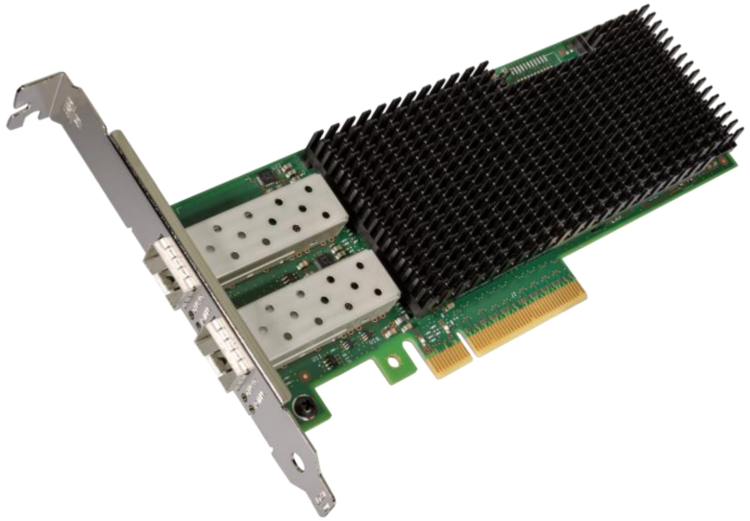
ADVANCED AVF RSS & FDIR

Yahui Cao

Agenda

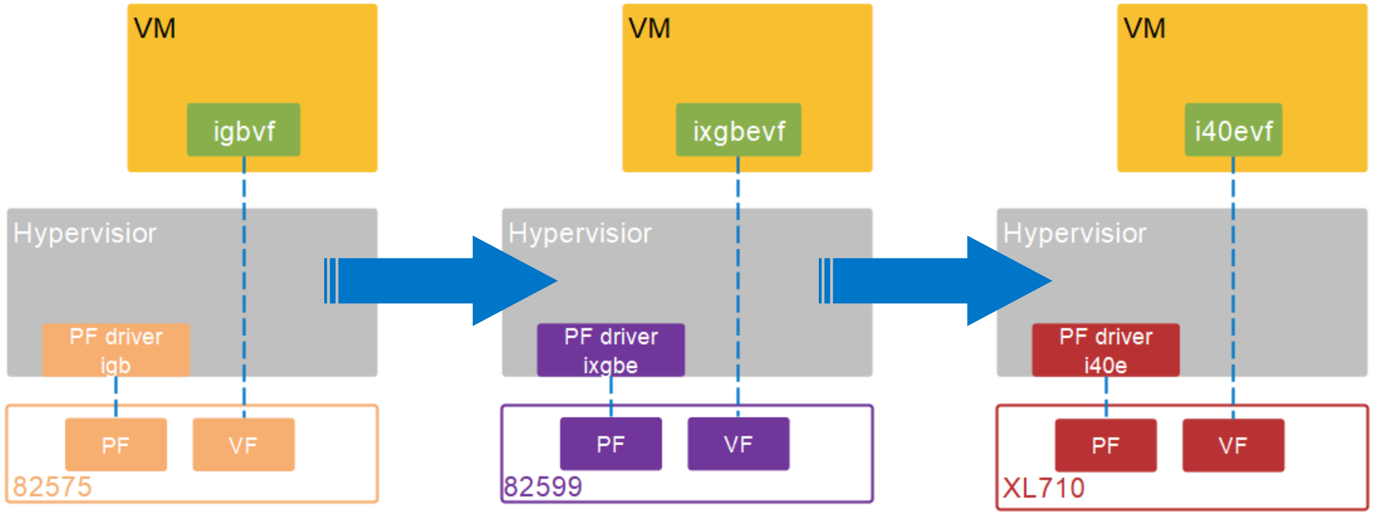


- AVF Introduction
- RSS Introduction
- FDIR Introduction
- Virtual Channel Introduction

AVF Intro

- ➢ Usually different SRIOV Virtual Function has different VF driver

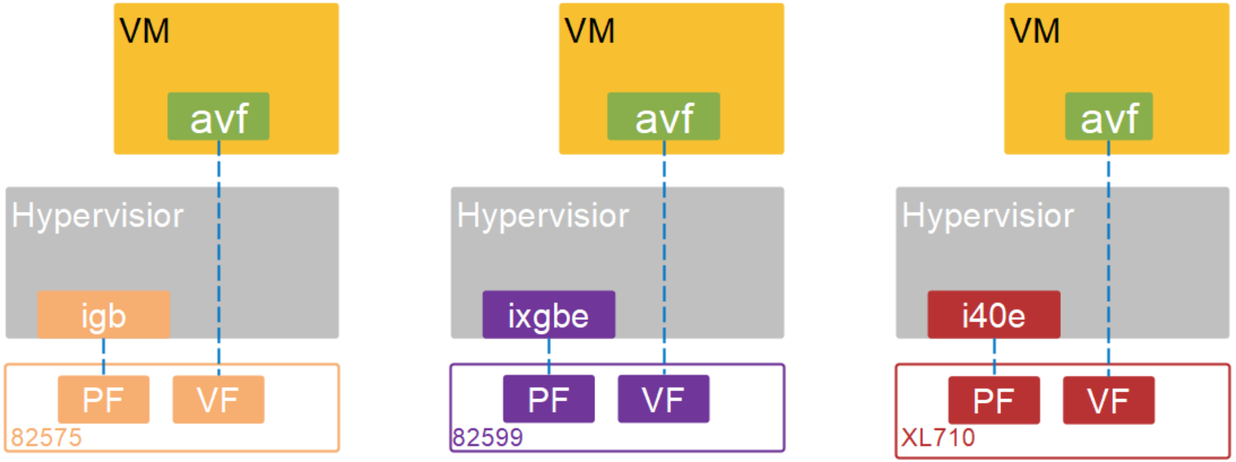
HW Upgrade



AVF Intro

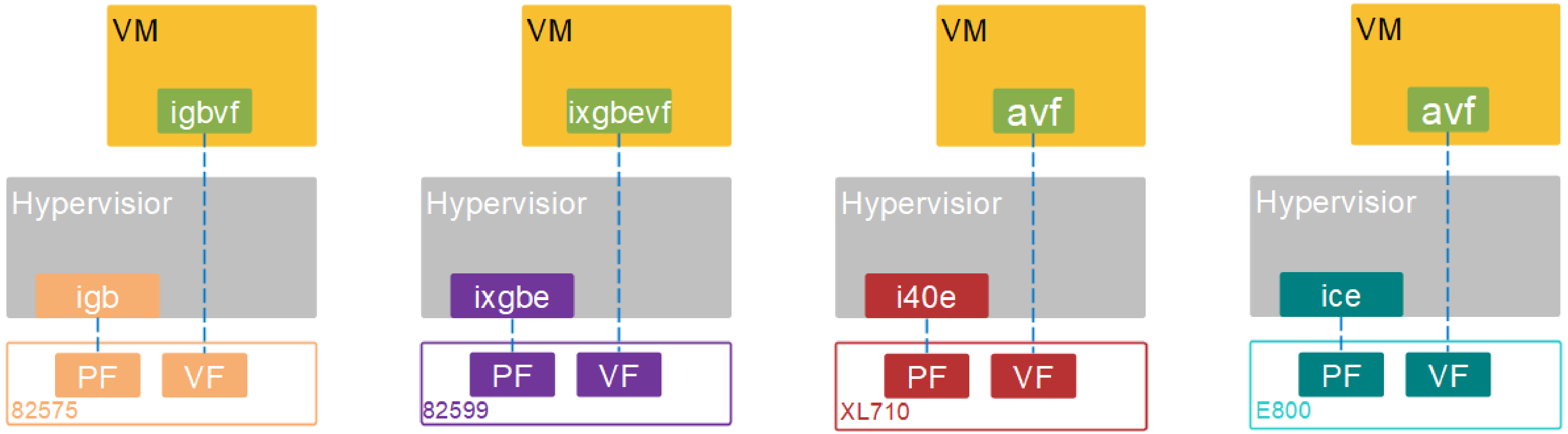
- AVF(Adaptive Virtual Function) is a specification:
  - Let one VF driver to drive different intel NIC
  - Let VM image run on the new NIC without changes.

Ideal AVF



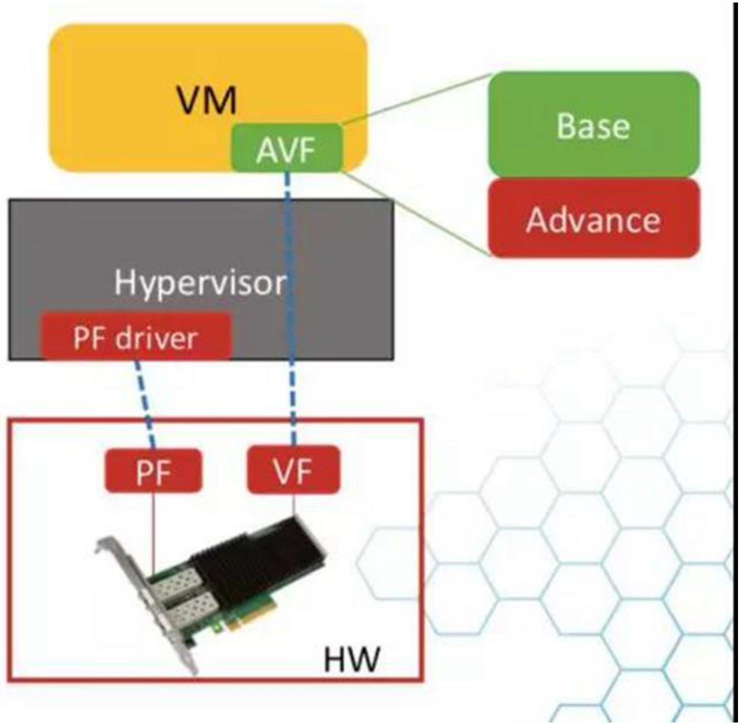
AVF Intro

Real World AVF



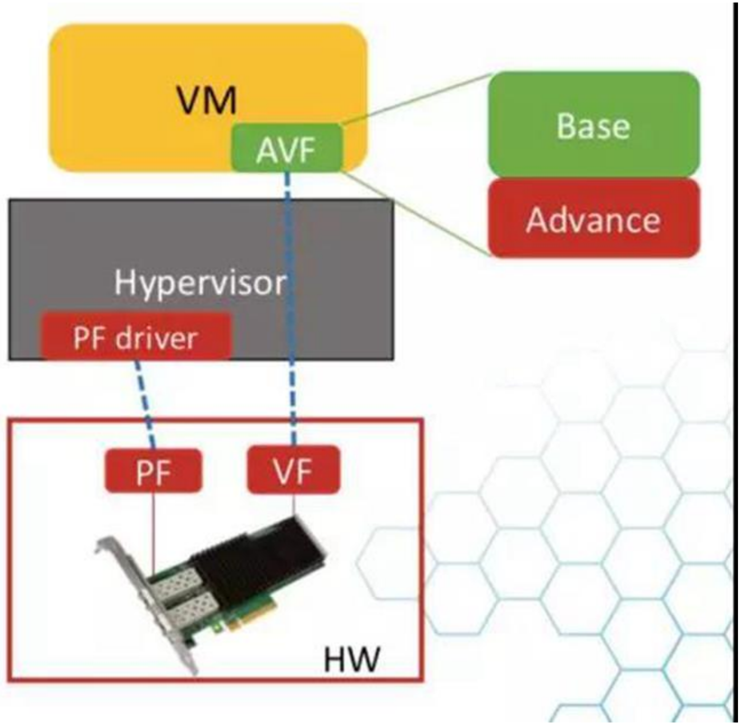
AVF Intro

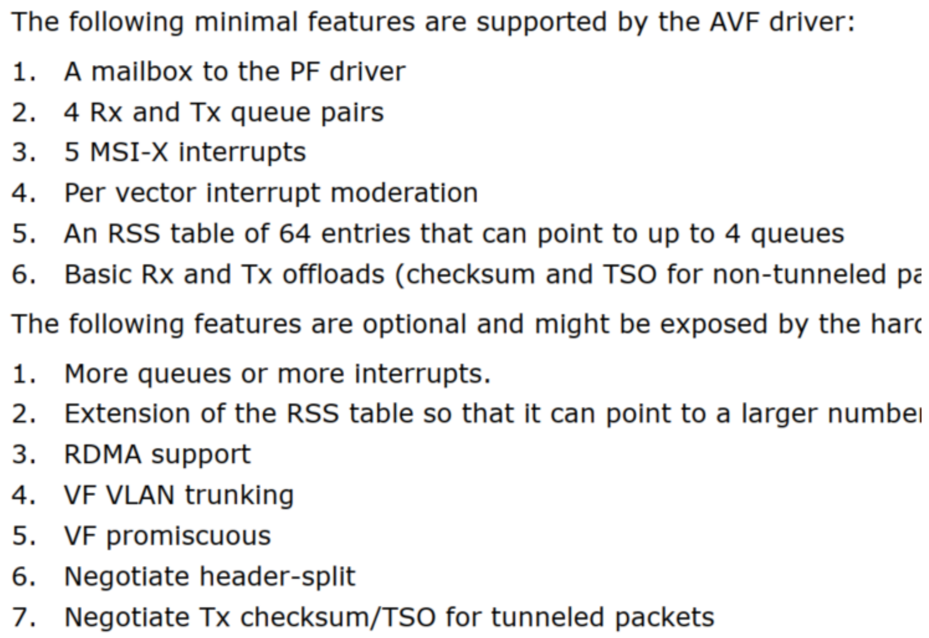
兼容性。



- How to support NIC's new feature: Base + Advanced
- Base mode support:
  - Device ID, mailbox
  - Checksum, TSO offload
  - Tx/RX, Multiqueue, MSIX
- Advanced feature:
  - Depend on NIC's capability
  - Need negotiation.

AVF Intro




The following minimal features are supported by the AVF driver:

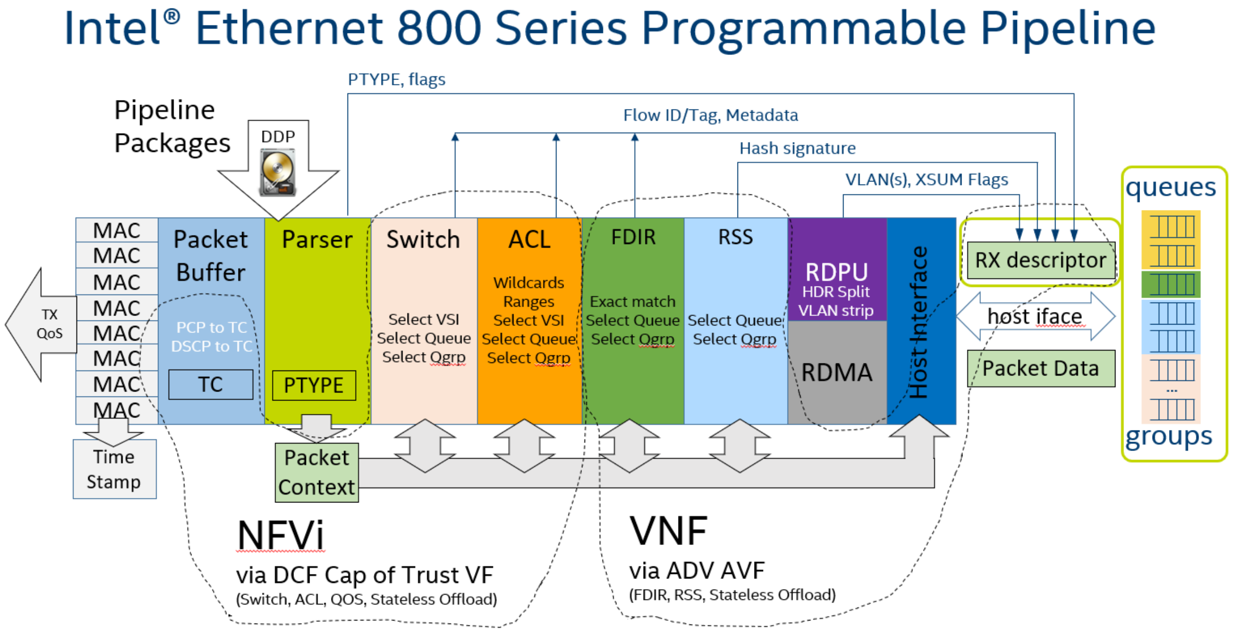
1. A mailbox to the PF driver
2. 4 Rx and Tx queue pairs
3. 5 MSI-X interrupts
4. Per vector interrupt moderation
5. An RSS table of 64 entries that can point to up to 4 queues
6. Basic Rx and Tx offloads (checksum and TSO for non-tunneled pa

The following features are optional and might be exposed by the har

1. More queues or more interrupts.
2. Extension of the RSS table so that it can point to a larger numbe
3. RDMA support
4. VF VLAN trunking
5. VF promiscuous
6. Negotiate header-split
7. Negotiate Tx checksum/TSO for tunneled packets

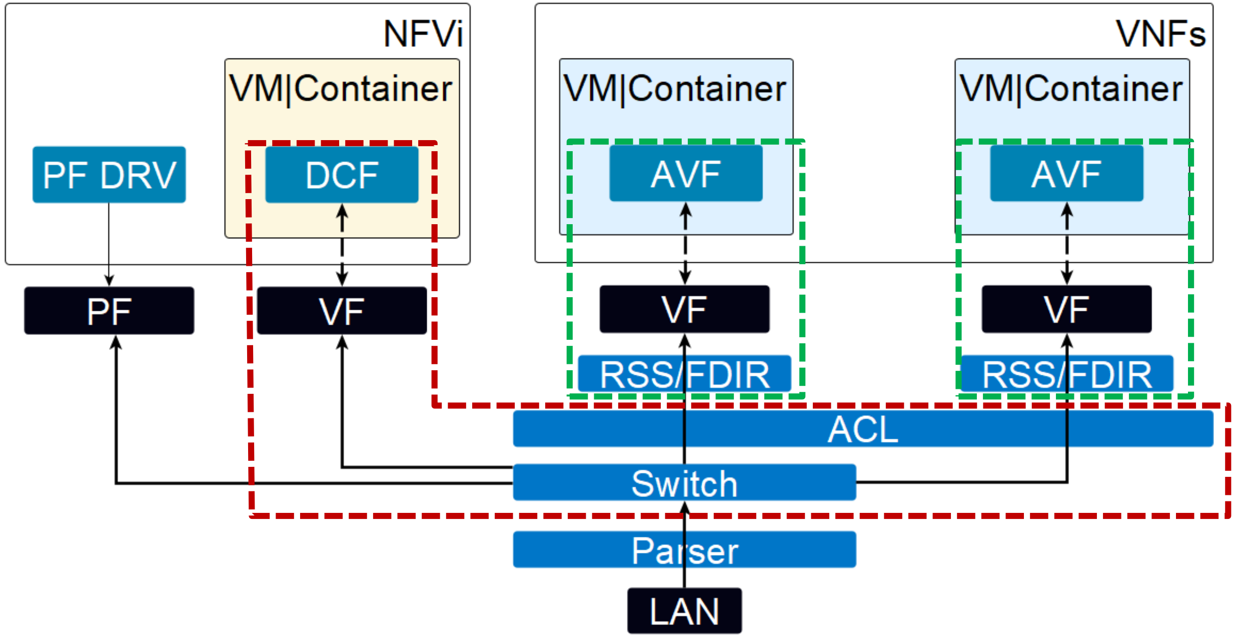
ADV Intro

➢ CVL RSS & FDIR capability mapping into AVF



ADV Intro

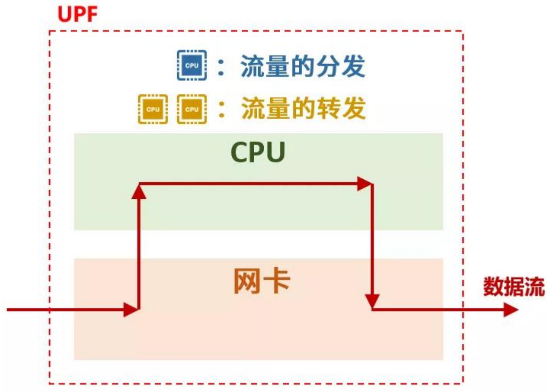
➢ CVL RSS & FDIR capability mapping into AVF

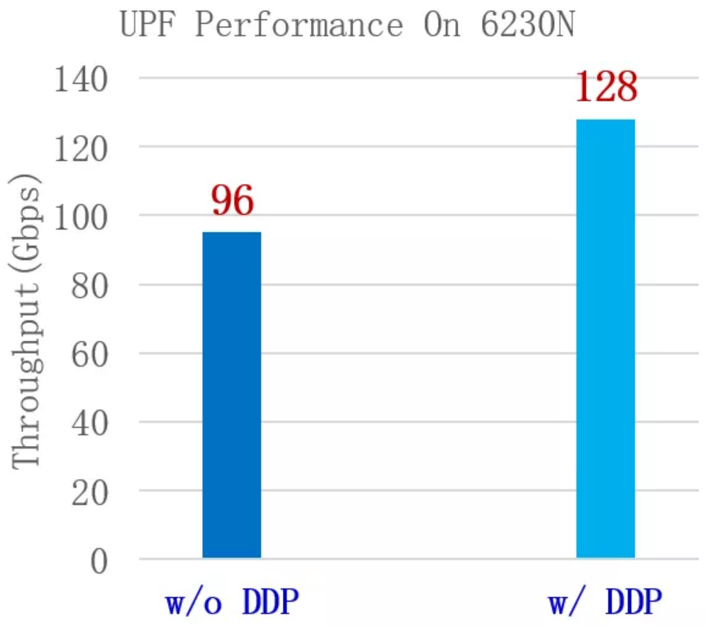


AVF Use Case

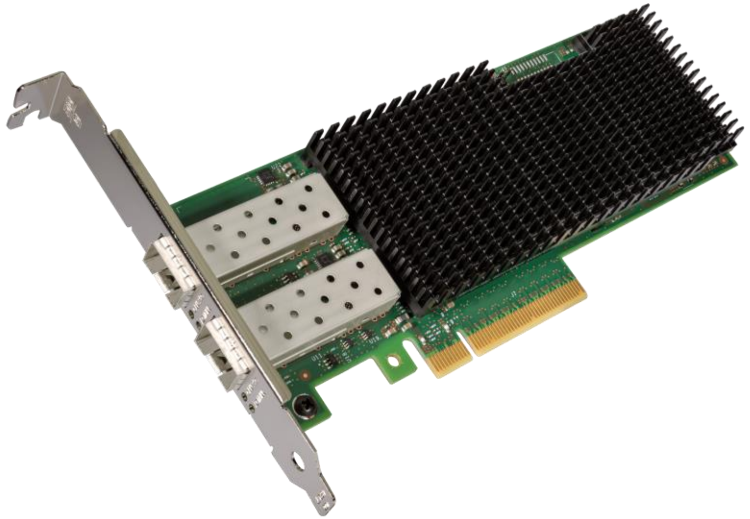
需要1/3 core进行流量分发。

- 5G UPF use 1/3 cores for traffic distribution.
- Use DCF + AVF can offload cpu





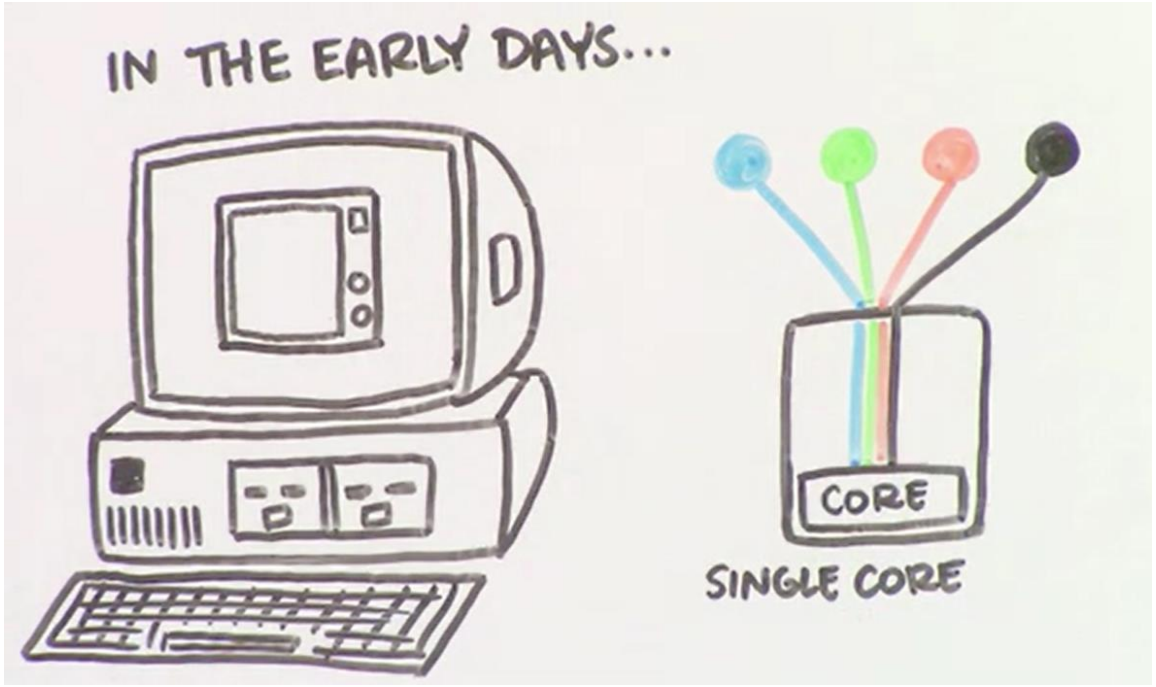
Agenda



- AVF Introduction
- **RSS Introduction**
- FDIR Introduction
- Virtual Channel Introduction

RSS Intro

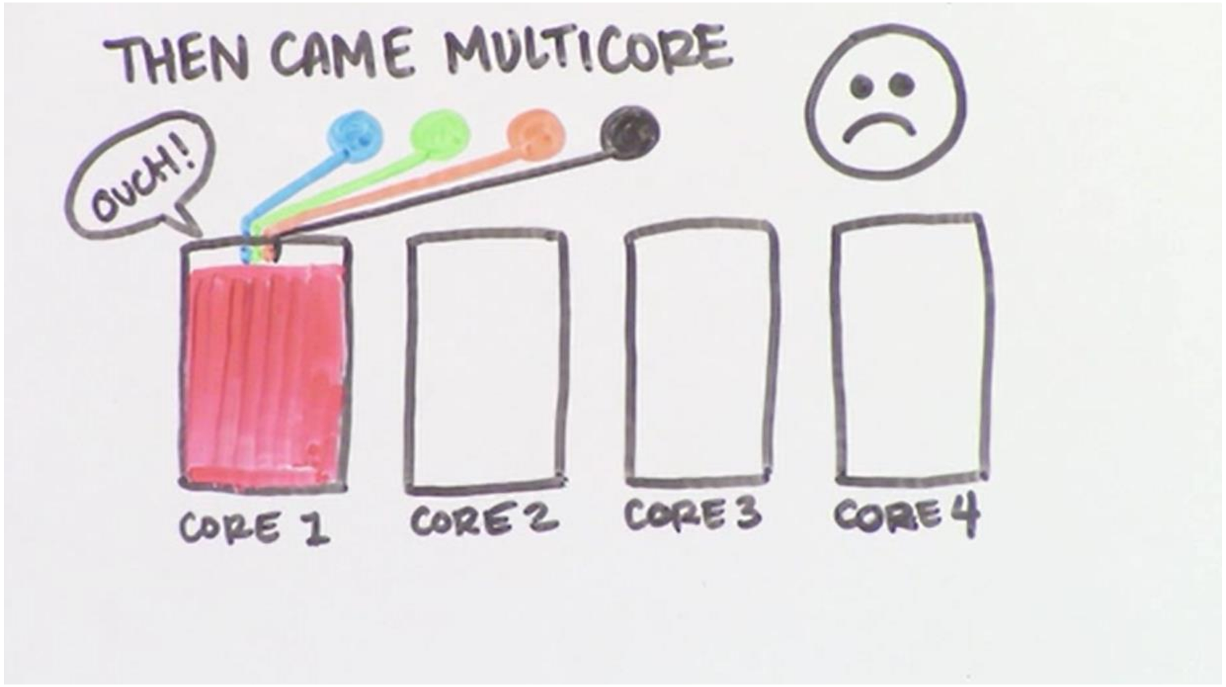
In early days, all network packets are forwarded to the single core.



早期，一个queue，一个core。

RSS Intro

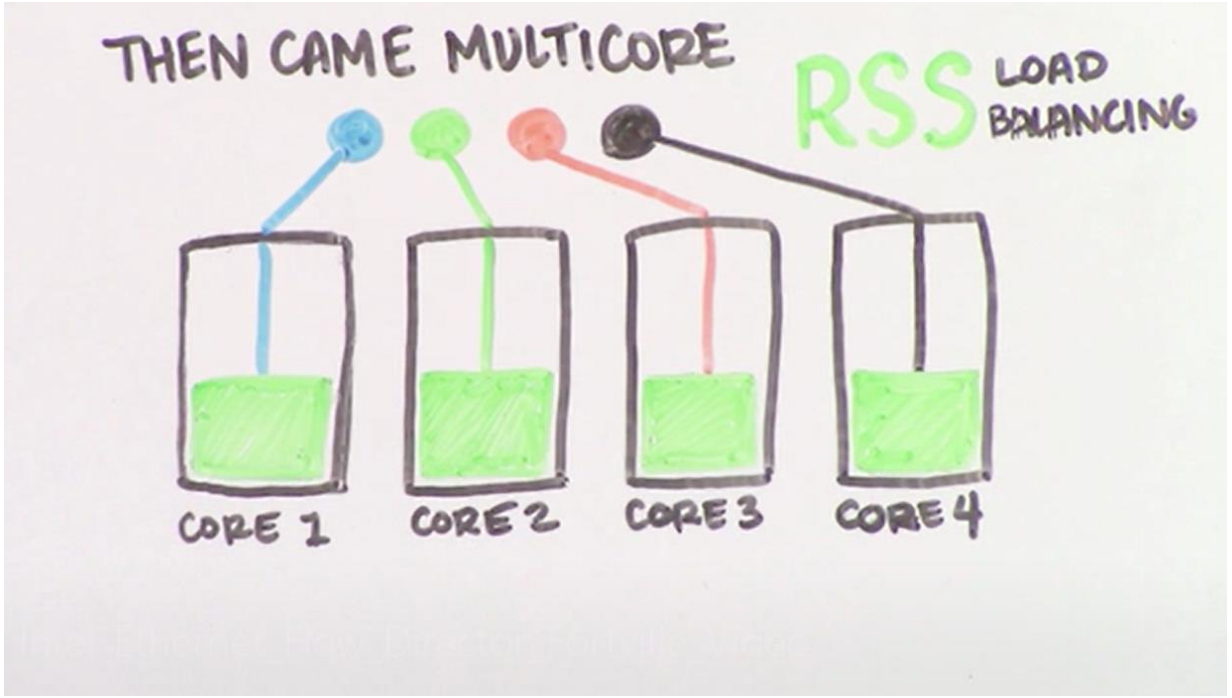
Here, all packets are forwarded only to the specific one process core.
Network does not benefit from multicore.



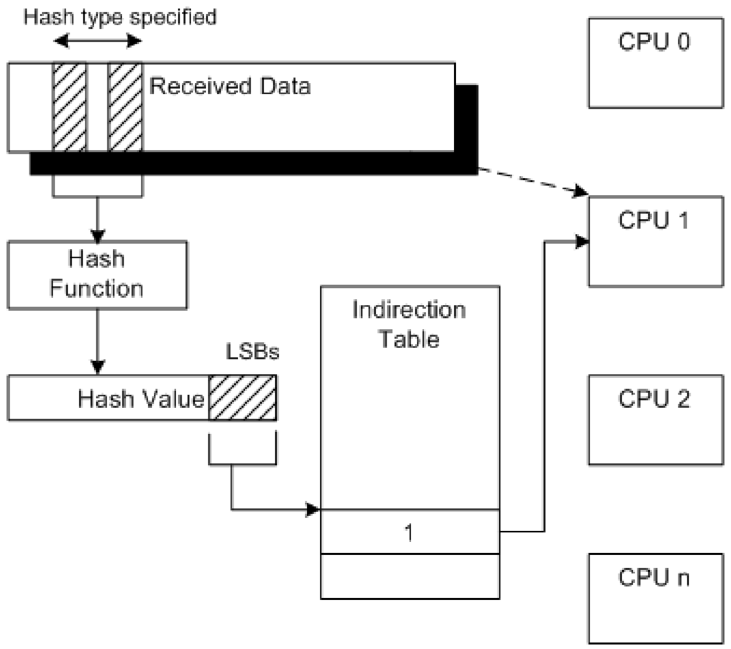
package是到queue，不是到core吧

RSS Intro

Then Microsoft introduces RSS to do load balancing.



RSS Intro



- Early days, NIC has only 1 queue. RSS is done by cpu cores on Windows.

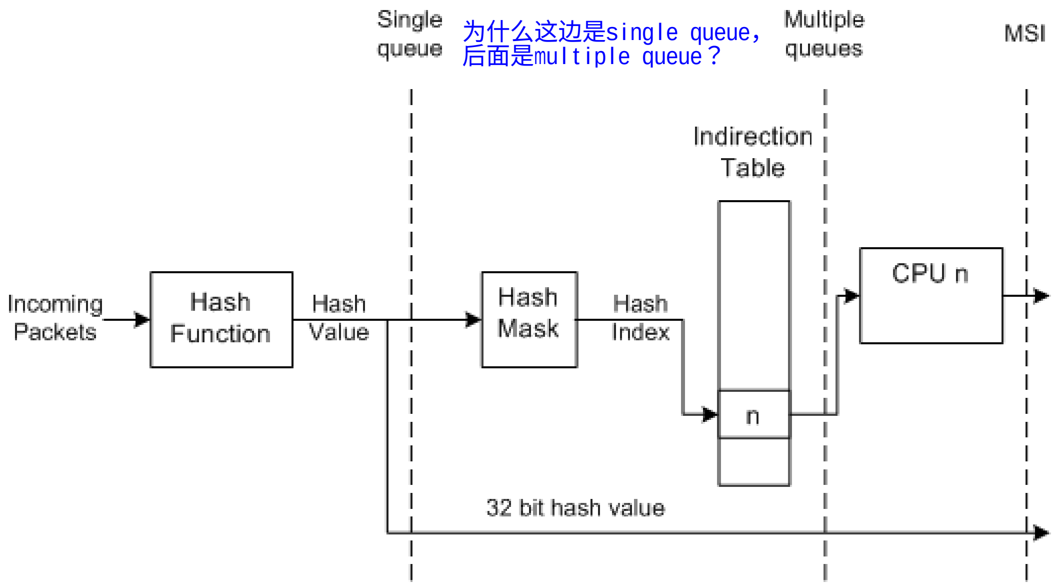
  只有一个queue怎么做RSS？

- Linux RSS software implementation:
  Receive Packet Steering
  Receive Flow Steering

RSS Intro

- NIC support for RSS:
  1. Hash for Single queue
  2. Hash for multi queue
  3. MSI/MSIX

算hash，查表，分发。



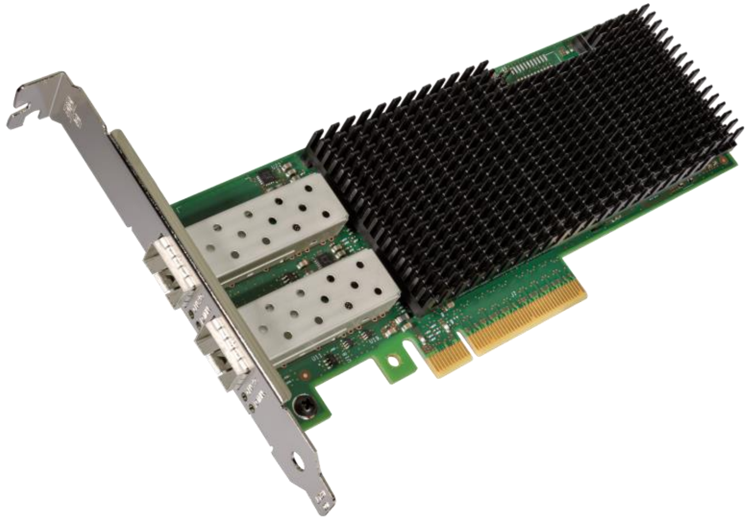
RSS Use Case

- Default setup:
  5-tuple

- Business specific distribution:
  PPPoE, GTP-U, etc

```
flow create 0 ingress pattern eth / ipv4 / udp / gtpu / gtp_psc / ipv4 / end
actions rss types ipv4 l3-src-only end key_len 0 queues end / end
```

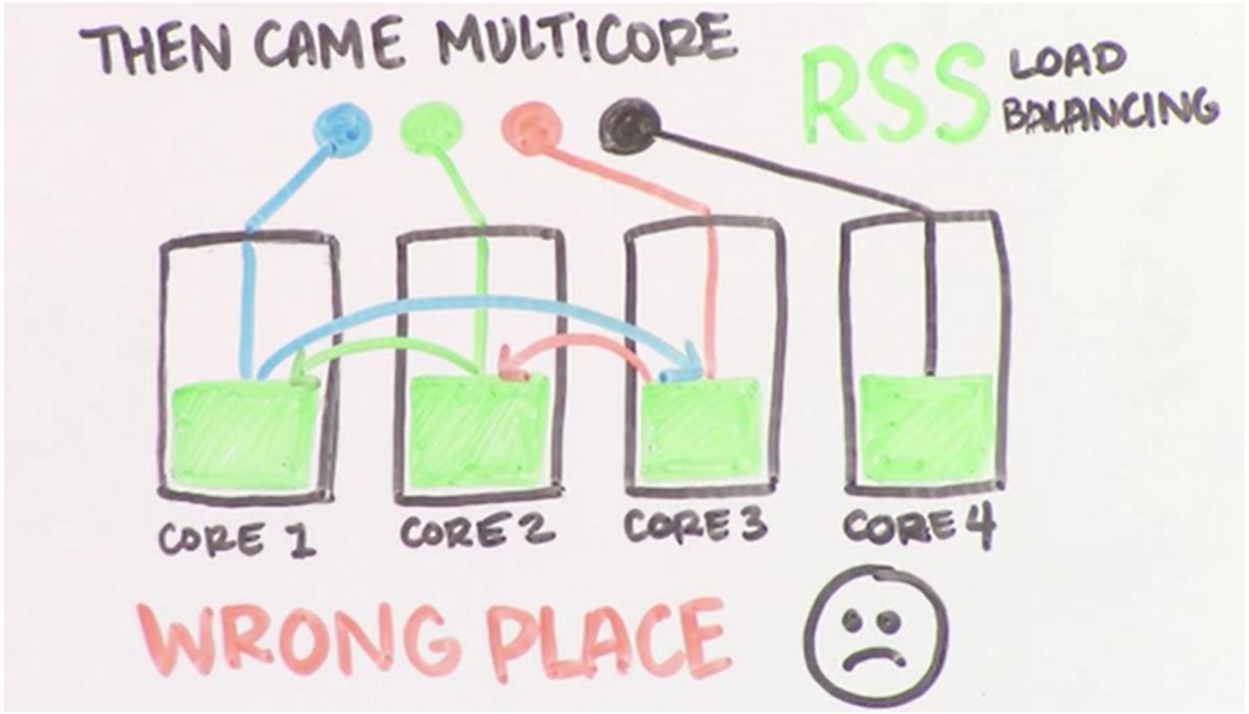
Agenda



- AVF Introduction
- RSS Introduction
- **FDIR Introduction**
- Virtual Channel Introduction

FDIR Intro

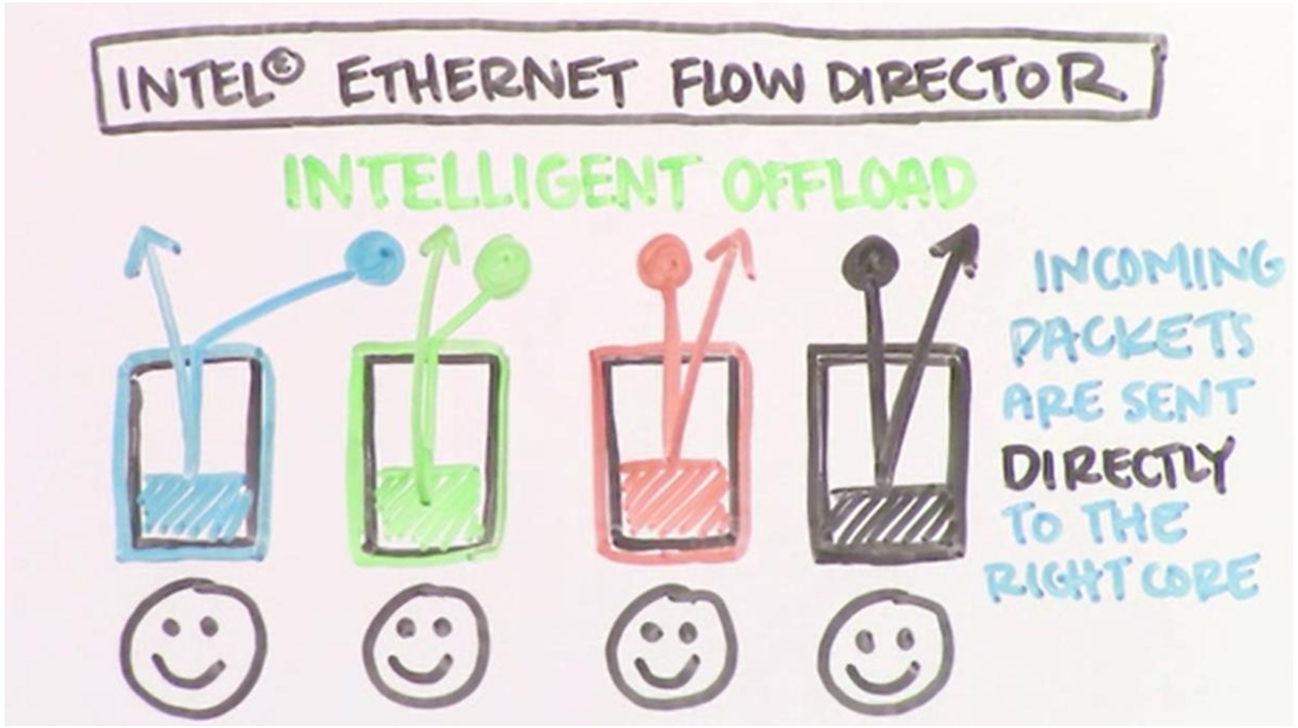
However we can't let data go to specific core, which still destroys multicore performance



FDIR Intro

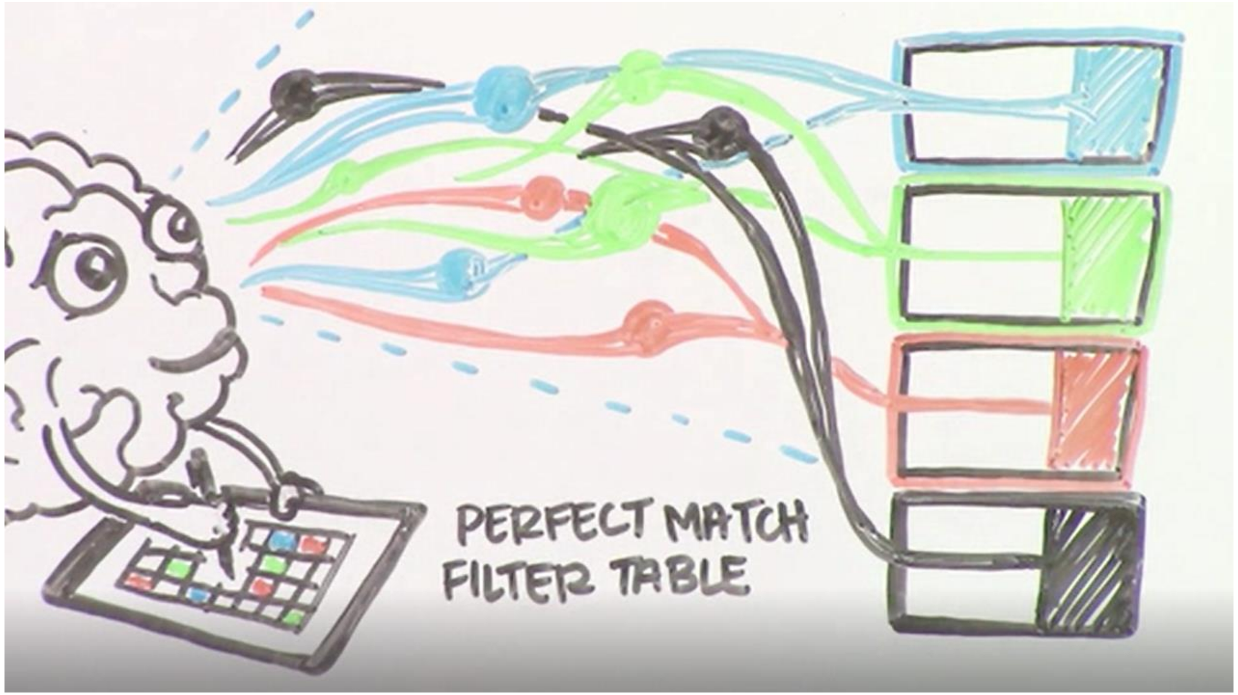
精准导流。细粒度优化。

Then Intel Flow Director is coming! FDIR can send incoming packets directly to the right core.


INTEL© ETHERNET FLOW DIRECTOR
INTELLIGENT OFFLOAD
INCOMING PACKETS ARE SENT DIRECTLY TO THE RIGHT CORE

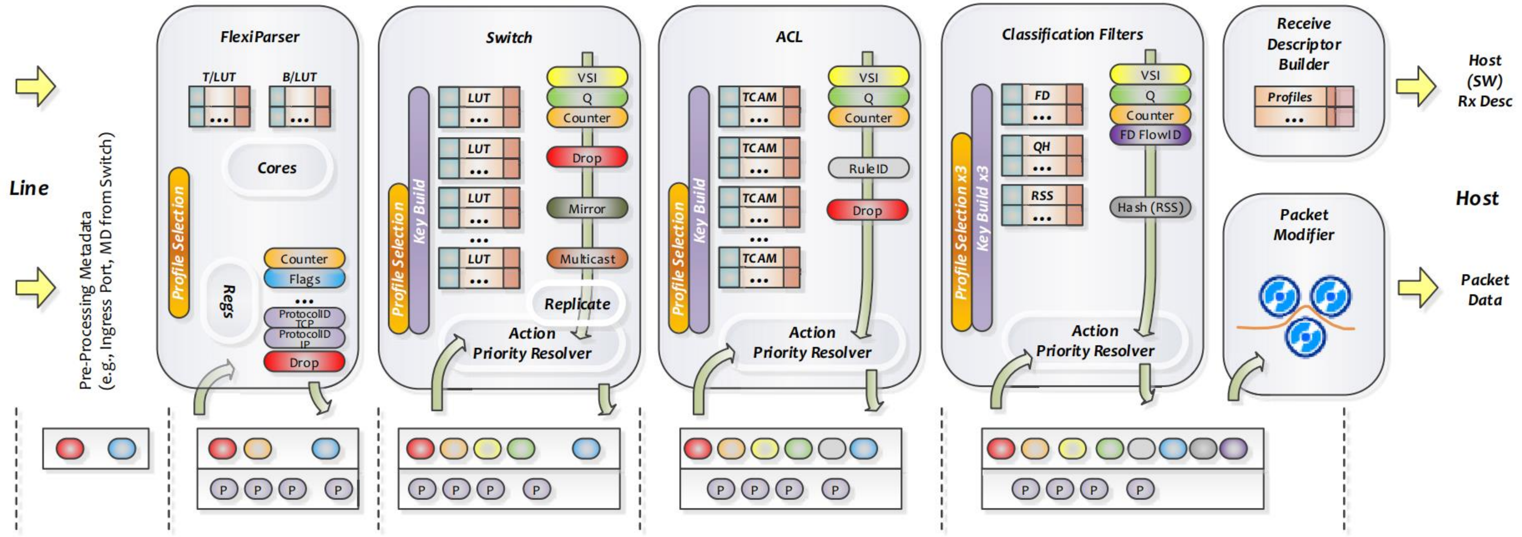
FDIR Intro

Flow Director has a match table to send matched packets to the desired process core.



FDIR Intro

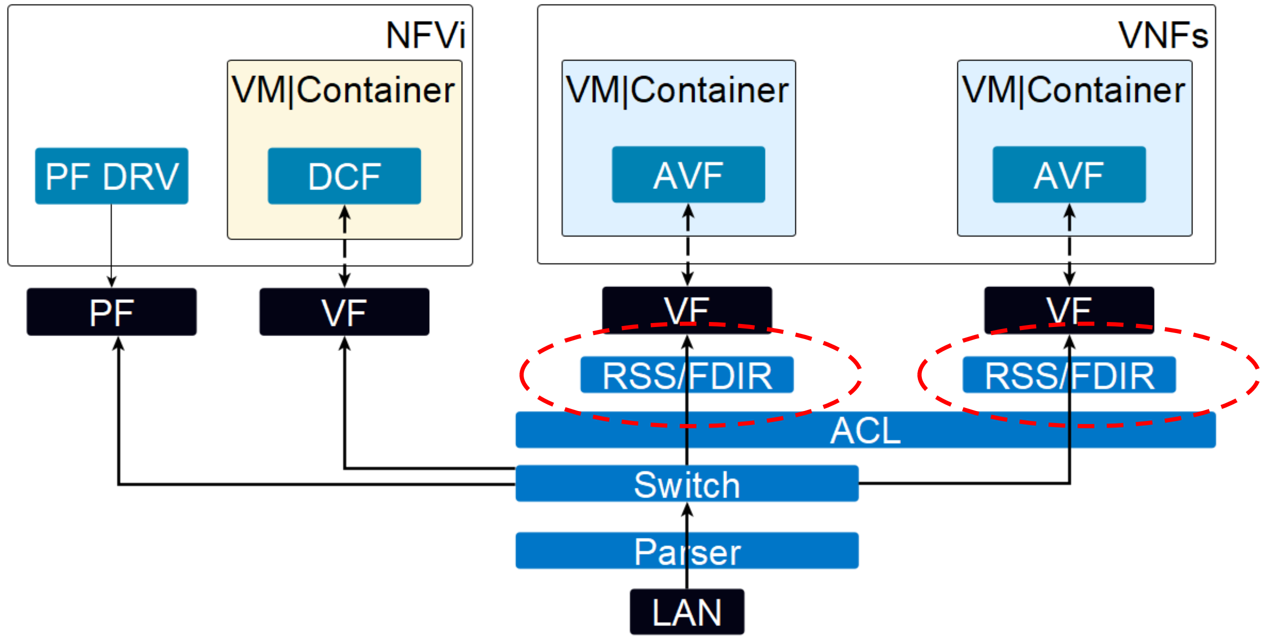
Where is FD in CVL packet processing pipeline?



不同颜色的圆圈代表不同的action（e.g. switch: to Q0, ACL: to Q1），它们有不同的priority。

ADV Intro

Where is FD in CVL packet processing pipeline?

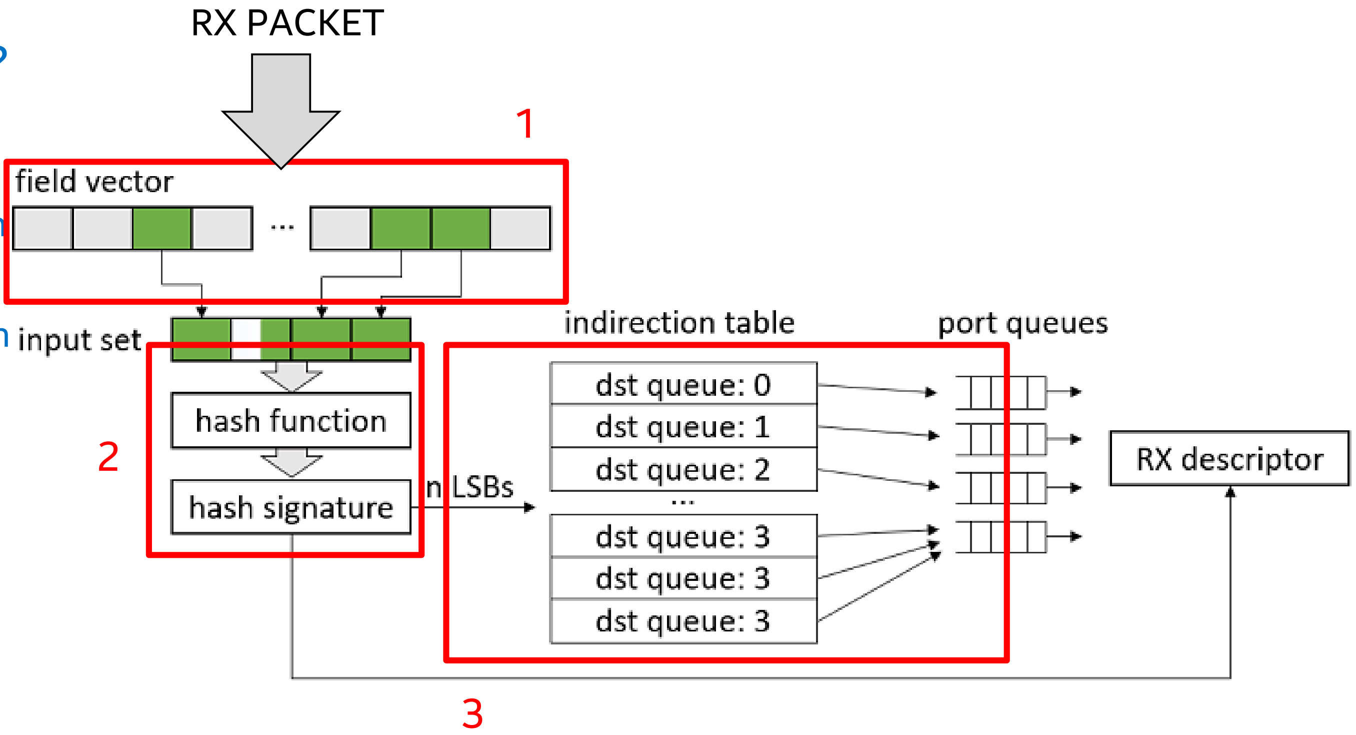


先FDIR,再RSS。
如果FDIR命中，则不会再做后面的匹配。

FDIR Intro

How does FDIR work?

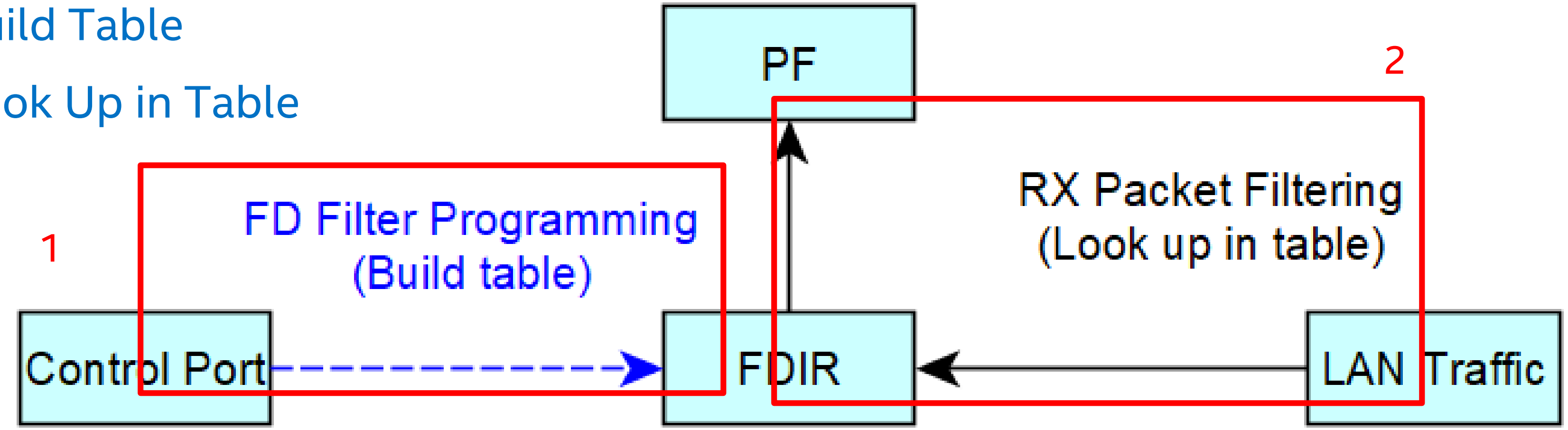
1. Field vector extracts rx packets' specific domain as input set
2. Input set generates hash signature by hash function
3. Hash signature looks up matched item in the indirection table and then do forwarding



FDIR Intro

How FDIR function block is designed ?

1. Build Table
2. Look Up in Table

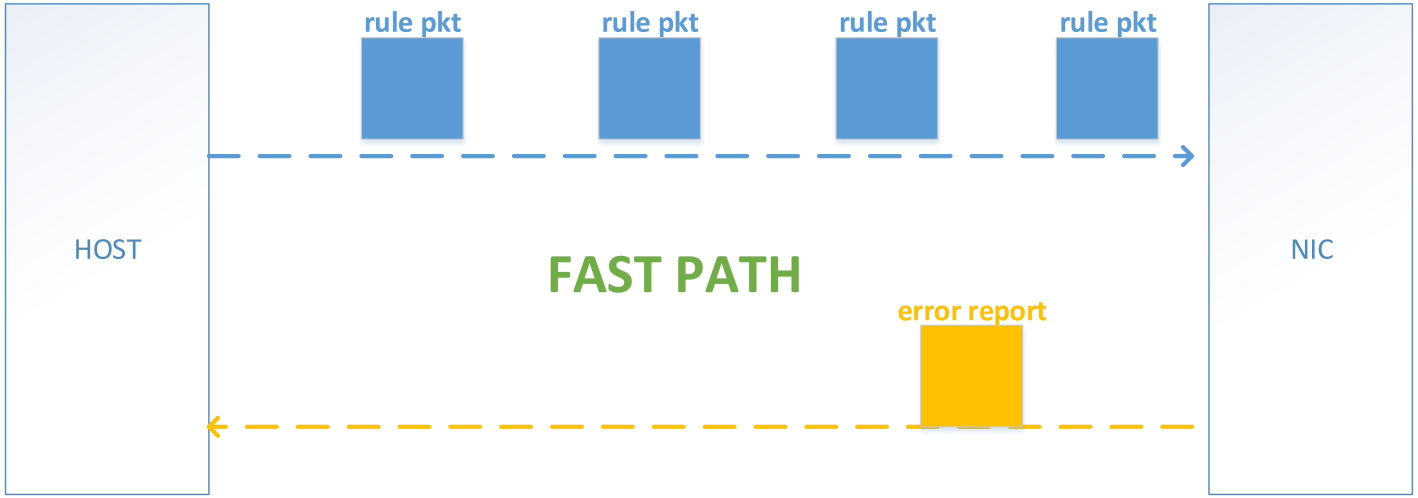


FDIR Intro

通过package写表。
写表的过程其实就是收发包的过程。

How to build FDIR look up table?

By FDIR TX/RX queues !

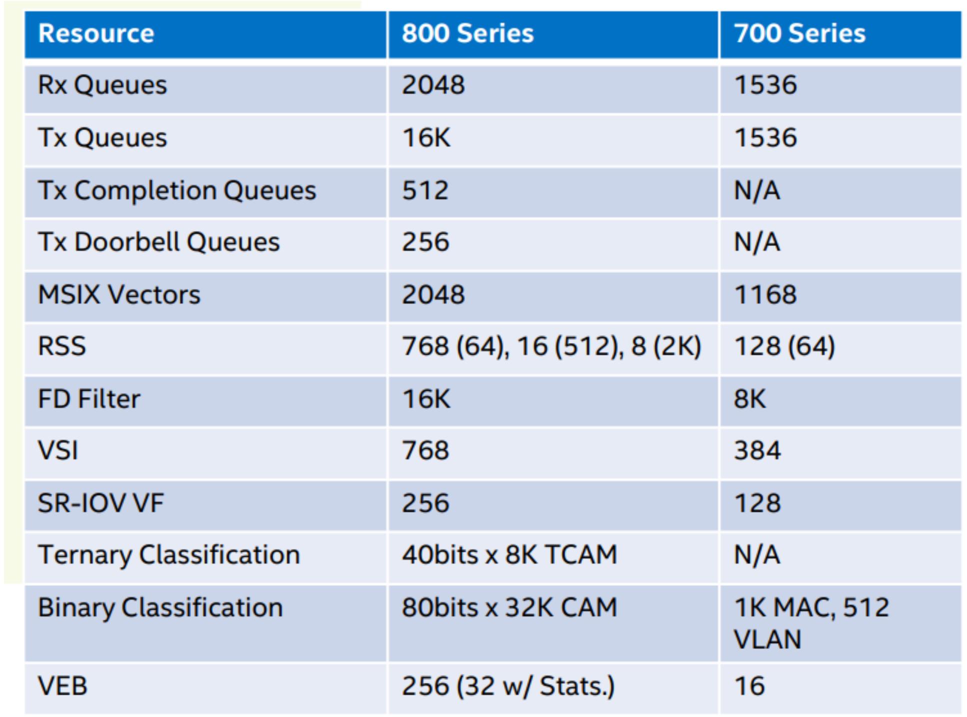


Picture From DPDK Summit North America 2019, *Rte_flow Optimization in i40e Driver* - Chenmin Sun, Intel

FDIR Intro

1. Enable FDIR queues:
   a) Request for VSI
   b) Request for TX/RX queues
   c) Request for TX/RX irq
   d) Prepare Ring buffer
2. Allocate FDIR rule space
3. Build FDIR rule table

how?



Resource	800 Series	700 Series
Rx Queues	2048	1536
Tx Queues	16K	1536
Tx Completion Queues	512	N/A
Tx Doorbell Queues	256	N/A
MSIX Vectors	2048	1168
RSS	768 (64), 16 (512), 8 (2K)	128 (64)
FD Filter	16K	8K
VSI	768	384
SR-IOV VF	256	128
Ternary Classification	40bits x 8K TCAM	N/A
Binary Classification	80bits x 32K CAM	1K MAC, 512 VLAN
VEB	256 (32 w/ Stats.)	16

8000条 rule->16000条 rule。

FVL vs CVL resource table

CVL Flow Director Enablement

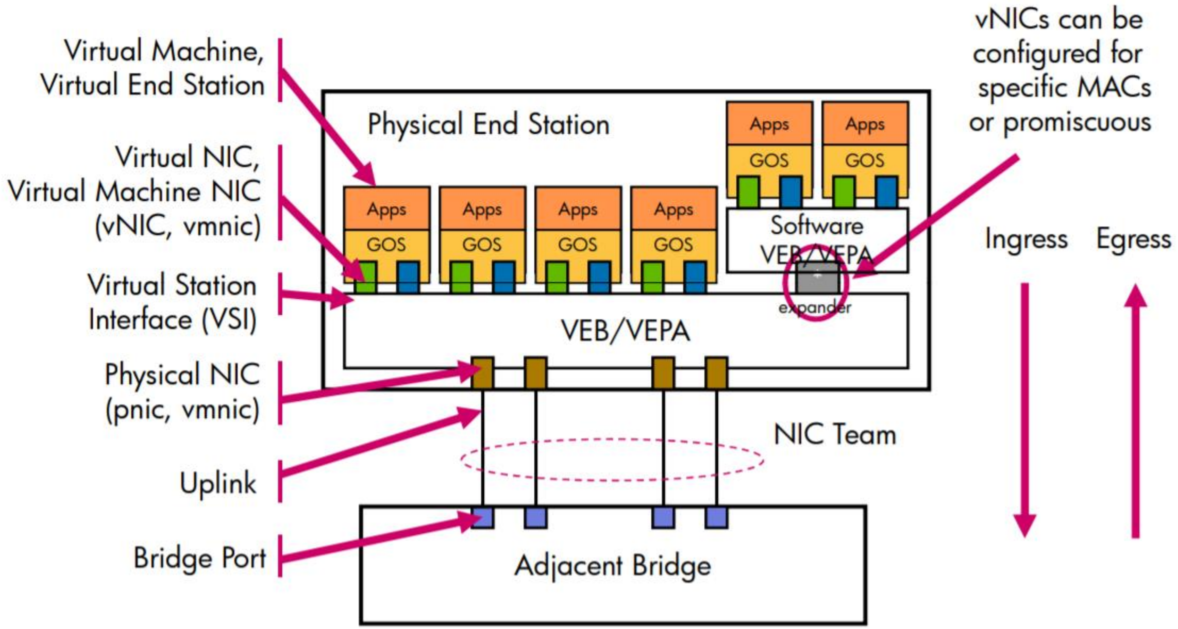
To enable FDIR queues

a) Request for VSI

1. What is VSI?
   Virtual Station Interface

   An internal point-to-point Ethernet connection which connects an Ethernet port of a VEB or VEPA to a vNIC

2. VSI type in CVL:
   VF, VMDq2, PF, EMP/MNG

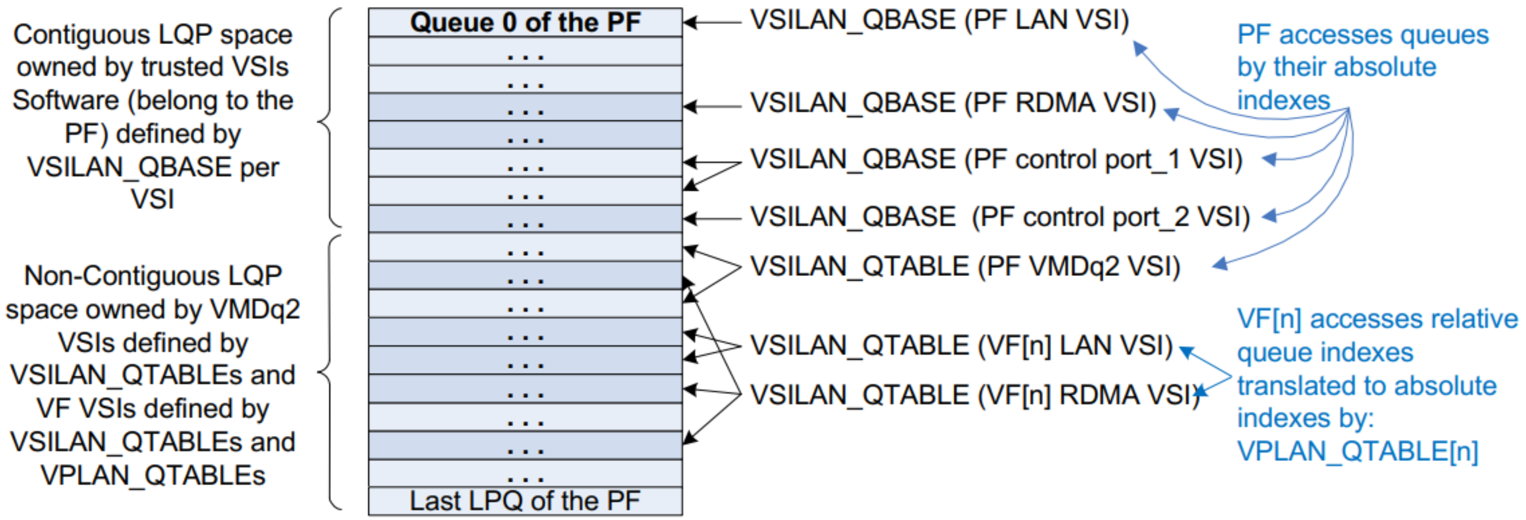

Virtual Machine, Virtual End Station
Virtual NIC, Virtual Machine NIC (vNIC, vmnic)
Virtual Station Interface (VSI)
Physical NIC (pnic, vmnic)
Uplink
Bridge Port
Physical End Station
Apps
GOS
Software VEB/VEPA
expander
VEB/VEPA
NIC Team
Adjacent Bridge
vNICs can be configured for specific MACs or promiscuous
Ingress
Egress

CVL Flow Director Enablement

To enable FDIR queues:

b) Request for TX/RX queue

Queues can be requested in Contiguous/Scattered mode

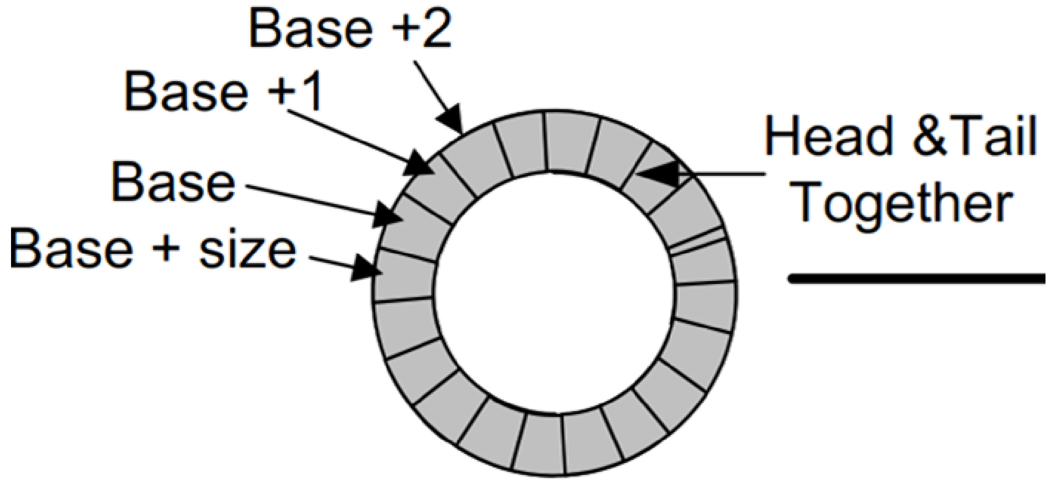


Contiguous LQP space owned by trusted VSIs Software (belong to the PF) defined by VSILAN_QBASE per VSI
Non-Contiguous LQP space owned by VMDq2 VSIs defined by VSILAN_QTABLEs and VF VSIs defined by VSILAN_QTABLEs and VPLAN_QTABLEs
Queue 0 of the PF
...
Last LPQ of the PF
VSILAN_QBASE (PF LAN VSI)
VSILAN_QBASE (PF RDMA VSI)
VSILAN_QBASE (PF control port_1 VSI)
VSILAN_QBASE (PF control port_2 VSI)
VSILAN_QTABLE (PF VMDq2 VSI)
VSILAN_QTABLE (VF[n] LAN VSI)
VSILAN_QTABLE (VF[n] RDMA VSI)
PF accesses queues by their absolute indexes
VF[n] accesses relative queue indexes translated to absolute indexes by: VPLAN_QTABLE[n]

CVL Flow Director Enablement

To enable FDIR queues:

d) Prepare Ring Buffer



CVL Flow Director Enablement

Allocate FDIR rule space:

How is FD rule space managed?

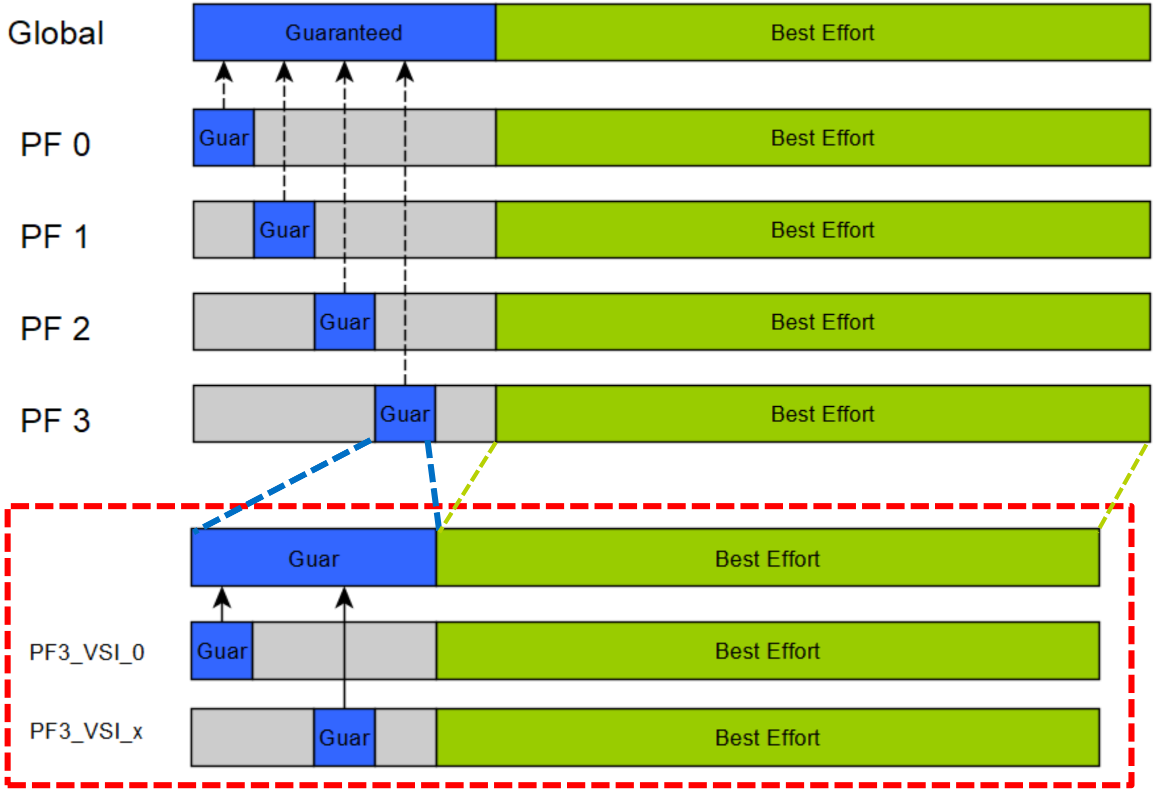
1. 16K filter rules are shared among PFs by:

   Unique ("guaranteed") 独占
   +
   Shared ("best effort") 共享，先到先得。

e.g. In 25Gx4 Card:
guaranteed = 512,
best_effort =

16K – 512 * Active_PF

目前，vf只用best effort里的资源。



CVL Flow Director Enablement

How to Build FDIR rule table?

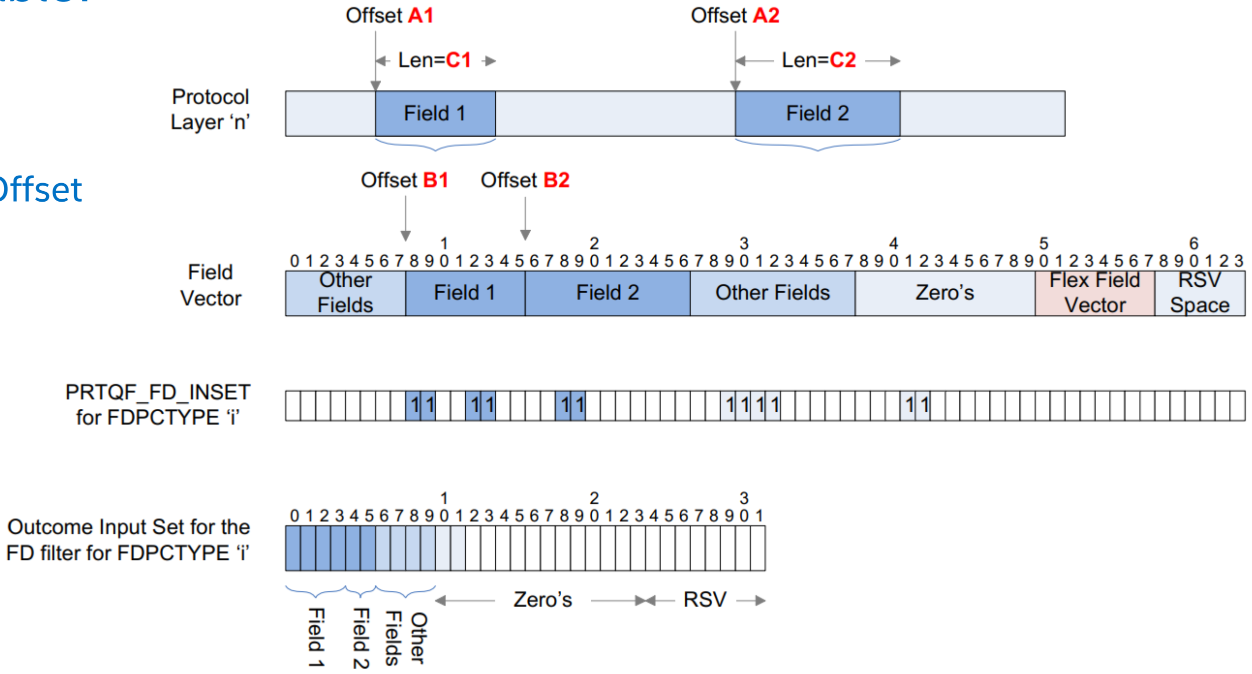
1. Initialize Field Vector Word

   What is Field Vector ?
   A combination set of Word Offset and Word Length

   CVL support up to 24 words matching

The field vector specifies which packet header fields are considered when creating FDIR filters,

字段匹配：需要知道字段在以太网包里的偏移和长度。



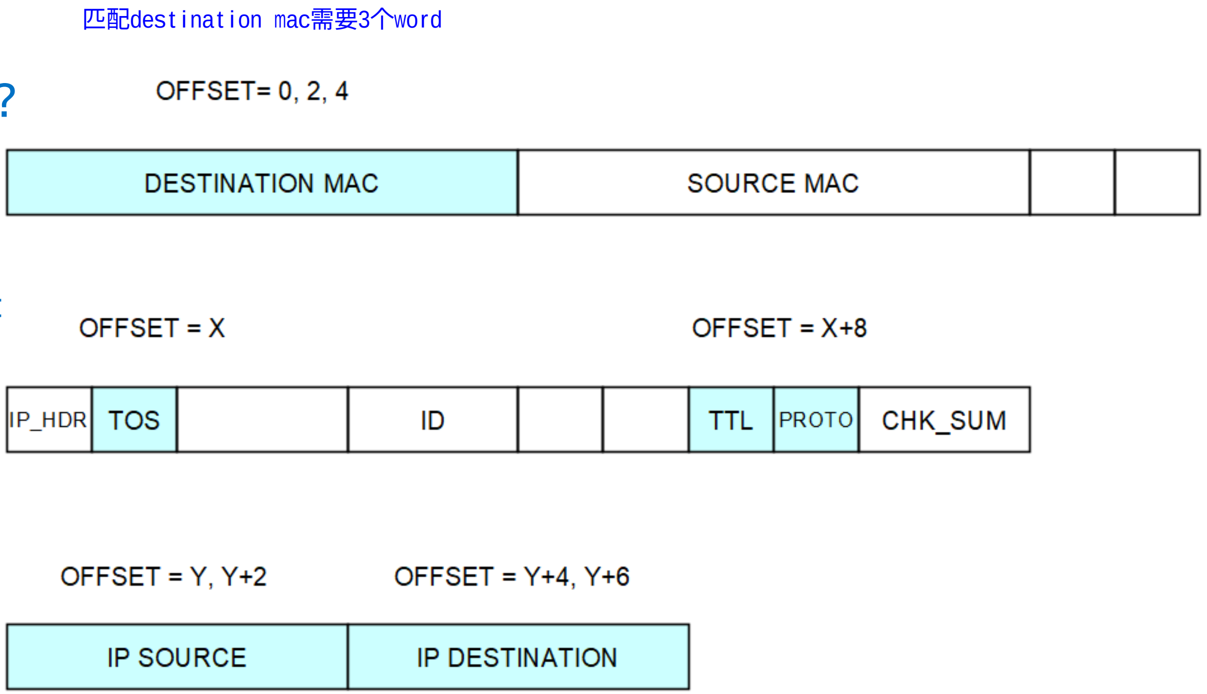
CVL Flow Director Enablement

How to Build FDIR rule table?

1. Initialize Field Vector Word

   What is Field Vector ?
   A combination set of Word Offset and Word Length

   CVL support up to 24 words matching



```
ice_fdir_filter.c
ice_fdir_parse_pattern(){} //这个过程可以看看
const struct rte_flow_item_ipv4 *ipv4_spec, *ipv4_mask;
```

CVL Flow Director Enablement

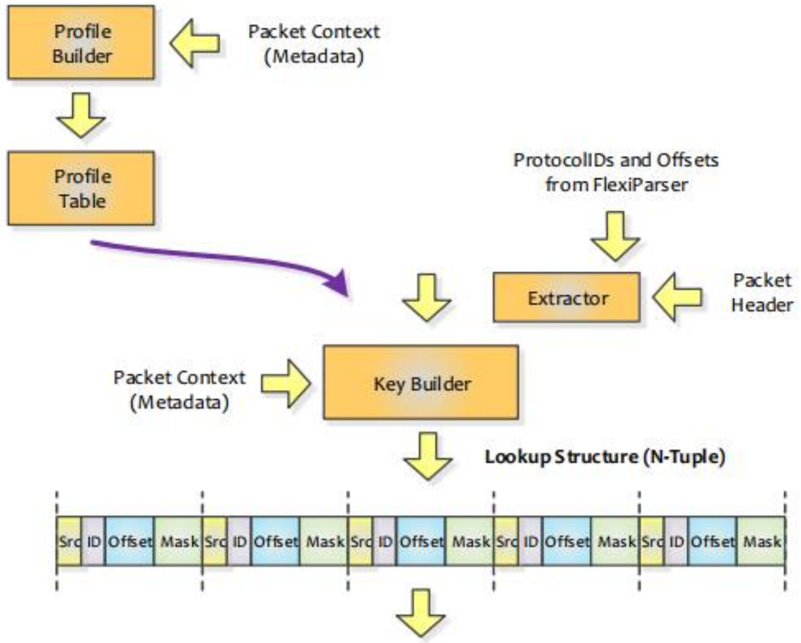
How to Build FDIR rule table?

1. Initialize Field Vector Word

   FV is initialized by creating Profile

   E.g.
   IPV4+TCP/UDP/SCTP
   IPV6+TCP/UDP/SCTP
   IPV4+UDP+VXLAN+MAC+IPV4+TCP/UDP/SCTP
   IPV4+UDP+GTPU+PDU+IPV4+TCP/UDP/SCTP

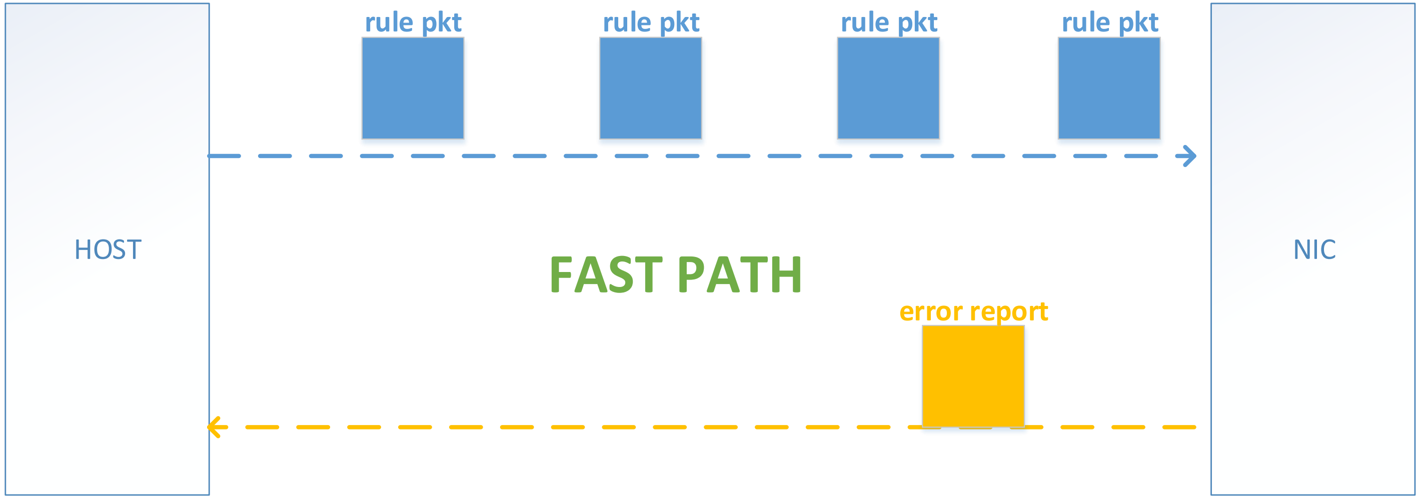
创建profile的过程就是在初始化field vector。


Profile Builder
Packet Context (Metadata)
Profile Table
ProtocolIDs and Offsets from FlexiParser
Extractor
Packet Header
Packet Context (Metadata)
Key Builder
Lookup Structure (N-Tuple)
Src ID Offset Mask Src ID Offset Mask Src ID Offset Mask Src ID Offset Mask Src ID Offset Mask

CVL Flow Director Enablement

How to Build FDIR rule table?

1. Create Profile
2. Program the input set value and table by FDIR TX queue.

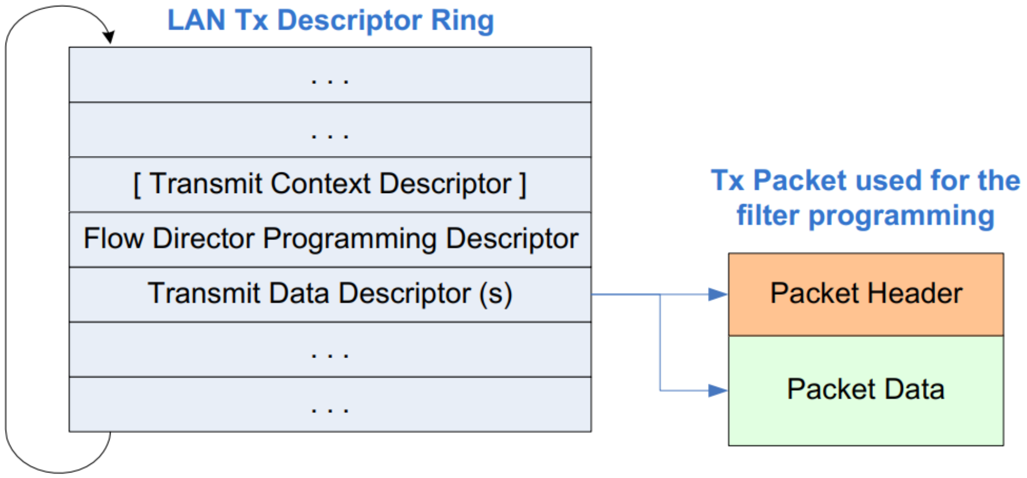


Picture From DPDK Summit North America 2019, *Rte_flow Optimization in i40e Driver* - Chenmin Sun, Intel

CVL Flow Director Enablement

How to Build FDIR rule table?

1. Create Profile
2. Program the input set value and table by FDIR TX queue.
   a) Get free queue descriptor



LAN Tx Descriptor Ring
. . .
. . .
[Transmit Context Descriptor]
Flow Director Programming Descriptor
Transmit Data Descriptor (s)
. . .
. . .
Tx Packet used for the filter programming
Packet Header
Packet Data

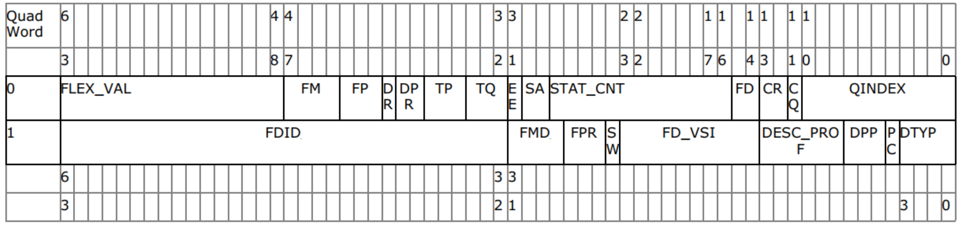
Flow Director Filter Programming Descriptor Queue

CVL Flow Director Enablement

填包。

How to Build FDIR rule table?

1. Create Profile
2. Program the input set value and table by FDIR TX queue.
   a) Get free queue descriptor
   b) Fill LAN+FDIR descriptor



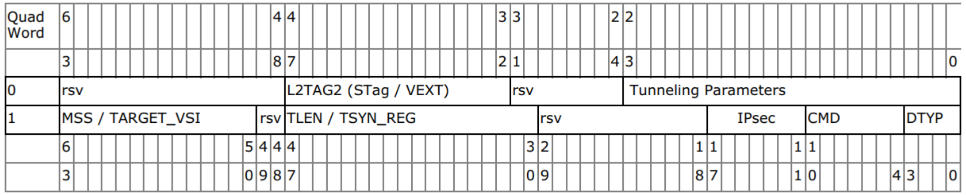
Quad Word — bits 63 48 | 47 32 | 31 23 | 22 17 | 16 14 | 13 11 | 10 0

Quad Word														
0	FLEX_VAL	FM	FP	DR	DPR	TP	TQ	EE	SA	STAT_CNT	FD	CR	CQ	QINDEX

Quad Word									
1	FDID	FMD	FPR	SW	FD_VSI	DESC_PROF	DPP	PC	DTYP

Bits 63 32 | 31 3 0

FD Filter Programming Descriptor



Quad Word — bits 63 48 | 47 32 | 31 24 | 23 0

Quad Word				
0	rsv	L2TAG2 (STag / VEXT)	rsv	Tunneling Parameters

Quad Word							
1	MSS / TARGET_VSI	rsv	TLEN / TSYN_REG	rsv	IPsec	CMD	DTYP

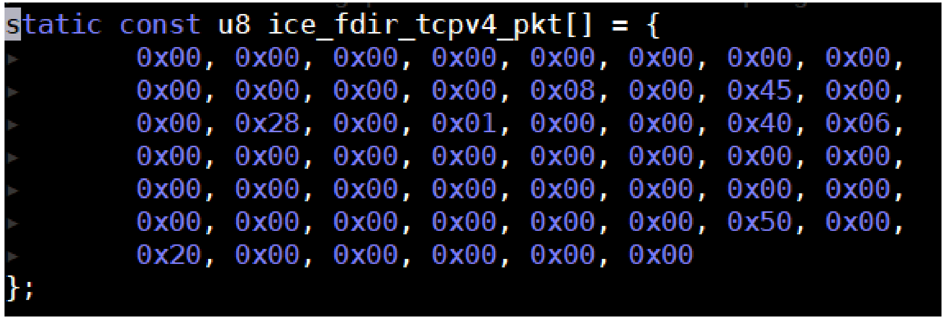
Bits 63 50 | 49 48 | 47 30 | 29 18 | 17 11 | 10 4 | 3 0

Transmit Data Descriptor

CVL Flow Director Enablement

How to Build FDIR rule table?

1. Create Profile
2. Program the input set value and table by FDIR TX queue.
   a) Get free queue descriptor
   b) Fill LAN+FDIR descriptor
   c) Prepare FDIR dummy packet

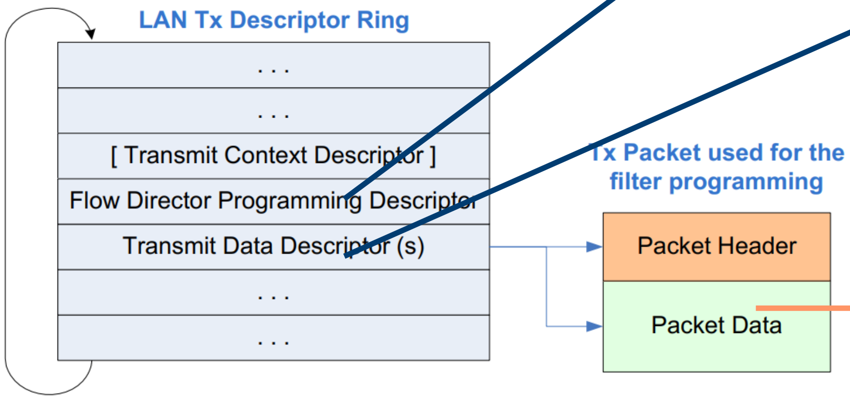


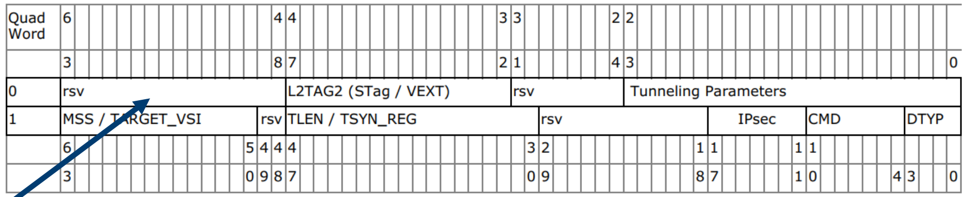
```
static const u8 ice_fdir_tcpv4_pkt[] = {
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
        0x00, 0x00, 0x00, 0x00, 0x08, 0x00, 0x45, 0x00,
        0x00, 0x28, 0x00, 0x01, 0x00, 0x00, 0x40, 0x06,
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x50, 0x00,
        0x20, 0x00, 0x00, 0x00, 0x00, 0x00
};
```

CVL Flow Director Enablement

How to Build FDIR rule table?

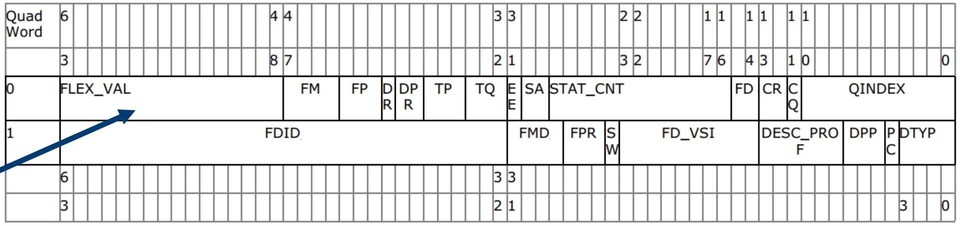
1. Create Profile
2. Program the input set value and table by FDIR TX queue.


LAN Tx Descriptor Ring
. . .
. . .
[Transmit Context Descriptor]
Flow Director Programming Descriptor
Transmit Data Descriptor (s)
. . .
. . .
Tx Packet used for the filter programming
Packet Header
Packet Data



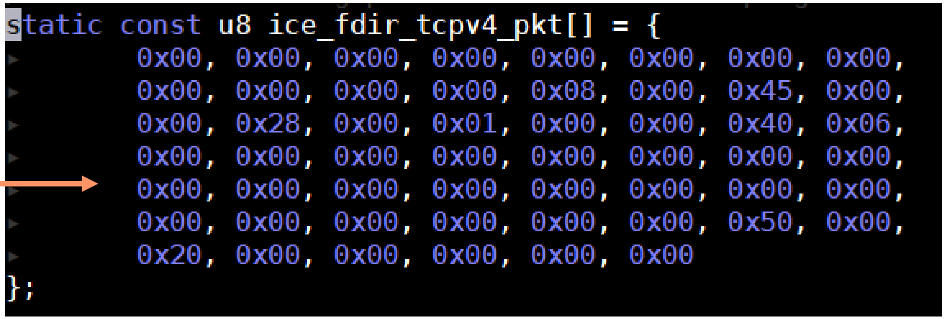
Quad Word	63 … 48	47 … 32	31 … 24	23 … 0
0	rsv	L2TAG2 (STag / VEXT)	rsv	Tunneling Parameters

Quad Word	MSS / TARGET_VSI	rsv	TLEN / TSYN_REG	rsv	IPsec	CMD	DTYP
1	63 … 50	49 48	47 … 30	29 … 18	17 … 11	10 … 4	3 … 0



Quad Word	FLEX_VAL	FM	FP	DR	DPR	TP	TQ	EE	SA	STAT_CNT	FD	CR	CQ	QINDEX
	63 … 48	47					… 32	31		… 23 22 … 17 16	14	13	11	10 … 0
0	FLEX_VAL	FM	FP	DR	DPR	TP	TQ	EE	SA	STAT_CNT	FD	CR	CQ	QINDEX

Quad Word	FDID	FMD	FPR	SW	FD_VSI	DESC_PROF	DPP	PC	DTYP
1	FDID	FMD	FPR	SW	FD_VSI	DESC_PROF	DPP	PC	DTYP
	63 … 32	31							3 … 0



```
static const u8 ice_fdir_tcpv4_pkt[] = {
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
        0x00, 0x00, 0x00, 0x00, 0x08, 0x00, 0x45, 0x00,
        0x00, 0x28, 0x00, 0x01, 0x00, 0x00, 0x40, 0x06,
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
        0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x50, 0x00,
        0x20, 0x00, 0x00, 0x00, 0x00, 0x00
};
```

CVL Flow Director Enablement

How to Build FDIR rule table?

1. Create Profile
2. Program the input set value and table by FDIR TX queue.
   a) Get free queue descriptor
   b) Fill LAN+FDIR descriptor
   c) Prepare FDIR dummy packet
   d) Notify HW (Doorbell) and HW Write Back
3. Check programming status (Optional)
4. Check rule table space and size (Highly recommend)

CVL Flow Director Debug

How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

1. Check if Profile is created successfully and correctly for both control port and PF port.

Notes:
What's the difference between control port and main port in FDIR ?

Control port in FDIR is responsible for building rule table by programming dummy packet.
PF Port in FDIR is responsible for filtering packets by looking up rule table.

FDIR set rule steps

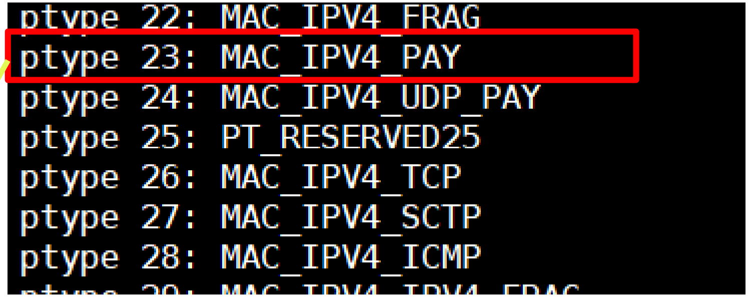
1. **Create Profile**
2. Set rule through Flow Director TX/RX queue
   a) Get free queue descriptor
   b) Fill LAN+FDIR descriptor
   c) Prepare FDIR dummy packet
   d) Notify HW (Doorbell) and HW Write Back
3. Check programming status (Optional)
4. Check rule table space and size (Highly recommend)

CVL Flow Director Debug

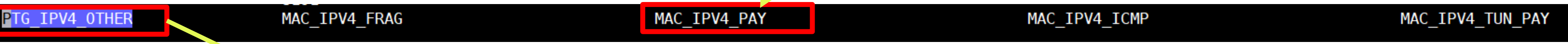
How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

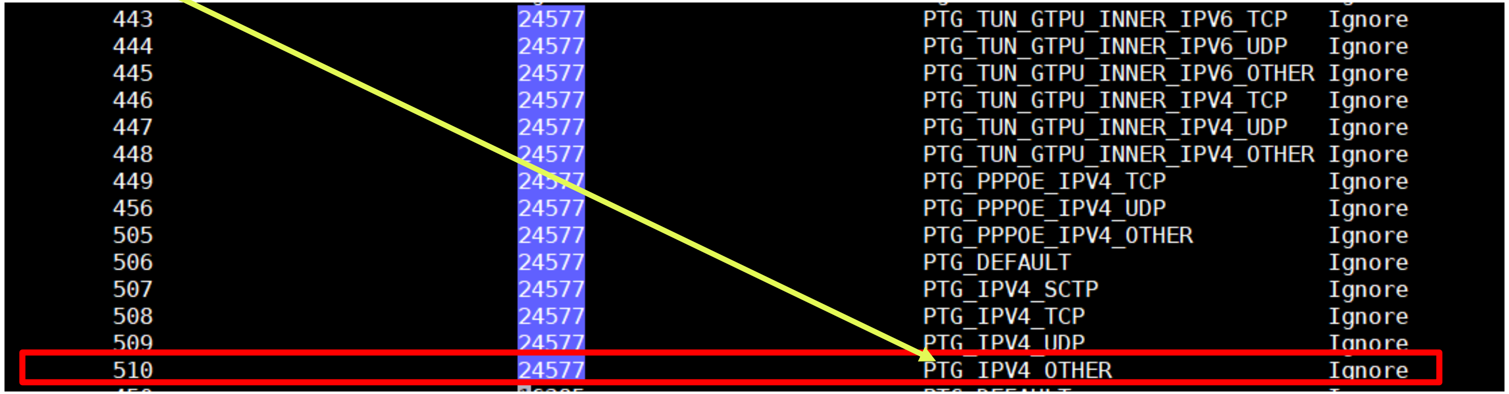
1. Check if Profile is created successfully and correctly: Bind correct ptype ?



```
ptype 22: MAC_IPV4_FRAG
ptype 23: MAC_IPV4_PAY
ptype 24: MAC_IPV4_UDP_PAY
ptype 25: PT_RESERVED25
ptype 26: MAC_IPV4_TCP
ptype 27: MAC_IPV4_SCTP
ptype 28: MAC_IPV4_ICMP
```



```
PTG_IPV4_OTHER          MAC_IPV4_FRAG           MAC_IPV4_PAY            MAC_IPV4_ICMP           MAC_IPV4_TUN_PAY
```



```
443        24577        PTG_TUN_GTPU_INNER_IPV6_TCP   Ignore
444        24577        PTG_TUN_GTPU_INNER_IPV6_UDP   Ignore
445        24577        PTG_TUN_GTPU_INNER_IPV6_OTHER Ignore
446        24577        PTG_TUN_GTPU_INNER_IPV4_TCP   Ignore
447        24577        PTG_TUN_GTPU_INNER_IPV4_UDP   Ignore
448        24577        PTG_TUN_GTPU_INNER_IPV4_OTHER Ignore
449        24577        PTG_PPPOE_IPV4_TCP            Ignore
456        24577        PTG_PPPOE_IPV4_UDP            Ignore
505        24577        PTG_PPPOE_IPV4_OTHER          Ignore
506        24577        PTG_DEFAULT                   Ignore
507        24577        PTG_IPV4_SCTP                 Ignore
508        24577        PTG_IPV4_TCP                  Ignore
509        24577        PTG_IPV4_UDP                  Ignore
510        24577        PTG_IPV4_OTHER                Ignore
```

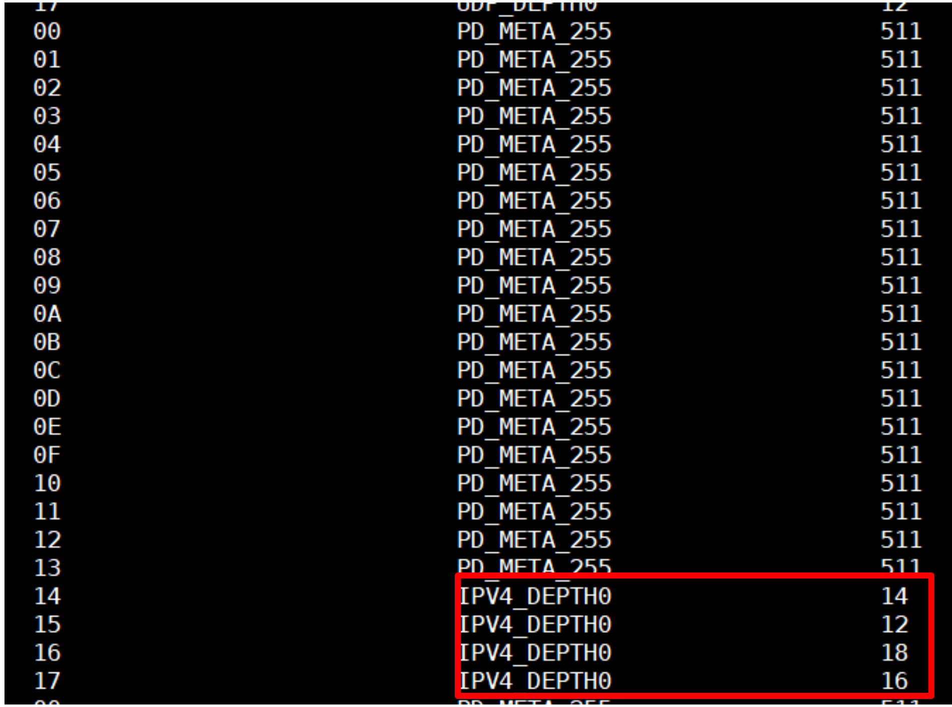
CVL Flow Director Debug

How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

1. Check if Profile is created successfully and correctly: Bind correct field vector ?

   Thanks for Zhang Qi's great package analysis utility !



```
00                             PD_META_255                    511
01                             PD_META_255                    511
02                             PD_META_255                    511
03                             PD_META_255                    511
04                             PD_META_255                    511
05                             PD_META_255                    511
06                             PD_META_255                    511
07                             PD_META_255                    511
08                             PD_META_255                    511
09                             PD_META_255                    511
0A                             PD_META_255                    511
0B                             PD_META_255                    511
0C                             PD_META_255                    511
0D                             PD_META_255                    511
0E                             PD_META_255                    511
0F                             PD_META_255                    511
10                             PD_META_255                    511
11                             PD_META_255                    511
12                             PD_META_255                    511
13                             PD_META_255                    511
14                             IPV4_DEPTH0                    14
15                             IPV4_DEPTH0                    12
16                             IPV4_DEPTH0                    18
17                             IPV4_DEPTH0                    16
```

CVL Flow Director Debug

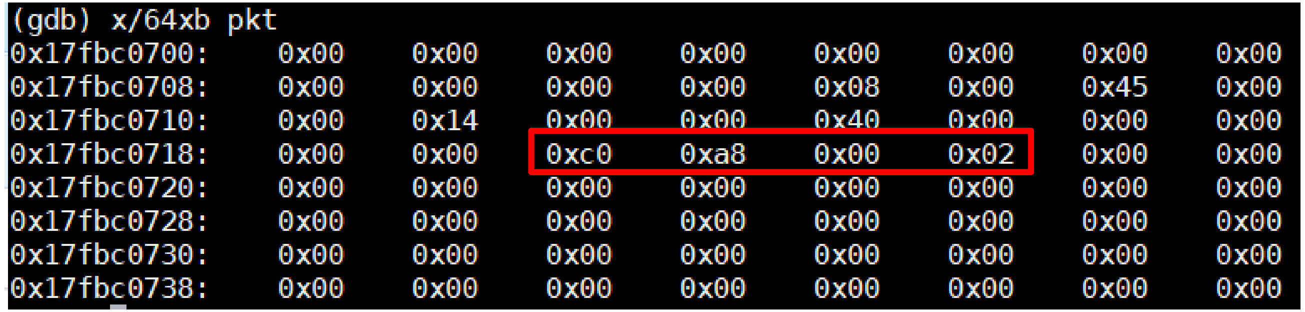
How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

1. Check if Profile is created successfully and correctly
2. Check if dummy packet contains the correct training data

FDIR set rule steps

1. Create Profile
2. **Set rule through Flow Director TX/RX queue**
   a) **Get free queue descriptor**
   b) **Fill LAN+FDIR descriptor**
   c) **Prepare FDIR dummy packet**



```
(gdb) x/64xb pkt
0x17fbc0700:    0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x17fbc0708:    0x00    0x00    0x00    0x00    0x08    0x00    0x45    0x00
0x17fbc0710:    0x00    0x14    0x00    0x00    0x40    0x00    0x00    0x00
0x17fbc0718:    0x00    0x00    0xc0    0xa8    0x00    0x02    0x00    0x00
0x17fbc0720:    0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x17fbc0728:    0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x17fbc0730:    0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x17fbc0738:    0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
```

CVL Flow Director Debug

How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

1. Check if Profile is created successfully and correctly
2. Check if dummy packet contains the correct training data
3. HW Write Back mechanism can confirm that packet is transmitted successfully.
4. Programming status report mechanism can confirm that packet is transmitted successfully.

FDIR set rule steps

1. Create Profile
2. Set rule through Flow Director TX/RX queue
   a) Get free queue descriptor
   b) Fill LAN+FDIR descriptor
   c) Prepare FDIR dummy packet
   d) Notify HW (Doorbell) and HW Write Back
3. **Check programming status (Optional)**

CVL Flow Director Debug

How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

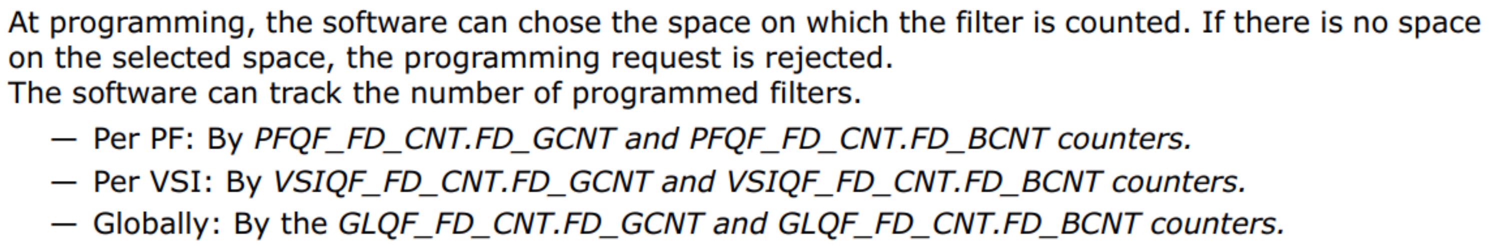
1. Check if Profile is created successfully and correctly
2. Check if dummy packet contains the correct training data
3. HW Write Back mechanism can confirm that packet is transmitted successfully.
4. Programming status report mechanism can confirm that packet is transmitted successfully.
5. Check Global/PF/VSI rule table space and size to make sure program succeed.

FDIR set rule steps

1. Create Profile
2. Set rule through Flow Director TX/RX queue
   a) Get free queue descriptor
   b) Fill LAN+FDIR descriptor
   c) Prepare FDIR dummy packet
   d) Notify HW (Doorbell) and HW Write Back
3. Check programming status (Optional)
4. **Check rule table space and size (Highly recommend)**

CVL Flow Director Debug

5. Check Global/PF/VSI rule table count increased to make sure program succeed.



At programming, the software can chose the space on which the filter is counted. If there is no space on the selected space, the programming request is rejected.
The software can track the number of programmed filters.

- Per PF: By *PFQF_FD_CNT.FD_GCNT and PFQF_FD_CNT.FD_BCNT counters.*
- Per VSI: By *VSIQF_FD_CNT.FD_GCNT and VSIQF_FD_CNT.FD_BCNT counters.*
- Globally: By the *GLQF_FD_CNT.FD_GCNT and GLQF_FD_CNT.FD_BCNT counters.*

Highly recommend:

- Important issue blocking DPDK PF FDIR POC and help a lot on IAVF FDIR POC.
- Almost necessary and sufficient condition of program success.
- Easy for debugging with little dependency on external tools.

CVL Flow Director Debug

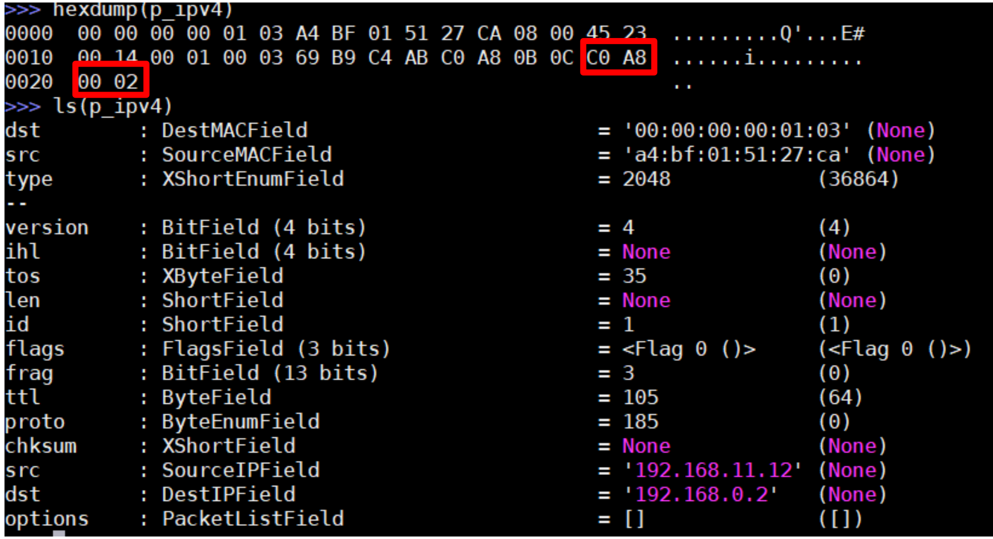
How to debug FDIR rule ?

(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end)

6. Send Test packets to LAN port and check if packets contains the desired input set value

E.g.

0xC0 0xA8 0x00 0x02→ 192.168.0.2



```
>>> hexdump(p_ipv4)
0000   00 00 00 00 01 03 A4 BF 01 51 27 CA 08 00 45 23  .........Q'...E#
0010   00 14 00 01 00 03 69 B9 C4 AB C0 A8 0B 0C C0 A8  ......i.........
0020   00 02                                            ..
>>> ls(p_ipv4)
dst        : DestMACField                   = '00:00:00:00:01:03' (None)
src        : SourceMACField                 = 'a4:bf:01:51:27:ca' (None)
type       : XShortEnumField                = 2048              (36864)
--
version    : BitField (4 bits)              = 4                 (4)
ihl        : BitField (4 bits)              = None              (None)
tos        : XByteField                     = 35                (0)
len        : ShortField                     = None              (None)
id         : ShortField                     = 1                 (1)
flags      : FlagsField (3 bits)            = <Flag 0 ()>       (<Flag 0 ()>)
frag       : BitField (13 bits)             = 3                 (0)
ttl        : ByteField                      = 105               (64)
proto      : ByteEnumField                  = 185               (0)
chksum     : XShortField                    = None              (None)
src        : SourceIPField                  = '192.168.11.12'   (None)
dst        : DestIPField                    = '192.168.0.2'     (None)
options    : PacketListField                = []                ([])
```

CVL Flow Director Debug

How to debug FDIR rule ?

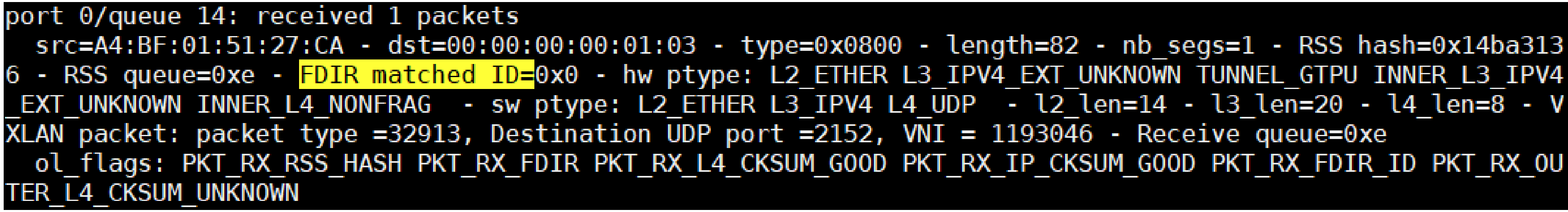
(E.g. pattern eth / ipv4 dst is 192.168.0.2 / end destinated to queue 14)

6. Check if Test LAN packets contains the desired input set value

7. Check LAN rx packets descriptor and FDIR domain flags, For dpdk, you can see:

Received packet with **FDIR matched ID=XXX** is directed to queue 14

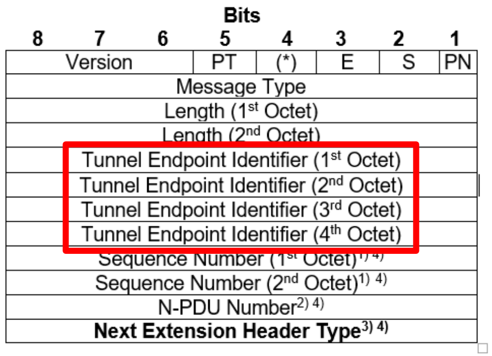
Notes:
Checking Queue ID is not the most reliable way since packets may be missed by FDIR and then distributed by RSS to the same queue index.
Please check FDIR flags like “FDIR matched”



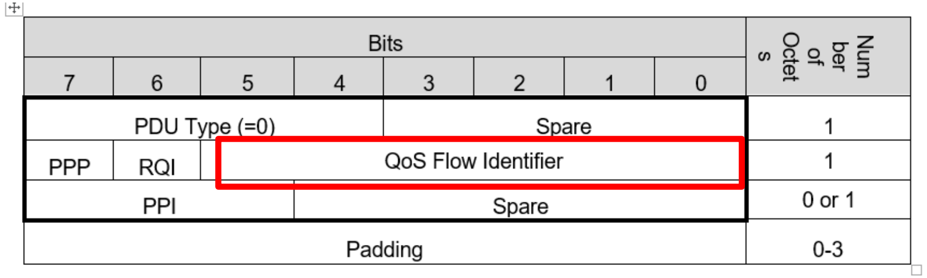
```
port 0/queue 14: received 1 packets
  src=A4:BF:01:51:27:CA - dst=00:00:00:00:01:03 - type=0x0800 - length=82 - nb_segs=1 - RSS hash=0x14ba313
6 - RSS queue=0xe - FDIR matched ID=0x0 - hw ptype: L2_ETHER L3_IPV4_EXT_UNKNOWN TUNNEL_GTPU INNER_L3_IPV4
_EXT_UNKNOWN INNER_L4_NONFRAG  - sw ptype: L2_ETHER L3_IPV4 L4_UDP  - l2_len=14 - l3_len=20 - l4_len=8 - V
XLAN packet: packet type =32913, Destination UDP port =2152, VNI = 1193046 - Receive queue=0xe
  ol_flags: PKT_RX_RSS_HASH PKT_RX_FDIR PKT_RX_L4_CKSUM_GOOD PKT_RX_IP_CKSUM_GOOD PKT_RX_FDIR_ID PKT_RX_OU
TER_L4_CKSUM_UNKNOWN
```

FDIR Intro

- **Use case:**
  PPPoE, GTP-U, etc.
  flow create 0 ingress pattern eth / ipv4 / udp / gtpu teid is 0x12345678 / gtp_psc qfi is 0x34 / end actions queue index 4 / end



Bits							
8	7	6	5	4	3	2	1
Version			PT	(*)	E	S	PN
Message Type							
Length (1st Octet)							
Length (2nd Octet)							
Tunnel Endpoint Identifier (1st Octet)							
Tunnel Endpoint Identifier (2nd Octet)							
Tunnel Endpoint Identifier (3rd Octet)							
Tunnel Endpoint Identifier (4th Octet)							
Sequence Number (1st Octet)1) 4)							
Sequence Number (2nd Octet)1) 4)							
N-PDU Number2) 4)							
Next Extension Header Type3) 4)							



Bits								Number of Octets
7	6	5	4	3	2	1	0	
PDU Type (=0)				Spare				1
PPP	RQI	QoS Flow Identifier						1
PPI			Spare					0 or 1
Padding								0-3

FDIR Use Case

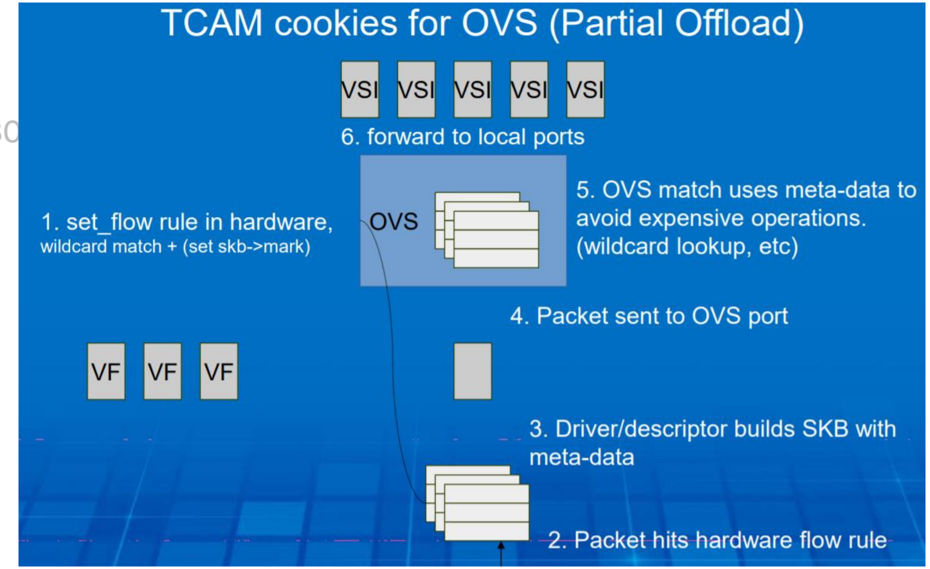
看一下。

- Improve Memcached Performance: https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/intel-ethernet-flow-director.pdf
- Impove Redis Performance: https://www.intel.com/content/www/us/en/architecture-and-technology/ethernet/application-device-queues-with-redis-brief.html

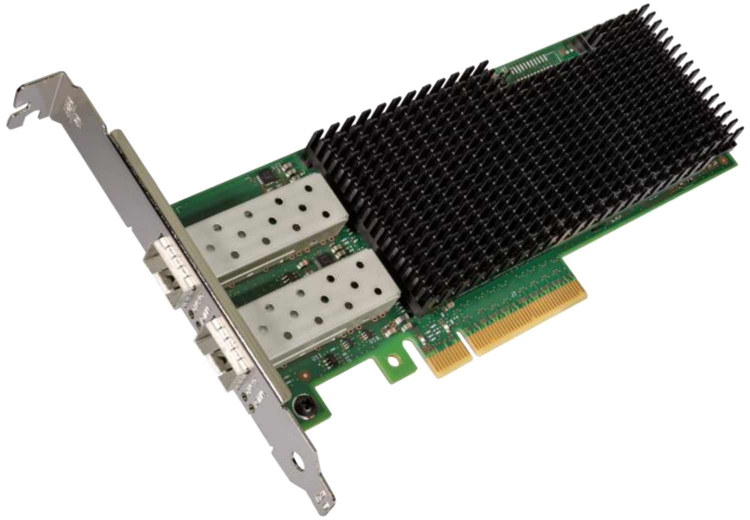
FDIR Use Case

- Use case:
  OVS Partial Offload:

1. Set_flow rule in HW
   flow create 0 ingress pattern eth / ipv4 / udp dst is 80 / end actions mark id 0x1234 end
2. Packet hits hw flow rule and mark desc with FD_ID field set to 0x1234
3. Driver builds SKB with marks in step 2
4. Packet sent to OVS Port
5. OVS match uses mark id 0x1234 to avoid expensive lookup
6. Forward to local ports


TCAM cookies for OVS (Partial Offload)
VSI
VSI
VSI
VSI
VSI
6. forward to local ports
OVS
5. OVS match uses meta-data to avoid expensive operations. (wildcard lookup, etc)
1. set_flow rule in hardware, wildcard match + (set skb->mark)
4. Packet sent to OVS port
VF
VF
VF
3. Driver/descriptor builds SKB with meta-data
2. Packet hits hardware flow rule

Agenda



Virtual Channel Intro

- What is Virtual Channel:

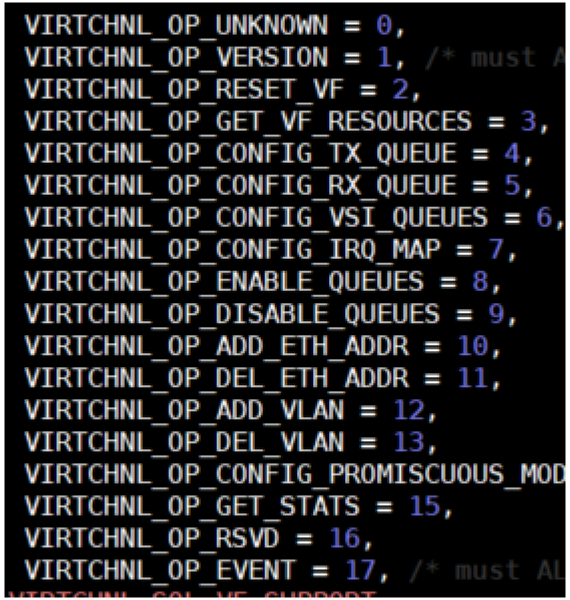
  VF PF virtual communication channel

avf channel分为base mode和advance mode。

- Why Virtual Channel?
  - Resource negotiation: RSS, VLAN, MAC, QUEUE, IRQ…
  - HW management: RESET, EVENT…

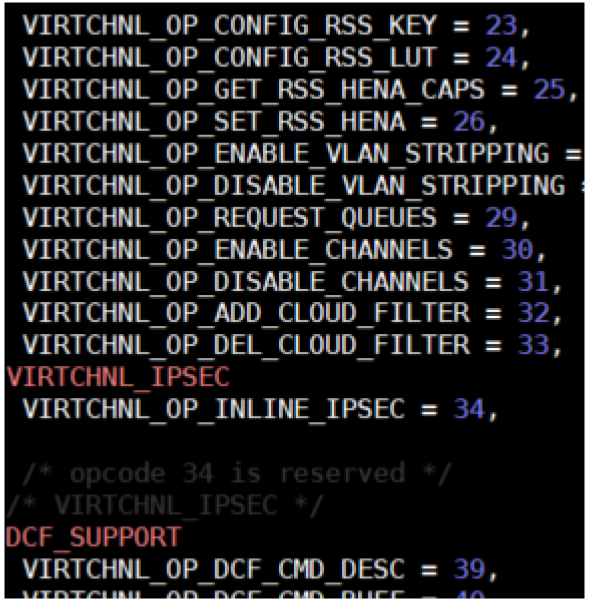
Virtual Channel Intro

➤ Supported Virtual Channel Command:



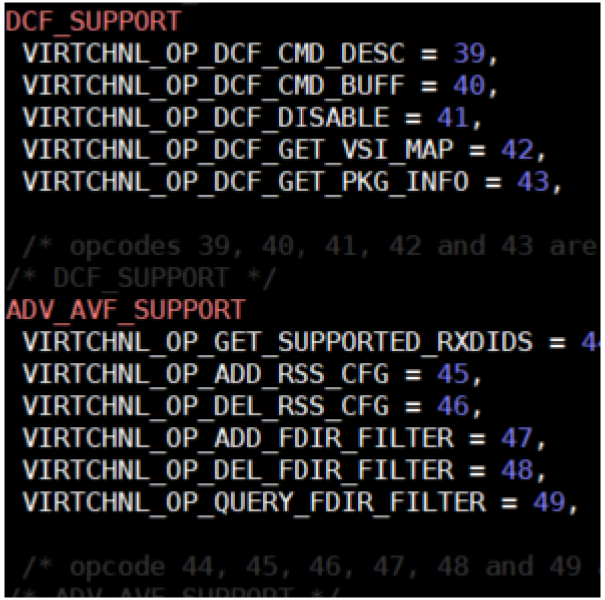
```
 VIRTCHNL_OP_UNKNOWN = 0,
 VIRTCHNL_OP_VERSION = 1, /* must A
 VIRTCHNL_OP_RESET_VF = 2,
 VIRTCHNL_OP_GET_VF_RESOURCES = 3,
 VIRTCHNL_OP_CONFIG_TX_QUEUE = 4,
 VIRTCHNL_OP_CONFIG_RX_QUEUE = 5,
 VIRTCHNL_OP_CONFIG_VSI_QUEUES = 6,
 VIRTCHNL_OP_CONFIG_IRQ_MAP = 7,
 VIRTCHNL_OP_ENABLE_QUEUES = 8,
 VIRTCHNL_OP_DISABLE_QUEUES = 9,
 VIRTCHNL_OP_ADD_ETH_ADDR = 10,
 VIRTCHNL_OP_DEL_ETH_ADDR = 11,
 VIRTCHNL_OP_ADD_VLAN = 12,
 VIRTCHNL_OP_DEL_VLAN = 13,
 VIRTCHNL_OP_CONFIG_PROMISCUOUS_MOD
 VIRTCHNL_OP_GET_STATS = 15,
 VIRTCHNL_OP_RSVD = 16,
 VIRTCHNL_OP_EVENT = 17, /* must AL
```

base mode



```
 VIRTCHNL_OP_CONFIG_RSS_KEY = 23,
 VIRTCHNL_OP_CONFIG_RSS_LUT = 24,
 VIRTCHNL_OP_GET_RSS_HENA_CAPS = 25,
 VIRTCHNL_OP_SET_RSS_HENA = 26,
 VIRTCHNL_OP_ENABLE_VLAN_STRIPPING =
 VIRTCHNL_OP_DISABLE_VLAN_STRIPPING
 VIRTCHNL_OP_REQUEST_QUEUES = 29,
 VIRTCHNL_OP_ENABLE_CHANNELS = 30,
 VIRTCHNL_OP_DISABLE_CHANNELS = 31,
 VIRTCHNL_OP_ADD_CLOUD_FILTER = 32,
 VIRTCHNL_OP_DEL_CLOUD_FILTER = 33,
VIRTCHNL_IPSEC
 VIRTCHNL_OP_INLINE_IPSEC = 34,

 /* opcode 34 is reserved */
/* VIRTCHNL_IPSEC */
DCF_SUPPORT
 VIRTCHNL_OP_DCF_CMD_DESC = 39,
```



```
DCF_SUPPORT
 VIRTCHNL_OP_DCF_CMD_DESC = 39,
 VIRTCHNL_OP_DCF_CMD_BUFF = 40,
 VIRTCHNL_OP_DCF_DISABLE = 41,
 VIRTCHNL_OP_DCF_GET_VSI_MAP = 42,
 VIRTCHNL_OP_DCF_GET_PKG_INFO = 43,

 /* opcodes 39, 40, 41, 42 and 43 are
/* DCF_SUPPORT */
ADV_AVF_SUPPORT
 VIRTCHNL_OP_GET_SUPPORTED_RXDIDS = 4
 VIRTCHNL_OP_ADD_RSS_CFG = 45,
 VIRTCHNL_OP_DEL_RSS_CFG = 46,
 VIRTCHNL_OP_ADD_FDIR_FILTER = 47,
 VIRTCHNL_OP_DEL_FDIR_FILTER = 48,
 VIRTCHNL_OP_QUERY_FDIR_FILTER = 49,

 /* opcode 44, 45, 46, 47, 48 and 49
```

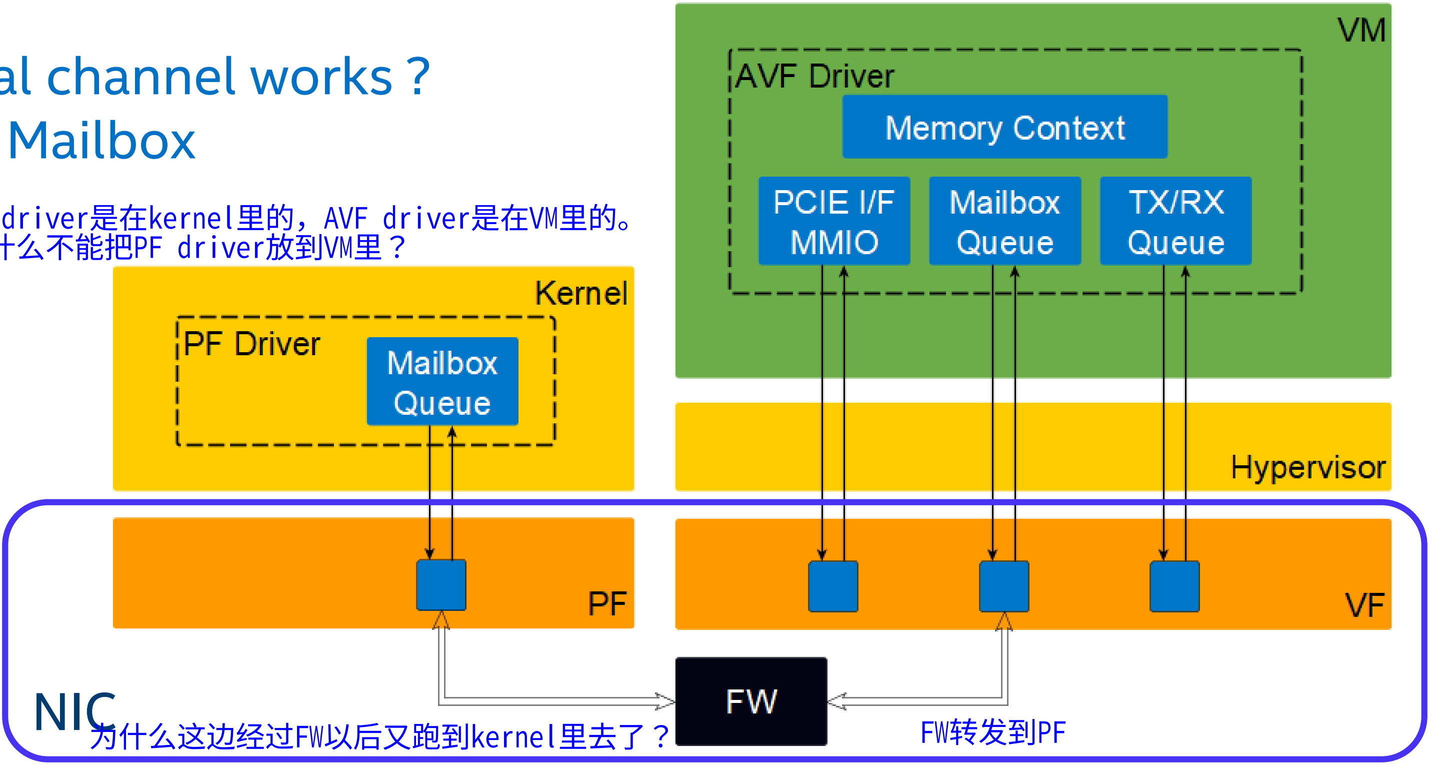
DCF，AVF相关（添加RSS config，添加flow director filter）

Virtual Channel Intro

和PF通信的三种方式：PCIE MMIO, Mailbox, TX/RX

- ➢ How virtual channel works？
  Hardware Mailbox

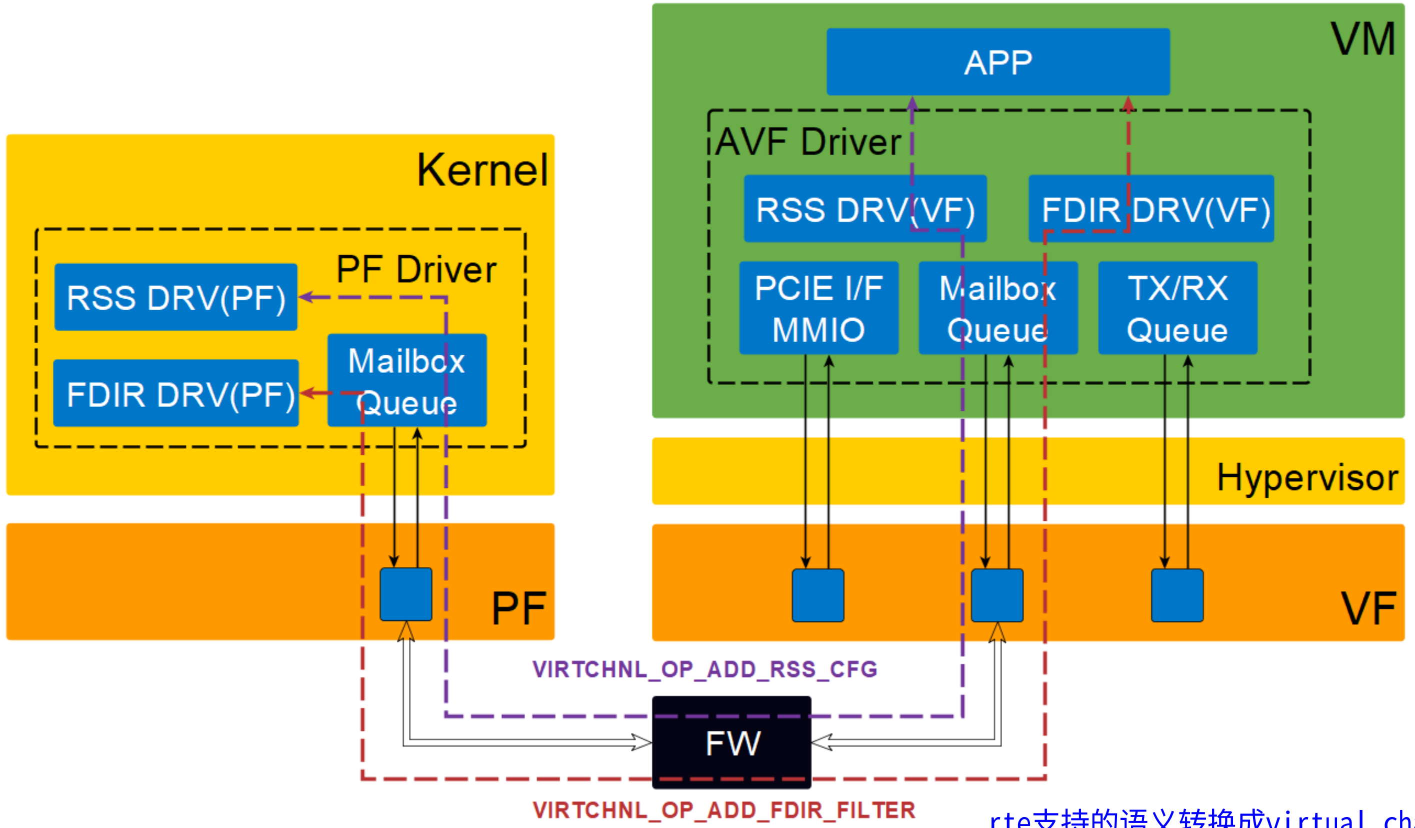
PF driver是在kernel里的，AVF driver是在VM里的。
为什么不能把PF driver放到VM里？



为什么这边经过FW以后又跑到kernel里去了？

FW转发到PF

Virtchnl for RSS/FDIR design



hackson: virtual channel的workload

rte支持的语义转换成virtual channel支持的格式。

Q & A