包含L3和L4信息的Nexus 7K上PBR中的ACL行为

目录

简介

本文档介绍根据第3层(L3)和第4层(L4)信息过滤时Nexus交换机上基于策略的路由(PBR)的行为。

背景信息

如果在PBR中添加序列以匹配特定L4信息，因为功能N7K会为访问控制条目(ACE)创建条目，并自动创建与匹配序列中指定的L3信息匹配的分段ACE。对于分段数据包，第一个称为初始分段的数据包包含L4报头，并在访问控制列表(ACL)中正确匹配。 但是，下一个称为非初始分段的分段不包含任何L4信息，因此，如果ACL条目的L3部分匹配，则允许非初始分段。因此，在根据L4信息过滤流量时，应当格外小心，因为没有L4信息时，非初始分段可能被错误路由。

拓扑

LAN路由器连接到接口E2.1、Vlan 700上的Nexus。要求将匹配简单网络管理协议(SNMP)、Web等的流量直接重定向到Optimizer和所有其他流量，以便将E2/2接口连接到防火墙。PBR在Nexus设备上的交换机虚拟接口(SVI)Vlan700上配置。此处提供了相同的配置。路由映射中的序列70将所有其他流量转发到防火墙。有一项新要求，UDP端口为920x的所有流量都需要通过优化程序，因为此序列50会添加到路由映射中。

请参阅此处，PBR如何响应按序列50命中且匹配L3和L4信息的分段和非分段数据包。

以下是Nexus接口Vlan700上用于重定向E2/1上的流量的配置：

```
interface Vlan700

 no shutdown

 mtu 9000
```

```
 vrf member ABC
 no ip redirects
 ip address 10.11.25.25/28
 ip policy route-map In_to_Out


Nexus# show route-map In_to_Out
route-map In_to_Out, permit, sequence 3
 Match clauses:
   ip address (access-lists): Toolbar
 Set clauses:
   ip next-hop 10.3.22.13
route-map In_to_Out, permit, sequence 5
 Match clauses:
   ip address (access-lists): Internet
 Set clauses:
   ip next-hop 10.11.25.19
route-map In_to_Out, permit, sequence 7
 Match clauses:
   ip address (access-lists): Web
 Set clauses:
   ip next-hop 10.11.25.19
route-map In_to_Out, permit, sequence 10
 Match clauses:
   ip address (access-lists): In_to_Out_Internet
 Set clauses:
   ip next-hop 10.11.25.23
route-map In_to_Out, permit, sequence 30
 Match clauses:
   ip address (access-lists): In_to_Out_www
 Set clauses:
   ip next-hop 10.11.25.23
route-map In_to_Out, permit, sequence 35
```

```
 Match clauses:

   ip address (access-lists): In_to_Out_https

 Set clauses:

   ip next-hop 10.11.25.23

route-map In_to_Out, permit, sequence 40

 Match clauses:

   ip address (access-lists): In_to_Out_8080

 Set clauses:

   ip next-hop 10.11.25.23

route-map In_to_Out, permit, sequence 50

 Match clauses:

   ip address (access-lists): UDP_Traffic

 Set clauses:

   ip next-hop 10.11.25.23 >>>>>>>>>>>>>>>>>>>> Towards Optimizer

route-map In_to_Out, permit, sequence 70

 Match clauses:

   ip address (access-lists): To_Firewall

 Set clauses:

    ip next-hop . 10.22.45.63 >>>>>>>>>>>>>>>> Towards Firewall



Nexus# show ip access-lists UDP_Traffic

IP access list UDP_Traffic

10 permit udp any any eq 9201
20 permit udp any any eq 9202

30 permit udp any any eq 9203



Nexus# sh ip access-lists To_Firewall

IP access list To_Firewall

       10 permit ip any any
```

在SVI上配置基于策略的路由后，Nexus会在硬件中为其创建一个条目。现在，我们来了解一下Nexus模块2上PBR的硬件编程：

```
Nexus# show system internal access-list vlan 700 input entries detail module 2

Flags: F - Fragment entry  E - Port Expansion
```

```
      D - DSCP Expansion  M - ACL Expansion

      T - Cross Feature Merge Expansion

INSTANCE 0x0

---------------

 Tcam 1 resource usage:

 ----------------------

 Label_b = 0x201

  Bank 0

  ------

    IPv4 Class

      Policies: PBR(GGSN_Toolbar)

      Netflow profile: 0

      Netflow deny profile: 0

      Entries:

        [Index] Entry [Stats]

        ---------------------

 [0019:000f:000f] prec 1 permit-routed ip 0.0.0.0/0 224.0.0.0/4   [0]

 [002d:0024:0024] prec 1 redirect(0x5d)-routed tcp 1.1.22.80/28 0.0.0.0/0 eq 80 flow-label 80
[0]

 [002e:0025:0025] prec 1 redirect(0x5d)-routed tcp 1.1.22.80/28 0.0.0.0/0 fragment   [0]

 [002f:0026:0026] prec 1 redirect(0x5d)-routed tcp 1.1.22.80/28 0.0.0.0/0 eq 8080 flow-label
8080  [0]

 [0030:0027:0027] prec 1 redirect(0x5d)-routed tcp 1.1.22.80/28 0.0.0.0/0 fragment   [0]

 [0031:0028:0028] prec 1 redirect(0x5d)-routed tcp 1.1.22.48/28 0.0.0.0/0 eq 80 flow-label 80
[0]

 [0032:0029:0029] prec 1 redirect(0x5d)-routed tcp 1.1.22.48/28 0.0.0.0/0 fragment   [0]

 [0033:002a:002a] prec 1 redirect(0x5d)-routed tcp 1.1.22.48/28 0.0.0.0/0 eq 8080 flow-label
8080  [0]

 [0034:002b:002b] prec 1 redirect(0x5d)-routed tcp 1.1.22.48/28 0.0.0.0/0 fragment   [0]

 [0035:002c:002c] prec 1 permit-routed ip 1.1.22.24/29 0.0.0.0/0   [0]

 [0036:002d:002d] prec 1 permit-routed ip 1.1.22.32/28 0.0.0.0/0   [0]

 [0037:002e:002e] prec 1 permit-routed ip 1.1.22.64/28 0.0.0.0/0   [0]

 [0038:002f:002f] prec 1 permit-routed ip 1.1.22.80/28 0.0.0.0/0   [0]

 [003d:0033:0033] prec 1 permit-routed ip 1.1.22.96/28 0.0.0.0/0   [0]
```

```
 [003e:0034:0034] prec 1 permit-routed tcp 0.0.0.0/0 196.11.146.149/32 eq 25 flow-label 25  [0]

 [0059:004f:004f] prec 1 permit-routed tcp 0.0.0.0/0 196.11.146.149/32 fragment   [0]

 [005a:0050:0050] prec 1 redirect(0x5e)-routed ip 1.1.22.16/29 0.0.0.0/0   [0]

 [005b:0051:0051] prec 1 redirect(0x5e)-routed tcp 0.0.0.0/0 0.0.0.0/0 eq 80 flow-label 80  [0]

 [005c:0052:0052] prec 1 redirect(0x5e)-routed tcp 0.0.0.0/0 0.0.0.0/0 fragment   [0]

 [005d:0053:0053] prec 1 redirect(0x5e)-routed tcp 0.0.0.0/0 0.0.0.0/0 eq 443 flow-label 443
[0]

 [005e:0054:0054] prec 1 redirect(0x5e)-routed tcp 0.0.0.0/0 0.0.0.0/0 fragment   [0]

 [005f:0055:0055] prec 1 redirect(0x5e)-routed tcp 0.0.0.0/0 0.0.0.0/0 eq 8080 flow-label 8080
[0]

 [0060:0056:0056] prec 1 redirect(0x5e)-routed tcp 0.0.0.0/0 0.0.0.0/0 fragment   [0]

********************Sequence 50 is to match the traffic for UDP ports
9201/9202/9203**********************

 [0061:0057:0057] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9201 flow-label 9201
[0]

 [0062:0058:0058] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment   [0]

 [0063:0059:0059] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9202 flow-label 9202
[0]

 [0064:005a:005a] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment   [0]

 [0065:005b:005b] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9203 flow-label 9203
[0]

 [0066:005c:005c] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment   [0]

********************Sequence 70 is to send all other traffic to Firewall**********************

 [0067:005d:005d] prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [23]

 [0068:005e:005e] prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [0]
```

除了匹配**udp 0.0.0.0/0 0.0.0.0/0 eq 9201的访问列表条目外，还有**另一个条目与**udp 0.0.0.0/0 0.0.0.0/0分段**匹配，但该条目没有任何UDP端口信息。此条目等同于与UDP数据包匹配的任何其他条目，因此其他UDP端口的数据包也会按照硬件生成的顺序进行匹配。

测试案例1:从LAN路由器发起到防火墙的流量

- 到达Nexus的数据包未分段，因此PBR中的流量匹配预期。
- 它已正确重定向到防火墙，在防火墙上运行的调试中可以看到。

UDP packet -port 500

```
        *Mar 26 04:07:48.959: IP: s=1.1.1.1 (GigabitEthernet0/0), d=3.3.3.3, len 28, rcvd 4  -à
```
Traffic entering from Nexus interface

```
*Mar 26 04:07:48.959:     UDP src=500, dst=500
```

TCP packet - port 80

```
*Mar 26 04:07:48.671: IP: s=1.1.1.1 (GigabitEthernet0/1), d=3.3.3.3, len 40, rcvd 4
-à  Traffic entering from Optimizer interface

*Mar 26 04:07:48.671:     TCP src=1720, dst=80, seq=0, ack=0, win=0
```

UDP packet -port 9201

```
*Mar 27 09:30:19.879: IP: s=1.1.1.1 (GigabitEthernet0/1), d=3.3.3.3, len 28, input
feature  à Traffic entering from Optimizer interface

*Mar 27 09:30:19.879:     UDP src=6000, dst=9201, MCI Check(80), rtype 0, forus FALSE,
sendself FALSE, mtu 0, fwdchk FALSE
```

测试案例2:通过嗅探器文件从LAN路由器发起的流量通过UDP 500防火墙

在嗅探器文件中生成了两个分段的流量：

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	18:40:45.015197	1.1.1.1	3.3.3.3	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=0, ID=061e)
2	18:40:45.015288	1.1.1.1	3.3.3.3	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=1480, ID=061e)

1. 路由映射的初始分段：

- 偏移= 0的**第一个**分段称为初始分段，它包含数据包中的UDP报头。
- 由于流量用于UDP 500，因此它按序列70匹配，以允许**ip any any any。**

```
prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [23]
```

- 因此，第一个同时包含第3层和第4层信息的数据包会被正确路由。

2. 具有路由映射的非初始分段数据包：

- Offset ≠ 0的**第二个**分段称为非初始分段，不包含任何UDP报头。它是纯IP数据包，协议类型为UDP(17)。
- 由于没有第4层信息，因此它按顺序70匹配：**permit-routed ip 0.0.0.0/0 0.0.0.0/0。**
- 但是，在序列50中，存在与UDP端口920x的流量匹配的访问列表。硬件会自动创建一个条目，以允许与指定的第3层信息匹配的UDP分段。
- 因此，每个分片的数据包都包含序列50中匹配的UDP协议，以获取任何第3层信息。

```
prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9201 flow-label 9201  [0]

prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment [23]
>>>>>>>>>>>>>>>>>>>>
```

- 这样，有一个分段被正确路由，另一个分段通过错误的顺序路由。

- 修改第二个片段，使偏移= 0，并按照预期在序列70中匹配。
- 每当收到第4层分片时，这是预期行为。
- 创建允许分段的额外条目的目的是允许在没有第4层信息的情况下接收的非初始分段。
- 在这种情况下，流量用于UDP 9201，并且没有允许分段的条目。然后，第二个分段将在序列70中匹配，以允**许ip any any** any，因此路由错误。

```
Nexus# sh route-map In_to_Out pbr-statistics

route-map In_to_Out, permit, sequence 3

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 5

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 7

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 10

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 30

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 35

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 40

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 50 ----------> 2nd Fragment for UDP 500 is matched here

 Policy routing matches: 4397 packets

route-map In_to_Out, permit, sequence 70------------> 1st Fragment for UDP 500 is matched here

 Policy routing matches: 4397 packets
```

- 创建另一个序列45，以允许UDP 500的流量，并观察两个分段在序列45中匹配。
- 由于UDP报头信息和序列45的分段行中的非初始匹配，初始分段匹配。

```
Nexus# sh route-map In_to_Out pbr-statistics

route-map In_to_Out, permit, sequence 3

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 5
```

```
 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 7

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 10

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 30

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 35

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 40

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 45----------> Both fragments matched here

 Policy routing matches: 213 packets

route-map In_to_Out, permit, sequence 50

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 70

 Policy routing matches: 0 packets


Default routing: 0 packets
```

序列45的访问列表：

```
Nexus# sh ip access-lists udptraffic

IP access list udptraffic

permit udp any any eq isakmp
```

3. 现在，让我们看看fragments关键字如何与ACL和路由映射一起运行

- 应用序列5以允许端口ACL上的任意随机UDP端口56。

```
          Nexus# sh ip access-lists TEST_UDP


       IP access list TEST_UDP

       statistics per-entry

        5 permit udp any any eq 56 [match=0]
```

```
        10 permit udp any any eq isakmp [match=0]

        20 permit ip any any [match=0]
```

- 使用分段的非初始数据包启动流量流，并观察到它按顺序5匹配。即使数据包用于UDP 500，它也按顺序5匹配，以允许UDP 56。

```
Nexus# sh ip access-lists TEST_UDP



IP access list TEST_UDP

       statistics per-entry

        5 permit udp any any eq 56 [match=56]

        10 permit udp any any eq isakmp [match=0]

        20 permit ip any any [match=0]
```

- 端口ACL上拒绝分段，并且观察到ACL中没有非初始数据包匹配，因为数据包实际在**udp any分段中由平台自**动创建。

```
NEXUS# sh ip access-lists TEST_UDP



IP access list TEST_UDP

       statistics per-entry

        fragments deny-all

       5 permit udp any any eq 56 [match=0]

        10 permit udp any any eq isakmp [match=0]

        20 permit ip any any [match=0]



 [0014:000a:000a] prec 3 permit udp 0.0.0.0/0 0.0.0.0/0 eq 56 flow-label 56  [0]-> Here we are
now not seeing any entry to allow UDP fragments

 [0015:000b:000b] prec 3 permit udp 0.0.0.0/0 0.0.0.0/0 eq 500 flow-label 500  [0]

 [0016:000c:000c] prec 3 permit ip 0.0.0.0/0 0.0.0.0/0   [0]

 [0017:000d:000d] prec 3 deny ip 0.0.0.0/0 0.0.0.0/0 fragment   [100]>> Getting matched in
fragments deny statement

 [001e:0014:0014] prec 3 deny ip 0.0.0.0/0 0.0.0.0/0   [0]
```

- 在PBR中拒绝有问题的ACL中的分段，但此解决方法不起作用，并且仍然可以看到序列50和70中的数据包匹配。这是由于访问列表和路由映射的编程行为。

```
                NEXUS# sh ip access-lists UDP_Traffic

       IP access list UDP_Traffic
```

```
       statistics per-entry

        fragments deny-all

       10 permit udp any any eq 9201

        20 permit udp any any eq 9202

        30 permit udp any any eq 9203


[0061:0057:0057] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9201 flow-label 9201
[0]

[0062:0058:0058] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment    [8027]

 [0063:0059:0059] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9202 flow-label 9202
[0]

 [0064:005a:005a] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment    [0]

 [0065:005b:005b] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9203 flow-label 9203
[0]

 [0066:005c:005c] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment    [0]

  [0067:005d:005d] prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [8027]

 [0068:005e:005e] prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [0]
```

- 在端口ACL和PBR ACL上应用分段拒绝时输出：

```
[0061:0057:0057] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9201 flow-label 9201
[0]

[0062:0058:0058] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment    [8027] ---
> Once the fragments are denied in port CAL, we observed non-initial packets to be getting
dropped (See the mismatch in number of packets between UDP and IP counter)

 [0063:0059:0059] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9202 flow-label 9202
[0]

 [0064:005a:005a] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment    [0]

 [0065:005b:005b] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 eq 9203 flow-label 9203
[0]

 [0066:005c:005c] prec 1 redirect(0x5e)-routed udp 0.0.0.0/0 0.0.0.0/0 fragment    [0]

  [0067:005d:005d] prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [8214]

 [0068:005e:005e] prec 1 permit-routed ip 0.0.0.0/0 0.0.0.0/0   [0]


VDC-1 Ethernet2/1 :

 ====================
```

```
INSTANCE 0x0

---------------

 Tcam 0 resource usage:

 ----------------------

 Label_a = 0x200

  Bank 0

  ------

    IPv4 Class

      Policies: PACL(TEST_UDP)

       Netflow profile: 0

      Netflow deny profile: 0

      Entries:

         [Index] Entry [Stats]

        ---------------------

 [0014:000a:000a] prec 3 permit udp 0.0.0.0/0 0.0.0.0/0 eq 56 flow-label 56  [8027]

 [0015:000b:000b] prec 3 permit udp 0.0.0.0/0 0.0.0.0/0 eq 500 flow-label 500  [8214]

 [0016:000c:000c] prec 3 permit ip 0.0.0.0/0 0.0.0.0/0   [0]

 [0017:000d:000d] prec 3 deny ip 0.0.0.0/0 0.0.0.0/0 fragment   [100]

 [001e:0014:0014] prec 3 deny ip 0.0.0.0/0 0.0.0.0/0   [0]
```

有几种方法可以克服此问题或限制使用L4信息的分段数据包：

- 可以调整路由映射以允许特定UDP端口的特定L3信息。

在当前配置中，如果提到L3源和目标信息，则根据该特定信息路由非初始数据包。但是，仅当序列与相同的L3信息匹配之前没有其他序列时，这才有用。

```
Nexus# show ip access-lists UDP_Traffic

IP access list UDP_Traffic

10 permit udp host 1.1.1.1 host 3.3.3.3 eq 9201
20 permit udp any any eq 9202

30 permit udp any any eq 9203
```

- 可以验证从源到目的地的路径，以检查MTU，这样数据包就不会被分段。
- 应用另一个序列的解决方法允许问题序列上方的UDP起作用，但是，该行为与应用序列45时前面解释的行为相同

```
Nexus# sh route-map In_to_Out pbr-statistics

route-map In_to_Out, permit, sequence 3

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 5

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 7

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 10

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 30

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 35

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 40

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 45----------> Both fragments matched here

 Policy routing matches: 213 packets

route-map In_to_Out, permit, sequence 50

 Policy routing matches: 0 packets

route-map In_to_Out, permit, sequence 70

 Policy routing matches: 0 packets
```

序列45的访问列表：

```
Nexus# sh ip access-lists udptraffic
```

IP访问列表udptraffic:

```
permit udp any any eq isakmp
```

文档错误：CSCve05428 N7K文档错误 || PBR中同时包含L3和L4信息的ACL。