DPDK Usability for OVS DPDK

IAN STOKES

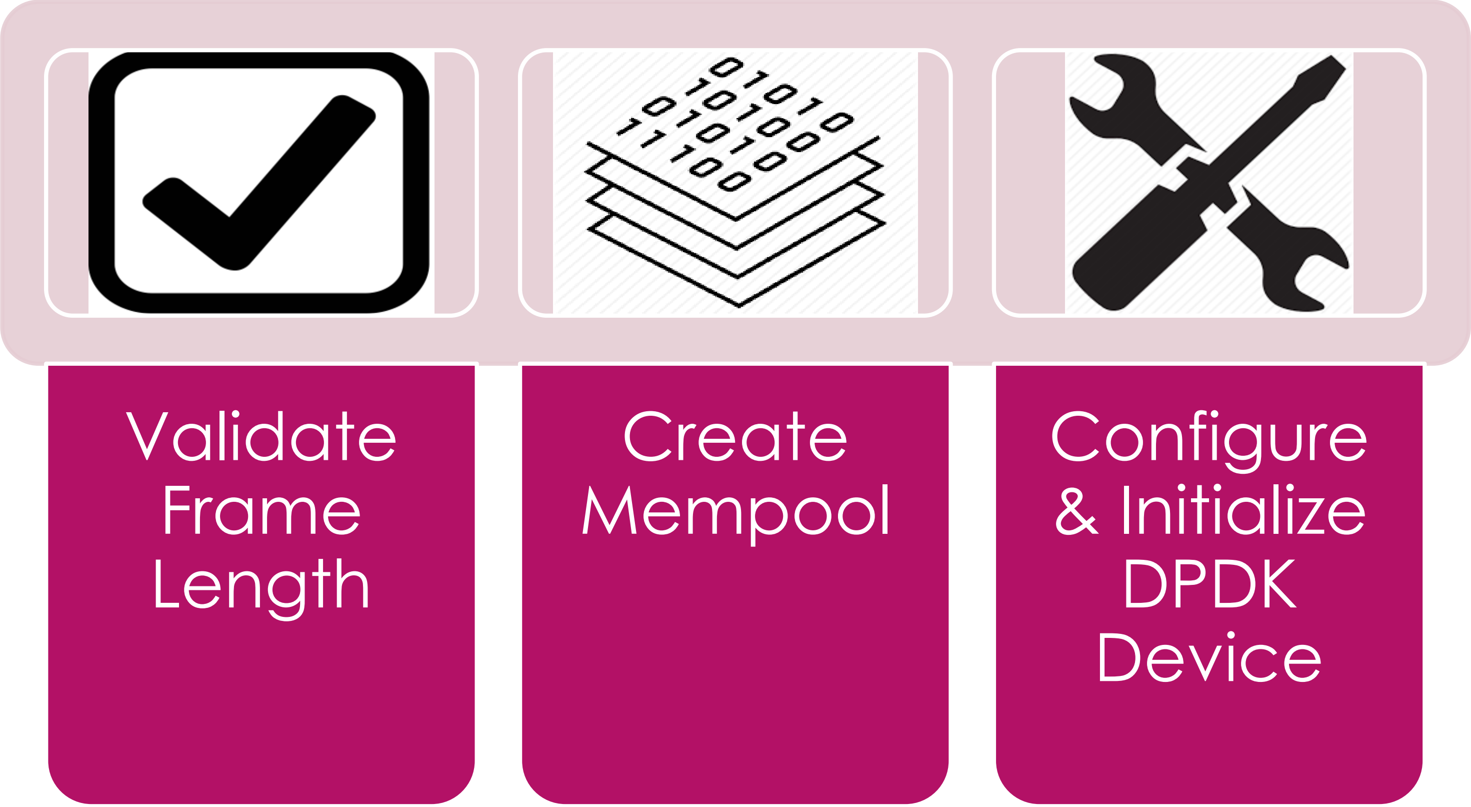
DPDK USERSPACE 2018

Content

- OVS DPDK MTU configuration steps
- Case Study 1: Device specific overhead
- Case Study 2: Scatter requirements
- Case Study 3: Device configuration state requirements
- Conclusion/Discussion

OVS DPDK MTU configuration

- OVS DPDK Uses DPDK 17.11 LTS.
- 3 stages to setting the MTU of a device.



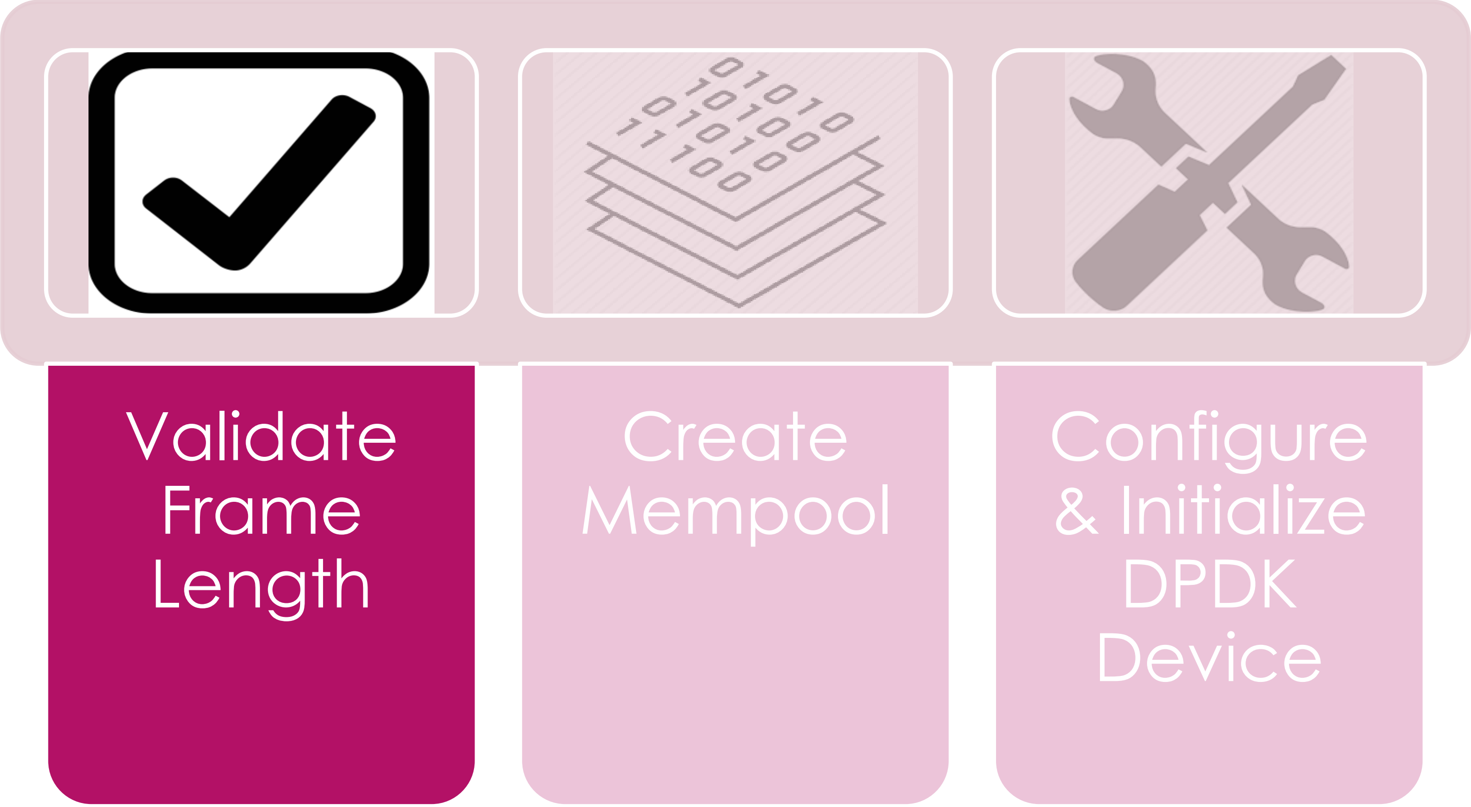
Validate Frame Length

Create Mempool

Configure & Initialize DPDK Device

OVS DPDK MTU configuration

- OVS DPDK Uses DPDK 17.11 LTS.
- 3 stages to setting the MTU of a device.



Validate Frame Length

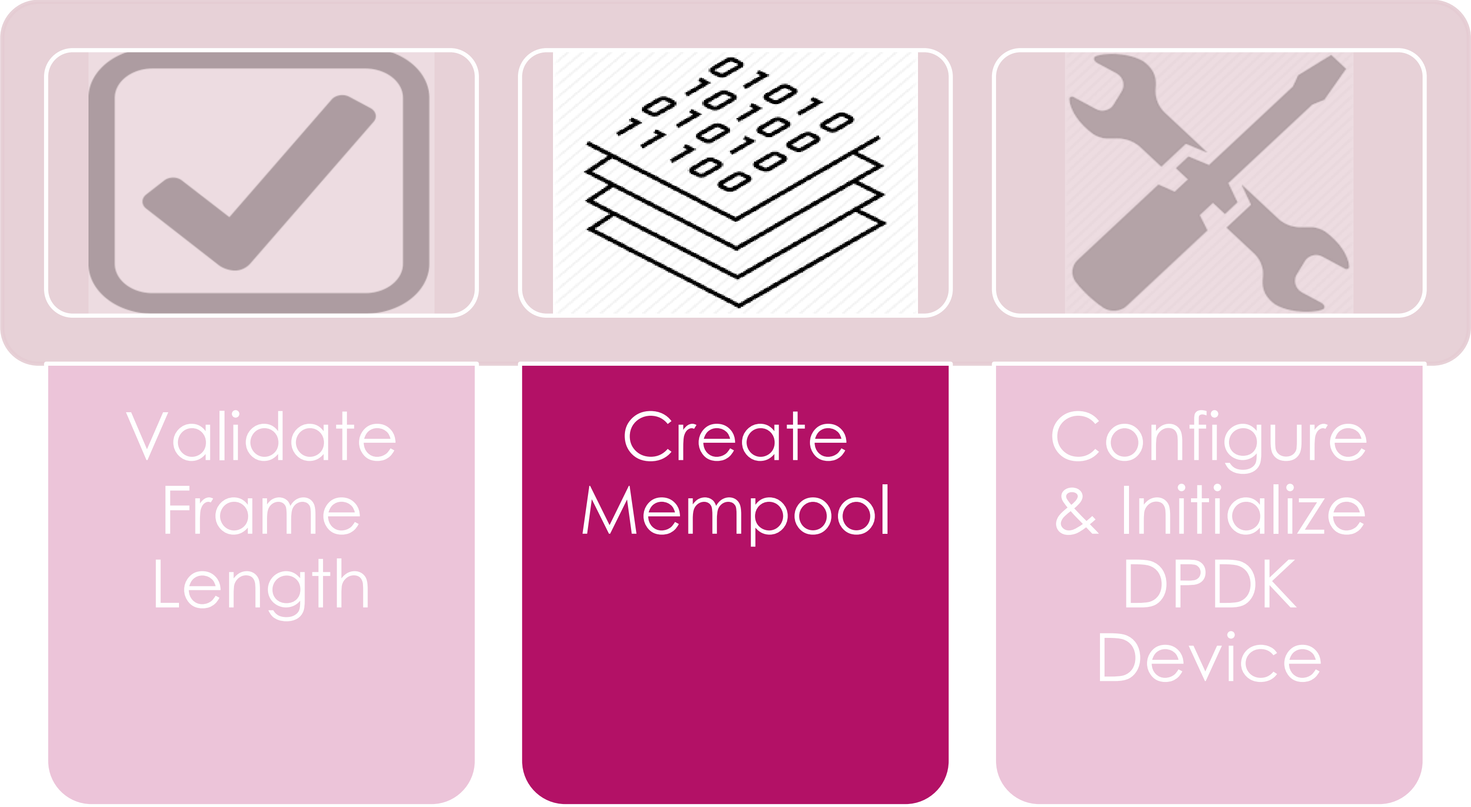
Create Mempool

Configure & Initialize DPDK Device

- Validate Frame Length
  - Requested MTU represents layer 3 MTU.
  - Must account for layer 2 header and CRC.
  - **Ensure overall frame length of the requested MTU does not surpass the NETDEV_DPDK_MAX_PKT_LEN (9728 B).**

OVS DPDK MTU configuration

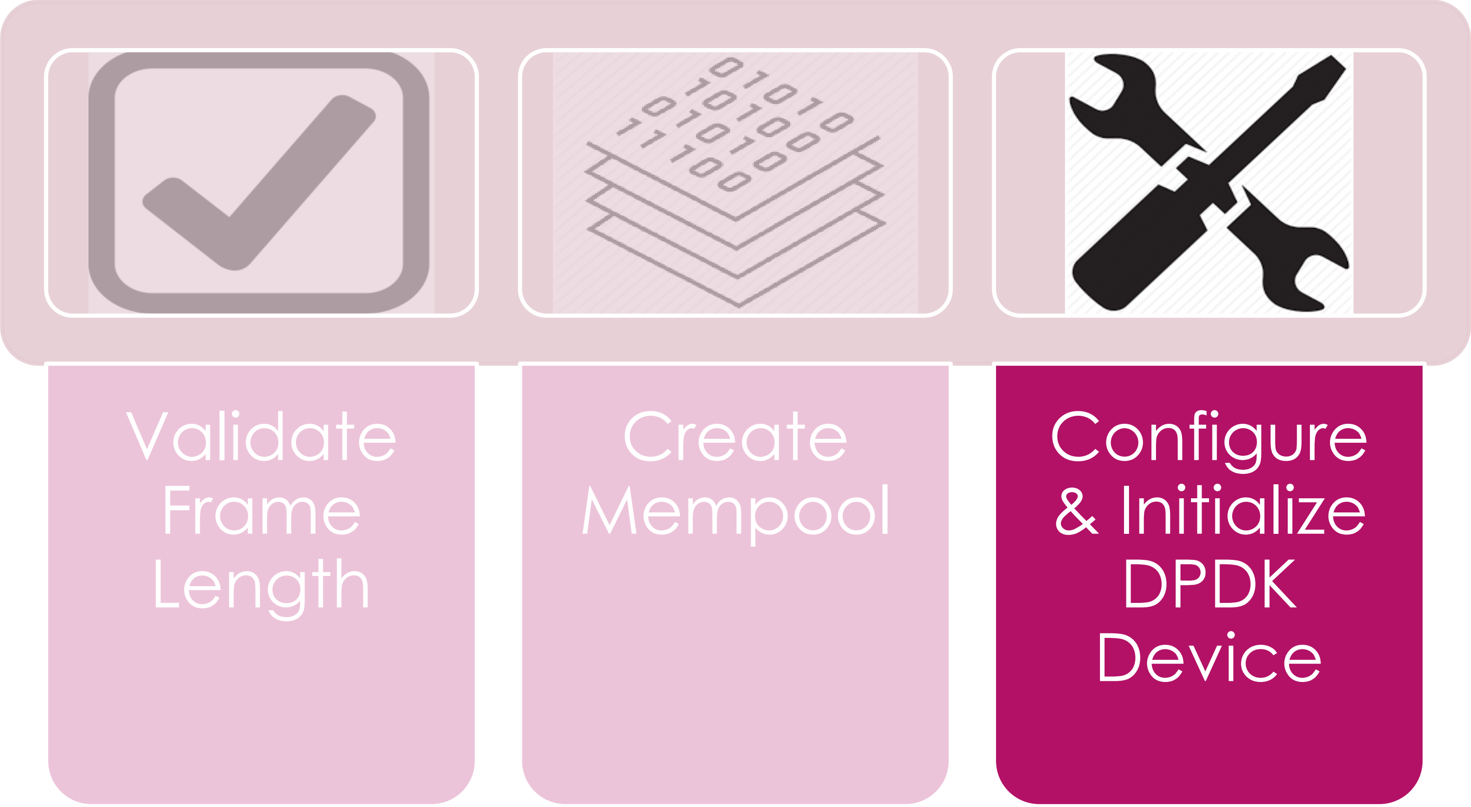
- OVS DPDK Uses DPDK 17.11 LTS.
- 3 stages to setting the MTU of a device.



- <u>Create Mempool</u>
  - **Requested MTU directly affects mbuf size.**
  - Mbuf size calculated as ‘Requested mtu + L2 headers + CRC + RTE_PKTMBUF_HEADROOM’
  - Round final value to be multiple of 1024.

OVS DPDK MTU configuration

- Note: OVS DPDK Uses DPDK 17.11 LTS.
- 3 stages to setting the MTU of a device.



Validate Frame Length

Create Mempool

Configure & Initialize DPDK Device

- Configure & Initialize DPDK Device
  - **Device stopped** as part of configuration,
  - MTU is configured with **`rte_eth_dev_set_mtu(port_id, mtu);`**
  - Various other configurations (TXQs, RXQs etc.).
  - **Device started** when configuration completes.

Case Study 1: Device specific overhead

IXGBE

MTU REQUEST 9710

□ Frame length Validated. ✓

□ Mempool created. ✓

□ Device configured & initialized. ✓

Case Study 1: Device specific overhead cont.

IXGBE

MTU REQUEST 9710

- □ Frame length Validated. ✓
- □ Mempool created. ✓
- □ Device configured & initialized. ✓

i40e

MTU REQUEST 9710

- □ Frame length Validated. ✓
- □ Mempool created. ✓
- □ Device configured & initialized. X

- Interface dpdk0 MTU (9710) setup error: Invalid argument (-EINVAL)
- Why?

Case Study 1: Device specific overhead cont.

IXGBE

i40e

□ MTU REQUEST 97

UEST 9710

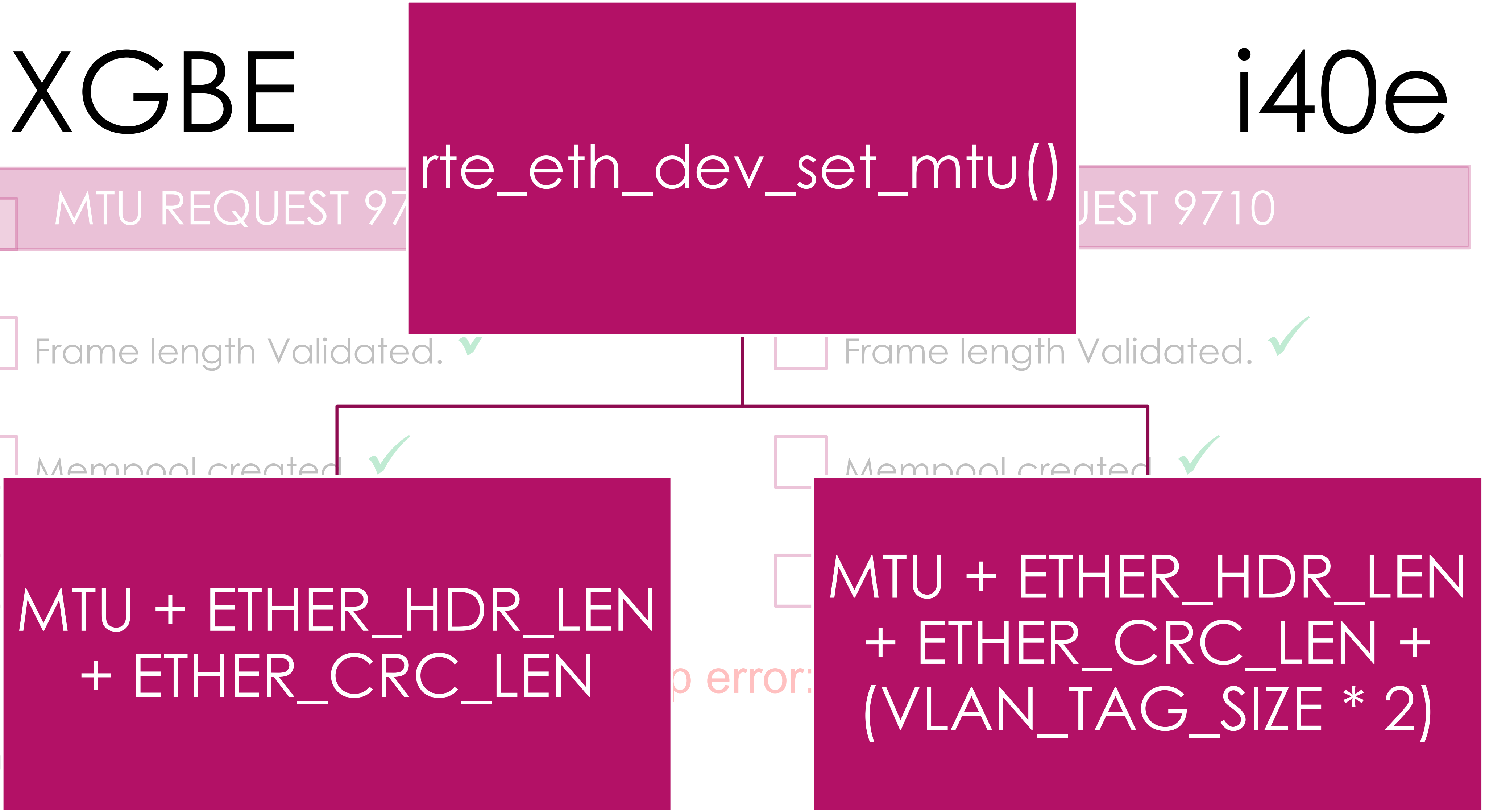
□ Frame length Validated. ✓

□ Frame length Validated. ✓

□ Mempool created ✓

□ Mempool created ✓

- I p error:
- V



rte_eth_dev_set_mtu()

MTU + ETHER_HDR_LEN + ETHER_CRC_LEN

MTU + ETHER_HDR_LEN + ETHER_CRC_LEN + (VLAN_TAG_SIZE * 2)

Case Study 1: PMD & Associated Overhead

PMD	Overhead	Total Bytes
qede	ETHER_HDR_LEN + ETHER_CRC_LEN	18
cxgbe	ETHER_HDR_LEN + ETHER_CRC_LEN	18
dpaa2	ETHER_HDR_LEN + ETHER_CRC_LEN	18
ixgbe	ETHER_HDR_LEN + ETHER_CRC_LEN	18
luiquidio	ETHER_HDR_LEN + ETHER_CRC_LEN	18
thunderx	ETHER_HDR_LEN + ETHER_CRC_LEN	18
em1000	ETHER_HDR_LEN + ETHER_CRC_LEN + VLAN_TAG_SIZE	22
igb	ETHER_HDR_LEN + ETHER_CRC_LEN + VLAN_TAG_SIZE	22
bnxt	ETHER_HDR_LEN + ETHER_CRC_LEN + (VLAN_TAG_SIZE * 2)	26
i40e	ETHER_HDR_LEN + ETHER_CRC_LEN + (VLAN_TAG_SIZE * 2)	26
mrvl	MV_MH_SIZE + ETHER_HDR_LEN + ETHER_CRC_LEN	?

Case Study 1: OVS Solution

OVS Solution

- Must account for vlan * 2 when
  - `MTU + ETHER_HDR_LEN + ETHER_CRC_LEN + (VLAN_TAG_SIZE * 2) > NETDEV_DPDK_MAX_PKT_LEN`

Case Study 1: OVS Solution cont.

OVS Solution

- Must account for vlan * 2 when
  - `MTU + ETHER_HDR_LEN + ETHER_CRC_LEN + (VLAN_TAG_SIZE * 2) > NETDEV_DPDK_MAX_PKT_LEN`

Problem:

- MTU upper limit will be reduced by 4 or 8 bytes for devices that do not have to account for 2 * VLAN headers in overhead.

Case Study 1: OVS Solution cont.

OVS Solution

- Must account for vlan * 2 when
  - `MTU + ETHER_HDR_LEN + ETHER_CRC_LEN + (VLAN_TAG_SIZE * 2) > NETDEV_DPDK_MAX_PKT_LEN`

Problem:

- MTU upper limit will be reduced by 4 or 8 bytes for devices that do not have to account for 2 * VLAN headers in overhead.

DPDK Solution

- Expose device specific overhead for PMDs.
- **Extend the existing ETH DEV API ?**
  - rte_eth_dev_get_max_mtu(port_id)
- **Make info available in rte_eth_dev_info struct ?**

Case Study 2: Scatter requirements

- PMDs can require **scatter** explicitly set for jumbo rx.

Case Study 2: Scatter requirements cont.

- PMDs can require **scatter** explicitly set for jumbo rx.

i40e

□ i40e: Not required, handled in i40e_set_mtu().

Case Study 2: Scatter requirements cont.

- PMDs can require **scatter** explicitly set for jumbo rx.

i40e/ixgbe

□ i40e: Not required, handled in i40e_set_mtu().

□ ixgbe: Required pre DPDK 17.11.

Case Study 2: Scatter requirements cont.

- PMDs can require **scatter** explicitly set for jumbo rx.

i40e/ixgbe/igb

□ i40e: Not required, handled in i40e_set_mtu().

□ ixgbe: Required pre DPDK 17.11.

□ igb: Required.

Case Study 2: Scatter requirements cont.

- PMDs can require **scatter** explicitly set for jumbo rx.

i40e/ixgbe/igb/nfp

□ i40e: Not required, handled in i40e_set_mtu().

□ ixgbe: Required pre DPDK 17.11.

□ igb: Required.

□ nfp: **Not supported**

Case Study 2: OVS Solution

OVS Solution

- Check for nfp driver explicitly before enabling scatter.
  - `if (strncmp(info.driver_name, "net_nfp", 7))`

Case Study 2: OVS Solution cont.

OVS Solution

- Check for nfp driver explicitly before enabling scatter.
  - `if (strncmp(info.driver_name, "net_nfp", 7))`

Problem:

- Device specific checks introduced to OVS DPDK code base.
- Only resolves issue for nfp PMD.

Case Study 2: OVS Solution cont.

OVS Solution

- Check for nfp driver explicitly before enabling scatter.
  - `if (strncmp(info.driver_name, "net_nfp", 7))`

Problem:

- Device specific checks introduced to OVS code base.
- Only resolves issue for nfp PMD.

DPDK Solution

- **Upcoming rx offload capability API.**
  - Implemented for nfp in 17.11, missing for ixgbe/i40e/igb.
- **Handle scatter configuration in class specifc mtu_set functions.**

Case Study 3: Device configuration state requirements

Configure & Initialize DPDK Device

- **Device stopped**
- MTU configured with **`rte_eth_dev_set_mtu();`**
- **Device started**

Case Study 3: Device configuration state requirements cont.

Configure & Initialize DPDK Device

- **Device stopped**
- MTU configured with `rte_eth_dev_set_mtu();`
- **Device started**

i40e/ixgbe/qede

□ i40e: Device must be stopped.

□ ixgbe: Stopped if scatter required.

□ qede: **Must be active** (pre 17.11.)

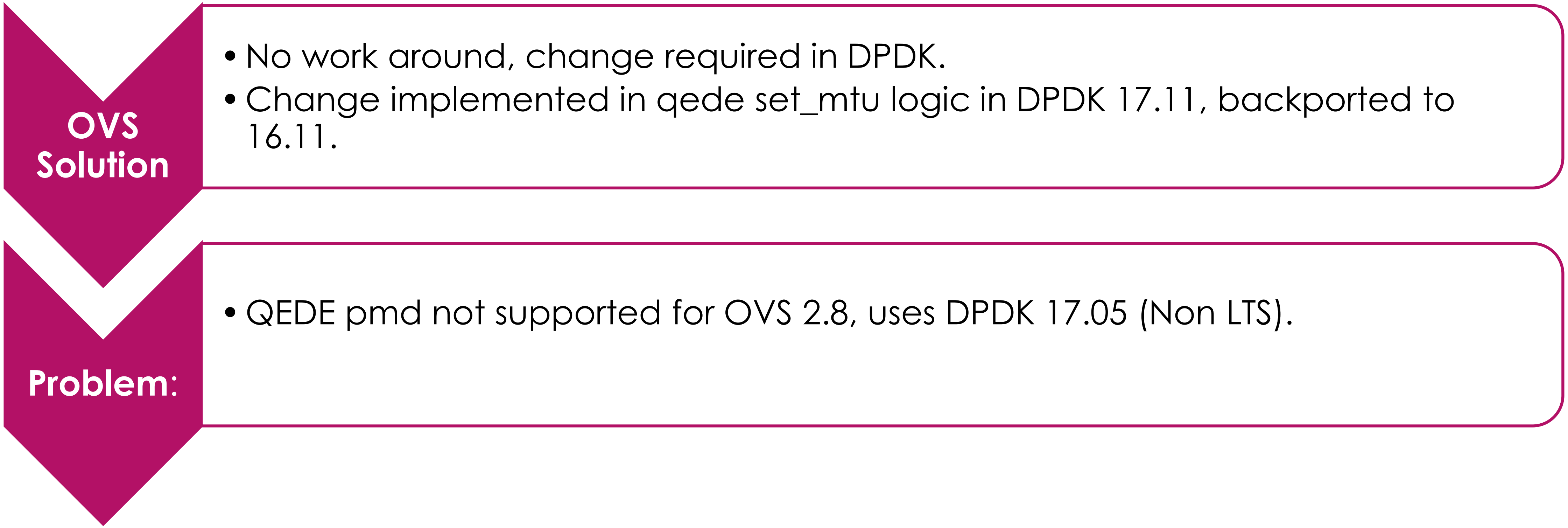
- **Explicit stop/start device within set_mtu()**

Case Study 3: OVS Solution

OVS Solution

- No work around, change required in DPDK.
- Change implemented in qede set_mtu logic in DPDK 17.11, backported to 16.11.

Case Study 3: OVS Solution cont.



OVS Solution

- No work around, change required in DPDK.
- Change implemented in qede set_mtu logic in DPDK 17.11, backported to 16.11.

Problem:

- QEDE pmd not supported for OVS 2.8, uses DPDK 17.05 (Non LTS).

Case Study 3: OVS Solution cont.

OVS Solution

- No work around, change required in DPDK.
- Change implemented in qede set_mtu logic in DPDK 17.11, backported to 16.11.

Problem:

- QEDE pmd not supported for OVS 2.8, uses DPDK 17.05 (Non LTS).

DPDK Solution

- **Solution already in place.**
- **Underlying behaviour should be uniform across PMDs.**

Conclusion/Discussion

- Ethdev API helps OVS DPDK be hardware agnostic.
- Corner cases can exist .e.g. behavior regarding rte_eth_dev_set_mtu().
- Solutions to avoid such cases
  - Expose device specific overhead via API extension or device info.
  - Expose device capabilities.
  - Follow uniform behavior in underlying API implementations.

DPDK
DATA PLANE DEVELOPMENT KIT

Questions?

Email: ian.stokes@intel.com

Legal Disclaimer

General Disclaimer:

© Copyright 2018 Intel Corporation. All rights reserved. Intel, the Intel logo, Intel Inside, the Intel Inside logo, Intel. Experience What's Inside are trademarks of Intel. Corporation in the U.S. and/or other countries. *Other names and brands may be claimed as the property of others.

Technology Disclaimer:

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at [intel.com].

Performance Disclaimers:

Cost reduction scenarios described are intended as examples of how a given Intel- based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction. Results have been estimated or simulated using internal Intel analysis or architecture simulation or modelling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.