



Switch programmability

7/2018

Switch Programmability

Old world



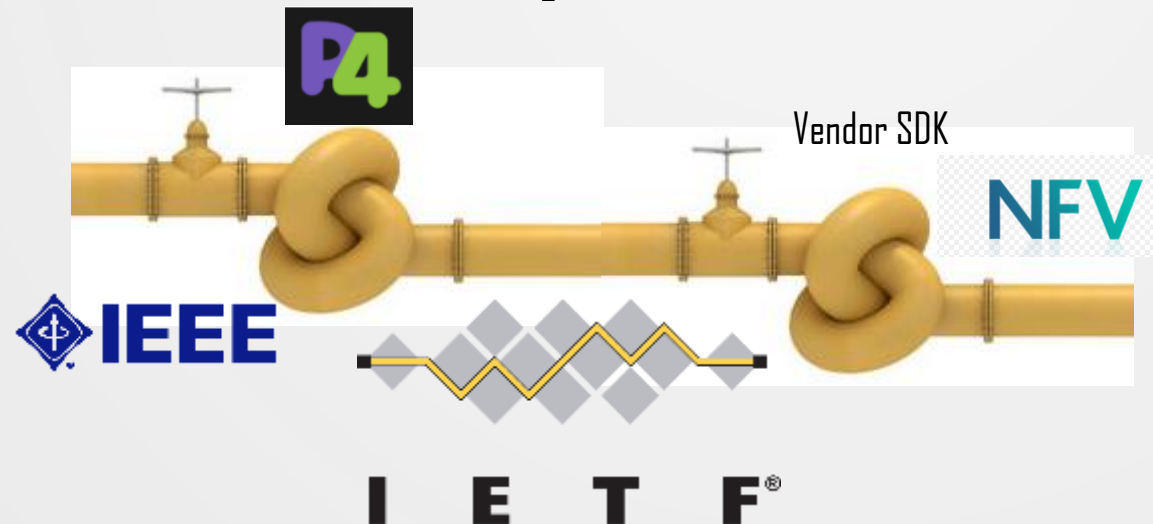
\$\$\$ on legacy protocols
 Best performance and stability
 Low feature velocity

Fully Programmable



Write everything from scratch
 Implement both standard and new applications
 Variant feature velocity

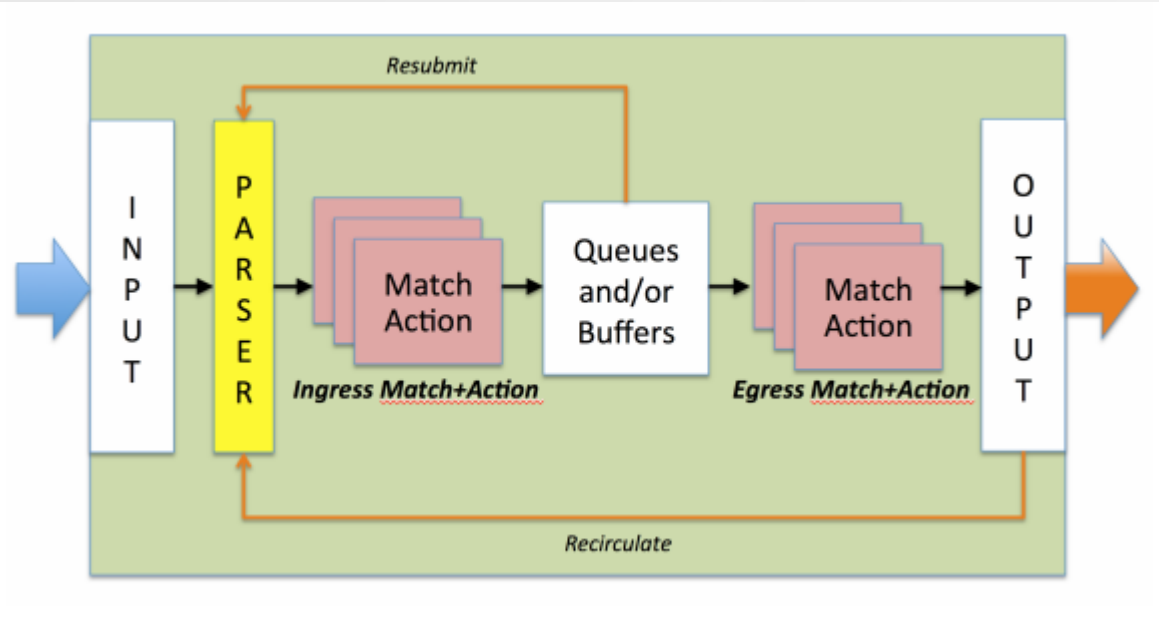
Hybrid



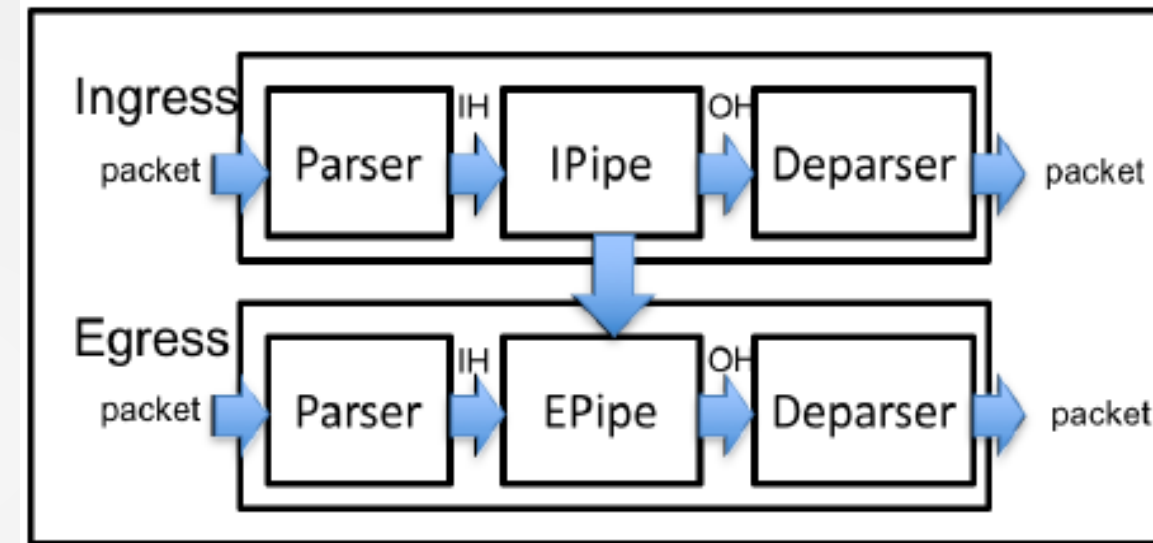
Legacy protocols don't change
 Application sand box for home grown needs
 Extended HW longevity
 High feature velocity

Introduction to P4

V14



V16

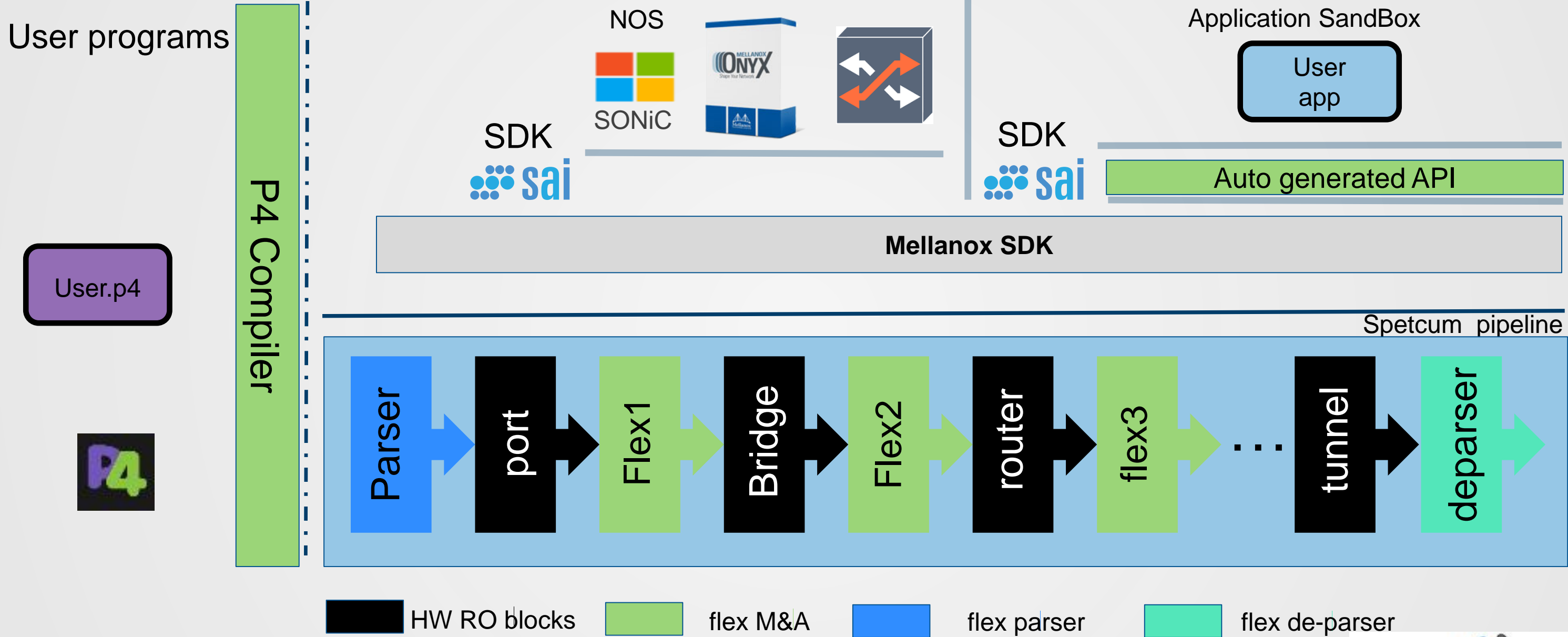


From the spec:

- Introducing P4 architecture description language
- “The P4 architecture can be thought of as a contract between the program and the target”



Programmability – Hybrid

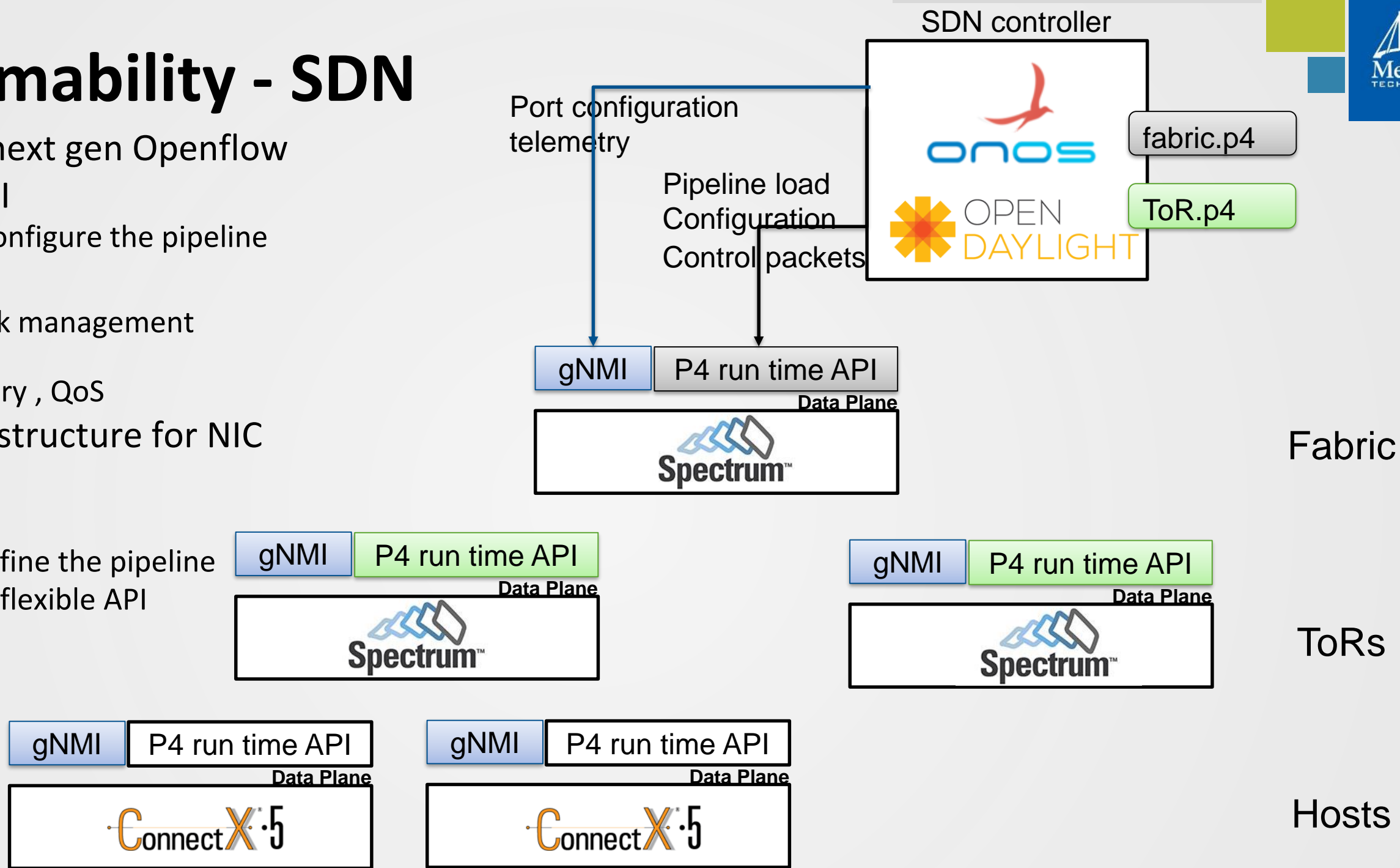


Multiple switching SW options, develop apps not NOS



Programmability - SDN

- P4runtime = next gen Openflow
- P4runtime API
 - Define and configure the pipeline
- gNMI
- gRPC network management interface)
 - Port ,telemetry , QoS
- Uniform infrastructure for NIC and Switch
- True SDN
 - Controller define the pipeline
 - Uniform and flexible API
 - Flexible





Demos

7/2018





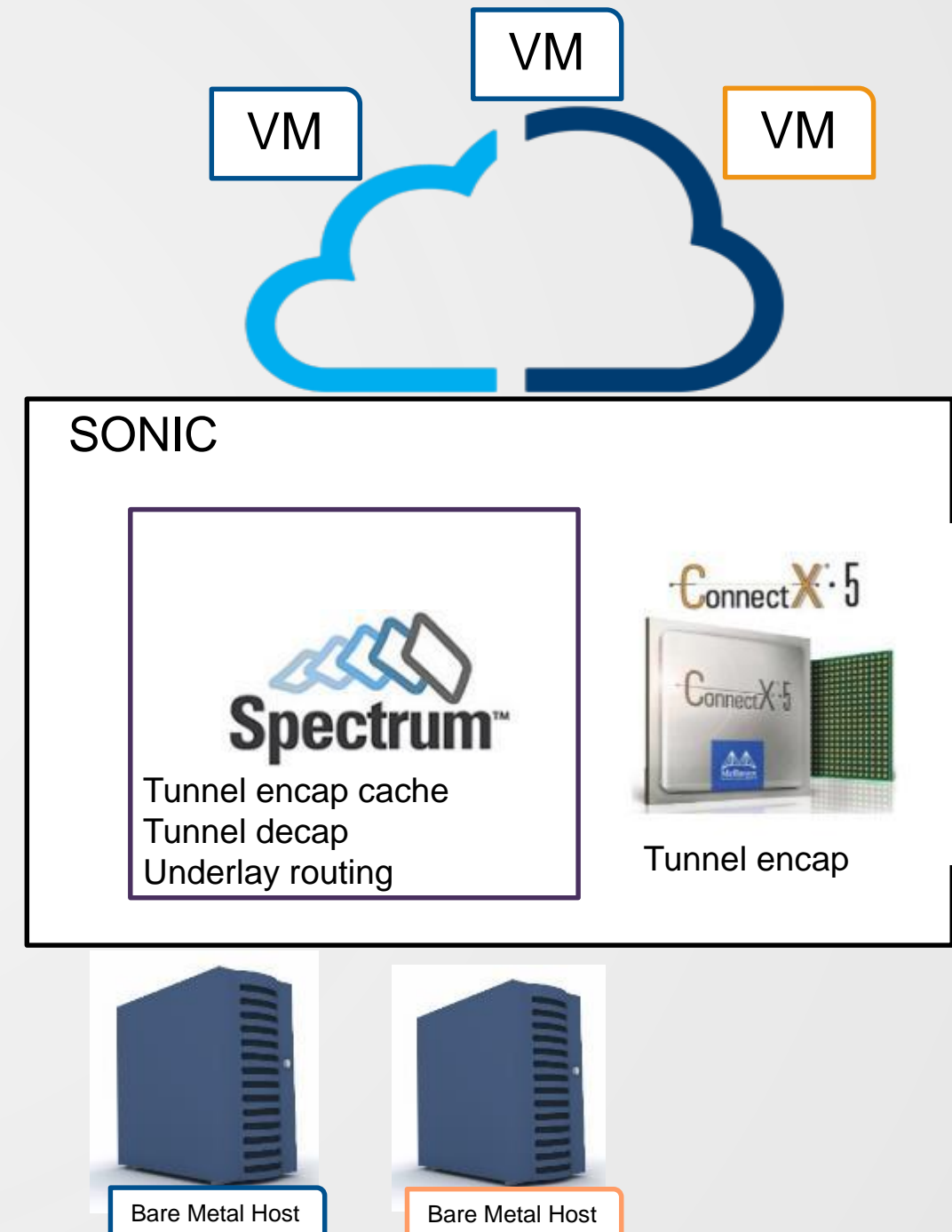
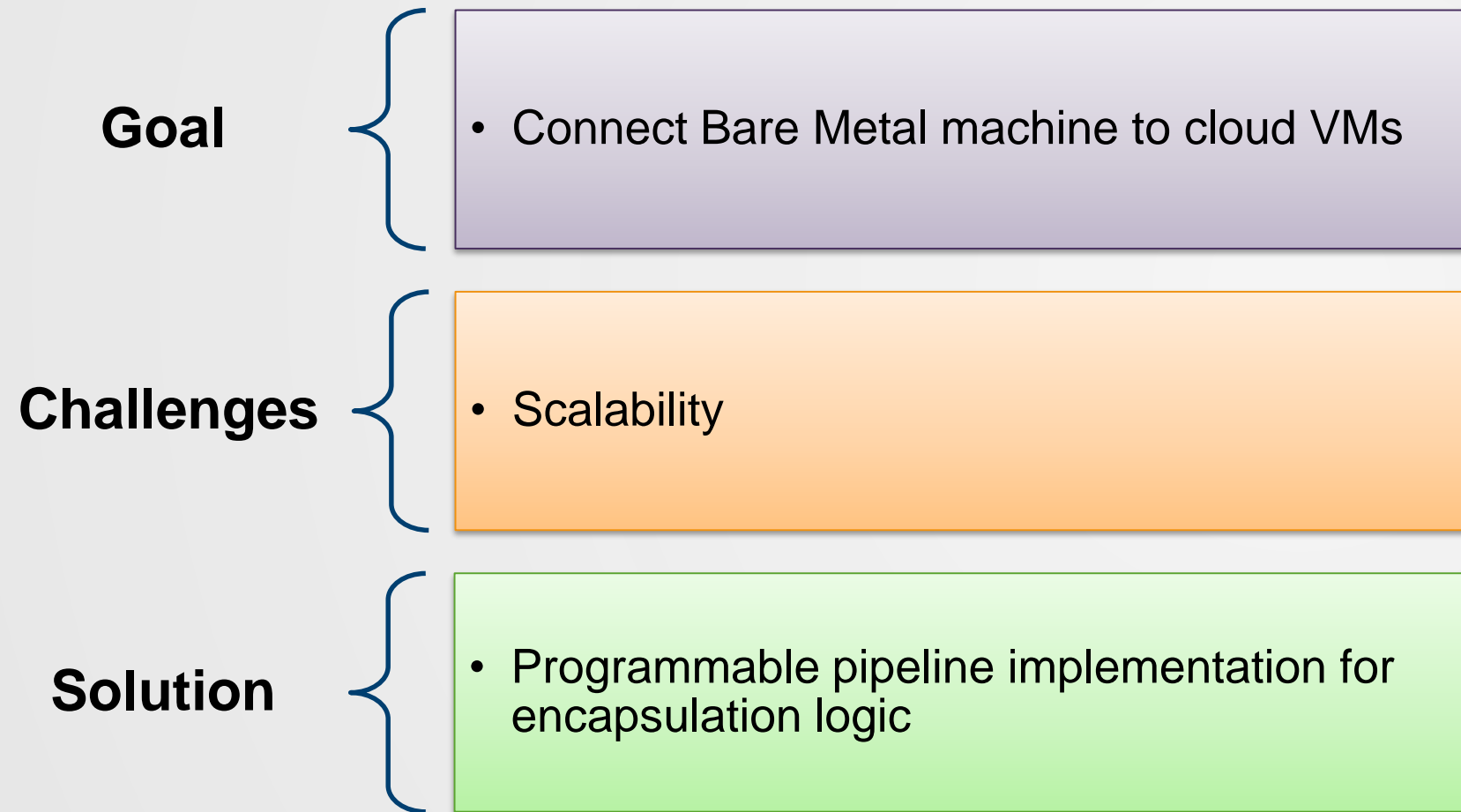
BMToR

Adding Bare Metal services to the Cloud

7/2018



Adding Bare Metal services to the Cloud

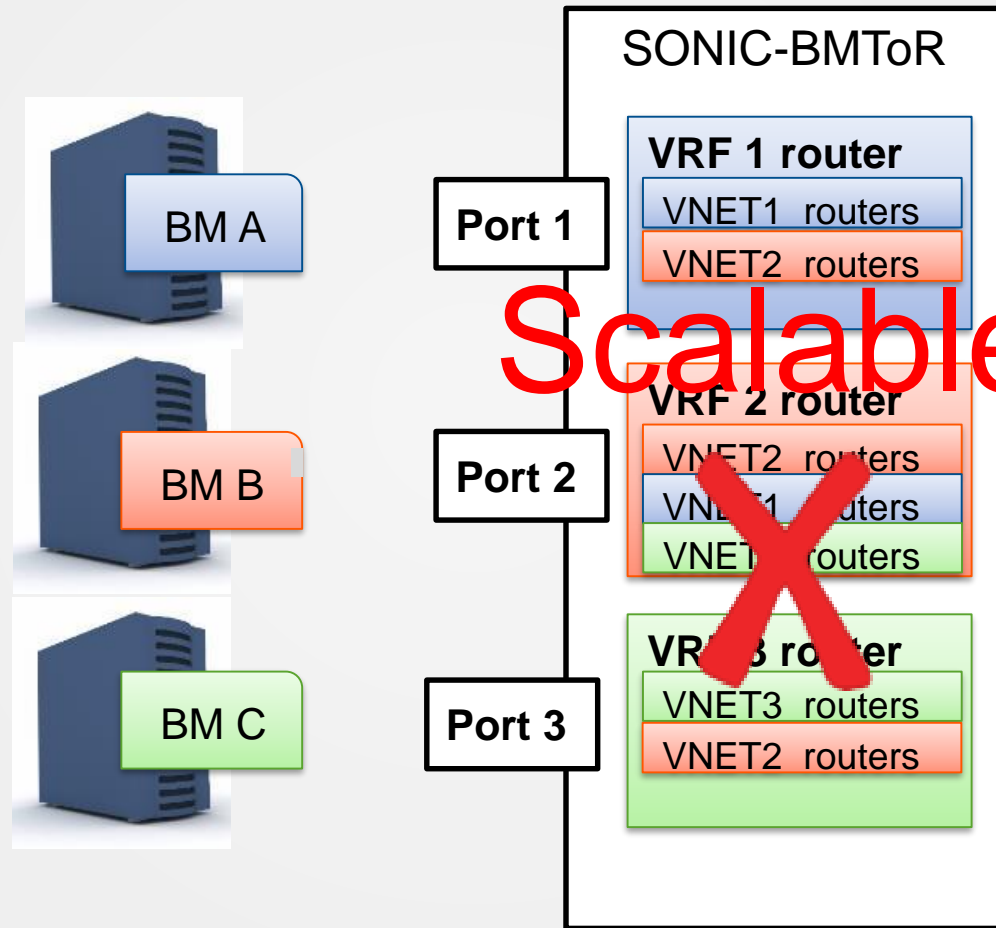


VNET peering in Legacy network

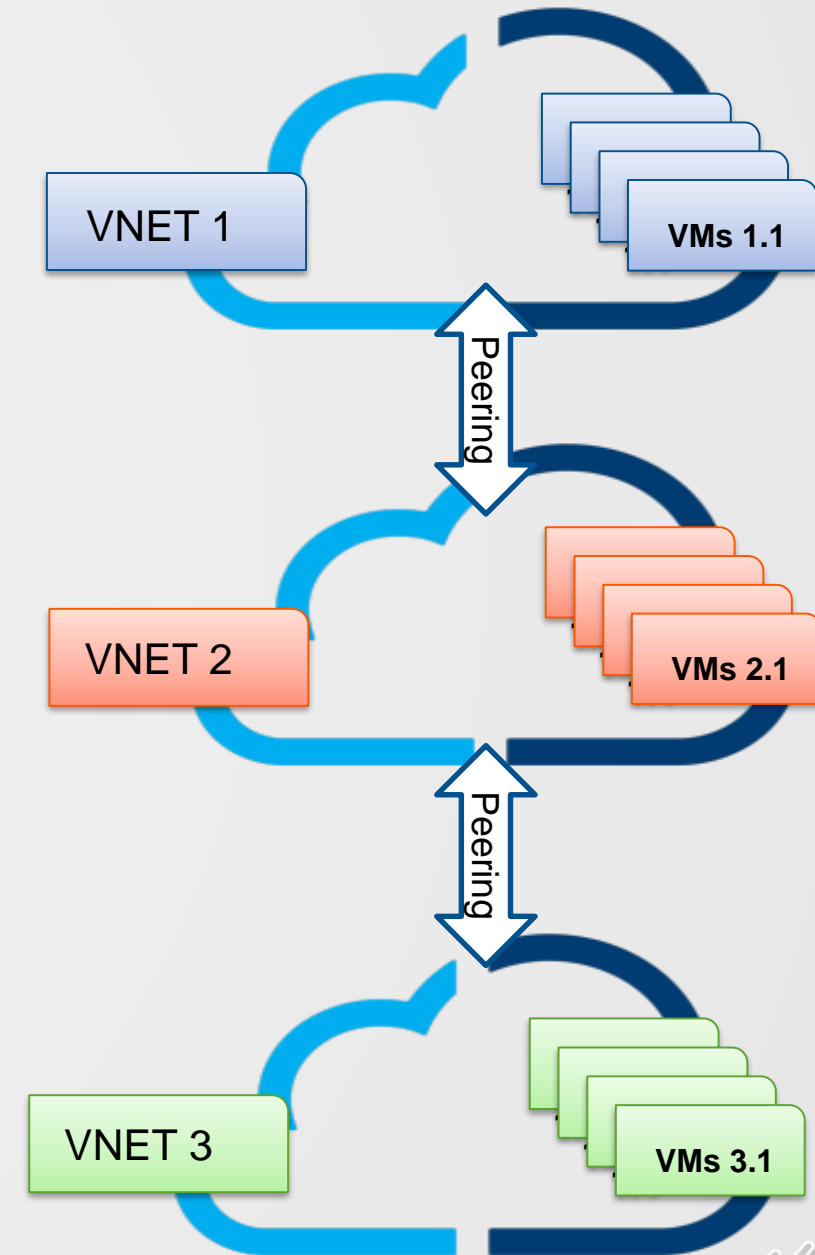
- VNET-virtual network
- VNET peering -Peering between virtual networks

Implementation:

- VNET -> VRF
- VNET1 peering with VNET2 -> copy route from VNET1 to VNET2 and vice versa

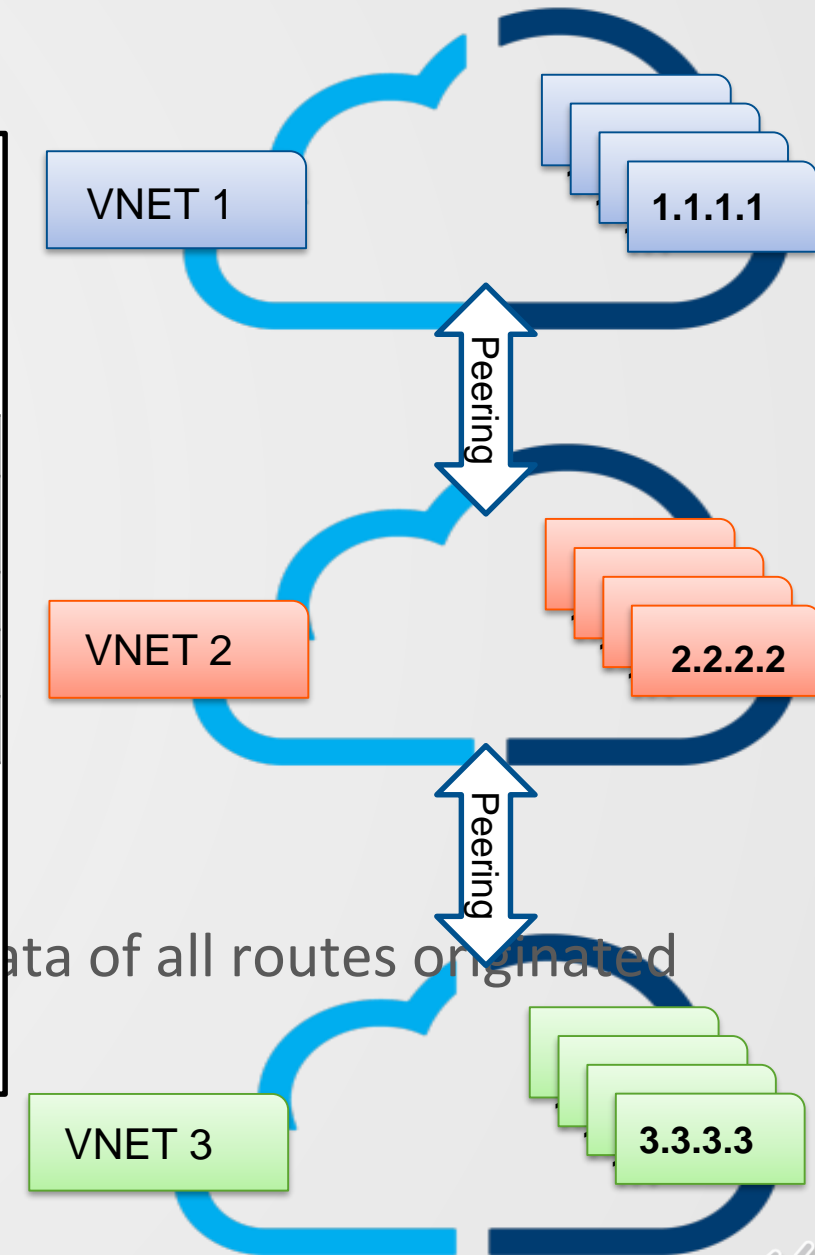
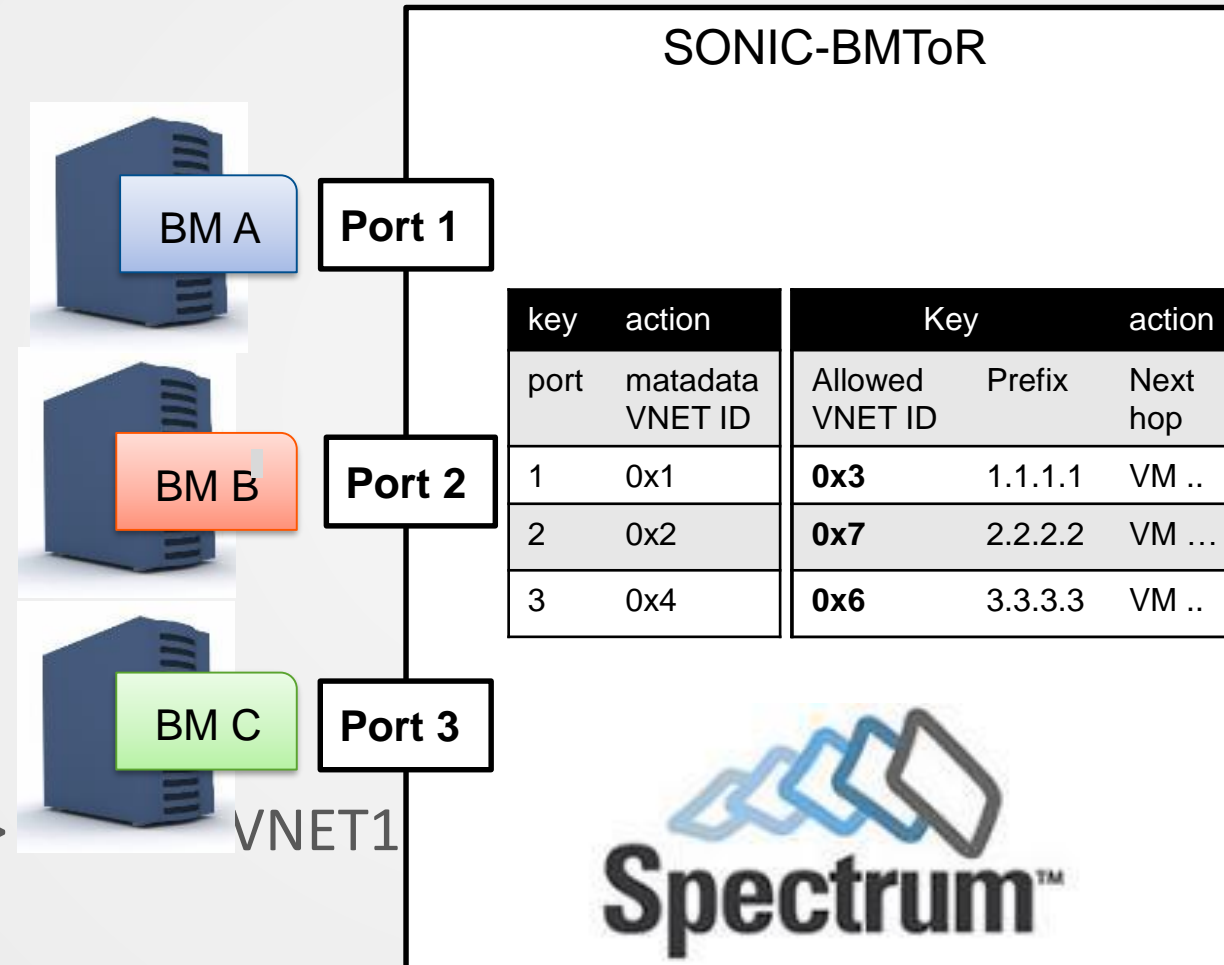


1K VMs and 100 VNETs will require up to 10M routes !!!



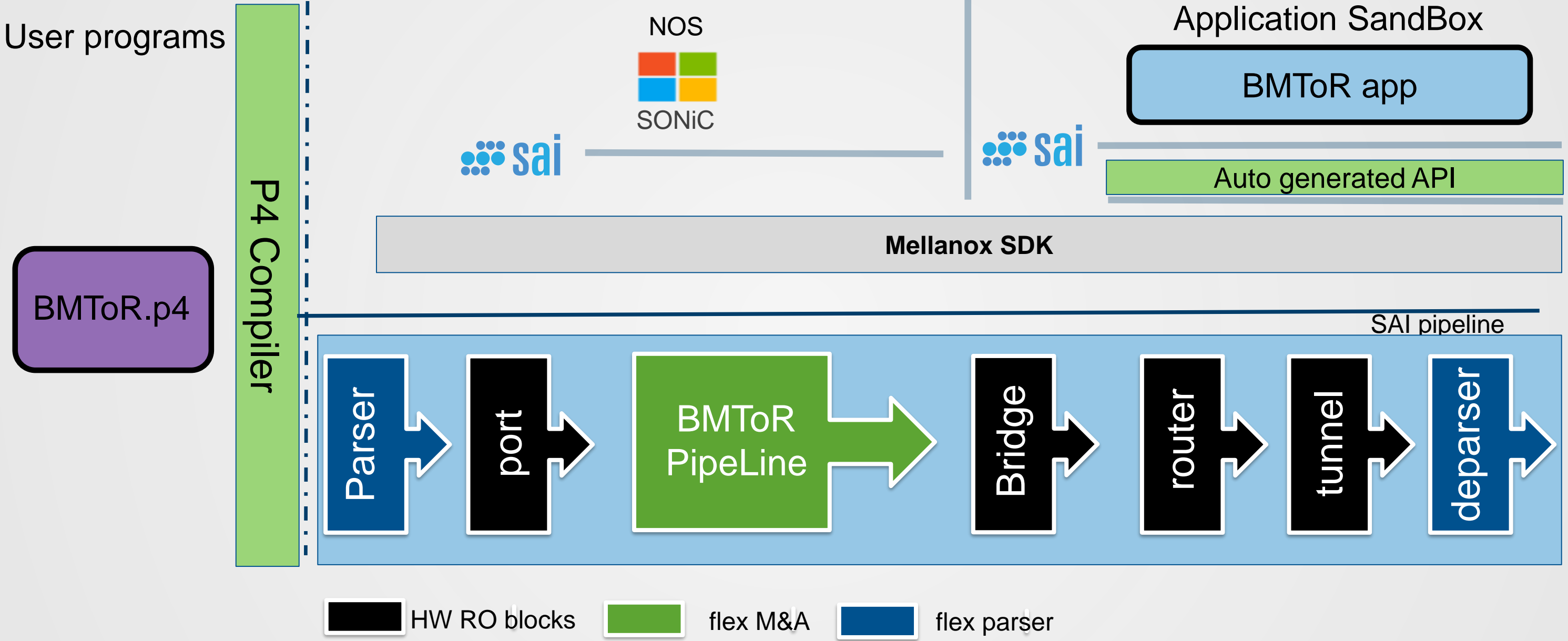
VNET peering in programmable network

- Two match action tables
- Port to VNET
 - Key: Port
 - Action Set metadata
 - metadata = VNET ID
- VNET routing
 - Key: metadata , prefix
 - metadata vector of VNET peers
 - Action: next hop
- VNET1 peering with VNET2 -> by VNET2
 - A single route per VM
 - Single update per VM route



data of all routes originated

Programmability – adding BMToR





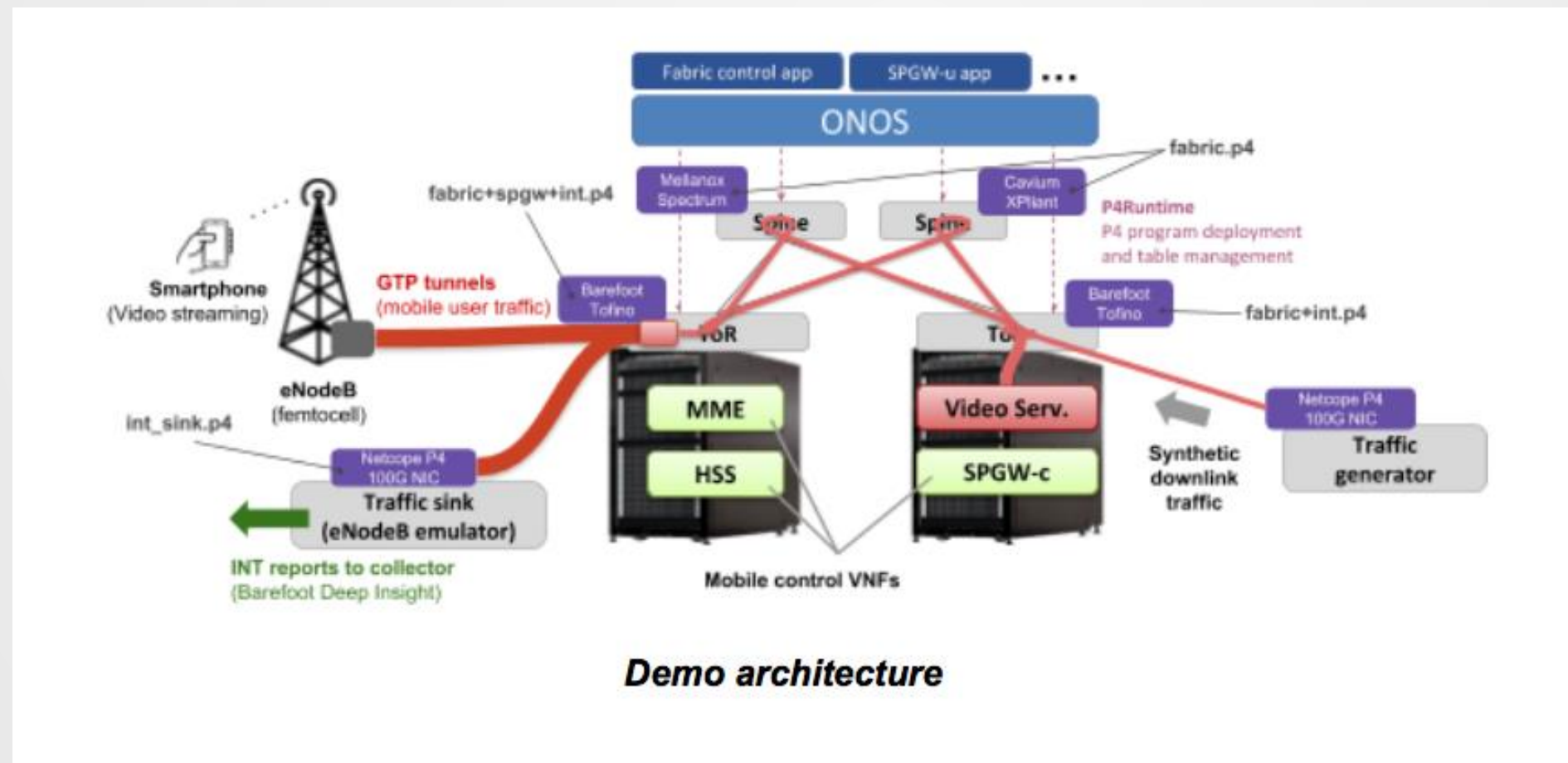
P4Runtime demo with CORD/ONOS

7/2018



ONOS/CORD integration

- Jan 2018 – Integrated at ONL
 - Mellanox Spectrum driver added to ONOS release 1.13
 - <https://github.com/opennetworkinglab/onos/releases/tag/1.13.1>
 - Spectrum/P4 Runtime wiki instructions added to onosproject.org
 - <https://wiki.onosproject.org/display/ONOS/Controlling+P4Runtime-enabled+Spectrum+switch+with+ONOS>
 - Spectrum demo at MWC Barcelona and ONS LA
 - Featured Spectrum as a fabric spine, next to Cavium spine and 2 BF leaves





Timed switch over

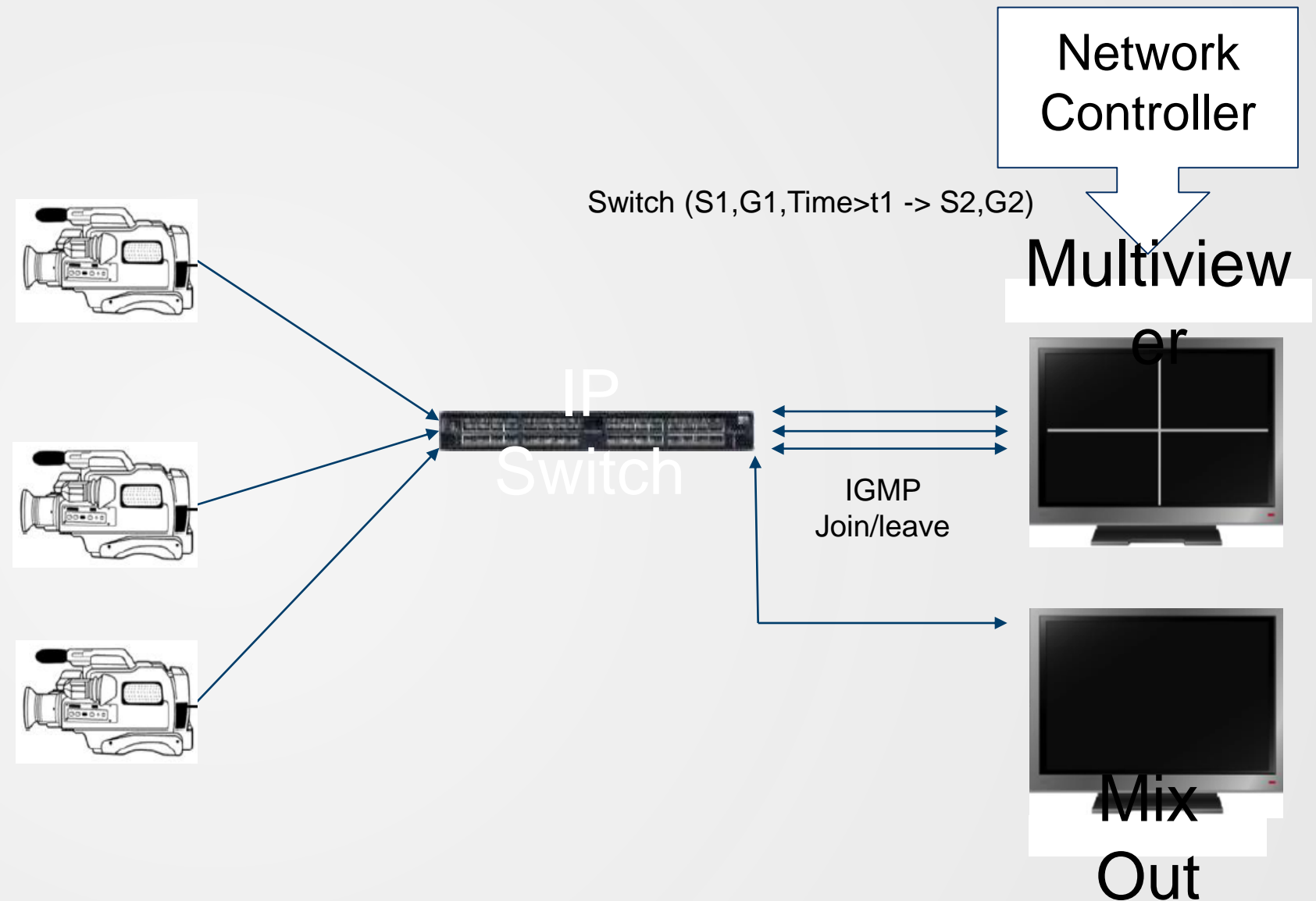
7/2018



Why is timed switch needed?

Current Solution:

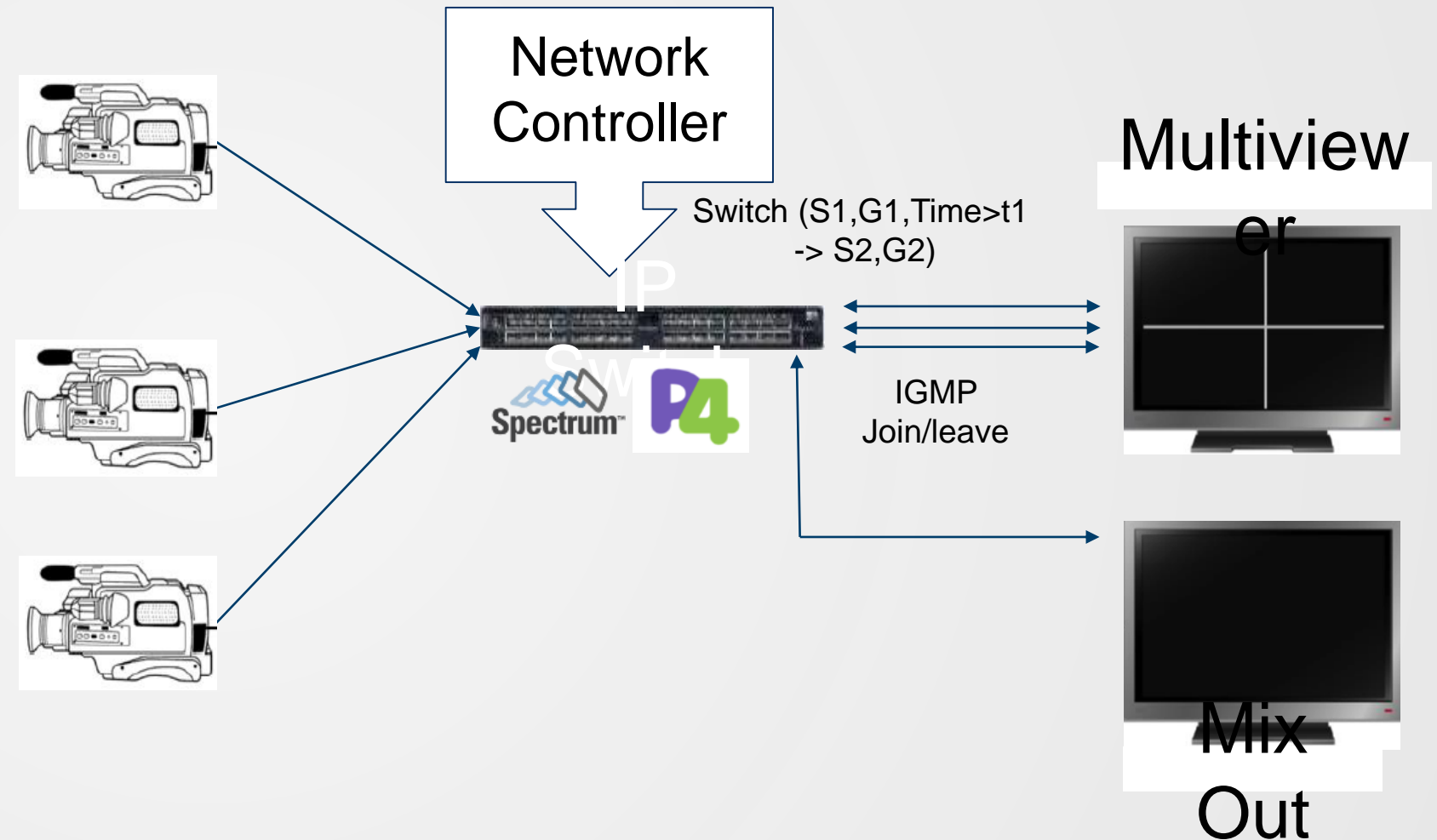
- The endpoint need to make a ‘clean’ switch between different media streams
 - Clean = switch the stream at the frame boundaries
- IGMP based implementation:
 - Use IGMP at the endpoint to join the new flow while receiving the old one
 - Buffer both streams at the endpoint and switch at the frame boundary to the new stream
 - IGMP leave the old flow
- Down side
 - **Endpoint link needs to reserve BW for both old and new streams**
 - Endpoint buffer need to have room for both streams
 - Latency due to buffer size



Why is timed switch needed?

Spectrum Programmable Pipeline Solution:

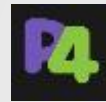
- Timed switch implementation:
 - Match on RTP timestamp on received media streams
 - All media flow time stamps are synchronized/locked. All packets from the same frame carry the same stamp
 - Switch between flows at the new timestamp value (exact match or regex)
- Advantages
 - Programmable hybrid pipeline: All the legacy protocols (IGMP/ PTP/ PIM/...) are operational along the per flow timed switch implementation
 - Network/endpoint links carries only relevant data i.e. link can be utilized to carry more streams
 - Reduced frame buffer and latency at the endpoints



Spectrum Programmable Hybrid Pipeline

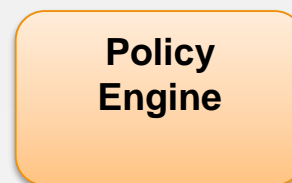
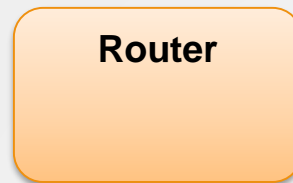


User programs

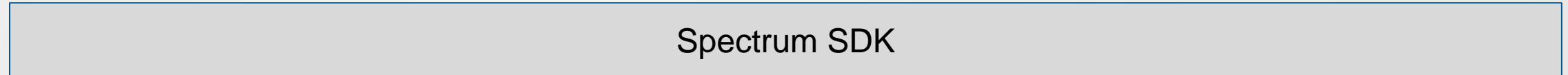


Application Containers

Switch timed



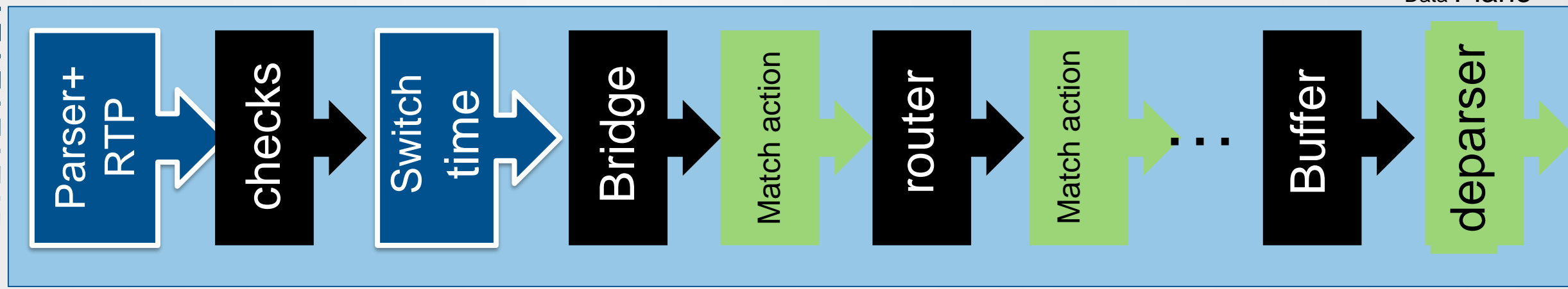
Auto generated SDK objects



SwitchTimed.p4

P4 Compiler

Data Plane



HW RO blocks

- Hybrid – the integration between legacy (switch router) and programmable pipeline
- NOS (ONYX) and user applications run in parallel

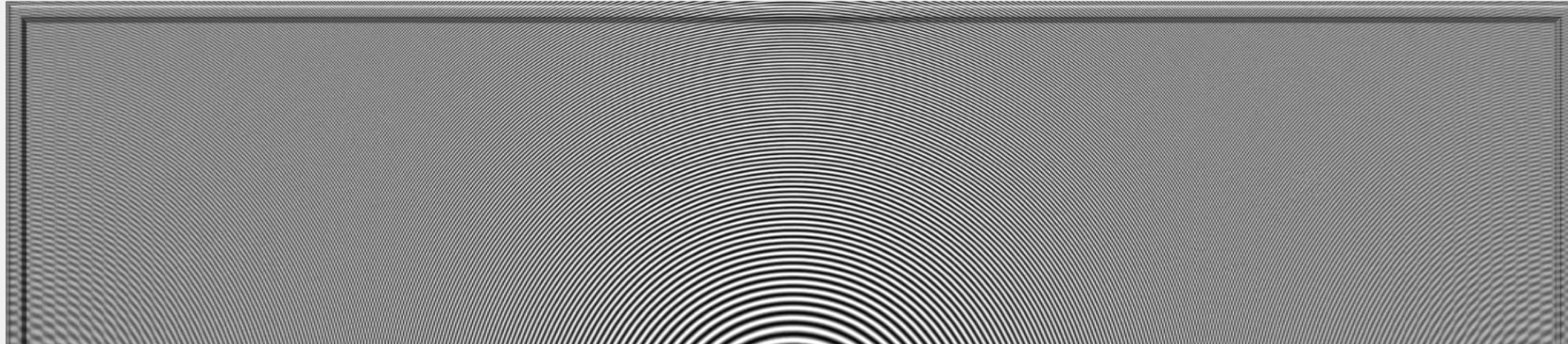


P4 timed switch/ salvo program

```
table table_timestamp {
    key = {
        headers.rtp.timestamp : range;
    }
    actions = {set_range_bitmap;}
    size = 256;
}
table table_ip_mc_forward{
    key = {
        standard_metadata.METADATA_REG : ternary;
        headers.ip.v4.dst_addr : exact;
        headers.ip.v4.src_addr : exact;
    }
    actions = {to_ports;}
    size = 256;
}
// pipe
apply{
    table_timestamp.apply();
    table_udp_port.apply();
    table_ip_mc_forward.apply();
}
}
control control_in_rif(inout Headers_t headers, inout metadata_t meta, inout standard_metadata_t standard_metadata){
    apply{}
}
control control_out_rif(inout Headers_t headers, inout metadata_t meta, inout standard_metadata_t standard_metadata){
    apply{}
}
control control_out_port(inout Headers_t headers, inout metadata_t meta, inout standard_metadata_t standard_metadata){
    apply{}
}
}
SpectrumSwitch(
    SalvoParser(),
    control_in_port(),
    control_in_rif(),
    control_out_rif(),
    control_out_port(),
    SalvoDeparser()
) main;
```


Timed Switch Demo

Switch between 2 streams on frame boundary, every 5 seconds



enps1s0f0.pcap

Apply a display filter ... <#>

No.	Time	Source	Destination	Protocol	Length	Info
178926	1.655554	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51403, Time=902367949
178927	1.655555	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51404, Time=902367949
178928	1.655555	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51405, Time=902367949
178929	1.655556	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51406, Time=902367949
178930	1.655556	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51407, Time=902367949
178931	1.655557	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51408, Time=902367949
178932	1.655557	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51409, Time=902367949, Mark
178933	1.656350	192.168.1.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51174, Time=902369749
178934	1.656353	192.168.1.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51175, Time=902369749
178935	1.656354	192.168.1.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51176, Time=902369749

Frame 178932: 1262 bytes on wire (10096 bits), 80 bytes captured (640 bits)

- Ethernet II, Src: Embrioni_4b:f0 (40:a3:6b:a0:4b:f0), Dst: IPv4mcast_01:02 (01:00:5e:00:01:02)
- Internet Protocol Version 4, Src: 192.168.0.1, Dst: 239.0.1.2
- User Datagram Protocol, Src Port: 10000, Dst Port: 20000
- Real-Time Transport Protocol
 - 10.. = Version: RFC 1889 Version (2)
 - ..0. = Padding: False
 - ...0 = Extension: False
 - 0000 = Contributing source identifiers count: 0
 - 1... = Marker: True
 - Payload type: DynamicRTP-Type-96 (96)
 - Sequence number: 51409
 - Timestamp: 902367949
 - Synchronization Source identifier: 0x00000000 (0)
- ST 2110_20 Data:SRD Segments length sum not n*180 - GPM
 - Extended Sequence Number: 0x528a
 - SRD Length: 1200
 - 0... = Field Identification: First field
 - .000 0010 0001 1011 = SRD Row Number: 539
 - 0... = Continuation: No

Sequence number (rtp.seq), 2 bytes

Packets: 583440 · Displayed: 583440 (100.0%)

Profile: Default

enps1s0f0.pcap

Apply a display filter ... <#>

No.	Time	Source	Destination	Protocol	Length	Info
178926	1.655554	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51403, Time=902367949
178927	1.655555	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51404, Time=902367949
178928	1.655555	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51405, Time=902367949
178929	1.655556	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51406, Time=902367949
178930	1.655556	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51407, Time=902367949
178931	1.655557	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51408, Time=902367949
178932	1.655557	192.168.0.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51409, Time=902367949, Mark
178933	1.656350	192.168.1.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51174, Time=902369749
178934	1.656353	192.168.1.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51175, Time=902369749
178935	1.656354	192.168.1.1	239.0.1.2	RTP	1262	PT=DynamicRTP-Type-96, SSRC=0x0, Seq=51176, Time=902369749

Frame 178933: 1262 bytes on wire (10096 bits), 80 bytes captured (640 bits)

- Ethernet II, Src: Embrioni_4b:f0 (40:a3:6b:a0:4b:f0), Dst: IPv4mcast_01:02 (01:00:5e:00:01:02)
- Internet Protocol Version 4, Src: 192.168.1.1, Dst: 239.0.1.2
- User Datagram Protocol, Src Port: 10000, Dst Port: 20000
- Real-Time Transport Protocol
 - 10.. = Version: RFC 1889 Version (2)
 - ..0. = Padding: False
 - ...0 = Extension: False
 - 0000 = Contributing source identifiers count: 0
 - 0... = Marker: False
 - Payload type: DynamicRTP-Type-96 (96)
 - Sequence number: 51174
 - Timestamp: 902369749
 - Synchronization Source identifier: 0x00000000 (0)
- ST 2110_20 Data:SRD Segments length sum not n*180 - GPM
 - Extended Sequence Number: 0x528a
 - SRD Length: 1200
 - 1... = Field Identification: Second field
 - .000 0000 0000 0000 = SRD Row Number: 0
 - 0... = Continuation: No

Sequence number (rtp.seq), 2 bytes

Packets: 583440 · Displayed: 583440 (100.0%)

Profile: Default



Thank You

