Comportement de la liste de contrôle d'accès dans PBR sur Nexus 7K contenant les informations de couche 3 et de couche 4

Contenu

Introduction

Ce document décrit le comportement du routage basé sur des politiques (PBR) sur les commutateurs Nexus lorsque vous filtrez en fonction des informations de couche 3 (L3) et de couche 4 (L4).

Informations générales

Si vous ajoutez une séquence dans PBR afin de correspondre à des informations de couche 4 spécifiques, en tant que fonctionnalité N7K crée des entrées pour les entrées de contrôle d'accès (ACE) et un ACE de fragment est créé automatiquement qui correspond aux informations de couche 3 spécifiées dans la séquence de correspondance. En cas de paquets fragmentés, le premier paquet appelé fragment initial contient l'en-tête L4 et est correctement mis en correspondance dans la liste de contrôle d'accès (ACL). Cependant, les fragments suivants appelés fragments non initiaux ne contiennent aucune information L4 et, par conséquent, si la partie L3 de l'entrée ACL correspond, le fragment non initial est autorisé. Il faut donc faire preuve de la plus grande prudence tout en filtrant le trafic en fonction des informations de couche 4, car les fragments non initiaux peuvent être mal acheminés en l'absence d'informations de couche 4.

Topologie

Le routeur LAN est connecté à Nexus sur l'interface E2.1, Vlan 700. La condition est de rediriger le trafic qui correspond au protocole SNMP (Simple Network Management Protocol), au Web, etc. vers Optimizer et tout autre trafic directement afin d'interface E2/2 vers Firewall. PBR est configuré sur le Vlan700 de l'interface virtuelle de commutateur (SVI) sur le périphérique Nexus. La configuration de la même configuration est fournie ici. La séquence 70 de la route-map transfère

tout autre trafic vers le pare-feu. Il est nécessaire que tout le trafic avec le port UDP 920x passe par Optimizer, car cette séquence 50 est ajoutée dans la route-map.

Voyez ici comment le PBR répond aux paquets fragmentés et non fragmentés qui frappent dans la séquence 50 et correspondent aux informations de couche 3 et de couche 4.

Voici la configuration de l'interface Nexus Vlan700 pour rediriger le trafic qui arrive sur E2/1 :

```
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
```

Une fois que le routage basé sur la stratégie est configuré sur SVI, Nexus crée une entrée dans le matériel pour le même. Examinons maintenant la programmation matérielle du PBR sur le module 2 de Nexus :

```
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]
```

Vous voyez qu'en plus de l'entrée de liste d'accès qui correspond au **fragment udp 0.0.0.0/0**

0.0.0.0/0 eq 9201, il y a une autre entrée qui correspond aux fragments **udp 0.0.0.0/0 0.0.0.0/0** mais cette entrée n'a aucune information de port UDP. Cette entrée est équivalente à toute autre qui correspond au paquet UDP, de sorte que les paquets pour d'autres ports UDP sont également appariés dans cette séquence générée par le matériel.

Cas d'essai 1 : Trafic initié du routeur LAN vers le pare-feu

- Le paquet qui atteint le Nexus n'était pas fragmenté et le trafic correspondait donc comme prévu dans le PBR.
- Il a été correctement redirigé vers le pare-feu et peut être vu dans les débogages exécutés sur le pare-feu.

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

Cas d'essai 2 : Trafic initié via le fichier de renifleur du routeur LAN vers le pare-feu avec UDP 500

Trafic avec deux fragments dans le fichier Sniffer généré ici :

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. Fragments initiaux avec Route-Map :

- Le premier fragment avec **Décalage = 0** est appelé fragment initial et contient l'en-tête UDP dans le paquet.

- Comme le trafic est pour UDP 500, il est mis en correspondance dans la séquence 70 pour permettre **ip any any.**

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

- Ainsi, le tout premier paquet contenant des informations de couche 3 et de couche 4 est correctement routé.

2. Paquets de fragments non initiaux avec Route-Map :

- Le deuxième fragment avec **Offset→ 0** est appelé fragment non initial et ne contient aucun en-tête UDP. Il s'agit d'un paquet IP avec le type de protocole UDP (17).
- Comme il n'y a aucune information de couche 4, elle correspond à la séquence 70 : **permit-routed ip 0.0.0.0/0 0.0.0.0/0.**
- Cependant, dans la séquence 50, il y a une liste d'accès qui correspond au trafic pour le port UDP 920x. Le matériel crée automatiquement une entrée pour autoriser les fragments UDP qui correspondent aux informations de couche 3 spécifiées.
- Par conséquent, chaque paquet fragmenté pour toute information de couche 3 avec le protocole UDP qui est mis en correspondance dans la séquence 50.

```
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]
>>>>>>>>>>>>>>>>>>>
```

- De cette façon, il y a un fragment qui est routé correctement et un autre qui est routé par une séquence incorrecte.
- Le deuxième fragment est modifié afin de faire **offset = 0**, et il est apparié dans la séquence 70 comme prévu.
- Il s'agit d'un comportement attendu chaque fois que les fragments de couche 4 sont reçus.
- L'intention de créer une entrée supplémentaire pour autoriser les fragments est d'autoriser les fragments non initiaux reçus sans les informations de couche 4.
- Dans le cas contraire, le trafic était pour UDP 9201 et il n'y avait aucune entrée pour autoriser les fragments. Ensuite, le deuxième fragment aurait apparié dans la séquence 70 pour permettre **ip any any** et donc être routé à tort.

```
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
```

- Une autre séquence 45 est créée afin de permettre le trafic pour UDP 500 et d'observer que les deux fragments sont appariés dans la séquence 45.
- Le fragment initial correspondait en raison des informations d'en-tête UDP et non initial correspondait dans la ligne de fragments pour la séquence 45.

```
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
```

Liste d'accès pour la séquence 45 :

```
Nexus# sh ip access-lists udptraffic

IP access list udptraffic

permit udp any any eq isakmp
```

3. Voyons maintenant comment se comporte le mot clé fragments avec ACL et Route-Map

- La séquence 5 est appliquée pour autoriser tout port UDP aléatoire 56 sur la liste de contrôle d'accès du port.

```
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]
```

- Initié un flux de trafic avec un paquet non initial fragmenté et observé qu'il correspondait dans la séquence 5. Même si le paquet est pour UDP 500, il correspond dans la séquence 5 afin d'autoriser 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]
```

- Les fragments sont refusés sur la liste de contrôle d'accès du port et il est observé qu'aucun paquet n'est mis en correspondance dans la liste de contrôle d'accès pour les paquets non initiaux, car le paquet est effectivement mis en correspondance dans l'entrée **udp tout fragment** créé automatiquement par la plate-forme.

```
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]
```

- Refusé les fragments dans la liste de contrôle d'accès problématique dans PBR, cependant cette solution de contournement n'a pas fonctionné et les paquets sont toujours considérés comme correspondant dans les séquences 50 et 70. Ceci est dû au comportement de programmation de la liste d'accès et de la route-map.

```
          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]
```

- Sorties lorsque des fragments refusés sont appliqués à la fois sur la liste de contrôle d'accès des ports et sur la liste de contrôle d'accès PBR :

```
[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]
```

Il existe plusieurs façons de surmonter ce problème ou cette limitation de paquets fragmentés avec des informations de couche 4 :

- La route-map peut être modifiée afin de permettre des informations de couche 3 spécifiques pour des ports UDP particuliers.

Dans la configuration actuelle, si les informations source et de destination de couche 3 sont mentionnées, le paquet non initial est routé en fonction de ces informations spécifiques. Cependant, ceci n'est utile que s'il n'y a aucune autre séquence avant qu'elle ne corresponde aux mêmes informations L3.

```
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
```

- Le chemin de la source à la destination peut être vérifié afin de vérifier le MTU de sorte que le paquet ne soit pas fragmenté.
- La solution de contournement de l'application d'une autre séquence permet au protocole UDP au-dessus de la séquence problématique de fonctionner, mais le comportement est identique à celui expliqué précédemment lorsque la séquence 45 a été appliquée

```
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
```

Liste d'accès pour la séquence 45 :

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

udptraffic de la liste d'accès IP :

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

Bogue du document : CSCve05428 N7K bogue Doc || ACL dans PBR qui contient les informations de couche 3 et de couche 4.