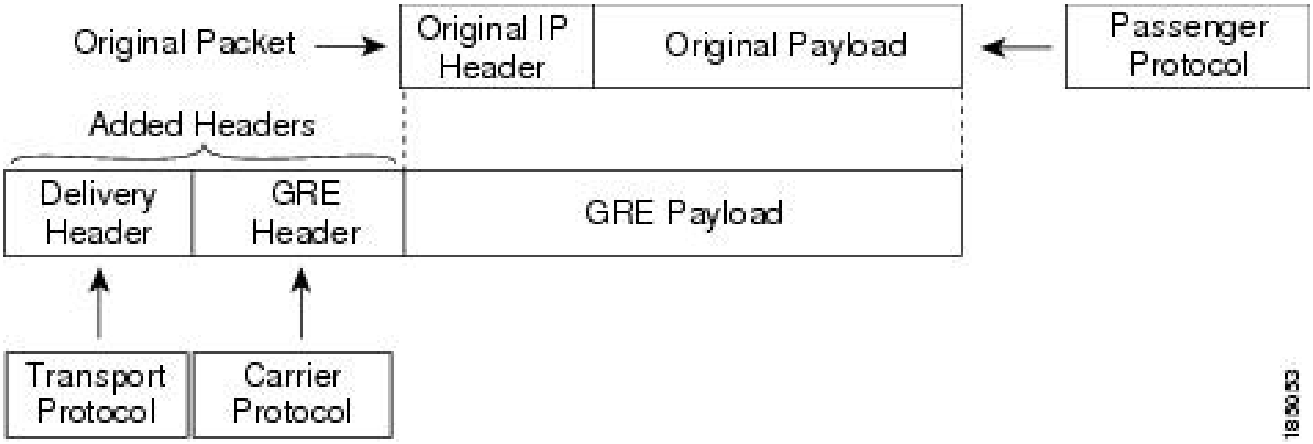
The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

You can use generic routing encapsulation (GRE) as the carrier protocol for a variety of passenger protocols.

You see in the image that the IP tunnel components for a GRE tunnel. The original passenger protocol packet becomes the GRE payload and the device adds a GRE header to the packet.

The device then adds the transport protocol header to the packet and transmits it.

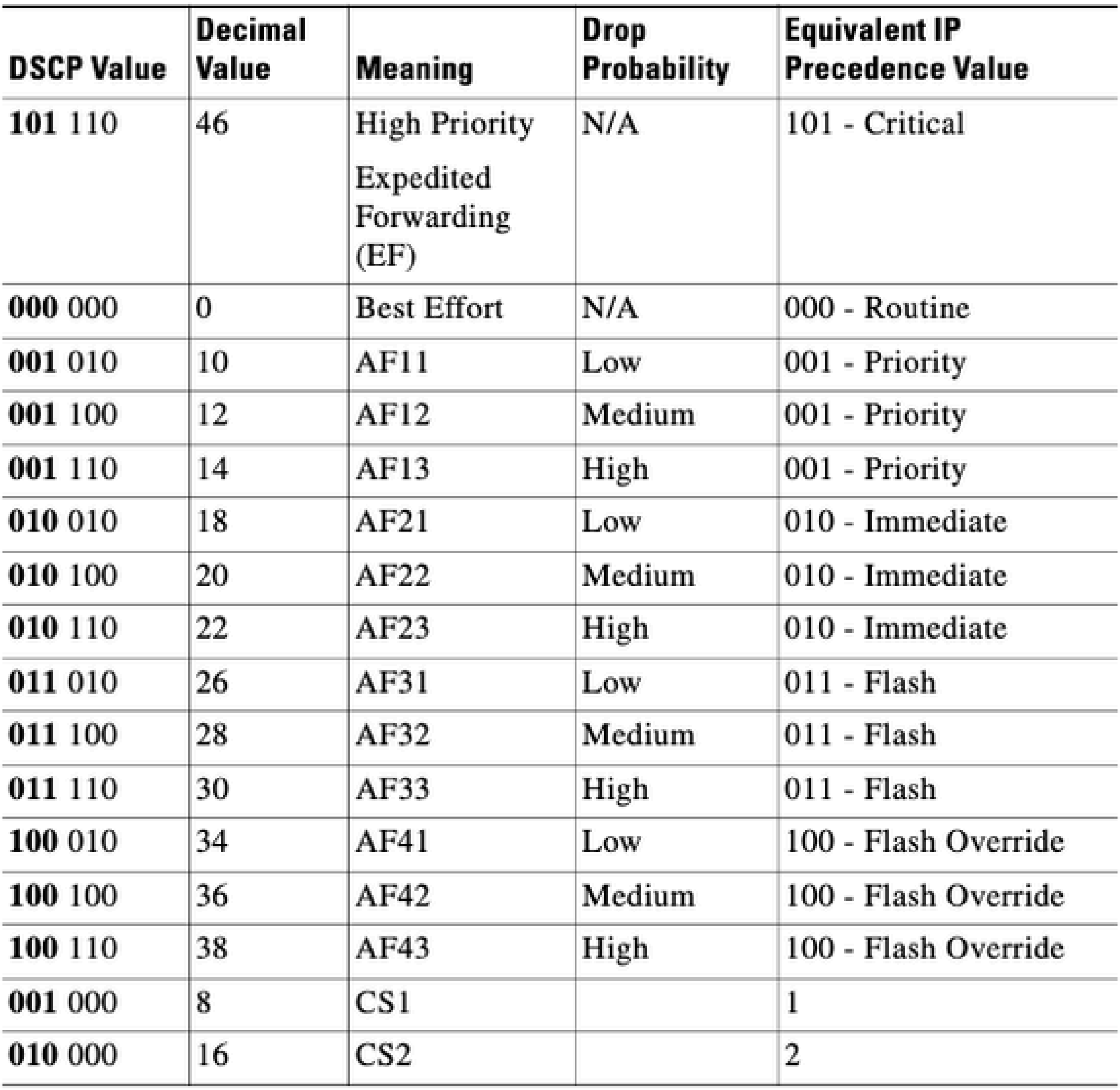


Traffic is processed based on how you classify it and the policies that you create and apply to traffic classes.

To configure QoS features, use these steps:

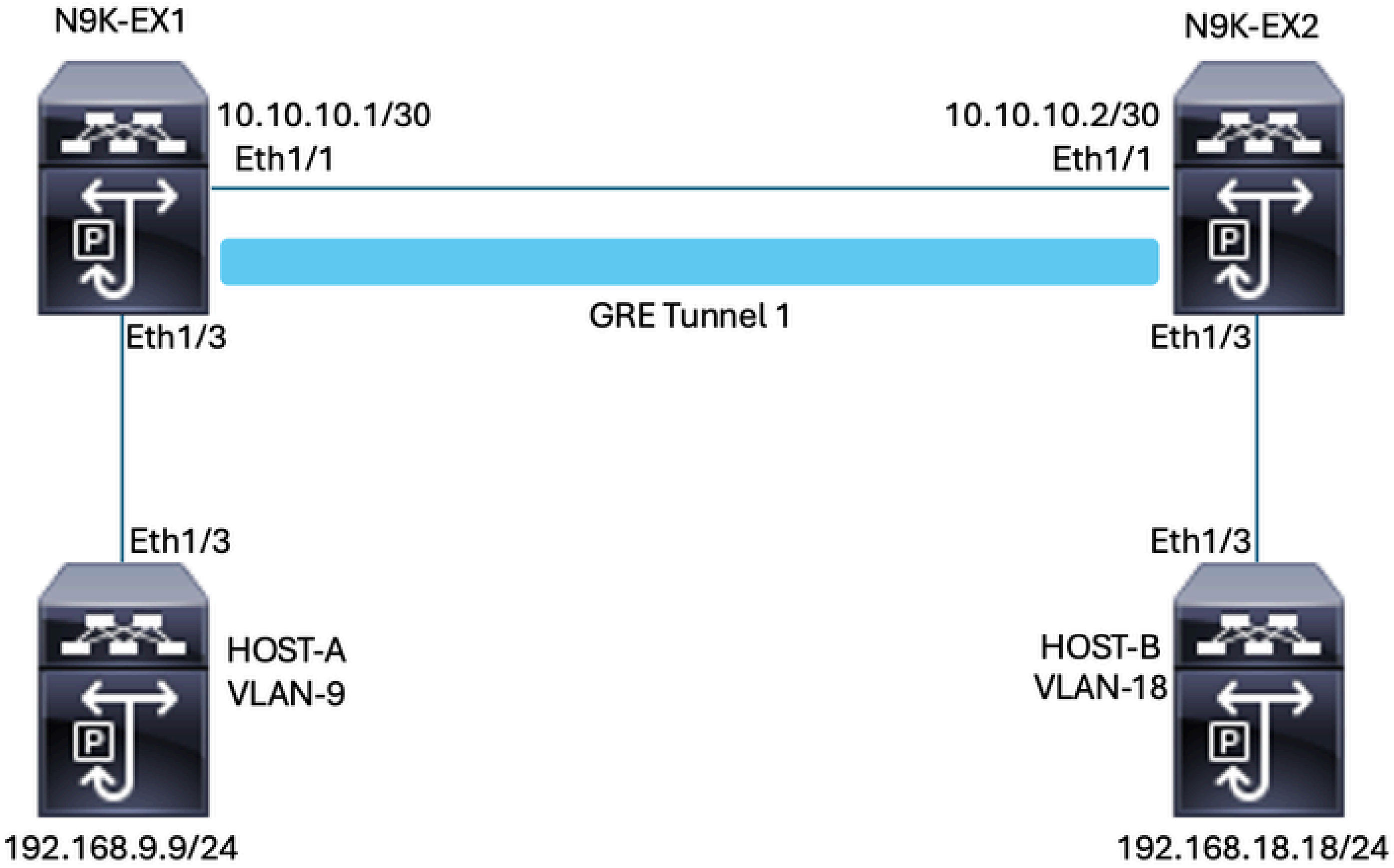
1. Classes are created that classify ingress packets to the nexus that match criteria such as IP address or QoS fields.
2. Creates policies that specify the actions to be taken on traffic classes, such as watch, mark, or discard packets.
3. Apply policies to a port, port channel, VLAN, or subinterface.

Commonly Used DSCP Values



DSCP Value	Decimal Value	Meaning	Drop Probability	Equivalent IP Precedence Value
101 110	46	High Priority Expedited Forwarding (EF)	N/A	101 - Critical
000 000	0	Best Effort	N/A	000 - Routine
001 010	10	AF11	Low	001 - Priority
001 100	12	AF12	Medium	001 - Priority
001 110	14	AF13	High	001 - Priority
010 010	18	AF21	Low	010 - Immediate
010 100	20	AF22	Medium	010 - Immediate
010 110	22	AF23	High	010 - Immediate
011 010	26	AF31	Low	011 - Flash
011 100	28	AF32	Medium	011 - Flash
011 110	30	AF33	High	011 - Flash
100 010	34	AF41	Low	100 - Flash Override
100 100	36	AF42	Medium	100 - Flash Override
100 110	38	AF43	High	100 - Flash Override
001 000	8	CS1		1
010 000	16	CS2		2

Network Diagram



Configure

The goal of the configuration of QoS over tunnel GRE is to set a DSCP for traffic of a certain VLAN to pass through the GRE Tunnel between N9K-EX1 and N9K-EX2.

The Nexus encapsulates the traffic and sends it on the Tunnel GRE without loss of QoS marking as you did previously in the VLAN for the DSCP value, for this case the value of DSCP AF-11 is used for VLAN 9.

Host-A

```
interface Ethernet1/3
  switchport
  switchport access vlan 9
  no shutdown

interface Vlan9
  no shutdown
  ip address 192.168.9.9/24
```

Host-B

```
interface Ethernet1/3
switchport
switchport access vlan 18
no shutdown
```

```
interface Vlan18
no shutdown
ip address 192.168.18.18/24
```

N9K-EX1 interfaces configuration

```
interface Ethernet1/1
ip address 10.10.10.1/30
no shutdown

interface Ethernet1/3
switchport
switchport access vlan 9
no shutdown

interface Tunnel1
ip address 172.16.1.1/30
tunnel source Ethernet1/1
tunnel destination 10.10.10.2
no shutdown

interface Vlan9
no shutdown
ip address 192.168.9.1/24
```

N9K-EX1 Routing configuration

```
ip route 0.0.0.0/0 Tunnel
```

N9K-EX1 QoS configuration

Since QoS is not supported on the GRE tunnel interface in NXOS, it is necessary to configure and apply the service policy in the VLAN configuration. As you can see, first create the ACL to match the source and destination, then set the QoS configuration with the desired DSCP, finally use the service policy the VLAN configuration.

```
ip access-list TAC-QoS-GRE
10 permit ip any 192.168.18.0/24
class-map type qos match-all CM-TAC-QoS-GRE
match access-group name TAC-QoS-GRE
policy-map type qos PM-TAC-QoS-GRE
class CM-TAC-QoS-GRE
set dscp 10

vlan configuration 9
service-policy type qos input PM-TAC-QoS-GRE
```

N9K-EX2 Interfaces configuration

```
interface Ethernet1/1
ip address 10.10.10.2/30
no shutdown

interface Ethernet1/3
switchport
switchport access vlan 18
no shutdown

interface Tunnel1
ip address 172.16.1.2/30
tunnel source Ethernet1/1
tunnel destination 10.10.10.1
no shutdown

interface Vlan18
no shutdown
ip address 192.168.18.1/24
```

N9K-EX2 Routing configuration

```
ip route 0.0.0.0/0 Tunnel
```

Troubleshoot

Tunnel Verification

Both commands:

- **show ip interface brief**
- **show interface tunnel 1 brief**

Displays if the tunnel is Up.

```
N9K-EX1# show ip interface brief

IP Interface Status for VRF "default"(1)
Interface IP Address Interface Status
Vlan9 192.168.9.1 protocol-up/link-up/admin-up
Tunnel1 172.16.1.1 protocol-up/link-up/admin-up
Eth1/1 10.10.10.1 protocol-up/link-up/admin-up
```

```
N9K-EX1# show interface tunnel 1 brief

--------------------------------------------------------------------------------
---------------------------
```

```
Interface Status IP Address
Encap type MTU
--------------------------------------------------------------------------------
---------------------------
Tunnel1 up 172.16.1.1/30
GRE/IP 1476
```

Both commands

- **show interface tunnel 1**
- **show interface tunnel 1 counters**

Displays similar information such as as received and transmitted packets.

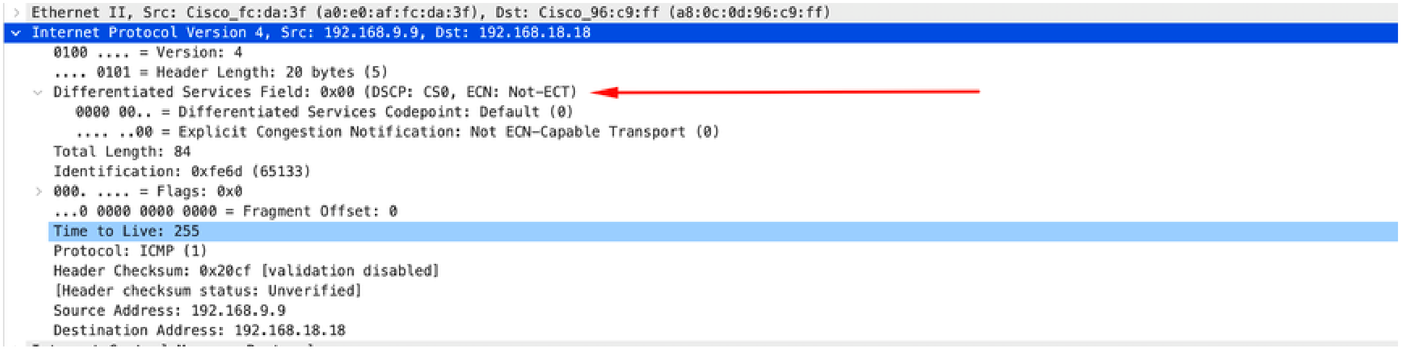
```
N9K-EX1# show interface tunnel 1
Tunnel1 is up
Admin State: up
Internet address is 172.16.1.1/30
MTU 1476 bytes, BW 9 Kbit
Tunnel protocol/transport GRE/IP
Tunnel source 10.10.10.1 (Ethernet1/1), destination 10.10.10.2
Transport protocol is in VRF "default"
Tunnel interface is in VRF "default"
Last clearing of "show interface" counters never
Tx
3647 packets output, 459522 bytes
Rx
3647 packets input, 459522 bytes

N9K-EX1# show interface tunnel 1 counters

--------------------------------------------------------------------------------
--
Port InOctets InUcastPk
ts
--------------------------------------------------------------------------------
--
Tunnel1 459522 36
47

--------------------------------------------------------------------------------
--
Port InMcastPkts InBcastPk
ts
--------------------------------------------------------------------------------
--
Tunnel1 --
--

--------------------------------------------------------------------------------
--
Port OutOctets OutUcastPk
ts
--------------------------------------------------------------------------------
--
Tunnel1 459522 36
47
```

```
--------------------------------------------------------------------------------
--
Port OutMcastPkts OutBcastPk
ts
--------------------------------------------------------------------------------
--
Tunnel1 --
--
N9K-EX1#
```

Traffic Captures

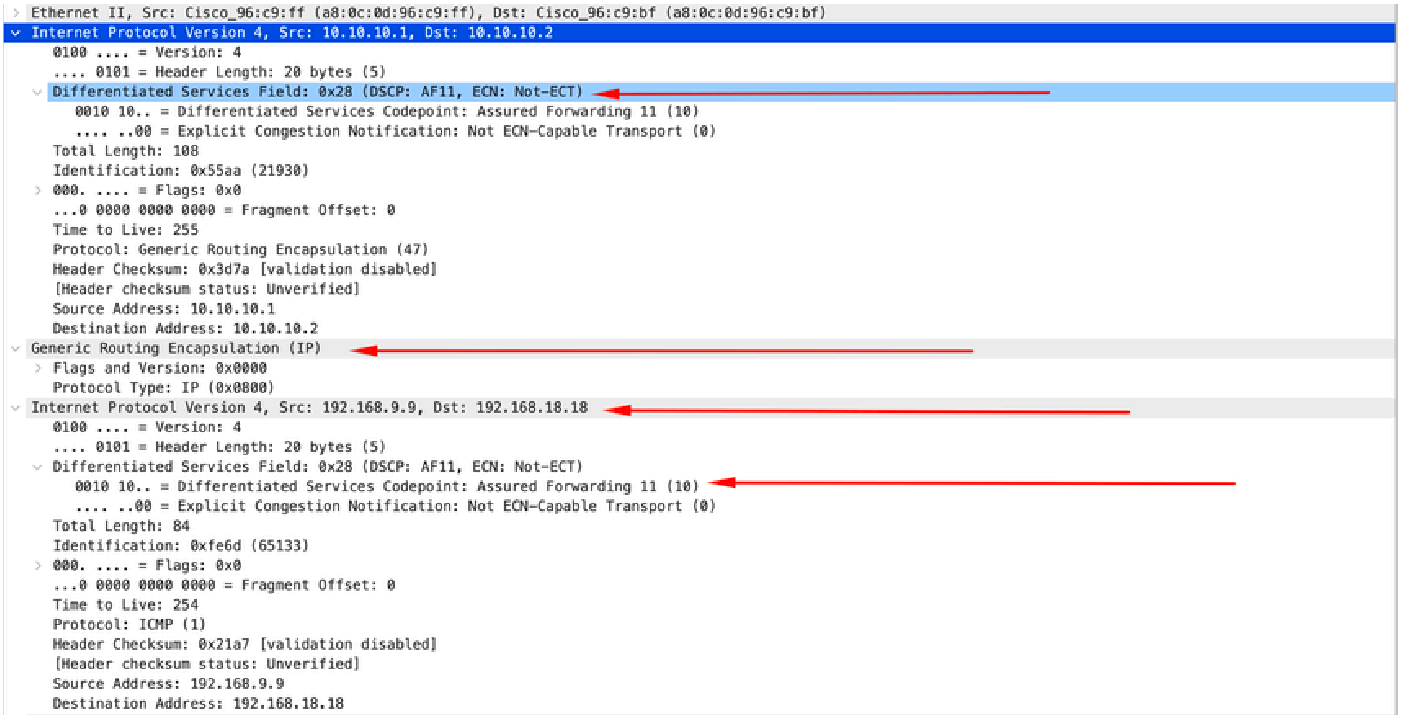
SPAN Captures

This image shows the capture of the ARP request at the entry of the Interface Ethernet 1/3 on the N9K-EX1 switch. You can see that the traffic is not marked with the DSCP (AF11) you want to use yet since the capture is at the input of the switch.



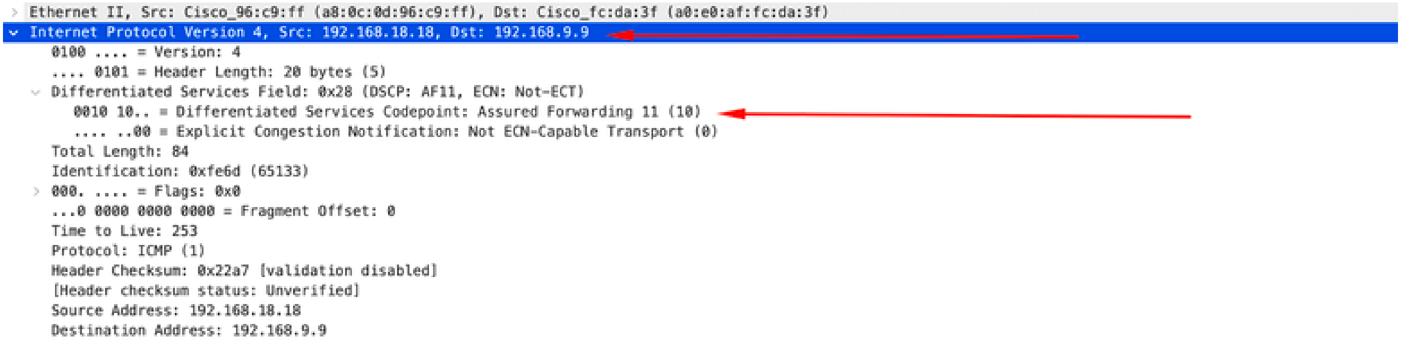
```
> Ethernet II, Src: Cisco_fc:da:3f (a0:e0:af:fc:da:3f), Dst: Cisco_96:c9:ff (a8:0c:0d:96:c9:ff)
v Internet Protocol Version 4, Src: 192.168.9.9, Dst: 192.168.18.18
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  v Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      0000 00.. = Differentiated Services Codepoint: Default (0)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 84
    Identification: 0xfe6d (65133)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
    Protocol: ICMP (1)
    Header Checksum: 0x20cf [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.9.9
    Destination Address: 192.168.18.18
```

The image shows the capture of the ARP request at the entry of the Interface Ethernet 1/1 on the N9K-EX2 switch. You can see that the traffic already has the DSCP AF11 value that you need to use. You also notice that the packet is encapsulated by the tunnel that is configured between the two Nexus.



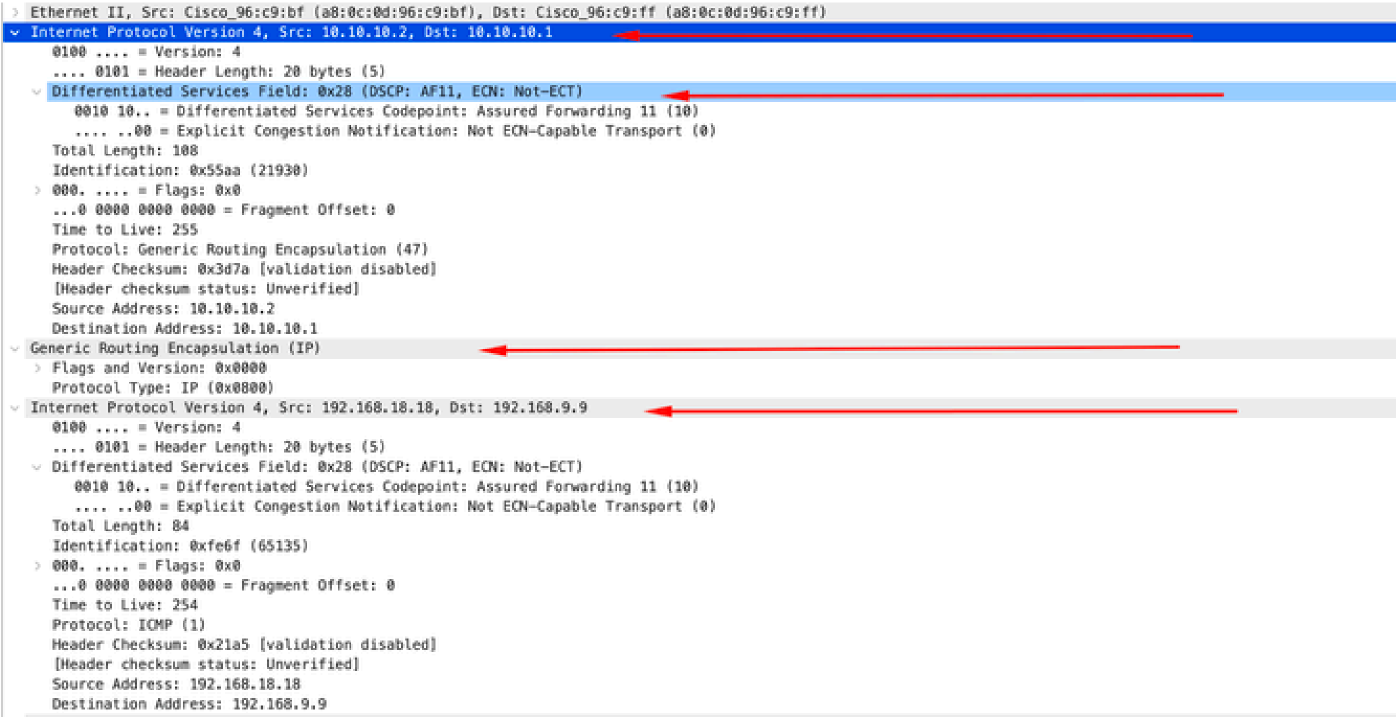
```
> Ethernet II, Src: Cisco_96:c9:ff (a8:0c:0d:96:c9:ff), Dst: Cisco_96:c9:bf (a8:0c:0d:96:c9:bf)
v Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.10.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  v Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)
      0010 10.. = Differentiated Services Codepoint: Assured Forwarding 11 (10)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 108
    Identification: 0x55aa (21930)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
    Protocol: Generic Routing Encapsulation (47)
    Header Checksum: 0x3d7a [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.10.10.1
    Destination Address: 10.10.10.2
v Generic Routing Encapsulation (IP)
  > Flags and Version: 0x0000
    Protocol Type: IP (0x0800)
v Internet Protocol Version 4, Src: 192.168.9.9, Dst: 192.168.18.18
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  v Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)
      0010 10.. = Differentiated Services Codepoint: Assured Forwarding 11 (10)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 84
    Identification: 0xfe6d (65133)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 254
    Protocol: ICMP (1)
    Header Checksum: 0x21a7 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.9.9
    Destination Address: 192.168.18.18
```

The image shows the capture of the ARP reply at the output of the interface Ethernet 1/3 on the N9K-EX1 switch. You can see that the traffic still has the DSCP AF11 value that you need to use. You also notice that the packet is not encapsulated by the tunnel that is configured between the two Nexus.



```
> Ethernet II, Src: Cisco_96:c9:ff (a8:0c:0d:96:c9:ff), Dst: Cisco_fc:da:3f (a0:e0:af:fc:da:3f)
v Internet Protocol Version 4, Src: 192.168.18.18, Dst: 192.168.9.9
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  v Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)
      0010 10.. = Differentiated Services Codepoint: Assured Forwarding 11 (10)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 84
    Identification: 0xfe6d (65133)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 253
    Protocol: ICMP (1)
    Header Checksum: 0x22a7 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.18.18
    Destination Address: 192.168.9.9
```

This image shows the capture of the ARP reply at the output of the Interface Ethernet 1/1 on the N9K-EX2 switch. You can see that the traffic still has the DSCP AF11 value that you need to use. You also notice that the packet is encapsulated by the tunnel that is configured between the two Nexus.



```
> Ethernet II, Src: Cisco_96:c9:bf (a8:0c:0d:96:c9:bf), Dst: Cisco_96:c9:ff (a8:0c:0d:96:c9:ff)
v Internet Protocol Version 4, Src: 10.10.10.2, Dst: 10.10.10.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  v Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)
      0010 10.. = Differentiated Services Codepoint: Assured Forwarding 11 (10)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 108
    Identification: 0x55aa (21930)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
    Protocol: Generic Routing Encapsulation (47)
    Header Checksum: 0x3d7a [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.10.10.2
    Destination Address: 10.10.10.1
v Generic Routing Encapsulation (IP)
  > Flags and Version: 0x0000
    Protocol Type: IP (0x0800)
v Internet Protocol Version 4, Src: 192.168.18.18, Dst: 192.168.9.9
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  v Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)
      0010 10.. = Differentiated Services Codepoint: Assured Forwarding 11 (10)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 84
    Identification: 0xfe6f (65135)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 254
    Protocol: ICMP (1)
    Header Checksum: 0x21a5 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.18.18
    Destination Address: 192.168.9.9
```

It is important to note that the packet captures do not show the tunnel IP for encapsulation since the Nexus uses the physical ones. This is the natural behavior of the Nexus when using GRE tunneling since they use the physical ips to route the packages.

ELAM Capture

You use the ELAM capture on N9KEX-2 with in-select 9 to see the outer l3 and inner l3 header. You must filter by the source and target IP.

```
debug platform internal tah elam
trigger init in-select 9
reset
set inner ipv4 src_ip 192.168.9.9 dst_ip 192.168.18.18
start
```

```
report
```

You can verify that the Nexus receives the packet via interface 1/1. Also, you see the outer l3 header is the physical IP address of the interfaces that are directly connected and the l3 inner header has the IPs of the host A and host B.

```
SUGARBOWL ELAM REPORT SUMMARY
slot - 3, asic - 1, slice - 0
============================

Incoming Interface: Eth1/1
Src Idx : 0x41, Src BD : 4433
Outgoing Interface Info: dmod 2, dpid 10
Dst Idx : 0x3, Dst BD : 18

Packet Type: IPv4

Outer Dst IPv4 address: 10.10.10.2
Outer Src IPv4 address: 10.10.10.1
Ver = 4, DSCP = 10, Don't Fragment = 0
Proto = 47, TTL = 255, More Fragments = 0
Hdr len = 20, Pkt len = 108, Checksum = 0x3d7a

Inner Payload
Type: IPv4

Inner Dst IPv4 address: 192.168.18.18
Inner Src IPv4 address: 192.168.9.9

L4 Protocol : 47
L4 info not available

Drop Info:
----------

LUA:
LUB:
LUC:
LUD:
Final Drops:
```

Troubleshooting QoS

You can check the QoS configuration as shown .

```
N9K-EX1# show running-config ipqos

!Command: show running-config ipqos
!Running configuration last done at: Thu Apr 4 11:45:37 2024
!Time: Fri Apr 5 11:50:54 2024

version 9.3(8) Bios:version 08.39
class-map type qos match-all CM-TAC-QoS-GRE
```

```
match access-group name TAC-QoS-GRE
policy-map type qos PM-TAC-QoS-GRE
class CM-TAC-QoS-GRE
set dscp 10

vlan configuration 9
service-policy type qos input PM-TAC-QoS-GRE
```

You can display the QoS policies configured on the specified VLAN, and also the packets that are matching with the ACL associated to the policy-map.

```
N9K-EX1# show policy-map vlan 9


Global statistics status : enabled

Vlan 9

Service-policy (qos) input: PM-TAC-QoS-GRE
SNMP Policy Index: 285219173

Class-map (qos): CM-TAC-QoS-GRE (match-all)

Slot 1
5 packets
Aggregate forwarded :
5 packets
Match: access-group TAC-QoS-GRE
set dscp 10
```

You can also clear the QoS statistics with the command shown here.

```
N9K-EX1# clear qos statistics
```

Verify ACL programmed in software.

```
N9K-EX1# show system internal access-list vlan 9 input entries detail

slot 1
=======


Flags: F - Fragment entry E - Port Expansion
D - DSCP Expansion M - ACL Expansion
T - Cross Feature Merge Expansion
N - NS Transit B - BCM Expansion C - COPP
```

```
INSTANCE 0x2
---------------

Tcam 1 resource usage:
----------------------
LBL B = 0x1
Bank 2
------
IPv4 Class
Policies: QoS
Netflow profile: 0
Netflow deny profile: 0
Entries:
[Index] Entry [Stats]
---------------------
[0x0000:0x0000:0x0700] permit ip 0.0.0.0/0 192.168.18.0/24 [5]
```

Verify ACL programmed in hardware.

```
N9K-EX1# show hardware access-list vlan 9 input entries detail

slot 1
=======


Flags: F - Fragment entry E - Port Expansion
D - DSCP Expansion M - ACL Expansion
T - Cross Feature Merge Expansion
N - NS Transit B - BCM Expansion C - COPP


INSTANCE 0x2
---------------

Tcam 1 resource usage:
----------------------
LBL B = 0x1
Bank 2
------
IPv4 Class
Policies: QoS
Netflow profile: 0
Netflow deny profile: 0
Entries:
[Index] Entry [Stats]
---------------------
[0x0000:0x0000:0x0700] permit ip 0.0.0.0/0 192.168.18.0/24 [5]
```

With the command shown here, you can verify the ports that are using the VLAN. In this example, it would be VLAN ID 9, and you can also note the QoS policy that is in use.

```
N9K-EX1# show system internal ipqos vlan-tbl 9
```

```
Vlan range asked: 9 - 9

=================================================

Vlan: 9, pointer: 0x132e3eb4, Node Type:  VLAN

IfIndex array:

    alloc count:    5, valid count:    1, array ptr : 0x13517aac          0: IfI

ndex: 0x1a000400 (Ethernet1/3)     Policy Lists (1): Flags: 01

        Type: INP QOS, Name: PM-TAC-QoS-GRE, Ghost Id: 0x45001c7, Real Id: 0x450

01c8

                       Defnode Id: 0x45001c9



=================================================

N9K-EX1#
```