DPDK

DATA PLANE DEVELOPMENT KIT

Adding Eventdev support in ipsec-gw application

Hemant Agrawal (NXP)

Akhil Goyal (NXP)

DPDK Userspace - 2019

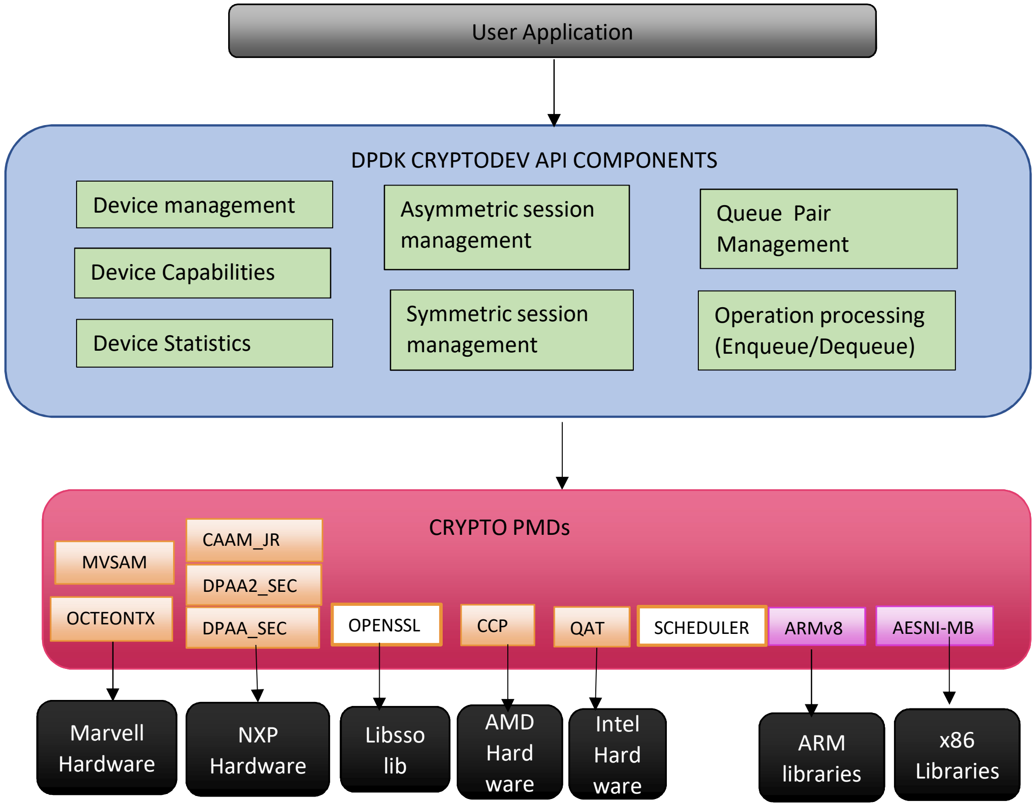
DPDK
DATA PLANE DEVELOPMENT KIT

Agenda

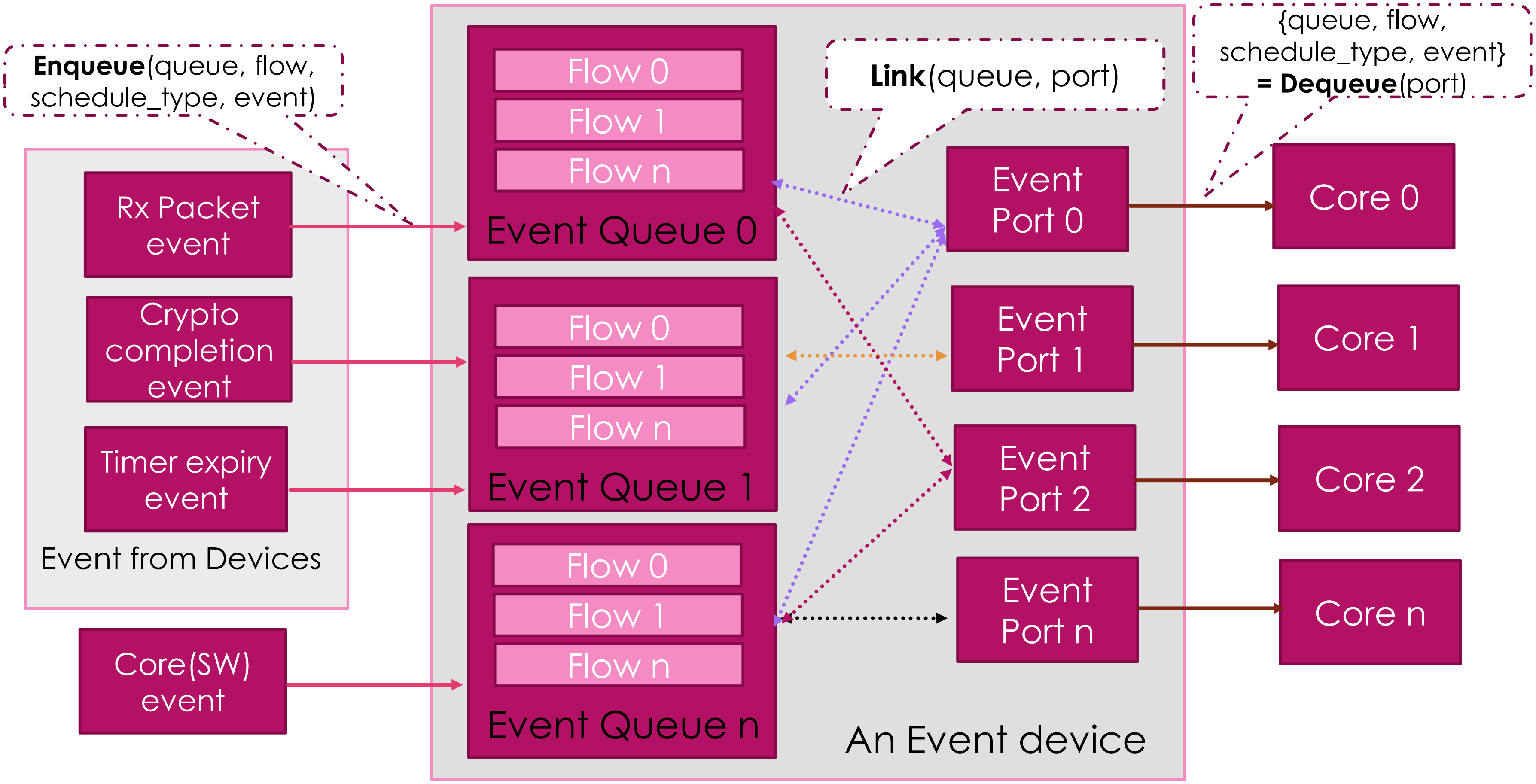
- Overview
  - Cryptodev
  - Event Crypto Adapter
- Crypto Event Adapter Processing Example
- Proposed changes in IPSEC event Gateway

CRYPTODEV

- A framework for processing symmetric and asymmetric crypto workload.
- Provides a standard API supporting transparent crypto processing for all vendors of crypto(SW/HW) PMD.
- Poll mode driver infrastructure with the recent addition of event mode support.
- User can choose to use any combination of software/hardware PMD and schedule work between them



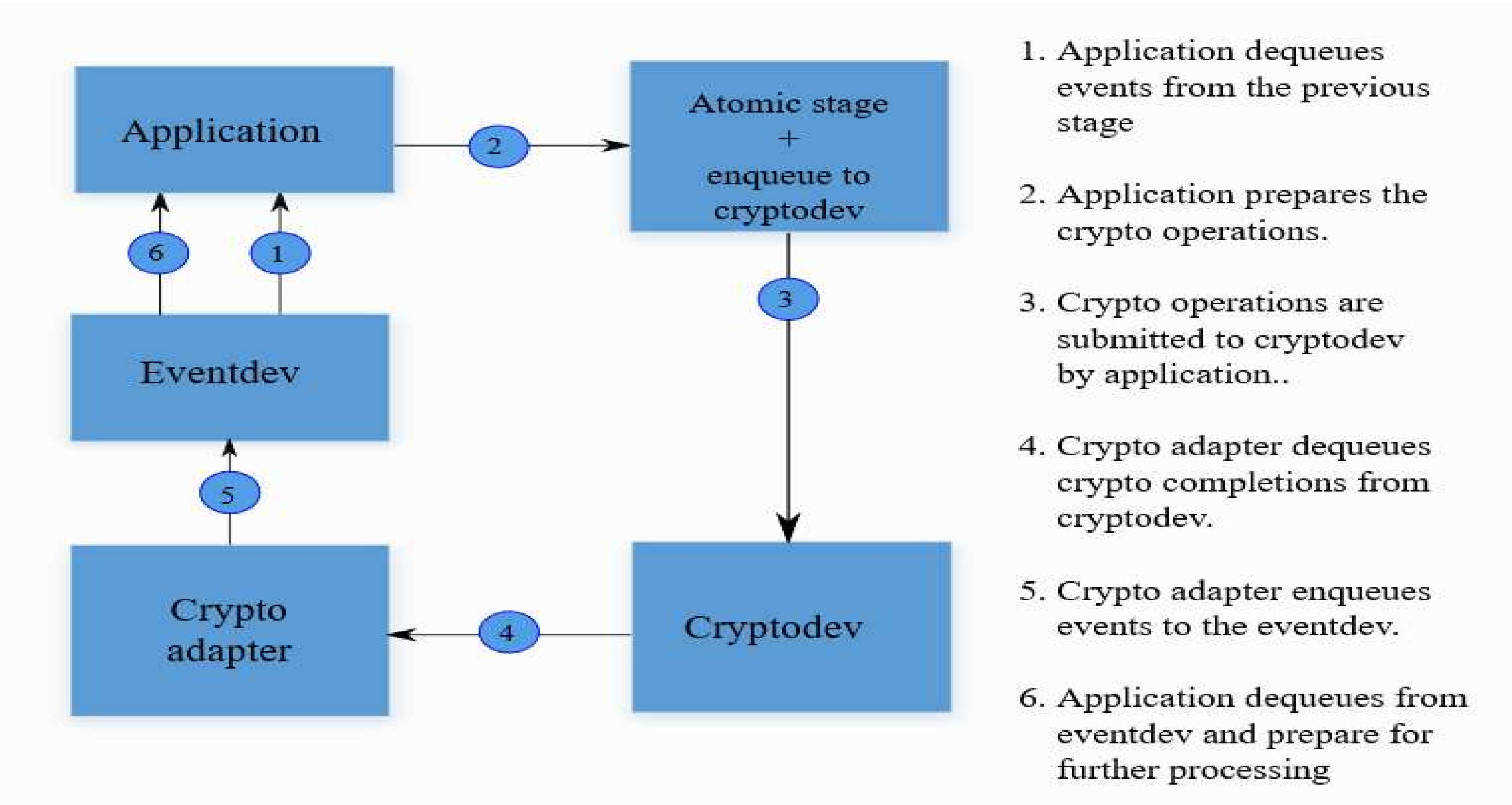
event driven model



Event Crypto Adapter

- Poll mode drivers means individual queue polling and 100% CPU utilization irrespective of amount of traffic being processed.
  - Event based processing can work with ‘n’ queue’
  - Event Port with Interrupt mode – no more wasted CPU cycles ☺
  - Each accelerator (Ethernet, Crypto, Timer etc) needs event adapter to connect eventdev
- Event crypto adapter adapts the crypto queues to work for event framework
- All crypto queues can be assigned to event device (hardware/ software scheduler)
- Event device schedule the traffic to multiple queues
  - Support ordered, atomic and parallel queues
- Reduces CPU utilization when traffic is low
- Better utilization of hardware resources

Event Crypto Adapter processing



1. Application dequeues events from the previous stage

2. Application prepares the crypto operations.

3. Crypto operations are submitted to cryptodev by application..

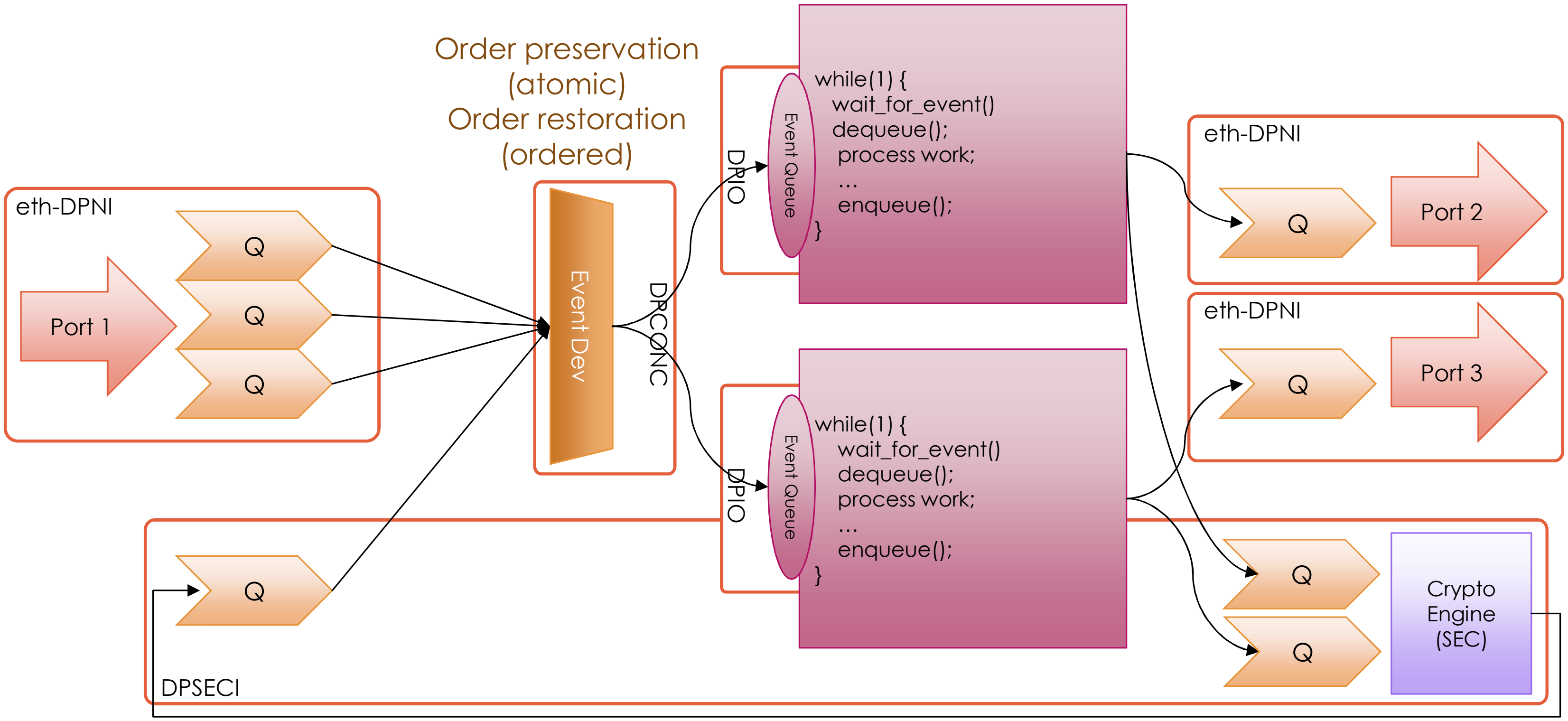
4. Crypto adapter dequeues crypto completions from cryptodev.

5. Crypto adapter enqueues events to the eventdev.

6. Application dequeues from eventdev and prepare for further processing

Crypto Adapter Example for NXP DPAA2 Platform

DPAA queuing & scheduling



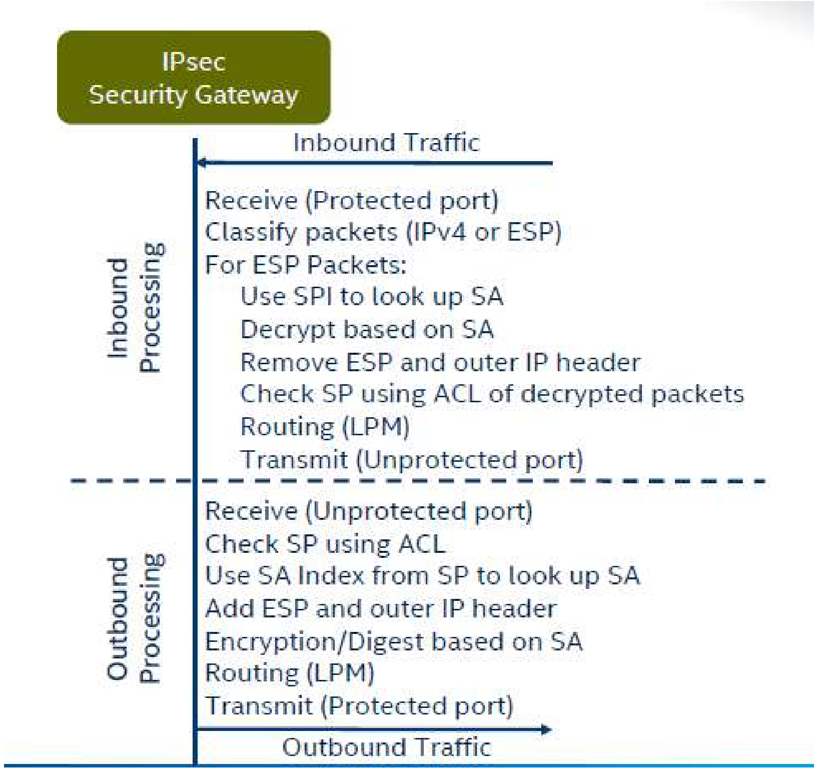
Order preservation (atomic)
Order restoration (ordered)

```
while(1) {
  wait_for_event()
  dequeue();
  process work;
  …
  enqueue();
}
```

```
while(1) {
  wait_for_event()
  dequeue();
  process work;
  …
  enqueue();
}
```

IPSEC-SECGW *Sample* Application

- Provide a L3 application for IPSEC forwarding
- Security Policies(SP) and Security Associations(SA) are manually configured using a cfg file.
- SPs are implemented as ACL rules
- SAs are stored in a table
- Routing is implemented using LPM
- Support all security acceleration modes.
- Support with and without IPSEC library
- Works well with both hardware and software devices



IPsec
Security Gateway

Inbound Traffic

Inbound Processing

Receive (Protected port)
Classify packets (IPv4 or ESP)
For ESP Packets:
  Use SPI to look up SA
  Decrypt based on SA
  Remove ESP and outer IP header
  Check SP using ACL of decrypted packets
  Routing (LPM)
  Transmit (Unprotected port)

Outbound Processing

Receive (Unprotected port)
Check SP using ACL
Use SA Index from SP to look up SA
Add ESP and outer IP header
Encryption/Digest based on SA
Routing (LPM)
Transmit (Protected port)

Outbound Traffic

Design principle for event in IPSec-secgw

Limitation in current design

- Synchronous Design
  - Crypto enqueue/dequeue APIs are defined to be used asynchronously but is getting used synchronously in the same thread.
  - Hardware crypto PMD may work slower on synchronous jobs – hence wasting cycles for empty dequeues
- Continuous polling on the ethernet/crypto dev for dequeue
  - No free CPU

Where event can help

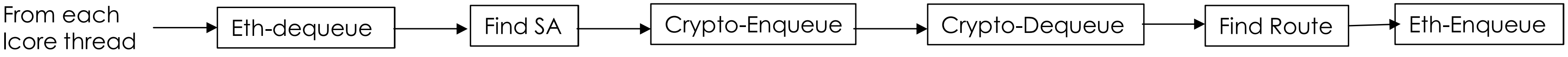
- When All (cryptodev/ethdev) queues are assigned to an eventdev, and we do dequeue from event device,
  - Segregate crypto/eth packets based on event type
  - Able to process both crypto and ethernet packets
  - So no more synchronous processing. ☺
- If the underneath Event PMD support interrupt mode, dequeue will happen only in case packets are available.
  - Free CPU at low traffic rates.

Proposed Changes

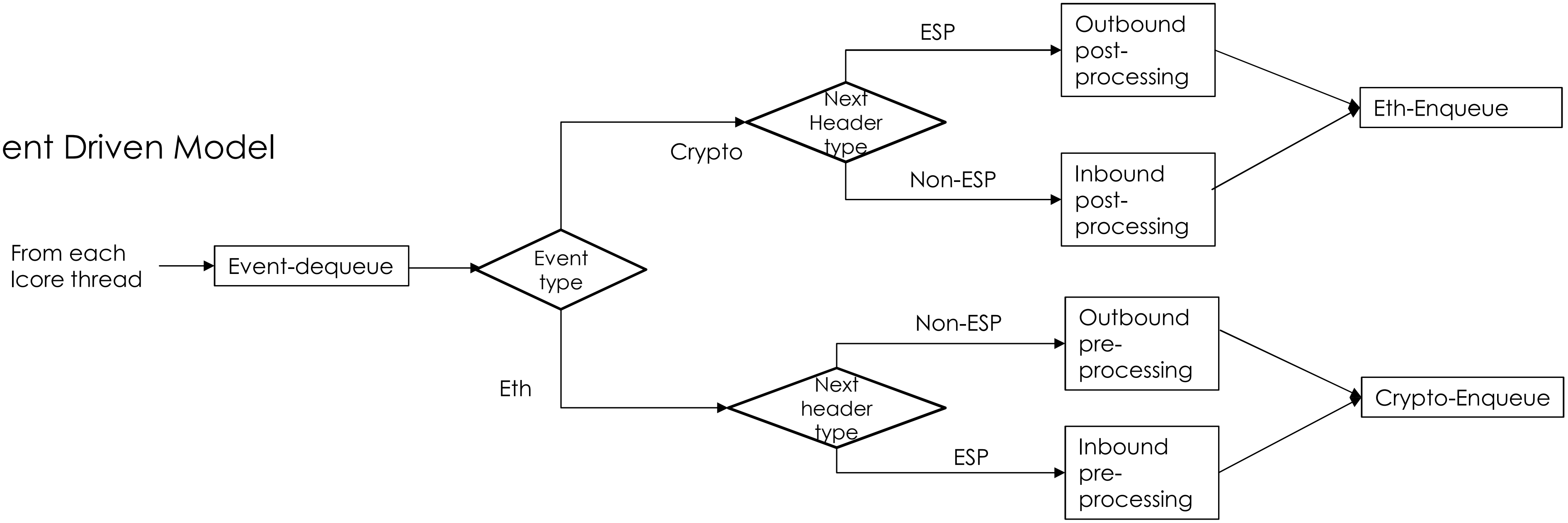
- Attach all crypto/ethernet queue to event device
- From Each thread, Instead of dequeue from ethernet/crypto device, dequeue from event device
  - Segregate packets from ethernet/crypto queues based on event type
- Packets from Crypto queues -> enqueue to ethernet device
  - Inbound packets will be plain packet
  - Outbound packets will be ESP packets
- Packets from Ethernet queues -> enqueue to crypto device
  - Inbound packets will be ESP packet coming for decryption
  - Outbound packets will be Plain packets for Encryption.

Proposed Execution changes

Existing Model



Event Driven Model



Challenges and Mitigation

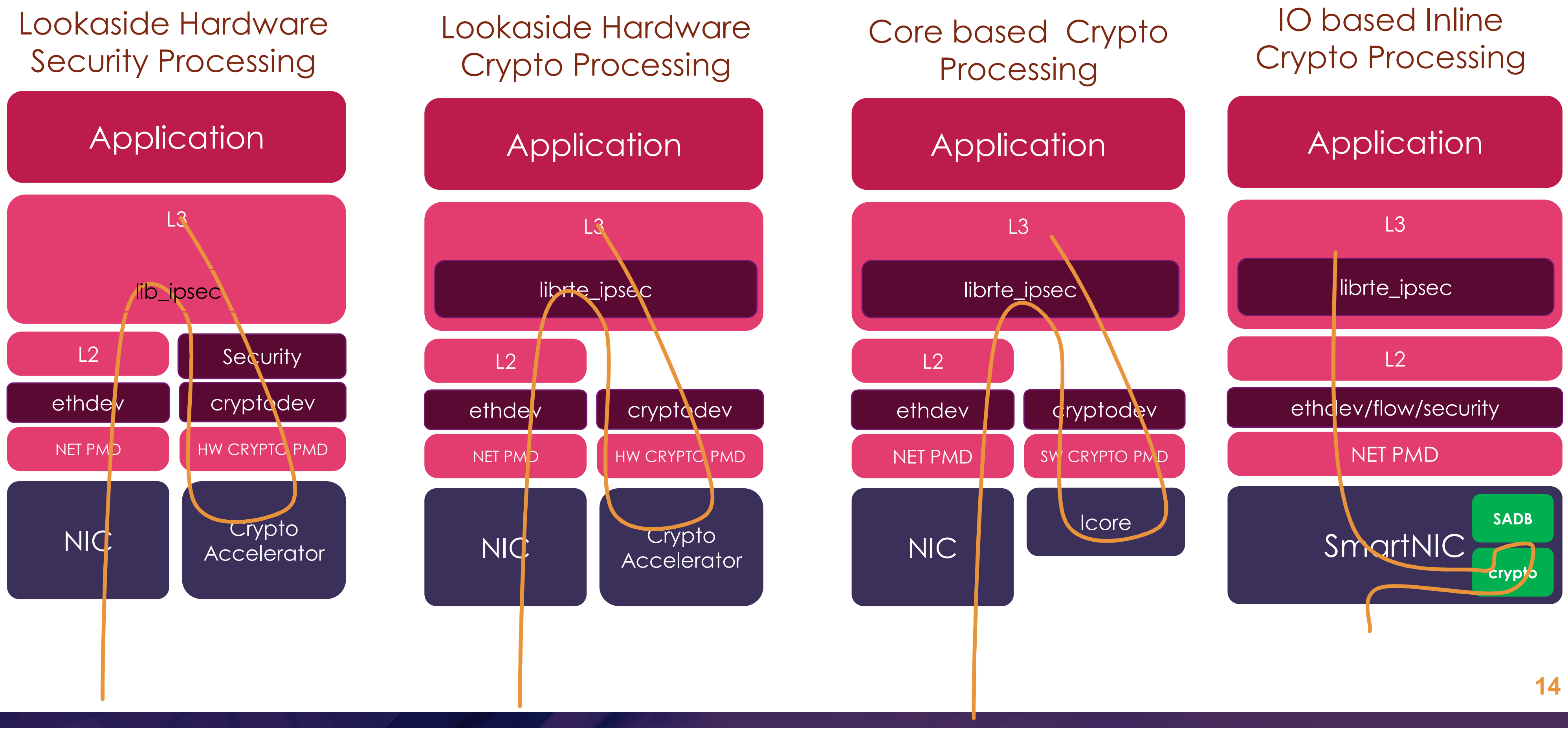
- Target is for 20.02.
- Plan to send RFC in 19.11 timeframe
- Main challenge is to co-exist with various crypto/security modes in the app.
- Packet re-ordering
  - Use Atomic queues (for order preservation)
  - Use ordered queues (for order restoration) depending on available support in PMD
- Cannot hold packets at any point to wait for next iteration
  - Process all packets from a burst and enqueue to next stage(crypto/eth) in the same context.

Questions?

Hemant Agrawal
<hemant.agrawal@nxp.com>

Akhil Goyal
<akhil.goyal@nxp.com>

Supported_processing_modes



Lookaside Hardware Security Processing

Application

L3

lib_ipsec

L2

Security

ethdev

cryptodev

NET PMD

HW CRYPTO PMD

NIC

Crypto Accelerator

Lookaside Hardware Crypto Processing

Application

L3

librte_ipsec

L2

ethdev

cryptodev

NET PMD

HW CRYPTO PMD

NIC

Crypto Accelerator

Core based Crypto Processing

Application

L3

librte_ipsec

L2

ethdev

cryptodev

NET PMD

SW CRYPTO PMD

NIC

lcore

IO based Inline Crypto Processing

Application

L3

librte_ipsec

L2

ethdev/flow/security

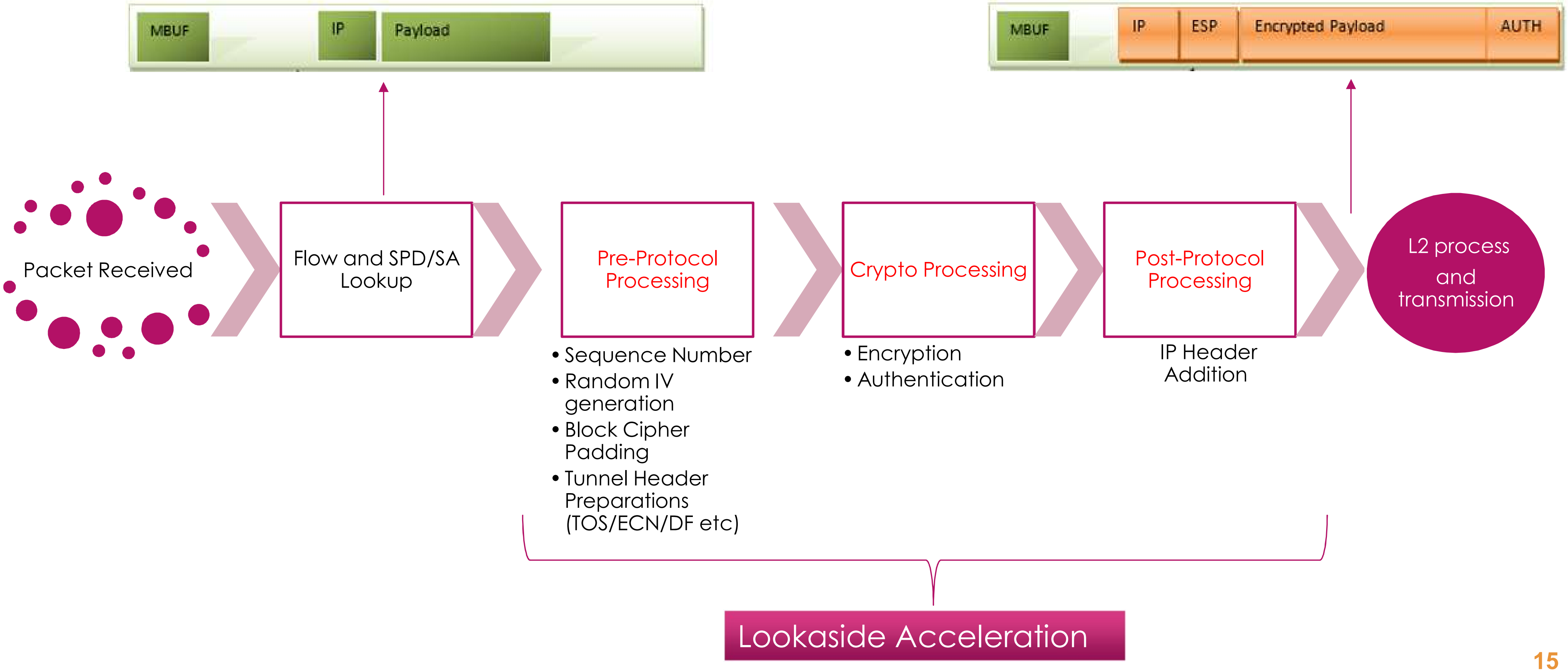
NET PMD

SmartNIC

SADB

crypto

IPSEC - Encrypt Packet Processing



Packet Received

Flow and SPD/SA Lookup

Pre-Protocol Processing

- Sequence Number
- Random IV generation
- Block Cipher Padding
- Tunnel Header Preparations (TOS/ECN/DF etc)

Crypto Processing

- Encryption
- Authentication

Post-Protocol Processing

IP Header Addition

L2 process and transmission

Lookaside Acceleration