



Programación de Redes – Becas Digitaliza - 2019

PUE – ITC – Formación de Instructores

Sesión 6 – Contextualización y SDN

Iván Lago - Técnico Cisco Networking Academy ASC/ITC
PUE - ITC/ASC/CA

- Networking evolution
- Why SDN?
- What is OpenFlow?
- Network Abstraction and Virtualization
- Network Management
- Devnet: Introduction to APIC-EM

NETWORK EVOLUTION

Cisco's evolution



Evolution

NetAcad courses



Networking



Cybersecurity



IoT



Hardware, Linux...

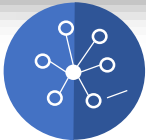


Programming



Entrepreneurship

Why are we here?



Everything
becomes
connected

Networking



Everything
becomes
software-based

Programmability



Everything
generates
data



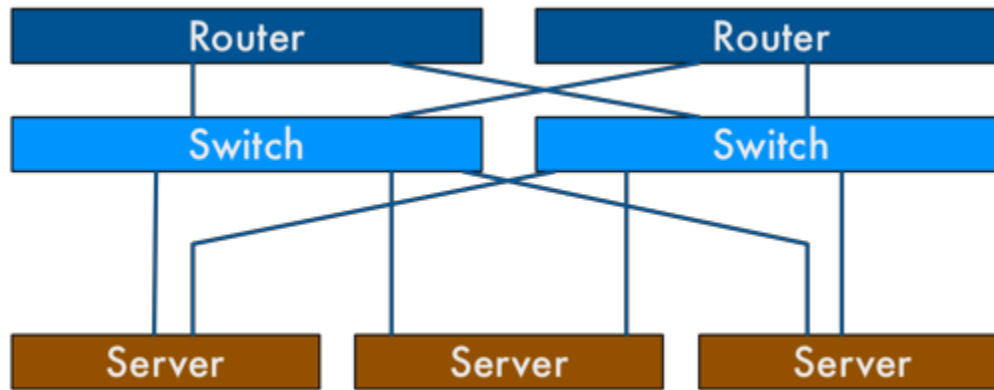
Everything
can be
automated



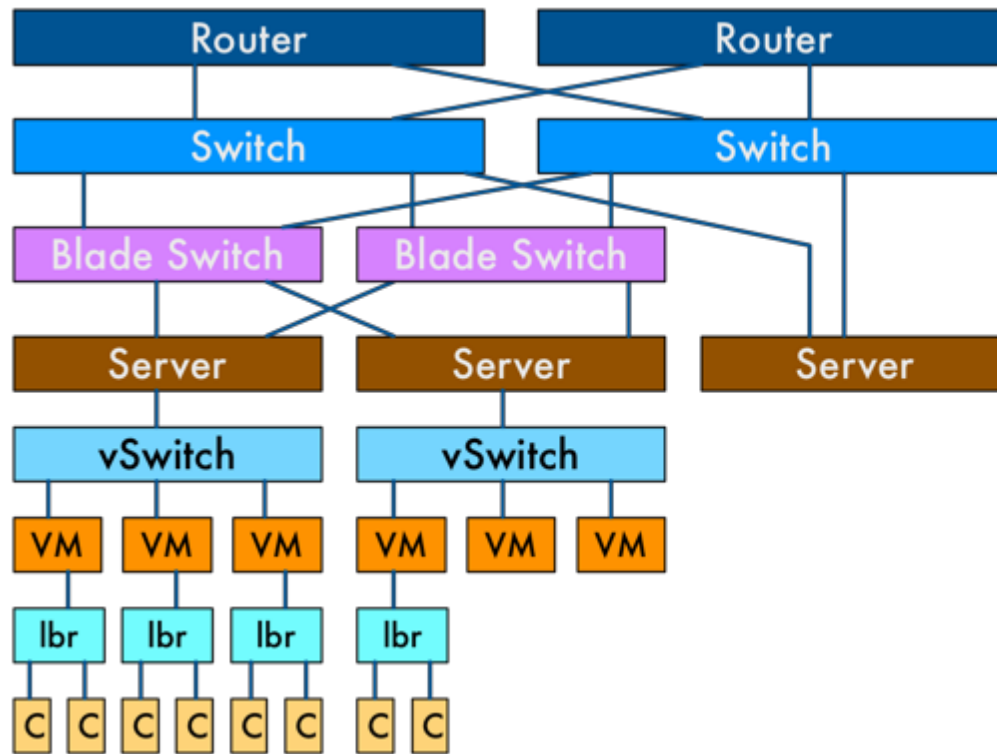
Everything
needs to be
secured

Security

Network: in the past



Network: nowadays



WHY SDN?

Problem I: Network Management

Network management



There are a lot of complex high-level network policies. Along the past years, there were proposals to reduce the complexity of the management, but they did not succeed because many of them implied an underlying infrastructure change.

Problem II: Network Instability



Network instability

Since its conditions are continually changing, operators must manually adjust network configurations in response to those changes. In addition, many times, administrators have to use external tools or building scripts to monitor some parameters and react accordingly when a change is produced.

Solution: SDN – Original Definition



SDN was a physical separation of the control plane from the data (packet forwarding) plane, and the decoupled control plane must control several devices.

SDN is so valuable, because the controller's general view about the network allows the server to adjust the affected configurations, as well as helping to a fast re-convergence.

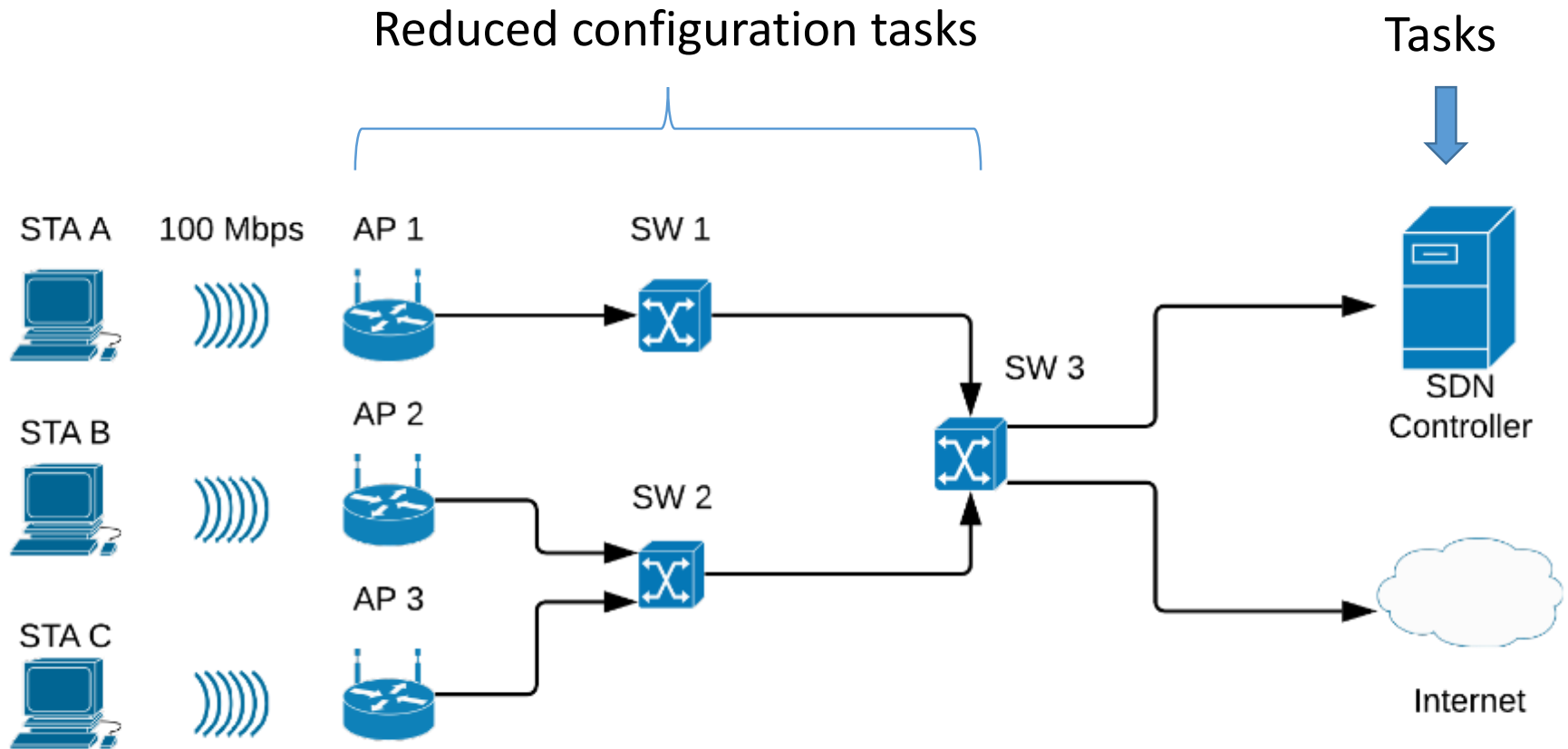
SDN: Problem Solver

Software-Defined Networks appeared in this market niche with the main idea of separating the control plane from the data plane.

Since the major part of the network configurations are done in a controller, this concept reduces the repetitive configuration process of each intermediate device, which is configured through a command line interface (CLI).

In addition, many of the implementations do not have to change anything in the client side, just to implement the OpenFlow protocol in the intermediate devices.

SDN: main concept



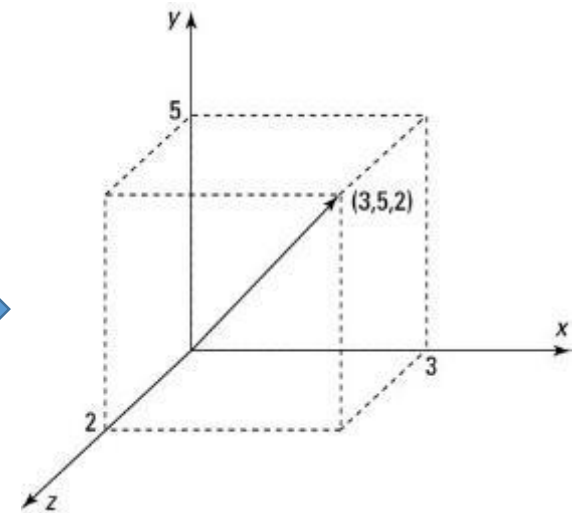
SDN is everywhere



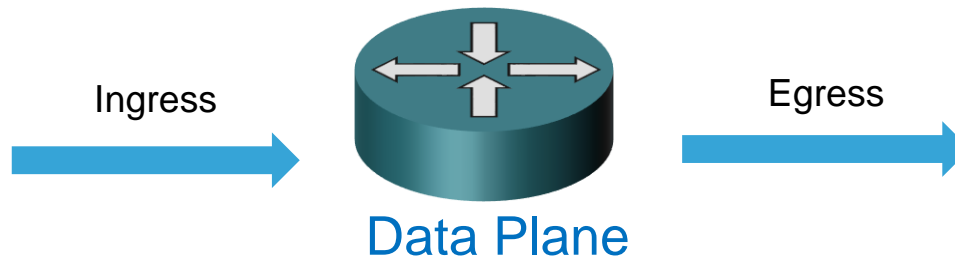
SDN: Objects vs planes



Object vs planes



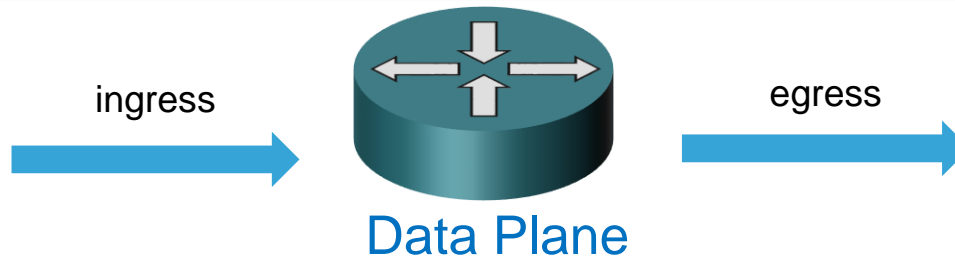
Just single devices?



SDN: Control and Data Planes (I)

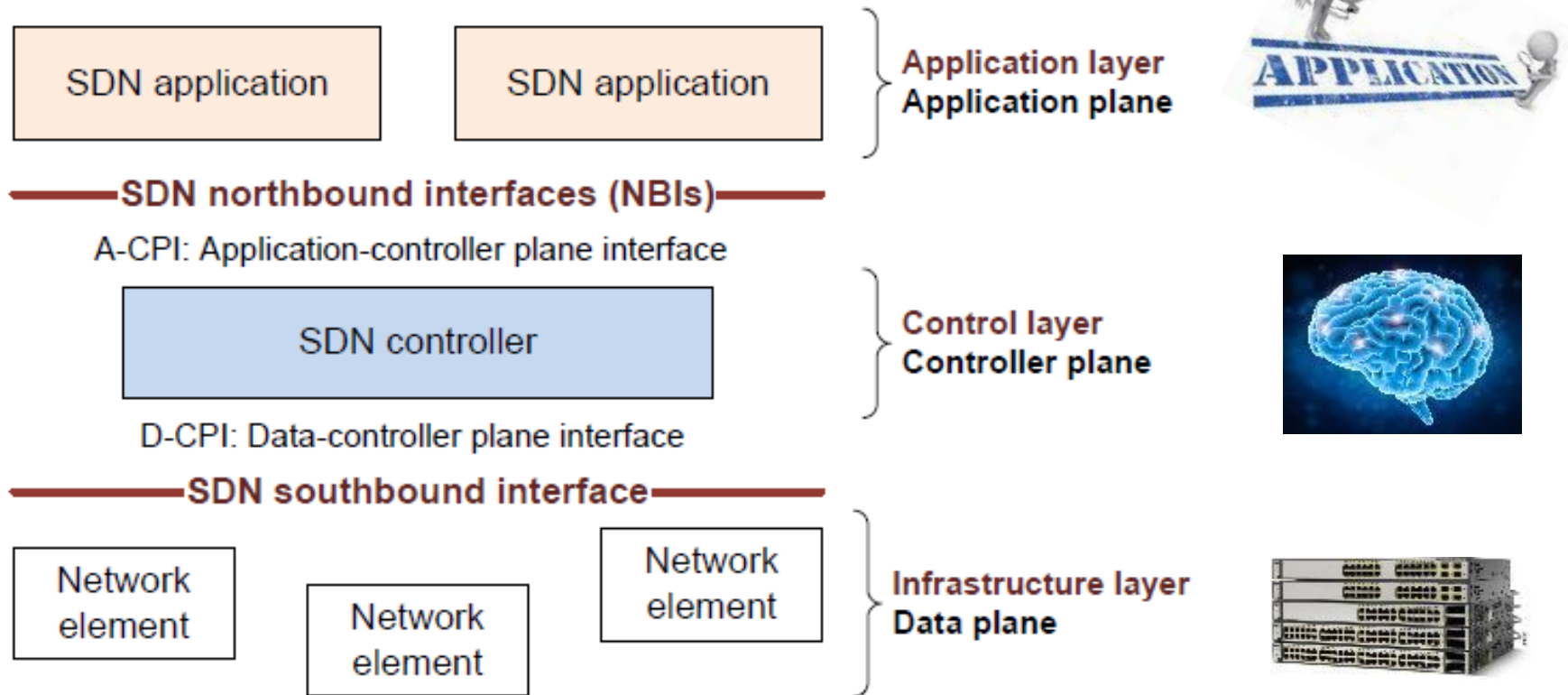
Control Plane

Hardware	Purpose	Example Processes
Device CPU	makes decisions about where traffic is sent	routing protocols, spanning tree, AAA, SNMP, CLI



Hardware	Purpose	Example Processes
Dedicated ASICs	forwards traffic to the selected destination	packet switching, L2 switching, QoS, policies, ACLs

SDN: Control and Data Planes (II)



SDN: Layers

- **Application layer (application plane):** applications with exclusive resources by one or more controllers. Applications communicate their network requirements toward the controller plane via northbound interfaces (NBIs). Finally, users can communicate with SDN services through the northbound API, such as REST, JSON or XML.
- **Control layer (controller plane):** it is the brain of this architecture, because this layer supervises the network behavior through an open interface and it has the power to manage the network configuration. In addition, the set of SDN controllers execute the requests from the applications and it give the statistics and events to the application layer.
- **Infrastructure layer (data plane):** it includes the network elements and talks with the controller through the southbound interface. The function of this layer includes traffic forwarding and processing functions, making easy the dynamic changes in the network. Nevertheless, these intermediate devices do not compute the routes, but they ask the controller about them.

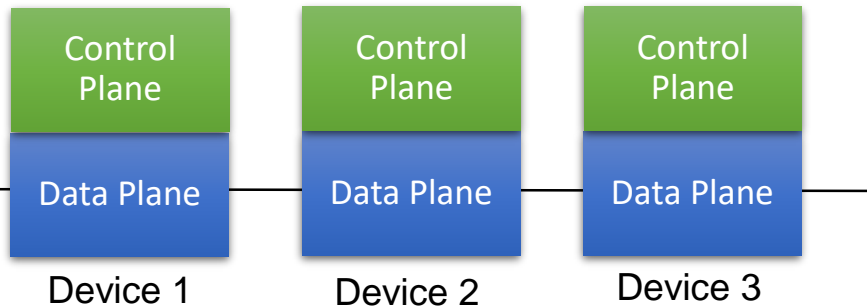
SDN: Interfaces

- **Northbound interfaces (NBIs):** it determines how to express tasks and network policies and how to translate them into a form the controller can understand.
- **Southbound interface (SBI):** it is the interface between data plane and the control layer, and it manages the protocol between programmable switches and the software controller. This interface is the core point to set up the communication between them.

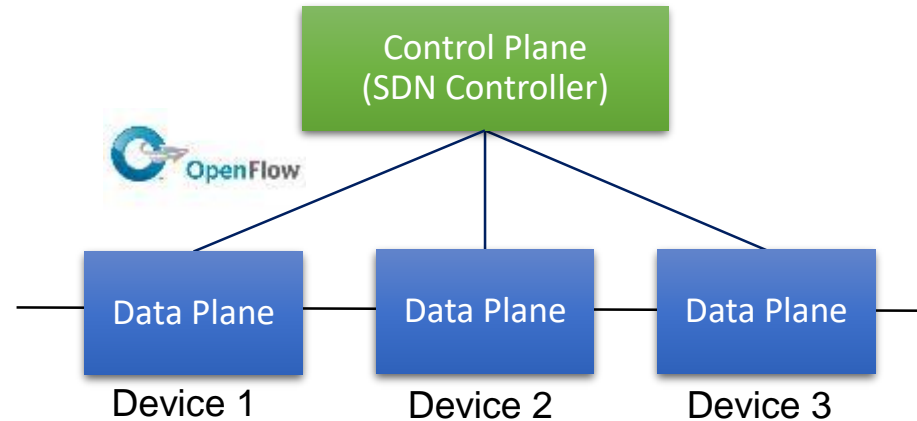


Traditional and SDN Architectures (I)

Traditional Architecture



SDN Architecture with Centralized Control Plane



OpenFlow is a protocol between SDN controllers and network devices, as well as a specification of the logical structure of the network switch functions.

Traditional and SDN Architectures (II)

Network elements belong to the data plane, which have to forward traffic, among other tasks.

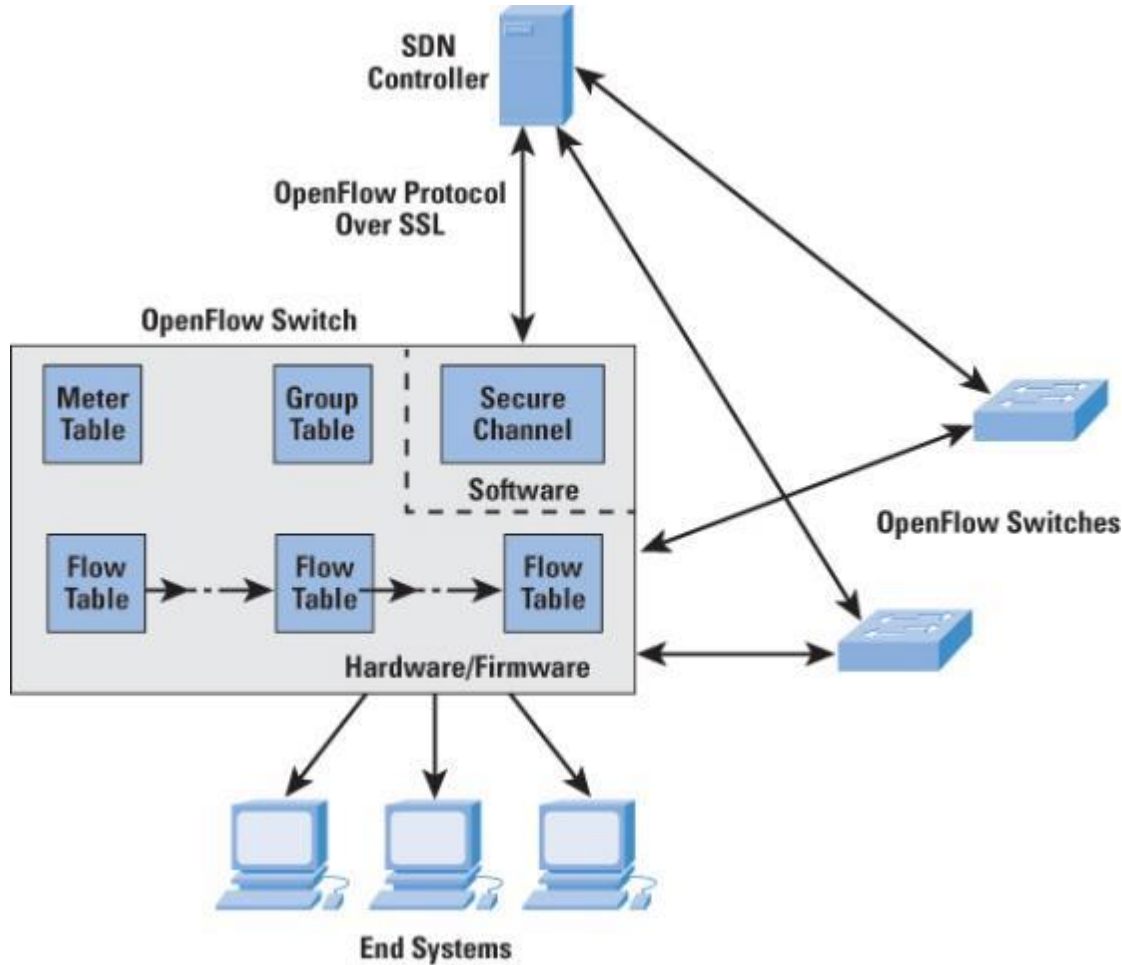
Whereas in classic networks both routers and switches carry on with the computation of the routes to forward the traffic, SDN paradigm moves this task to the controller.

Here, forwarding will be done depending on the instructions provided by the server, commonly known as flow entries, which are stored in a flow entry table in the Switches' RAM.

Obviously, not any switch can be used, because it has to be able to implement the OpenFlow protocol.

WHAT IS OPENFLOW?

OpenFlow protocol



OpenFlow is the current SDN protocol standard to control switches, establish the management communications and deciding what parameters must be considered.

Openflow Tables

Match Fields	Counter	Instructions	Timer
--------------	---------	--------------	-------

- **Match Fields:** parameters to compare the entry with the incoming flow in order to know if the rule can be applied to it.
- **Counter:** number of times that a rule is matched with a flow.
- **Instructions:** information about how to treat the packet or which path is the one to forward it.
- **Timer:** entries can be set permanently but, by default, they are stored 60 seconds in memory [14]. This field indicates the current remaining time to delete the entry. Each time that a match is produced, the timer is reseted.

Openflow: how it works? (I)

Switches seek the headers of the incoming packets and verify the match fields of their flow entries for a possible match.

If the match is produced, the counter of the flow entry increases, and the actions indicated in the instructions field are applied to that flow in order to forward or discard it.

The counter is used to debug and knowing how many matches with entry flows are produced. In addition, since the entry flow has a timer, every time that the counter increases, the timer is reseted.

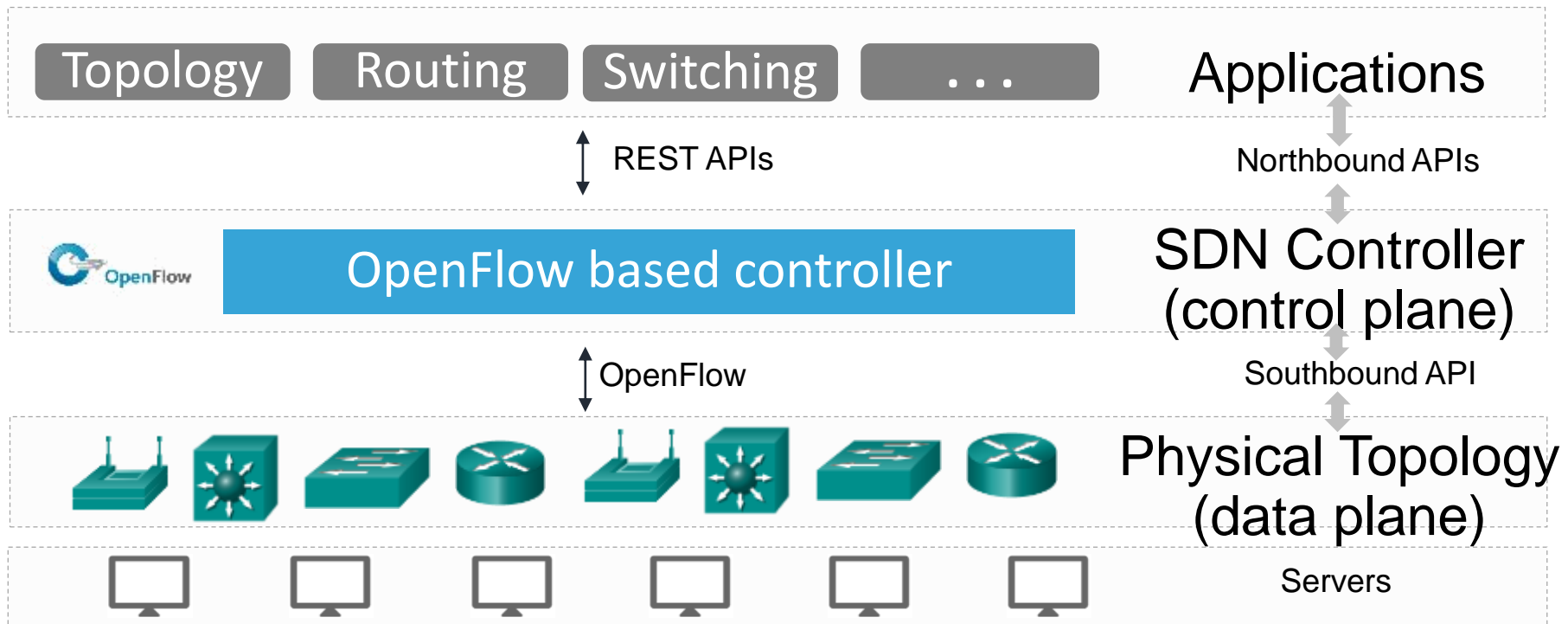
Openflow: how it works? (II)

If there is no match for the incoming packet, a management request has to be sent to the controller, and the answered rule is stored in memory.

Communications between the OpenFlow Switch and the controller is through a secure channel implementing SSL or TLS.

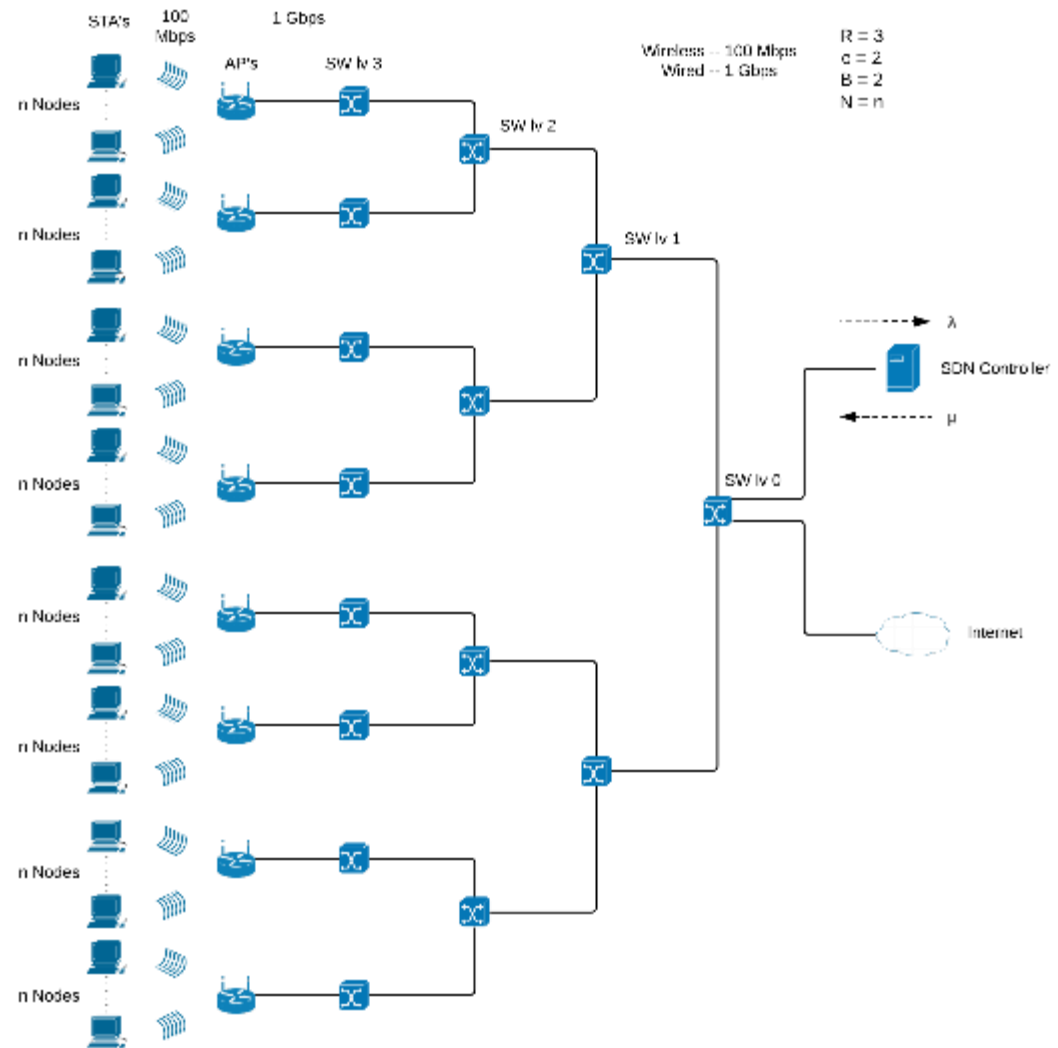
In addition, it can be seen a group table, which collects many entries to apply to a flow, and a meter table to enable the implementation of simple QoS operations, such as rate-limiting.

OpenFlow SDN Model



Bottleneck?

While there are some use cases for using an OpenFlow-based centralized control plane (Data Center), with the growth of the network infrastructure and the number of connected devices, a fully centralized control plane might not scale to the desired levels with consistent performance.



SDN: more than planes decoupling

Nowadays, many more technologies get put under the SDN umbrella, including controller-based networks, APIs on network devices, network automation, whitebox switches, policy networking, Network Functions Virtualization (NFV), and the list goes on.

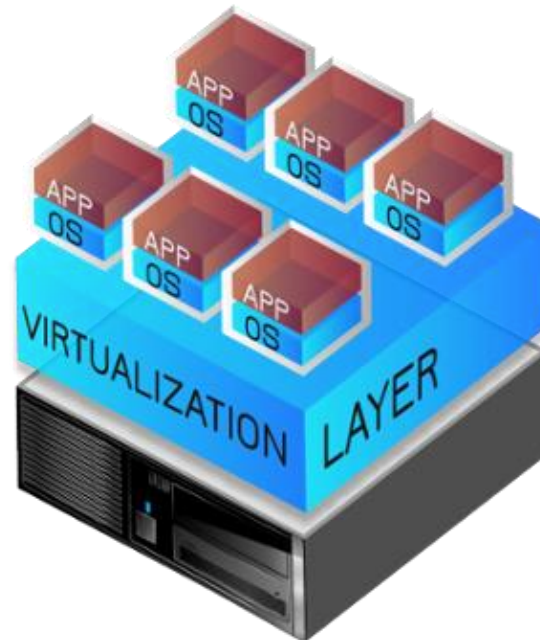
Therefore, the trend is referring to SDN solutions as solutions that include a network controller as part of the solution, and improve manageability of the network but don't necessarily decouple the control plane from the data plane.

NETWORK ABSTRACTION AND VIRTUALIZATION

Traditional vs Virtualized Server



Traditional Server Architecture



Virtualized Server Architecture

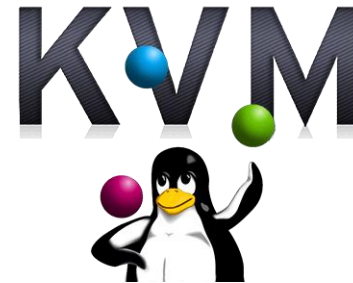
Virtualization



Oracle
Virtualbox



VMware
Workstation



KVM



VMware vSphere: ESXi + vCenter

Network and Host Virtualization

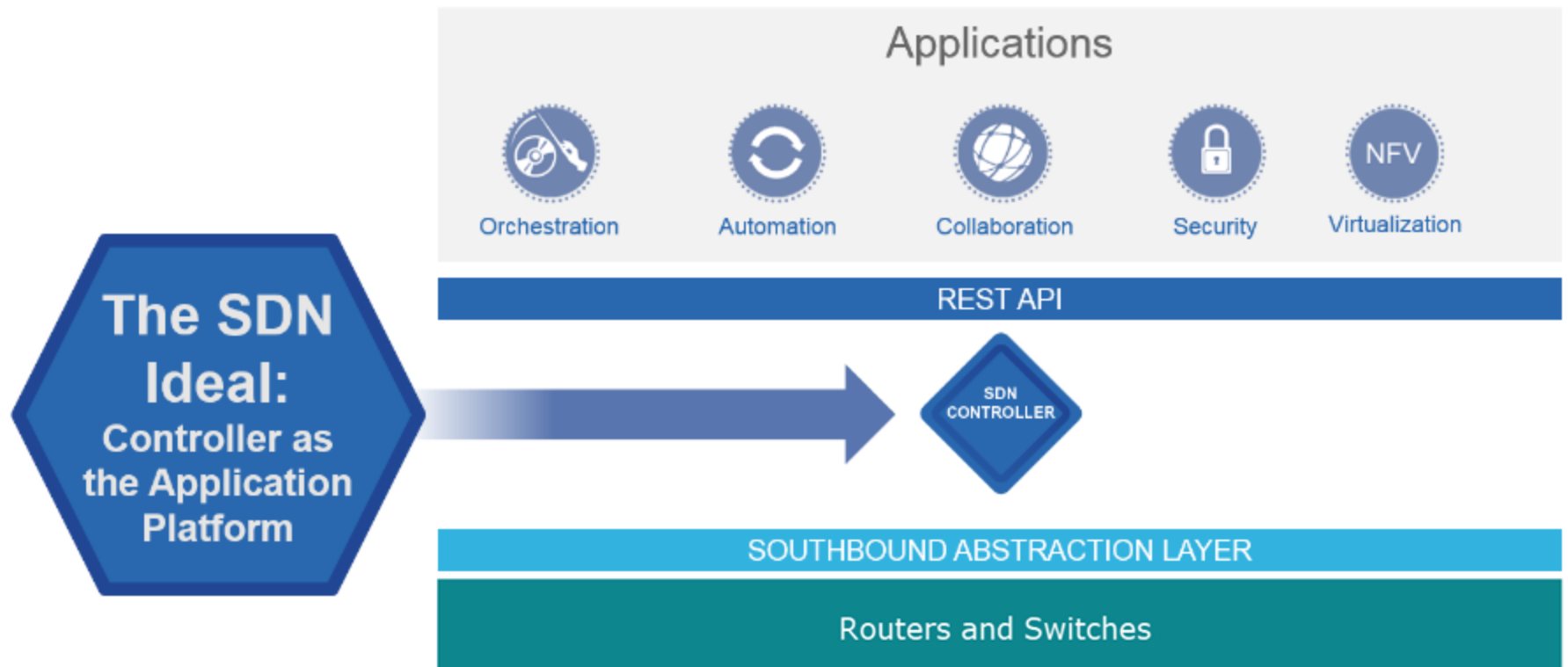
Virtualization network's technology increases the quality of software defined networks by decoupled the network resources of the underlying hardware.

Something similar to server virtualization, which simulates a physical server inside the software, the network virtualization simulates the components of the network's services and its security through the software.

In this way, the virtualized network implements and manage the hardware independently. Physical devices are just vehicles to forward packets.

Network abstraction (I)

Network-Wide Abstractions Simplify the Network



Network abstraction (II)

Controllers create abstraction from physical network devices in the application layer.

In this case, the control plane does not need to be decoupled from the data plane. The control plane and the data Plane resides in every device, which provides providing local distributed intelligence where it is needed.

This approach removes the potential for the controller to become the bottleneck of the network. This approach supports greater resiliency, availability, and scalability, by taking advantage of leveraging the existing intelligence of out to the devices, using established networking protocols. Existing network devices such as routers and switches continue to be used to build scalable and high performance distributed networks.

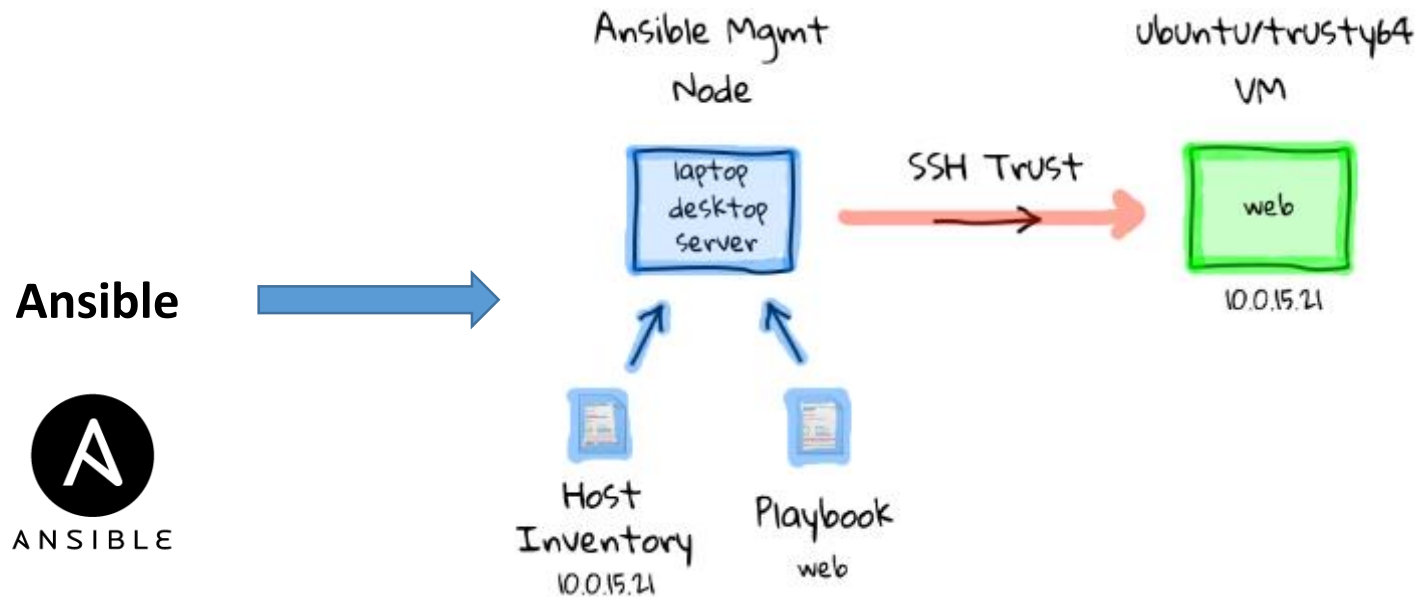
Network abstraction (III)

In the past, network staff had to spend a lot of time on the infrastructure -- switches and routers -- connecting them to build a network architecture. Where people really need to spend their time – trend of digital business – is the application layer.

The SDN controller presents the **application layer with abstractions of the network** that is **that's underneath**. Because of this, applications can easily consume network services, which has never been possible in the past. This is critical because the digital enterprise is entirely application driven. These software applications control orchestration, automation, collaboration, policy, and security so that the network staff can work with the abstract and use representation of the network fabric.

NETWORK MANAGEMENT

Manual vs Ansible management



Host Inventory and Playbook

Playbook

Host Inventory

```
[loadbalancer]
haproxy-01.example.com
haproxy-02
10.0.15.15
```

```
[web]
web1
web2
10.0.15.21
```

```
[testing]
demo.example.com
```

```
---
- hosts: web
  sudo: yes

  tasks:

    - name: install nginx
      apt: name=nginx state=installed update_cache=yes

    - name: write our nginx.conf
      template: src=templates/nginx.conf.j2 dest=/etc/nginx/nginx.conf
      notify: restart nginx

    - name: deploy website content
      git: repo=https://github.com/jweissig/episode-47.git
          dest=/usr/share/nginx/html/
          version=release-0.01

    - name: start ntp
      service: name=nginx state=started

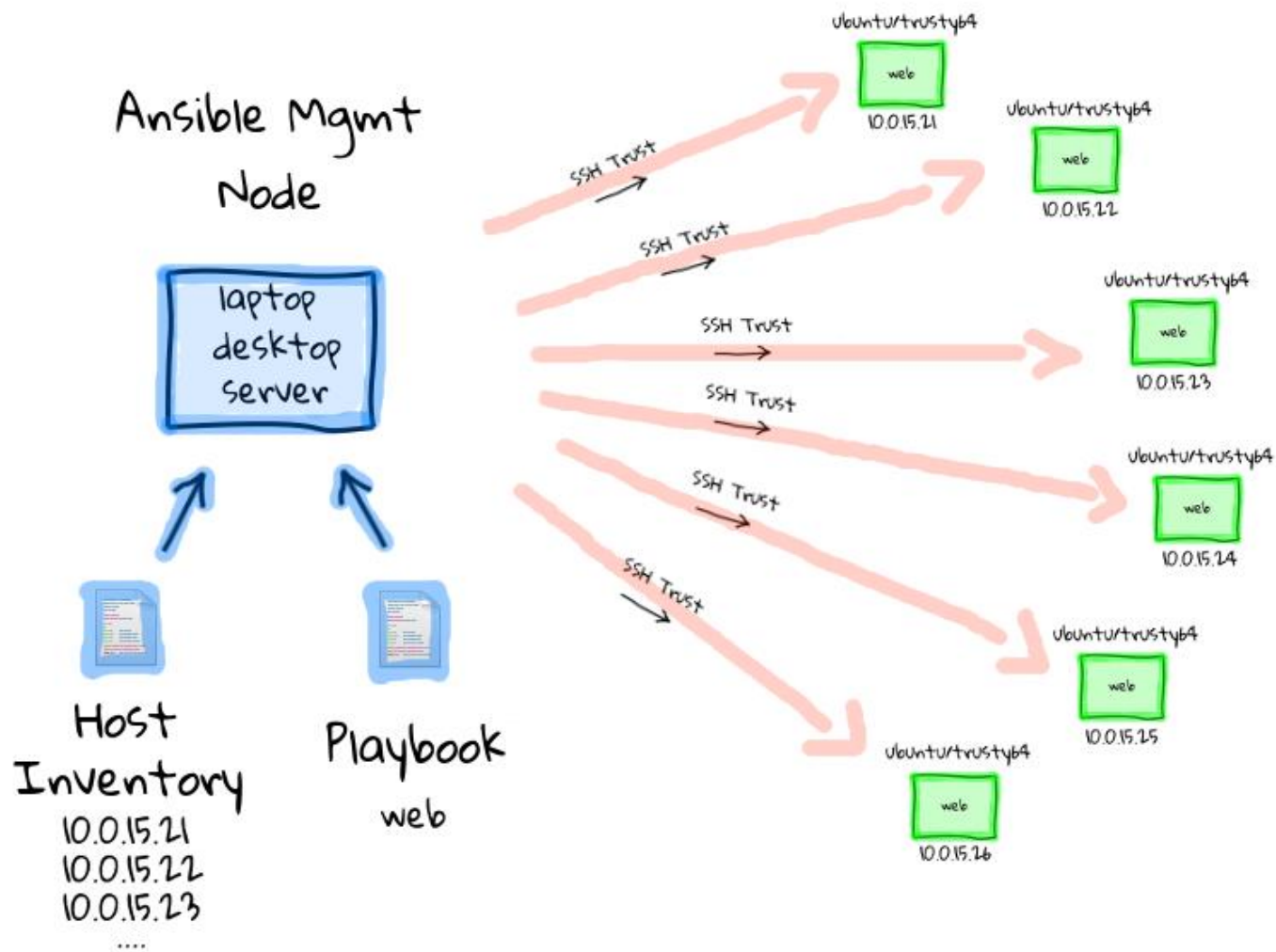
  handlers:

    - name: restart nginx
      service: name=nginx state=restarted
```


Ansible: the basics

- Free and open-source tool mainly used on UNIX-like machines.
- We need install Ansible onto a management node, define a hosts inventory (can also be a database), and have some playbooks defined.
- Ansible reads from inventory, connects through ssh to the hosts and applies the tasks of the playbook onto them.

Ansible: easy to manage multiple nodes



Why Ansible? (I)

- Ansible pushes the configurations through ssh.
- Only requires installing software in the management node. The remote machines just need python (every major distribution comes with it by default). It is only needed to install some package if we use special modules. Therefore, updates only over one machine.
- Fantastic documentation, helper modules and functions to construct smart playbooks, install ssh keys, permissions, deploy code, install packages, interact with cloud, modifying load balancer...

Why Ansible? (II)

- Playbooks are configuration files, so we don't need to be management experts, because we can quickly read and understand existing playbooks (reason why Ansible is becoming so popular).
- You do not need to give the root of the machines. Since the connection is over ssh, it can be used with a user which have some permissions.
- Ansible Best Practices guide to advice about how to layout files, using version control, naming...

Environment deployment, services and functionalities

Manually



- Virtual Machine environment
- Customized machines
- Portable and replicable environment
- Microservice heavy environments
- Software packetization



Solution: Vagrant, Docker, Ansible, Chef, Puppet...



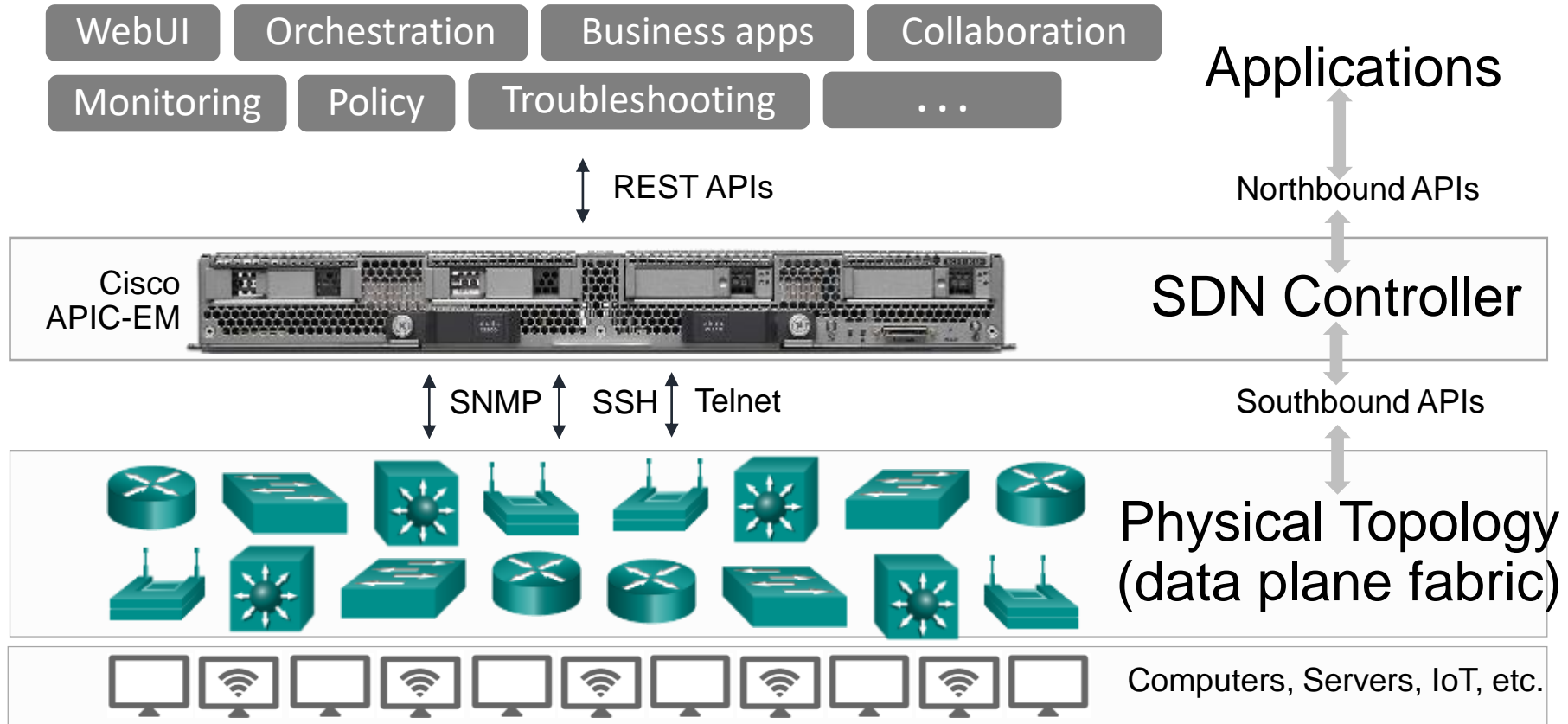
DEVNET

DevNet: opening software



Opening up the software environment to developers encourages innovation and growth. DevNet is a prime resource for facilitating relationships between Cisco and developers. Here then is the need for developers of network applications that interact with the SDN controller, through REST APIs, which we will learn more about shortly, to realize business needs in the operation of the physical infrastructure.

Cisco SDN Model with APIC-EM



What is the APIC-EM?



The Cisco Application Policy Infrastructure Controller Enterprise Module (APIC-EM):

- A Software-Defined Networking (SDN) controller for enterprise networks
- A virtual, software-only, or physical appliance
- Creates an intelligent, open, programmable network with open APIs
- Can transform business-intent policies into dynamic network configuration
- Provides a single point for network-wide automation and control

APIC-EM functionality (I)

Cisco APIC-EM SDN controller communicates with the Physical Topology using standard Southbound API protocols such as SNMP, SSH and Telnet rather than a protocol like OpenFlow.

Easy abstraction of the network fabric to applications using a standard REST API interface. Having such network abstraction in place removes the need for the network staff to configure and manage every single networking device (routers, switches, access points, wireless controllers, etc.) one by one.

Instead, they can concentrate on innovating network-integrated the innovations with network integrated applications that accelerate the digital business growth.

APIC-EM functionality (II)

These applications can run span from big “all in one” orchestration tools, to simple scripts that use a single API call to answer questions such as “How many computers are connected to our network?”

Imagine answering that question SSHing to every router and switch (hundreds of them in many companies), by entering commands at typing multiple times on the CLI like the “show ip route”, “show ip arp”, “show mac address-table”, and then, trying to make some sense of all that output to create meaningful answers.



Does this remember you about Ansible?

APIC-EM: log in

Virtualized APIC-EM Controllers are available in several DevNet Sandboxes:

Always On, NetAcad instances

- For NetAcad users only
- <https://DevNetSBX-NetAcad-APICEM-3.cisco.com>
 - **User:** *devnetuser* **PW:** Xj3BDqbU

Always on, public instance

- For to all DevNet users
- <https://SandBoxAPICEM.cisco.com>
 - **User:** *devnetuser* **PW:** *Cisco123!*



APIC-EM: home page

Expand Navigation Bar

Services

Applications

API documentation

The screenshot shows the APIC-EM home page interface. The left sidebar contains a navigation menu with icons for Home, Discovery, Device Inventory, Host Inventory, Topology, IWAN, Path Trace, Network Plug and Play, and EasyQoS. The main content area displays a 'DEVICE INVENTORY' donut chart showing 13 devices: 1 Managed (green), 11 In-Progress (yellow), and 1 Collection Failure (red). Other sections include 'DISCOVERY - UNREACHABLE DEVICES' (4 Devices), 'BRANCH SITES' (with a link to the IWAN app), 'NETWORK PLUG AND PLAY PROJECTS' (4 projects: 1 Provisioned, 3 Pre-Provisioned, 0 In-Progress, 0 Failed), 'EASYQOS SCOPES', and 'PATH TRACE' (All Path traces are successful).

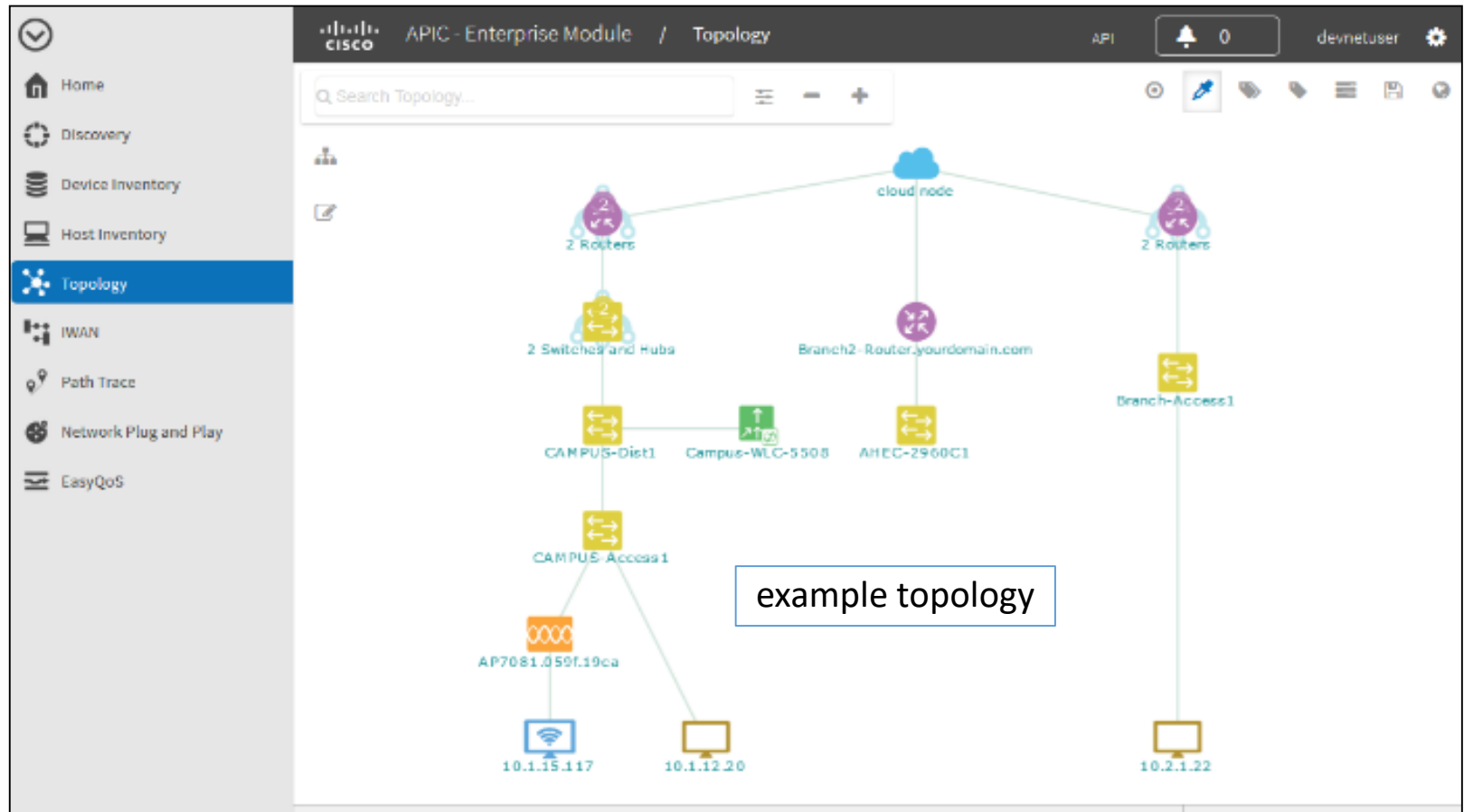
Status	Count
Managed	1
In-Progress	11
Collection Failure	1
Total	13

Status	Count
Provisioned	1
Pre-Provisioned	3
In-Progress	0
Failed	0
Total	4


APIC-EM: applications










- **Network Plug-and-Play (PnP)**
Provides a unified approach to provision enterprise networks comprised of Cisco routers, switches, and wireless access points with a near-zero-touch deployment experience.
- **Easy QoS**
Provides a simple way to classify and assign application priority.
- **Intelligent WAN (IWAN)**
Simplifies WAN deployments by providing an intuitive, policy-based interface that helps IT abstract network complexity and design for business intent.
- **Path Trace**
Greatly eases and accelerates the task of connection monitoring and troubleshooting.

APIC-EM: topology page



APIC-EM: API (I)

 APIC - Enterprise Module / Swagger



Available APIs
[File](#)
[Flow Analysis](#)
[Grouping](#)
[IP Geolocation](#)
[IP Pool Manager](#)
[Identity-Manager](#)
Inventory
[Network Discovery](#)
[Network Plug and Play](#)
[PKI Broker Service](#)
[Policy Administration](#)
[Role Based Access Control](#)
[Scheduler](#)
[Task](#)
[Topology](#)
[Visibility](#)

Inventory

APIC-EM Service API based on the Swagger™ 1.2 specification

[Terms of service](#)
[Cisco DevNet](#)

device-credential : Device Credential API	Show/Hide	List Operations	Expand Operations	Raw
discovery : Discovery API	Show/Hide	List Operations	Expand Operations	Raw
host : host API	Show/Hide	List Operations	Expand Operations	Raw
GET /host	Retrieve hosts			
GET /host/count	Gives total number of hosts			
GET /host/{id}	Retrieves host based on id			
interface : Interface API	Show/Hide	List Operations	Expand Operations	Raw
license : license API	Show/Hide	List Operations	Expand Operations	Raw
location : Location API	Show/Hide	List Operations	Expand Operations	Raw
network-device : network-device API	Show/Hide	List Operations	Expand Operations	Raw
network-device-config : Network Device Configuration API	Show/Hide	List Operations	Expand Operations	Raw
segment : Segment API. Currently, wireless type is supported	Show/Hide	List Operations	Expand Operations	Raw
tag : Tag API	Show/Hide	List Operations	Expand Operations	Raw
vlan : Vlan API	Show/Hide	List Operations	Expand Operations	Raw

[BASE URL: <https://devnetsbx-netacad-apicem-3.cisco.com/api/v1/api-docs/inventory-manager> , API VERSION: 1.0]

APIC-EM: API (II)

host : host API

Show/Hide

List Operations

Expand Operations

Raw

GET

/host

Retrieve hosts

Implementation Notes

Get Hosts

Response Class

Model | Model Schema

HostListResult {

version (string, optional),

response (array[HostDTO], optional)

}

HostDTO {

hostName (string, optional): Name of the host,

source (string): Source from which the host gets collected. Available option:200 for inventory collection and 300 for trap based data collection,

id (string): Id of the host,

vlanId (string, optional): Vlan Id of the host,

subType (string, optional) = ['UNKNOWN' or 'IP_PHONE' or 'TELEPRESENCE' or 'VIDEO_SURVEILLANCE_IP_CAMERA' or 'VIDEO_ENDPOINT'],

lastUpdated (string): Time when the host info last got updated,

avgUpdateFrequency (string): Frequency in which host info gets updated,

connectedAPMacAddress (string, optional): Mac address of the AP to which wireless host gets connected,

connectedAPName (string, optional): Name of the AP to which wireless host gets connected,

connectedInterfaceId (string, optional): Id of the interface to which host gets connected,

connectedInterfaceName (string, optional): Name of the interface to which host gets connected,

connectedNetworkDeviceId (string): Id of the network device to which host gets connected,

connectedNetworkDeviceIpAddress (string): Ip address of the network device to which host gets connected,

hostIp (string): Ip address of the host,

hostMac (string): Mac address of the host,

hostType (string): Type of the host. Available options are: Wired, Wireless,

pointOfAttachment (string, optional): Id of the Host's Point of attachment network device (wlc). Based on mobility,

pointOfPresence (string, optional): Id of the Host's Point of presence network device (wlc). Based on mobility,

attributeInfo (object, optional)

}

Response Content Type: application/json

APIC-EM: API (III)

Parameters

Parameter	Value	Description	Parameter Type	Data Type
limit	<input type="text"/>	limit	query	string
offset	<input type="text"/>	offset	query	string
sortBy	<input type="text"/>	sortBy	query	string
order	<input type="text"/>	order	query	string
hostName	<input type="text"/>	hostName	query	List
hostMac	<input type="text"/>	hostMac	query	List
hostType	<input type="text"/>	hostType	query	List
connectedInterfaceName	<input type="text"/>	connectedInterfaceName	query	List
hostIp	<input type="text"/>	hostIp	query	List
connectedNetworkDeviceIpAddress	<input type="text"/>	connectedNetworkDeviceIpAddress	query	List
subType	<input type="text"/>	Available values: 'UNKNOWN' or 'IP_PHONE' or 'TELEPRESENCE' or 'VIDEO_SURVEILLANCE_IP_CAMERA' or 'VIDEO_ENDPOINT'. Only exact match filtering supported on this field	query	List
filterOperation	<input type="text"/>	startswith/contains/endswith	query	string

Error Status Codes

HTTP Status Code	Reason
200	This Request is OK
403	This user is Forbidden Access to this Resource
401	Not Authorized Yet, Credentials to be supplied
404	No Resource Found

Try it out!

Gracias por vuestra atención



pue

IMPULSANDO EL CONOCIMIENTO
TIC CUALIFICADO

Iván Lago - Técnico Cisco Networking Academy ASC/ITC
PUE - ITC/ASC/CA
Área de Proyectos de Educación