P4-OVS Split Architecture

Namrata Limaye
Brian O'Connor
Ajay Kumar Dubey

Agenda

- Moving to P4-OVS Split Architecture
- Architecture Update
- Stratum changes and upstreaming
- FPGA Use Case for P4-OVS programming
- Reference Links

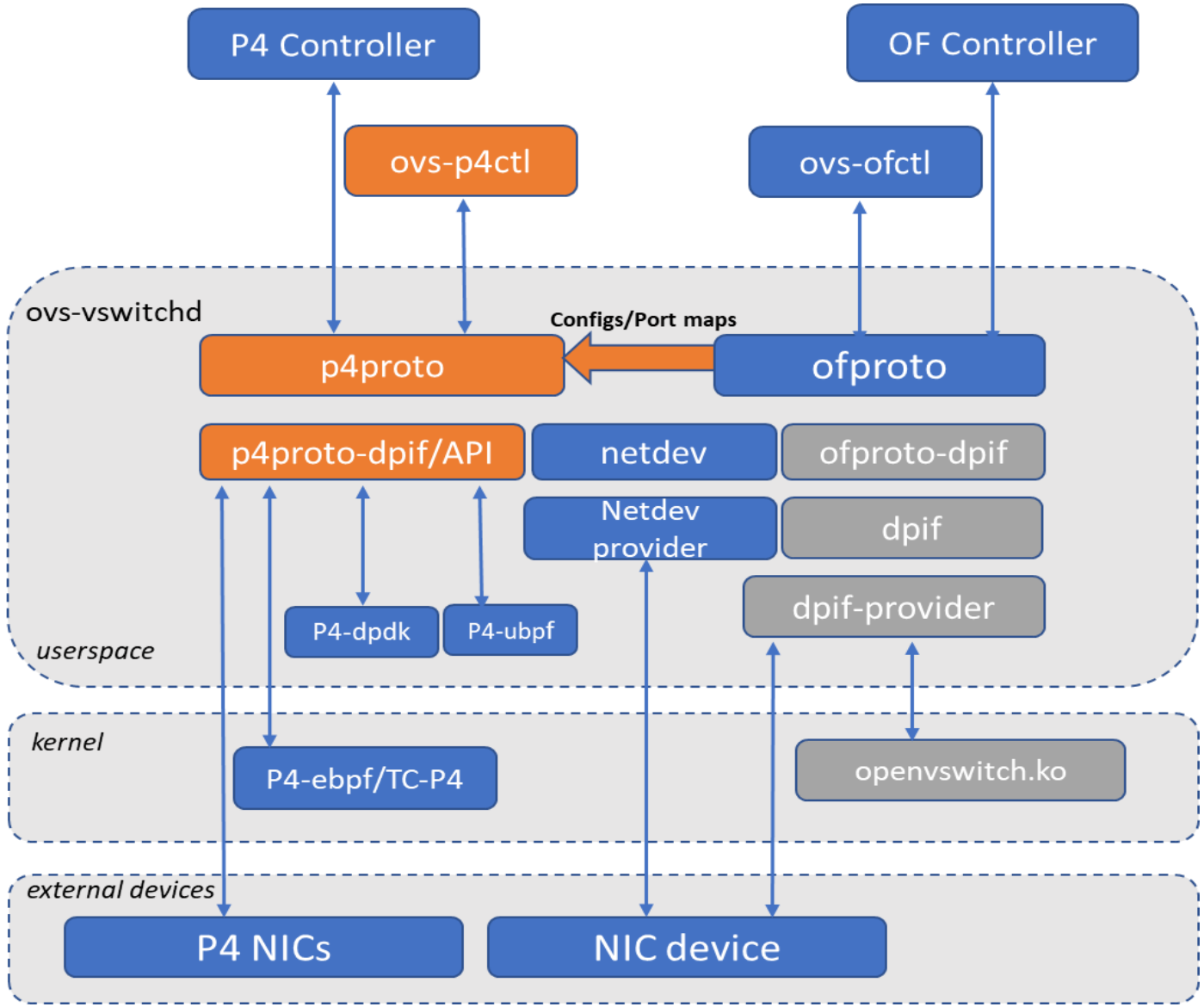
OVS to P4-OVS Architecture

Control planes:

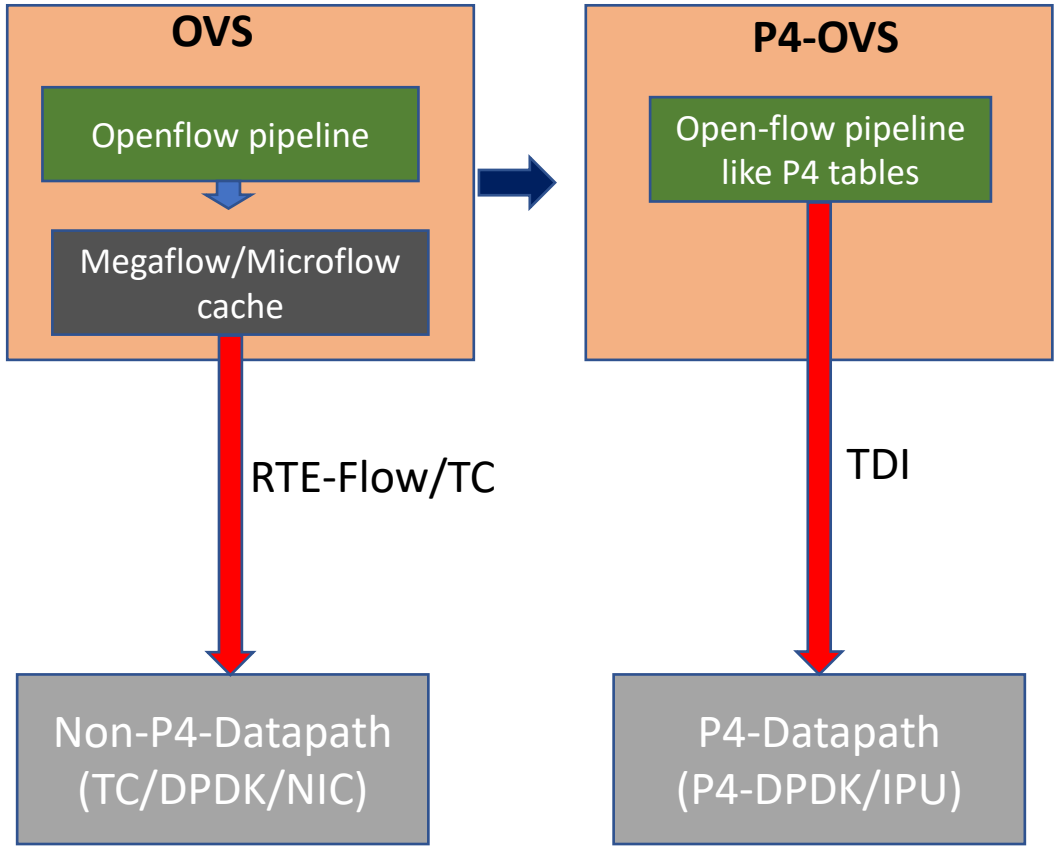
- OVS Reactive to P4-OVS Proactive
- P4Runtime + Openconfig – Configures P4 tables explicitly (E.g. Container load-balancing)
- Kernel Control Plane– Maps Kernel configurations (via SAI) to P4 Tables (E.g. ECMP w/ FRR)
- All three control planes can used to program the same P4 target.
- Multiple P4Runtime clients can connect and program different P4 pipelines

Data Planes:

- Physical NICs (Intel P4 IPUs (Mount Evans, FPGAs (Big Springs Canyon), Tofino 1, 2
- P4-DPDK (userspace)
- P4-ebpf (kernel)

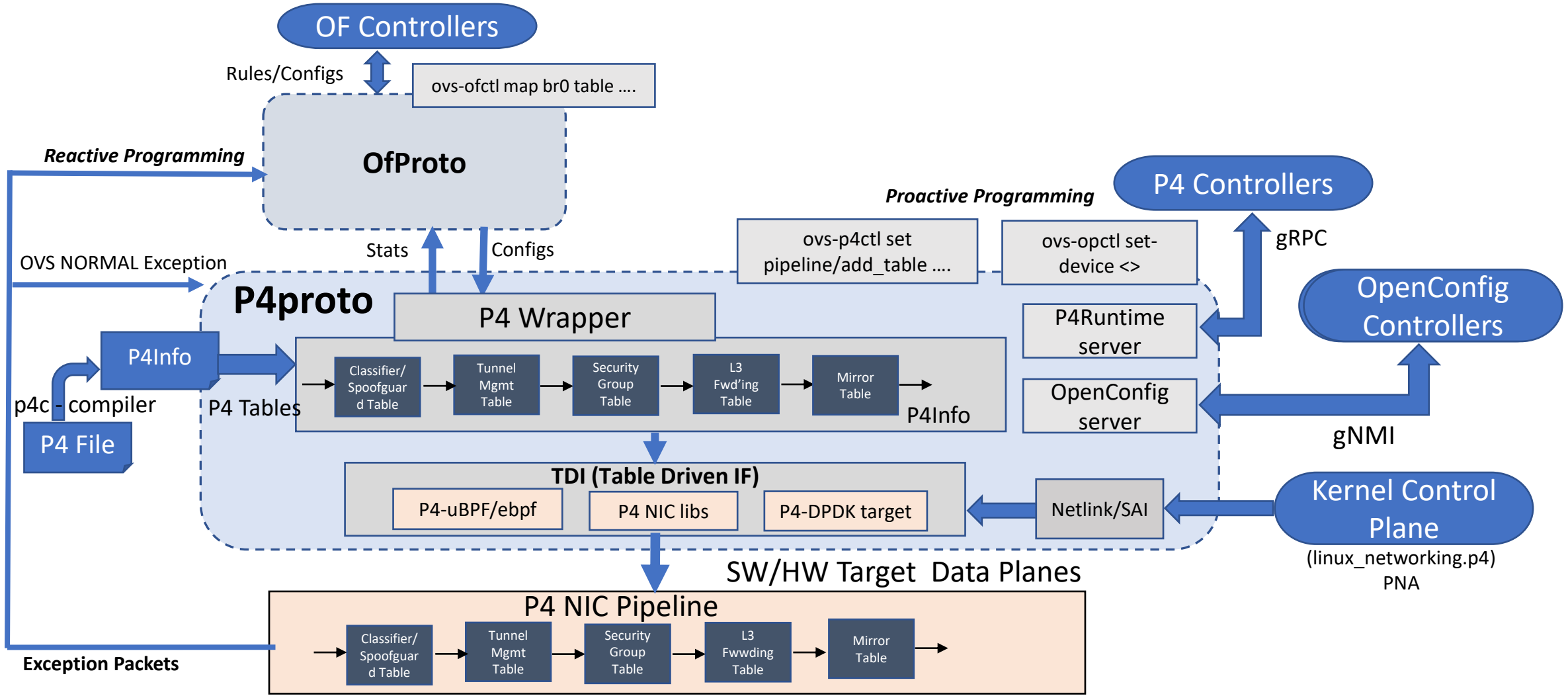


OVS acceleration to P4-OVS

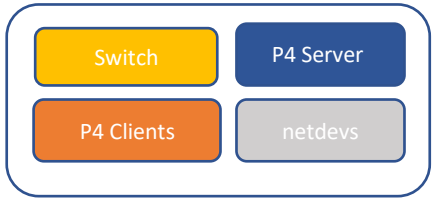


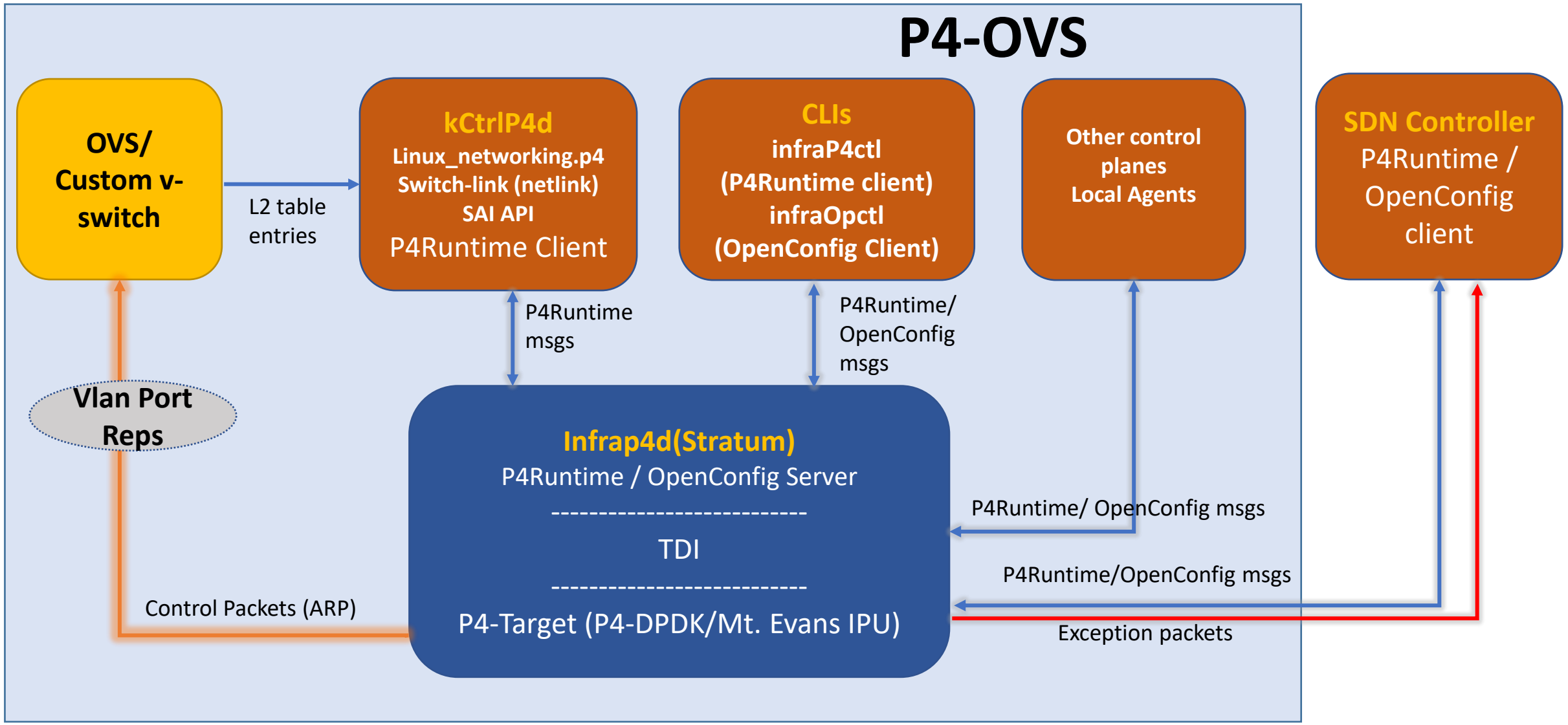
OVS	P4-OVS
Fixed protocols and network functions	User-defined protocols and network functions
Reactive offload – every packet comes up to SW	Proactive Offload – High level customer flows. Reactive is supported as well.
Cache offload	Openflow like P4 table offload
Exact Match only tables in SW/HW datapath	Longest Prefix Match, Ternary, Exact Match, Hash tables in SW/HW P4 datapath
Connection Tracking/Firewall happens in SW	Connection Tracking/Firewall compete offload to HW using PNA add-on-miss

Architecture Update – Current design

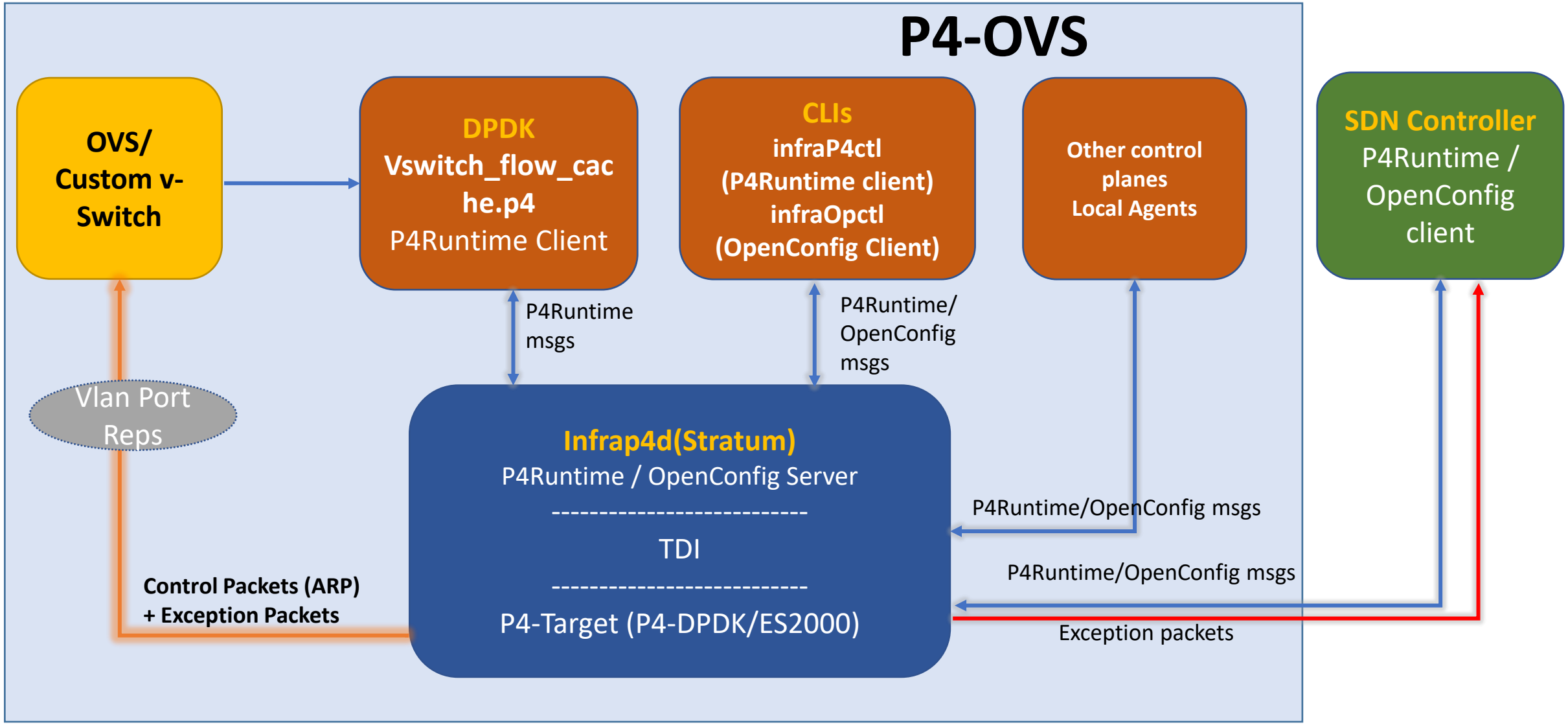


P4-OVS Split Architecture (Proactive)





P4-OVS Split Architecture (Reactive)



Split Architecture Components

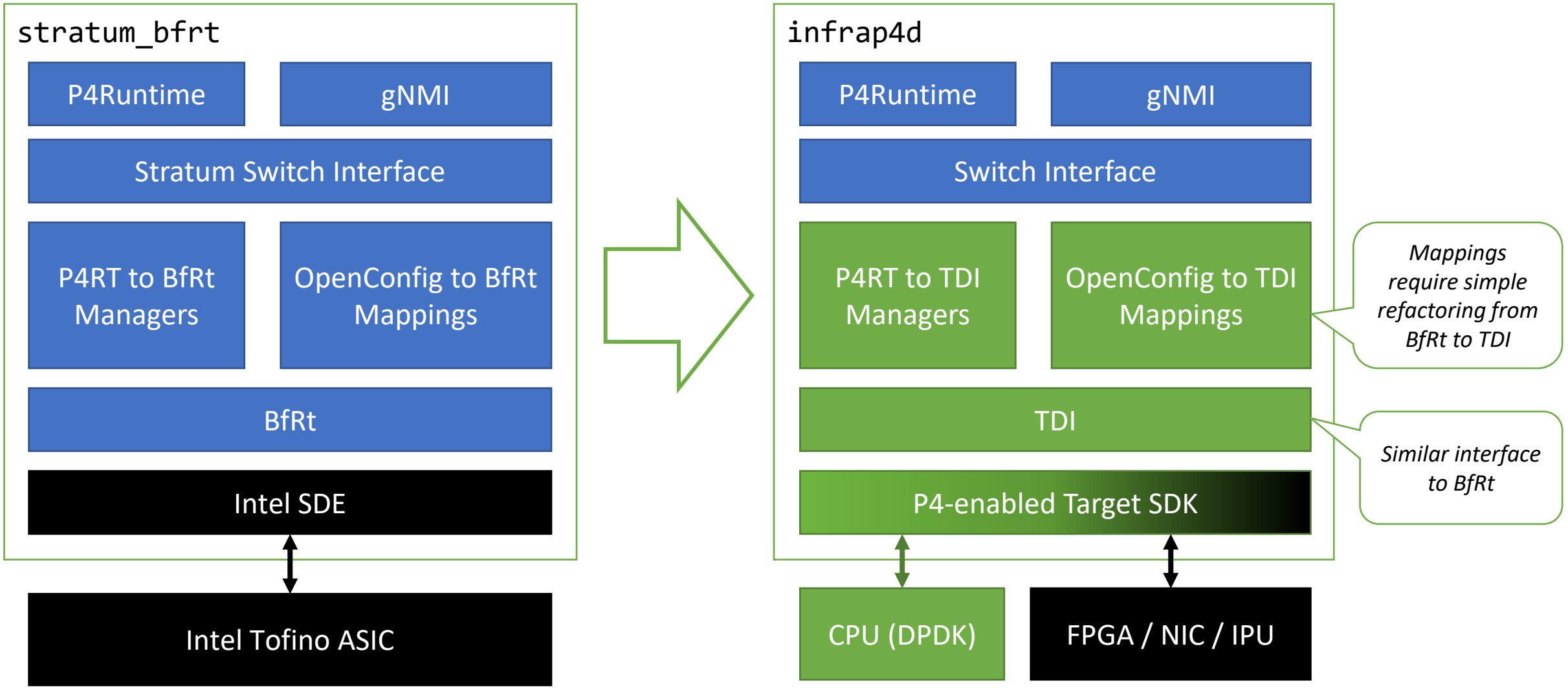
- **Stratum**
  - P4Runtime and OpenConfig Server Functionality
  - Minimize memory footprint in Stratum for IPU Targets
  - Integrates with TDI eventually
  - Interface to target backend (HW/SW) via TDI
  - Contains ovsdb-client for port reads from OVS.
  - Generates P4 Client library code in Go/Python/C++/Java/etc.
- **OVS (Or any vswitch)**
  - Provides L2 bridging for control packets
  - Any OpenFlow rules that are needed
  - Interface to kctrlp4d
- **kCtrlP4d**
  - Owner of linux_networking.p4 and drives kernel configurations via P4Runtime
  - Integrated P4Runtime client library generated by Stratum
  - Set linux_networking.p4 pipe and provide add/del rules in the P4 file
  - Interface to OVS for get L2 tables entries in inner bridge.
- **CLIs**
  - Infra-p4ctl – (ovs-p4ctl earlier) – Provides P4Runtime Client CLI
  - Infra-opctl – (gnmi-cli earlier) – Provides OpenConfig Client CLI
- **Any other control plane (storage, strongSwan, microservices, firewall, etc.)**
  - Integrated P4Runtime Client functionality generated by Stratum

Stratum Overview

- Provides P4Runtime and gNMI/OpenConfig interfaces
- P4 program independent architecture
- Lightweight and high performance
- Hardened and deployed in production on Intel® Tofino™
- Quarterly releases available from stable, open source main branch

https://github.com/stratum/stratum

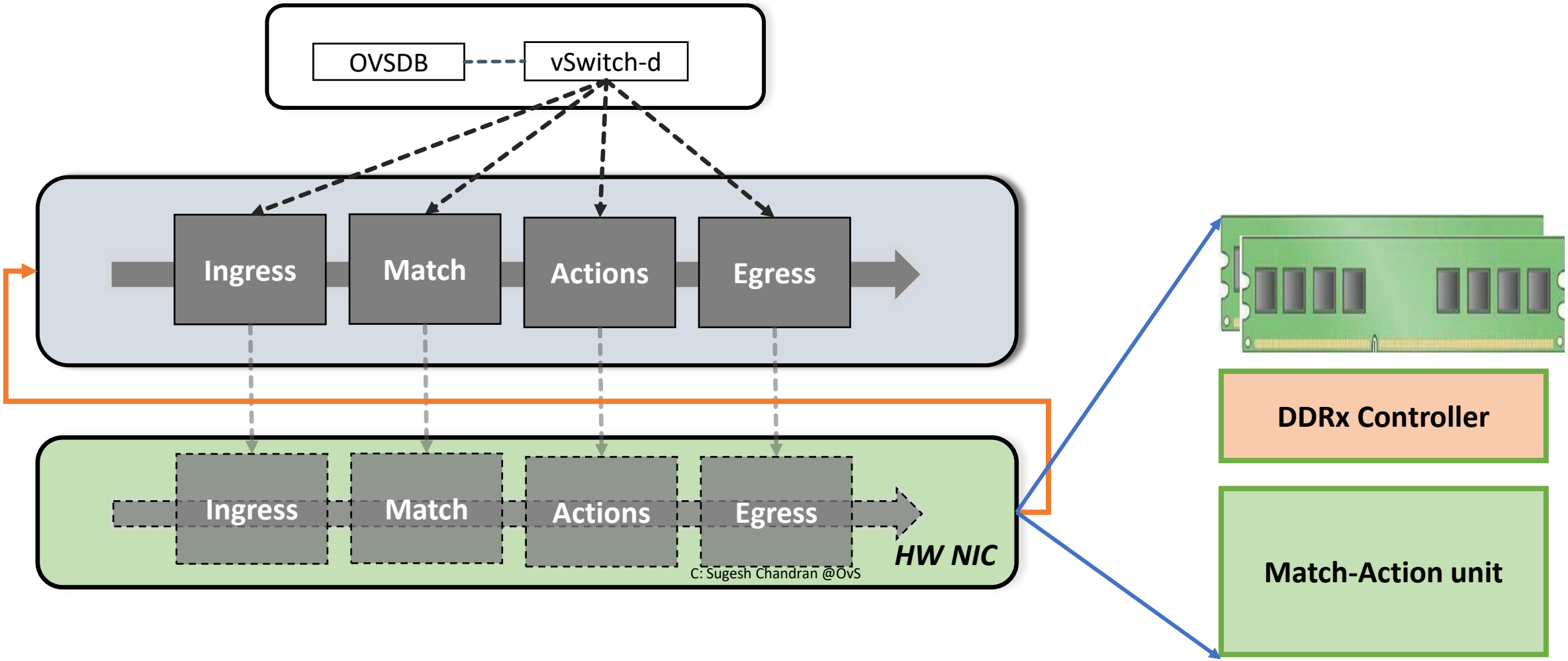
Adapting Stratum for P4-OvS



Stratum Next Steps

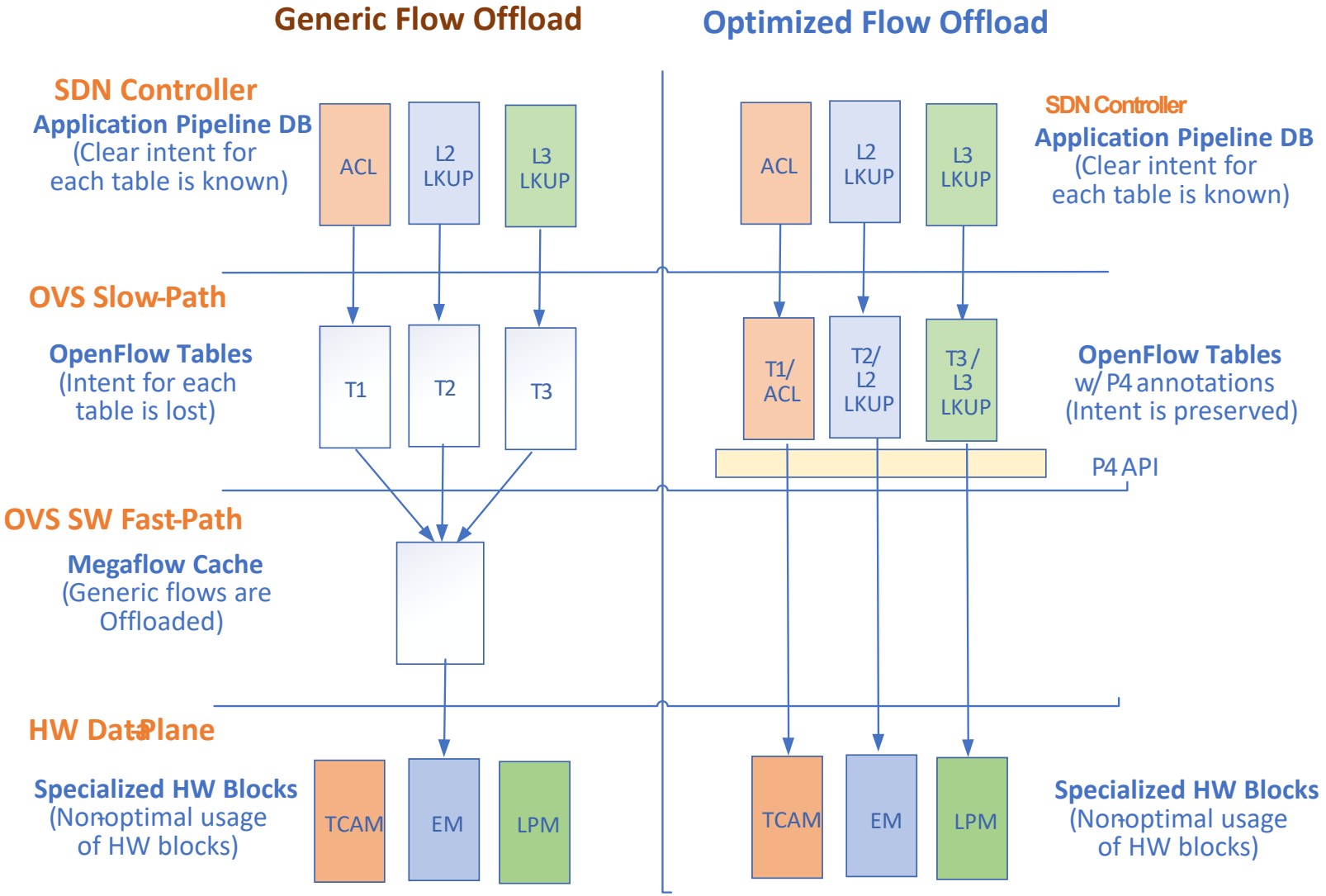
- TDI is already open sourced: https://github.com/p4lang/tdi
- Internal version of **`infrap4d`** has been developed for P4-OvS
  - Based on **`stratum_bfrt`**
  - BfRt-based code refactored to use TDI
- Later in 2022, plan to upstream new **`stratum_tdi`** target that will support:
  - P4-DPDK, Mt. Evens IPU, Tofino
  - FPGA, Additional Switching ASIC (future)
- Upstreaming brings many benefits
  - Provide a high(er) performance Stratum software switch target, based on DPDK
  - Allow P4-OvS to more easily consume Stratum fixes and enhancements
  - Path for unified software stack for different P4-enabled targets

FPGA Current – Reactive Offload Model

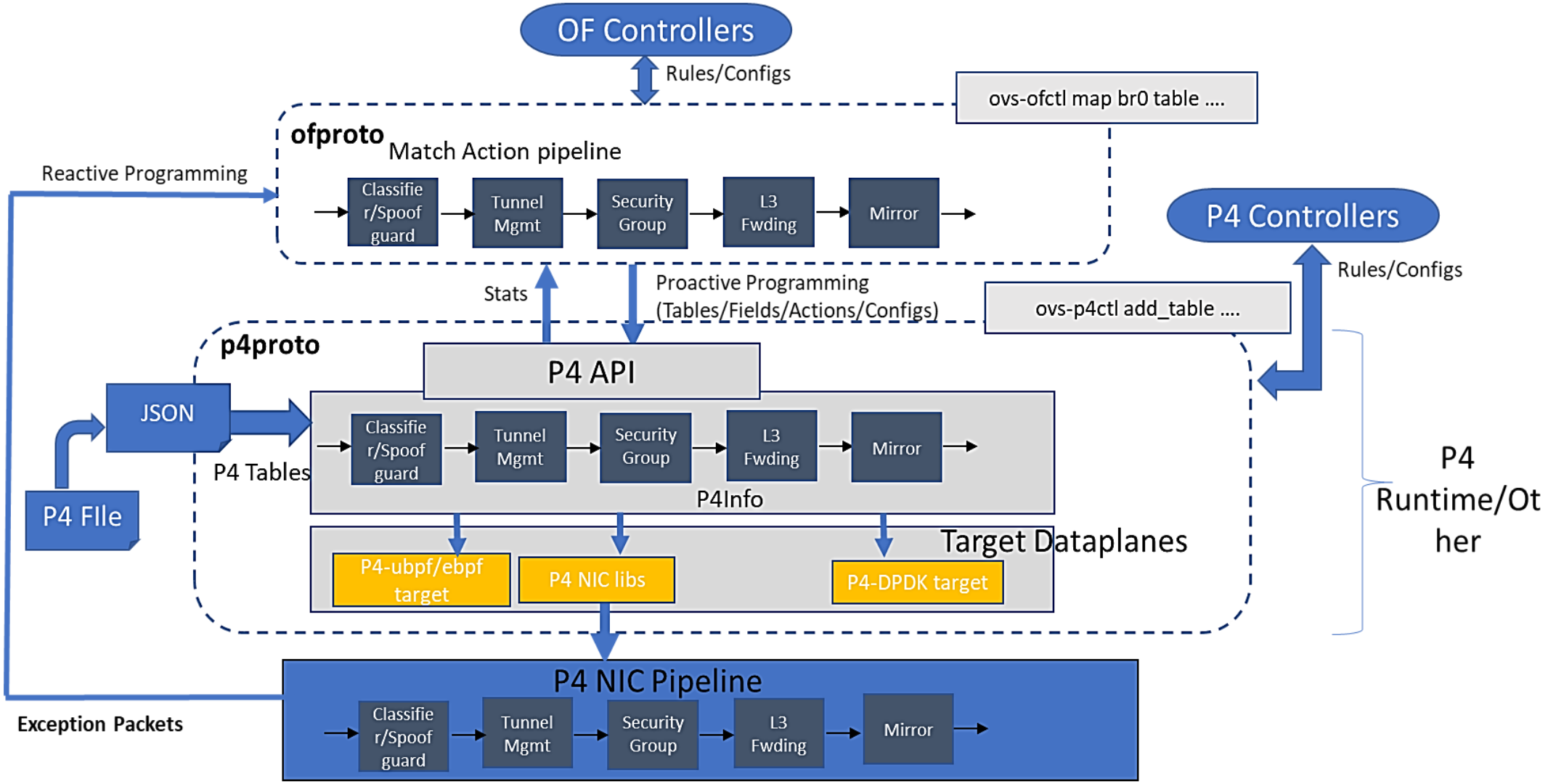


OVS FPGA offload challenges

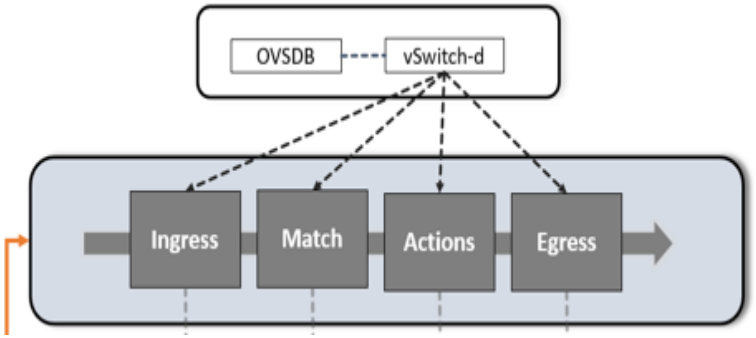
- The reactive behavior requires high slow path performance; maintaining high flow set up rate is a challenge
- The aggregation followed by disaggregation -> the intent is lost
- Better -> skip the caches and offload directly to the HW tables
- Solves the other associated problems

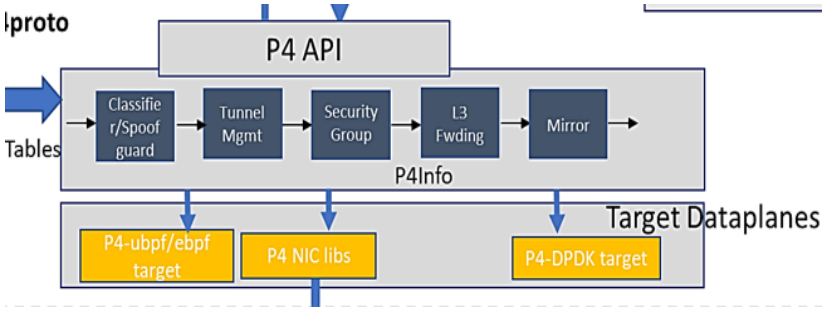


Current P4-OVS Proactive Flow

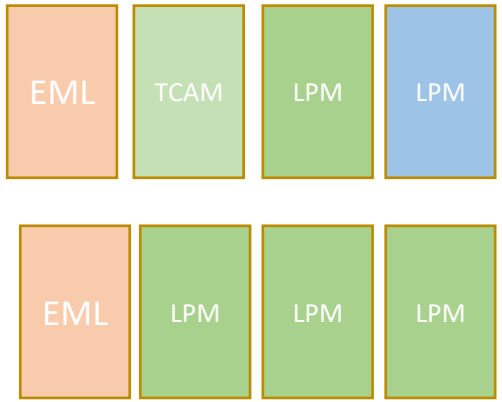


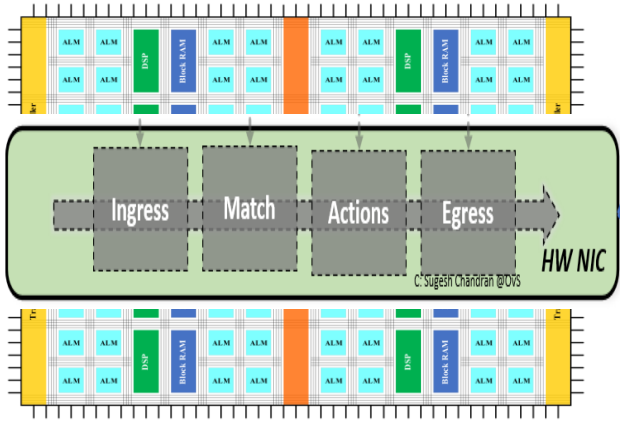
FPGA for Proactive Offload

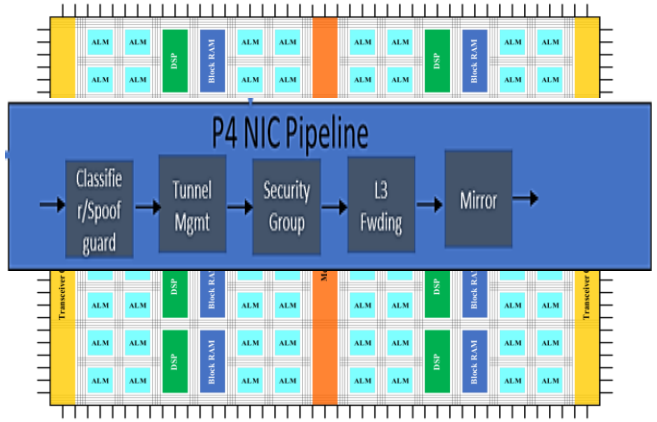




New Target with changed number and Types of processing elements







Key FPGA Advantages

- Ability to adapt to new algorithms and HW-SW division
- FPGA alone HW offload
- FPGA to augment CPU and SOCs
- Ability to recreate a Target with completely different sets of processing elements

Thank You

Reference Links:

https://ipdk.io/
https://github.com/ipdk-io
https://github.com/ipdk-io/ovs
https://github.com/ipdk-io/ipdk/tree/main/build/networking