

Solución de problemas del paquete Ethernet dañado en Cisco Nexus 9000

Contenido

[Introducción](#)

[Antecedentes](#)

[Cómo procesa un paquete un switch](#)

[Relleno modificado con VLAN etiquetadas cuando el tráfico atraviesa N9K](#)

[Solución](#)

Introducción

Este documento describe cómo resolver problemas del paquete Ethernet dañado en Cisco Nexus 9000 cuando la información de relleno está dañada o mal formada.

Antecedentes

El tamaño mínimo de una trama Ethernet es de 64 bytes, independientemente de que la etiqueta VLAN esté presente o no.

El tamaño mínimo de carga útil Ethernet es:

- 46 bytes si la etiqueta VLAN está ausente.
- 42 bytes si la etiqueta VLAN está presente.

Puede verificar este hecho:

- En Wikipedia, sección **Carga útil**: https://en.wikipedia.org/wiki/Ethernet_frame
- En el estándar IEEE 802.3 (http://people.ee.duke.edu/~mbrooke/EE164.02/Spring_2004/group_2/index_files/8023.pdf), donde el formato de trama MAC (sin VLAN) se define en la sección 3.1.1, página 39, y los elementos de una trama MAC etiquetada se definen en la sección 3.5 de la página 43.

El tamaño mínimo de un paquete Ethernet es de 64 bytes, independientemente de que el encabezado VLAN esté o no presente. El servidor puede enviar un paquete de 64 bytes que contiene una VLAN, que debe aceptar y procesar correctamente.

Nota: Este comportamiento es manejado correctamente por un Catalyst 4500x no por Nexus 9k.

Cómo procesa un paquete un switch

Paso 1. Reciba una trama Ethernet **VÁLIDA** de 64 bytes.

Paso 2. Quite la secuencia de verificación de tramas (FCS), de modo que el paquete tenga 60

bytes.

Paso 3. Quite la etiqueta VLAN, de modo que el paquete tenga 56 bytes.

Paso 4. Agregue relleno para hacer que el paquete tenga 60 bytes.

Paso 5. Agregue el FCS, haciendo que el paquete tenga 64 bytes de longitud.

El relleno no debe modificarse cuando un paquete pasa por un switch de conexión directa.

Relleno modificado con VLAN etiquetadas cuando el tráfico atraviesa N9K

En lugar de relleno con ceros, el paquete se agrega con caracteres basura, en la mayoría de los casos no tiene impacto porque las sumas de comprobación no se modifican y por lo tanto nadie utiliza estos datos. Sin embargo, si los clientes tienen un uso especial y necesitan volver a calcular las sumas de comprobación, estos datos no utilizados provocan una corrupción de las sumas de comprobación al final (otros dispositivos, como los equilibradores de carga/NAT, podrían ver el problema también).

El dispositivo es un N9K 93120TX (detectado inicialmente en un 9372TX), la versión es la última versión de NXOS 7.0(3)I2(2a).

Utilice hosts Linux con hardware conectado directamente al N9K (sin virtualización de ningún tipo) aquí (enlaces 1000base-T).

Utilice esta configuración:

```
interface Ethernet1/59
    switchport mode trunk
!
```

```
interface Ethernet1/60
    switchport mode trunk
```

linux configurations:

```
inet 10.2.1.1/24 brd 10.2.1.255 scope global eth1 <= native vlan
inet 10.1.1.1/24 brd 10.1.1.255 scope global eth1.100 <= tagged vlan 100
```

or

Sólo tiene que conectar el host de windows y enviar las tramas etiquetadas, deben activar el problema. Además, asegúrese de que la tarjeta de interfaz de red (NIC) tenga la capacidad de etiquetar el paquete.

El switch agrega el relleno distinto de cero a las tramas que pasan.

Por ejemplo: Host — [Trunk] N9K [Trunk] — Host

Puede utilizar netcat para enviar y recibir los paquetes.

Como se muestra en la imagen, envía Side (VLAN 100 etiquetada), puerto e1/59 en el switch.

```
6: eth1.100@eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc noqueue state UP group default
link/ether 44:a8:42:2c:5f:c4 brd ff:ff:ff:ff:ff:ff
inet 10.1.1.1/24 brd 10.1.1.255 scope global eth1.100
    valid_lft forever preferred_lft forever
inet6 fe80::46a8:42ff:fe2c:5fc4/64 scope link
    valid_lft forever preferred_lft forever
```

```
root@s35-c2-0:~# nc 10.1.1.2 3002 -u
a
^C
root@s35-c2-0:~#
```

Recibe Side (VLAN 100 etiquetada), puerto e1/60 en el switch, como se muestra en la imagen:

```
7: eth1.100@eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc noqueue state UP group default
link/ether 44:a8:42:2c:63:d1 brd ff:ff:ff:ff:ff:ff
inet 10.1.1.2/24 brd 10.1.1.255 scope global eth1.100
    valid_lft forever preferred_lft forever
inet6 fe80::46a8:42ff:fe2c:63d1/64 scope link
    valid_lft forever preferred_lft forever
```

```
root@s35-c2:~# nc -l -u -p 3002
a
^C
root@s35-c2:~#
```

Como se muestra en la imagen, el paquete se transmite.

```
root@s35-c2-0:~# tcpdump -i eth1.100 -nvex
tcpdump: listening on eth1.100, link-type EN10MB (Ethernet), capture size 65535 bytes
10:42:20.953994 44:a8:42:2c:5f:c4 > 44:a8:42:2c:63:d1, ethertype IPv4 (0x0800), length 44: (tos 0x0, ttl 64, id 64283, offset 0, flags [DF], proto UDP (17), length 30)
10.1.1.1.41675 > 10.1.1.2.3002: UDP, length 2
0x0000: 4500 001e fb1b 4000 4011 29af 0a01 0101
0x0010: 0a01 0102 a2cb 0bba 000a 1620 610a

^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
root@s35-c2-0:~#
```

Se recibe el paquete, como se muestra en la imagen:

```

10:43:12.665897 44:a8:42:2c:5f:c4 > 44:a8:42:2c:63:d1, ethertype IPv4 (0x0800), length 60: (tos 0x0, ttl 64, id 64283, offset 0, flags [DF], proto UDP
(17), length 30)
  10.1.1.1.41675 > 10.1.1.2.3002: UDP, length 2
    0x0000: 4500 001e fb1b 4000 4011 29af 0a01 0101
    0x0010: 0a01 0102 a2cb 0bba 000a da45 610a 0000
    0x0020: 0000 0000 0000 0000 0000 7562 710e
AC
7 packets captured
7 packets received by filter
0 packets dropped by kernel
root@s35-c2:~#

```

Como se muestra en la imagen, se resalta el relleno incorrecto.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.1.1.1	10.1.1.2	UDP	60	Source port: 40849 Destination port: 3002


```

▶ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
▼ Ethernet II, Src: Dell_2c:5f:c4 (44:a8:42:2c:5f:c4), Dst: Dell_2c:63:d1 (44:a8:42:2c:63:d1)
  ▶ Destination: Dell_2c:63:d1 (44:a8:42:2c:63:d1)
  ▶ Source: Dell_2c:5f:c4 (44:a8:42:2c:5f:c4)
  Type: IP (0x0800)
  Padding: 000000000000000000000000f1b7bc5c
▼ Internet Protocol Version 4, Src: 10.1.1.1 (10.1.1.1), Dst: 10.1.1.2 (10.1.1.2)
  0100 ... = Version: 4
  ... 0101 = Header Length: 20 bytes
  ▶ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
  Total Length: 30
  Identification: 0xfb1d (64285)
  ▶ Flags: 0x02 (Don't Fragment)
  Fragment offset: 0
  Time to live: 64
  Protocol: UDP (17)
  ▶ Header checksum: 0x29ad [validation disabled]
  Source: 10.1.1.1 (10.1.1.1)
  Destination: 10.1.1.2 (10.1.1.2)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
▼ User Datagram Protocol, Src Port: 40849 (40849), Dst Port: 3002 (3002)
  Source Port: 40849 (40849)
  Destination Port: 3002 (3002)
  Length: 10
  ▼ Checksum: 0xdd7f [validation disabled]
    [Good Checksum: False]
    [Bad Checksum: False]
    [Stream index: 0]
▼ Data (2 bytes)
  Data: 610a
  [Length: 2]

```



```

0000 44 a8 42 2c 63 d1 44 a8 42 2c 5f c4 08 00 45 00  D.B,c.D. B,....E.
0010 00 1e fb 1d 40 00 40 11 29 ad 0a 01 01 01 0a 01  ....@.@.).....
0020 01 02 9f 91 0b ba 00 0a dd 7f 61 0a 00 00 00 00  .....a.....
0030 00 00 00 00 00 00 00 00 f1 b7 bc 5c  .......\

```

Esto también se muestra con un analizador de paquetes (en otro paquete, los datos son diferentes de las capturas de pantalla anteriores, pero la prueba y el error son idénticos),

Solución

La solución alternativa es inhabilitar el [buffer-up](#) en la interfaz donde tenemos este servidor conectado.

```

C9396PX-1(config)# int et 1/7
C9396PX-1(config-if)# no buffer-boost

```

Defecto relacionado:

[CSCva46849](#) Trama de 60 bytes con conmutación de encabezado L2 dot1q en N9k