

Содержание

Beta-версия 14.2	3
<i>Изменения в версии 14.1 BETA1</i>	3

Beta-версия 14.2

Изменения в версии 14.1 BETA1

DPI

1. [dpdk] Переход на новую версию dpdk 25.11
2. [DPI][nat] Оптимизация при переполнении кэша серый-белый
3. [CLI][vlan] В команду `vlan rule dump` добавлен параметр - какой тип правил выводить

```
vlan rule dump [type]
    type - rule type: 'perm', 'dhcp', 'all' (default)
Вывести разрешения для VLAN:
vlan rule dump perm
Вывести правила только для DHCP:
vlan rule dump dhcp
Вывести все правила:
vlan rule dump
```

4. [CLI][DPI] Вывод команды `fdpi_cli dump flow cache format` расширен новыми полями

Формат вывода 'dump flow'

Пример :

```
nthr=1 slic=3 proto=6 ip_1=192.168.4.20:65163 ip_2=217.69.133.145:443
ssid=1675E5CF5FB1337 dpip=91 ittr=16 tmlb='2019/10/30 02:02:51,
-357.642147s (4148500652028035 ticks)' ialf=0 drct=0x1 iown=1 ilst=1
btsip=0x2 tcpbts_0='-APRSF' tcpbts_1='-AP-S-' qoest=0 qoef_0=0 qoef_1=0
qoer_0=6 qoer_1=6 whip=94.140.198.86:33326 itrnsld=1 igcache=0
gre_pid=0 gre_mtd=0
```

по полям :

```
nthr=1    - номер потока куда помещена запись ( для мультикластера может не
совпадать с iown )
slic=3    - номер slice кэша
proto=6   - ip протокол
ip_1=192.168.4.20:65163 ip_2=217.69.133.145:443 - пара ip адресов и
портов идентифицирующие запись. Если у протокола нет портов - последние 0
ssid=1675E5CF5FB1337 - идентификатор сессии
dpip=91    - протокол DPI
ittr=16    - индекс в очереди использования повторных записей
tmlb='2019/10/30 02:02:51, -357.642147s (4148500652028035 ticks)' -
время последнего обращения к записи
ialf=0     - номер очереди обработки :
```

```
en_nalfs_shrt = 0, //
```

очередь с коротким временем жизни

```
en_nalfs_long = 1, //
```

долгоиграющая очередь

```
drct=0x1          - при каких условиях создана запись :
                   младшие 4 бита задают :
                   направление пакета , при котором создан ключ :
                   и соответственно принадлежность src_ip и dst_ip
                   drct = h_ip_1 < h_ip_2 :
                   drct == 0 - h_ip_1 - src_ip
                   drct == 1 - h_ip_1 - dst_ip
                   старшие 4 бита задают flw_dir при котором был создан ключ

iown=1            - номер потока, который создал запись
ilst=1            - номер потока, который последний раз обрабатывал запись
btsip=0x2         - служебные биты обработки flow

tcpbts_0='-APRSF' tcpbts_1='-AP-S-' - биты TCP соединения в двух
направлениях :

                                     ( tcp_bits_ & 0x0020 )
? 'U' : '-'
                                     ( tcp_bits_ & 0x0010 )
? 'A' : '-'
                                     ( tcp_bits_ & 0x0008 )
? 'P' : '-'
                                     ( tcp_bits_ & 0x0004 )
? 'R' : '-'
                                     ( tcp_bits_ & 0x0002 )
? 'S' : '-'
                                     ( tcp_bits_ & 0x0001 )
? 'F' : '-'

qoest=0           - статус QoE :
                   enst_none = 0,    //
                   enst_ack,         //
ждем подтверждающий ACK от клиента на SYN+ACK от сервера
                   enst_fin_ack,     //
ждем подтверждающий FIN+ACK от сервера на FYN от клиента
                   enst_ack_srvfin,  //
ждем подтверждающий ACK от сервера на FIN+ACK от клиента ( сервер первый послал
FIN )

qoef_0=0 qoef_1=0 - кол-во фрагментированных пакетов в двух направлениях
qoer_0=6 qoer_1=6 - кол-во ретрансмитов в двух направлениях
pktp_0=1 pktp_1=0 - количество пакетов с payload в двух направлениях, но не
более 65000
btsp_0=1 btsp_1=0 - объем payload в двух направлениях, но не более 65K
whoisc=0 или 1    - кто инициировал соединение

Опционально - если еть NAT трансляция :
whip=94.140.198.86:33326 - выделенный белый адрес+порт
itrnsld=1          - индекс данных профиля по которому был выделен белый
адрес
```

igcache=0	- индекс в соответствующем кэше-slice перекодировки
серый --> белый	
gre_pid=0	- определенный callid
gre_mtd=0	- метод выделения белого адреса для GRE

5. [BALANCER] Добавлена возможность использования vlan rule для фильтрации пакетов

6. [dpdk] Add: новая опция `dpdk_max_memzone`

[cold] Установка DPDK max memzone count
 По умолчанию, в DPDK max memzone count = 5120 (зависит от версии DPDK)
 0 - использовать default-значение, зашитое в DPDK
 Устанавливать больше имеет смысл для huge-конфигураций со многими картами, если на старте fastdpi получаем ошибку "Number of requested memzone segments exceeds maximum 5120"
 #dpdk_max_memzone=0

7. [CLI][dhcp-dual] Add: поддержка команды `dhcp show stat vrf`

8. [dpdk] Новый engine dpdk_engine=7 с поддержкой явного указания диспетчеров

Данный движок поддерживает гетерогенные конфигурации, когда в одном кластере находятся порты разного типа - например, in-dev 100G порт, out-dev - несколько 10G портов.

Диспетчеры задаются в опциях dpdk_dispatch:

```

dpdk\_dispatch=<список-портов>[;params]\*  
 ```

`<Список-портов>` задает, какие порты обслуживает данный диспетчер, а `params` - дополнительные опции данного диспетчера. Доступные опции:
 - `rss=N` - включение RSS на всех портах данного диспетчера; будет создано N диспетчеров - по одному на каждую rx-очередь.

- `mempool_size=N` - размер mbuf_pool для данного диспетчера. Каждый `dpdk_dispatch` имеет свой mempool, размер пулов может быть разным для разных `dpdk_dispatch`.

Опций `dpdk_dispatch` может быть много, каждая описывает отдельный диспетчер (или группу диспетчеров, если задано rss). Каждый порт кластера должен входить ровно в один параметр `dpdk_dispatch`.

Для on-stick в `dpdk_dispatch` описывается *базовый* физический порт, а не on-stick порты, основанные на нем.

Фатальными ошибками конфигурации считаются следующие случаи:

- порт кластера не входит ни в какой `<список-портов>` параметра `dpdk_dispatch`
- порт кластера входит в `<список-портов>` нескольких различных параметров `dpdk_dispatch`
- в `<списке-портов>` параметра `dpdk_dispatch` перечислены порты из разных кластеров, - каждый диспетчер должен обслуживать порты только одного кластера

Данный движок является универсальным в том смысле, что через него могут быть выражены все

остальные движки, например:

```

dpdk\_engine=0: один диспетчер на все порты

```
in_dev=port1:port2
out_dev=port3:port4
dpdk_dispatch=port1,port2,port3,port4
```

dpdk\_engine=1: диспетчер на направление

```
in_dev=port1:port2
out_dev=port3:port4
dpdk_dispatch=port1,port2
dpdk_dispatch=port3,port4
```

dpdk\_dispatch=2: диспетчер на направление с поддержкой RSS

```
dpdk_rss=4
in_dev=port1:port2
out_dev=port3:port4
dpdk_dispatch=port1,port2;rss=4
dpdk_dispatch=port3,port4;rss=4
```

dpdk\_engine=3: диспетчер на мост

```
in_dev=port1:port2
out_dev=port3:port4
dpdk_dispatch=port1,port3
dpdk_dispatch=port2,port4
```

dpdk\_engine=4: диспетчер на порт

```
in_dev=port1:port2
out_dev=port3:port4
dpdk_dispatch=port1
dpdk_dispatch=port2
dpdk_dispatch=port3
dpdk_dispatch=port4
```

dpdk\_engine=6: диспетчер на мост с поддержкой RSS

```
dpdk_rss=4
in_dev=port1:port2
out_dev=port3:port4
dpdk_dispatch=port1,port3;rss=4
dpdk_dispatch=port2,port4;rss=4
```

Для диспетчера необходимо указывать используемый mempool.

// Описатель mempool - только для dpdk\_engine=7

// Формат:

```
// dpdk_mempool=name=<name>;size=N
// name задает имя mempool (max 15 символов)
// size задает размер (число элементов) mempool
```

// Оба параметра являются обязательными

// Опций dpdk\_mempool может быть много, каждая описывает отдельный mempool

dpdk\_mempool=...

// Описатель диспетчера - только для dpdk\_engine=7

// Формат:

```
// dpdk_dispatch=<список-портов>;mempool=<имя>[;params]*
```

```
// <Список-портов> задает, какие порты обслуживает данный диспетчер, а
params - дополнительные
// опции данного диспетчера. Доступные опции:
// mempool=<имя> - задает имя mempool для данного диспетчера
(обязательный параметр)
// rss=N - включение RSS на всех портах данного диспетчера;
будет создано N диспетчеров -
// по одному на каждую rx-очередь.
// Опций dpdk_dispatch может быть много, каждая описывает отдельный
диспетчер (или группу
// диспетчеров, если задано rss)
dpdk_dispatch=...
```

9. [IPFIX] Исправлена ошибка при изменении опции ipfix\_dev