Lesson 14: SDN & P4

Van-Linh Nguyen

Fall 2024

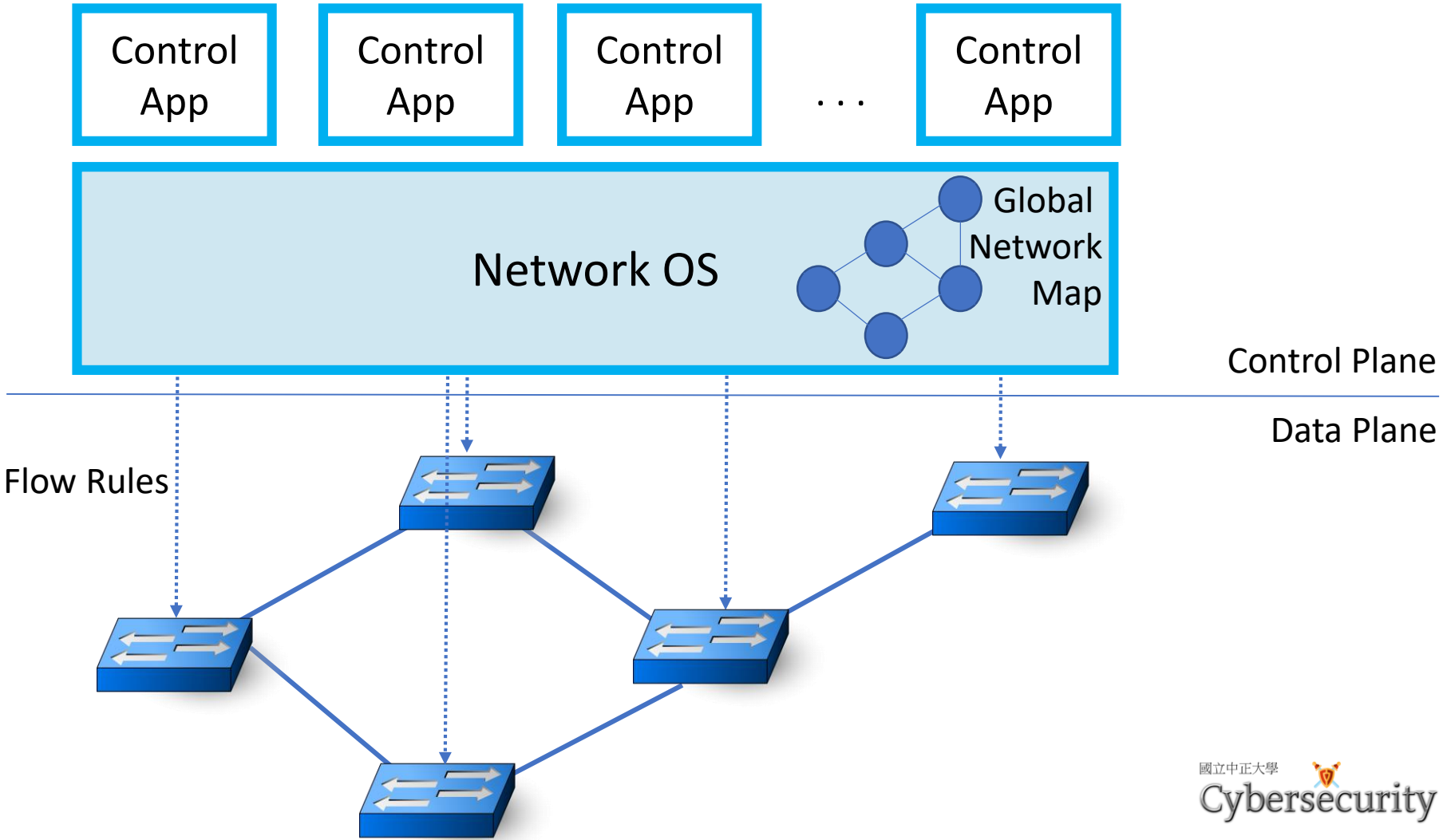
國立中正大學
Cybersecurity Lab

Outline

- SDN/ONOS – Part 2
- Programmable language for switch – P4

SDN applications

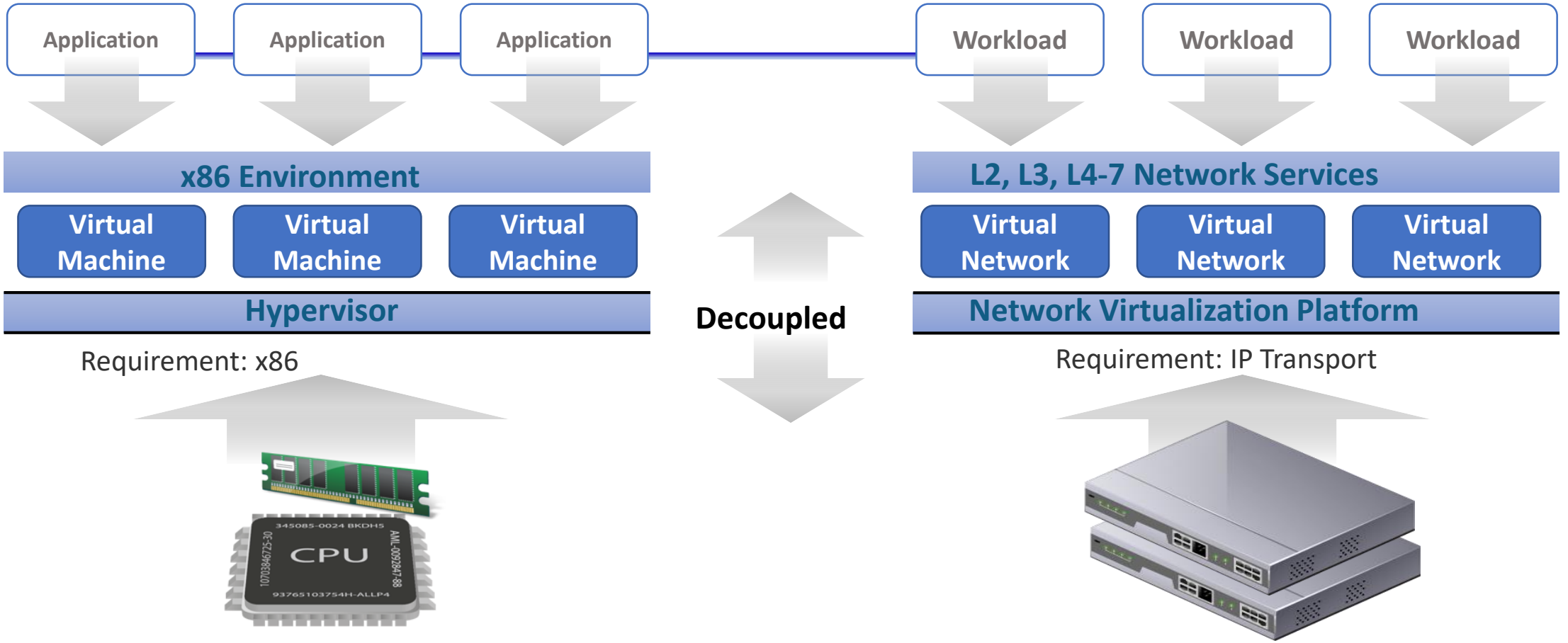
SDN Control and Data Planes



Use Cases

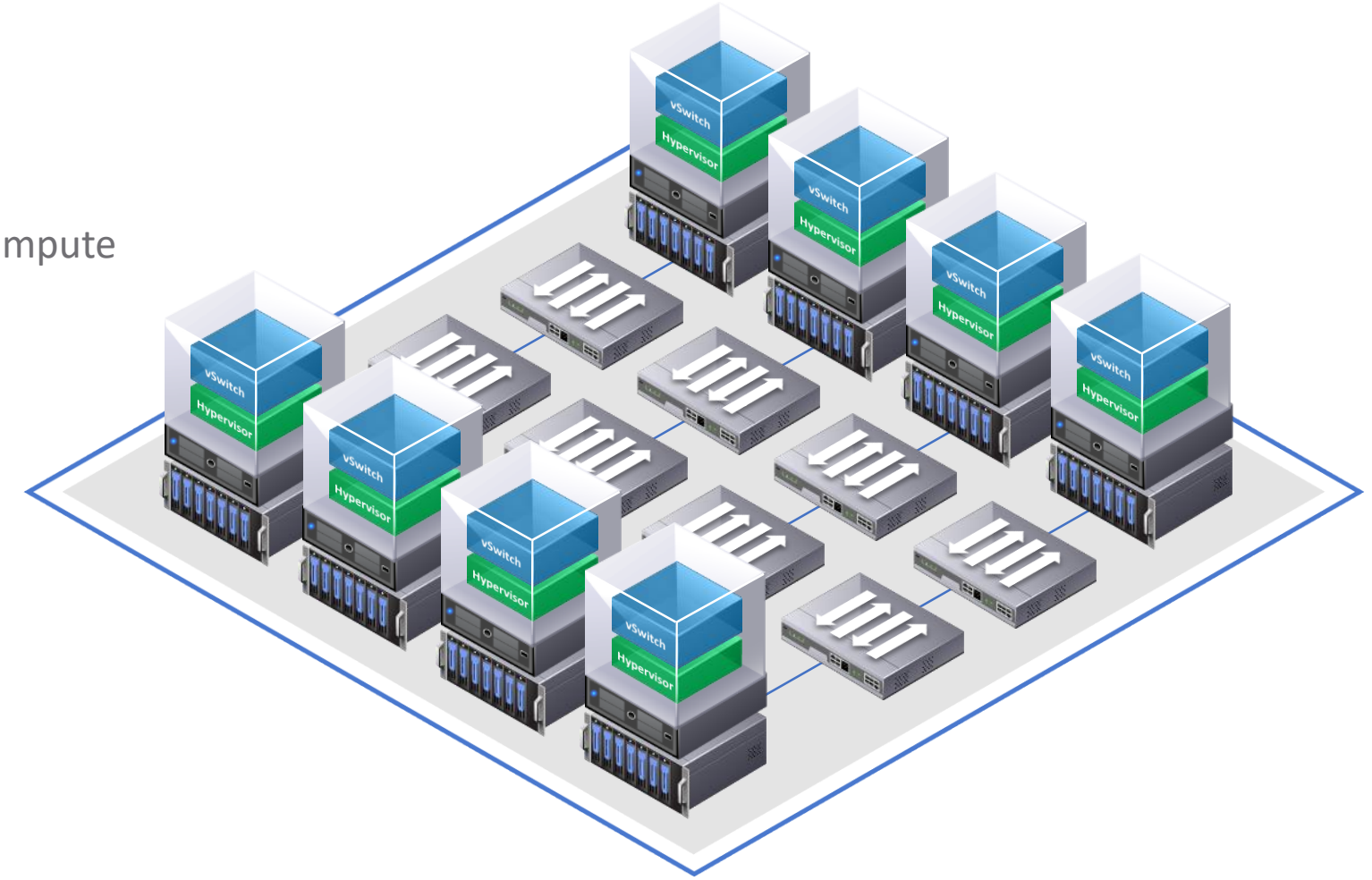
- Network Virtualization
- SD-WAN
- Traffic Engineering
- Bare Metal Switching
- Inband Network Telemetry

Network Virtualization – An Analogy



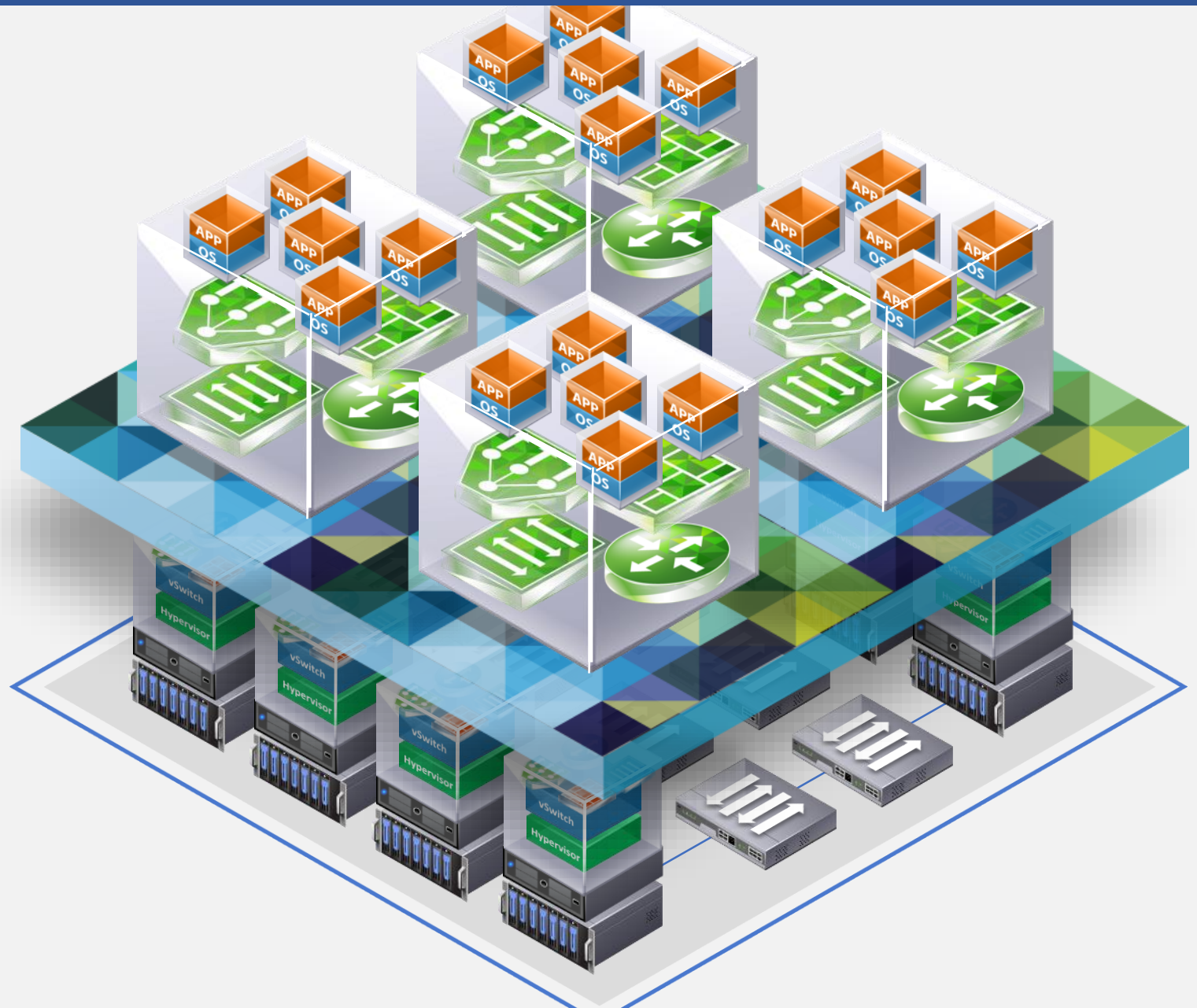
Virtual Machines to Virtual Networks

- Virtualization layer
- Network, storage, compute



Virtual Machines to Virtual Networks

- Virtual Data Centers
- “Network hypervisor”
- Virtualization layer
- Network, storage, compute



Network Virtualization Components

Cloud Consumption

- Self Service Portal
- OpenStack, Kubernetes, etc

Manager

- Single configuration portal
- REST API entry-point

Controller

- Manages Logical networks
- Run-time state
- Scale out, HA
- Separation of Control and Data Plane

Data Plane

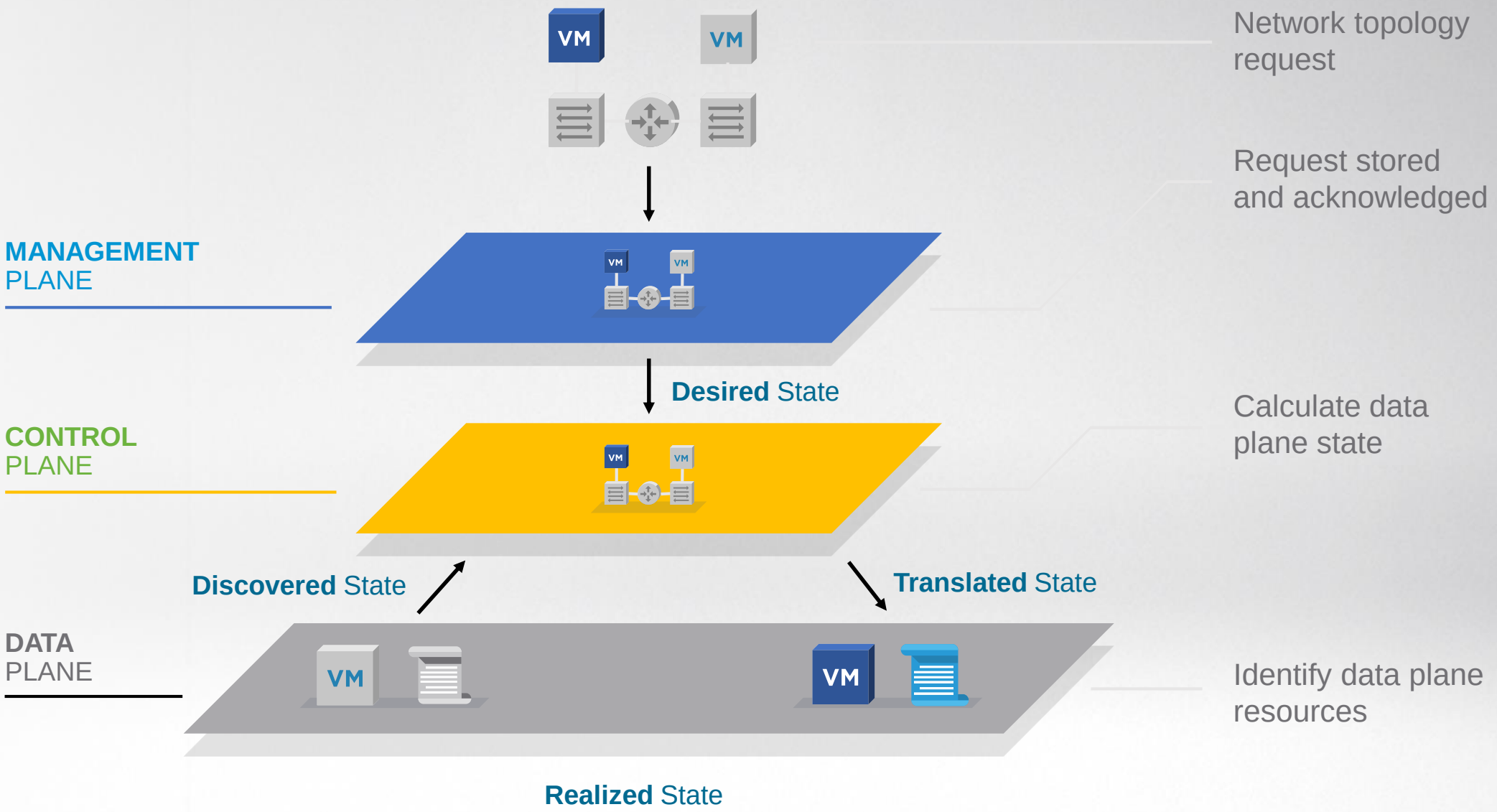
- High–Performance Data Plane
- Scale-out Distributed Forwarding Model

Distributed Services

- Logical Switch
- Distributed Logical Router
- Firewall
- Load Balancer

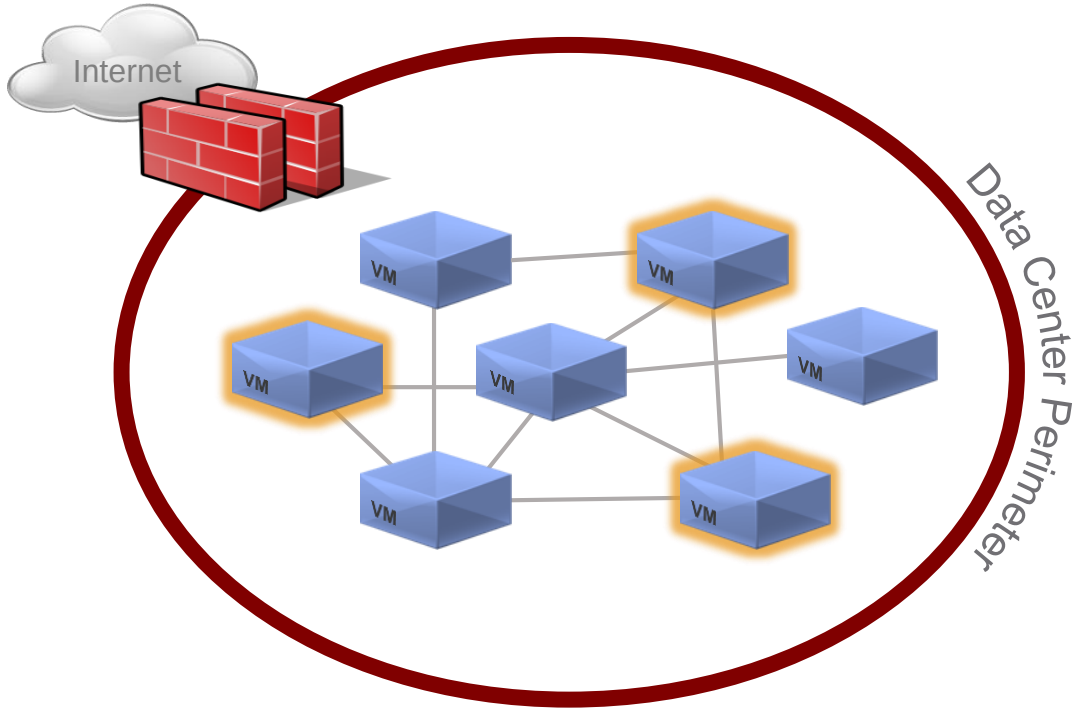
Virtual Edge

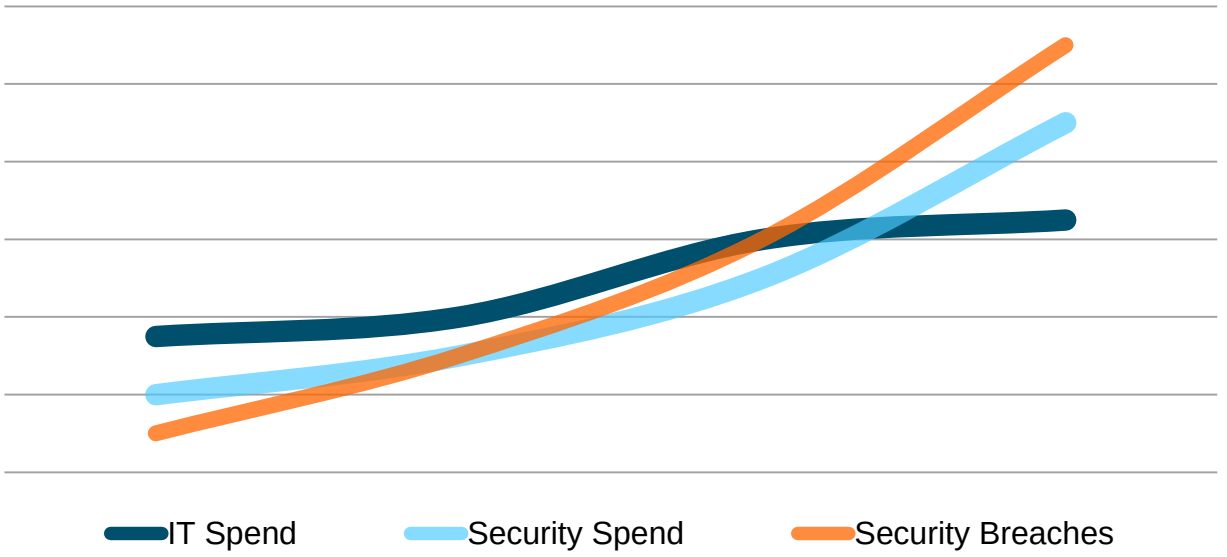
Management, Control and Data Planes



Problem: Data Center Network Security

Perimeter-centric network security has proven insufficient

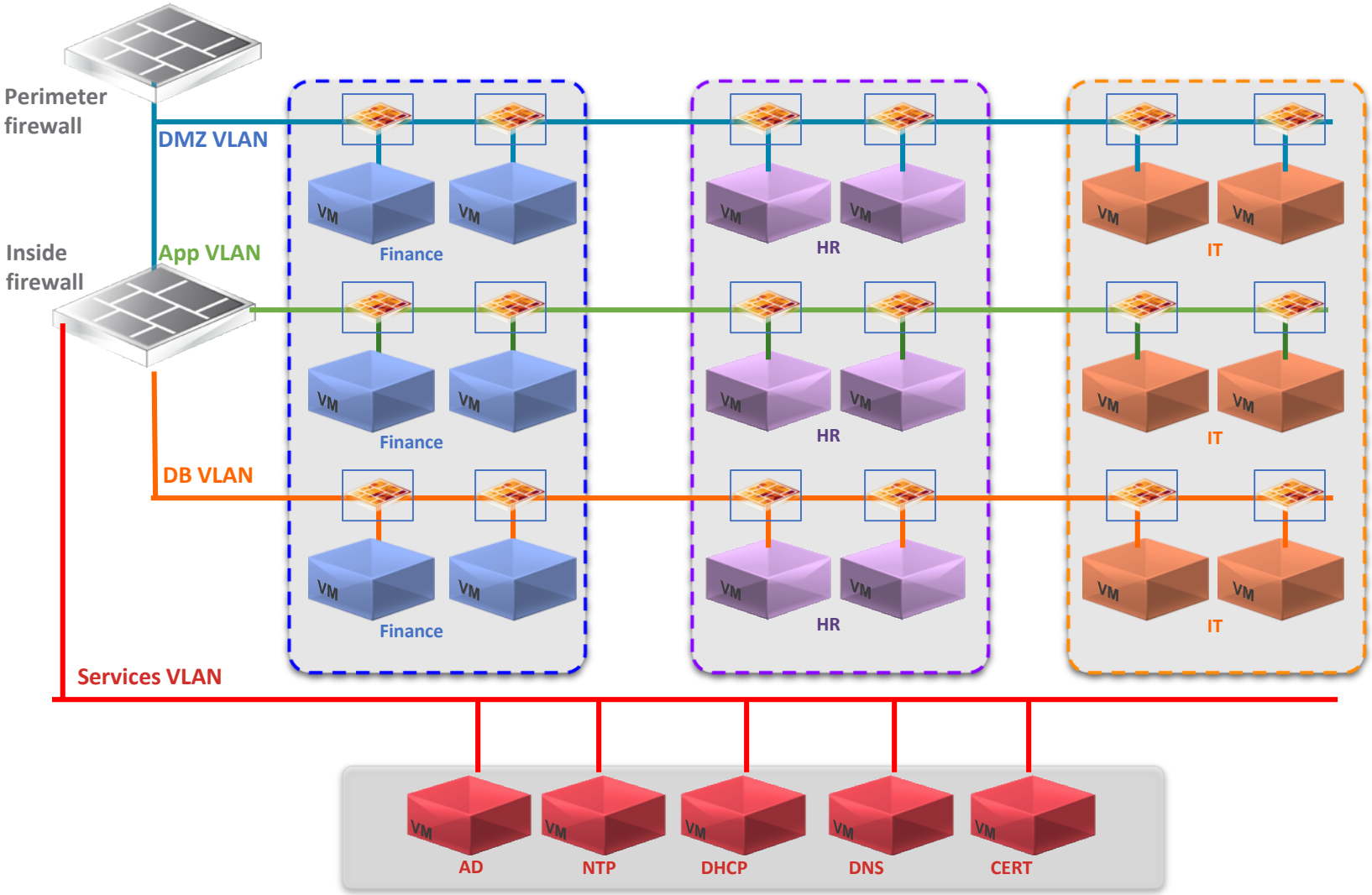




Today's security model focuses on perimeter defense

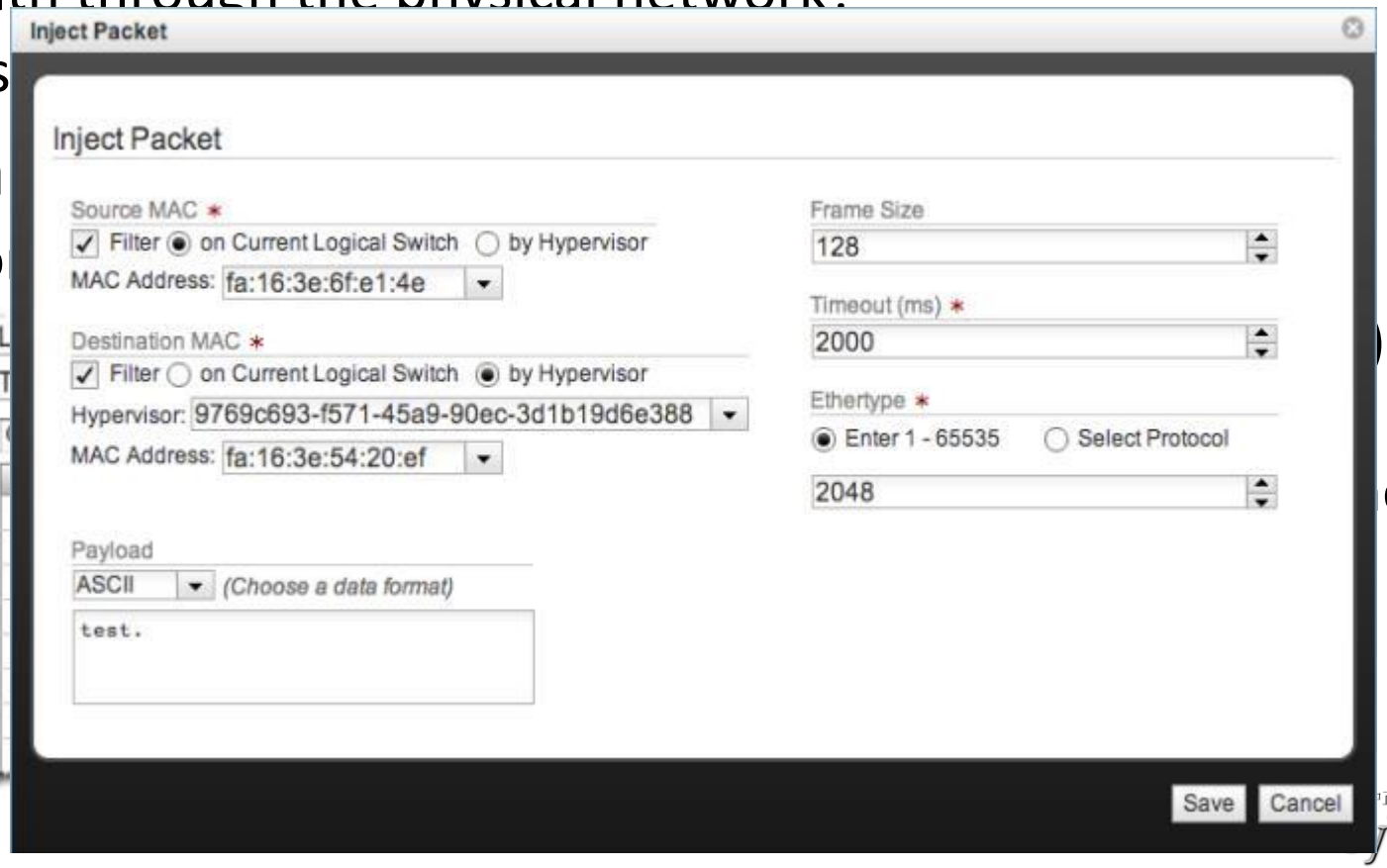
But continued security breaches show this model is not enough

Microsegmentation and Zero Trust



Visibility: changing the laws of physics

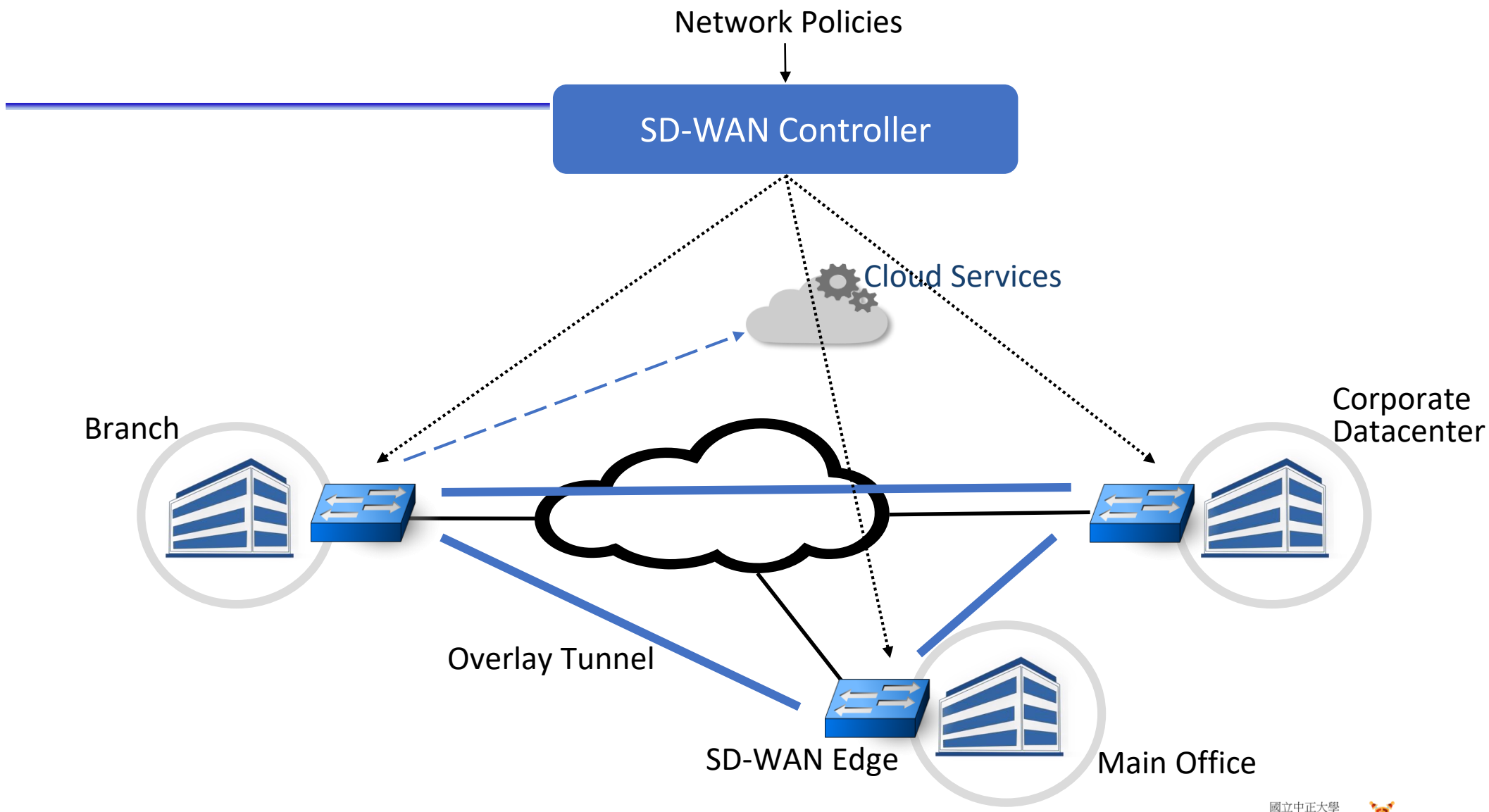
- Historically challenging to troubleshoot connectivity between VMs
  - Is the problem in vswitch or physical network?
  - What's the path through the physical network?
  - Is there a (mis
- Network virtua
  - Decompositio
  - Global view) and tunnel health from
  - Synthetic tr em



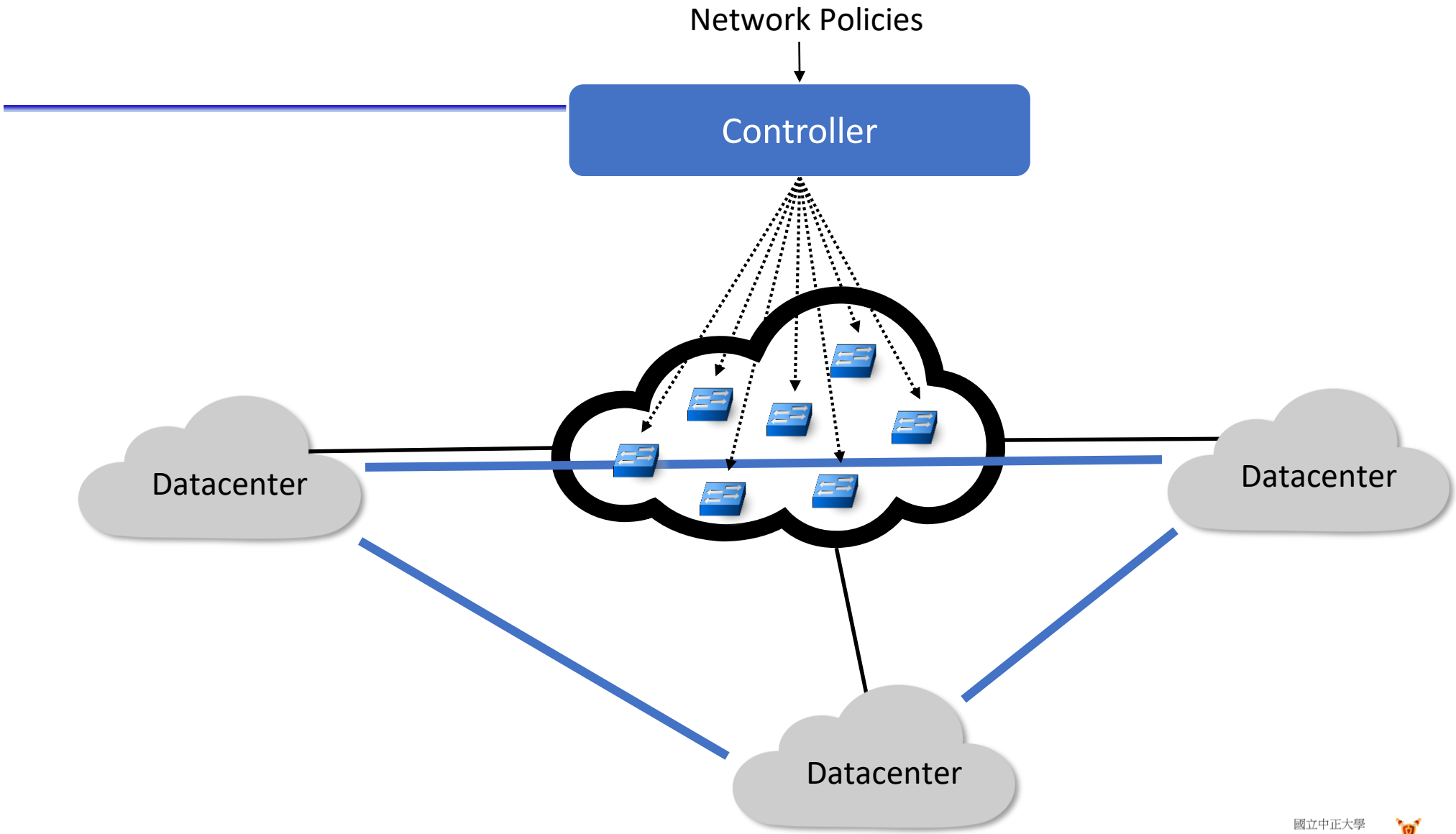
Network Virtualization – Discussion

- 90% of Fortune 100 have deployed network virtualization
- Foundational to hyperscale data centers
- Network configuration no longer the “long pole”
- A key step towards better network security (but much work remains)
- Increasingly important for microservices, kubernetes etc.
- Commodifying effect on physical networking
- Service Mesh can be viewed as a form of Network Virtualization

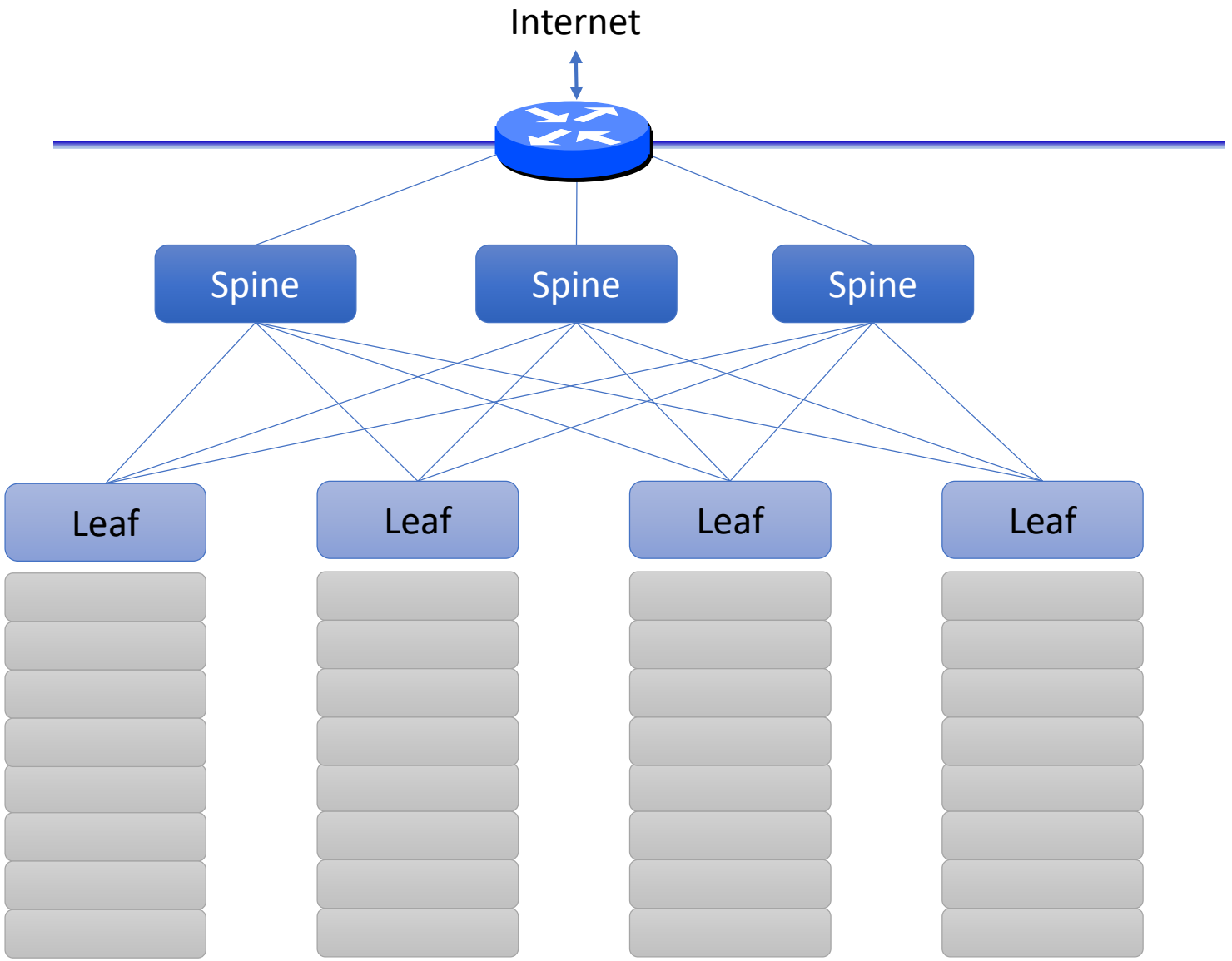
SD-WAN



Traffic Engineering



Datacenter Switching Fabric



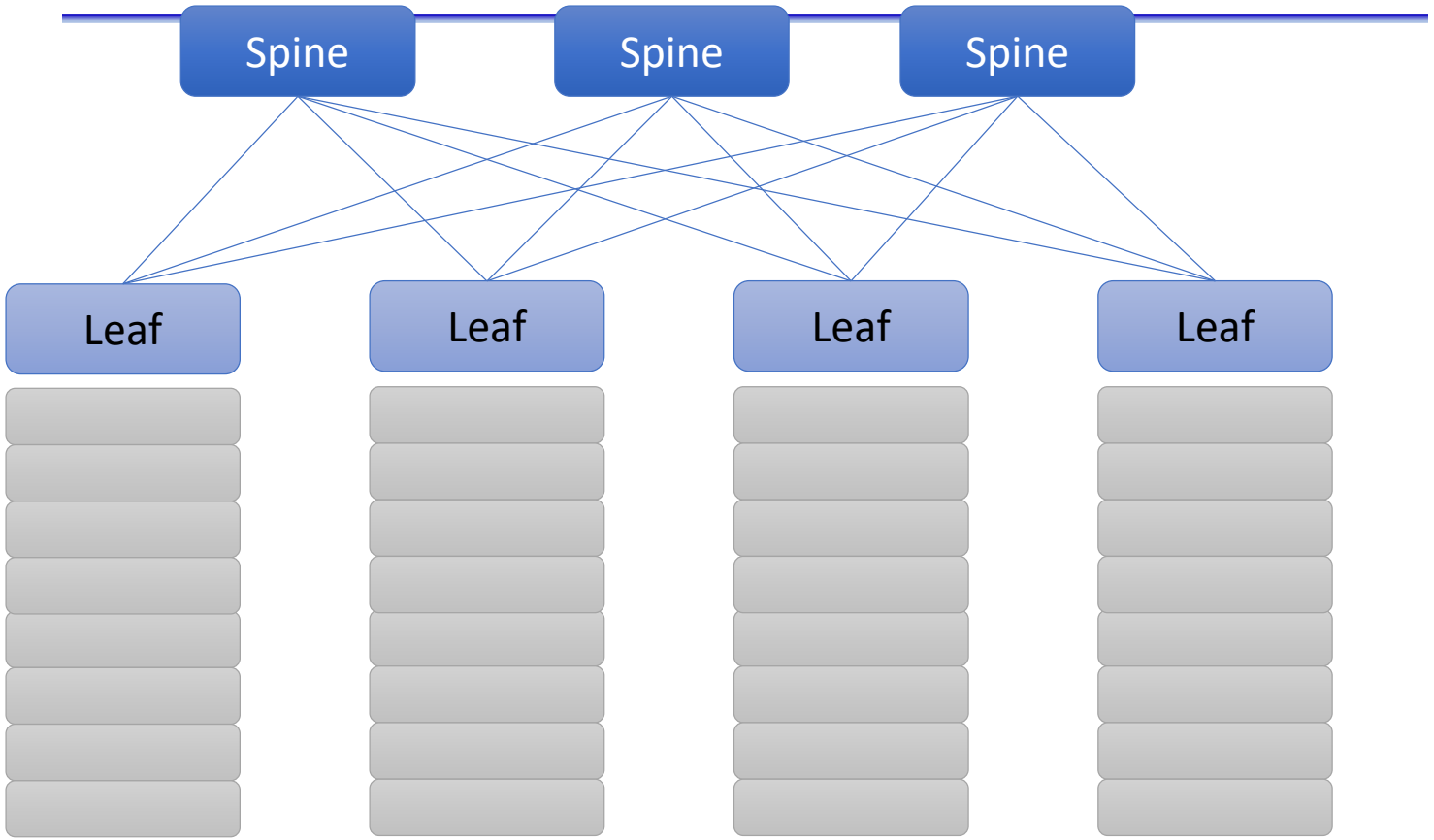
Leaf-Spine Topology

- Leaf Switches = Top-of-Rack (ToR)
- Optimized for East-West Traffic
- Built-in Redundancy (not shown)
- Scale with additional layers

Well-Established in Commodity Clouds

- Bare-Metal Switches
- Control Plane running in the cloud

Leaf-Spine Switching Fabric



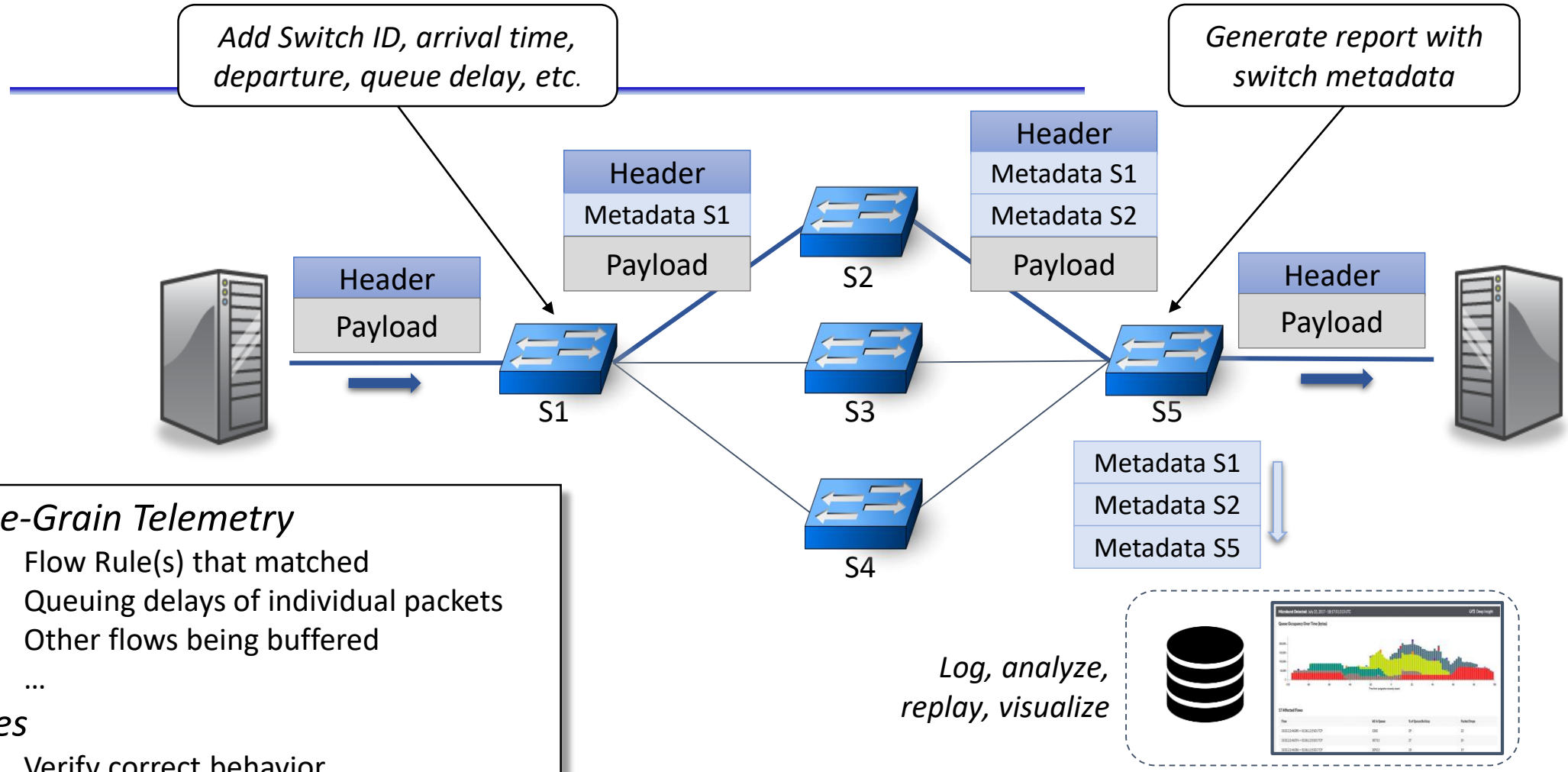
Trellis Design

- Intra-Rack: L2 Domain within L3 Subnet
- Inter-Rack: L3 Routing between Subnets
- Segment Routing across Fabric

Trellis Features

- VLANs / QinQ
- End-to-End L2 Tunnels
- IPv4 / IPv6 Routing
- Multicast (with IGMP)
- ARP (IPv4) / NDP (IPv6)
- DHCPv4 / DHCPv6
- High Availability

Inband Network Telemetry (INT)



Fine-Grain Telemetry

- Flow Rule(s) that matched
- Queuing delays of individual packets
- Other flows being buffered
- …

Uses

- Verify correct behavior
- Identify micro-bursts
- …

Recall ONOS
Open Network Operating System

What is ONOS?

SDN/NFV Controller Platform

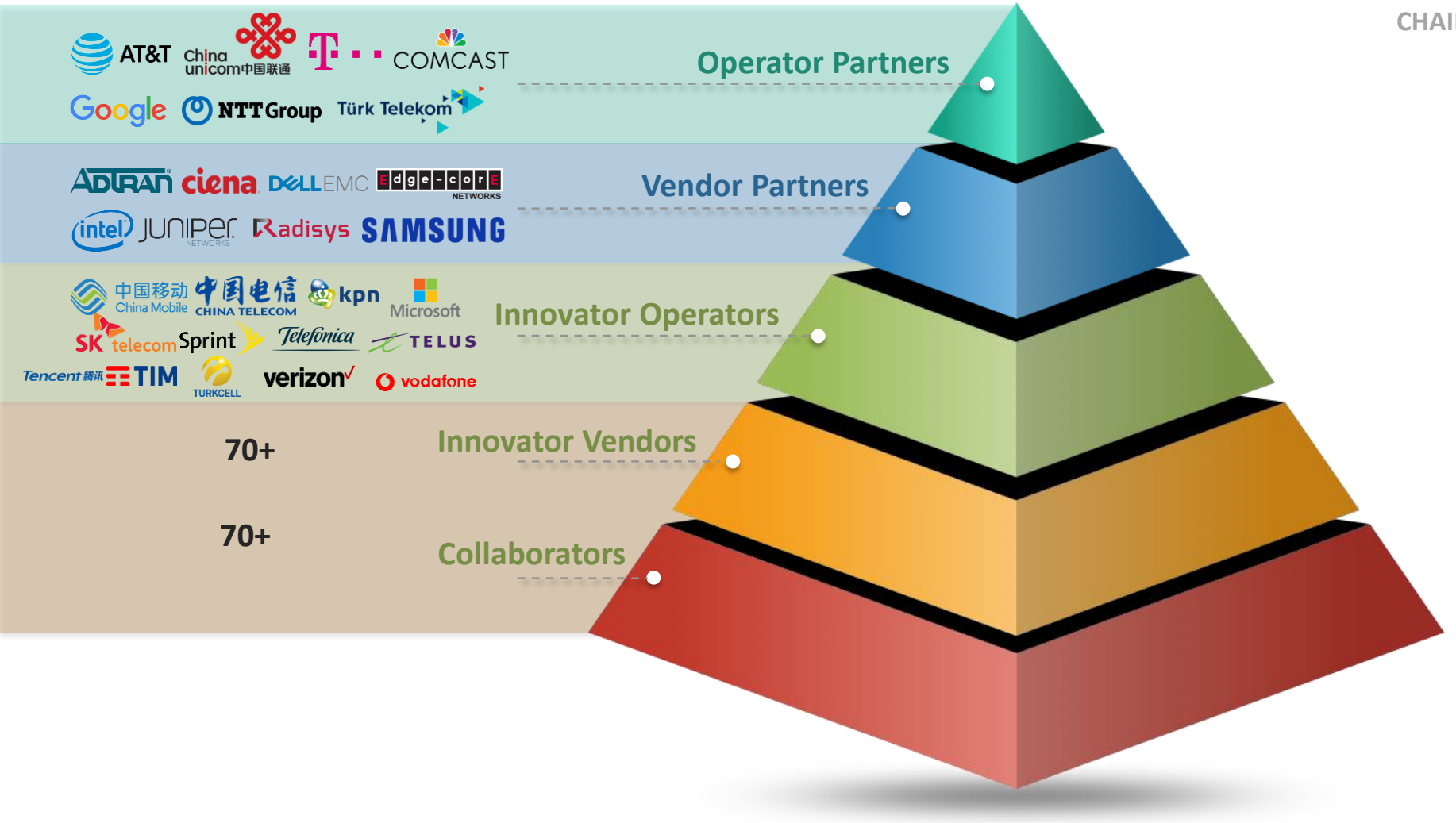
Java

Open Source: Apache 2.0

Source: https://github.com/opennetworkinglab/onos

The ONF Ecosystem – 160+ Members Strong

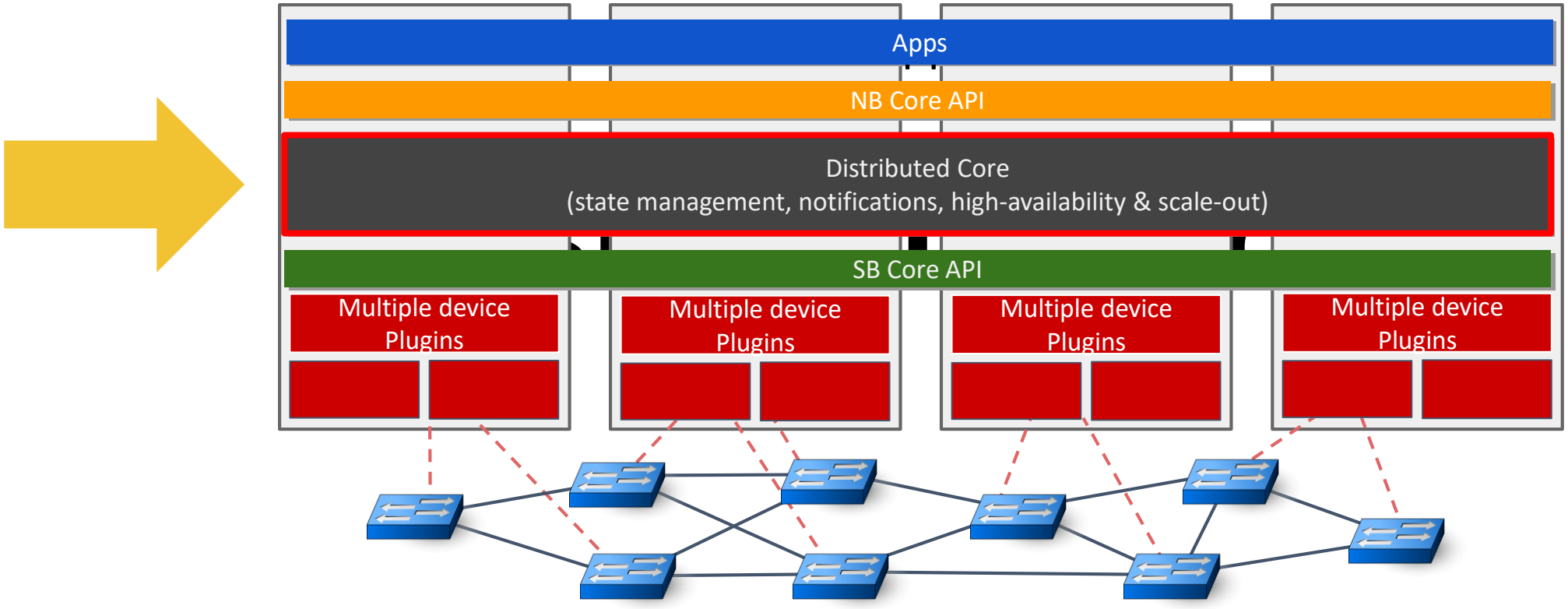
Vibrant Operator Led Consortium Positioned for Success

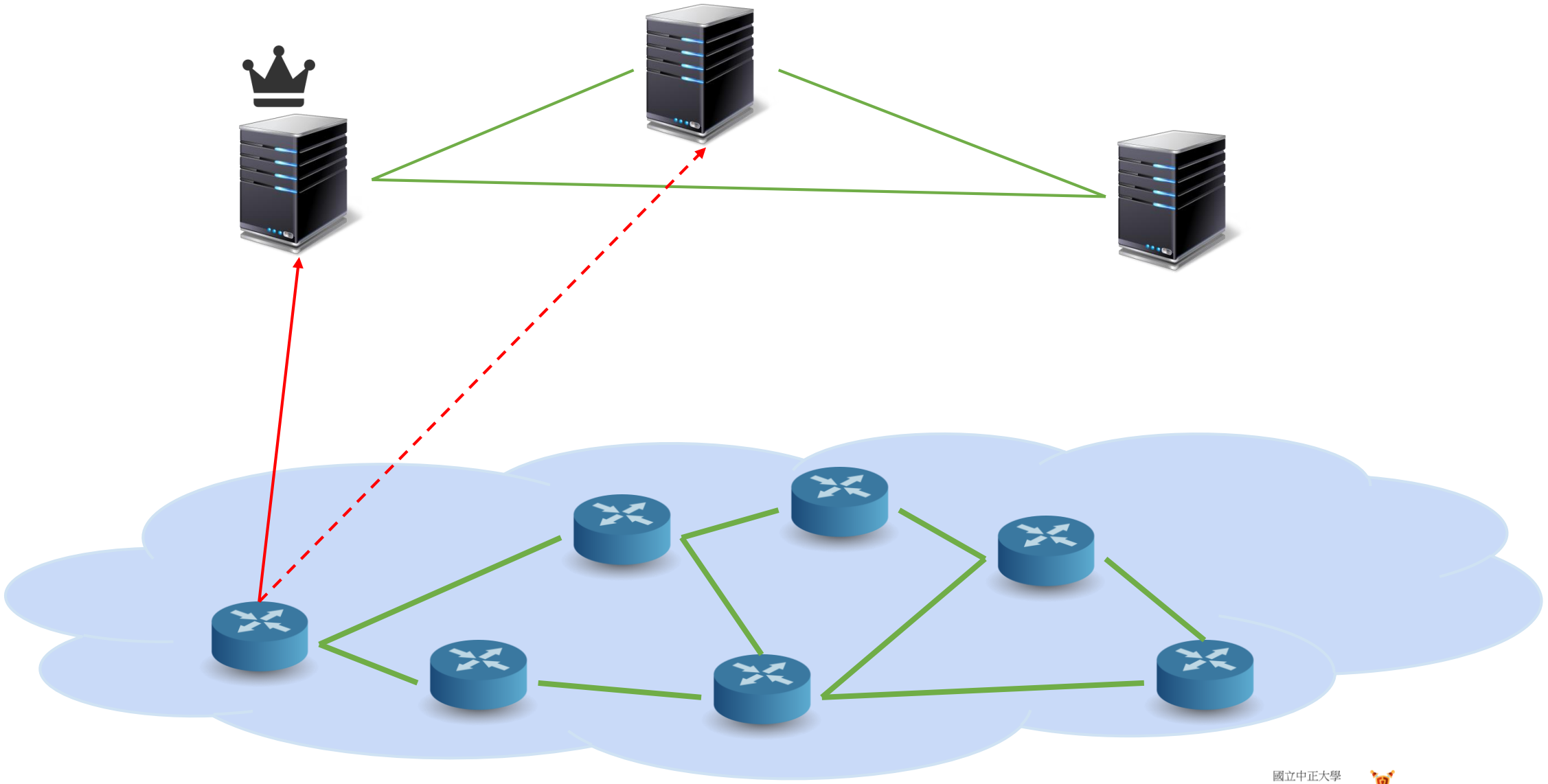


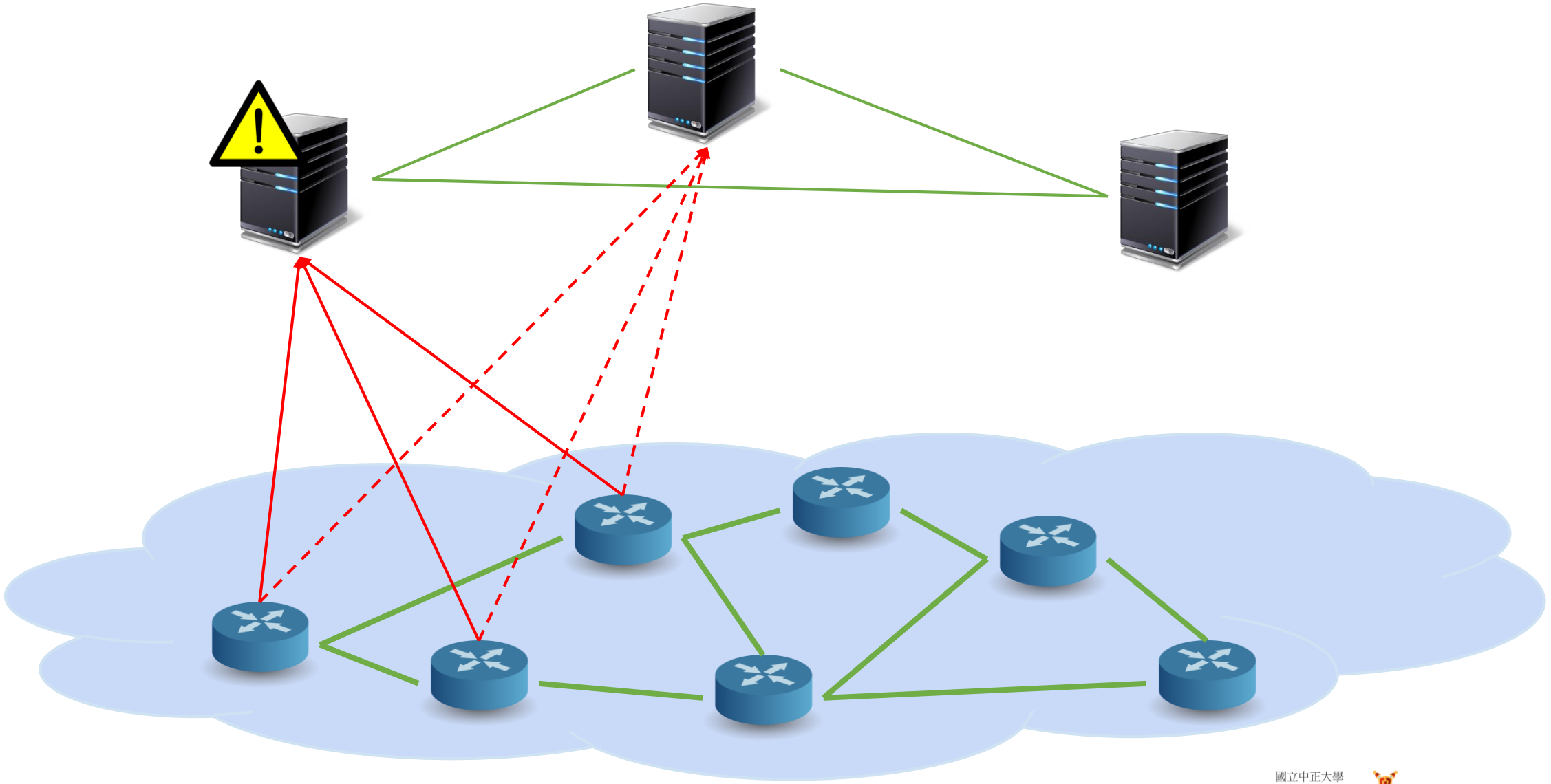
ONF BOARD

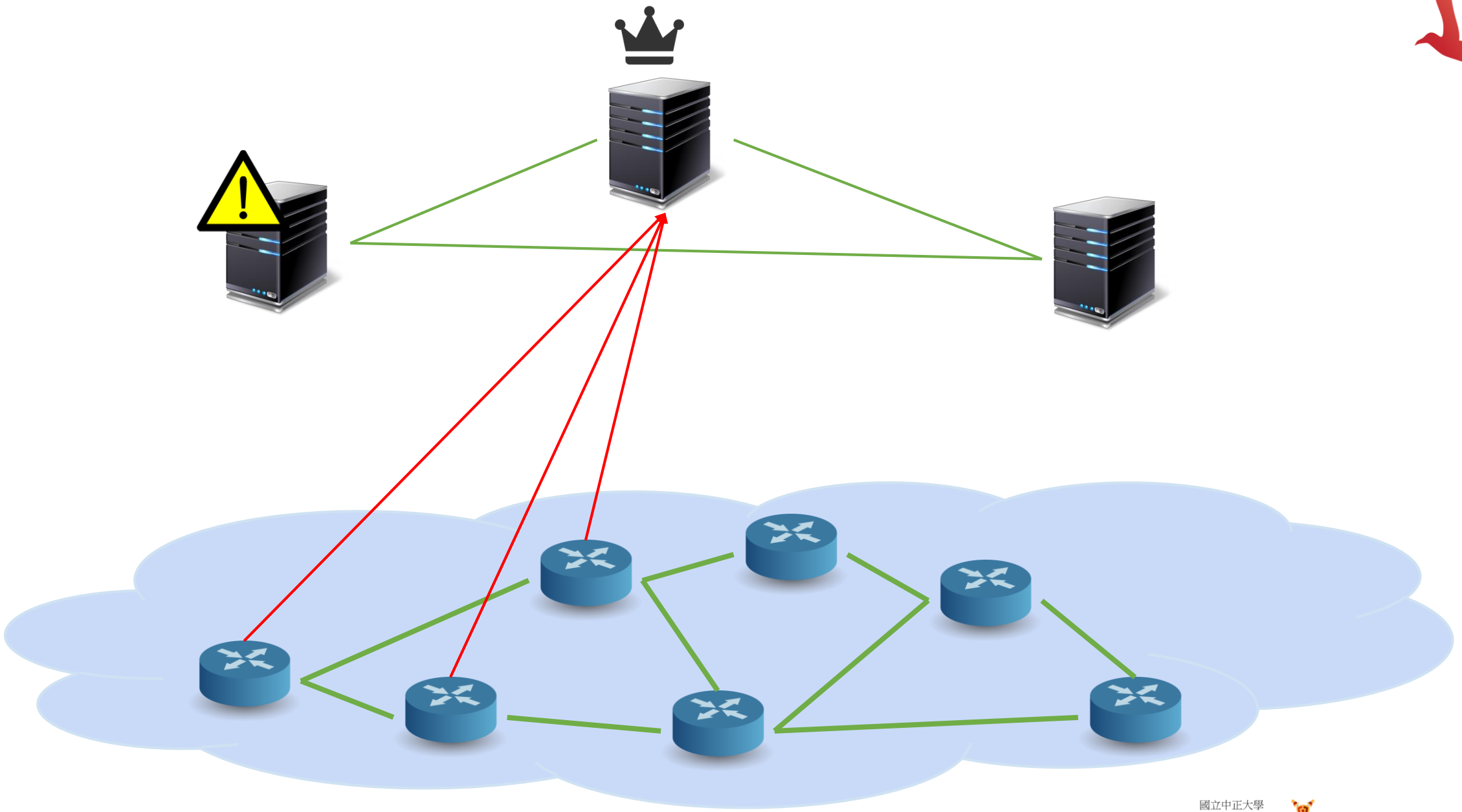
CHAIR: AT&T	Andre Feutsch - CTO
T-Mobile	Jochen Appel - VP
Google	Amin Vahdat - Fellow
NTT Group	Dai Kashiwa - Director
COMCAST	Rob Howald - VP
China unicom中国联通	Shao Guanglu - SVP
Telefónica	Patric Lopez - VP
Türk Telekom	Firay Yaman Er - CSO
STANFORD	Nick McKeown - Prof
ONF STANFORD	Guru Parulkar, Exec Dir

Distributed Core

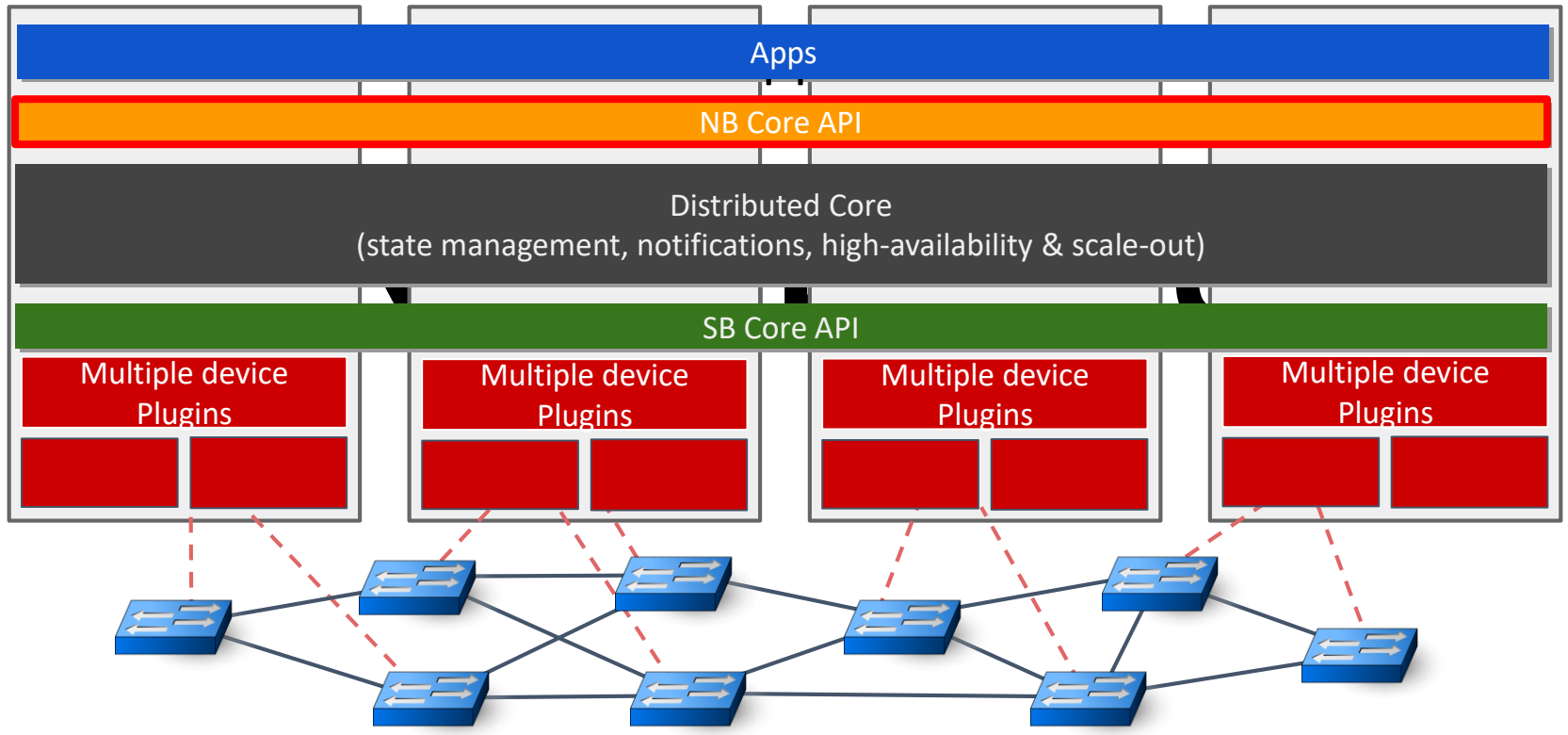






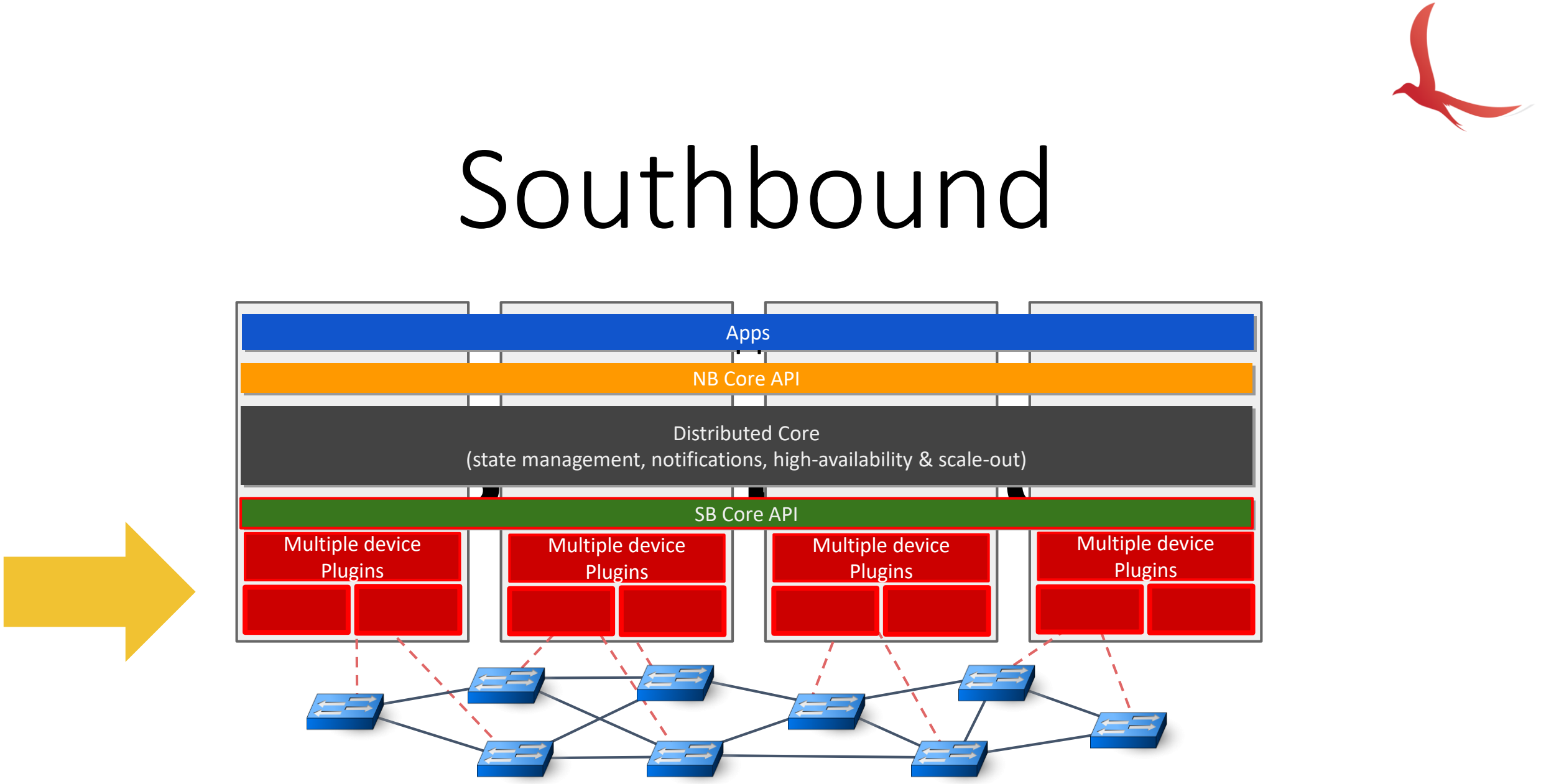


Northbound


Apps
NB Core API
Distributed Core
(state management, notifications, high-availability & scale-out)
SB Core API
Multiple device Plugins
Multiple device Plugins
Multiple device Plugins
Multiple device Plugins

Key Northbound Abstractions

- **Network Graph**
  - Directed, cyclic graph comprising of infrastructure devices, infrastructure links and end-station hosts
- **Flow Objective**
  - Device-centric abstraction for programming data-plane flows in version and vendor-independent manner
- **Intent**
  - Network-centric abstraction for programming data-plane in topology-independent manner

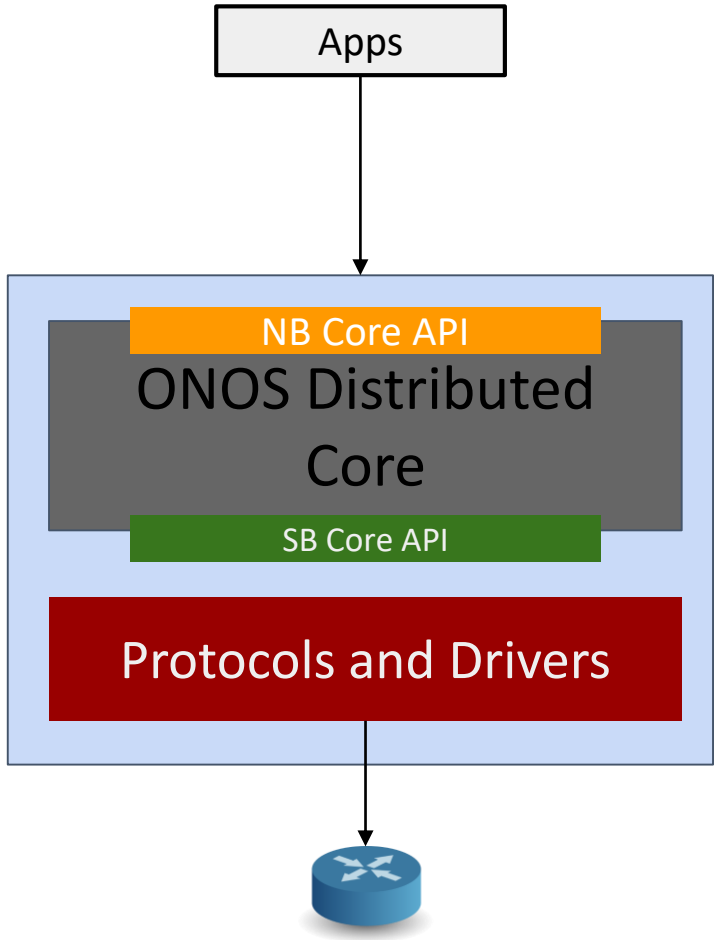

Apps
NB Core API
Distributed Core
(state management, notifications, high-availability & scale-out)
SB Core API
Multiple device Plugins
Multiple device Plugins
Multiple device Plugins
Multiple device Plugins

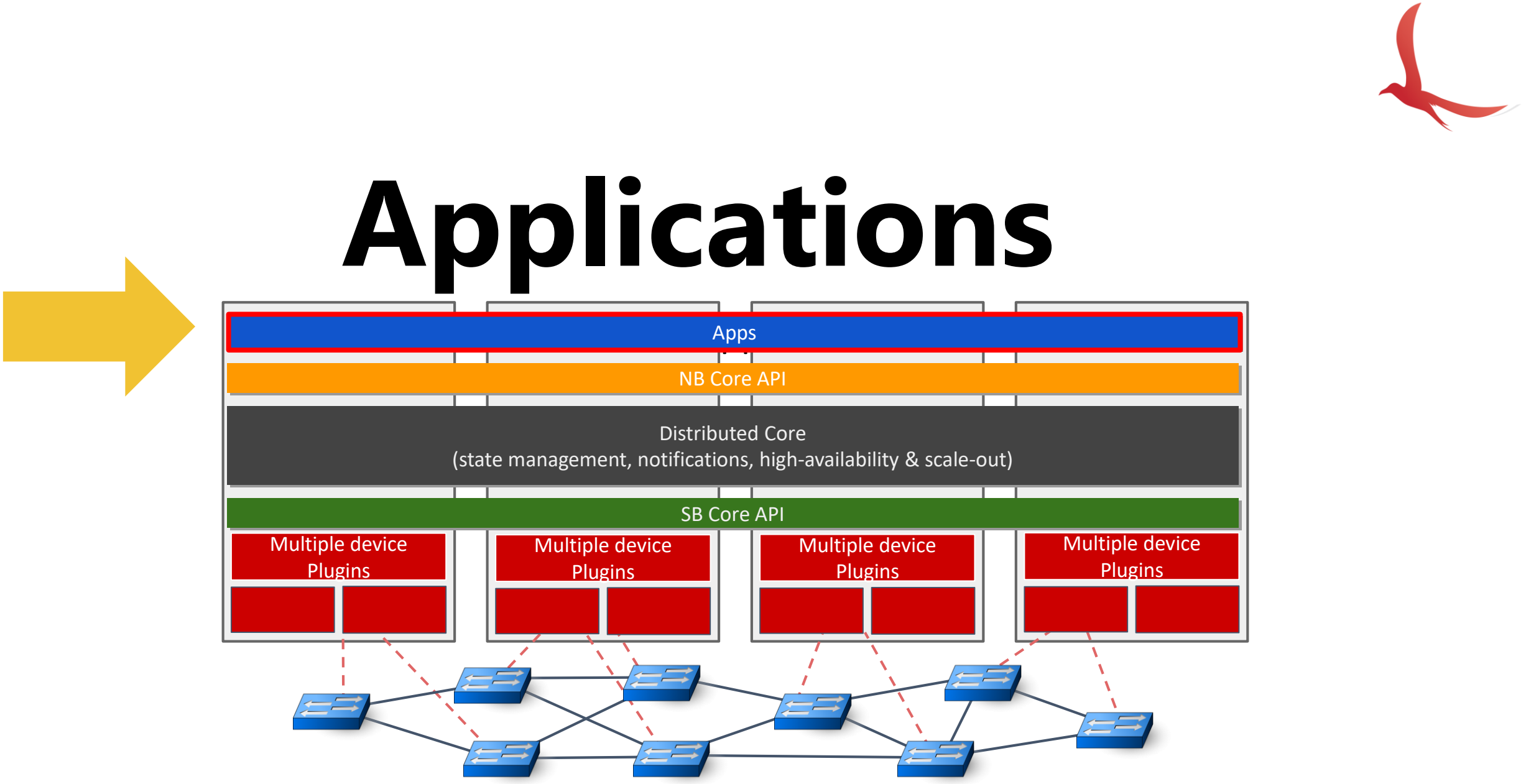
Southbound

Southbound overview

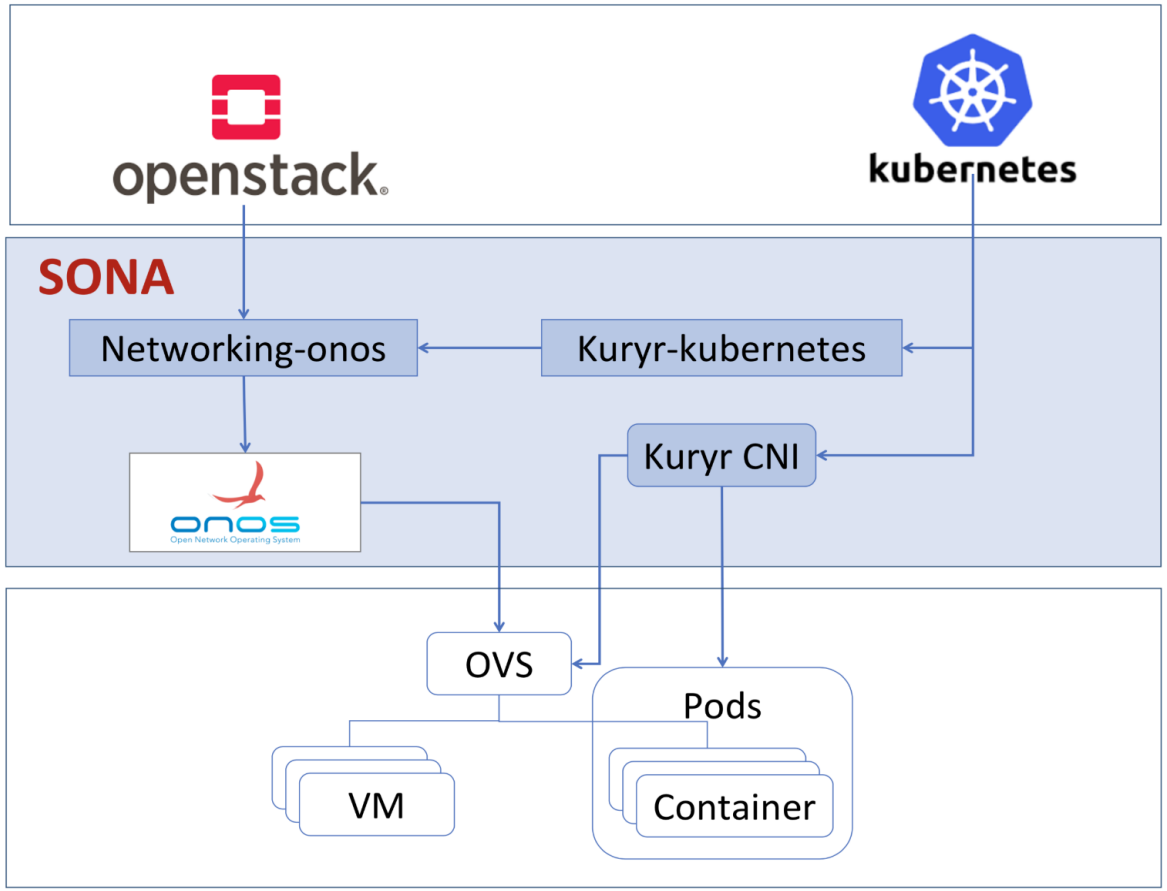
Southbound protocols:

- OpenFlow 1.0 to 1.5
- OVSDB
- NETCONF + YANG
- SNMP
- P4 → P4Runtime
- BGP, ISIS, OSPF
- PCEP
- REST
- LISP
- gNMI




Applications
Apps
NB Core API
Distributed Core
(state management, notifications, high-availability & scale-out)
SB Core API
Multiple device Plugins
Multiple device Plugins
Multiple device Plugins
Multiple device Plugins

Simplified Overlay Network Architecture



Apps/Use Cases

Trellis and Segment Routing

- DC leaf-spine fabric

Optical Disaggregated Transport Network

- Control of optical devices
- OpenRoadm

P4

- Control of Data-Plane programmable switches
- INT, VNF Offloading, Fabric.P4

vRouter

- CORD virtual Router

OpenStack Integration

- Sona Project

DHCP

- DHCP app
- DHCP relay

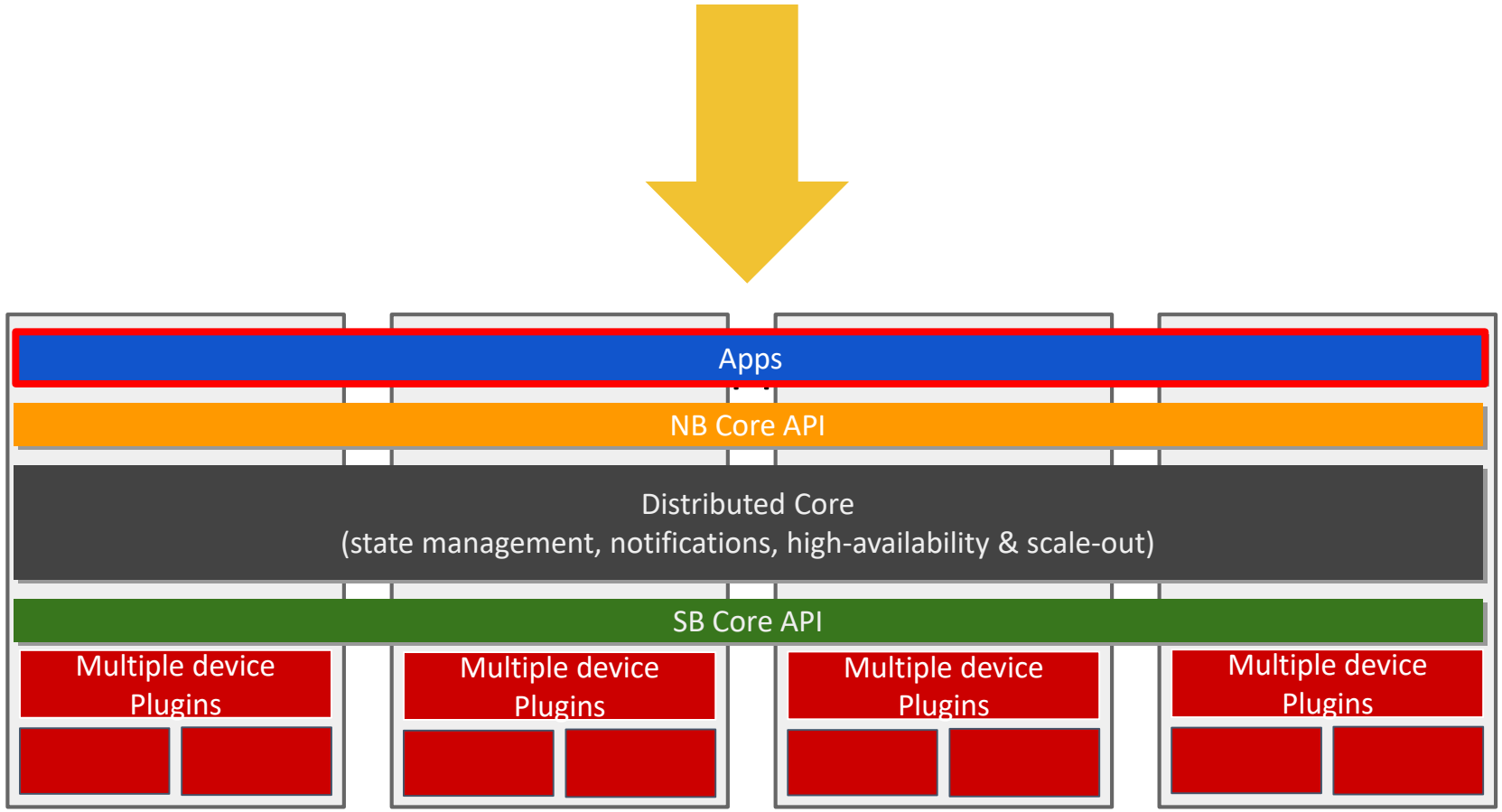
L2Monitoring and FaultManagement

Air Traffic Control

How to develop applications

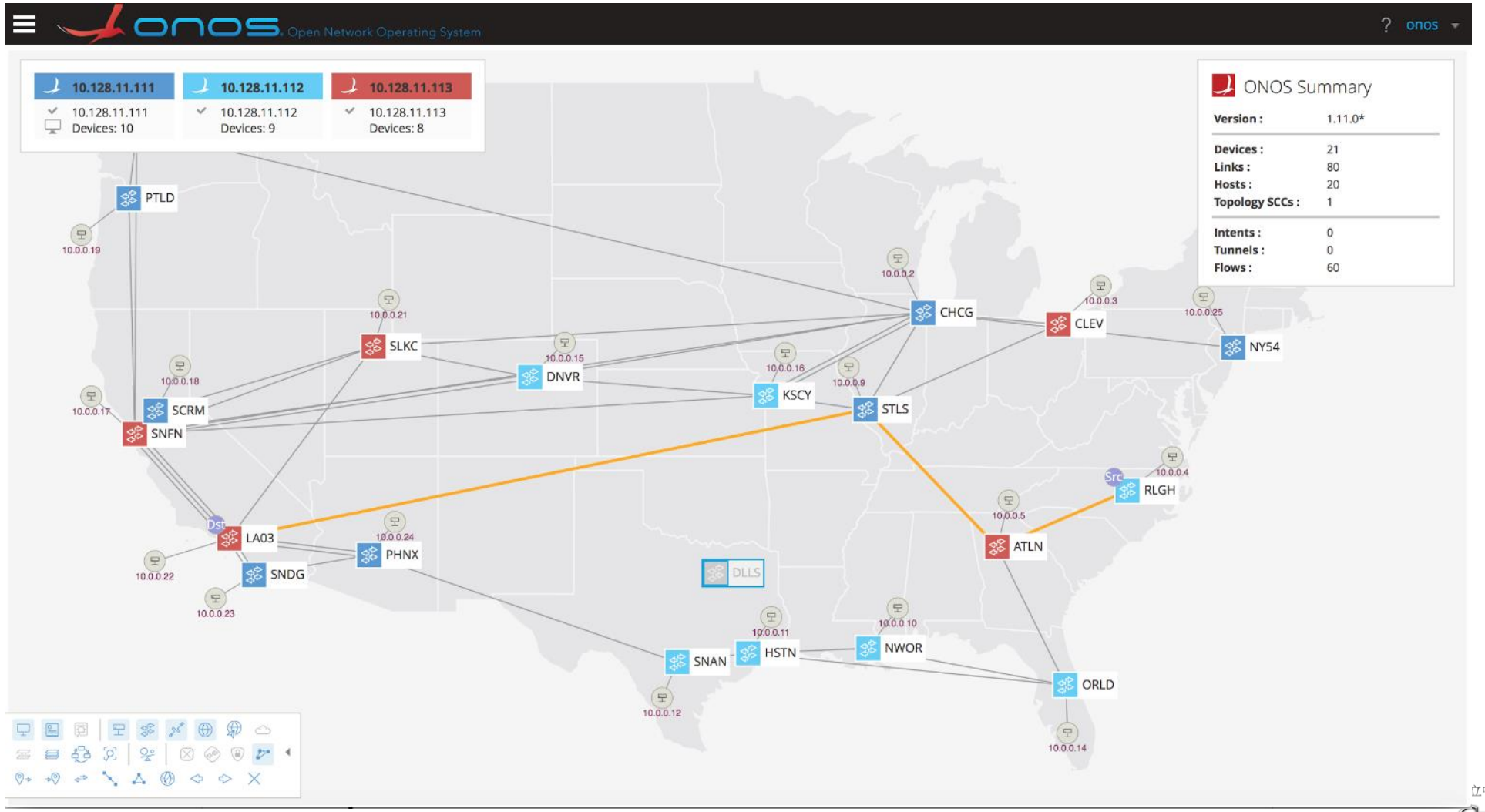
- Clone or create new repository
- Create a temple application or clone an existing
- Write your code (Java, Javascript, etc)
- Integrate it into the buildsystem
- Build (Bazel or Maven)
- Launch ONOS
- Activate the application (OSGI)
- Test, debug, and release

Interact with ONOS



Interact with ONOS: GUI

UI: <onos-ip>:8181/onos/ui



Interact with ONOS: CLI

$onos <controller_address>

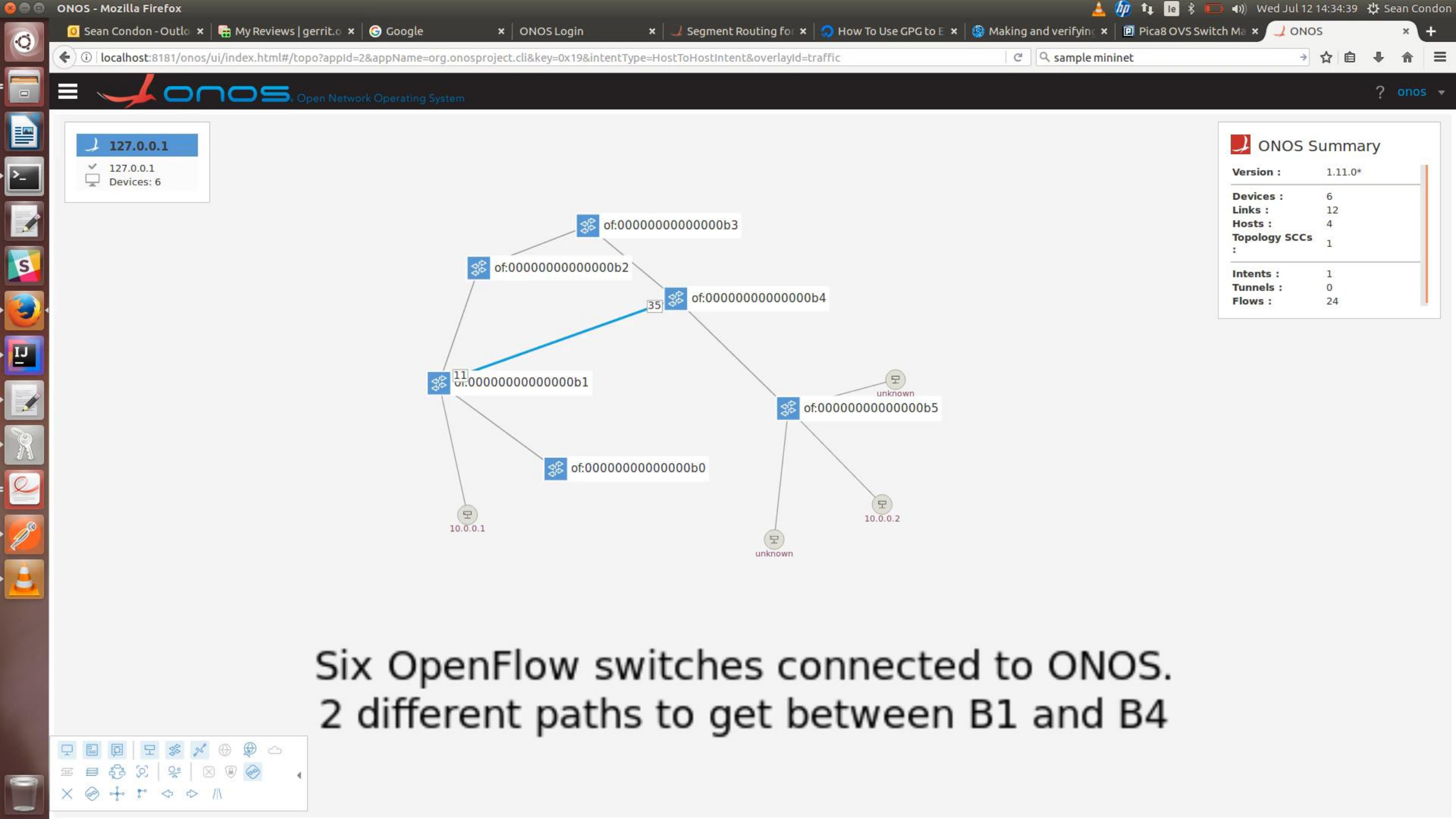
```
Welcome to Open Network Operating System (ONOS)!

Documentation: wiki.onosproject.org
Tutorials:     tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos>
```

Interact with ONOS: REST and GRPC

REST APIs: <onos-ip>:8181/onos/v1/docs/

Northbound GRPC with protocol buffers (.proto) for ONOS network model

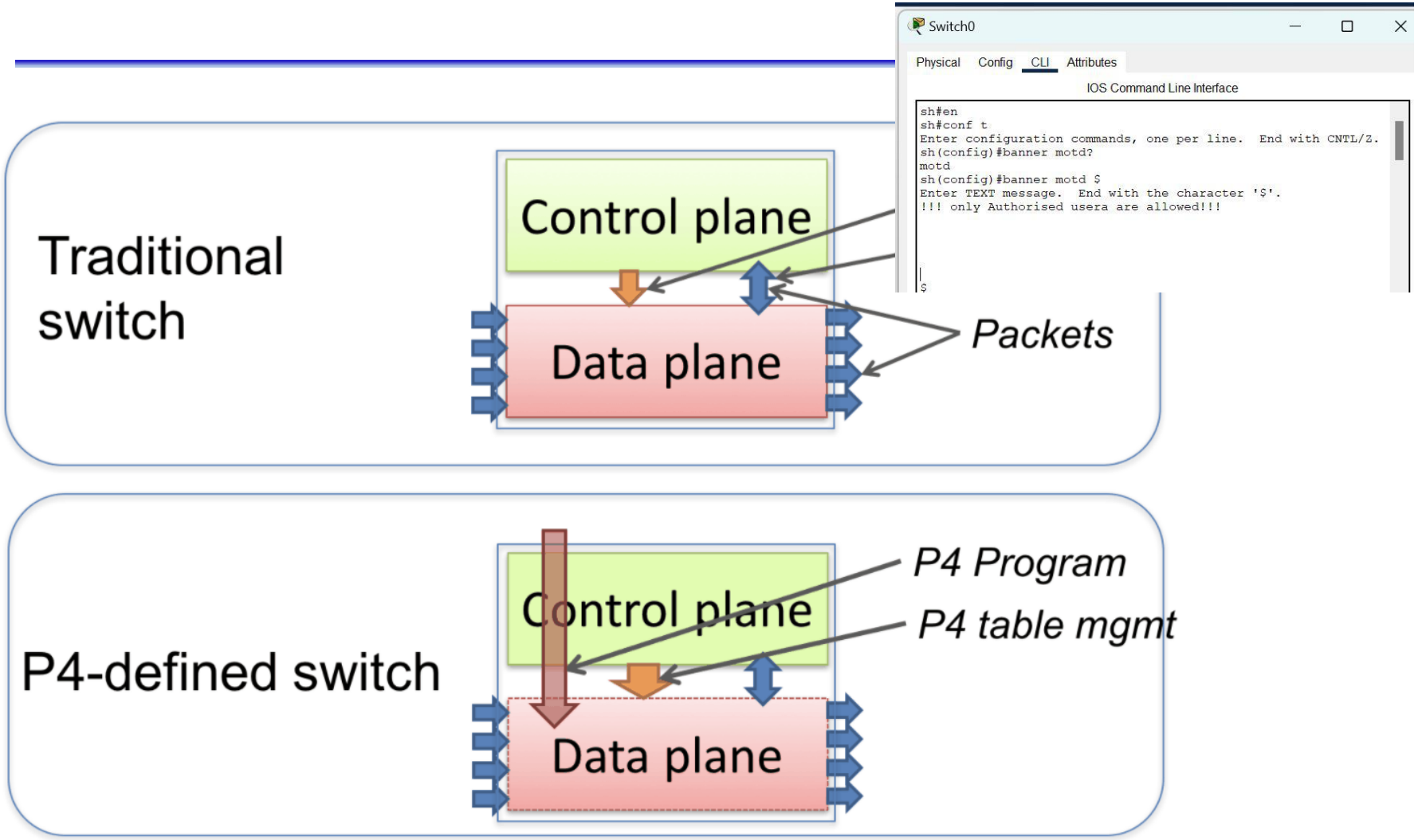
Demo


ONOS - Mozilla Firefox
Wed Jul 12 14:34:39
Sean Condon
Sean Condon - Outlo
My Reviews | gerrit.o
Google
ONOS Login
Segment Routing for
How To Use GPG to E
Making and verifying
Pica8 OVS Switch Ma
ONOS
localhost:8181/onos/ui/index.html#/topo?appId=2&appName=org.onosproject.cli&key=0x19&intentType=HostToHostIntent&overlayId=traffic
sample mininet
Open Network Operating System
onos
127.0.0.1
127.0.0.1
Devices: 6
ONOS Summary
Version : 1.11.0*
Devices : 6
Links : 12
Hosts : 4
Topology SCCs : 1
Intents : 1
Tunnels : 0
Flows : 24
of:00000000000000b3
of:00000000000000b2
of:00000000000000b4
35
11
of:00000000000000b1
of:00000000000000b0
of:00000000000000b5
unknown
10.0.0.1
10.0.0.2
unknown
Six OpenFlow switches connected to ONOS.
2 different paths to get between B1 and B4

P4

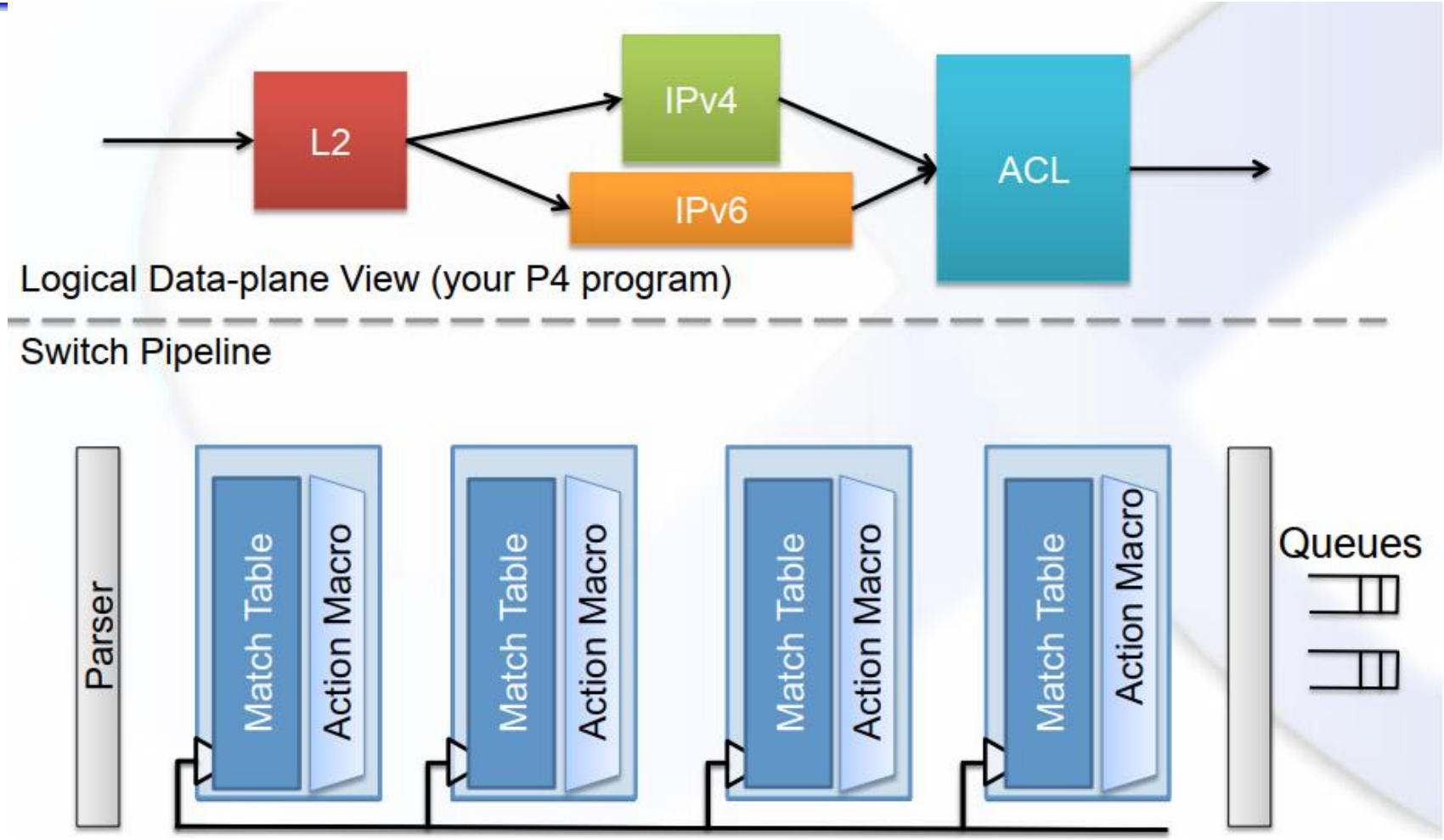
Learning to do programming for switches, instead of configuring it.

P4 vs traditional switch



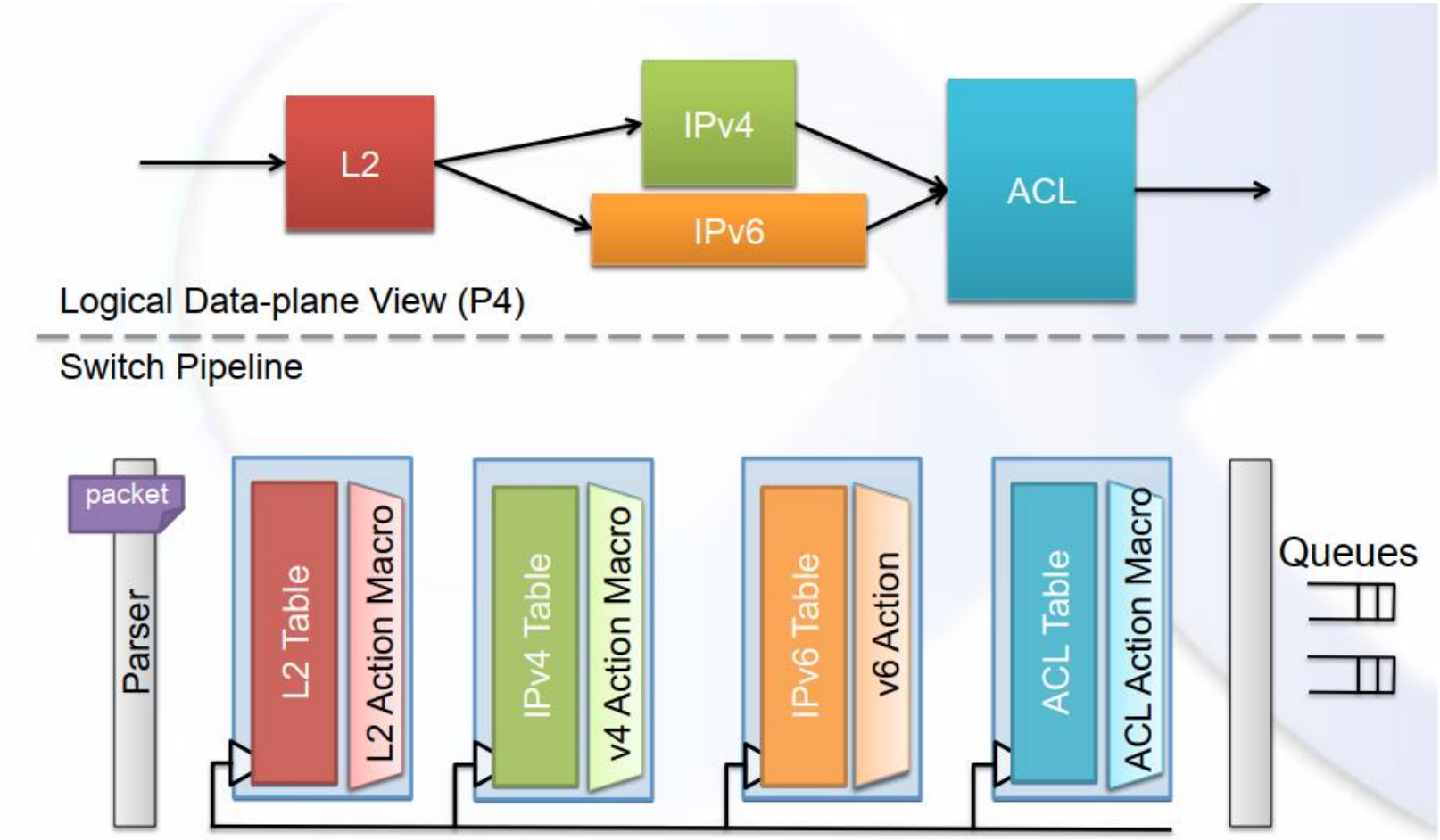
This section is recompiled from barefoot networks' slides

Protocol-Independent Switch Architecture



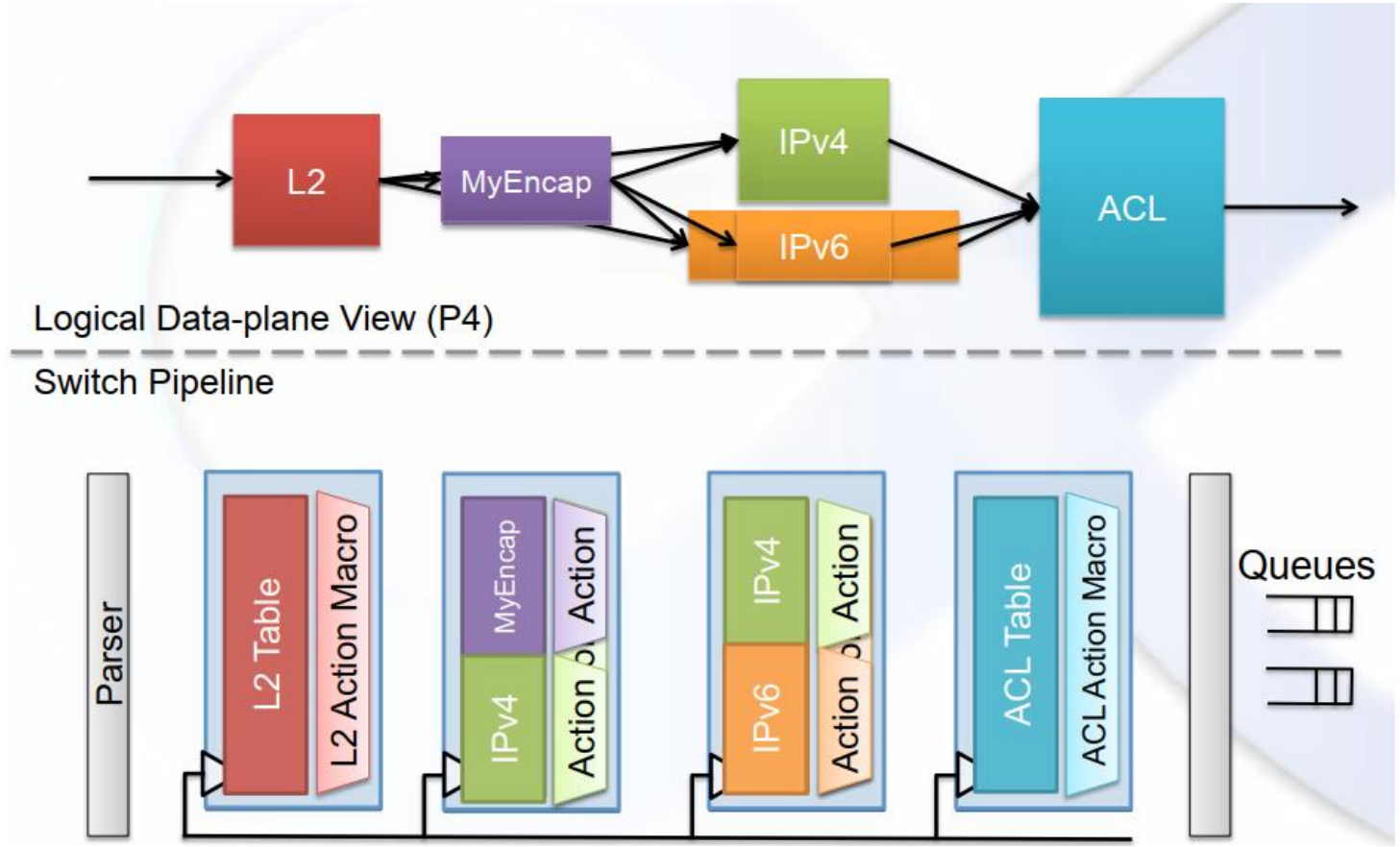
Credit to barefoot networks

Mapping to the physical resource



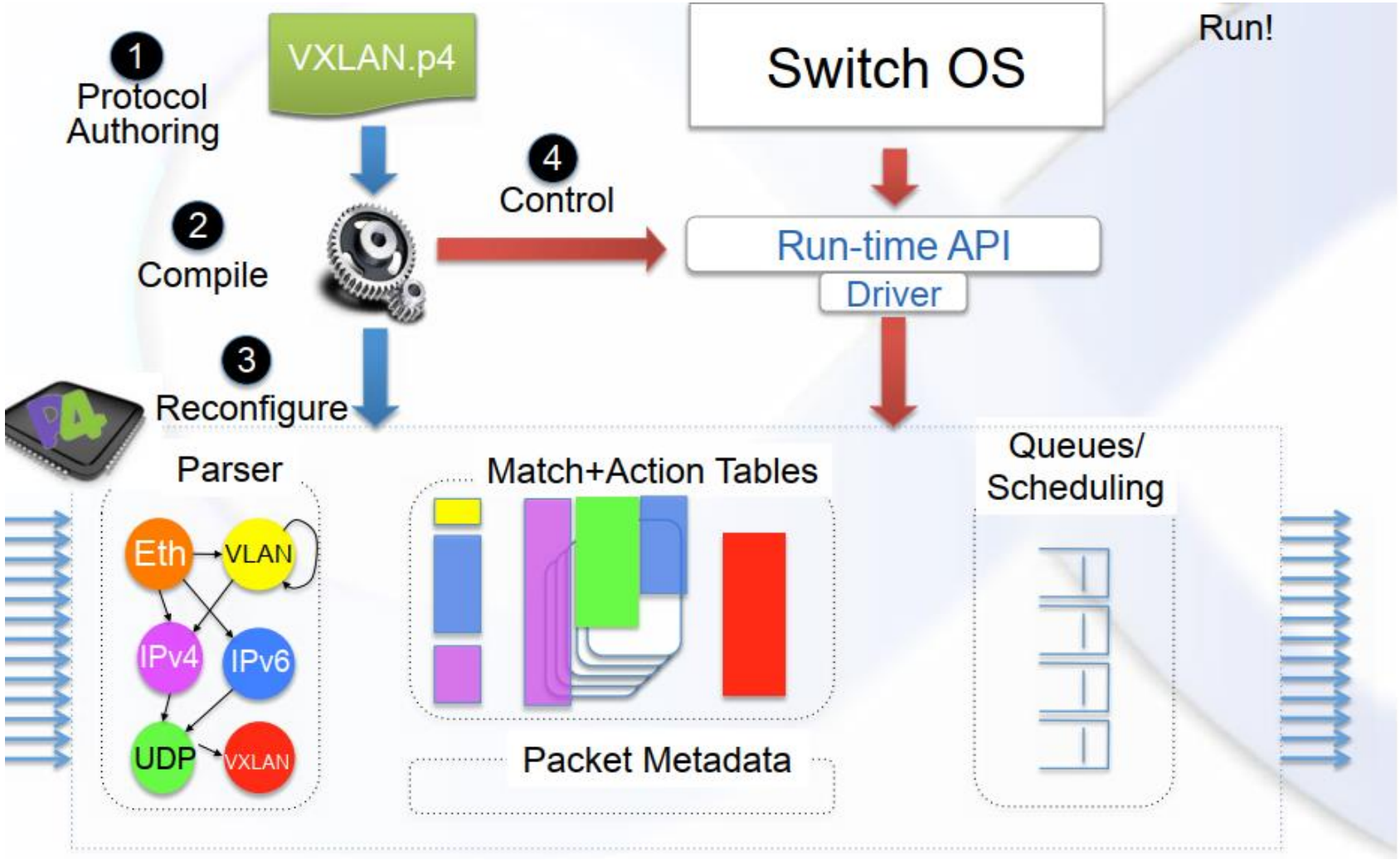
Credit to barefoot networks

Re-configurability



Credit to barefoot networks

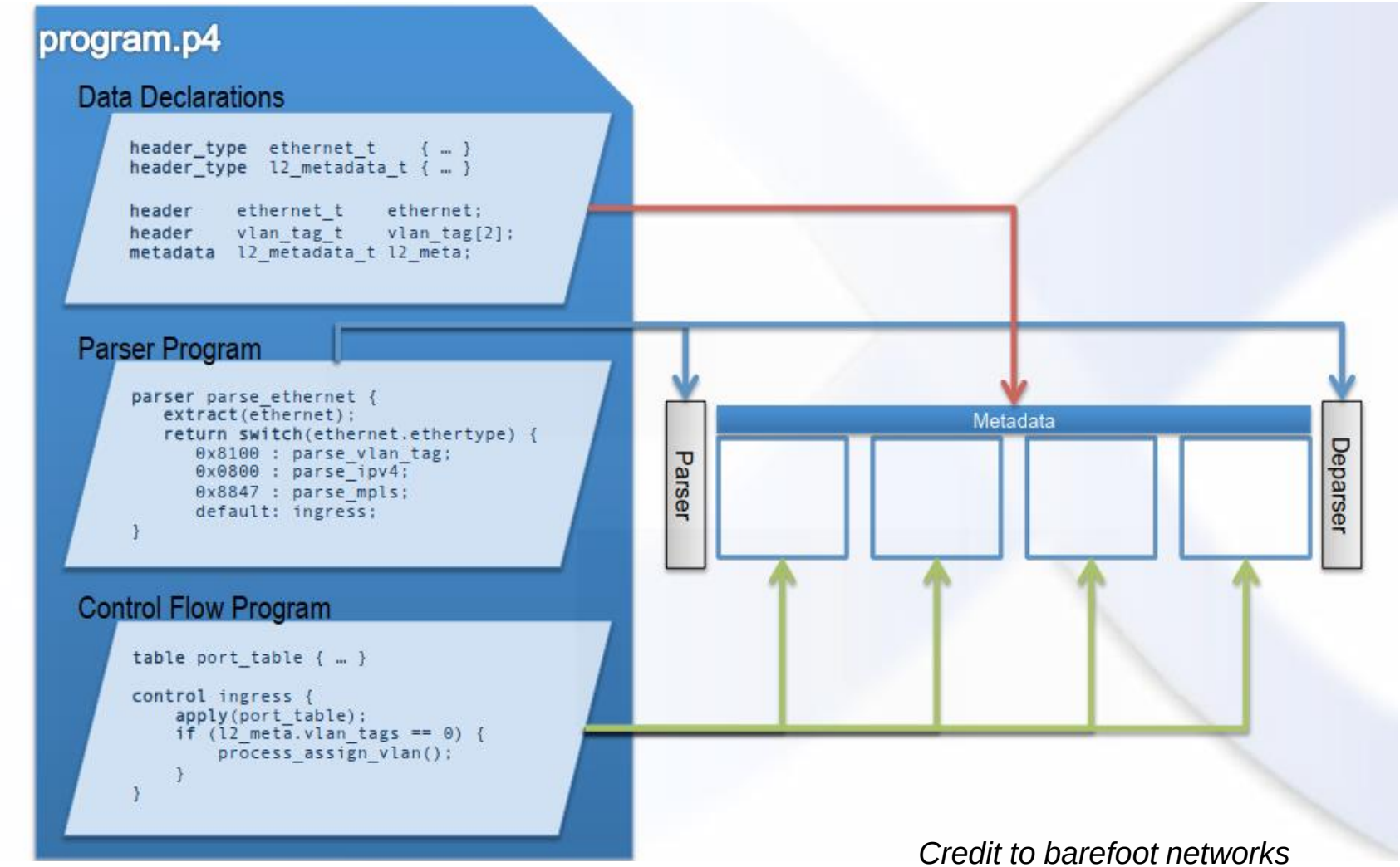
P4-based workflow



Credit to barefoot networks

P4 membership

P4 program



Credit to barefoot networks

P4 language

- **Data declarations**
  - Packet Headers and Metadata
- **Parser Programming**
  - Parser Functions (Parser states)
  - Checksum Units
- **Packet Flow Programming**
  - Actions
    - Primitive and compound actions
    - Counters, Meters, Registers
  - Tables
    - Match keys
    - Attributes
  - Control Functions (Imperative Programs)

No: pointers, loops, recursion, floating point

Credit to barefoot networks

Packet header and Metadata (1/2)

Example: Declaring packet headers

```
header_type ethernet_t {
    fields {
        dstAddr    : 48;
        srcAddr    : 48;
        etherType  : 16;
    }
}

header_type vlan_tag_t {
    fields {
        pcp        : 3;
        cfi        : 1;
        vid        : 12;
        etherType  : 16;
    }
}

header ethernet_t ethernet;
header vlan_tag_t vlan_tag[3];
```

Header Type Declarations

Actual Header Instantiation

Handy Arrays for Header Stacks

Credit to barefoot networks

Packet header and Metadata (2/2)

Example: Declaring Metadata

```
header_type ingress_metadata_t {
    fields {
     /* Inputs */
        ingress_port          : 9;  /* Available prior to parsing    */
        packet_length         : 16; /* Might not be always available */
        instance_type         : 2;  /* Normal, clone, recirculated   */
        ingress_global_tstamp : 48;
        parser_status         : 8;  /* Parsing Error */

     /* Outputs from Ingress Pipeline */
        egress_spec           : 16;
        queue_id              : 9;
    }
}

metadata ingress_metadata_t ingress_metadata;
```

Metadata is a header too

Actual Metadata Instantiations

Credit to barefoot networks

Variable-Length Fields

Example: Declaring IPv4 packet header

```
header_type ipv4_t {
    fields {
        version        : 4;
        ihl            : 4;
        diffserv       : 8;
        totalLen       : 16;
        identification : 16;
        flags          : 3;
        fragOffset     : 13;
        ttl            : 8;
        protocol       : 8;
        hdrChecksum    : 16;
        srcAddr        : 32;
        dstAddr        : 32;
        options        : *;
    }
    length     : (ihl << 2);
    max_length : 60;
}
```

Variable-length Field

Calculated, based on another field

Credit to barefoot networks

Defining a Parser Tree

Example: Simple Parser for L2/L3 Packets

```
header ethernet_t ethernet;
header vlan_tag_t vlan_tag[2];
header ipv4_t ipv4;
header ipv6_t ipv6;

parser start {
    extract(ethernet);
    return select(latest.etherType) {
        0x8100, 0x9100 : parse_vlan_tag;
        0x0800         : parse_ipv4;
        0x86DD         : parse_ipv6;
        default        : ingress;
    }
}
parser parse_vlan_tag {
    extract(vlan_tag[next]);
    return select(latest.etherType) {
        0x8100 mask 0xEFFF : parse_vlan_tag;
        0x0800             : parse_ipv4;
        0x86DD             : parse_ipv6;
        default            : ingress;
    }
}
```

Transitions to the next parser states

Depending on the state, it can be:
- ethernet.ethertype
- vlan_tag[0].ethertype
- vlan_tag[1].ethertype

The loop is bounded by the number of elements in vlan_tag[] array

This is not a reserved word, but a name of the Control Flow Function

Credit to barefoot networks

Defining a Parser Tree

Example: Simple Parser for L2/L3 Packets

```
header ethernet_t ethernet;
header vlan_tag_t vlan_tag[2];
header ipv4_t ipv4;
header ipv6_t ipv6;

parser start {
    extract(ethernet);
    return select(latest.etherType) {
        0x8100, 0x9100 : parse_vlan_tag;
        0x0800         : parse_ipv4;
        0x86DD         : parse_ipv6;
        default        : ingress;
    }
}
parser parse_vlan_tag {
    extract(vlan_tag[next]);
    return select(latest.etherType) {
        0x8100 mask 0xEFFF : parse_vlan_tag;
        0x0800             : parse_ipv4;
        0x86DD             : parse_ipv6;
        default            : ingress;
    }
}
```

```
parser parse_ipv4 {
    extract(ipv4);
    return ingress;
}

parser parse_ipv6 {
    extract(ipv6);
    return ingress;
}
```

Credit to barefoot networks

Using Calculated Fields

Example: Calculated fields for IPv4

```
field_list ipv4_checksum_list {
        ipv4.version;
        ipv4.ihl;
        ipv4.diffserv;
        ipv4.totalLen;
        ipv4.identification;
        ipv4.flags;
        ipv4.fragOffset;
        ipv4.ttl;
        ipv4.protocol;
        ipv4.srcAddr;
        ipv4.dstAddr;
}
field_list_calculation ipv4_checksum {
    input        { ipv4_checksum_list; }
    algorithm    : csum16;
    output_width : 16;
}

calculated_field ipv4.hdrChecksum  {
    verify ipv4_checksum;
    update ipv4_checksum;
}
```

```
parser parse_ipv4 {
    extract(ipv4);
    return ingress;
}
```

Credit to barefoot networks

Multi-field select statement

Example: Ipv4 Header Parsing

```
parser parse_ipv4 {
    extract(ipv4);
    set_metadata(ipv4_metadata.lkp_ipv4_sa, ipv4.srcAddr);
    set_metadata(ipv4_metadata.lkp_ipv4_da, ipv4.dstAddr);
    set_metadata(l3_metadata.lkp_ip_proto, ipv4.protocol);
    set_metadata(l3_metadata.lkp_ip_ttl, ipv4.ttl);

    return select(latest.fragOffset, latest.ihl, latest.protocol) {
        0x0000_5_01 : parse_icmp;
        0x0000_5_06 : parse_tcp;
        0x0000_5_11 : parse_udp;
        default     : ingress;
}
```

Metadata can be set from the parser

Fields are joined for a match

"_" are ignored in numerical constants

Credit to barefoot networks

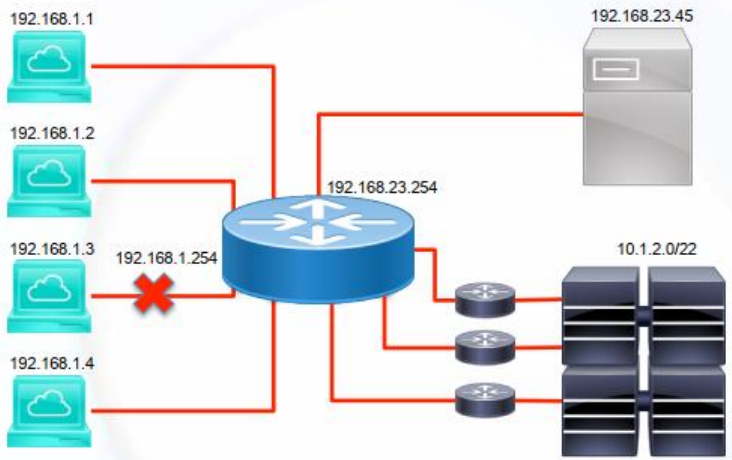
Deparsing (Serializing packet headers)

- **Primitive actions**
  - no_op, drop
  - modify_field, modify_field_with_hash_based_index
  - add_header, remove_header, copy_header
  - push/pop (a header)
  - count, execute_meter
  - generate_digest, truncate
  - resubmit, recirculate, clone{_i2i, _e2i, _i2e, _e2e}
- **Compound actions**

```
action route_ipv4(dst_port, dst_mac, src_mac, vid) {
    modify_field(standard_metadata.egress_spec, dst_port);
    modify_field(ethernet.dst_addr, dst_mac);
    modify_field(ethernet.src_addr, src_mac);
    modify_field(vlan_tag.vid, vid);
    modify_field(ipv4.ttl, ipv4.ttl - 1);
}
```

Credit to barefoot networks

Example

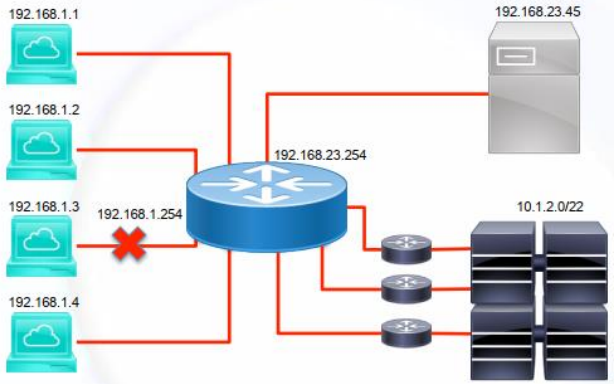


Key	Action	Action Data
192.168.1.1	l3_switch	port= mac_da= mac_sa= vlan=
192.168.1.2	l3_switch	port= mac_da= mac_sa= vlan=...
192.168.1.3	l3_drop	
192.168.1.254	l3_l2_switch	port=
192.168.1.0/24	l3_l2_switch	port=
10.1.2.0/22	l3_switch_ecmp	ecmp_group=

```
action l3_switch(port, mac_da, mac_sa, vlan) {
    modify_field(metadata.egress_spec, port);
    modify_field(ethernet.dstAddr, mac_da);
    modify_field(ethernet.srcAddr, mac_sa);
    modify_field(vlan_tag[0].vlanid, vlan);
    modify_field(ipv4.ttl, ipv4.ttl - 1);
}

action l3_l2_switch(port) {
    modify_field(metadata.egress_spec, port);
}

action l3_drop() {
    drop();
}

action l3_switch_nexthop(nexthop_index) {
    modify_field(l3_metadata.nexthop, nexthop_index);
    modify_field(l3_metadata.nexthop_type, NEXTHOP_TYPE_SIMPLE);
}
action l3_switch_ecmp(ecmp_group) {
    modify_field(l3_metadata.nexthop, ecmp_group);
    modify_field(l3_metadata.nexthop_type, NEXTHOP_TYPE_ECMP);
}
```

Credit to barefoot networks

Match-Action Table (Exact Match)



Key	Action	Action Data
192.168.1.1	l3_switch	port= mac_da= mac_sa= vlan=
192.168.1.2	l3_switch	port= mac_da= mac_sa= vlan=...
192.168.1.3	l3_drop	
192.168.1.254	l3_l2_switch	port=
192.168.1.0/24	l3_l2_switch	port=
10.1.2.0/22	l3_switch_ecmp	ecmp_group=

Example: A typical L3 (IPv4) Host table

```
table ipv4_host {
    reads {
        ingress_metadata.vrf    : exact;
        ipv4.dstAddr            : exact;
    }
    actions {
        nop;
        l3_switch;
        l3_l2_switch;
        l3_switch_nexthop;
        l3_switch_ecmp;
        l3_drop;
    }
    size : HOST_TABLE_SIZE;
}
```

These are the only possible actions. Each particular entry can have only ONE of them.

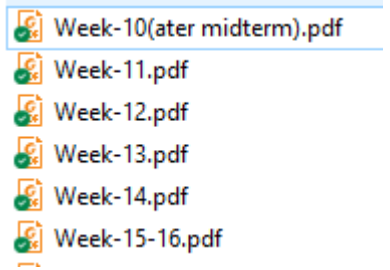
vrf	ipv4.dstAddr	action	data
1	192.168.1.10	l3_switch	port_id= mac_da= mac_sa=
100	192.168.1.10	l3_l2_switch	port_id=<CPU>
1	192.168.1.3	l3_drop	
5	10.10.1.1	l3_switch_ecmp	ecmp_group=127

Credit to barefoot networks

Summary

- Software and Hardware are independence
- SDN, ONOS, P4 is the current state-of-the-art networking technology → focus on software-based → faster innovation
- Help to relax the dependence on a single vendor (Cisco, Juniper…)

Important notices



- The final exam scope: From mid-term exam to today
- Allowed to bring to the exam: Contents/Writings in 4 A4
- Not allowed to bring to the exam: Any electric device

- Dec 28: Report (certificate/paper) for bonus points in Ecourse2
- Dec 28 lesson: SD-WAN/Fabric/Demo lab + Answer all questions