DPDK SUMMIT
Powering the Future of Networking Software

12–13 May 2026
Stockholm, Sweden

Support for IKE in DPDK Cloud Native Router

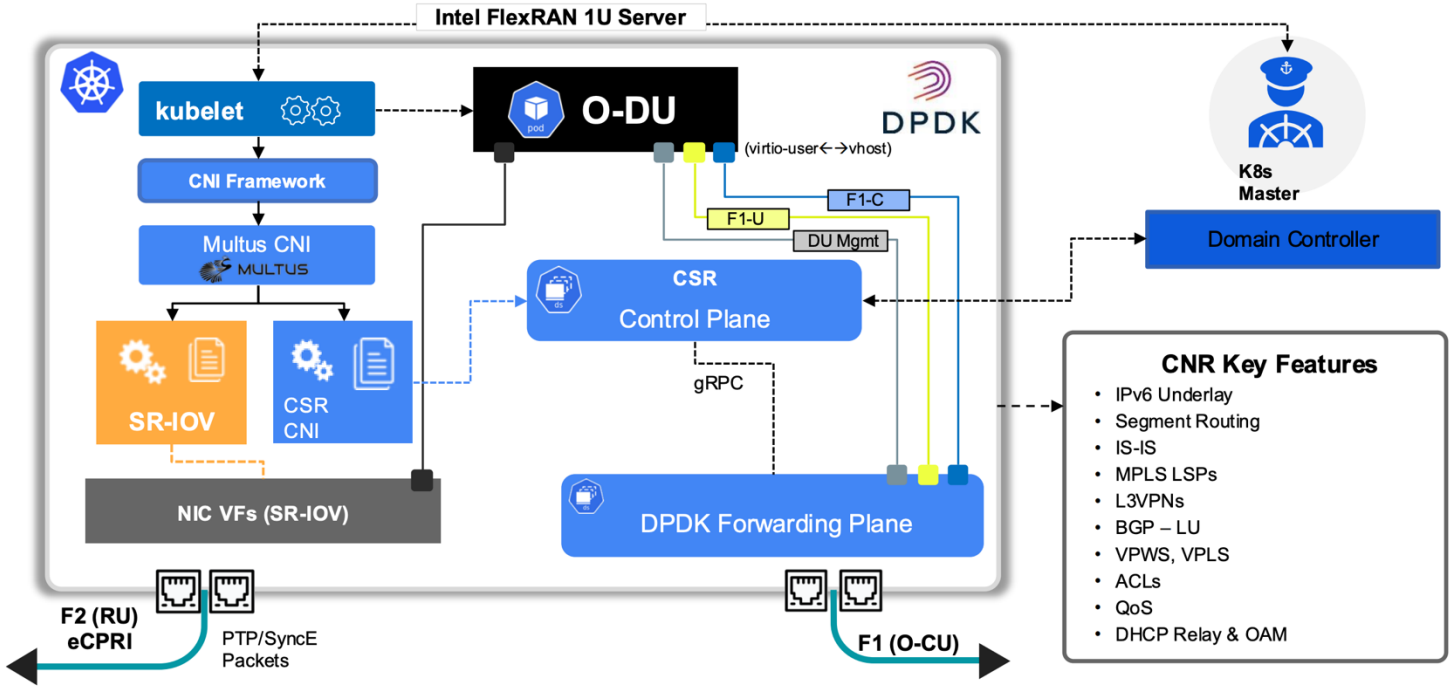
Srikanth Revanuru
Kiran K N
HPE

HPE

Agenda

- Introduction
  - What is Cloud Native Router
  - And it’s use case
- Problem Statement –IKEv2 Engine for the Cloud Native Router
- Proposed Solution – Strongswan
- Approach & Discussion
  - sswan-vrouter plugin
  - Integration to the CNR
- Q & A

DPDK Cloud Native Router



DPDK SUMMIT
Powering the Future of Networking Software

HPE Juniper networking

Juniper Business Use Only

IPSec in Cloud Native Router

Usecase 1 : 5G Transport beyond the Cell Site Router

In 5G, IPSec is used to secure RAN and edge transport. Secures traffic from multiple cell sites toward regional or core routers.

Usecase 2 : Secure Cloud Routing

CNR can act as secure edge router between data centers. This enables Secure East-West traffic.

Usecase 3 : Management and Control Plane Security

IPSec protects the routing protocol adjacencies and and management traffic, enforcing zero-trust principles in cloud native, disaggregated network deployments.

IKEv2 Control plane gap

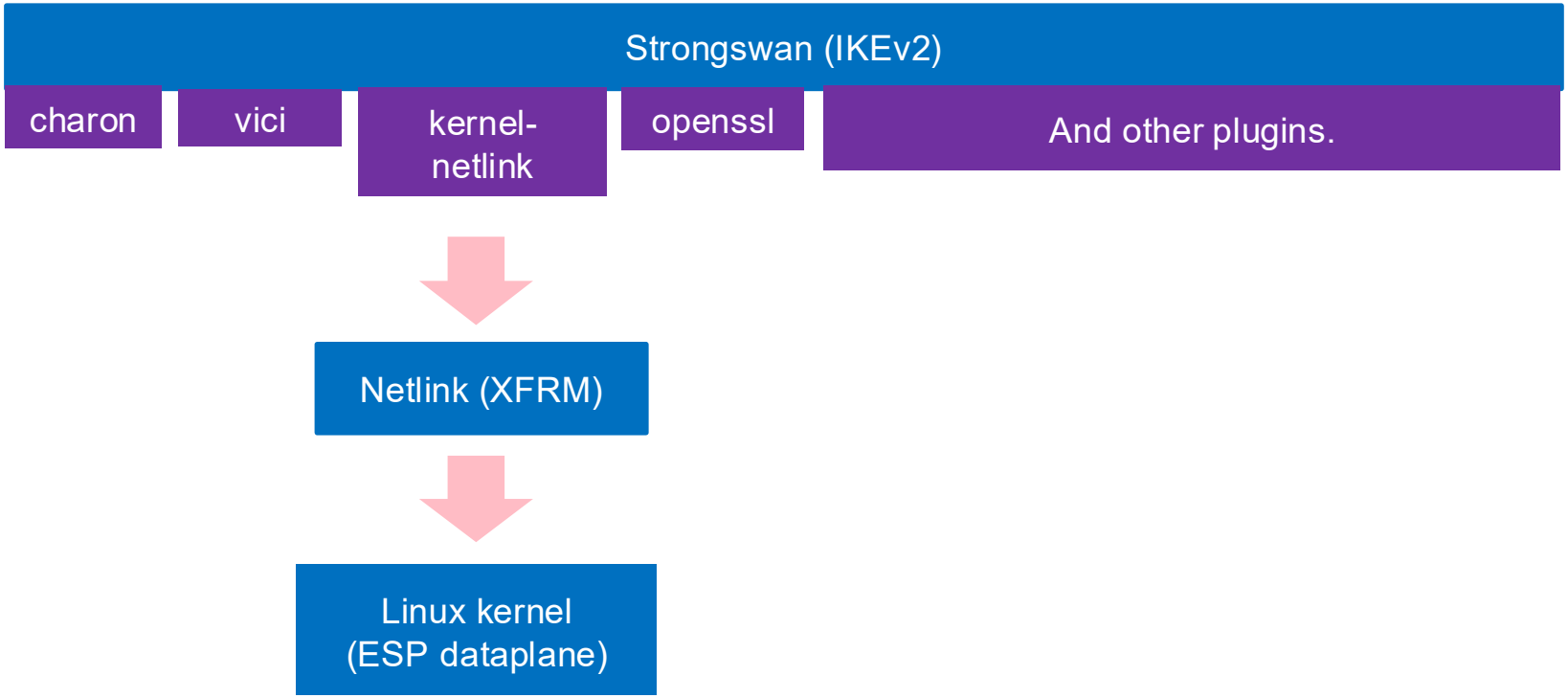
- DPDK cryptodev works with static keys.
- Static keys are not an option in core networking deployments.
- Dataplane is agnostic to IKEv2 negotiation and generation of keys.
- Production IPSec needs **dynamic key exchange** (IKEv2)
- Need a production ready IKEv2 engine, that can work with DPDK dataplane

Strongswan(IKEv2)

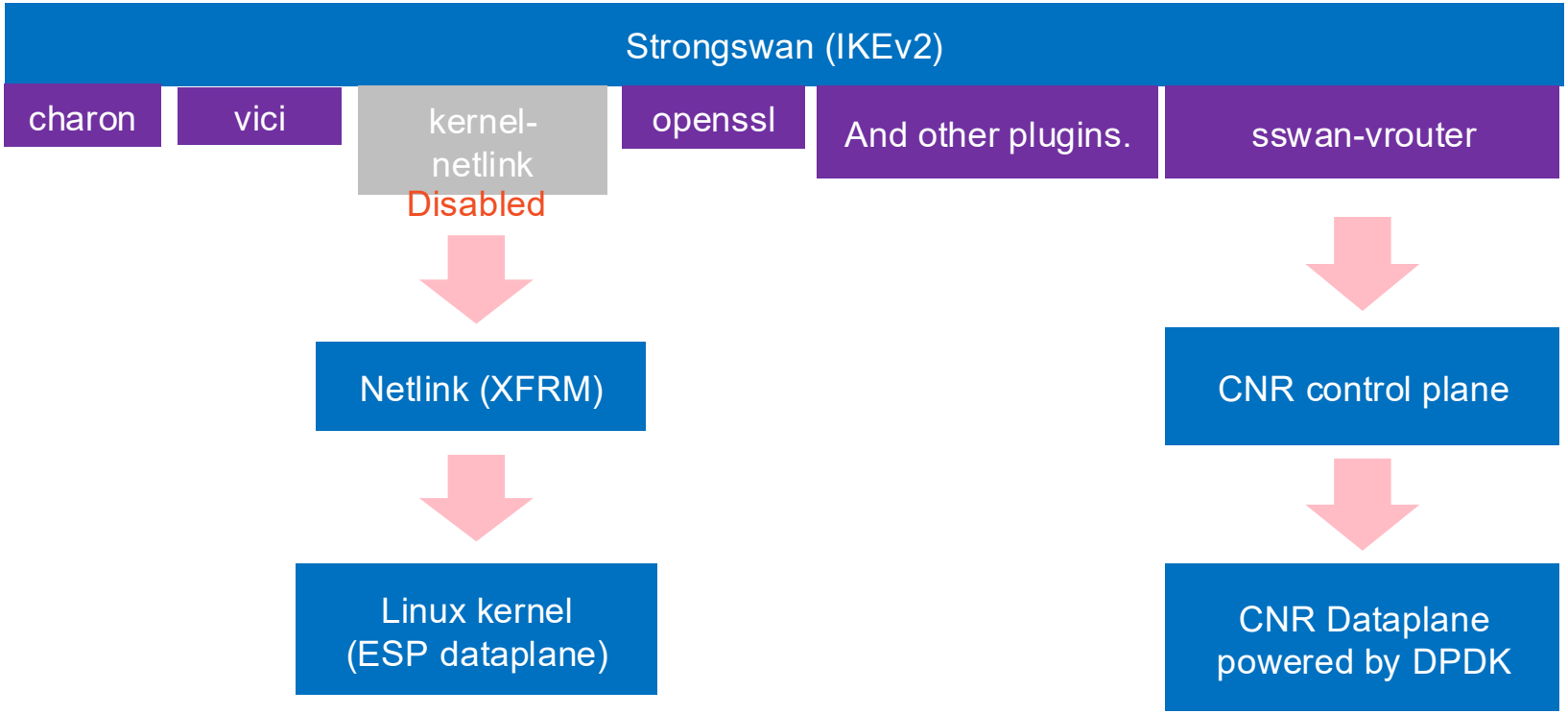
Comprehensive implementation of the **Internet Key Exchange** (IKE) protocols that allows securing IP traffic in policy and route-based **IPSec** scenarios.

- Clean separation between Control plane (IKE) & Dataplane (ESP)
- Dedicated IKEv2 control plane (charon)
- Plugin based architecture
- Well defined control interface (VICI)
- No hard dependency on Linux Kernel ESP processing

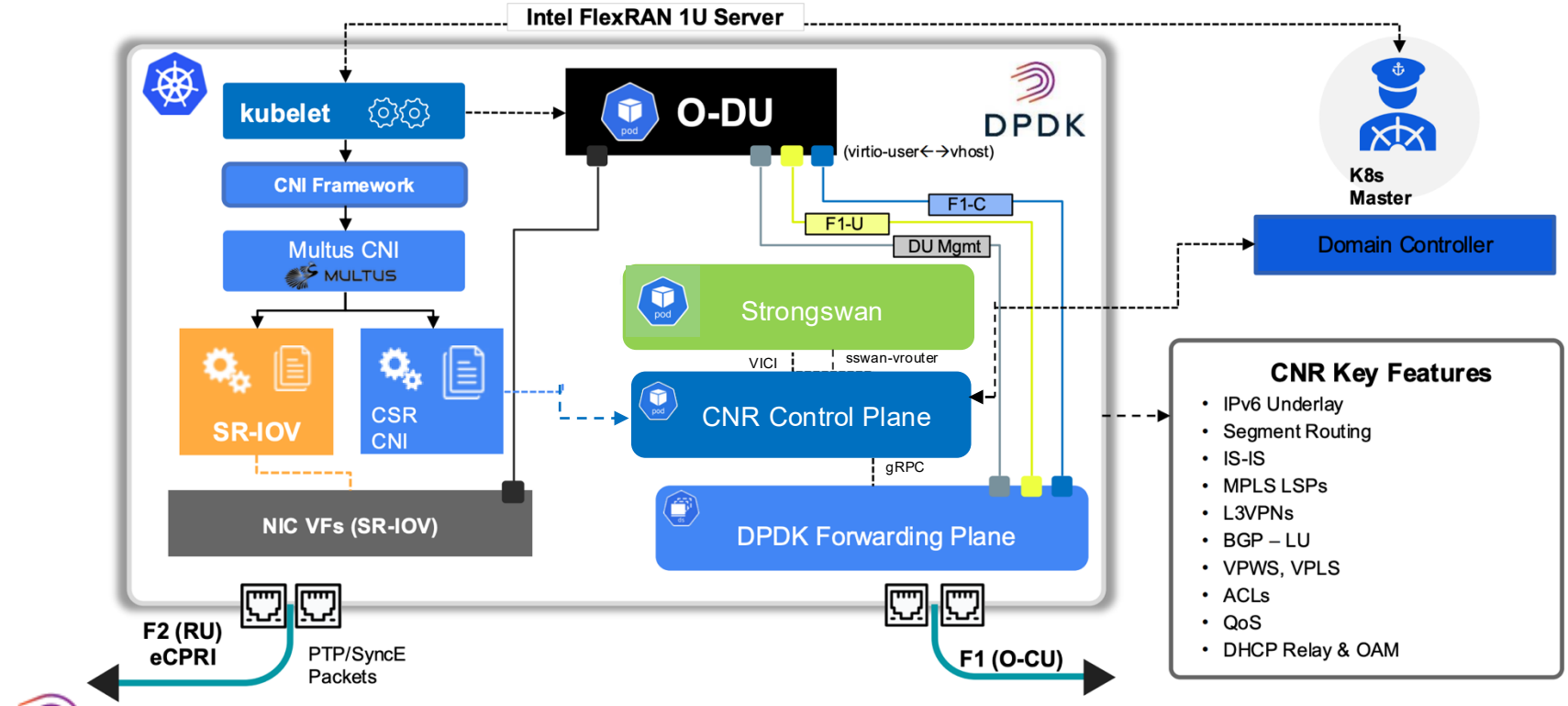
Strongswan – plugins



sswan-vrouter plugin

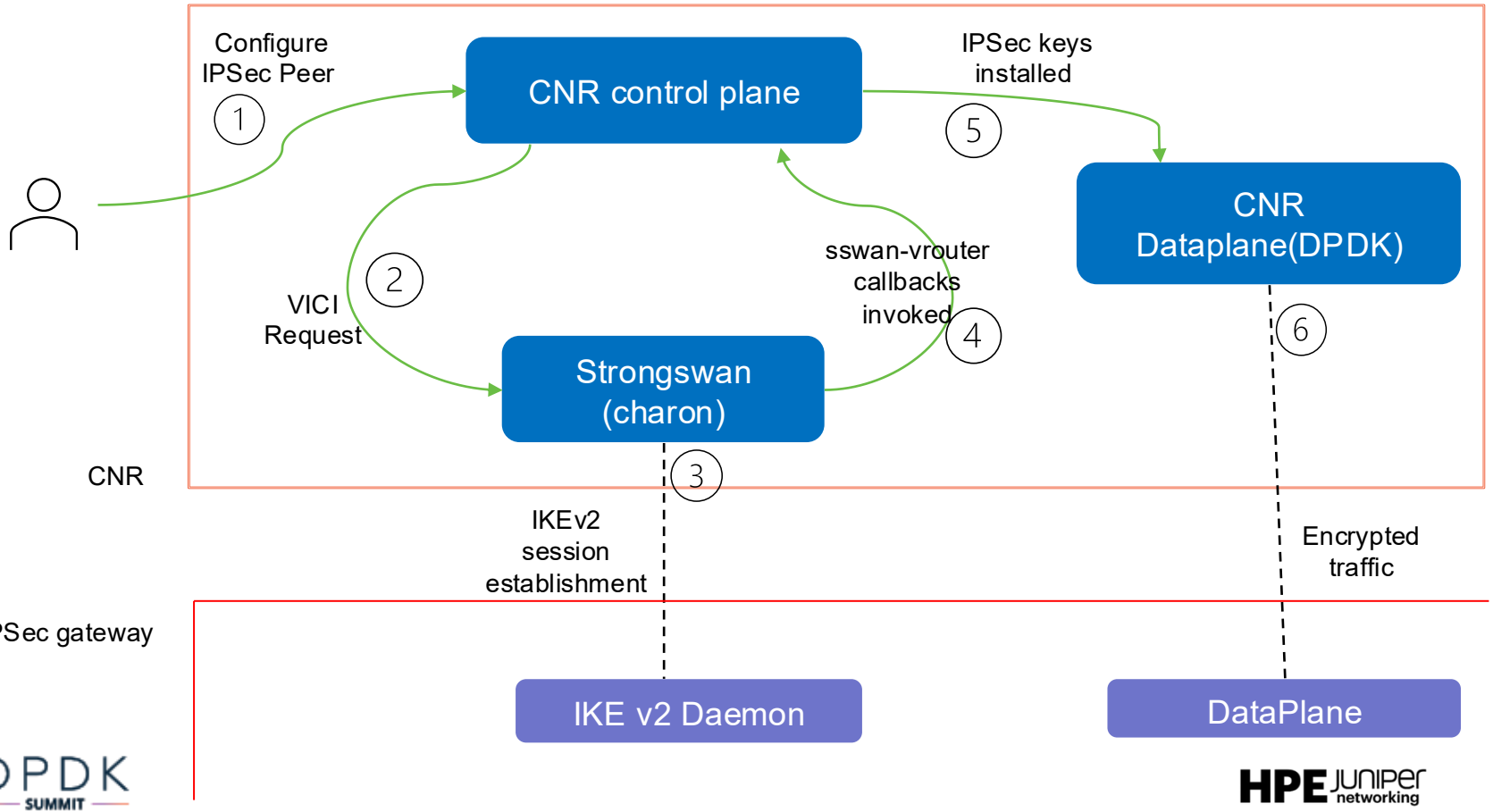


DPDK Cloud Native Router + Strongswan



Juniper Business Use Only

End to end flow for CNR + Strongswan



VICI to program Strongswan

- VICI acts as programmable control interface that translates the IPSec intent into dynamic strongswan IKEv2 and IPSec state
- The CLI configuration can be mapped to the VICI operations that initiate the tunnel

```
ipsec:
  ike-policies:
    - name: ike-policy-1
      ike-version: v2
      proposals:
        - encryption: aes256-gcm
          integrity: none
          prf: sha384
          dh-group: ecp384
      rekey-time: 14400
      mobike: false
      nat-traversal: false
      authentication:
        method: pre-shared-key
        local-id: 172.37.1.7
        remote-id: 172.37.1.8
        pre-shared-key:IGRhe+l34uOJO1rPaYx/kdHq90FYb7AG

  gateways:
    - name: ike-gw-1
      ike-policy: ike-policy-1
      local-address: 172.37.1.7
      remote-address: 172.37.1.8
      connection-type: unicast

  tunnels:
    - name: ipsec-tunnel-1
      gateway: ike-gw-1
      child-policy: child-policy-1
      enable: true
```

Sample python VICI code

- Create the session and load the Pre Shared Keys
- Update the configuration in a dict object and load the configuration
- Initiate the IKE tunnel

```
s = vici.Session()

# 1. Load PSK
s.load_shared({
    "type": "IKE",
    "data": "IGRhe+I34uOJO1rPaYx/kdHq90FYb7AG",
    "owners": ["172.37.1.7", "172.37.1.8"],
})

# 2. Load connection
s.load_conn({CONN_NAME: build_conn()})

# 3. Initiate IKE SA (this triggers CHILD SA)
s.initiate({"ike": CONN_NAME}):
```

Sswan-vrouter callback APIs

New plugin implement of these callbacks; forwarding each event to the CNR control plane instead of Linux xfrm.

Strongswan plugin interface:

- get_spi() ← allocate an SPI for a new SA
- add_sa() ← install negotiated SA into data plane
- update_sa() ← update SA on address change
- del_sa() ← remove SA from data plane
- flush_sas() ← remove all SAs (e.g. daemon restart)
- add_policy() ← install SP
- del_policy() ← remove SP
- query_sa() ← Check the state of already installed SA (rekey trigger)

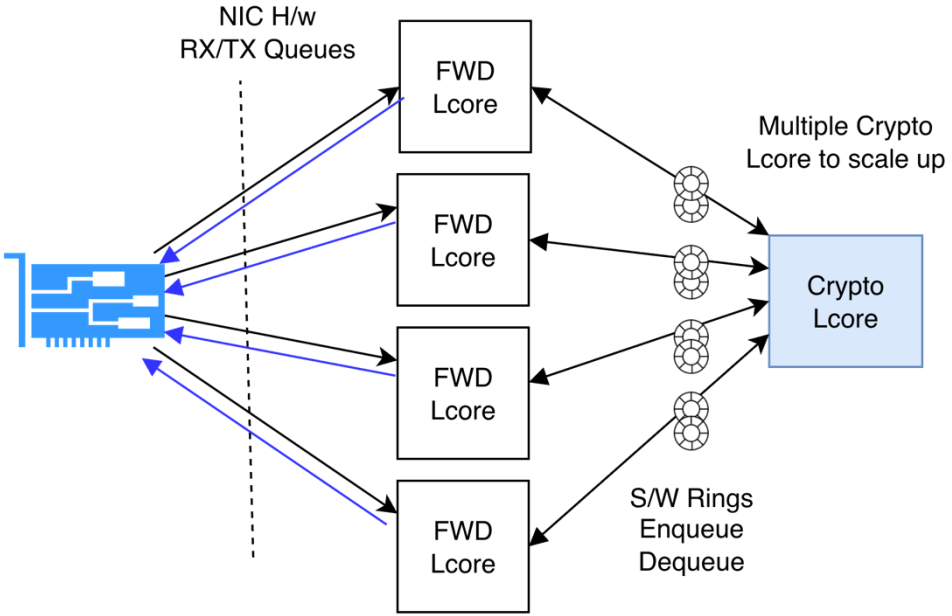
add_sa()

On add_sa() invocation, plugin delivers

- SPI, src/dst addresses, protocol (ESP/AH)
- Cipher: AES-GCM-128/256, AES-CBC + key material
- Auth: SHA2-256/512, SHA1-HMAC (null for AEAD)
- Mode: tunnel / transport, lifetime, replay window

DPDK crypto dev is eventually called upon the invocation of add_sa()

Packet Pipeline: Dedicated Crypto Cores



Thank You!
Q&A