

Есть ли альтернатива BGP Flowspec?

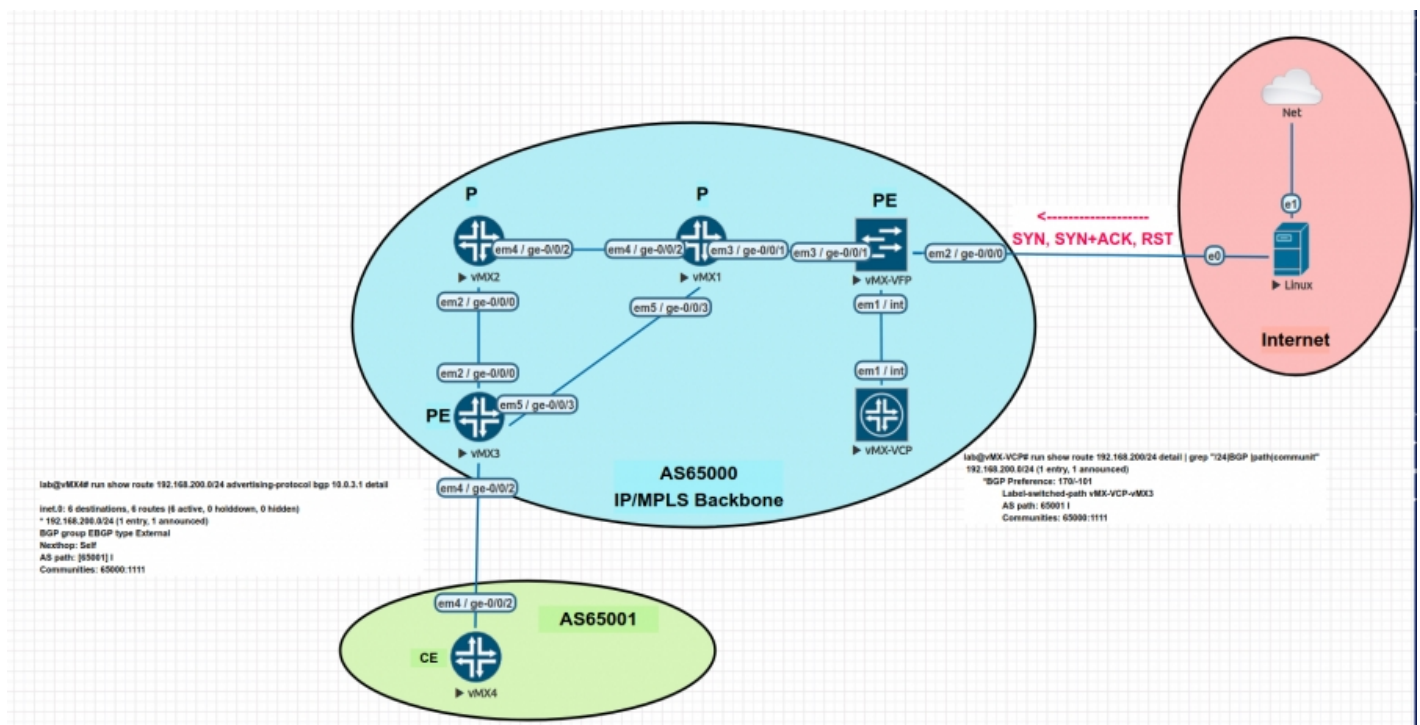
В отличие от [предыдущих публикаций](#) статья написана в соавторстве с коллегой. Выражаем искренние признания компании Ростелеком, в который мы получили бесценные знания и опыт, позволившие нам стать настоящими сетевыми инженерами компании, в которой мы работаем в настоящее время, и получили реальную возможность реализовать свой потенциал в одном из самых интересных направлений информационной безопасности - защите от DDoS. Отдельные слова благодарности адресуем portalу NAG.RU, ее главному редактору, любезно предоставившим возможность делиться полученным опытом с коллегами по отрасли.

В одной из предыдущих публикаций [описывалась](#) реализация геофильтрации трафика на оборудовании Juniper с использованием технологии Source Class Usage (SCU), как одном из методов применения prefirewall фильтров в сетевых решениях защиты от DDoS. Необходимость применения подобных фильтров, в независимости от схемного решения (централизованная или распределенные схемы фильтрации), обусловлена, прежде всего, большими волюметриками современных атак, отсутствием возможности (целесообразности) транзита всего этого "грязного" трафика в сетях Операторов, дороговизной решений по его очистке в огромных объемах.

Одним из самых распространенных видов prefirewall-фильтров является BGP Flowspec, широко применяемый как в рамках одного домена, так и на междоменном уровне, и позволяющий с помощью сигнализации протокола BGP (AFI flow) передавать описанные правила на удаленные узлы в сети, где они выполняются Firewall. Более подробно описано в RFC5575, с дополнениями в RFC7674, и сопутствующим количеством документов со статусом Internet-Draft.

Невзирая на заголовок статьи, изложенный в ней материал - не попытка преподнести полноценную замену многолетнему стандарту, давно нашедшему реализацию во многих вендорских и операторских решениях. У каждой технологии есть свои особенности, преимущества и недостатки. Речь пойдет о совместном применении всем хорошо известного транзитивного атрибута community протокола BGP, вендорских решениях от Juniper Destination Class Usage и Prefix-Specific Action (у Cisco - microflow policing).

Рассмотрим более подробно совместную реализацию данных решений на приведенных схеме, примерах конфигурации, скриншотах тестирования.



Тестовая схема состоит из AS65001, представляющую автономную систему клиента, и анонсирующую защищаемый префикс 192.168.200/24 с community 65000:1111; из AS65000, представляющей IP/MPLS домен ISP; из сегмента Internet с хостом, генерирующим тестовый трафик. Основная конфигурация выполняется на пограничных маршрутизаторах ISP, в который включены аплинки (в нашем примере это PE-маршрутизатор vMX-VCP).

Рассмотрим подробнее на примере конфигурации.

1. Опишем community, по которому policy будет осуществлять match определенных префиксов:

```
set policy-options community dst-prefixes members 65000:1111
```

2. Опишем саму policy, в котором в качестве match указан протокол bgp и ранее описанное community, а в качестве action назначаться destination-class:

```
set policy-options policy-statement dcu-psa term 1 from protocol bgp
set policy-options policy-statement dcu-psa term 1 from community dst-prefixes
set policy-options policy-statement dcu-psa term 1 then destination-class psa
```

3. Создадим policer, который будет применяться в prefix-action для ограничения определенного трафика (ограничение в 2 Mbps указано в качестве примера):

```
set firewall policer lim2m-PSA if-exceeding bandwidth-limit 2m
set firewall policer lim2m-PSA if-exceeding burst-size-limit 15k
set firewall policer lim2m-PSA then discard
```

4. Создаем firewall prefix-action, в котором указываем длину префикса (в нашем случае это /24), его градацию по destination (в примере указано /32), а также policer, ограничивающий определенный трафик на каждый destination (т. е. на каждый IP-адрес):

```
set firewall family inet prefix-action psa-2m-dest-24-32 policer lim2m-PSA
set firewall family inet prefix-action psa-2m-dest-24-32 count
set firewall family inet prefix-action psa-2m-dest-24-32 filter-specific
set firewall family inet prefix-action psa-2m-dest-24-32 subnet-prefix-length 24
set firewall family inet prefix-action psa-2m-dest-24-32 destination-prefix-length 32
```

4. В firewall filter PSA ссылаемся на ранее описанный destination-class, описываем критерии, по которым необходимо матчить трафик (в примере TCP-пакеты с флагами syn, ack, syn+ack, rst, но в реальности может быть все что угодно, например, тот же UDP) и в качестве action указываем ранее созданный prefix-action:

```
set firewall filter PSA term psa from destination-class psa
set firewall filter PSA term psa from source-address 0.0.0.0/0
set firewall filter PSA term psa from protocol tcp
set firewall filter PSA term psa from tcp-flags "((syn & ack) | rst) | syn) | ack"
set firewall filter PSA term psa then next term
set firewall filter PSA term psa then prefix-action psa-2m-dest-24-32
set firewall filter PSA term accept then accept
```

5. Применяем созданный фильтр к forwarding-options:

```
set forwarding-options family inet filter output PSA
```

Ниже приведены данные тестирования.

1. Пример TCP-пакетами с флагом syn. С тестового сервера отправляется syn-flood на адрес 192.168.200.1:

```
[16:38:24] root@ubuntu-noc2:~# hping3 192.168.200.1 -S --flood
HPING 192.168.200.1 (ens3 192.168.200.1): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.200.1 hping statistic ---
4542121 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
[16:39:41] root@ubuntu-noc2:~#
```

Смотрим срабатывание счетчиков и полисера:

```
[edit]
lab@vMX-VCP# run show firewall prefix-action-stats prefix-action psa-2m-dest-24-32 filter PSA from 0 to 2
Dec 22 16:39:38

Filter: PSA
Counters:
Name                               Bytes          Packets
psa-2m-dest-24-32-0                0              0
psa-2m-dest-24-32-1              14969720       374243
psa-2m-dest-24-32-2                0              0
Policers:
Name                               Bytes          Packets
psa-2m-dest-24-32-0                0              0
psa-2m-dest-24-32-1              8847560       221189
psa-2m-dest-24-32-2                0              0

[edit]
lab@vMX-VCP#
```

2. Пример TCP-пакетами с флагом syn-ack. Флуд на IP 192.168.200.100:

```
[16:49:54] root@ubuntu-noc2:~# hping3 192.168.200.100 -S -A --flood
HPING 192.168.200.100 (ens3 192.168.200.100): SA set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.200.100 hping statistic ---
2130669 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
[16:53:50] root@ubuntu-noc2:~#
```

Ниже данные счетчиков и полисера:

```
lab@vMX-VCP# run show firewall prefix-action-stats prefix-action psa-2m-dest-24-32 filter PSA from 99 to 101
Dec 22 16:54:03

Filter: PSA
Counters:
Name                               Bytes          Packets
psa-2m-dest-24-32-99              0              0
psa-2m-dest-24-32-100            4668080       116702
psa-2m-dest-24-32-101            0              0
Policers:
Name                               Bytes          Packets
psa-2m-dest-24-32-99              0              0
psa-2m-dest-24-32-100            4790160       119754
psa-2m-dest-24-32-101            0              0

[edit]
lab@vMX-VCP#
```

3. Пример TCP-пакетами с флагом ack. Тестовый трафик отправляется на IP 192.168.200.10:

```
[16:39:41] root@ubuntu-noc2:~# hping3 192.168.200.10 -A --flood
HPING 192.168.200.10 (ens3 192.168.200.10): A set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.200.10 hping statistic ---
1408020 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
[16:46:25] root@ubuntu-noc2:~#
```

Данные счетчиков и полисера:

```
[edit]
lab@vMX-VCP# run show firewall prefix-action-stats prefix-action psa-2m-dest-24-32 filter PSA from 9 to 11
Dec 22 16:46:28

Filter: PSA
Counters:
Name                               Bytes          Packets
psa-2m-dest-24-32-9               0              0
psa-2m-dest-24-32-10            6890720       172268
psa-2m-dest-24-32-11            0              0
Policers:
Name                               Bytes          Packets
psa-2m-dest-24-32-9               0              0
psa-2m-dest-24-32-10            4756800       118920
psa-2m-dest-24-32-11            0              0

[edit]
lab@vMX-VCP#
```

В заключение, несколько комментариев.

Применение firewall-фильтров на основе классов дает дополнительные возможности реализации механизмов защиты от DDoS-атак (некоторые ISPs предоставляют в виде дополнительного сервиса,

такого как bgp extended blackhole community, например). Но все они имеют один существенный недостаток: при таргетированных на сервис атаках, последний однозначно пострадает, т. к., например, при syn-flood фильтром будет дропать, в том числе, и какая-то доля легитимных пакетов, что приведет к затруднениям в установке новых соединений. Этот недостаток характерен для всех prefirewall-фильтров, умеющих, в лучшем случае, полисить трафик. Идеальным является решение, когда есть возможность не использовать prefirewall, а "прогонять" все объемы трафика через Threat Management System, где его фильтрация осуществляется более интеллектуально. К сожалению, подобные решения не под силу многим ISPs, порой даже крупным.

Также при схемных реализациях (особенно с предоставленной клиентам возможностью управлять специфичными анонсами) с применением prefirewall, необходим постоянный контроль утилизации ресурсов сетевого оборудования, отсутствие которого может привести к плачевным последствиям.

Ниже приведены примеры контроля памяти (Lookup Block ASIC) линейной карты тестового маршрутизатора:

```
lab@vMX-VCP# run request pfe execute target fpc0 command "show jnh 0 pool usage"
SENT: Ukern command: show jnh 0 pool usage
EDMEM overall usage:
[NH////////\FW////////\CNTR////////\HASH////\ENCAPS////]-----]
0 4.0 8.0 14.0 17.2 21.3 32.0M
Next Hop
[*****]-----[RRRRRRRRRRRRRRRRRRRRR] 4.0M (28% | 72%)
Firewall
[|]-----[RRRRRRRRRRRRRRRRRRRRR] 4.0M (<1% | >99%)
Counters
[*****]-----[RRRRRRRRRRRRRRRRRRRRR] 6.0M (35% | 65%)
HASH
[*****] 3.2M (100% | 0%)
ENCAPS
[*****] 4.1M (100% | 0%)
Shared Memory - NH/FW/CNTR/HASH/ENCAPS
[-----] 10.7M (0% | 100%)
Free Shared Memory - NH/FW/CNTR/HASH/ENCAPS
[-----] 10.7M (0% | 100%)
```

Здесь все замечательно, FW использует менее 1% (плюс зарезервированный пул динамически распределяемой памяти, RLDRAM), в реальности же все немного иначе.

Использованные материалы:

- [Prefix-Specific Counting and Policing Actions.](#)
- [Understanding Source Class Usage and Destination Class Usage Options.](#)
- [Configuring SCU or DCU.](#)

Надеемся на продолжение публикаций, описывающих методы и способы обеспечения безопасности в Операторских сетях.