XDP Workshop

Netdev conference 0x17

Agenda

- XDP QAT (Zhan Xue) – 15 mins
- XDP: Past, present and future (Toke Høiland-Jørgensen) – 15 mins
- XDP offloads status (Stanislav Fomichev) – 15 mins
- AF_XDP virtio_net support (Xuan Zhuo) – 10 mins

XDP Inline Accelerator QAT

Zhan Xue

Agenda

- Why XDP Inline QAT for Crypto
- XDP Inline Accelerator Overview
- Performance
  - QAT XDP V.S. QAT LKCF
  - XDP Drop Post Decryption
  - QAT XDP Inline AF_XDP
- Discussion for Redirection to Accelerator

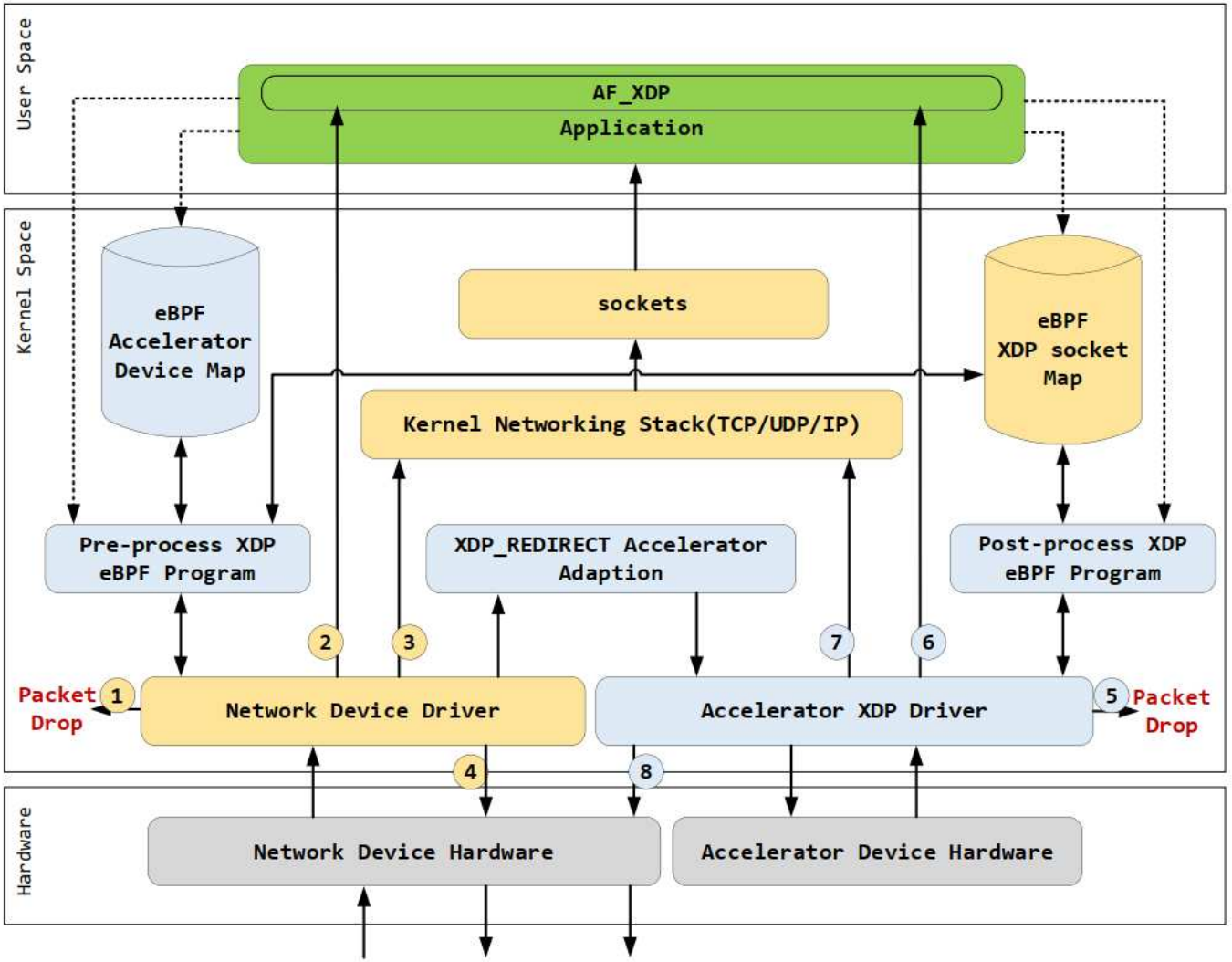
Why XDP Inline QAT for Crypto

- In pursuit of security and privacy, the encrypted packets are increasing quickly in cloud and edge networks. It brings significant challenges to eBPF/XDP which works on the plaintext packet data.

- Given XDP runs before packet data touched by kernel networking stack, it has no correlation to the existing kernel network cryptography framework.

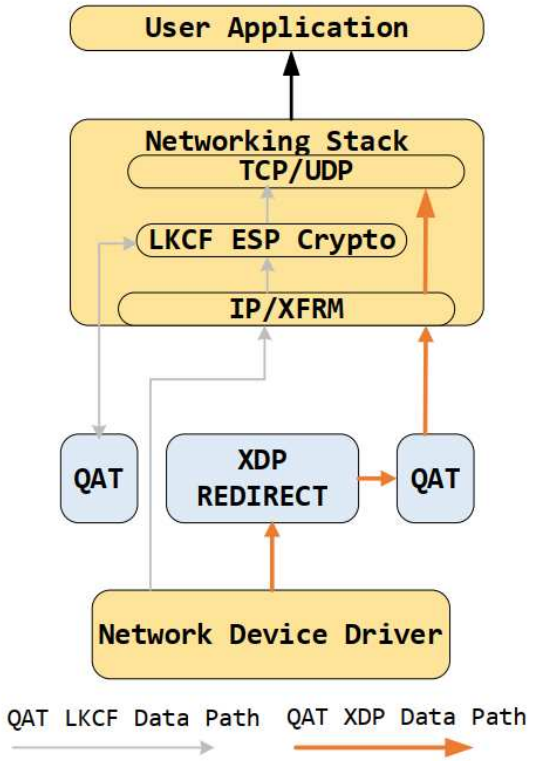
- An attempt of introducing the hardware-based accelerator (QAT) to enable and accelerate inline crypto in XDP layer could be an option to unlock the capabilities of XDP on encrypted packet.

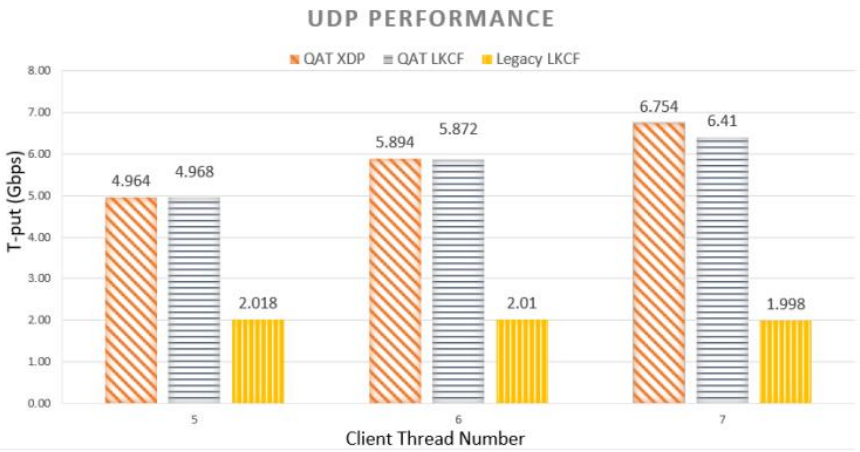
XDP Inline Accelerator Overview



Performance: QAT XDP V.S. QAT LKCF

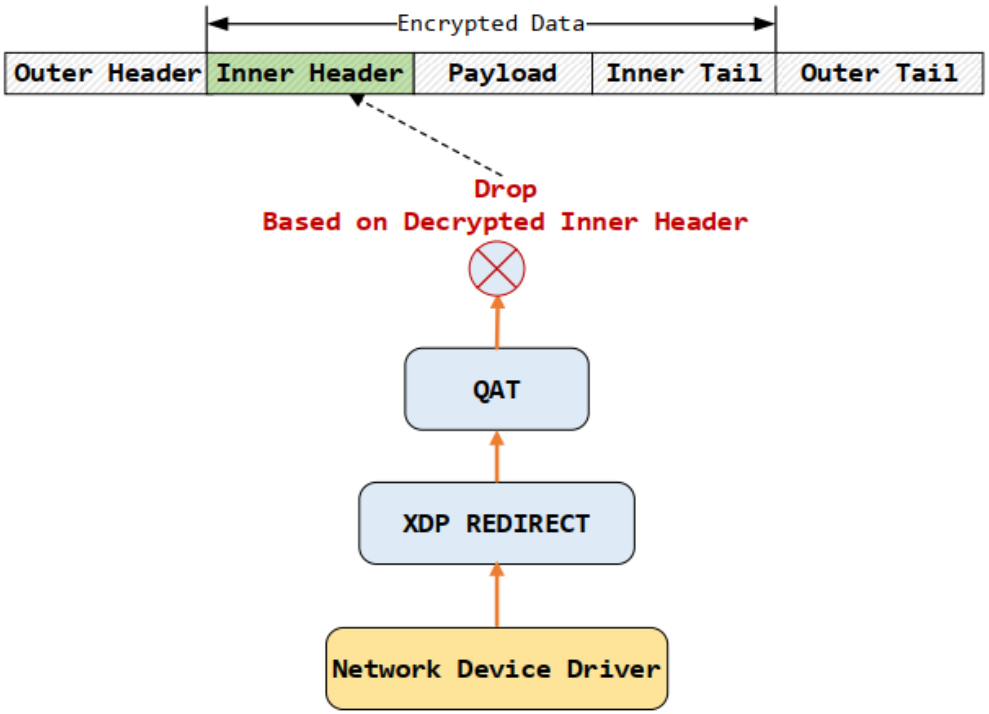
- End-to-End IPsec Decryption with Kernel Networking Stack (AES-CBC 128 and SHA1, MTU 1500, UDP payload length 1386 Bytes).





Performance: XDP Drop Post Decryption

- IPsec Tunnel mode, UDP payload length 1386 Bytes , 1 QAT instance
  - Packet loss rate <= 1%
  - XDP Drop Rate Post Decryption: > 13 Gbps



Performance: QAT XDP Inline AF_XDP

- Pattern 1: All the NIC and QAT related operations are on the same CPU core.
- Pattern 2: Load balancing of QAT enqueue operation to a separate CPU core via a software FIFO. Use different cores to process NIC and QAT related operations.

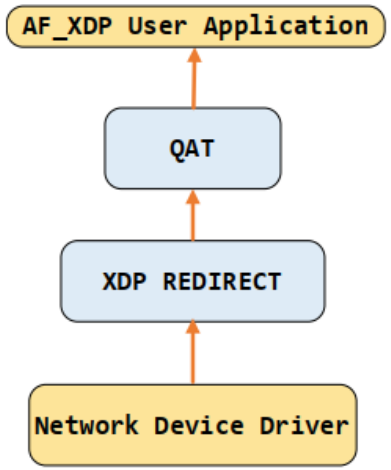
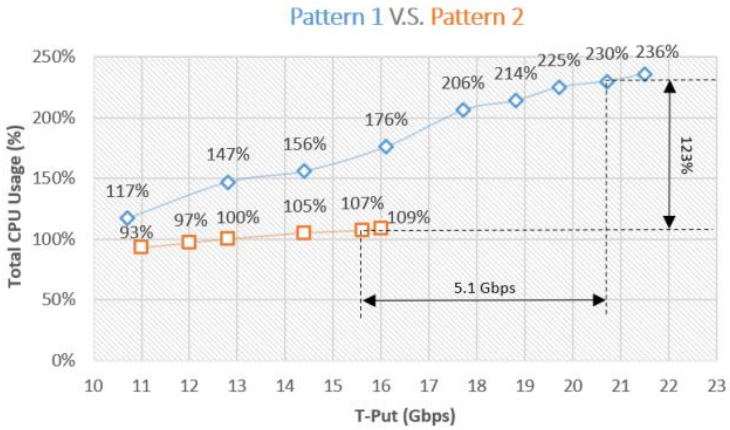


TABLE I. TEST RESULT OF PATTERN 1

Core1 Usage	*Core2 Usage*	*Total Usage*	*T-Put (Gbps)*	*T-Put (Kpps)*	*Packet Loss Rate*
82%	11%	93%	11	1000	< 0.1%
87%	10%	97%	12	1082	< 0.1%
90%	10%	100%	12.8	1189	< 0.1%
95%	10%	105%	14.4	1300	< 0.1%
98%	9%	107%	15.6	1400	< 1%
100%	9%	109%	16.0	1400	2%

TABLE II. TEST RESULT OF PATTERN 2

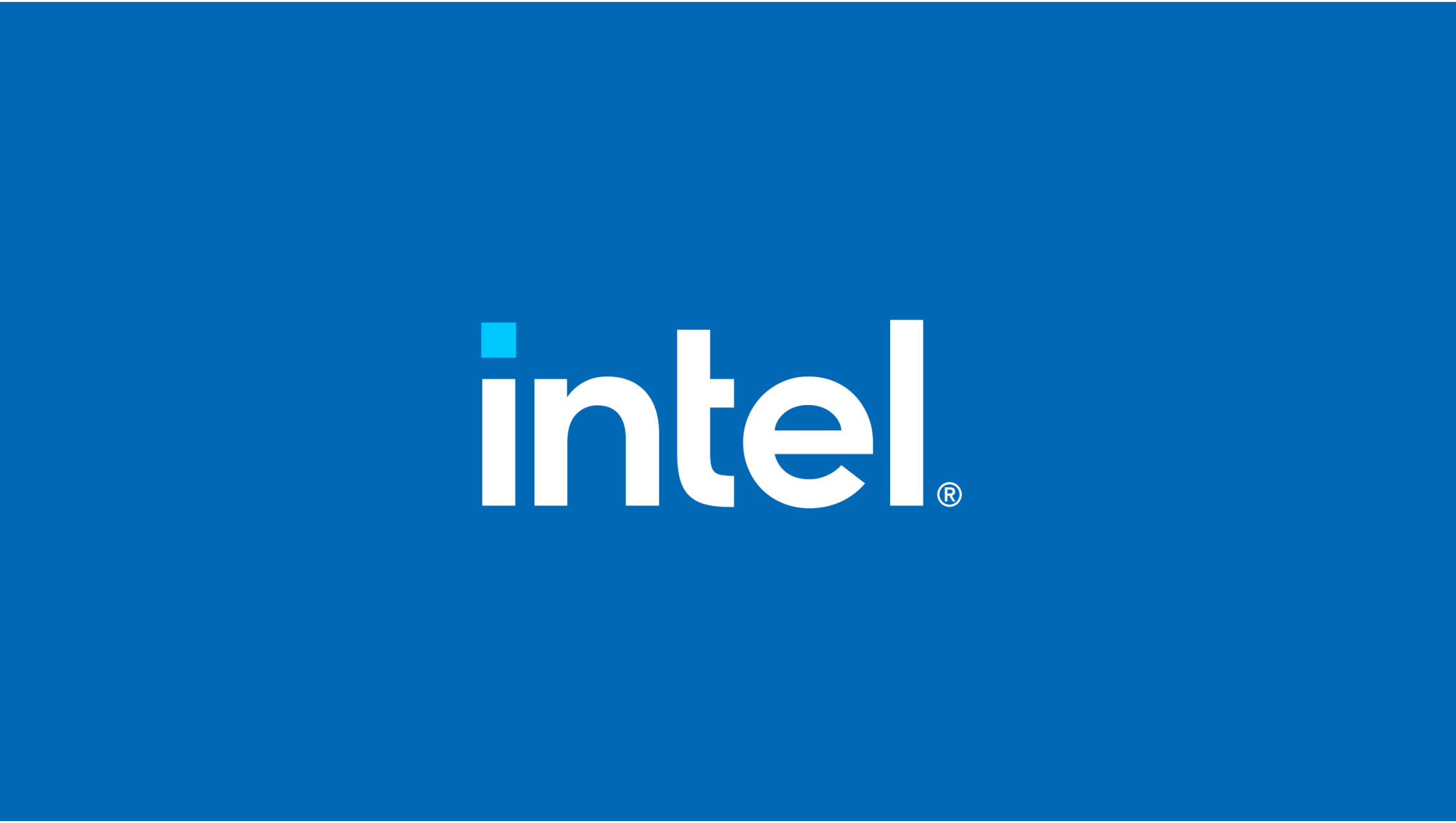
Core1 Usage	*Core2 Usage*	*Core3 Usage*	*Core4 Usage*	*Total Usage*	*T-Put (Gbps)*	*T-Put (Kpps)*	*Packet Loss Rate*
27%	46%	35%	9%	117%	10.7	964	0.002%
32%	60%	43%	12%	147%	12.8	1150	0.041%
32%	64%	48%	12%	156%	14.4	1298	0.02%
34%	77%	54%	11%	176%	16.1	1450	0.056%
39%	90%	63%	14%	206%	17.7	1605	0.047%
41%	96%	67%	10%	214%	18.8	1720	0.06%
44%	99%	71%	11%	225%	19.7	1800	0.09%
46%	100%	72%	12%	230%	20.7	1870	0.85%
47%	100%	73%	16%	236%	21.5	1903	2.3%



Discussion for Redirection to Accelerator

- Methods of redirecting packet to accelerator: XDP_REDIRECT and Kfuncs

Redirection Method	XDP_REDIRECT	Kfuncs
Pros.	- General framework for different accelerators. - Easy extension: add acceldev based on existing devmap. - Batch operation for performance. - Unified accelerators management.	- Less impact to kernel code due to specific implementation within accelerator itself. - Custom solution with quick turnaround.
Cons.	- Efforts of acceldev map implementation in kernel. - Efforts of redirection adaption in kernel.	- Lack of scalability and flexibility, duplicate efforts for different accelerators. - Performance loss without batch operation.


intel

XDP: Past, Present and Future

Toke Høiland-Jørgensen
Red Hat

XDP: Recently introduced features

- Multibuf (Lozenzo, Maciej, Eelco)
- XDP hints on RX (Jesper, Stanislav, Toke)
- Feature flags (Lorenzo)
- More stack helpers (conntrack, synproxy) (Kartikeya, Lorenzo, Maxim)
- XDP_REDIRECT improvements
  - Multicast (Hangbin)
  - Map lookup, hashmap type, bpf_redirect() performance (Toke)
  - Programs in devmap/cpumap (David Ahern, Lorenzo)
- Eliminating indirect calls (Björn)
- Live mode BPF_PROG_RUN (Toke)
- AF_XDP need_wakeup mode (Magnus and Maxim)
- Atomic replace and bpf_link attachment (Toke, Andrii)

XDP driver support

	Basic	Redirect	ndo_xmit	Multibuf RX	Multibuf TX	XSK ZC	HW offload
atlantic	✓	✓	✓	✓	✓		
bnxt	✓	✓	(✓)	✓	(✓)		
cpsw	✓	✓	✓				
cpsw_new	✓	✓	✓				
dpaa	✓	✓	✓				
dpaa2	✓	✓	✓			✓	
ena	(✓)	(✓)					
enetc	✓	✓	✓	✓	✓		
fec	(✓)	(✓)					
funeth	✓	✓	(✓)		(✓)		
gve	(✓)	(✓)	(✓)		(✓)		
hv_netvsc	✓	✓	✓				
i40e	✓	✓	(✓)	✓	(✓)	✓	
ice	✓	✓	(✓)	✓	(✓)	✓	
igb	✓	✓	(✓)		(✓)		
igc	✓	✓	(✓)		(✓)	✓	
ixgbe	✓	✓	(✓)		(✓)	✓	
ixgbevf	✓						
lan966x	(✓)	(✓)	(✓)				
mana	✓	✓	✓				

	Basic	Redirect	ndo_xmit	Multibuf RX	Multibuf TX	XSK ZC	HW offload
mlx4	✓	✓					
mlx5	(✓)	(✓)	(✓)	(✓)	(✓)	(✓)	
mtk_eth	(✓)	(✓)	(✓)		(✓)		
mvneta	✓	✓	✓	✓	✓		
mvpp2	(✓)	(✓)	(✓)				
netsec	✓	✓	✓				
nfp	✓					(✓)	(✓)
octeontx2	✓	✓	(✓)				
qede	✓	✓	✓				
sfc	✓	✓	✓				
sfc-siena	✓	✓	✓				
stmmac	✓	✓	(✓)			✓	
thunder	(✓)						
tsnep	✓	✓	✓		✓	✓	
tun	(✓)	(✓)	(✓)				
veth	(✓)	(✓)	(✓)	(✓)	(✓)		
virtio_net	✓	✓	(✓)	(✓)	(✓)		
vmxnet3	✓	✓	✓				
xen-netfront	✓	✓	✓				

✓: Always enabled. (✓): Configuration-dependent.

XDP: Ongoing work

- XDP hints on TX (Stanislav)
- Veth optimisations (Jesper)
- Multiprog attachment (Daniel)
- Queueing (Toke)

Generic XDP

- Bulking on redirect
- GSO frames through multibuf API
- Veth redirect conversion

AF_XDP xmit path

- Getting rid of socket-allocated SKBs?
- Why allocate SKBs at all?
- TX hints (ongoing, Stanislav)

XDP Hints

- More metadata fields
- Saving for redirect and skb creation (in xdp_frame)
- Naming: “XDP metadata” is not very googlable!

Datapath helpers

What do we need to build a transparent fast path using XDP?

- Bridge lookup(?)
- Netfilter / flowtables lookup (or acceleration?)

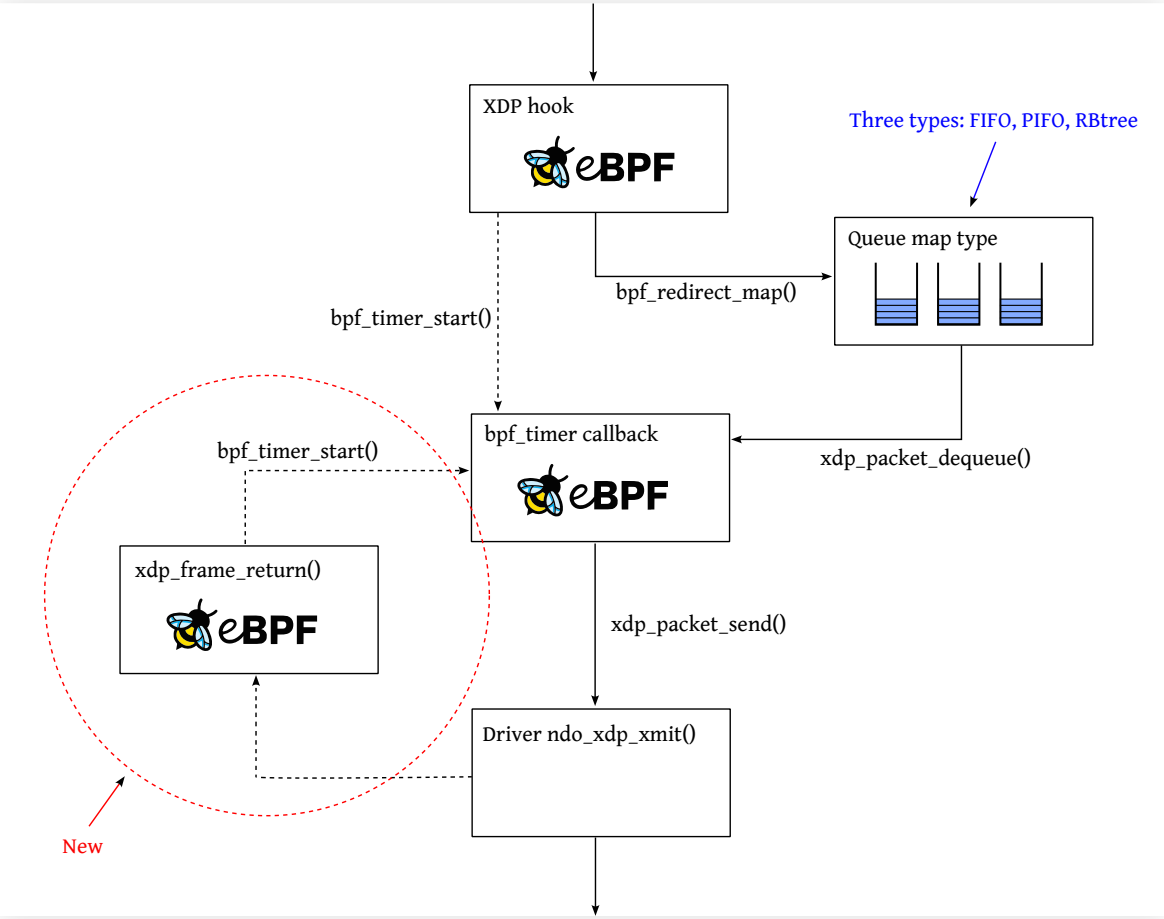
XDP queueing

Last presented at LPC 2022: https://lpc.events/event/16/contributions/1351/

Progress stalled a bit since then, but currently working on:

- Rebase on top of bpf-next in progress (convert to kfuncs, dynptr)
- Finding a way to control HW queue depth (like BQL)

XDP queueing - high-level diagram



XDP queueing - outstanding issues

- Context of the callback function
  - Softirq (which?) or move to kthread (like cpumap)?
  - Minimising overhead of callback
- Validating HWQ depth limiting approach
- Handling competition with netstack?
- Which type(s) of queue map are really useful?
- Finish rebase, lots more validation and testing

XDP Workshop

Stanislav Fomichev

Next offloads to support?

- scatter-gather - done
- rx/tx checksum - almost done
- departure-time - have patches from Intel's Song Yoong Siang
- tso?
  - saw concerns with "userspace tcp", how does the community feel about this overall?
- crypto offloads
  - psp if/when it lands

How to incentivize vendors to support all offloads?

- xdp_hw_metadata as a qualification tool? with pass/non-pass output?
- netdev is working on ci, will those have vendor nics in them?
- would be useful to have test stends with vendor hw to run the patches against
  - if the ci runs xdp_hw_metadata, enthusiasts can maybe work on the offloads?

ALIBABA CLOUD
INTELLIGENCE GROUP

Virtio-Net & XDP/AF-XDP

Anolis and Alibaba cloud create the best XDP practices on cloud

Xuan Zhuo（丁雪峰）

ALIBABA CLOUD
INTELLIGENCE GROUP

Difficulties of supporting XDP inside virtio-net

01

Virtio-net did not support queue reset

- AF-XDP needs this

02

Virtio-net supports rx partial csum

- XDP can not handle partial rx csum

03

The number of virtio-net queues is fixed

- No more free queues for xdp tx

Virtio-net did not support queue reset

AF-XDP needs this

ALIBABA CLOUD INTELLIGENCE GROUP

Virtqueue reset is an important ability to support AF_XDP.

Most NICs support this. So we introduce this feature to the virtio spec 1.2.

Then if the virtio-net device supports VIRTIO_F_RING_RESET, the driver can support the AF_XDP.

```
commit a4ce81a8378066cfcec6ca98b18640622a8f5ffc
Author: Xuan Zhuo <xuanzhuo@linux.alibaba.com>
Date:   Mon Nov 8 14:22:43 2021 +0800

    virtio: mmio support virtqueue reset

    mmio support virtqueue reset.

    MMIO Device Register Layout "QueueReady" to support virtqueue reset.
    The driver uses this to selectively reset the queue.

    Fixes: https://github.com/oasis-tcs/virtio-spec/issues/124
    Reviewed-by: Jason Wang <jasowang@redhat.com>
    Signed-off-by: Xuan Zhuo <xuanzhuo@linux.alibaba.com>
    Signed-off-by: Cornelia Huck <cohuck@redhat.com>

commit 12998e73862186d4c9e949ae542645040e33de2b
Author: Xuan Zhuo <xuanzhuo@linux.alibaba.com>
Date:   Mon Nov 8 14:22:42 2021 +0800

    virtio: pci support virtqueue reset

    PCI support virtqueue reset.

    virtio_pci_common_cfg add "queue_reset" to support virtqueue reset.
    The driver uses this to selectively reset the queue.

    Fixes: https://github.com/oasis-tcs/virtio-spec/issues/124
    Reviewed-by: Jason Wang <jasowang@redhat.com>
    Signed-off-by: Xuan Zhuo <xuanzhuo@linux.alibaba.com>
    Signed-off-by: Cornelia Huck <cohuck@redhat.com>

commit 3b5378d70a42dffb2cfc0cee619d40c4c5acd4c8
Author: Xuan Zhuo <xuanzhuo@linux.alibaba.com>
Date:   Mon Nov 8 14:22:41 2021 +0800

    virtio: introduce virtqueue reset as basic facility

    This patch allows the driver to reset a queue individually.

    This is very common on general network equipment. By disabling a queue,
    you can quickly reclaim the buffer currently on the queue. If necessary,
    we can reinitialize the queue separately.

    For example, when virtio-net implements support for AF_XDP, we need to
    disable a queue to release all the original buffers when AF_XDP setup.
    And quickly release all the AF_XDP buffers that have been placed in the
    queue when AF_XDP exits.

    Fixes: https://github.com/oasis-tcs/virtio-spec/issues/124
    Reviewed-by: Jason Wang <jasowang@redhat.com>
    Signed-off-by: Xuan Zhuo <xuanzhuo@linux.alibaba.com>
    Signed-off-by: Cornelia Huck <cohuck@redhat.com>
```

Virtio-net supports rx partial csum

XDP can not handle partial rx csum

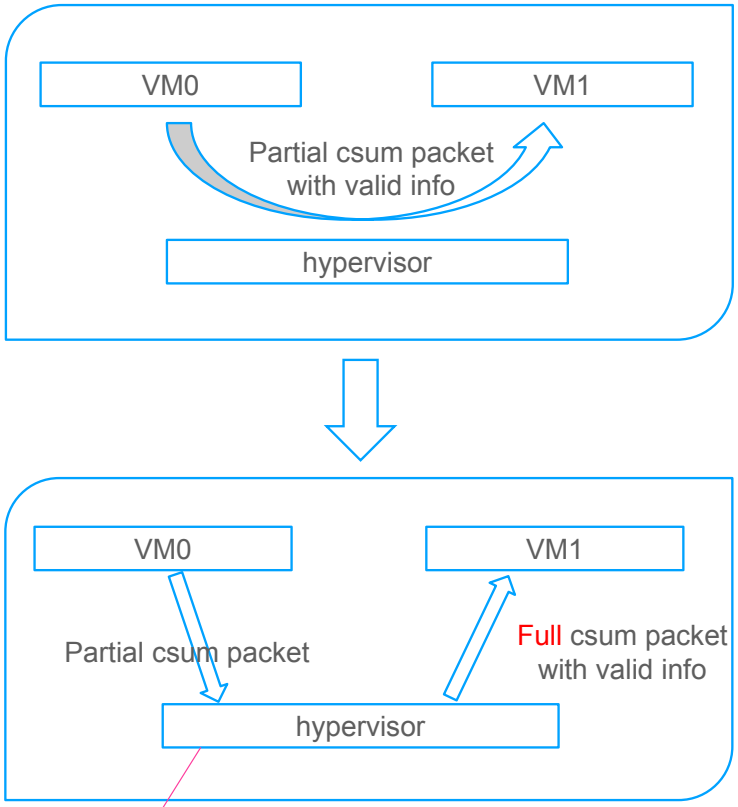
ALIBABA CLOUD INTELLIGENCE GROUP

Because the virtio-net can receive packet from the vm that works on the same host. So the hypervisor can transfer the packet without the full csum (for the sender the tx csum is offload to device) to other vm. @Heng Qi

Solution:

1. We try to do the csum inside the driver, but if we works with transfer that will be difficult.

2. We try to introduce a new feature to the virtio-net spec to let the device to calculate the csum. Then the driver will not receive a partial csum packet.



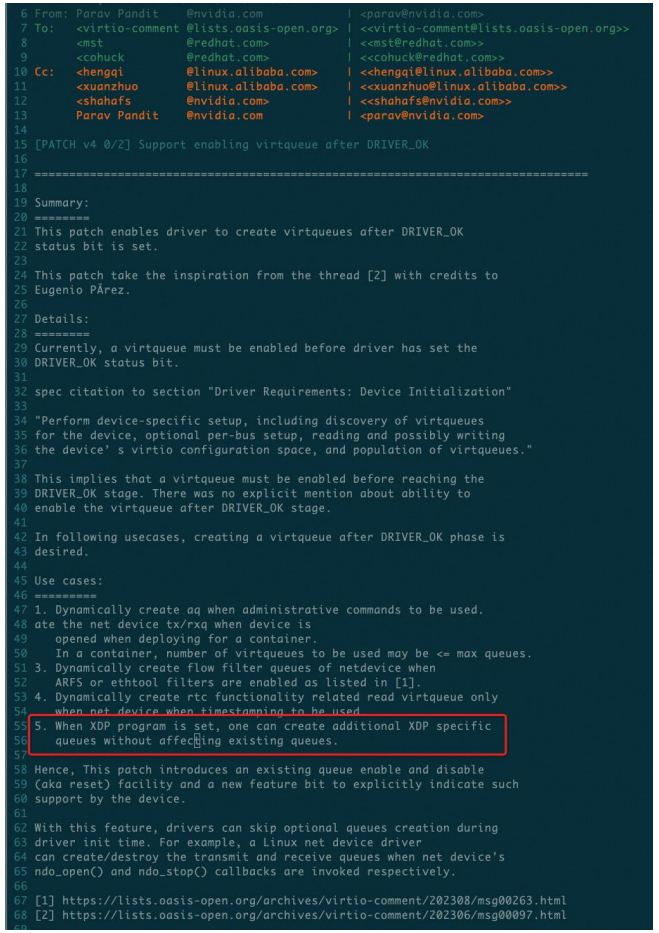
More devices have FPGA, calculating the csum does not occupy too much resource. The virtio-net can work more like the physical net card.

The number of virtio-net queues is fixed

No more free queues for xdp tx

ALIBABA CLOUD INTELLIGENCE GROUP

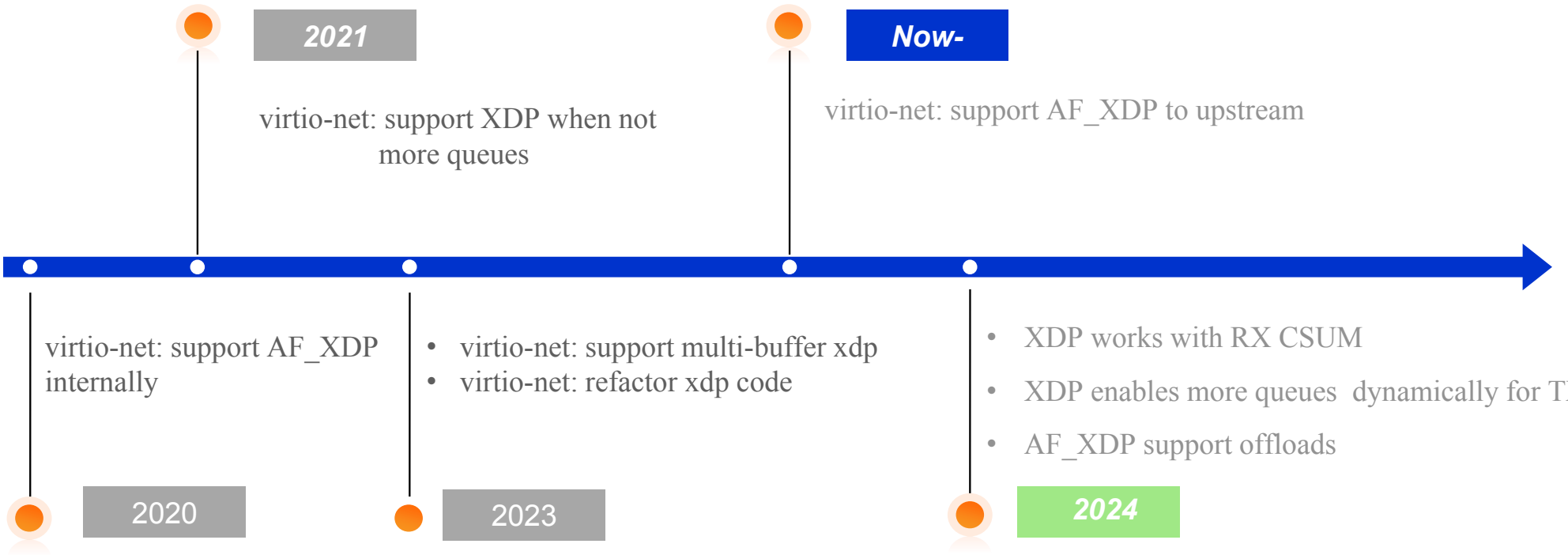
Now, Parav Pandit from Nvidia is introducing a new feature to virtio. We work with him.
If that successed, the virtio-net can create additional XDP specific queues to do xdp tx.



```
From: Parav Pandit    @nvidia.com           | <parav@nvidia.com>
To:   <virtio-comment @lists.oasis-open.org> | <<virtio-comment@lists.oasis-open.org>>
      <mst            @redhat.com>          | <<mst@redhat.com>>
      <cohuck         @redhat.com>          | <<cohuck@redhat.com>>
Cc:   <hengqi         @linux.alibaba.com>   | <<hengqi@linux.alibaba.com>>
      <xuanzhuo       @linux.alibaba.com>   | <<xuanzhuo@linux.alibaba.com>>
      <shahafs        @nvidia.com>          | <<shahafs@nvidia.com>>
      Parav Pandit    @nvidia.com           | <parav@nvidia.com>

[PATCH v4 0/2] Support enabling virtqueue after DRIVER_OK

================================================================================

Summary:
========
This patch enables driver to create virtqueues after DRIVER_OK
status bit is set.

This patch take the inspiration from the thread [2] with credits to
Eugenio PÃrez.

Details:
========
Currently, a virtqueue must be enabled before driver has set the
DRIVER_OK status bit.

spec citation to section "Driver Requirements: Device Initialization"

"Perform device-specific setup, including discovery of virtqueues
for the device, optional per-bus setup, reading and possibly writing
the device' s virtio configuration space, and population of virtqueues."

This implies that a virtqueue must be enabled before reaching the
DRIVER_OK stage. There was no explicit mention about ability to
enable the virtqueue after DRIVER_OK stage.

In following usecases, creating a virtqueue after DRIVER_OK phase is
desired.

Use cases:
=========
1. Dynamically create aq when administrative commands to be used.
ate the net device tx/rxq when device is
   opened when deploying for a container.
   In a container, number of virtqueues to be used may be <= max queues.
3. Dynamically create flow filter queues of netdevice when
   ARFS or ethtool filters are enabled as listed in [1].
4. Dynamically create rtc functionality related read virtqueue only
   when net device when timestamping to be used.
5. When XDP program is set, one can create additional XDP specific
   queues without affecting existing queues.

Hence, This patch introduces an existing queue enable and disable
(aka reset) facility and a new feature bit to explicitly indicate such
support by the device.

With this feature, drivers can skip optional queues creation during
driver init time. For example, a Linux net device driver
can create/destroy the transmit and receive queues when net device's
ndo_open() and ndo_stop() callbacks are invoked respectively.

[1] https://lists.oasis-open.org/archives/virtio-comment/202308/msg00263.html
[2] https://lists.oasis-open.org/archives/virtio-comment/202306/msg00097.html
```

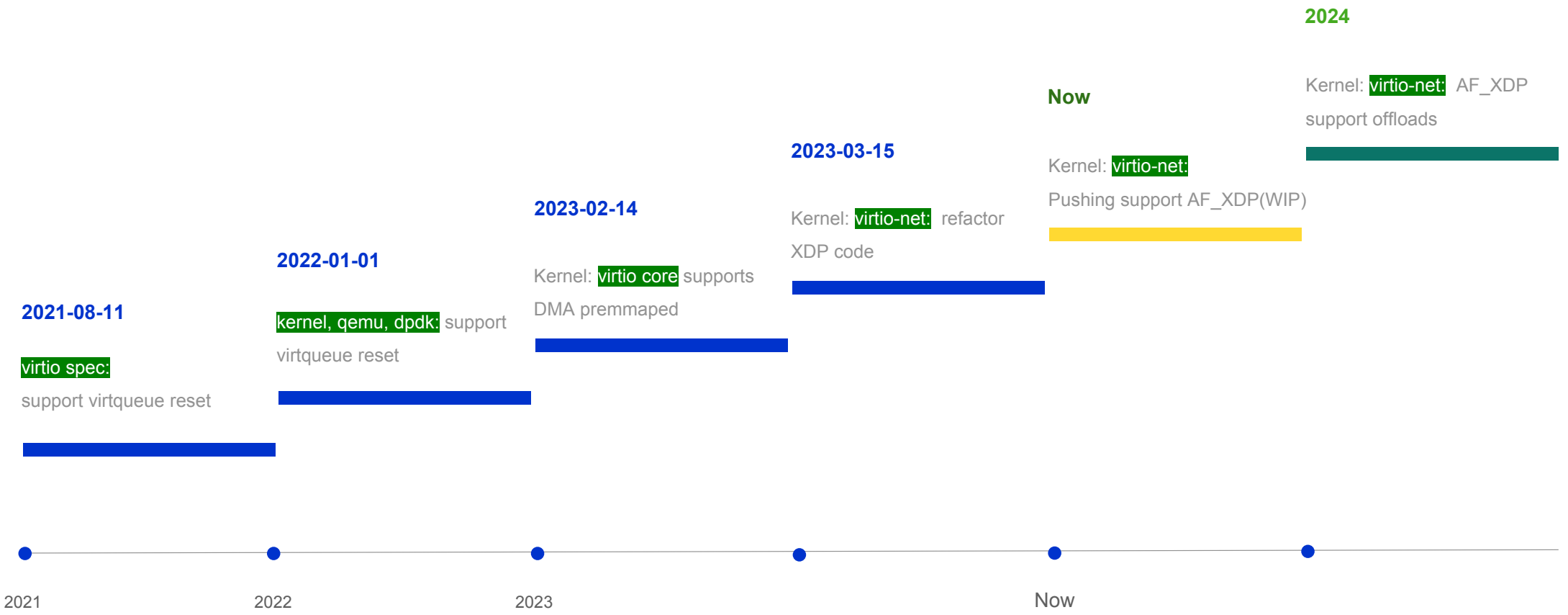
Virtio-net supports XDP/AF-XDP

ALIBABA CLOUD INTELLIGENCE GROUP



Virtio-net supports AF_XDP

ALIBABA CLOUD INTELLIGENCE GROUP



AF_XDP accelerates QUIC/HTTP3

ALIBABA CLOUD INTELLIGENCE GROUP

We have utilized this approach extensively internally. Initially, we developed a library called Express UDP to serve as the UDP stack. This library allows applications to receive and send UDP packets. Additionally, our Linux release, Anolis/Alinux, incorporates the feature of virtio-net supporting AF_XDP.

Using these components, the XQuic (Alibaba's QUIC library) can leverage AF_XDP to enhance the acceleration of QUIC/HTTP3. This work was completed approximately two years ago and has been widely applied on a large scale.

The performance achieved through this imple mentation can reach up to a 30% improvement.

ALIBABA CLOUD
INTELLIGENCE GROUP

THANKS