We’re ready. Are you?
Who is Wolfgang Riedel ???

- **Personal:**
  - Location: Erlangen, Germany (between Munich – Frankfurt)
  - Other Interests: Alpine Snowboarding, High-End Audio, AS51871, Data Center, Real World LAB, High-performance sports cars, Geothermal DC cooling research project, …

- **Background:**
  - Joined CISCO January 2001
  - Before; self-employed as an in-depended consultant in the Networking and IT space for more than fifteen years.
    - SE – RS Germany (2001 – 2006) -> Campus with a DC attached
    - CSE – DC EMEA (2006 - 2008) -> DC with Campus attached
    - PE – CTO Team ENG (2013 - 2014)
    - PE – Architecture Team ENG (2014 – …)
  - HA Campus & DC Design, Routed Access, DC POD Design
  - CCIE RS, VCP 3/4/5 and pile of CPOC’s
  - Worked with more than 250 customers within several projects over the last +15 years
  - Individual Contributor: Cat4k, Cat6k, N7k, ASR1k, FC, FCoE, DCB, UCS, N5k, N2k, N1k, PoE FEX, vPC, OTV, LISP (Pioneer Award), OF, SDN

- **Stuff I am currently working on:**
  - Network Transformation, Architecture (Mark, Matthias, Tim, Dave, Jason, Simone, I)
  - APIC-EM, DNS-AS, AVC, USP
  - TECSDN-3600 + BRKCRS-3011 + BRKSDN-3004
Agenda

1. Introduction
2. World of Controllers and Technologies
3. Controllers
   1. OpenFlow – Stanford Clean Slate
   2. APIC-DC - Application Policy Infrastructure Controller for the Datacenter
   3. Virtual Managed Services
   4. OSC – Open SDN Controller
   5. ODL – Open Day Light Controller
   6. APIC-EM - Application Policy Infrastructure Controller for the Enterprise
   7. PI and APIC-EM
4. APIC-EM
   1. Policy Infrastructure
   2. Auto Scale Architecture
   3. Grapevine Cloud Deployment
   4. Use Cases
5. Demo
6. A Few Conclusions and Q&A, if we have time
1. Introduction
Industry trends in Networking

Cloud (2008)

Software Defined Networking (2012)

Open Daylight Project (2013)

DevOps, The API Driven Datacenter (2013)

Network Function Virtualization (2013)

Managing Networks through abstractions (2014)

Metadata Driven Networking (2016)

Atomic Services (2018)
SDN – Still Don’t kNow – Stanford Defined Networking

The Promise of OF/SDN had been “Decoupling Policy from Configuration”

Physical separation of control and data plane

Managing the network through abstractions

Software Defined Networking

You can’t just buy SDN. It’s an architecture which you have to embrace and live

With SDN I can develop solutions to my problems far faster – “at software speeds”. I don’t have to work with my network vendor or go through lengthy standardization processes.

Whitebox routing and switching

Running networks in agile DEV-OPS model

Cisco live!
SDN – Hype Cycle

Where we are with SDN 2016, five years later

- Technology Trigger
- Peak of Inflated Expectations
- **Trough of Disillusionment**
  - Interest wanes as experiments and implementations fail to deliver.
  - Producers of the technology shake out or fail.
  - Investments continue only if the surviving providers improve their products to the satisfaction of early adopters.
- Slope of Enlightenment
- Plateau of Productivity

Gartner Hype Cycle
Today’s DC Architectural Battle

System administration is over - we should stop doing it

**Web Approach (MSDC)**
- IT infrastructure core of its business
- Warehouse Datacenter
- Scale-Out Architecture
- ~100,000 of physical servers
- Single Application Optimization
- Small Number of Applications, like Gmail, Google+, Office 360, Xbox, Bing, …
- Application Designed for Failure
- Automate everything possible
- It’s all about being super-cheap commodity systems; costs must grow in a "sub-linear" fashion
- Open Source
- Backbone Bandwidth Calendaring
- TDM style provisioning with custom TCP stack
- L3 Topology

**Enterprise Approach (EPDC)**
- IT infrastructure is an expense
- “Discovery” Datacenter
- Scale-Up Architecture
- ~10,000 physical servers
- Thousands of Applications
- Application trust boundaries
- HA failover model
- Transactional
- Application specific Infrastructure
- Commercial Of The Shelf
- L2 Topology
Today’s DC Architectural Battle
Device to Admin Ratio

2009

Traditional IT: 50:1
Amazon: 200:1
Google: 10000:1

2013

Traditional IT 50:1
Amazon 10000:1
Google: 30000:1
Today’s DC Architectural Battle

It’s all about the Application

Web Approach (MSDC)

Cooperative Applications

- Client Signaling
- Server Signaling
- DNS-AS
- Collaboration for DPI Signatures

Enterprise Approach (EPDC)

Non-Cooperative Applications

- DPI Custom Matches
- DPI (Various Techniques and Sources)
- Statistical
- Active Databases (e.g. IP Reputation)
SDx Influences Network Purchases

Source: SDx Central
SDx Spending by Customer and Use Cases

Source: SDx Central
Managing the network through abstractions

There are two approaches to Control Systems

**IMPERATIVE CONTROL**
Baggage handlers follow sequences of simple, basic instructions

**DECLARATIVE CONTROL**
Air traffic control tells where to take off from, but not *how* to fly the plane
Managing the network through abstractions
There are two approaches to Control Systems

**IMPERATIVE CONTROL**

![Image of complex control panel]

**DECLARATIVE CONTROL**

![Image of simple control panel]

It’s 2016 and network admins still enjoy being “masters of complexity”
### APIC - EM Design Points

**Validated with FAT & CAT**

<table>
<thead>
<tr>
<th>Design Point</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce Network Complexity</td>
<td>Abstraction and Automation of Manual Network Operations</td>
</tr>
<tr>
<td>Brownfield Support – No Software / Hardware upgrade required</td>
<td>Advanced Visualization (HTML5/Java code with object oriented interface)</td>
</tr>
<tr>
<td>Low Risk adoption of SDN</td>
<td>Start with small set of solvable problems management, Zero Touch Deployment and IWAN as key applications with identifiable metrics (OPEX savings, ROI)</td>
</tr>
<tr>
<td>Enterprise Scale to production network use</td>
<td>Elastic Services Infrastructure ensures scaling as adoption grows</td>
</tr>
<tr>
<td>Problem set can be minimal to no programming requirement</td>
<td>Auto-Translation of high level business intent into network control function</td>
</tr>
<tr>
<td></td>
<td>Advanced analytics for real time network visibility and faster response time</td>
</tr>
</tbody>
</table>

Don’t make it simpler by making it more complex!
Enterprise SDN customer asks in an iPhone world

By Eric Burke

STUFFTHATHAPPENS.COM BY ERIC BURKE
New Network Needs for the Digital Business

**Information Era Network**
- Closed and Hardware Centric
- Manual Box-by-Box Management
- Perimeter Based Reactive Security
- IT & Historical Analytics

**Digital Ready Network**
- Open, Programmable, Software Driven
- Network Wide Policy Based Automation
- Proactive Context-Based Security Everywhere
- Business & Real Time Analytics
Principles for the New Network Architecture

Open and Software-Driven

Cloud
Services & Apps Built for Cloud Consumption
On-Demand Scale
Faster IT Innovation

Controllers
Complete Controller-based Automation
Simplicity through Abstraction
Centralized Policy

Virtualization
Virtualized Networks Functions and App Hosting
Freedom of Choice - Any Platform
Run Applications over the Network
Digital Network Architecture (DNA)

Cloud and On-Premise Services

Controller Based Network Abstraction

Virtualized and Physical Network Elements

NETWORK
- Automation Assurance

SECURITY
- Sensor Enforcer

DIGITAL
- Experience Analytics

Open APIs | Extensible | Developers

Orchestration

Policy

Visibility

Programmable | Standards-Based

Flexible Pipeline
- Run Any Feature Anywhere

Smart Sensors

Any Platform

App Hosting

Cisco ASIC
- IOS | VNF

Virtual Networks
## DNA Service Innovation

### Available Today

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIC EM</td>
<td>Enterprise Network Controller</td>
</tr>
<tr>
<td>IWAN App</td>
<td>Automate Hybrid WAN Services</td>
</tr>
<tr>
<td>Path Trace App</td>
<td>Faster Troubleshooting</td>
</tr>
</tbody>
</table>

### New Services

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plug and Play</td>
<td>Automate Day 0 Deployment</td>
<td>March 2016</td>
</tr>
<tr>
<td>NFV</td>
<td>Run Services Anywhere</td>
<td>March 2016</td>
</tr>
<tr>
<td>Easy QoS</td>
<td>Automate Collab Traffic Priority</td>
<td>Available</td>
</tr>
<tr>
<td>Lancope</td>
<td>Network as a Sensor</td>
<td>January 2016</td>
</tr>
<tr>
<td>CMX Cloud</td>
<td>Location Service Subscription</td>
<td>Available</td>
</tr>
</tbody>
</table>

Available on DNA-Ready Infrastructure through Cisco ONE Software

- ISR 4000
- ASR 1000
- Catalyst 6800
- Catalyst 4000-E
- Catalyst 3850
- Catalyst 3650
- Aironet 802.11ac
“People who are really serious about software should make their own hardware.”

Alan Kay, 1982
1.1 Analogies
Distributed Networking has **worked**

Resiliency/Scale has been **proven**
Distributed Networking has worked

However

Distributed Networking adds complexity to manage/comprehend
Admin still makes network behavior decisions

But uses controller to mask complexity
Translation of high level constructs to network control functions reduces skills gaps and clarifies policy procedures.
Allow Protocol/API choice while maintaining stack integrity

- Web UI
- YANG
- REST API

- CLI
- SNMP
- Web UI
- NETCONF
- XML
- onePK
- Openstack

Network Devices

Controller

Applications
Both at one time had **direct admin control**
Direction to abstract complexity

Network Management should follow Web Development

Focus on the What and not How

WWW Admin

Web Dev GUI

Controller

Network Admin

Network

WWW

2005
Power Technologist

2010
Application Developers

2013
Non Technical Users

2014
Intent Networking

2015
Partial Automation

2018
Self Healing
What I am doing next?

So this begs the Question?
2. World of Controllers and Technologies
Network Programmability Models

1. Programmable APIs
   - Applications
     - Vendor-specific APIs
   - Control Plane
   - Data Plane

2a. Classic SDN
   - Applications
     - Vendor-specific APIs
   - Control Plane
   - Data Plane
   - OpenFlow and/or Vendor specific

2b. Hybrid “SDN”
   - Applications
     - Vendor-specific APIs
   - Control Plane
   - Data Plane
   - OpenFlow and/or Vendor specific

3. Network Virtualization/Virtual Overlays
   - Applications
     - Vendor-specific APIs
   - Control Plane
   - Virtual Control Plane
   - Virtual Data Plane
   - Overlay Protocols

4. Policy Intent
   - Applications
     - Policy Authority
     - Policy Controller
     - Policy Plane
     - Policy Agent
     - Vendor-specific APIs
SDN Controllers – Types

There’s nothing like the SDN controller

- **SDN Config-Pusher**
  - Orchestration (robot micromanaging manual to-do’s)
  - NCM (Network Configuration Management)
  - Customers may see or edit any part of the config
  - Prime Infrastructure, Action Packed, Solarwinds
  - Puppet, Chef
  - Openstack
  - Netconf

- **SDN Policy-Compiler**
  - Customer is never exposed to nor has access to nor influence over direct snippets of configuration elements.
  - They express their intent only – like in a programming language – and the conversion to machine language is invisible.
  - Cisco APIC-EM

- **SDN Policy-Enabler**
  - Cisco APIC-DC

- **SDN Overlay Controller**
  - VMWare: VCS, VCD, NSX
  - VSM (N1kv), EVP, VTS
  - Windows Server, Microsoft System Center

- **SDN Open Flow Controller**
  - Primary for research
SDN Controllers – Types

Start with the End in Mind - the RYF-complex (Fragile/Robust)

Five dimensions of robustness in complex systems
(1) Reliability
(2) Efficiency
(3) Scalability
(4) Modularity
(5) Evolvability

See J. Doyle, et. al., “Robustness and the Internet: Theoretical Foundations.”

Alderson and Doyle identify four kinds of constraints on system robustness:
(1) Component-level
(2) System-level
(3) Protocols
(4) Emergent constraints

Complex systems science as conflicting constraints
John C. Doyle, HOT and SF networks

Grateful Dead Sources
How the Dragon Unbouros (Giga Exponentia) Makes Us Go Round and Round

Ciscolive!
Device Programmability Options

No Single Answer!

Application Frameworks, Management Systems, Controllers, ...

- C/Java
- Python
- NETCONF
- REST
- OpenFlow
- ACI Fabric
- OpenStack
- Puppet
- Protocols

API and Data Models

- YANG
- JSON

Operating Systems – IOS / NX-OS / IOS-XR

Management
Orchestration
Network Services
Control
Forwarding
Cross Domain Controller Architecture

ODL - **Group Based Policy Repository - Network Intent Composition**

- **Service Provider Applications**
  - XML + REST API
  - Domain-Controller: OSS
  - VMS (Virtual Managed Services)
  - Netconf/YANG
  - Open API's
  - Network Device ASR9k – CRS – KVM
  - Third Party

- **Enterprise Applications**
  - REST API
  - Domain-Controller: PRIME
  - Enterprise-Fabric
  - APIC – EM
  - Network Info Database
  - Policy Infrastructure
  - Automation

- **Data Center Applications**
  - REST API
  - Domain-Controller: UCS Director
  - Application-Fabric
  - APIC – DC (Physical+Virtual)
  - Network Info Database
  - Policy Infrastructure
  - Automation + AVS

- **Data Center Legacy**
  - REST API
  - Domain-Controller: JCS Director
  - Standalone-Fabric
  - DCNM - underlay
  - VTS - overlay
  - Network Info Database
  - Policy Infrastructure
  - Automation + VSM

- **Domain-Controller:**
  - PRIME
  - UCIM (UCI)
  - SSH, telnet, https, http, snmp
  - Network Devices
  - Catalyst, ASR, ISR, WLC, NEXUS 7k

- **Domain-Controller:**
  - UCS Director
  - VTS
  - NX-API's, XML, Puppet, Chef, Ansible
  - Network Devices
  - NEXUS 2k, 3k, 5k, 6k, 7k

- **Policy Renderers:**
  - ODL
  - ODL - Group Based Policy Repository - Network Intent Composition
  - Group Based Policy Composition
  - Network Content Composition

- **APIs:**
  - Netconf/YANG
  - Open API's
  - UCI
  - ACI
  - OpFlex
  - LISP
  - EVPN

- **EVPN:**
  - EVPN
  - LISP
  - ODL – Group Based Policy Repository - Network Intent Composition

- **Networking:**
  - VMS (Virtual Managed Services)
  - Network Devices
  - Catalyst, ASR, ISR, WLC, NEXUS 7k

- **Automation:**
  - AVS
  - ESP

- **Legacy:**
  - Data Center
  - Applications
  - Network Info Database
  - Policy Infrastructure
  - Automation

- **Standalone-Fabric:**
  - DCNM - underlay
  - VTS - overlay
  - Network Info Database
  - Policy Infrastructure
  - Automation + VSM

- **Network Devices:**
  - NEXUS 9k

- **Third Party:**
  - Nexus Interface
  - Nexus Interface
  - Nexus Interface

- **UCI:**
  - SSH, telnet, https, http, snmp
  - Network Devices
  - Catalyst, ASR, ISR, WLC, NEXUS 7k

- **Policy Rendering:**
  - ODL
  - ODL – Group Based Policy Repository - Network Intent Composition
  - Group Based Policy Composition
  - Network Content Composition

- **Network Info Database:**
  - Network Devices
  - Catalyst, ASR, ISR, WLC, NEXUS 7k

- **Policy Infrastructure:**
  - Network Devices
  - Catalyst, ASR, ISR, WLC, NEXUS 7k

- **Automation:**
  - AVS
  - ESP

- **EVPN:**
  - EVPN
  - LISP
  - ODL – Group Based Policy Repository - Network Intent Composition

- **Networking:**
  - VMS (Virtual Managed Services)
  - Network Devices
  - Catalyst, ASR, ISR, WLC, NEXUS 7k
SDN Controller – Overview

OK that looks really ugly but wait a minute…

… all cars

• Four wheels
• Steering wheel
• Gas pedal
• Brake pedal

But complete different use-cases
3. Controllers
3.1 OpenFlow - Stanford Clean Slate
• Mandates a separation of control and data plane and an interface to it
• Higher level abstraction to store state information
• Classical networking relies on distributed state but local decision making
• SDN Networks is all about distributed state but central decision making
• Enabler to deal with the networks in a different way as we do today

- OF allows up to 12-tuple wildcard and/or exact match
SDN Controllers – Types
There’s nothing like “a OpenFlow controller”

<table>
<thead>
<tr>
<th>SDN Open Flow Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NOX</strong> (C++/Python) NOX was the first OpenFlow controller.</td>
</tr>
<tr>
<td><strong>POX</strong> (Python) POX as a general SDN controller that supports OpenFlow. It has a high-level SDN API including a queriable topology graph and support for virtualization.</td>
</tr>
<tr>
<td><strong>Jaxon</strong> (Java) Jaxon is a NOX-dependent Java-based OpenFlow Controller.</td>
</tr>
<tr>
<td><strong>Trema</strong> (C/Ruby) Trema is a full-stack framework for developing OpenFlow controllers in Ruby and C.</td>
</tr>
<tr>
<td><strong>Beacon</strong> (Java) Beacon is a Java-based controller that supports both event-based and threaded operation.</td>
</tr>
<tr>
<td><strong>Floodlight</strong> (Java) The Floodlight controller is Java-based OpenFlow Controller. It was forked from the Beacon controller, originally developed by David Erickson at Stanford.</td>
</tr>
<tr>
<td><strong>Maestro</strong> (Java) Maestro is an OpenFlow “operating system” for orchestrating network control applications.</td>
</tr>
<tr>
<td><strong>NDDI - OESS</strong> OESS is an application to configure and control OpenFlow Enabled switches through a very simple and user friendly User Interface.</td>
</tr>
<tr>
<td><strong>Ryu</strong> (Python) Ryu is an open-sourced Network Operating System (NOS) that supports OpenFlow.</td>
</tr>
<tr>
<td><strong>NodeFlow</strong> (JavaScript) NodeFlow is an OpenFlow controller written in pure JavaScript for Node.JS.</td>
</tr>
<tr>
<td><strong>ovs-controller</strong> (C) Trivial reference controller packaged with Open vSwitch.</td>
</tr>
<tr>
<td><strong>RouteFlow</strong> RouteFlow, is an open source project to provide virtualized IP routing services over OpenFlow enabled hardware. RouteFlow is composed by an OpenFlow Controller application, an independent RouteFlow Server, and a virtual network environment that reproduces the connectivity of a physical infrastructure and runs IP routing engines (e.g. Quagga).</td>
</tr>
<tr>
<td><strong>Flowvisor</strong> (Java) FlowVisor is a special purpose OpenFlow controller that acts as a transparent proxy between OpenFlow switches and multiple OpenFlow controllers.</td>
</tr>
<tr>
<td><strong>SNAC</strong> (C++) SNAC is an OpenFlow controller built on NOX, which uses a web-based policy manager to manage the network.</td>
</tr>
<tr>
<td><strong>Resonance</strong> Resonance is a Network Access Control application built using NOX and OpenFlow.</td>
</tr>
<tr>
<td><strong>OFlops</strong> (C) OFlops (OpenFlow Operations Per Second) is a standalone controller that benchmarks various aspects of an OpenFlow switch.</td>
</tr>
<tr>
<td><strong>RouteFlow</strong> RouteFlow, is an open source project to provide virtualized IP routing services over OpenFlow enabled hardware. RouteFlow is composed by an OpenFlow Controller application, an independent RouteFlow Server, and a virtual network environment that reproduces the connectivity of a physical infrastructure and runs IP routing engines (e.g. Quagga).</td>
</tr>
<tr>
<td><strong>Flowvisor</strong> (Java) FlowVisor is a special purpose OpenFlow controller that acts as a transparent proxy between OpenFlow switches and multiple OpenFlow controllers.</td>
</tr>
<tr>
<td><strong>XNC</strong> Cisco Extensible Network Controller (XNC) is the first commercial version of the OpenDaylight controller.</td>
</tr>
<tr>
<td><strong>ODL</strong> Linux-Foundation: community-driven, open source controller framework (Brocade, Cisco, Citrix, Ericsson, IBM, Juniper, Microsoft, RedHat)</td>
</tr>
</tbody>
</table>
### SDN Controllers – Types

#### Comparison of Open Source Controllers

<table>
<thead>
<tr>
<th>Use-Cases</th>
<th>Trema</th>
<th>Nox/Pox</th>
<th>Ryu</th>
<th>Floodlight</th>
<th>ODL</th>
<th>ONOS***</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Virtualization by Virtual Overlays</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>PARTIAL</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Hop-by-hop Network Virtualization</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>OpenStack Neutron Support</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Legacy Network Interoperability</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>PARTIAL</td>
</tr>
<tr>
<td>Service Insertion and Chaining</td>
<td>NO</td>
<td>NO</td>
<td>PARTIAL</td>
<td>NO</td>
<td>YES</td>
<td>PARTIAL</td>
</tr>
<tr>
<td>Network Monitoring</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Policy Enforcement</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>PARTIAL</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Load Balancing</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Traffic Engineering</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>YES</td>
<td>PARTIAL</td>
</tr>
<tr>
<td>Dynamic Network Taps</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Multi-Layer Network Optimization</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
</tr>
<tr>
<td>Transport Networks - NV, Traffic-Rerouting, Interconnecting DCs, etc.</td>
<td>NO</td>
<td>NO</td>
<td>PARTIAL</td>
<td>NO</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
</tr>
<tr>
<td>Campus Networks</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>PARTIAL</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Routing</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
</tbody>
</table>
3.2 APIC-DC

(Application Policy Infrastructure Controller for the Datacenter)
Cisco ACI: Full Stack SDN in Data Center

Centralized Automation and Fabric Management

ACI = CONTROLLER + POLICY MODEL+ NEXUS 9k

- Turnkey integrated solution with security, centralized management, compliance and scale
- L4-L7 Service Graph
- Automated application centric-policy model with embedded security
- Simplify provisioning, operating through relational object-model
- Fully programmable (REST API, Python bindings)
- Broad and deep ecosystem
What is APIC-DC

End Point ➔ End Points Groups ➔ Contracts

Contract specifies rules and policies on groups of physical or virtual end-points without understanding of specific identifiers and regardless of physical location.

End points in group WEB can access end-points in group APP SERVER according to rules specified in the contract.

Contract identifies what traffic
identifies actions applied

filter

action

filter

action

filter

action

... defined bi-directionally in the “provider” centric way
Application Network Profiles

Applying Contracts between Application Tiers

Application Network Profiles are a group of EPGs and the policies that define the communication between them.

Rules specifying communication between application tiers.

Programmable Infrastructure
Cisco APIC Provides Full FCAPS

fault, configuration, accounting, performance, security

Troubleshooting Wizards

Capacity Dashboard

Drag and Drop Configuration

App Health Score
3.3 Virtual Managed Services
(aka Mozart – ESP – DSC - VMS)
Cisco Virtual Managed Services (VMS)
Flexible Service Chains

1. **vIPVPN with FW and RA**
   - vFW with NAT and FW policy.
   - vFW with IPSec/SSL remote access (RA) incl. remote end-host security posture verification.

2. **vIPVPN with BYOD, FW and RA**
   - vFW with NAT and FW policy.
   - vFW with IPSec/SSL remote access incl. remote end-host security posture verification.
   - vISE for BYOD svc auth (AAA, trust-sec label to IP binding)

3. **vIPVPN with BYOD, FW, RA, WebSec**
   - vFW with NAT and FW policy.
   - vFW with IPSec/SSL remote access incl. remote end-host security posture verification.
   - vISE for BYOD svc auth (AAA, trust-sec label to IP binding)
   - vWSA for Enhanced Web Security

4. **vIPVPN with BYOD, FW, RA, EmailSec**
   - vFW with NAT and FW policy.
   - vFW with IPSec/SSL remote access incl. remote end-host security posture verification.
   - vWSA for Critical Information Protection (inbound and outbound Emails)

5. **vIPVPN with BYOD, FW, RA, WebSec, ngIPS**
   - vFW with NAT and FW policy.
   - vFW with IPSec/SSL remote access incl. remote end-host security posture verification.
   - vISE for BYOD svc auth (AAA, trust-sec label to IP binding)
   - vWSA for Enhanced Web Security
   - vNG-IPS (SourceFire) for advanced threat protection and real-time contextual awareness

6. **vIPVPN with BYOD, FW, RA, WebSec, DDoS**
   - vFW with NAT and FW policy.
   - vFW with IPSec/SSL remote access incl. remote end-host security posture verification.
   - vISE for BYOD svc auth (AAA, trust-sec label to IP binding)
   - vWSA for Enhanced Web Security
   - vDDoS (Radware DefensePro) for volumetric and application DDoS visibility and mitigation services
Cisco VMS Service Delivery Workflow

Model Driven Automation
3.4 OSC - Open SDN Controller
CISCO - Open SDN Controller

Network Applications

Application 1  Application 2  Application 3  Application 4  ...  Application ‘n’

REST APIs

DLux User Interface

BASE NETWORK SERVICE FUNCTIONS
- Topology Manager
- Statistics Manager
- FRM
- Host Tracker
- L2 Switch
- AAA Service
- GBP Service

3rd PARTY NETWORK SERVICE FUNCTIONS
- Network Service 1
- Network Service 2
- Network Service 3
- Network Service 4
- ... Network Service ‘n’

Model Driven Service Abstraction Layer
(Plugin Manager, Capability Abstraction, Flow programming, Inventory, etc)

Cisco Open SDN Controller Platform

Data Plane Elements

OpenFlow Interface
OVSDDB Interface
NETCONF Interface
BGPLS Interface
PCEP Interface

OpenFlow Enabled Devices
Open vSwitches
Cisco & 3rd Virtual & Physical Devices

Ciscolive!

wiredel@cisco.com
© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
CISCO - Open SDN Controller
Re-bases XNC on OpenDaylight Helium Release
CISCO - Open SDN Controller
Open SDN Controller vs OpenDaylight Helium

OpenDaylight "Helium" Community Support

OpenContrail Plugin
LISP Flow Mapping
Defense4all
Precluded OpenDaylight Content
VTN Project
SNMP4SDN
PacketCable PCMM
AD-SAL
SDNi

Common Content

Openflow Plugin
OVSDB
Group Policy
L2 Switch
Controller

DLUX
AAA
BGP-LS
MD-SAL
Controller

Basic Clustering
PCEP
Secure Network Bootstrap Infra

Service Function Chaining
Yang Tools

Central Mgmt & Admin
Monitoring
OVA Distribution
One Click Install
Incremental Cisco Value
Plug-in Clustering
Developer Support

Cisco Supported
Logs
Metrics

One Click Install
Incremental Cisco Value
Plug-in Clustering
Developer Support

Ciscolive!

wolfgang@cisco.com
apio-em
BRKCRS-3011
© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
3.5 ODL- Open Day Light Controller
OpenDaylight Platform

Open Source SDN Platform
Lithium, 3rd Release, June 2015

Hydrogen
- Released February 2014

Helium
- Released October 2014
- 1.87M+ lines of code
- 28 Projects
- 256 Contributors

Lithium
- Lithium-SR3
- December 3, 2015
Network Application Life Cycle (Today)

Hop-by-Hop API-Driven Architecture

- Application change
- GUI/API change
- Controller change
- API change
- Feature change
Network Application Life Cycle (Tomorrow)
End-to-End Model-Driven Architecture
3.6 APIC-EM
APIC-EM
Platform Architecture

APIC-EM Applications
- Network PnP
- IWAN
- Path Trace
- Network Inventory
  - Advanced Topology Visualizer

APIC-EM Controller
- Northbound REST APIs

APIC-EM Services
- Inventory Manager
- RBAC
- Policy Analysis
- Policy Programmer
- Topology Services
- Data Access Service
- Network PnP
- IWAN Services

Grapevine
- Elastic Service Infrastructure

Addresses Scale Out and HA Requirements
APIC-EM

Services and Apps

APIC-EM Applications
- Easy QoS Visualizer
- Application Visualizer
- Network Tapping
- Policy Manager
- IWAN App
- Discovery
- Inventory Visualizer
- Topology Visualizer
- Compliance Check
- ACL Visualizer
- ZTD

Northbound REST APIs

APIC-EM Services
- Policy Engine
  - Business Intent to Network Intent Conversion
  - Conflict Detection and Resolution (BI and NI)
- IWAN (PFR, WaaS)
- PXgrid Client + LDAP client
- Radius Proxy + LDAP client
- AD Client + LDAP client
- NetFlow Collector
- Statistics Manager
- QoS Compliance
- DAS

Grapevine
- Policy Programmer
- Network Tapping
- Application Visibility
- ACL Analysis
- Topology
- Network Programmer
- ZTD
- NIB

Network

User Identity Helper Services
- Basic Services
- Policy Creation Services
- Policy Helper Services
- Policy Analysis Services
- Network Information Base
- Legacy Support Services
- IWAN Services

Cisco live!
APIC-EM

High-Availability (HA) Design

- Multiple instances of the GV root across different physical hosts and operating in **Active-Active** mode for optimal performance, load-sharing, and high availability
- Data persistence layer that has instances spread across different physical nodes; provides support for HA and scale
- Non-HA deployment (single/dual hosts):
  - Supports SW failure (APIC-EM services)
  - No support for HW (host) failure
- HA deployment (3 hosts):
  - Supports SW failure (APIC-EM services)
  - Supports HW failure of single host
APIC-EM

Multi-Host Deployment

APIC-EM Cluster

Node 1
IP Addr1

Node 2
IP Addr2

Node 3
IP Addr3

Virtual IP Address

Cisco® Cloud, NTP, DNS, etc.

DNS
NTP

REST APIs and APIC-EM UI

Network Devices

Cisco live!

wolfgang@cisco.com

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
3.7 PI and APIC-EM
Prime Infrastructure and APIC-EM - Today

East-West API and Work Flow Integration
Prime Infrastructure and APIC-EM - Tomorrow

Wolfgang’s view: Common Services with Common northbound App’s and API’s

- System of Record
- Cisco ONE Applications
  - Northbound API’s
  - Restful API’s
- APIC-EM Cluster
  - Distributed, Synchronized, Replicated
- CATALYST
- ISR
- ASR
- WIRELESS
- ENCS
- NFVIS

APIC-EM Cluster

- APIC-EM Cluster
- XML (Publish/Subscribe)
- NIB
- Grapevine Fabric - RabbitMQ
- XML (Publish/Subscribe)
4.1 APIC-EM - Policy Infrastructure
APIC-DC Policy Model

Recap: EPGs and Contract

Contract

Filter
Named collection of L4 port ranges
- HTTP = [TCP], [80, 443]
- MSSQL = [UDP], [1433-1434]
- MySQL = [TCP], [3306, 25565]

ACI Model will be extended for APIC EM Utilization
APIC-EM Policy Construct

Event Triggers

- High Level Business Intent Policies
- Automatically converted to Network Language
- Conflict Detection and Resolution
- Extensible
- Supports different patterns of policies:
  - Access Policies
  - Event – Condition – Action
  - Includes Collections (Ex: a group of userids, a group of applications, etc.)
  - Choose custom tags for policies
  - Choose multiple attributes in each category

Network Users

- User-identifier (tenant/user)
- Application
- Device Type
- Location

Resources

- User-identifier (tenant/user)
- Application
- Device Type
- Location

Actions

- Permit
- Deny
- Copy
- Monitor
- Redirect (L3, L4, L7)
- No copy
- No redirect

Action Properties

- Priority Level
- Resource Level
- Experience Level
- Trust Level
- Destination
- Sample Rate

Policy Properties

- Policy Creator
- Policy Name
- Policy Scope
- Policy Priority
- Policy Time:
  - Start Time
  - End Time
  - Hard timeout
  - Idle timeout
  - recurrence
APIC-EM

Extensions for Enterprise use cases

• Accommodation for Groups
  ✓ Every EP is part of multiple groups in real-life
  ✓ Groups are sometimes overlapping
  ✓ Groups could be defined from multiple context-attributes

• Finer grain access
  ✓ involves combination of consumer EP attributes and producer EP
  ✓ implies overlapping rules. Resolution TBD

• Contract extensions
  ✓ Need to extend contracts to include DPI-based application/groups.
  ✓ Need rich set of actions such as Permit, Monitor, Permit with Warning, etc.
  ✓ Actions include additional rule profiles such as: IPS-profile, File-filter-profile, QOS-profile etc.

• Question about implicit deny:
  ✓ explicit ‘permit’ action
  ✓ explicit ‘deny’ action
APIC-EM
Common Policy Model from Branch to Data Center

POLICY (Common Namespace for Business Intent)

DATA CENTER
Application Intent

WAN AND ACCESS
User Intent

Application Network Flow Profile
SLA, Security, QoS, Load Balancing

User and Things Network Profile
QoS, Security, SLA, Device, Location, Role

CISCO® ADVANTAGE
BROWNFIELD AND GREENFIELD

END TO END

POLICY FRAMEWORK: FOCUS ON APPLICATION AND USER ENABLEMENT
Common End-Points

End 2 End Communication, do we talk?

Hey, I meant from a policy Intent point of view!
4.2 APIC-EM – Auto Scale Architecture
APIC-EM Grapevine

Why do we need a "Platform for Service Elasticity"?

- Distributed service behavior is both unpredictable and dissimilar
- A "one size fits all" approach to service scaling and management lacks the comprehension to manage both, the autonomic and bespoke requirements of a service ecosystem.
- Service groups can be managed by monitoring the container (the virtual machine)
- Events as common as log overflows, memory leaks, and runaway processes will quickly fool any system lacking both service introspection and strong policy into generating all of the classic distributed system failure conditions: storms, flaps, unmanaged contention, and deadlocks.
- Services themselves require support for:
  - specialized policies for scaling in both directions
  - inter-instance communication for building quorum and consensus on scale events
  - unified security for access and authorization
  - unified model and data views for elements managed by multiple services

Remember Cacti – Spine – Poller issues?

output: Time: 42.6984 Method: spine Processes: 8 Threads: 32 Hosts: 79
HostsPerProcess: 10 DataSources: 8985 RRDsProcessed: 2616
APIC-EM Grapevine

What is Grapevine?

- **Grapevine** and **APIC-EM** are de-coupled from a technical perspective. Grapevine is the horizontal scale "platform" on which "services" such as those for APIC-EM run.
- **Cisco groups** wanted to create a new solution XYZ (that was completely unrelated to APIC-EM) that needed scale, HA, rolling-upgrades, service life-cycle management, etc... **could use Grapevine** (as long as they adhere to the Grapevine service design requirements) without needing to deploy/use any of the APIC-EM services.

- Is a PaaS (**Platform as a Service**) with an associated SDK
- SDN developers can use to write their "services" (similar to a Google AppEngine or VMware Cloud Foundry model).
- Is a simplified refinement of the PaaS model provided by both Amazon and Google for their cloud services. While you can run any program you like on their IaaS, using the PaaS requires adherence to a framework.
- The major difference is that Grapevine **introspects at the service level and autoscales at the VM level** rather than breaking scaled resources down to the level of compute, block storage, network, etc.
- It is important to note that Grapevine **controls elasticity at the granularity of "services"** rather than at the more coarse-grained, virtual machine granularity.
- You can run Grapevine on bare-metal as also within VM's or in a mix of physical and virtual machines
- **Advantages of controlling elasticity at the service granularity are:**
  - Avoids VM boot up / shutdown time in the
  - Better determine whether or not a service is indeed healthy and is working as expected vs just knowing whether or not a VM is running or not
  - Better utilize a VM's capacity by running instances of different services within the same VM instance
  - Perform service-specific monitoring to better determine whether an instance is "under heavy load"
As load increases…

…Grapevine spins up more service instances in response…
APIC-EM Grapevine

Grapevine, the 20,000 foot view

- With Grapevine you would define "service bundles".
- Each "service bundle" deployed runs as a separate process.
- Can deploy a single instance of these services or multiple instances of these services, on the same server or across multiple virtual as also physical servers.
- You can add, remove, start, stop, update these services at runtime without downtime.
- Services can be written in pretty much any programming language (Java, C/C++, Go, Python, Ruby, Perl, Tcl, Bash, etc) and would communicate with each other via remote APIs based on HTTP, AMQP, Thrift, etc.
- Given this, you can easily deploy services like OSGi within Grapevine.
- Grapevine will monitor the load of these services.
- Grapevine will provide scale for these services.
  - In the presence of increased load, Grapevine will "grow" multiple instances of the services to provide horizontal scale.
  - In the presence of decreased load, Grapevine will "harvest" service instances.
- Grapevine will provide HA for these services. In the presence of software/hardware failures Grapevine will grow replacement service instances to take over the workload of those instances that have failed.
- Grapevine will provide "rolling upgrades" for these services.
  - You can deploy new services, or updates to existing services to the cloud.
  - Grapevine would periodically poll the cloud for updates and would download and deploy them onto the Grapevine cluster when they're available with minimal to no downtime.
APIC-EM Grapevine

High Availability

When a service fails, Grapevine starts a replacement instance, ensuring service’s “min instance count” requirements are maintained…
Cisco deploys new version of service to the cloud…

… and service catalogs are updated with new version…

GV Appliance or VM

GV Appliance or VM

GV Appliance or VM

APIC-EM
Service Upgrades

Ciscolive!
Grapevine automatically deploys the new version of the service…
### APIC-EM Grapevine

**Grapevine Components: Grapevine**

- **Grapevine Root**
  - Service Manager
  - Capacity Manager
  - Load Monitor
  - Service Catalog

  - Provides on demand capacity to run services...
  - Repository of service bundles that can be deployed on Grapevine nodes...

- **Grapevine Client**
  - Service Monitor
  - Download Manager

  - Downloads and deploys service bundle on Grapevine node...
  - Starts, stops, monitors service instances running on a single Grapevine node...

- Monitors load / health of services across Grapevine...

- Starts, stops, monitors service instances across Grapevine...
APIC-EM Grapevine

Grapevine Components: Services

Public Network
- Load Balancer / Reverse Proxy

Private Network
- SDN Service #1
- SDN Service #2
- SDN Service #3
- ... SDN Service #N

Common Services
- Data Store
- SAL/PAL
- MQ
- Tasks / Events
- AuthN / AuthZ

UI
Applications
APIC-EM Grapevine Services

Grapevine Services Console
**APIC-EM Grapevine**

**Deployment Considerations - System Requirements**

The APIC-EM platform and its hosted applications can run as a virtual appliance when installed on a hypervisor or a bare-metal server. It is also available as a hardware appliance. System resources to run these two different form factors follow.

---

### Physical Appliance Specification:
- **Server**: 64-bit x86 (should be supported by Ubuntu 14.04 LTS)
- **CPU (cores)**: 6
- **CPU speed**: 2.4 GHz
- **RAM**: 64 GB (Single Node), 32 GB (Per Host for Multi-Node)
- **Storage**: 500 GB net
- **RAID level**: level 10
- **Disk I/O speed**: 200 MBps
- **Network adapter**: 1 or more
- **Browser**: Chrome (44.0 or later)
- **Web access required**: Outbound secure web (HTTPS) access from the Cisco APIC-EM to the Internet for automatic updates of the controller software

### Virtual Appliance Requirements:
- **VMware ESXi Version**: 5.1/5.5
- **Server**: 64-bit x86
- **Virtual CPU (vCPU)**: 6
- **CPU speed**: 2.4 GHz
- **RAM**: 64 GB (Single Node), 32 GB (Per Host for Multi-Node)
- **Storage**: 500 net
- **RAID level**: level 10
- **Disk I/O speed**: 200 MBps
- **Network adapter**: 1 or more
- **Browser**: Chrome (44.0 or later)
- **Web access required**: Outbound secure web (HTTPS) access from the Cisco APIC-EM to the Internet for automatic updates of the controller software
4.3 APIC-EM – Grapevine Cloud Deployment
APIC-EM Grapevine

GV Deployment: Elastic Service Management Framework

Mandatory Requirements:
- Easy to adopt
- Low cost of operation
- Cloud-like user experience

Goals:
- Manages mix of physical and virtual machines
- Common solution for physical and virtual
- Balances service instances between containers
- Services set elasticity policies
- Admin sets service priority policy
- Provides introspection of physical capacity
- Provides intelligent service routing to ensure optimal utilization
- Scales automatically into any provided resource
- No operational overhead to user
- Provides high-scale common services - data, queue, security, etc
APIC-EM Grapevine

GV Deployment: Platform Wide-Geo Deployment

LAN-Local Grapevine Network Control

“Admin Role” Grapevine Policy Generation

Metadata, Policy and Reporting Replication

Cloud Platform: Global Reporting, Backup, DR, Conflict and Split-Brain Resolution
APIC-EM

Cloud Connect Support Model

- Modern software uses cloud today
- Controller releases will be incremental (no big releases)
- Partially opt-in and fully auditable
- Core value is seamless, “never-touch-it” upgrade
- Data secured in Cisco cloud
- Single, global reporting system for your networks
- Config, state, and policy backup
- Split-brain resolution
- Push notification to mobile devices
4.4 APIC-EM - Use Cases
APIC-EM Applications

Things we have on our radar….

• Use Case: Path Trace
  One Click Host to Host connection analysis

• Use Case: Traffic Prioritization
  One Click QoS Policy Enforcement (Easy QoS)

• Use Case: Granular Control
  Per User Per Application Access Policy Enforcement

• Use Case: Next Generation Security Management
  Sourcefire and APIC-EM

• Use Case: DDoS Protection:
  Per User Network Traffic Redirection

• Use Case: Traffic Monitoring
  Per User Per Application Network Traffic Tapping

• Use Case: IWAN - Smart Routing
  Automated Provisioning of Routing Paths

• Use Case: Zero Touch Deployment (ZTD)
  Automated Provisioning and Deployment
APIC-EM – Application Slide burst

Fasten your seatbelts
Controller Application - Network Discovery

Add a New Discovery

Use Discovery to scan and find devices in your network and place them in your inventory. When you run Discovery again, APIC-EM will scan the network and update your inventory with any new devices it finds.

DISCOVERY TYPE

Choose from two types of scans: Cisco Discovery Protocol (CDP) or Range (range of IP addresses). For CDP, you enter a single IP address, which CDP uses to begin the process of obtaining information about other directly connected Cisco devices. For Range, you enter beginning and ending IP addresses that APIC-EM scans sequentially beginning with the first IP address and stepping with the ending IP address.

CREDENTIALS

Enter the CLI Credentials used to log into the device. If an Enable Password is used for added security on the devices in your network, enter that password as well.

SNMP

SNMPv2c uses a community-based form of security. The community of SNMP managers that are able to access the agent MIB is defined by an IP address access control list (ACL) and password.

SNMPv3 uses authentication and encryption to ensure SNMP data packet integrity. It provides AuthPriv (authentication based on the HMAC-MD5 or HMAC-SHA algorithms), DES 56-bit encryption in addition to authentication based on the CBC DES (DES-EA) and AES-128 standards, AuthNoPriv (authentication based on the HMAC-MD5 or HMAC-SHA algorithms), and NoAuthNoPriv (uses a username match for authentication).
Network Discovery - Input Parameters

- **Seed IP address for CDP-based network discovery**
- **IP address range for discovery scope - Click on the Add icon to provide multiple IP address ranges**
Network Discovery - Input Parameters

**CLI Credentials**
Credentials are what you use to log in to the devices.

- **Username**: [Input Field]
- **Password**: [Input Field]
- **Enable Password**: [Input Field]

**Advanced**
Specify advanced settings

- **Protocol Order**: Drag and drop the order, you may also deselect any.

**SNMP**
Try different SNMP settings than global ones

- **SNMP v2c**
  - **Read Community**: [Input Field]
  - **Write Community**: [Input Field]

- **SNMP v3**
  - **Username**: [Input Field]
  - **Mode**: [Dropdown]
  - **Auth Type**: [Dropdown]
  - **Auth Password**: [Input Field]
  - **Privacy Type**: [Dropdown]
  - **Privacy Password**: [Input Field]

- **Timeout (in Seconds)**: [Input Field]
- **Retry Count**: [Input Field]

---

SNMP settings and device credentials for collecting network inventory information from the network devices.
Network Discovery - Discovery Status

---

### DISCOVERY DETAILS
- **CDP Level**: 16
- **Protocol Order**: ssh telnet
- **Retry Count**: 3
- **Timeout**: 5
- **Discovery Condition**: Complete
- **IP List**: 207.3.1.1

### DEVICES FOUND IN THIS DISCOVERY
- **Branch-Router1**: 207.3.1.1, Success
- **Branch-Router2**: 207.3.1.2, Success
- **Branch-Access1**: 207.1.10.1, Success

---

**Detailed information about all existing discovery jobs**
### Network Discovery - Northbound REST APIs

#### Inventory

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

**Terms of service**
Cisco DevNet

device-credential : Device Credential API

<table>
<thead>
<tr>
<th>Method</th>
<th>Route</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PUT</strong></td>
<td>/discovery</td>
<td>Updates an existing discovery specified by id - only for starting/stoping the discovery</td>
</tr>
<tr>
<td><strong>DELETE</strong></td>
<td>/discovery</td>
<td>Deletes all discovery</td>
</tr>
<tr>
<td><strong>POST</strong></td>
<td>/discovery</td>
<td>Starts a new discovery process and returns a task-id</td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td>/discovery/count</td>
<td>Returns the number of discovery</td>
</tr>
<tr>
<td><strong>DELETE</strong></td>
<td>/discovery/(id)</td>
<td>Deletes the discovery specified by id</td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td>/discovery/(id)</td>
<td>Returns the discovery specified by id</td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td>/discovery/(id)/network-device</td>
<td>Returns the network devices discovered in the discovery specified by id</td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td>/discovery/(id)/network-device/count</td>
<td>Returns the number of network devices discovered in the discovery specified by id</td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td>/discovery/(id)/network-device/{startIndex}/{recordsToDelete}</td>
<td>Returns the network devices discovered in the given range</td>
</tr>
<tr>
<td><strong>DELETE</strong></td>
<td>/discovery/{startIndex}/{recordsToDelete}</td>
<td>Deletes the discovery in the given range</td>
</tr>
<tr>
<td><strong>GET</strong></td>
<td>/discovery/(startIndex)/{recordsToReturn}</td>
<td>Returns the discovery in the given range</td>
</tr>
</tbody>
</table>

---

**Available APIs**

- File
- Flow Analysis
- IP Geolocation
- IP Pool Manager
- Inventory
- Network Discovery
- Network Plug and Play
- PKI Broker Service
- Policy Administration
- Role Based Access Control
- Scheduler
- Task
- Topology

---

**Cisco**

wolfgang@cisco.com

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
## Controller Applications - Device Inventory

<table>
<thead>
<tr>
<th>Device Name</th>
<th>IP Address</th>
<th>MAC Address</th>
<th>IOS/Firmware</th>
<th>Platform</th>
<th>Serial Number</th>
<th>Config</th>
<th>Device Role</th>
<th>Device Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch-Access1</td>
<td>207.1.10.1</td>
<td>64:08:6a:98:01</td>
<td>12.2(5)SE3</td>
<td>WS-C2960-48LP-S-L</td>
<td>View ACCESS</td>
<td>▼</td>
<td>Switches and Hubs</td>
<td></td>
</tr>
<tr>
<td>Branch-Router1</td>
<td>207.3.1.1</td>
<td>7c:05:0f:3c:0b</td>
<td>15.2(0)W6a</td>
<td>C8S039112K9</td>
<td>View BORDER ROUTER</td>
<td>▼</td>
<td>Routers</td>
<td></td>
</tr>
<tr>
<td>Branch-Router2</td>
<td>207.3.1.2</td>
<td>6706f6b:dc:81</td>
<td>15.2(0)W6a</td>
<td>C8S039112K9</td>
<td>View BORDER ROUTER</td>
<td>▼</td>
<td>Routers</td>
<td></td>
</tr>
<tr>
<td>CAMPUS-Access1</td>
<td>212.1.10.1</td>
<td>0c:29:2b:3c:0b</td>
<td>03.03.00.3E</td>
<td>WS-C6850-48U</td>
<td>View ACCESS</td>
<td>▼</td>
<td>Switches and Hubs</td>
<td></td>
</tr>
<tr>
<td>CAMPUS-Core1</td>
<td>211.1.2.10</td>
<td>0b:03:1b:00</td>
<td>15.1(0)SE3</td>
<td>WS-6633-E</td>
<td>View COPES</td>
<td>▼</td>
<td>Switches and Hubs</td>
<td></td>
</tr>
<tr>
<td>CAMPUS-Core2</td>
<td>211.2.2.1</td>
<td>0b:03:1b:00</td>
<td>15.1(0)SE3</td>
<td>WS-6633-E</td>
<td>View COPES</td>
<td>▼</td>
<td>Switches and Hubs</td>
<td></td>
</tr>
<tr>
<td>CAMPUS-Data1</td>
<td>56.1.1.100</td>
<td>0c:07:7d:55:e7f</td>
<td>03.02.00.XO</td>
<td>WS-C4604-E</td>
<td>View DISTRIBUTION</td>
<td>▼</td>
<td>Switches and Hubs</td>
<td></td>
</tr>
<tr>
<td>CAMPUS-Data2</td>
<td>212.3.1.2</td>
<td>3e:db:7f:3f</td>
<td>03.04.00.SG</td>
<td>WS-C4604-E</td>
<td>View DISTRIBUTION</td>
<td>▼</td>
<td>Switches and Hubs</td>
<td></td>
</tr>
<tr>
<td>CAMPUS-Router1</td>
<td>210.1.1.1</td>
<td>1e:44:0c:02:00</td>
<td>15.4(0)G5</td>
<td>I9R4451-0X09</td>
<td>View BORDER ROUTER</td>
<td>▼</td>
<td>Routers</td>
<td></td>
</tr>
</tbody>
</table>

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
Device Inventory - Hardware Layout

<table>
<thead>
<tr>
<th>Device Name</th>
<th>IP Address</th>
<th>MAC Address</th>
<th>IOS/Firmware</th>
<th>Platform</th>
<th>Serial Number</th>
<th>Config</th>
<th>Device Role</th>
<th>Device Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch-Access1</td>
<td>207.1.10.1</td>
<td>64:a:6d:4:9:bo1</td>
<td>12.2(65)SE3</td>
<td>WS-C2960S-48LPS-L</td>
<td>FOC1537W1ZY</td>
<td>View</td>
<td>ACCESS ▼</td>
<td>Switches and Hubs</td>
</tr>
<tr>
<td>Branch-Router1</td>
<td>207.3.1.1</td>
<td>7:c:e:e:b:3:cd9</td>
<td>15.2.2(M6a)</td>
<td>CISCO2911/K9</td>
<td>FTX1840ALC1</td>
<td>View</td>
<td>BORDER ROUTER ▼</td>
<td>Routers</td>
</tr>
<tr>
<td>Branch-Router2</td>
<td>207.3.1.2</td>
<td>1:07:0:bb:dc:81</td>
<td>15.2(4)M6a</td>
<td>CISCO2911/K9</td>
<td>FTX1840ALBY</td>
<td>View</td>
<td>BORDER ROUTER ▼</td>
<td>Routers</td>
</tr>
<tr>
<td>CAMPUS-Access1</td>
<td>212.1.10.1</td>
<td>1:29:29:3:3:e2</td>
<td>03.00:00:SE</td>
<td>WS-C880G-48U</td>
<td>FOC1703V36B</td>
<td>View</td>
<td>ACCESS ▼</td>
<td>Switches and Hubs</td>
</tr>
</tbody>
</table>

Detailed device inventory information
### Device Inventory - Hardware Layout

**Real-time Device Configuration**

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Config</th>
<th>Device Role</th>
<th>Device Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>4BS2YF</td>
<td>View</td>
<td>ACCESS</td>
<td>Unified AP</td>
</tr>
<tr>
<td>87W1ZY</td>
<td>View</td>
<td>ACCESS</td>
<td>Switches and Hubs</td>
</tr>
<tr>
<td>0ALC1</td>
<td>View</td>
<td>BORDER ROUTER</td>
<td>Routers</td>
</tr>
<tr>
<td>0ALBY</td>
<td>View</td>
<td>BORDER ROUTER</td>
<td>Routers</td>
</tr>
<tr>
<td>0V36B</td>
<td>View</td>
<td>ACCESS</td>
<td>Switches and Hubs</td>
</tr>
</tbody>
</table>

**Building configuration...**

Current configuration : 4641 bytes

- version 12.2
- no service pad
- service timestamps debug datetime msec
- service timestamps log datetime msec
- no service password-encryption

- hostname Branch-Access1
- boot-start-marker
- boot-end-marker
- enable password xxxxxxxx
- username xxxxxx

**Cisco Live!**

wolfgang@cisco.com

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
Device Inventory - Tagging Layout

Sophisticated and automated devices are given a role assignment based on intelligent matching against pre-set templates and attributes.

Geo-site (location) and custom tags for complete flexibility in grouping and classification of devices based on business logic (for example, lines of business, service mix, etc.).
Device Inventory - Status Layout

### Real-time device inventory status

<table>
<thead>
<tr>
<th>Device Name</th>
<th>IP Address</th>
<th>Device Status</th>
<th>Up Time</th>
<th>Last Updated Time</th>
<th>Last Inventory Collection Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP7081.0581.19ca</td>
<td>55.1.1.3</td>
<td>Reachable</td>
<td>NA</td>
<td>a few seconds ago</td>
<td>Managed</td>
</tr>
<tr>
<td>Branch-Access1</td>
<td>207.1.10.1</td>
<td>Reachable</td>
<td>308 days, 17:55:44.99</td>
<td>6 minutes ago</td>
<td>Managed</td>
</tr>
<tr>
<td>Branch-Router1</td>
<td>207.3.1.1</td>
<td>Reachable</td>
<td>308 days, 17:28:08.24</td>
<td>6 minutes ago</td>
<td>Managed</td>
</tr>
<tr>
<td>Branch-Router2</td>
<td>207.3.1.2</td>
<td>Reachable</td>
<td>308 days, 17:26:40.80</td>
<td>7 minutes ago</td>
<td>Managed</td>
</tr>
<tr>
<td>CAMPUS-Access1</td>
<td>212.1.10.1</td>
<td>Reachable</td>
<td>54 days, 4:31:37.69</td>
<td>27 minutes ago</td>
<td>Managed</td>
</tr>
</tbody>
</table>
## Device Inventory - Northbound REST APIs

### Available APIs
- File
- Flex Analytics
- IP Geolocation
- IP Pool Manager
- Inventory
- Network Discovery
- Network Plug and Play
- PIN Broker Service
- Policy Administration
- Role-Based Access Control
- Scheduler
- Task
- Topology

### Inventory

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

**Terms of service**

**Cisco DevNet**

<table>
<thead>
<tr>
<th>API Type</th>
<th>Description</th>
<th>Show/Hide</th>
<th>List Operations</th>
<th>Expand Operations</th>
<th>Raw</th>
</tr>
</thead>
<tbody>
<tr>
<td>device-credential</td>
<td>Device Credential API</td>
<td>Show/Hide</td>
<td>List Operations</td>
<td>Expand Operations</td>
<td>Raw</td>
</tr>
<tr>
<td>discovery</td>
<td>Discovery API</td>
<td>Show/Hide</td>
<td>List Operations</td>
<td>Expand Operations</td>
<td>Raw</td>
</tr>
<tr>
<td>host</td>
<td>host API</td>
<td>Show/Hide</td>
<td>List Operations</td>
<td>Expand Operations</td>
<td>Raw</td>
</tr>
<tr>
<td>interface</td>
<td>Interface API</td>
<td>Show/Hide</td>
<td>List Operations</td>
<td>Expand Operations</td>
<td>Raw</td>
</tr>
<tr>
<td>location</td>
<td>Location API</td>
<td>Show/Hide</td>
<td>List Operations</td>
<td>Expand Operations</td>
<td>Raw</td>
</tr>
<tr>
<td>network-device</td>
<td>network-device API</td>
<td>Show/Hide</td>
<td>List Operations</td>
<td>Expand Operations</td>
<td>Raw</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Route</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/network-device</td>
<td>getNetworkDevice</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/brief</td>
<td>getNetworkDeviceBrief</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/count</td>
<td>getNetworkDeviceCount</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/ip-address/{ipAddress}</td>
<td>getNetworkDeviceIp</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/location</td>
<td>getNetworkDeviceLocation</td>
</tr>
<tr>
<td>POST</td>
<td>/network-device/location</td>
<td>addNetworkDeviceLocation</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/location/{locationId}</td>
<td>getNetworkDeviceByLocationId</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/location/{locationId}/[startIndex]/[recordsToReturn]</td>
<td>getNetworkDeviceByLocationIdByRange</td>
</tr>
<tr>
<td>GET</td>
<td>/network-device/location/{startIndex}/[recordsToReturn]</td>
<td>getNetworkDeviceByLocationByRange</td>
</tr>
</tbody>
</table>
Controller Applications - Host Inventory

<table>
<thead>
<tr>
<th>Host MAC Address</th>
<th>Host IP Address</th>
<th>Host Type</th>
<th>Connected Network Device IP Address</th>
<th>Connected Interface Name</th>
<th>Host Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>30:e4:db:25:75:3f</td>
<td>212.1.20.2</td>
<td>WIRED</td>
<td>212.1.10.1</td>
<td>GigabitEthernet1/0/2</td>
<td></td>
</tr>
<tr>
<td>5c:9d:dd:52:07:78</td>
<td>212.1.10.20</td>
<td>WIRED</td>
<td>212.1.10.1</td>
<td>GigabitEthernet1/0/47</td>
<td></td>
</tr>
<tr>
<td>e8:9a:6f:7a:22:99</td>
<td>207.1.10.20</td>
<td>WIRED</td>
<td>207.1.10.1</td>
<td>GigabitEthernet1/0/47</td>
<td></td>
</tr>
</tbody>
</table>

- **Detailed host information**
- **Network attachment point for host**

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public.
# Host Inventory - Northbound REST APIs

## Inventory

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

**Terms of service**

Cisco DevNet

<table>
<thead>
<tr>
<th>API</th>
<th>ShowHide</th>
<th>List Operations</th>
<th>Expand Operations</th>
<th>Raw</th>
</tr>
</thead>
<tbody>
<tr>
<td>device-credential</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>discovery</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>host</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>interface</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>location</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>network-device</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>network-device-config</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Host API

- **GET** `/host` : Retrieve hosts
- **GET** `/host/count` : Gives total number of hosts
- **GET** `/host/{id}` : Retrieves host based on id

### Device-credential API

### Discovery API

### Interface API

### Location API

### Network-device API

### Network-device-config API

### Tag API
Controller Applications - Topology Visualizer
Topology Visualizer - TAG View

Ability to visualize device TAGs in the topology view
Topology Visualizer - L2/L3 Topology View

Ability to visualize Layer 2/Layer 3 topologies
Topology Visualizer - Saved Layouts

Ability to save multiple topology layouts
Topology Visualizer - Northbound REST APIs

Available APIs
- File
- Flow Analysis
- IP Geolocation
- IP Pool Manager
- Inventory
- Network Discovery
- Network Plug and Play
- PKI Broker Service
- Policy Administration
- Role Based Access Control
- Scheduler
- Task
- Topology

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

Terms of service
Cisco DevNet

topology : Topology Service

GET /topology/custom

POST /topology/custom

GET /topology/l2/{vlanID}

GET /topology/l3/{topologyType}

GET /topology/physical-topology

topology-application-page-view : Topology Application-Page-View Service

vlan : Vlan Service

GET /topology/vlan/vlan-names

GET /vlan/vlan-names
Path Trace
5-Tuple Input

Path Trace

Enter two host IP's (required) and their ports and protocol (optional) to visualize the path.

Host Source IP 65.1.1.6
Host Destination IP 212.1.10.20
Source Port (Optional) 80
Destination Port (Optional) 80
Protocol (Optional) tcp

Trace Results

Please enter the fields above and press Trace to view a path.
Path Trace

Results

Path Trace | Host: 61.1.1.6 → 212.1.10.20 | Source Port: 80 | Destination Port: 80 | Protocol: tcp

| Expand Path | View Small | Show Reverse | Scroll Lock |

Trace Results

- **65.1.1.6**
  - Type: WIRELESS
  - Forwarding Type: Switched

- **AP7061.00F.19ca**
  - Type: accesspoint
  - Forwarding Type: Switched
  - Tunnels: Source CAPWAP

- **CAMPUS-Access1**
  - Type: SWITCH
  - Forwarding Type: Switched
  - Tunnels: Source CAPWAP
  - Ingress Interface: GigabitEthernet1/24
  - Egress Interface: GigabitEthernet2/21

- **CAMPUS-Dist1**
  - Type: SWITCH
Policy Analysis

ACL Analysis

For Your Reference

There are 9 conflicts found in this acl.
Select any line to the left to view more details.

<table>
<thead>
<tr>
<th>Conflict</th>
<th>Type</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DENY</td>
<td>TCP host 140.192.37.20 any eq WWW</td>
</tr>
<tr>
<td>2</td>
<td>PERMIT</td>
<td>TCP any host 161.120.33.40 eq WWW</td>
</tr>
<tr>
<td>3</td>
<td>PERMIT</td>
<td>TCP host 161.120.33.41 host 161.120.33.40 eq WWW</td>
</tr>
<tr>
<td>4</td>
<td>DENY</td>
<td>TCP 140.192.37.1 host 140.111.25.35 eq WWW</td>
</tr>
<tr>
<td>5</td>
<td>DENY</td>
<td>TCP 100.6.3.24 any eq FTP</td>
</tr>
<tr>
<td>6</td>
<td>PERMIT</td>
<td>TCP 140.192.37.0/24 any eq FTP</td>
</tr>
<tr>
<td>7</td>
<td>PERMIT</td>
<td>TCP 140.192.37.0/24 host 161.120.33.40 eq FTP</td>
</tr>
<tr>
<td>8</td>
<td>DENY</td>
<td>TCP 140.192.37.0/24 host 161.120.33.40 eq WWW</td>
</tr>
<tr>
<td>9</td>
<td>DENY</td>
<td>TCP any eq FTP</td>
</tr>
<tr>
<td>10</td>
<td>DENY</td>
<td>TCP any any eq 428</td>
</tr>
<tr>
<td>11</td>
<td>DENY</td>
<td>UDP any any eq 428</td>
</tr>
<tr>
<td>12</td>
<td>PERMIT</td>
<td>UDP any any</td>
</tr>
</tbody>
</table>

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
Policy Analysis
ACL Trace

Boxes greyed out once traffic is blocked for easy visualization

For Your Reference

apple quick time
ports tcp 458 udp 458

A

100.6.2.24

BLD6-Access2
- GigabitEthernet1/0/10 (ingress)
- GigabitEthernet1/0/46 (egress)

BLD6-Dist
- GigabitEthernet1/0/2 (ingress)
  - one_big_acl_for_conflict
    - 10: DENY TCP any eq 458 (blocks tcp 458)
    - 11: DENY UDP any eq 458 (blocks udp 458)
- GigabitEthernet1/0/25 (egress)

EW-CORE1
Easy QoS

Easy customization of policies
Dynamic QoS Classification
5. Demo
• World Of Solutions
• Whisper suite sessions
6. Conclusion and Open Discussion
The World is CHANGING
And it’s CHANGING FAST

In a “share economy” world of “real time” and “co-innovation”, the relationship between supplier and customer is blurring.
“The biggest risk is not taking any risk...

In a world that changing really quickly, the only strategy that is guaranteed to fail is not taking risks.”

Zuck’s
SDN Hard Problems
Some musings on SDN for EN

**Technology**
- Separation of Control and Data Planes:
  - Control Plane Scalability and Resilience
  - State Management: Logically Centralized?
  - State Distribution Trade-offs in SDN
    - Control State Consistency / Application Optimality
    - Application Complexity / Robustness to Inconsistency
  - Combinatorial state explosion: Feasibility, CAP theorem
  - Control Plane Performance:
    \[
    Q = \text{RTT}(\text{switch2packet}) + \text{pps}(\text{switch}) + \text{pps}(\text{controller})
    \]
  - Hybrid Switch Implications
  - Flow Setup Scalability and Performance
  - Topology Discovery and response times
  - CPUs / TCAMs = overlay / underlay = state / Speed
- Abstractions
  - Sweet spot: Leverage ideas from distributed systems,
    - programming languages, and other areas to bridge the gap
    between the centralized controller abstraction and the
    distributed/hierarchical reality
  - “northbound” + “southbound” abstractions
  - Forwarding targets – ASICs and TCAMs
- Policy Controller
- Reasoning Systems, Big Data
- “network as a computer”, network compilers…
- OpenFlow, A Retrospective on Evolving SDN ⇒ MPLS
- OpenStack

**Sociology**
- OF/SDN approach challenges much of our central dogma
- Remember QoS trust boundaries
- Not the least of which are
  - Circuits vs. Hop-by-hop forwarding
  - Centralized / Distributed control planes / “flow-based”
- Operational Models
  - Operational change is quite substantial (ITIL & ITSM)
  - How to you build/operate/debug these networks?
  - Who is in charge of creating a 12-tuple?
  - How to Combine Compute, Storage, Networking and App teams
  - How to translate business intent into policies
  - Convolution of policy and configuration
- A solution looking for a problem
  - Controller – Agent – Troubleshooting / Single BU
  - Have we been unwilling or unable to abstract complexity.
  - Believe network teams do NOT have the skills and experience to implement and manage SDN
  - Influence shift from from NetOps ⇒ DevOps
  - Is it really about NetOps or more about DEV / OPS ?

**Economics**
- Well…all of the above
- RYF-complex (Fragile/Robust)
- Product “de-siloing”
- Does it really become “cheaper”???
Lunch and Learn
LALCRS-0006 - APIC-EM - Thursday 18 February 13:00 – 14:15

During lunch on Tuesday, Wednesday and Thursday, you can join Cisco subject matter experts and your peers in these casual conversations about topics of interest to you.

The Lunch and Learn tables are located in the Catering Area in Hall 4.1.

For a full list of topics on each day, go to: 

http://cs.co/berlin-lal
SDN @ CiscoLive

- Recommended Learning Path on SDN
- 60+ Sessions
  - Technical Seminars
  - Breakout Sessions
  - Hands-on Labs
  - Panel Discussion
- DevNet Zone
- Demos, MTE, Lunch&Learn, Whisper Suites, and more ....

Use Filters in Content Catalog
https://ciscorainfocus.com/scripts/catalog/cleu16.jsp
## Enterprise SDN @ CiscoLive

<table>
<thead>
<tr>
<th>Day</th>
<th>Session</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>Advanced APIC Enterprise Module: SDN Controller for the Campus and Branch</td>
<td>TECSDN-3600</td>
</tr>
<tr>
<td>Monday</td>
<td>Enterprise SDN: Architectures and Key Concepts</td>
<td>TECSDN-2602</td>
</tr>
<tr>
<td>Tuesday</td>
<td>APIC-EM: Controller Workflow and Use Cases</td>
<td>BRKARC-3004</td>
</tr>
<tr>
<td>Tuesday</td>
<td>IWAN management via APIC-EM (SDN Controller)</td>
<td>BRKSDN-2099</td>
</tr>
<tr>
<td>Tuesday</td>
<td>CCIE Skill Transformation to SDN Kungfu Master</td>
<td>BRKSDN-4005</td>
</tr>
<tr>
<td>Wednesday</td>
<td>SDN Enabled QoS-A Deep Dive</td>
<td>BRKSDN-2046</td>
</tr>
<tr>
<td>Wednesday</td>
<td>Hitchhiker’s Guide to Device APIs</td>
<td>BRKSDN-1119</td>
</tr>
<tr>
<td>Wednesday</td>
<td>Containers on routers and switches: Run your apps and tools natively on Cisco boxes</td>
<td>BRKSDN-2116</td>
</tr>
<tr>
<td>Wednesday</td>
<td>Playing With Your Traffic: Exploring Software-Defined Packet Control</td>
<td>BRKSDN-3014</td>
</tr>
<tr>
<td>Wednesday</td>
<td>Cisco Application Policy Infrastructure Controller Enterprise Module (APIC-EM) – Hands on Lab</td>
<td>LTRSDN-1914</td>
</tr>
<tr>
<td>Thursday</td>
<td>APIC-EM: The evolution from traditional management to SDN-led, policy-based automation</td>
<td>BRKNMS-2031</td>
</tr>
<tr>
<td>Thursday</td>
<td>Cisco Open SDN Controller Hands-on Lab</td>
<td>LTRSDN-1913</td>
</tr>
<tr>
<td>Thursday</td>
<td>Deploying Cisco IOS Autonomic Networking Infrastructure</td>
<td>BRKSDN-2047</td>
</tr>
<tr>
<td>Thursday</td>
<td>DNS-AS: Done with SDN and Tired of Dealing with Snowflake Network Complexity? Change the Game with a Simple TXT String!</td>
<td>BRKSDN-3004</td>
</tr>
<tr>
<td>Friday</td>
<td>Solutions Enablement by Cisco Open SDN Controller</td>
<td>BRKSDN-1020</td>
</tr>
</tbody>
</table>

More SDN Sessions in the Recommended Learning Path
Thank you
Some more fun stuff to watch…

- Fundamentals of Cisco APIC-EM
  https://www.youtube.com/watch?v=17IDRT9tuWY

- Metadata-Defined Data Center, Mike Dvorkin, Cisco Systems
  http://techfieldday.com/appearance/introducing-the-next-generation-sddc-leaders-1

- Developing OpenDaylight Apps with MD-SAL
  https://www.youtube.com/watch?v=uBnDJNsd6Qo

- Application Centric Infrastructure (ACI) Overview
  http://www.youtube.com/watch?v=VZWwjNAiUpl

- APIC EM Demo, Apr 2014 - VT Recording
  http://videosharing.cisco.com/p.jsp?i=10394

- CCO: