Offloading data plane functions to the multi-tenant cloud infrastructure using P4

Tomasz Osiński / *Orange, WUT*

Mateusz Kossakowski / *Orange, WUT*

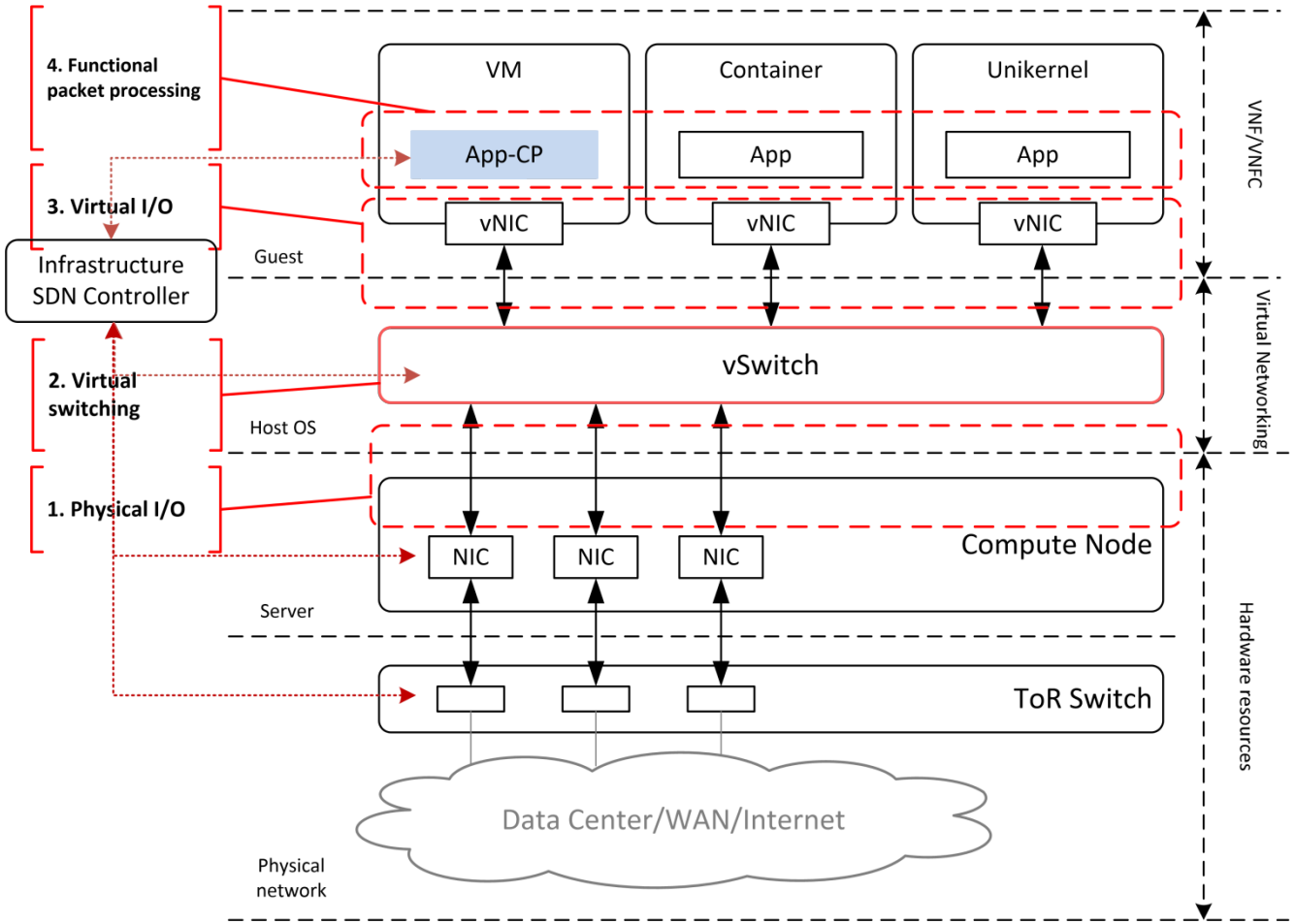
Halina Tarasiuk / *WUT*

Roland Picard / Orange

Agenda

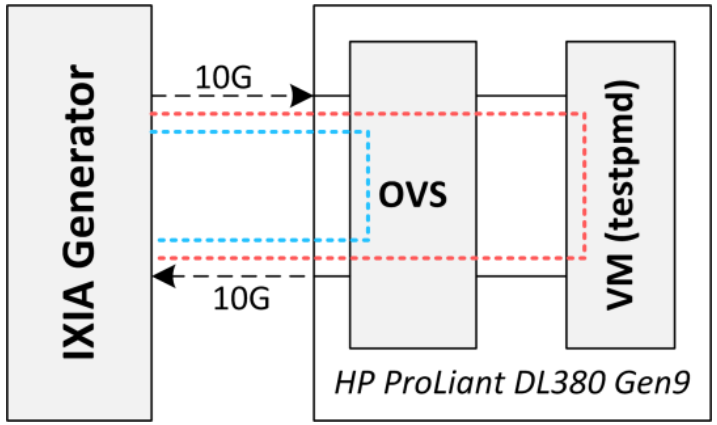
- Motivation & research objective
- Architecture of the VNF offloading framework
- Future work & research challenges
- Summary

Motivation & research objective

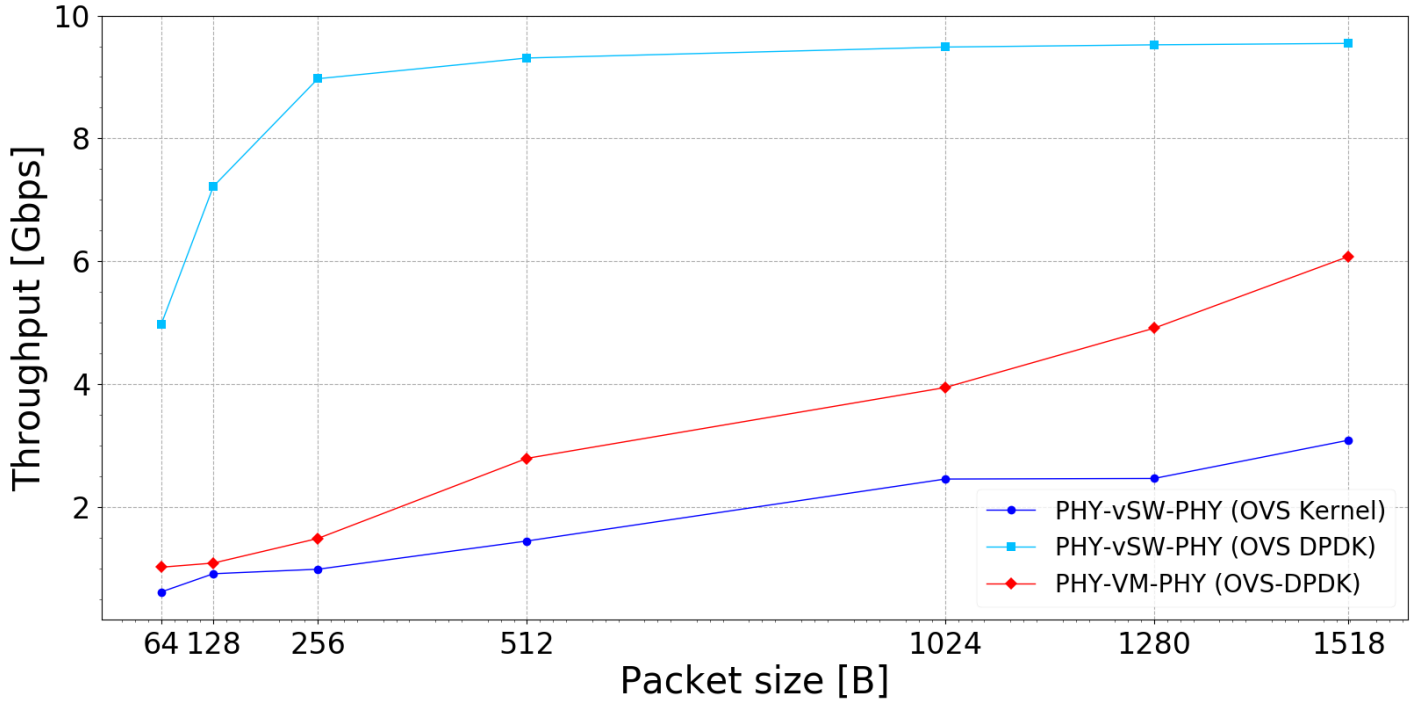


Motivation – performance gains

- **Test scenarios*:**
  - **PHY-VM-PHY (red line)**
  - **PHY-vSW-PHY (blue line)**



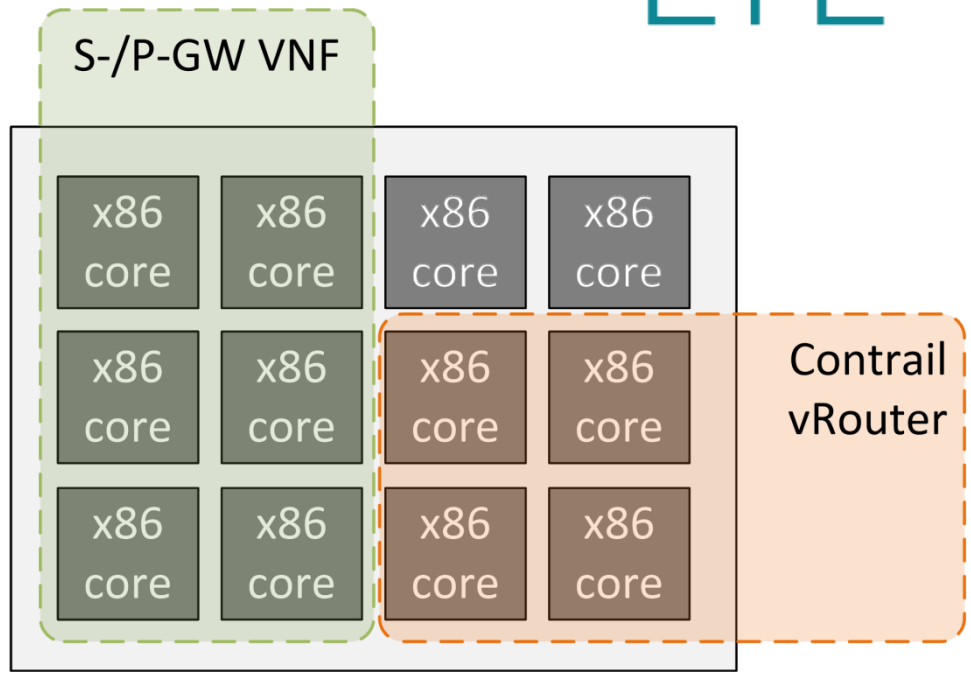
- **Performance results:**
  - **For large packets:**
    - **~6.07 Gbps (PHY-VM-PHY) vs. line-rate speed**
  - **For small packets:**
    - **~1 Gbps (PHY-VM-PHY) vs. ~5 Gbps (vSW)**



** not optimal DPDK configuration*

Why the software-based workloads are not sufficient for data plane?

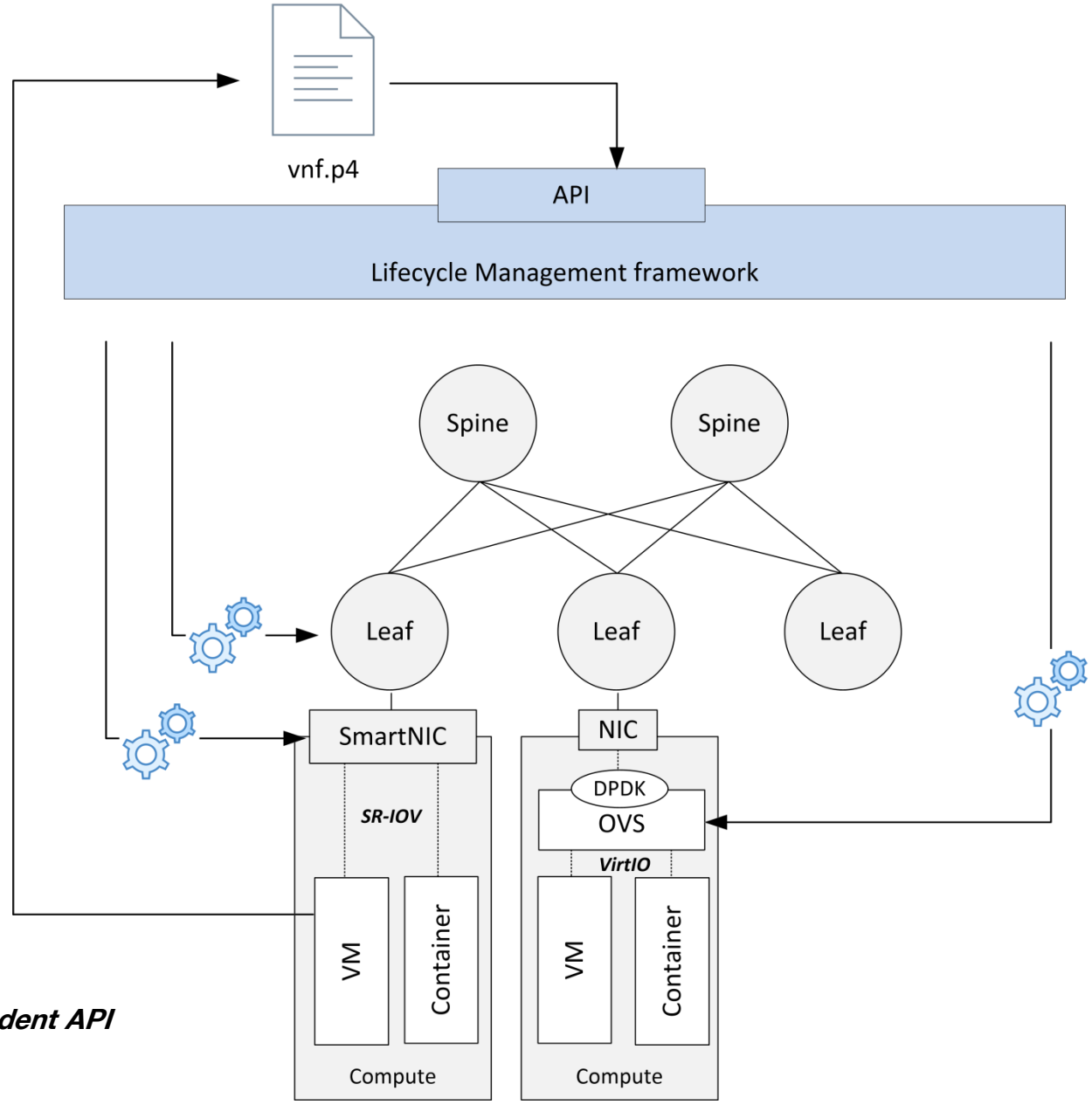
- **The case study of vEPC:**
  - **OpenStack + Contrail vRouter (DPDK)**
  - **DPDK-based vS-/P-GW component of vEPC**
  - **Compute node with 12 x86 CPU cores**
- **Key findings:**
  1. **Waste of resources**
  2. **High „cost per bit", need to scale out physical servers to provide better performance**



NFV Compute Node

Architecture of the VNF offloading framework

- **Standard set of APIs to offload data plane functions**
- **Design principles:**
  - **Use P4 for VNF dissaggregation**
  - **Multi-tenancy**
  - **Target-independent framework**
  - **P4Runtime-based CUPS interface***
- **Tenant's responsibilities:**
  - **Write P4 code for data plane functions**
  - **Choose „hookpoint" (execution platform)**
  - **Implement control plane for offloaded data plane function**



** ETSI NFV calls it „Network **Acceleration** Interface" with fixed, protocol-dependent API*

The VNF offloading framework – set of high-level APIs

- Full set of high-level APIs to manage lifecycle of P4 modules
- Implemented as PoC plugin for OpenStack Neutron using Service Function Chaining and BMv2 [1]
- REST API design:
  - Create/Request/Update/Delete of P4 module
  - Attach/Detach module
    - FlowFilter, e.g.:

      Match dstMAC <VM-MAC>, dstIP <VM-IP>, port 80
    - Configure/unconfigure flow rules for module

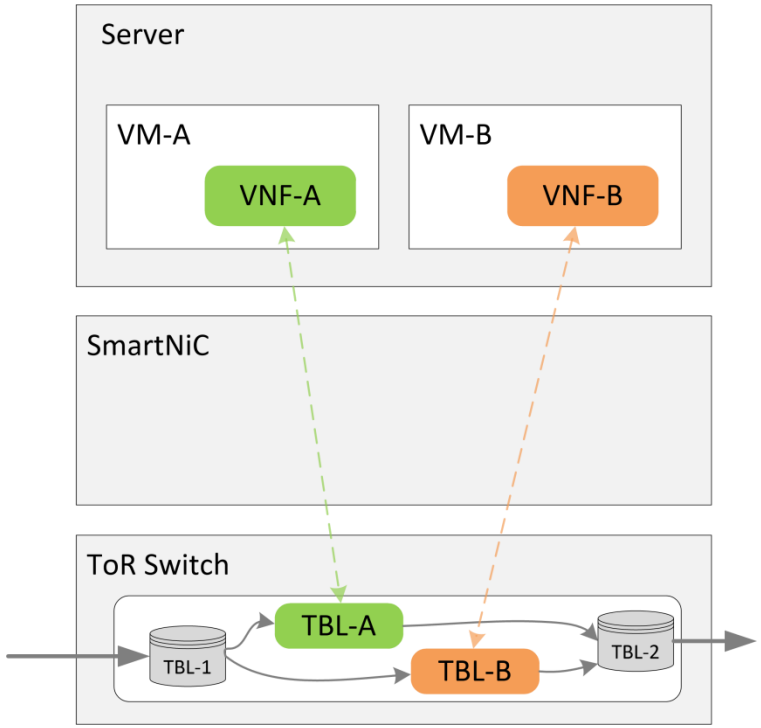
TABLE I
FUNCTIONALITY OF THE DPPX PLUGIN FOR OPENSTACK NEUTRON

Endpoint	Arguments	Description
POST /p4/modules	P4Program, NetworkID, TenantID	Create module based on P4 program, associated with given tenant and network
PUT /p4/modules/id	P4Program	Update given module with the new P4 program
DELETE /p4/modules/id	-	Delete given module
GET /p4/modules/id	-	Get information about given module
GET /p4/modules/	-	List information about all modules
PUT /p4/modules/id/attach	FlowFilter, VmID	Attach module with VM and push traffic matching flow filter
PUT /p4/modules/id/detach	VmID	Detach module from VM and stop pushing traffic to it
PUT /p4/modules/id/configure	FlowRules	Install flow rules for module
PUT /p4/modules/id/unconfigure	FlowRules	Remove given flow rules from module

[1] T. Osiński et al. 2019. DPPx: A P4-based Data Plane Programmability and Exposure framework to enhance NFV services. In Proceedings of the 5th IEEE Conference on Network Softwarization (NetSoft).

VNF offloading options – target (P4) platforms*

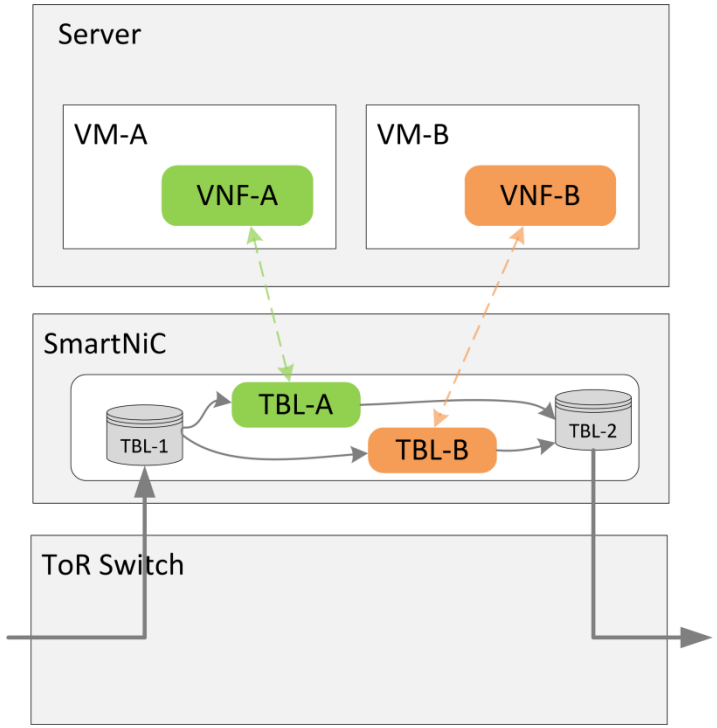
µVNFs in ToR switches



e.g. Barefoot Tofino / Tofino2 ASIC

µVNF as dedicated P4 table(s)
~ Tb/s perf.

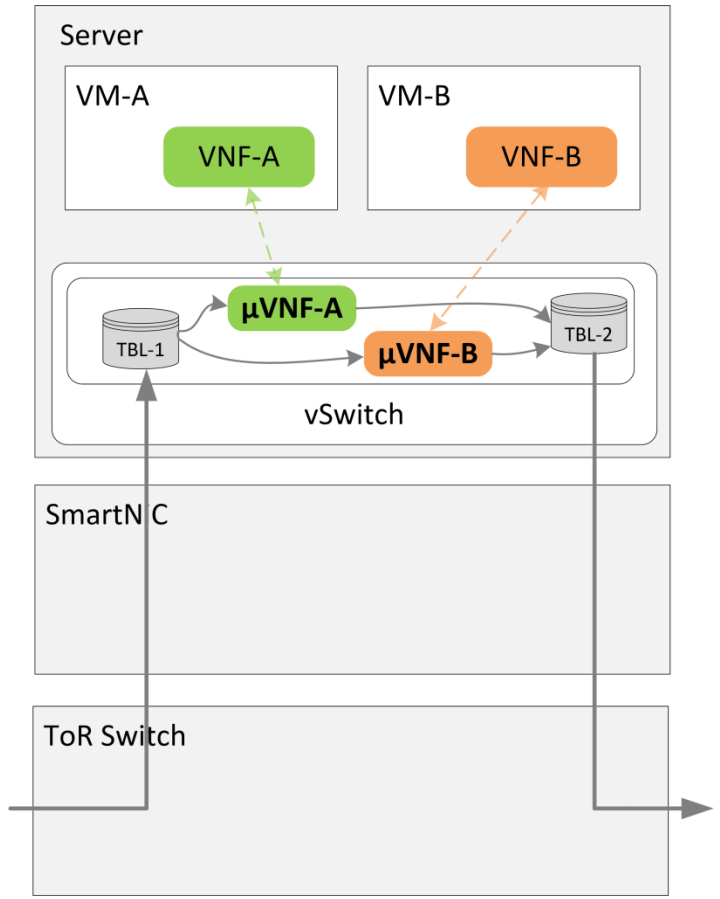
µVNFs in SmartNIC



e.g. Netronome Agilio CX

µVNF as dedicated P4 table(s) or C plugins
10-100 Gb/s perf.

µVNFs in software switch



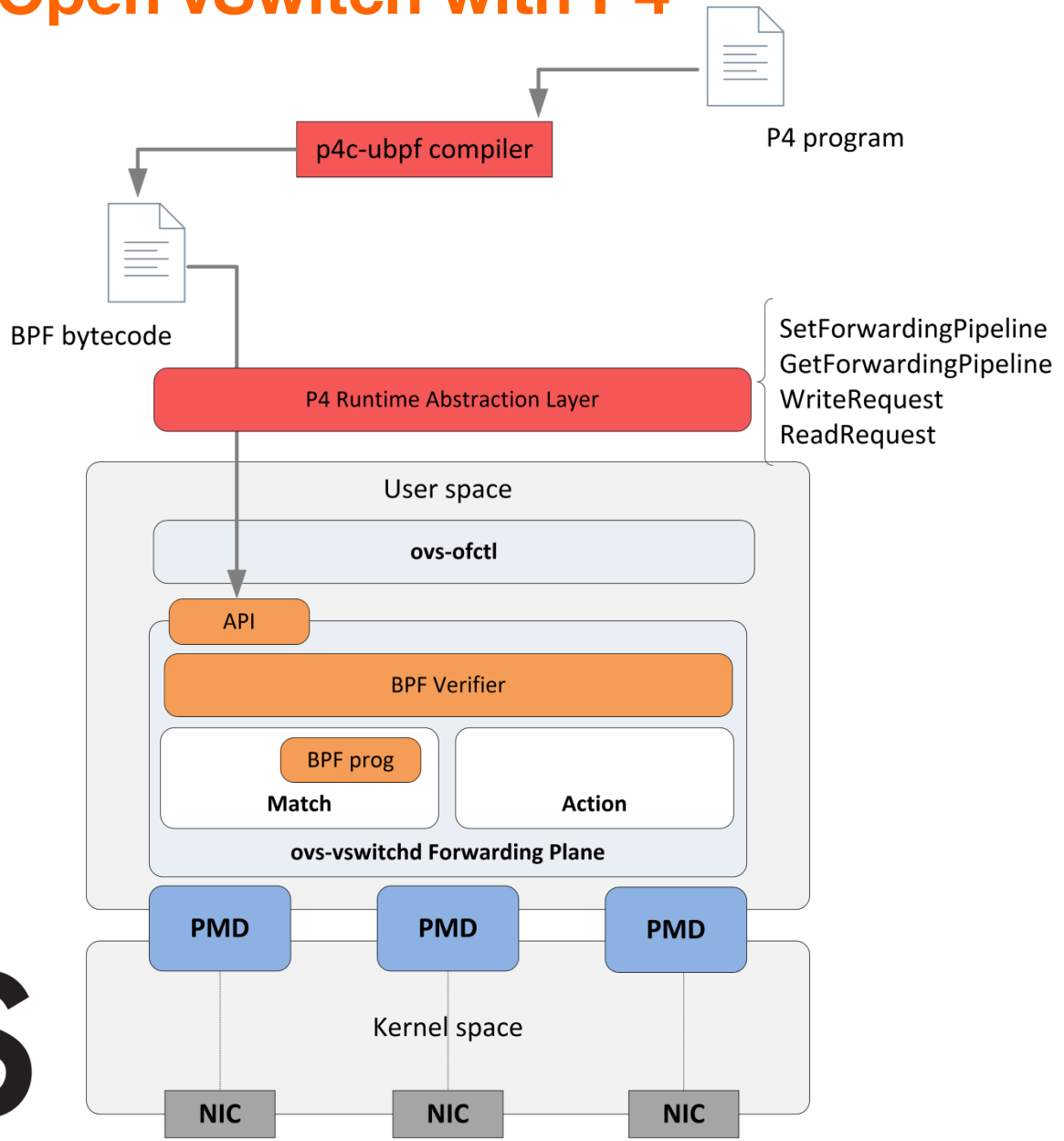
e.g. Open vSwitch

µVNF as OVS actions (BPF programs)
Tens of Gb/s perf.

** Based on „Open-NFP, Data Plane and VNF Acceleration", OPNFV Mini Summit, 12.06.2017*

„Programming runtime extensions for Open vSwitch with P4”

- Based on Oko switch [1] – extending OVS with stateful packet filters
- Oko v2:
  - Programmable actions
  - P4-to-uBPF compiler
  - Enhanced OpenFlow API to control BPF maps
  - P4Runtime Abstraction Layer
- To be published by the end of 2019



[1] Paul Chaignon et al. 2018. Oko: Extending Open vSwitch with Stateful Filters. In Proceedings of the Symposium on SDN Research (SOSR ’18). 13:1–13:13.

Open vSwitch

Use cases…

Use case #1: anti-DDoS as middlebox function in software switch

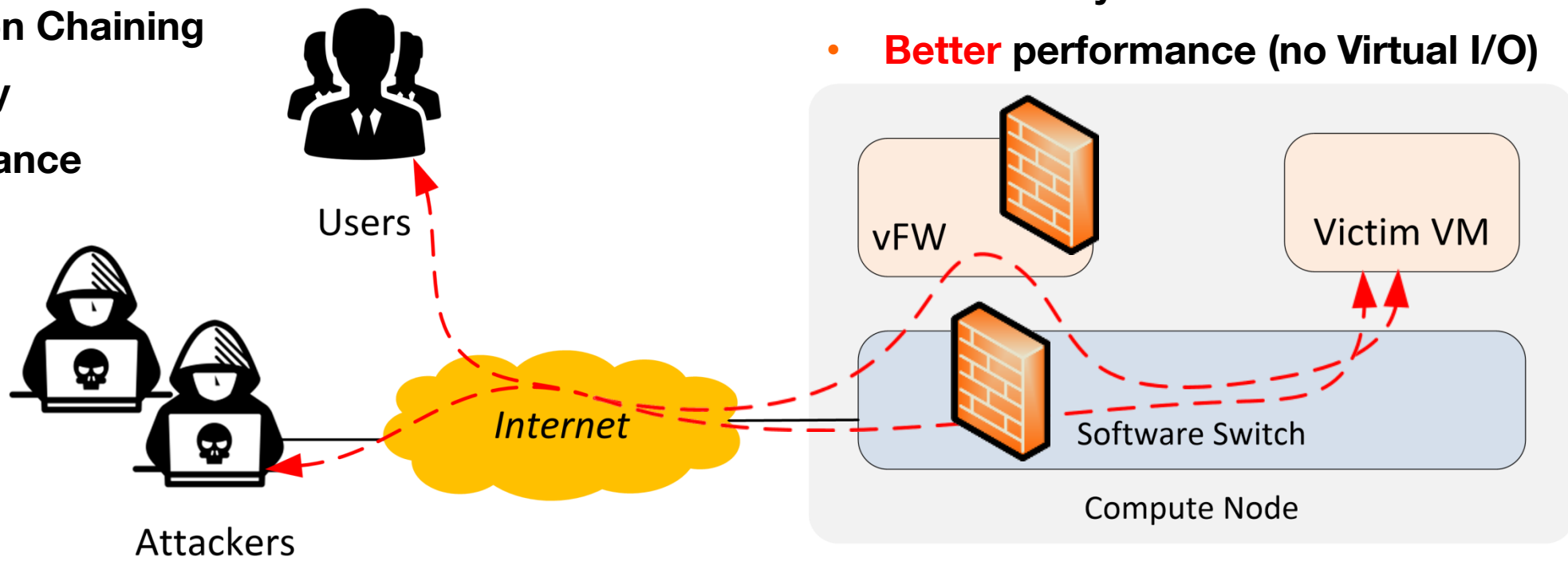
DDoS attack: „TCP SYN Flooding with Spoofing”

„vFW as Middlebox VM”

- Service Function Chaining
- Additional delay
- Worse performance

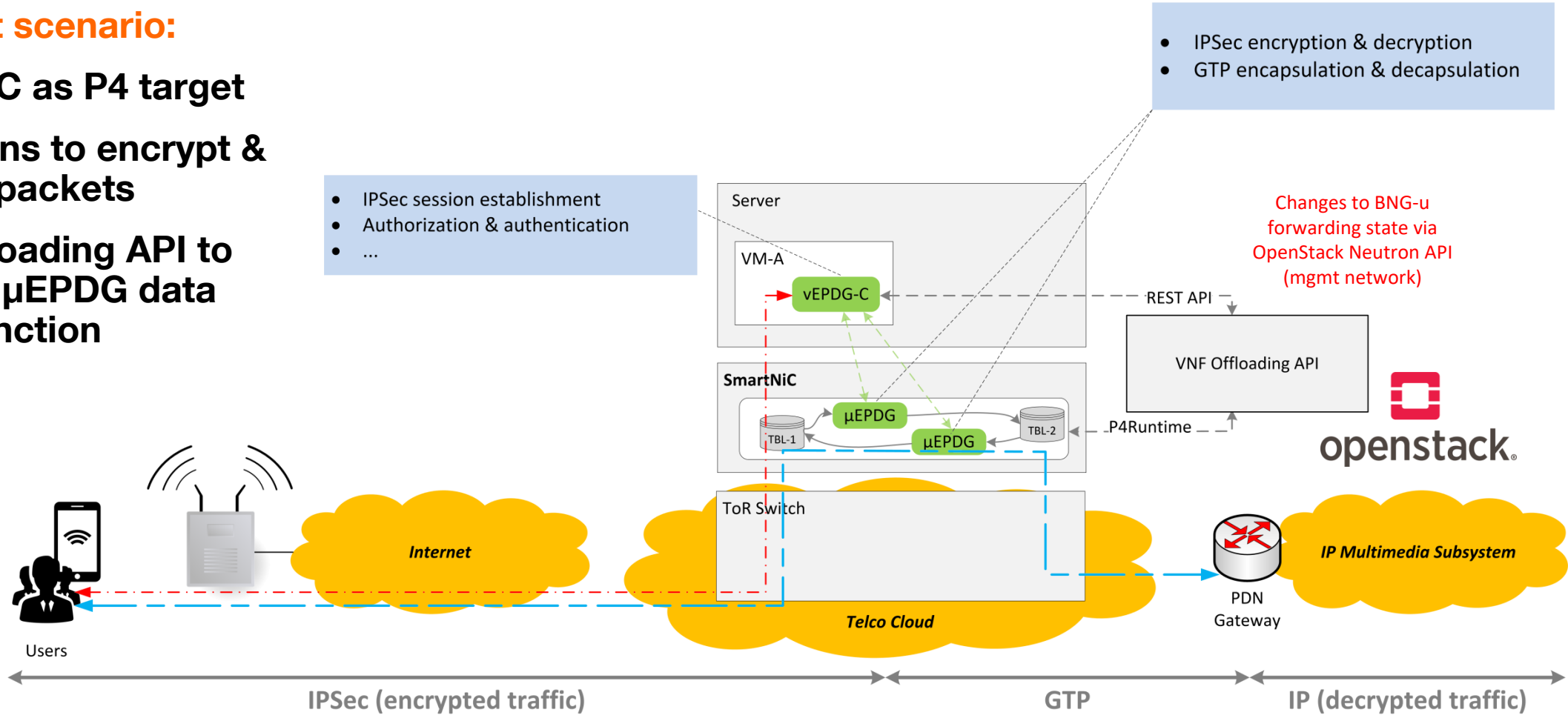
„vFW as middlebox function embbeded in software switch”

- **No** traffic mirroring/bypassing/chaining!
- Expected:
  - **Lower** delay
  - **Better** performance (no Virtual I/O)



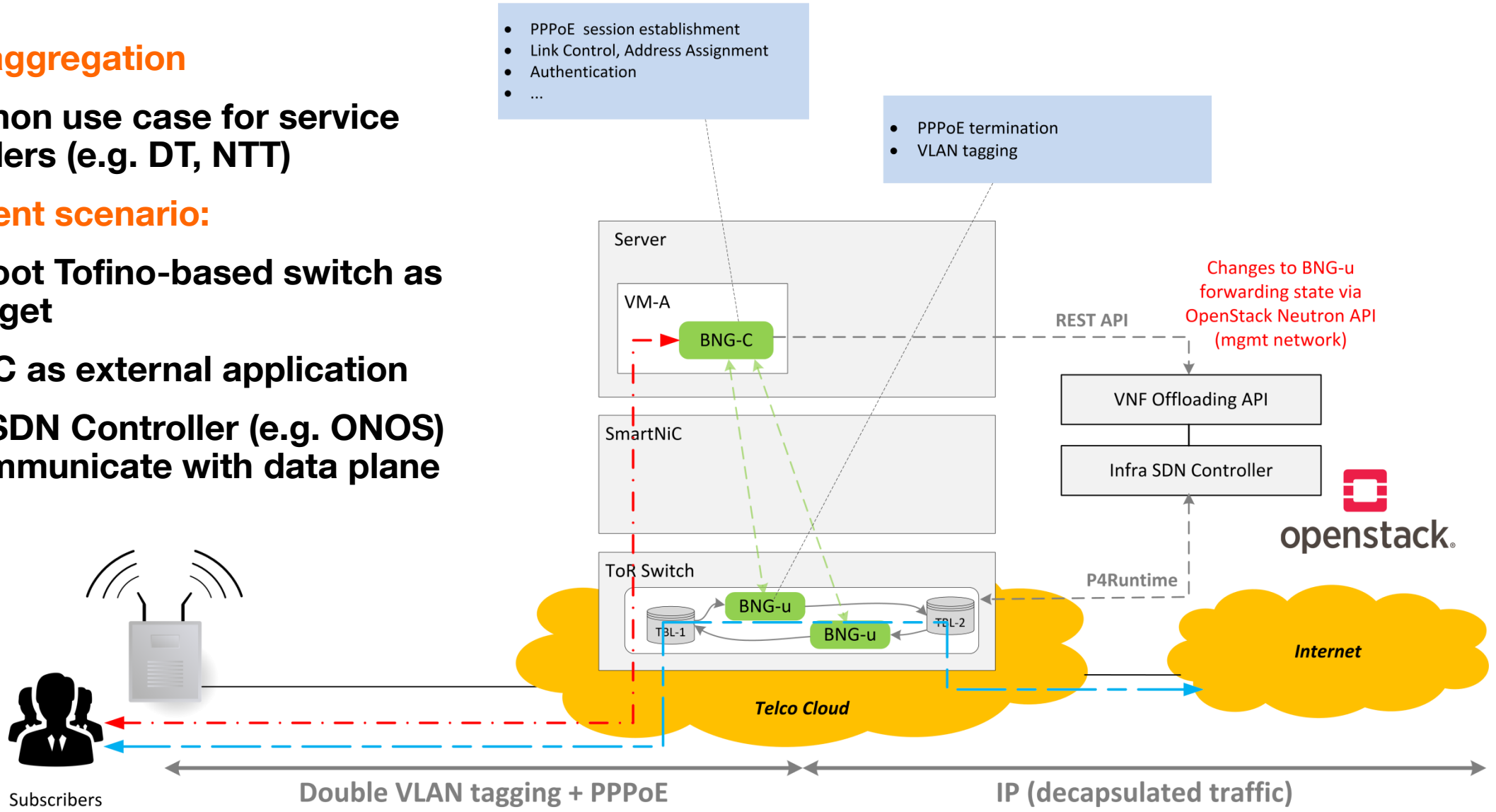
Use case #2: vEPDG disaggregation on SmartNiC

- WiFi Calling network service
- Deployment scenario:
  - SmartNIC as P4 target
  - P4 Externs to encrypt & decrypt packets
  - VNF Offloading API to manage µEPDG data plane function



Use case #3: vBNG disaggregation on white-box switch

- **BNG Disaggregation**
  - **Common use case for service providers (e.g. DT, NTT)**
- **Deployment scenario:**
  - **Barefoot Tofino-based switch as P4 target**
  - **BNG-C as external application**
  - **Infra SDN Controller (e.g. ONOS) to communicate with data plane**



Open challenges

- How to provide isolation between tenant’s code in the P4 switch?
  - Soft isolation vs. Hard isolation
- How to ensure stability of the platform?
  - Program verification tools
  - Limited set of capabilities provided to tenants (architecture model, disable forwarding between ports, packet cloning, etc.)
  - The framework responsible for forwarding & routing (isolating traffic of tenants)
- How to provide modularity and in-place software upgrade?
  - Compile time modularity, e.g. Hyper4 [1], ClickP4 [2]
  - Platform-level modularity, e.g. eBPF, XDP, Oko v2
- What range of VNF’s functionalities can we offload?
  - TLS, L7 Application Firewall, DPI, etc. ?
  - Currently, we need to rely on P4 externs (next session about P4DNS)

[1] David Hancock et al. 2016. HyPer4: Using P4 to Virtualize the Programmable Data Plane. In *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies* (CoNEXT '16). ACM, New York, NY, USA

[2] Yu Zhou and Jun Bi. 2017. ClickP4: Towards Modular Programming of P4. In *Proceedings of the SIGCOMM Posters and Demos* (SIGCOMM Posters and Demos '17). ACM, New York, NY, USA, 100-102

Summary

- We proposed the common VNF offloading framework with standard set of APIs to disaggregate network-intensive VNFs
- The purpose of this talk is to animate the work on the common, standardized and open-source VNF offloading framework
- Prospective research directions:
  - Investigate the use of hardware platforms to offload VNFs
    - vEPDG using SmartNiC
    - vBNG using Barefoot Tofino
  - Standardize APIs under the ETSI NFV umbrella

Thank you for attention!

tomasz.osinski2@orange.com
mateusz.kossakowski@orange.com