SESSION 4
Use cases of ONOS+Stratum

Use cases

1. Trellis: Silicon-independent fabric
2. BNG offloading in SEBA
3. S/PGW offloading in M-CORD

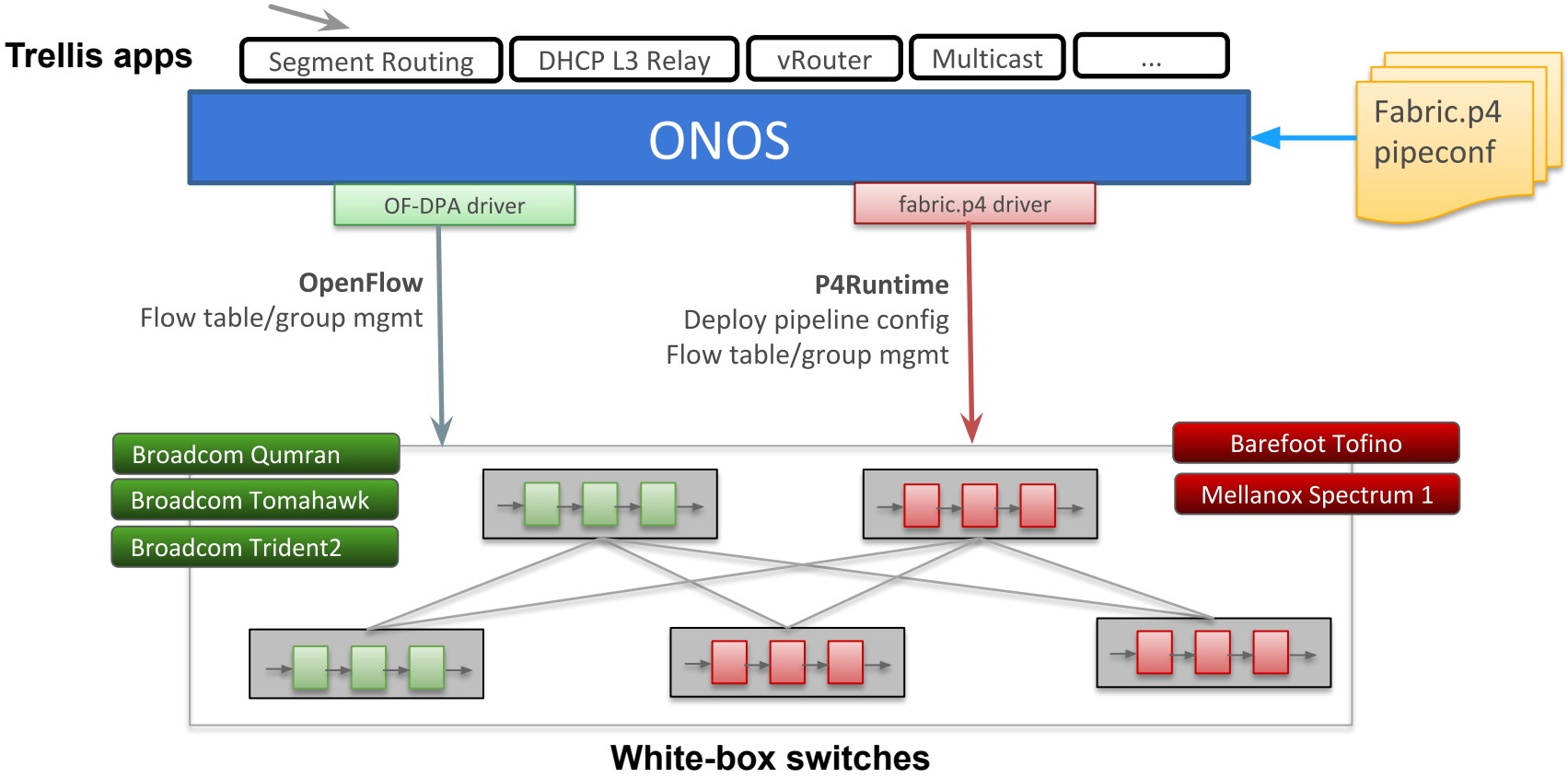
Silicon-independent fabric

Trellis – Multi-purpose Leaf-Spine Fabric

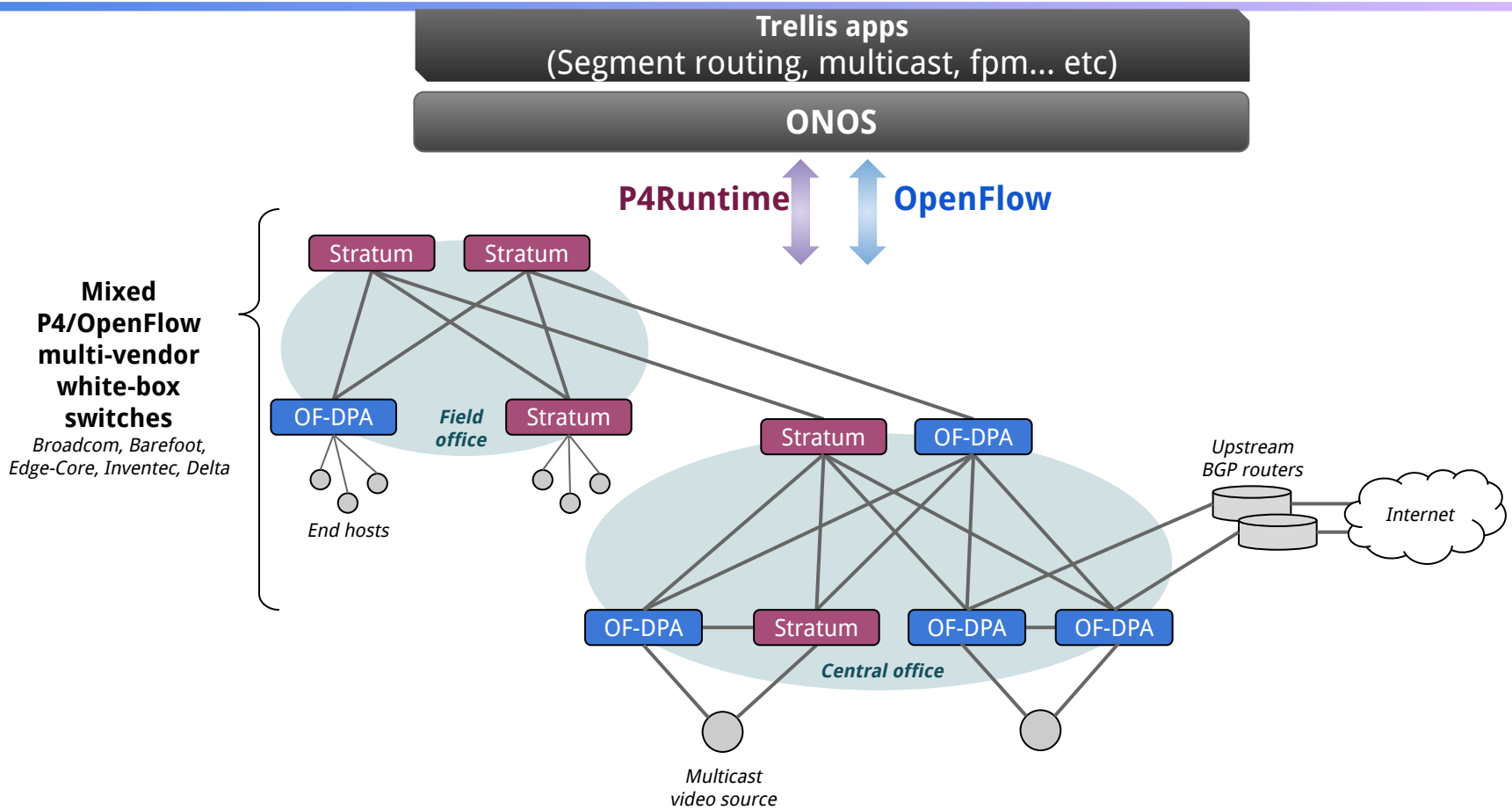
- **Prominent example of ONOS apps**
  - In production at tier-1 operator in the US
- **Designed for NFV and access/edge applications**
  - Built with white-box switches, open source software, SDN based
- **Extensive feature set**
  - Bridging/VLANs, IPv4/v6 unicast and multicast routing, DHCP-relay, pseudowires, QinQ, vRouter & more
- **Initially designed to work with Broadcom silicon**
  - Using the OF-DPA OpenFlow-based agent

Trellis & P4

Pipeline-agnostic apps - use ONOS FlowObjective API



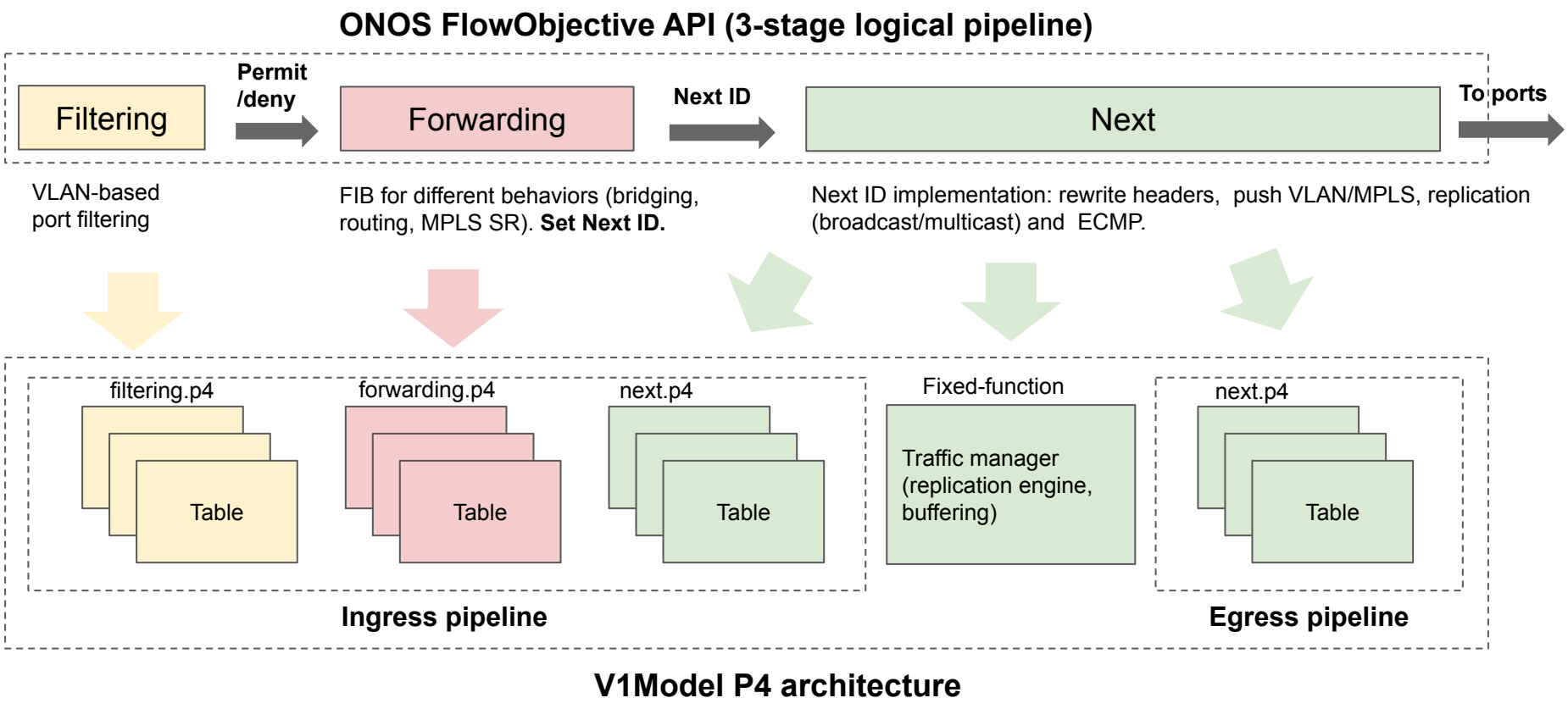
Trellis with mixed fabric demo (2018)



fabric.p4

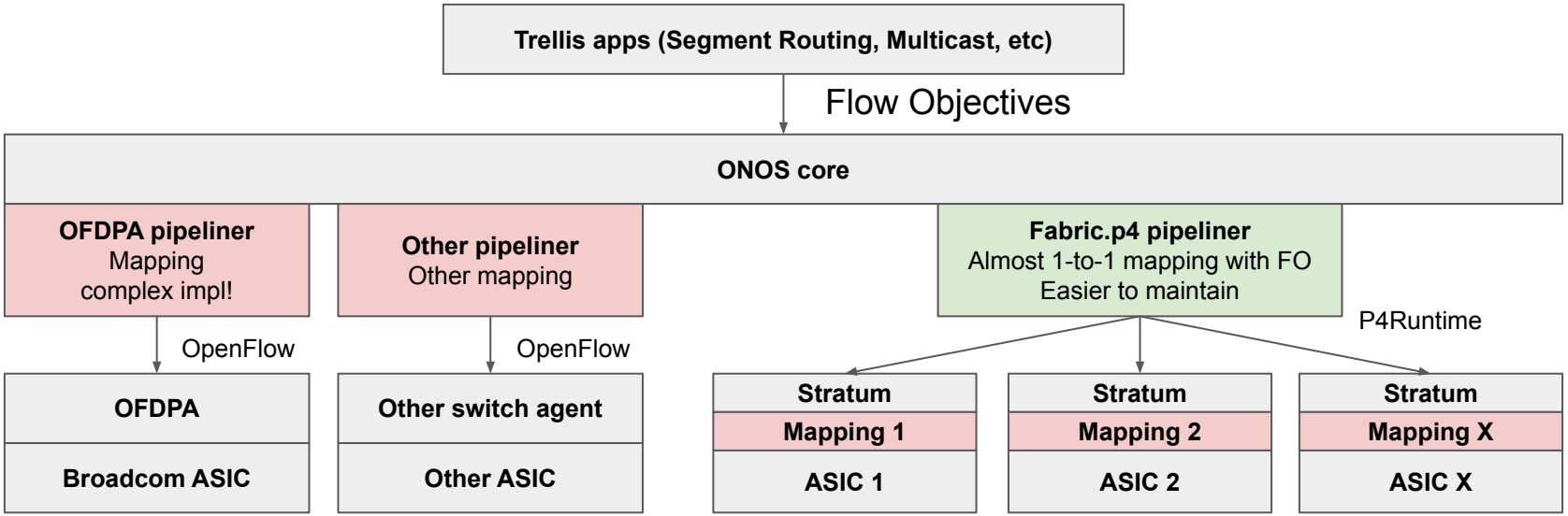
- **P4 implementation of the Trellis reference pipeline**
  - Inspired by Broadcom OF-DPA pipeline
  - Tailored to Trellis needs (fewer tables, easier to control)
- **Bring more heterogeneity in Trellis with P4-capable silicon**
  - Works with both programmable and fixed-function chips
  - Logical simplified pipeline of L2/L3/MPLS features
  - Any switch pipeline that can be mapped to fabric.p4 can be used with Trellis
- **Extensible open-source implementation**
  - https://github.com/opennetworkinglab/onos/.../fabric.p4

Design rationale: simplify control plane development



P4 provides “easier” silicon independence

- **Mapping FlowObjective to new HW is hard!**
  - Underspecified/ambiguous pipeline abstraction
- **Any switch ASIC that can be mapped to fabric.p4 can be used with Trellis**
  - Both programmable and fixed function
- **With P4, the mapping effort left to compilers, not ONOS drivers**
  - E.g. using Stratum p4c-fpm backend for Broadcom ASICs



OF-DPA vs fabric.p4 mapping driver in ONOS

fabric.p4

```
$ cd onos/pipelines/fabric/.../pipeliner
$ wc -l *.java
     106 AbstractObjectiveTranslator.java
     284 FabricPipeliner.java
      58 FabricPipelinerException.java
     237 FilteringObjectiveTranslator.java
     252 ForwardingFunctionType.java
      43 ForwardingFunctionTypeCommons.java
     284 ForwardingObjectiveTranslator.java
     498 NextObjectiveTranslator.java
     209 ObjectiveTranslation.java
      20 package-info.java
    1991 total
```

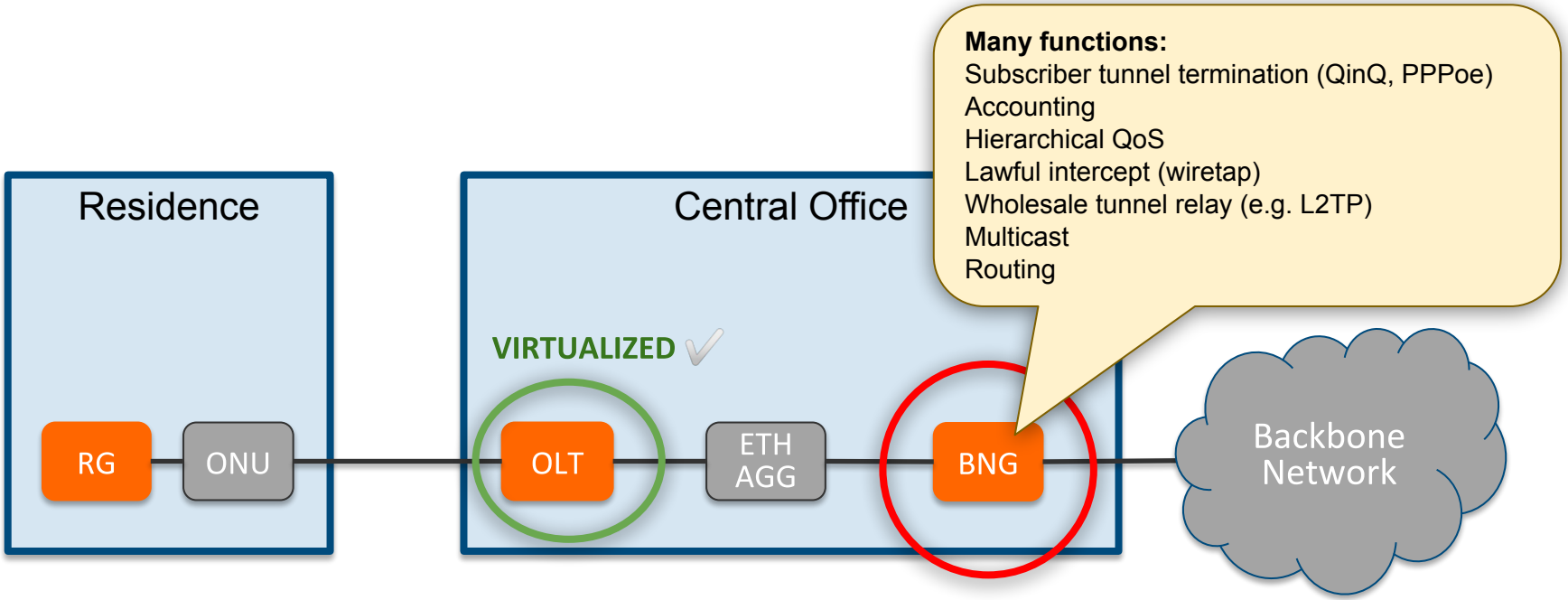
OF-DPA

```
$ cd onos/drivers/.../pipeline/ofdpa/
$ wc -l Ofdpa*.java
    1985 Ofdpa2GroupHandler.java
    1933 Ofdpa2Pipeline.java
     514 Ofdpa3GroupHandler.java
     913 Ofdpa3Pipeline.java
      49 Ofdpa3QmxPipeline.java
     772 OfdpaGroupHandlerUtility.java
    6166 total
```

x3 more LOCs

BNG offloading in SEBA

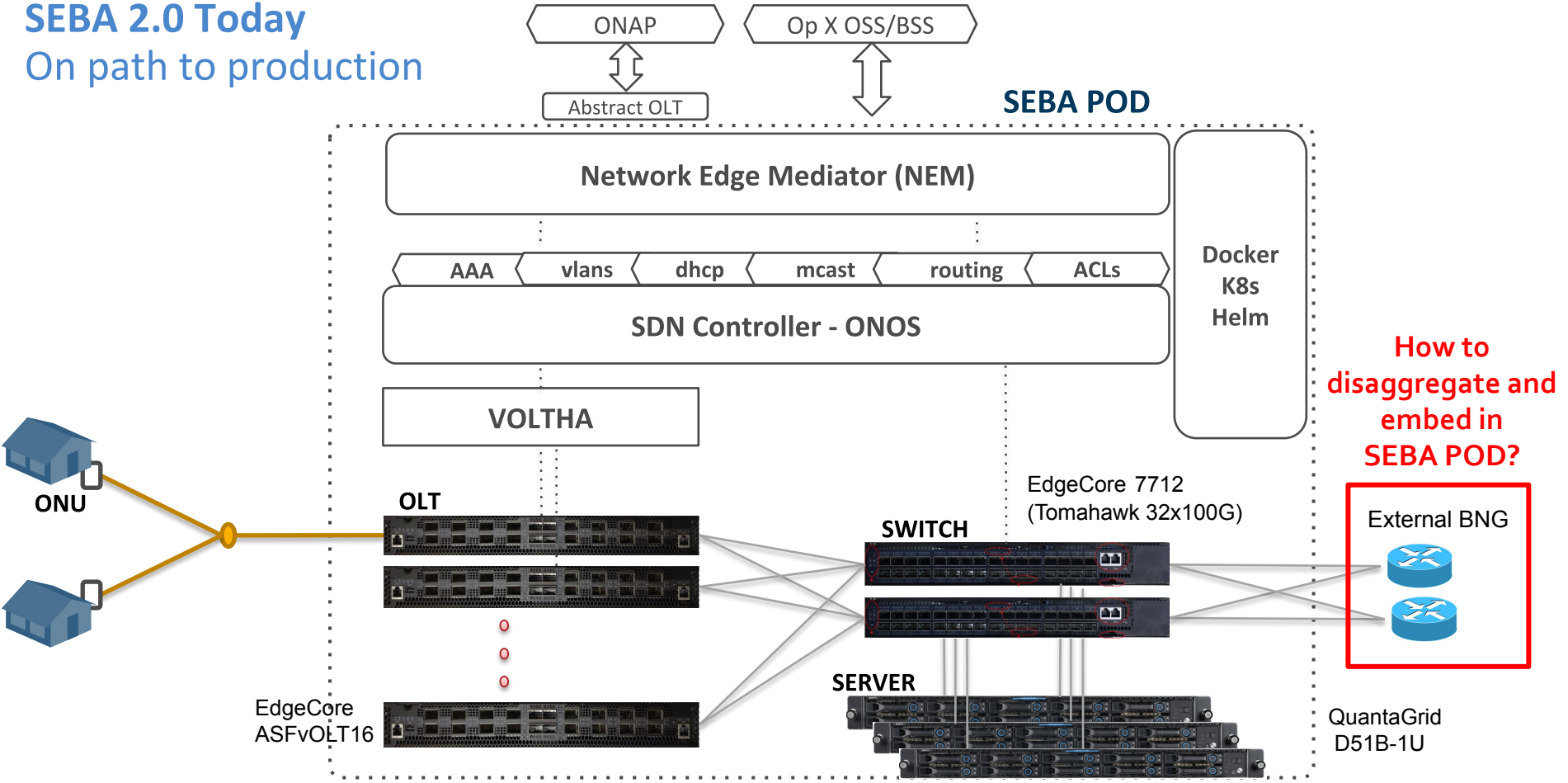
Residential Access Recap



RG – Residential Gateway
OLT – Optical Line Termination
BNG – Broadband Network Gateway

$2B+ market

SEBA 2.0 Today
On path to production

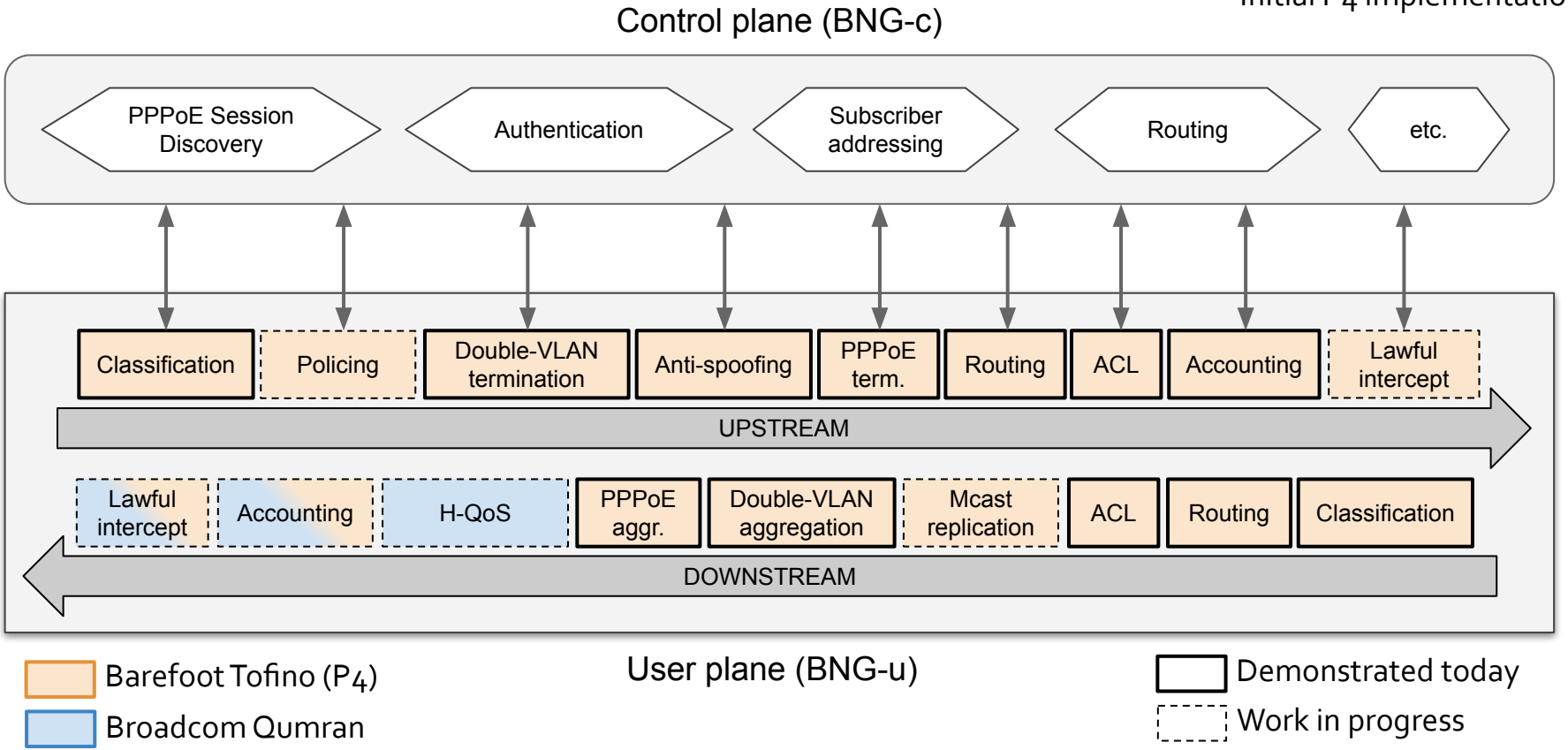


Plan for BNG embedding in SEBA

- **BNG user plane (BNG-u)**
  - Implement "in-fabric" using P4 and merchant silicon
  - Functional distribution over different chipsets
    - Barefoot Tofino, Broadcom Qumran
- **BNG control plane (BNG-c)**
  - App running on top of ONOS
  - Integrate with existing control planes when possible
    - e.g. external PPPoE server, BGP speaker

Disaggregated BNG

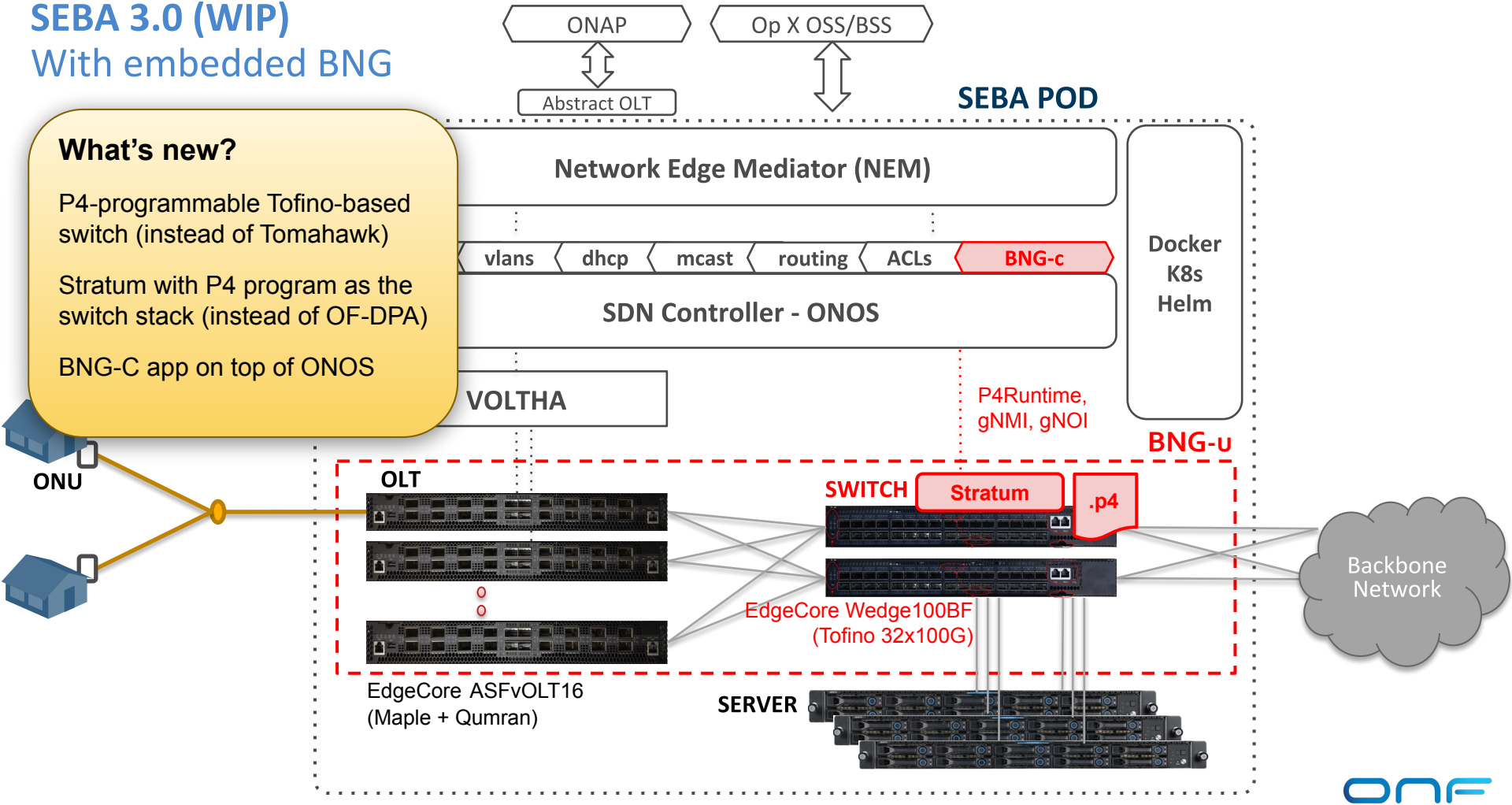
Credits: Deutsche Telekom
Access 4.0 project
Initial P4 implementation



Barefoot Tofino (P4)
Broadcom Qumran

Demonstrated today
Work in progress

SEBA 3.0 (WIP)
With embedded BNG



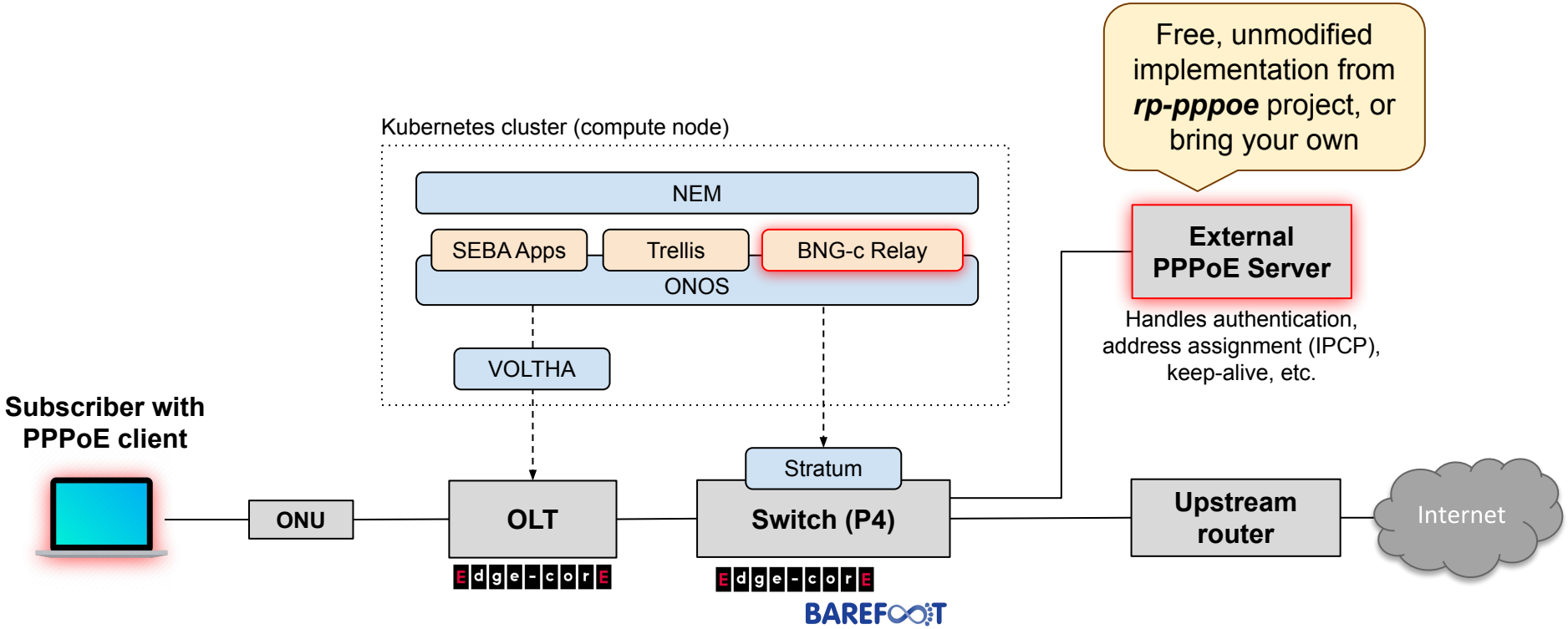
What's new?

P4-programmable Tofino-based switch (instead of Tomahawk)

Stratum with P4 program as the switch stack (instead of OF-DPA)

BNG-C app on top of ONOS

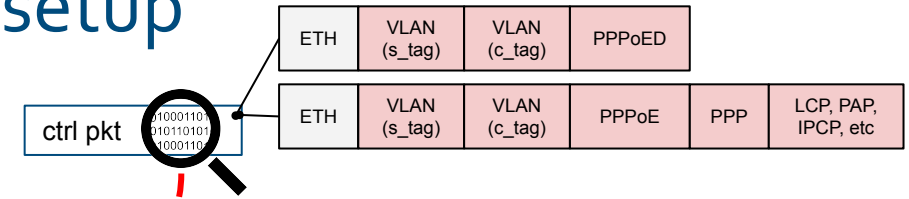
ONF Connect 2019 - Demo setup

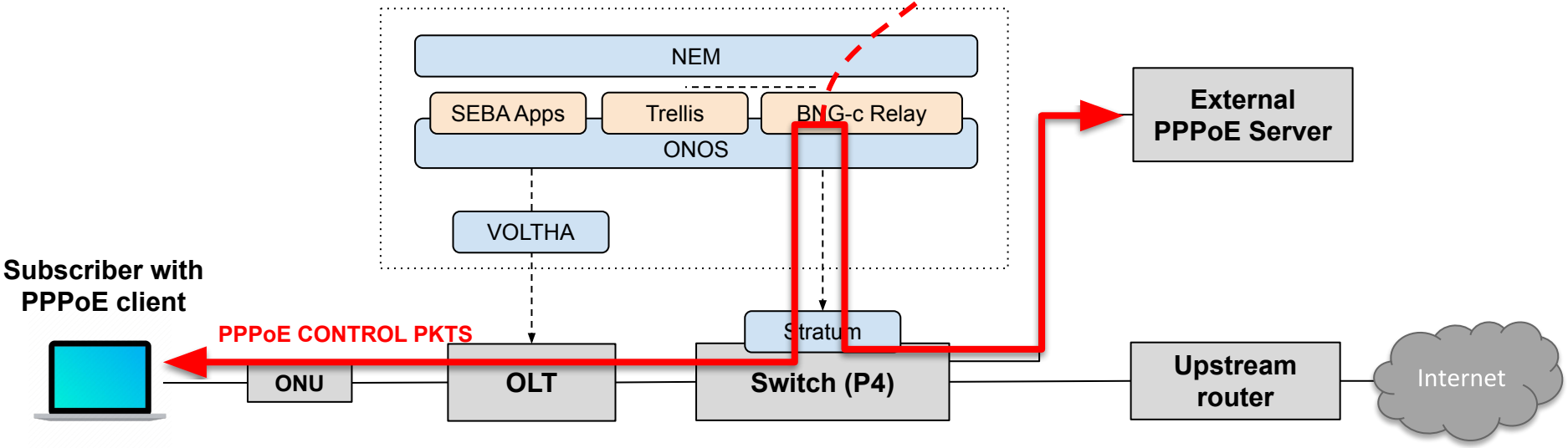


ONF Connect 2019 - Demo setup

1 - PPPoE session establishment

BNG-c Relay learns subscriber info by looking into PPPoE control packets relayed via packet-in/out

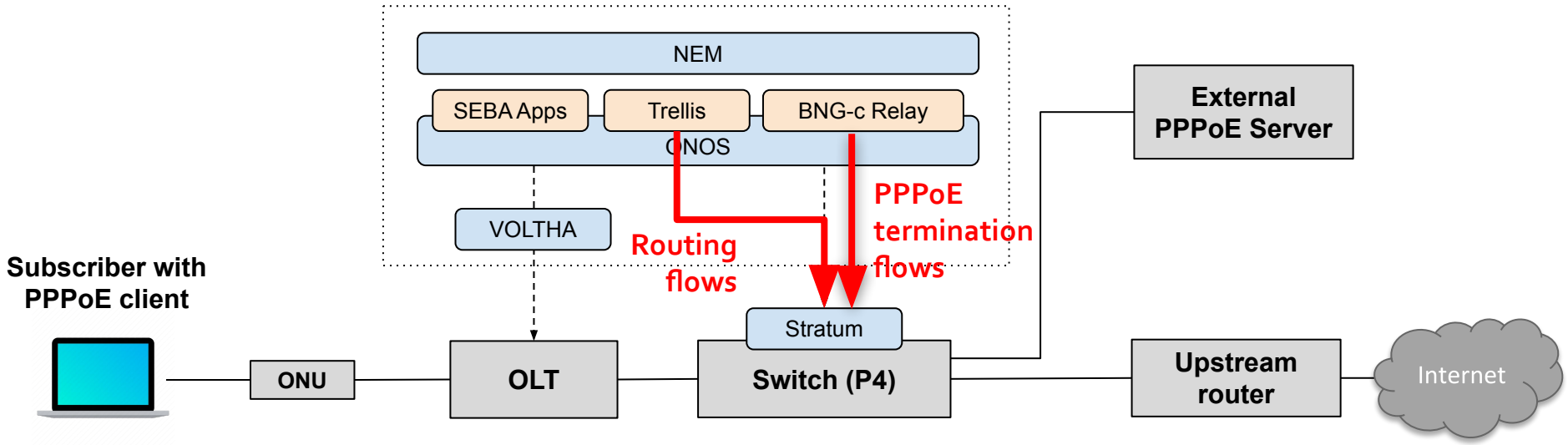




ONF Connect 2019 - Demo setup

2 - User plane termination setup

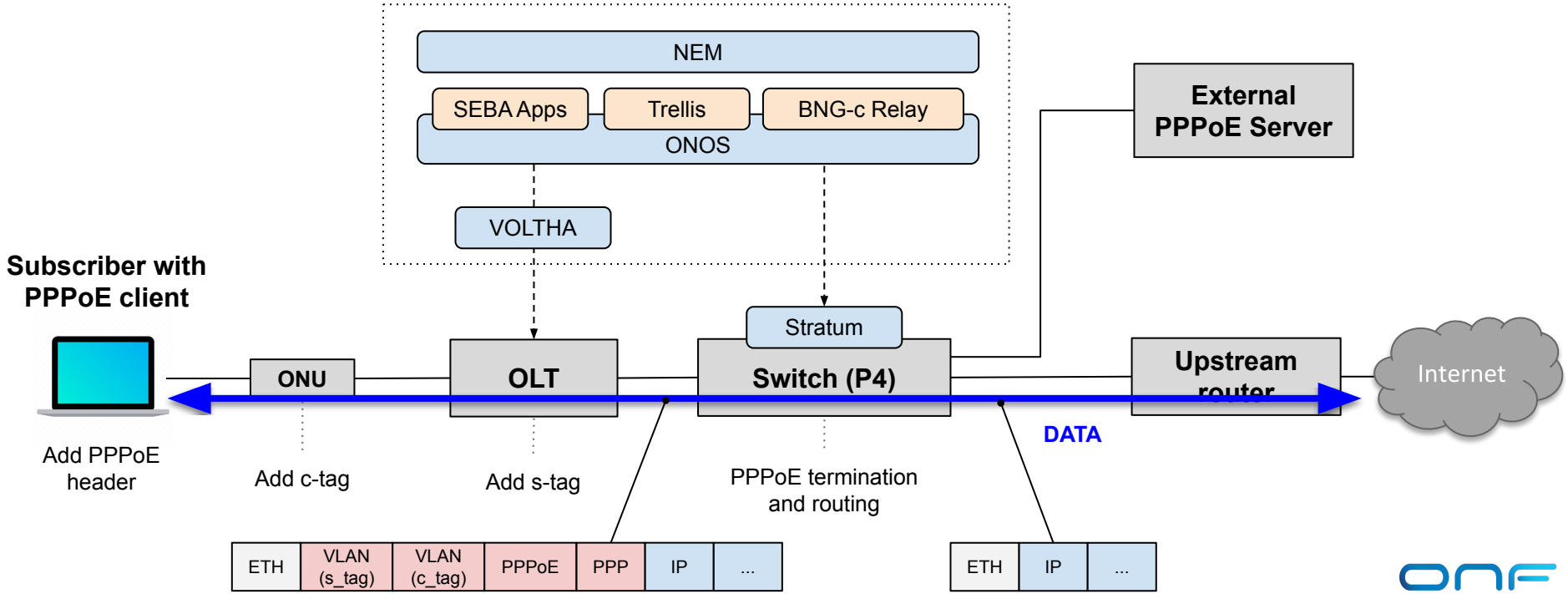
ONOS apps write P4Runtime entries to terminate and route PPPoE data packets to/from the subscriber



ONF Connect 2019 - Demo setup

3 - Subscriber is connected

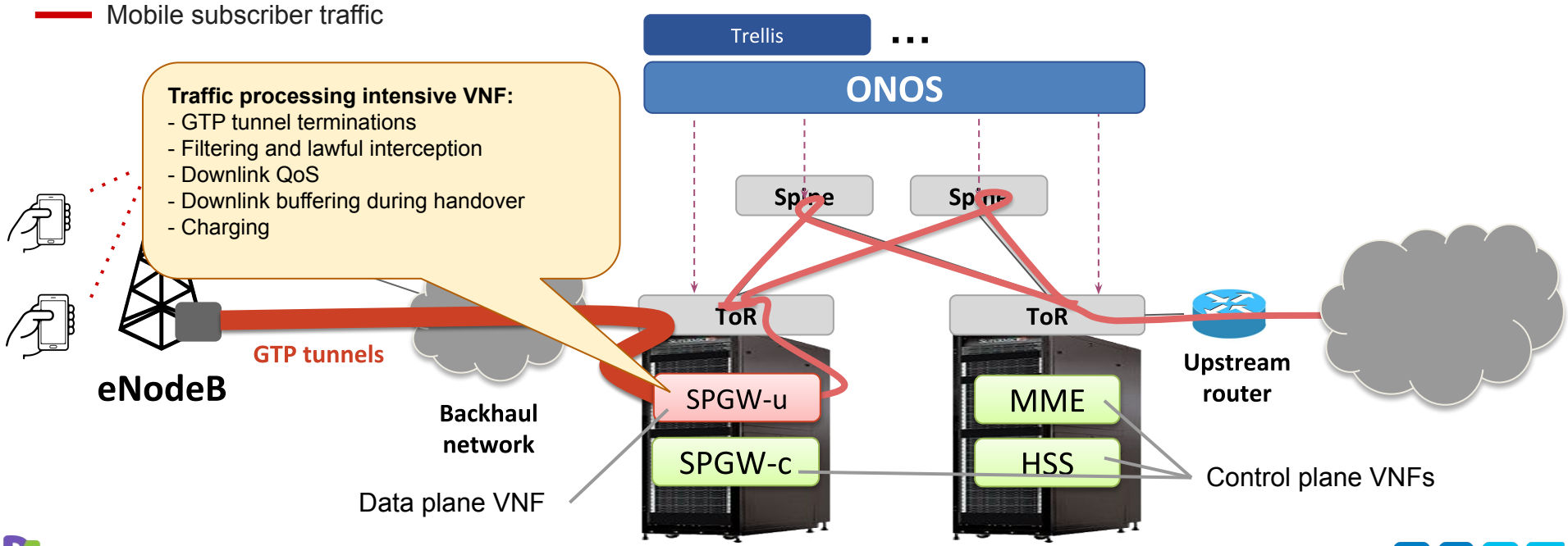
P4 switch performs termination and routing of PPPoE data packets



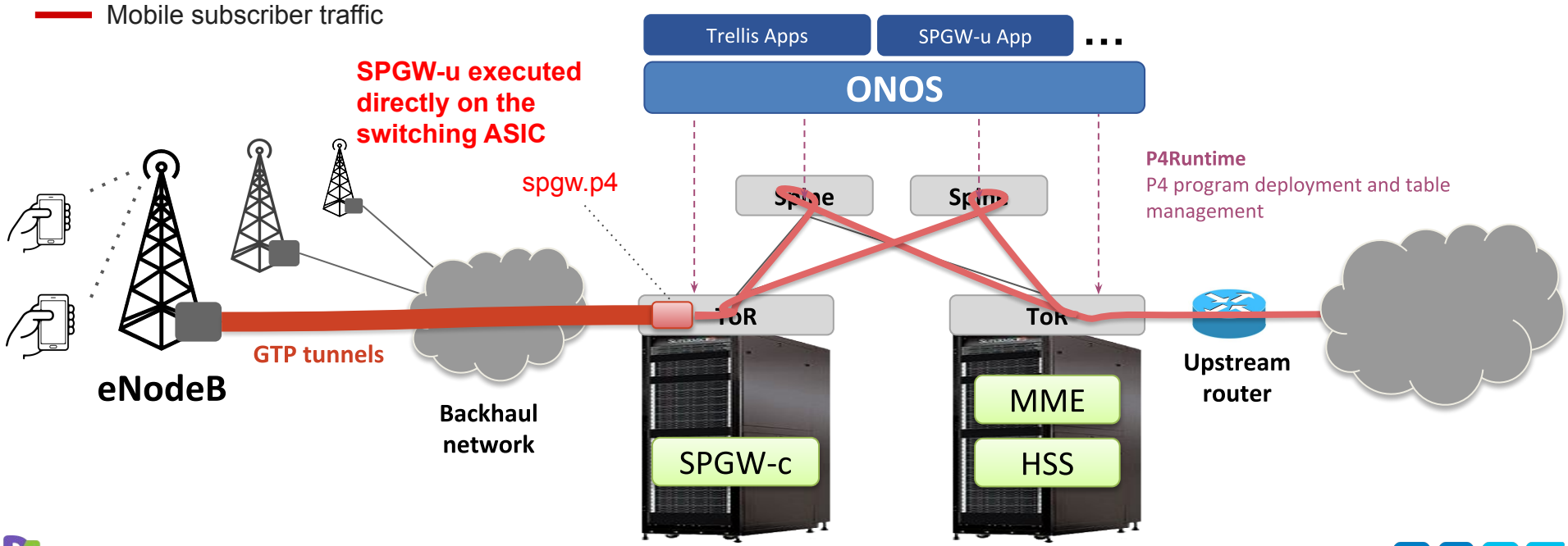
S/PGW offloading in M-CORD

NFV-based mobile access

- **CORD**: ONF NFV platform for the Telco central office
- **M-CORD**: CORD **M**obile profile (other profiles exist, e.g. for **residential** access)



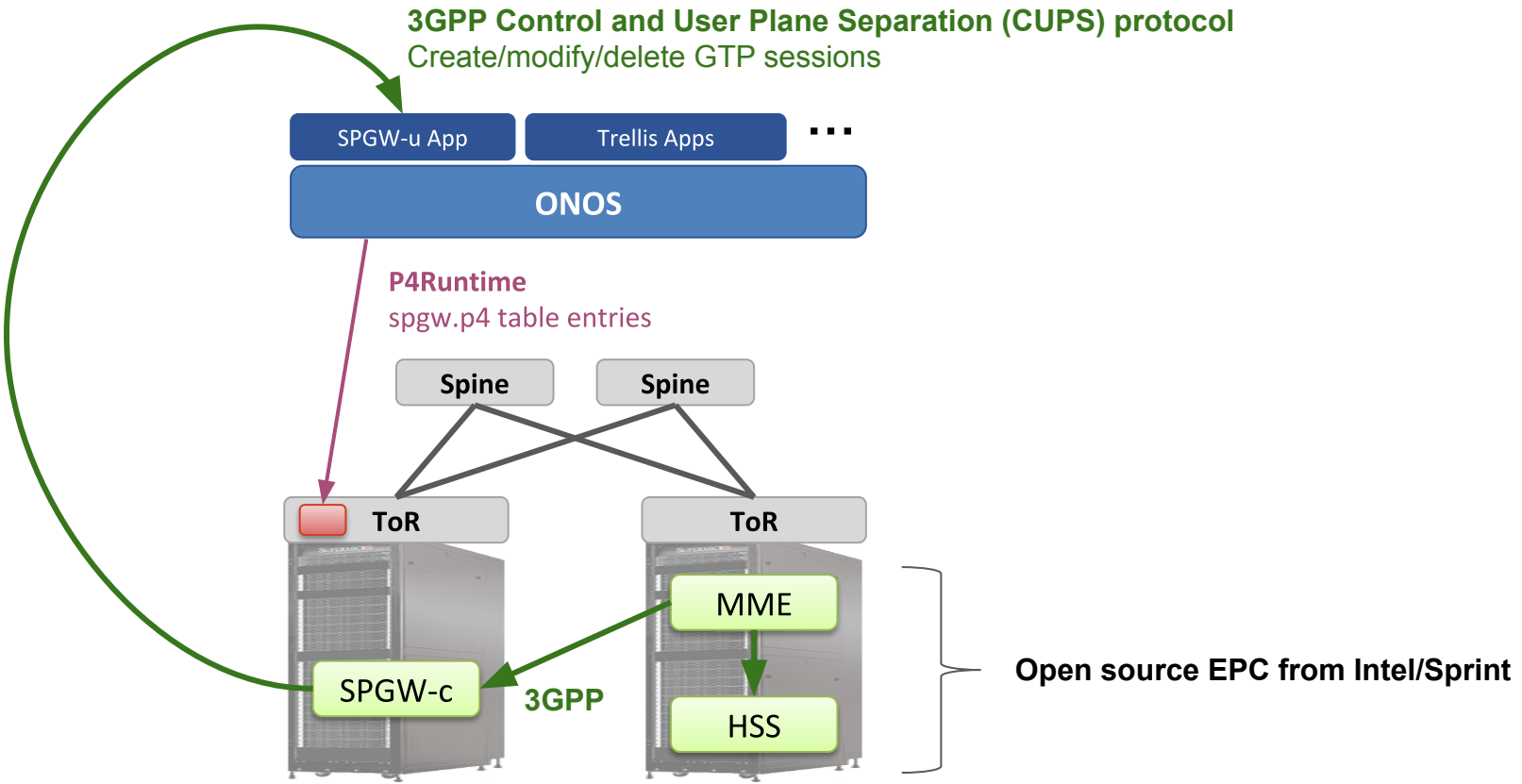
M-CORD with offloaded SPGW-u VNF



spgw.p4

- **PoC P4 implementation of the SPGW-u data plane**
  - ~300 lines of P4_16 code
  - Integrated with fabric.p4
  - https://github.com/opennetworkinglab/onos/.../spgw.p4
- **Good enough to demonstrate end-to-end connectivity**
  - Support GTP encap/decap, filtering, charging functionalities
- **Important missing features**
  - *Downlink buffering during handovers*
  - *Downlink QoS*

SPGW-u App



Exercise 4:
Modify code to enable IPv6 routing

Exercise 3 steps

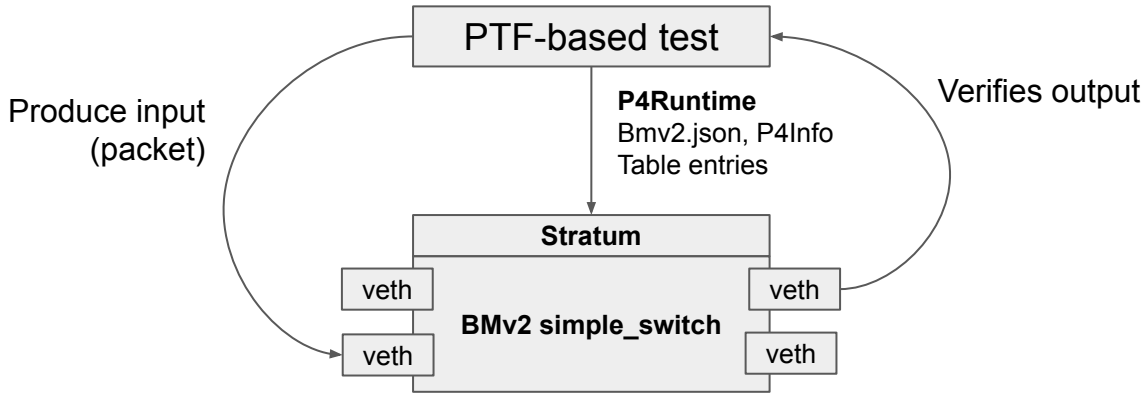
We want the topology to behave like a traditional IPv6 fabric, but… Implementation is broken, your task is to fix it!

Steps:

- **Debug the issue (with step-by-step instructions)**
- **Update P4 program**
- **Run PTF unit tests to validate P4 changes**
- **Update ONOS app implementation**
- **Test connectivity between hosts on different IPv6 subnets**

PTF overview

- **Python-based dataplane test framework**
- **Similar to OFTest framework**
  - But focuses on the dataplane and is independent of OpenFlow/P4Runtime
- **High-level P4Runtime lib provided with starter code**
  - Add/remove table entries, groups, packet-in/out, etc.



IPv6 fabric recap

- **Leaf switches should behave as a traditional router (simplified)**
  - I.e., with IPv6 configuration on interfaces (address and subnet)
- **Hosts configured with “gateway” IPv6 address the leaf switch one**
- **Hosts should be able to resolve the MAC address of their gateway**
  - i.e. the leaf switch should reply to NDP Neighbor Solicitation messages sent by the hosts
- **Not all packets need to be “routed” by leaf switches**
  - Only those with destination MAC address the “gateway” one (myStationMac)
- **Switch maps IPv6 prefixes (LPM) to next hops (routing table)**
- **Support ECMP when forwarding to spine switches**

netcfg.json (fabricDeviceConfig)

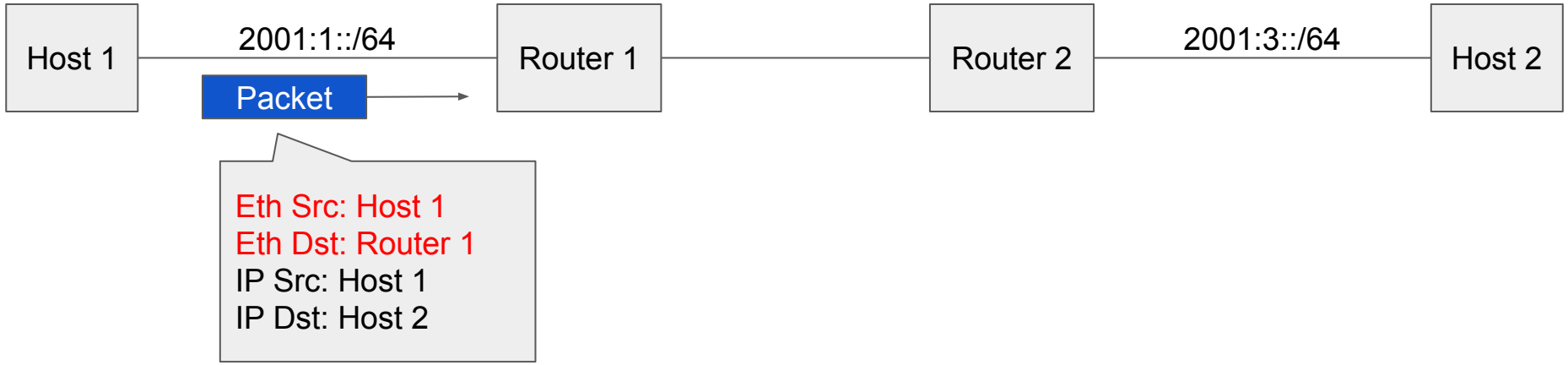
```
{
  "devices": {
    "device:leaf1": {
      "basic": {
        "managementAddress": "grpc://mininet:50001?device_id=1",
        "driver": "stratum-bmv2",
        "pipeconf": "org.onosproject.ngsdn-tutorial"
      },
      "fabricDeviceConfig": {
        "myStationMac": "00:aa:00:00:00:01",
        "isSpine": false
      }
    },
    ...
```

netcfg.json (fabricDeviceConfig)

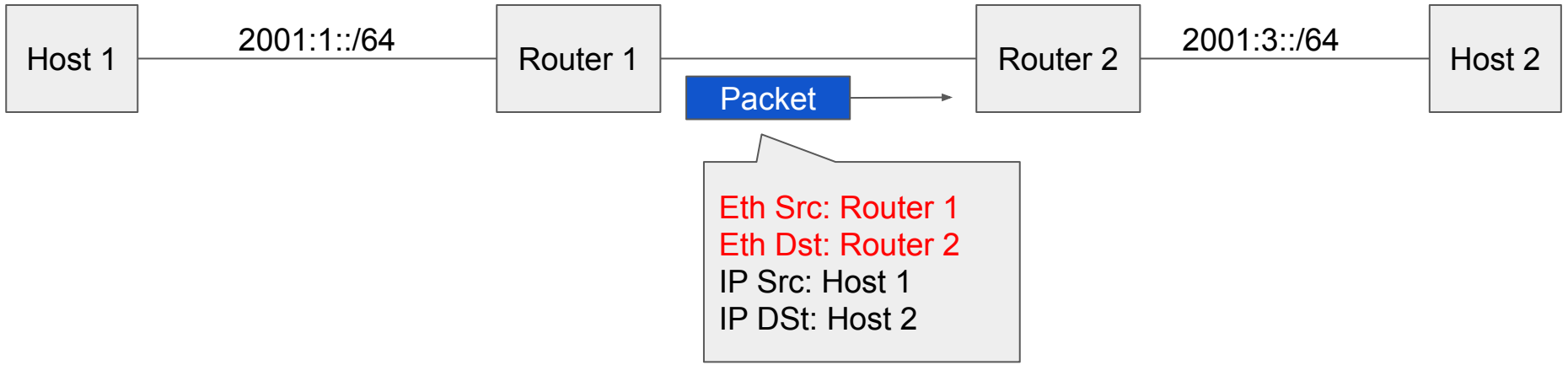
```
...
"ports": {
    "device:leaf1/3": {
      "interfaces": [
        {
          "name": "leaf1-3",
          "ips": ["2001:1:1::ff/64"]
        }
      ]
    },

    ...
```

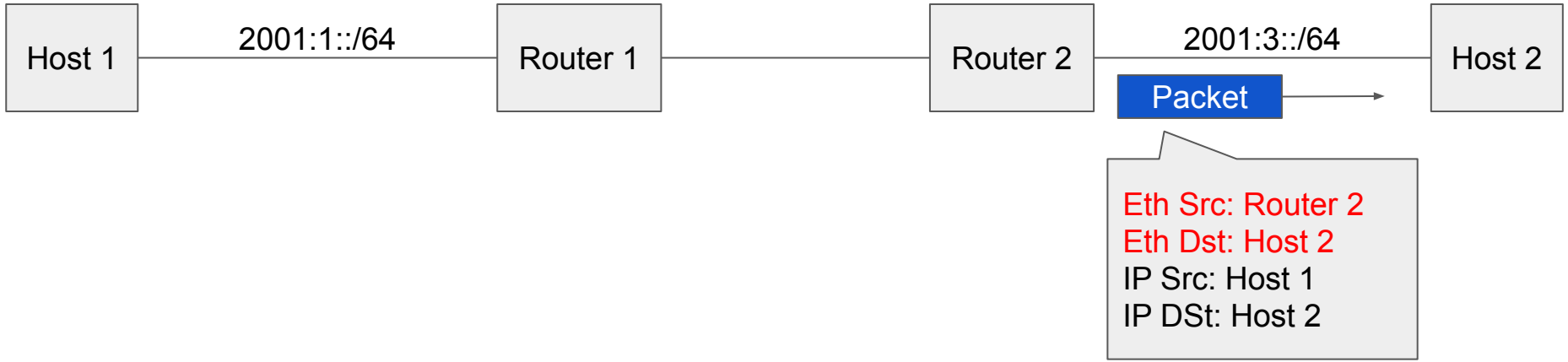
IPv6 unicast routing



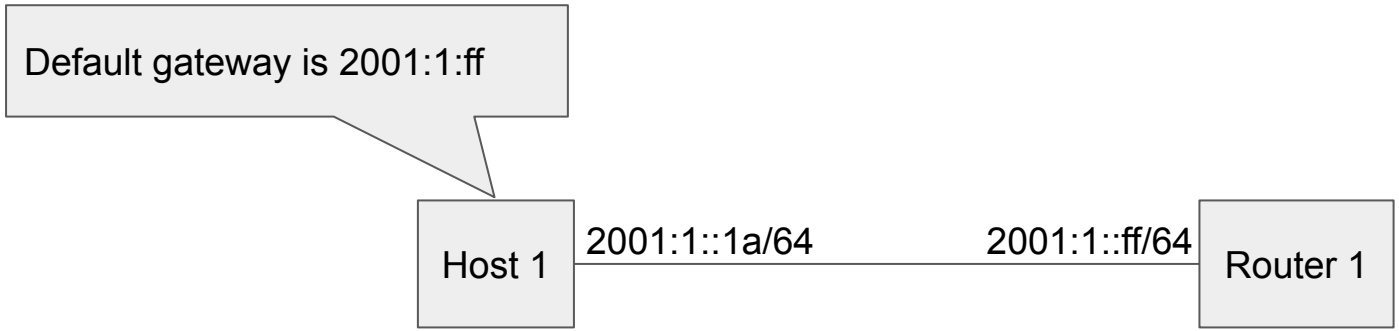
IP unicast routing



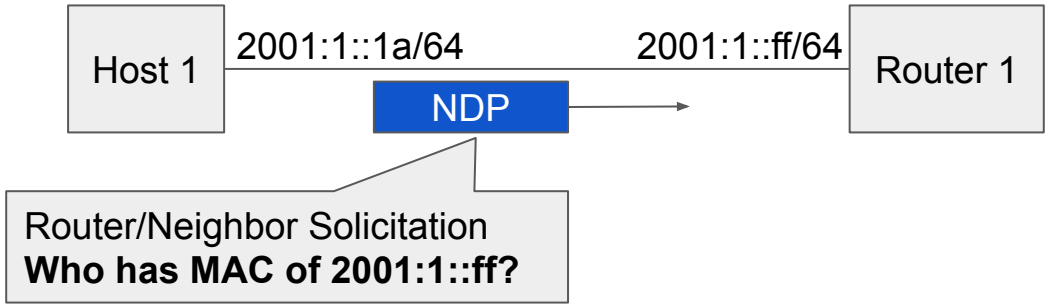
IP unicast routing



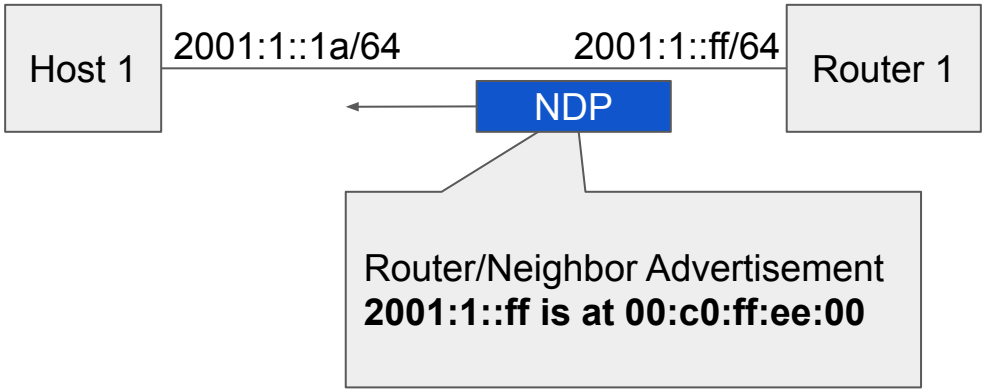
Neighbor Discovery Protocol (NDP)



Neighbor Discovery Protocol

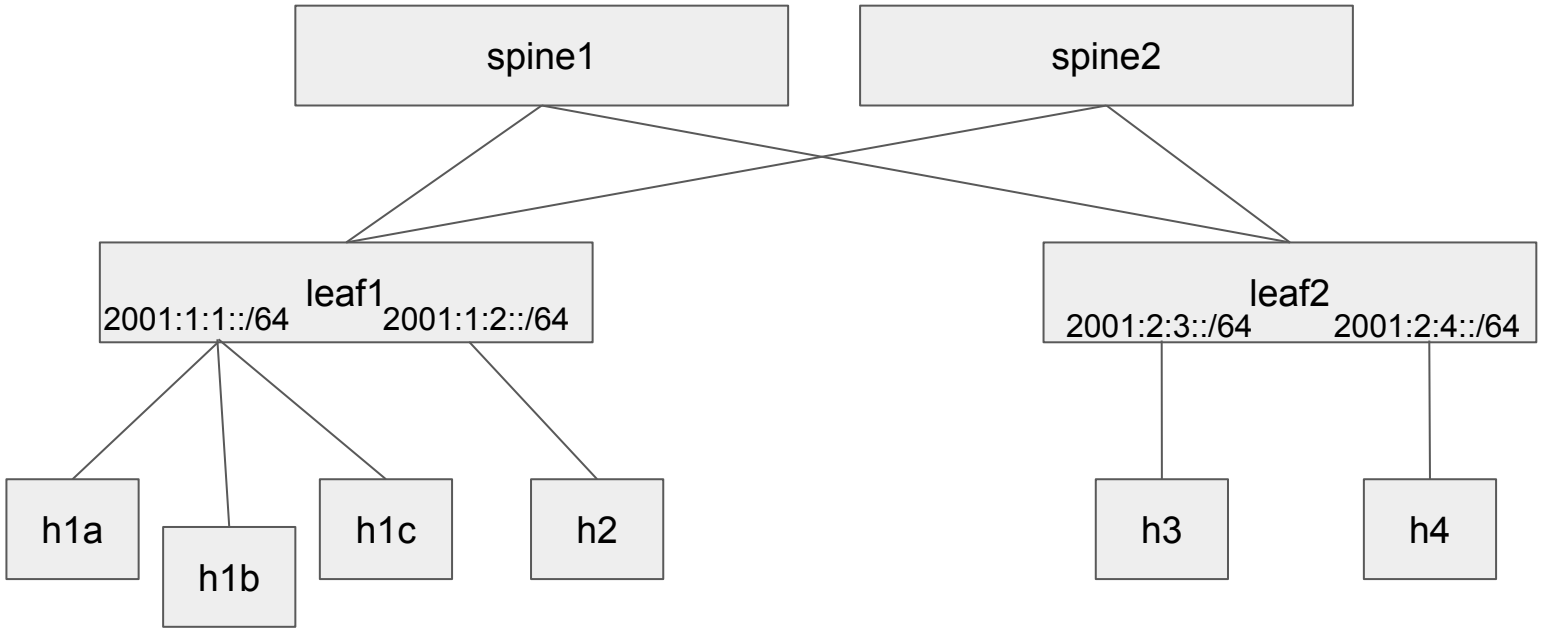


Neighbor Discovery Protocol

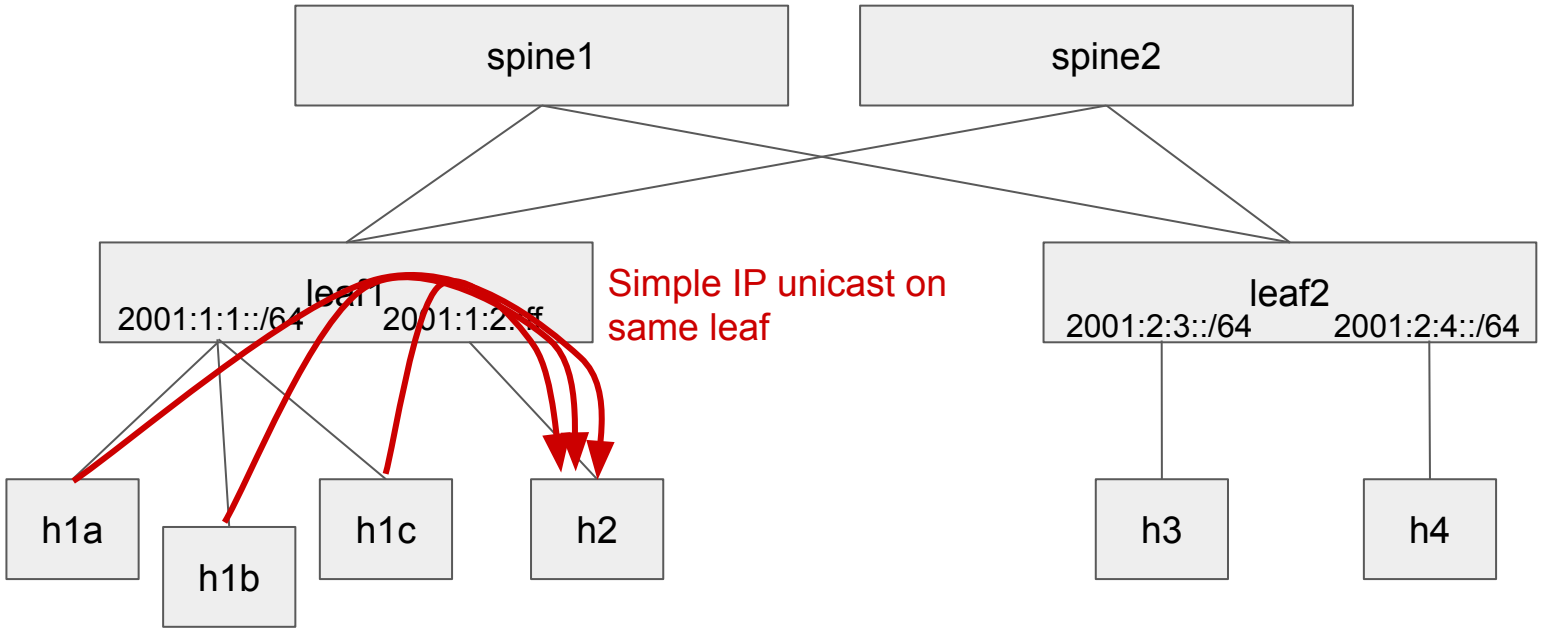


How to generate NDP replies from the switch?

Same-leaf routing

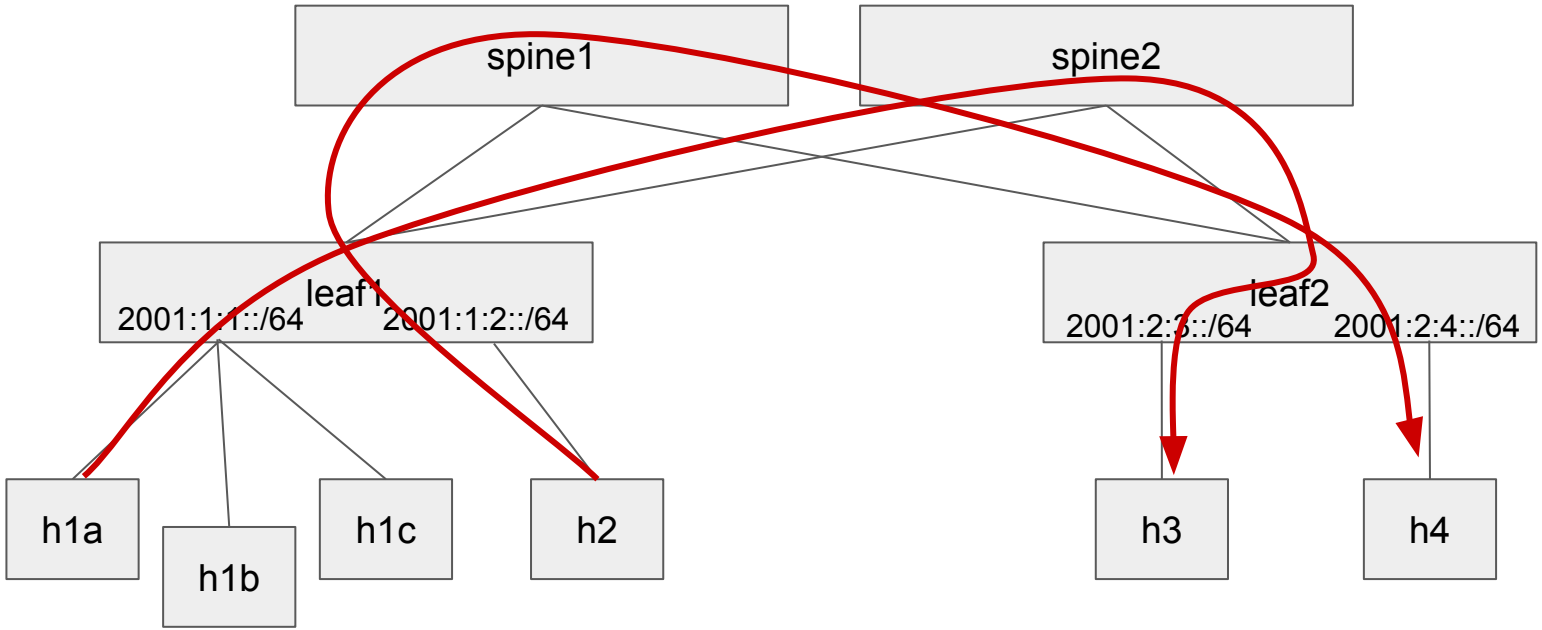


Same-leaf routing



ECMP

Route to other leaves via ECMP



`Ipv6RoutingComponent.java`

- **Listens to device and topology events**
- **For each device, provision flow rules to:**
  - Match on `myStationMac` to enable routing table
  - Route packets to attached hosts (/128)
  - Route packets to spines when matching on other leaves IPv6 subnets (from interface config)
  - Groups used for both attached hosts (one next hop) and ECMP (multiple next hops, one per link/spine)
- **Looks at topology information (leaf-spine links) and netcfg to compute path, generate flow rules, and groups**

Exercise 4: Get Started

These slides:
http://bit.ly/ngsdn-tutorial-slides

Open lab README on GitHub:

http://bit.ly/ngsdn-tutorial-lab

Or open in text editor:

~/ngsdn-tutorial/README.md

~/ngsdn-tutorial/EXERCISE-4.md

Solution:

~/ngsdn-tutorial/solution

Before starting!
Update tutorial repo
(requires Internet access)

```
cd ~/ngsdn-tutorial
git pull origin master
make pull-deps
```

P4 language cheat sheet:
http://bit.ly/p4-cs

You can work on your own using the instructions.
Ask for instructors help when needed.

Wrap Up

Recap

- **Domain specific languages**
  - P4 (pipeline modeling), YANG (configuration modeling)
- **Models**
  - Tutorial P4 program (IPv6 router), OpenConfig
- **APIs**
  - P4Runtime, gNMI, gNOI
- **Switch OS**
  - Stratum - implementation of P4Runtime, gNMI, and gNOI
- **Controller platforms**
  - ONOS - with support for Stratum

Learn more @ ONF Connect 2019

- **Talks at Next-Gen SDN Track:**
  - Operator's update on P4 use cases for the Edge Cloud
  - P4 compiler for fixed-function switches
  - Validation of fixed-function switches against a P4 Program
  - Refactoring OpenFlow solutions to P4Runtime
  - µONOS project overview
  - and more…
- **Demos**
  - BNG disaggregation with Stratum and SEBA
  - Stratum interoperability: Broadcom Tomahawk and Barefoot Tofino
  - µONOS demo with Stratum

Thanks!