Introduction to Software Defined Networking

Part 2
Robert Yee, CCIE 11716
Cisco Systems Engineer
October 2013
ONE = Open Network Environment

Cisco’s Strategy for SDN
Simple Vendor Deployments in the Industry

Vendor A
- Virtual Overlays
- Physical and Virtual
- Network

Vendor B
- APIs
- Network

Vendor C
- Controller
- OpenFlow Device
- Device w/ OpenFlow
- Device

Cisco Approach: Flexibility to Choose—The Power of “AND”
Three Component to Cisco One

Overlays
- Nexus 1000v
- Multi-Hypervisor
- Multi-Service
- Multi-Cloud
- Openstack Support

OnePK
- Programmable API’s
  - IOS
  - IOS-XE
  - IOS-XR
  - NX-OS

Controller
- Controller Software
  - Openflow 1.x
  - REST API
  - OnePK API

Cisco Confidential – Strictly for Internal Cisco Use Only
Overlays

Let us explore Overlays and the Nexus 1000v
What is the Nexus 1000v?
Nexus 1000v = A Virtual Switch

“A software based switch that runs on the hypervisor and lives in the server...”
How does it compare to a physical switch?
Cisco Nexus 1000V Architecture

Modular Switch

- Supervisor-1
- Supervisor-2
- Linecard-1
- Linecard-2
- …
- Linecard-N

Virtual Appliance

Nexus 1110

- VSM-A1
- VSM-A4
- VSM-B1
- VSM-B4

- ESX
- …

VSM: Virtual Supervisor Module
VEM: Virtual Ethernet Module
VSM can run on a dedicated appliance – the Nexus 1110

VSM can also run as a Virtual Machine (VM) alongside the VEM
Replacement for VMWare’s distributed switch
VEM sits in the Hypervisor kernel
VEM adds another (logical) switch layer into the switch hierarchy
There is another ingredient we need to talk about:

**VXLAN**

*(Virtual Extensible LAN)*
VXLAN solves a Layer 2 Adjacency problem for multi tenant networks
What is Multi-Tenancy?

Ability to support multiple customer networks on the same physical network without them interfering with one another


doesn't explore that...
Data Center has server racks available to host customer applications.
The racks have been setup each with their own IP Subnet.
A customer buys up server space to host their business applications.
Over time their business grows and they consume all server space in the rack.

This customer needs to expand into another rack.
This is where we have a problem.

There are already other customers' networks in those racks that use the same racks IP address subnet range...
Green servers are now located in dis-contiguous subnets?

Problem – How do you provide Layer 2 adjacency for all Green servers across the Layer 3 network?
The answer is VXLAN

*It provides connectivity between Layer 2 networks across a Layer 3 network*
Let us see how the Nexus 1000V uses VXLAN
We start out with our network scenario
We have physical connections between servers and network devices

That is real cables and plugs you can touch!!
And then we add our overlay
Remember the VEM is a switch

*It's our “virtual” switch*
We then setup “connections” between virtual switches.

Creating our “logical” virtual network
These “connections” are VXLAN tunnels

*Providing the data transport between VM’s*
VXLAN also helps overcome a scalability issue
In a normal Data Frame the VLAN TAG is a 12 bit field

$2^{12} = 4096$ VLANs
Before VXLAN Encapsulation

MAC DA  MAC SA  VLAN TAG  IP DA  IP SA  DATA

After VXLAN Encapsulation

MAC SA  VLAN TAG  IP DA  IP SA  UDP HDR  VXLAN ID

New Layer 2 (MAC) header is added
New IP HDR Source/Dest Tunnel ADDR

Original Layer 2 Header and DATA placed in Data Payload of new frame
In the VXLAN header, the ID TAG is a 24 bit field.

\[ 2^{24} = 16,777,216 \text{ VLANs}^* \]

*Technically not quite a VLAN but we will use this for comparison purposes.
OnePK

Let us explore OnePK
What the heck is OnePK?
OnePK = One Programmable Kit

“Allows an external application to access, extend or customize the software capabilities of Cisco’s routers and switches via API’s...”
What is an API?
API = Application Programming Interface

Protocol enabling software components to talk to one another

Application

Component #1

Component #2

Component #n

Application

Component #x

Component #y

Component #z
Who is going to use OnePK?

Developers | Operators | Engineers

Anyone who wants to
*customize, automate or extend*
the functionality of their Cisco Switches and Routers
What can you do with OnePK?

Program your networking device
Create services | Automate Tasks | Manipulate Forwarding Topology | Modify Security Policies | etc.

Gain access to information
Statistics | Counters | Configuration | Topology “State”
Where does it fit in the stack?
Start with the physical switch or router

Hardware Silicon
Then add the device OS

Hardware Silicon

IOS-XE | NX-OS | IOS-XR
OnePK sits on top of the OS

Cisco OnePK API’s

IOS-XE | NX-OS | IOS-XR

Hardware Silicon

Physical Device
Cisco OnePK API’s

IOS-XE | NX-OS | IOS-XR

Hardware Silicon

Applications

Physical Device
What APIs does OnePK provide?

OnePK’s API’s are known as “Service Sets”

Available Service Sets Include

Data Path | Policy | Routing | Element
Discovery | Utility | LISP | Developer
DATA PATH: Packet delivery services to application – e.g. copy, punt, inject

POLICY: Filtering (NBAR), classification (class-map, policy-map), actions (marking, policing, queuing), applying policy to interfaces

ROUTING: Read RIB Routes, add/remove routes, receive RIB notifications

ELEMENT: CPU/Memory statistics, interface statistics, element and interface events

DISCOVERY: Layer 3 topology and local service discovery

UTILITY: SYSLOG event and path tracing capability

DEVELOPER: Debug capability and CLI extension (invoke CLI from application)
### OnePK Customer Examples - many from diverse backgrounds

<table>
<thead>
<tr>
<th>Start Ups</th>
<th>MSDC</th>
<th>SP</th>
<th>Universities</th>
<th>Defense</th>
<th>Integrators</th>
<th>Other EL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubicity</td>
<td>Facebook</td>
<td>Comcast</td>
<td>Georgia Tech</td>
<td>US Army</td>
<td>Tech Mahindra</td>
<td>Sony</td>
</tr>
<tr>
<td>Zenoss</td>
<td></td>
<td>British Telecom</td>
<td>Indiana State</td>
<td>US Navy</td>
<td></td>
<td>Apple</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AT&amp;T</td>
<td>NC State</td>
<td>Marine Corp.</td>
<td></td>
<td>Symantec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Primus Canada</td>
<td>Princeton</td>
<td>Dutch MOD</td>
<td></td>
<td>Tavve</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Verizon</td>
<td>Harvard</td>
<td>Norwegian Defense</td>
<td></td>
<td>CalArts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Orange</td>
<td></td>
<td></td>
<td></td>
<td>Citibank</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ikeae</td>
</tr>
</tbody>
</table>

Cisco Confidential – Strictly for Internal Cisco Use Only
<table>
<thead>
<tr>
<th>Device</th>
<th>Q1 CY13</th>
<th>Q2 CY13</th>
<th>Q3 CY13</th>
<th>Q4 CY13</th>
<th>H1 CY14</th>
<th>H2 CY14</th>
</tr>
</thead>
<tbody>
<tr>
<td>XE</td>
<td>ISRG2 PI21 FCS/CA</td>
<td>ISRG2 PI22 GA</td>
<td>ISRG2 PI22 GA</td>
<td>ASR1K XE3.10 FCS/CA</td>
<td>ASR1K XE3.11 – GA</td>
<td>CAT3K/4K Amur CA</td>
</tr>
<tr>
<td>E</td>
<td>ASR1K XE3.10 FCS/CA</td>
<td>ISR4400 XE3.10 FCS/CA</td>
<td>ISR4400 XE3.10 FCS/CA</td>
<td>CSR1000V XE3.10 FCS/CA (phase 1)</td>
<td>CSR1000V XE3.11 (phase 2) FCS/CA</td>
<td>CSR1000V XE3.12 - GA</td>
</tr>
<tr>
<td>R</td>
<td>ASR9K 4.3.1 EFT</td>
<td>ASR9K 5.1.0 EFT</td>
<td>ASR9K 5.1.1 FCS/CA</td>
<td>CRS 5.1.1 EFT</td>
<td>CR 5.X(?) FCS/CA</td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>N3K Eden EFT, Limited CA</td>
<td>N3K/3500 Fiji FCS/CA 100%</td>
<td>N5K/6K Iulka CA</td>
<td>N3K/3500 Goa GA</td>
<td>N5K/6K (Jan-Juc) GA</td>
<td>N7K Gibraltar GA</td>
</tr>
<tr>
<td>C and Java Support (FCS/CA)</td>
<td>SDK GA including Java, C, Python, IOS Emulator GA</td>
<td>REST Support</td>
<td>IOS XE Emulator</td>
<td>IOS XR Emulator</td>
<td>Nexus Emulator Packaging Utility GA</td>
<td>Ruby Support</td>
</tr>
<tr>
<td>Python (EFT)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Cisco OnePK Roadmap - Agent Plans**
Let us explore the Cisco One Controller
XNC is downloadable as of 10/9/2013 from cisco.com.
Let's look at the Controller
The controller code was developed by Cisco and includes some open source elements.

Written in Java.

It is being enhanced by Cisco engineers with many years of networking knowledge.
The controller utilizes OSGI framework
Open Services Gateway Initiative

This is a cool feature...
Why you might ask?

It allows apps to be installed, removed, started or stopped without requiring a reboot.
Southbound API supports Openflow (as expected)

It also supports OnePK API

It is extensible to support other options in the future also
Northbound API supports the *RESTFUL API*

Representational State Transfer

Uses familiar web (HTTP) methods for an application to request something to be done

(GET | PUT | POST | DELETE)
Oh, there is 2 more SDN initiatives at Cisco
What is Project Daylight?

Daylight is an open source project formed by industry leaders and others under the Linux Foundation with the mutual goal of furthering the adoption and innovation of Software Defined Networking (SDN) through the creation of a common vendor supported framework.
OpenDaylight Controller

Network Applications

OpenDaylight Controller

JAVA BUNDLE

Northbound APIs

OSGI

RESTful

GUI

H/A

Basic Operation Infrastructure

Dijkstra SPF

Forwarding Rules Manager

Host Tracker

Physical and Logical Topology Manager

Device Manager

Service Abstraction Layer (SAL)

Southbound APIs

OF 1.0

NETWORK DEVICES

Cisco Contributing the *Foundation* Controller Technology

Cisco Confidential – Strictly for Internal Cisco Use Only
Importance of OpenDaylight

Market is Demanding SDN Be Open
  Daylight Adoption Means Cisco Controller is Open Source Based
  Industry-Wide Alignment for Controller Functionality
  Reduces Fragmentation
  Promotes a Robust Ecosystem for Programmable Networking
  Contributions within OpenDaylight
  New Third Parties Opportunities