# Application Programming Interfaces



Ben Piper Author, CCNP Enterprise Certification Study Guide

www.benpiper.com

# Network Automation

### Application programming interfaces (APIs)

Fabric network devices use to communicate with Cisco DNA Center

### **Device programmability**

- Cisco DNAC and vManage NMS use NETCONF to push device configuration to fabric devices
- RESTCONF



# Cisco DNA Center Intent API Python



**Cisco DNA Center Intent API** 

**Python** 

vManage REST API



**Cisco DNA Center Intent API** 

**Python** 

vManage REST API

**NETCONF** 

**YANG** 



**Cisco DNA Center Intent API** 

**Python** 

vManage REST API

**NETCONF** 

**YANG** 

**RESTCONF** 



Cisco DNA Center Intent API

**Python** 

vManage REST API

**NETCONF** 

**YANG** 

**RESTCONF** 

**Embedded Event Manager (EEM)** 



Cisco DNA Center Intent API

**Python** 

vManage REST API

**NETCONF** 

**YANG** 

**RESTCONF** 

**Embedded Event Manager (EEM)** 

Configuration management platforms

# Python



Download and install Python 3 or later

https://www.python.org

Course exercise files

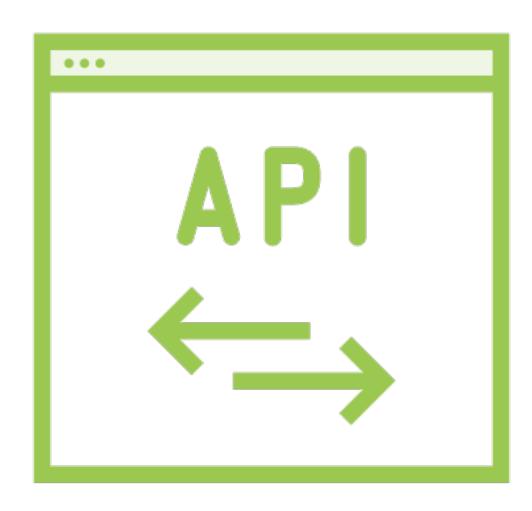
https://github.com/benpiper/ccnp-enterprise

# Cisco DNA Center Intent API

Intent API

Cisco DNA Center web interface communicates with DNAC network controller platform (NCP) using the Intent API

### Intent API



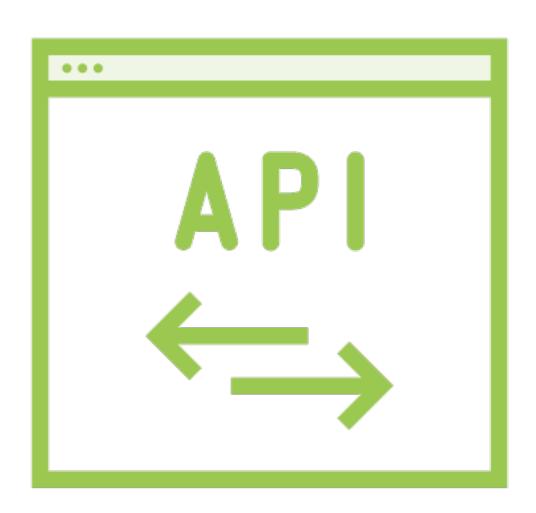
# Uses standard HTTPS actions to send commands and receive data from the NCP

- GET
- POST
- PUT
- DELETE

#### **RESTful API**

- Uses representational state transfer

### Intent API



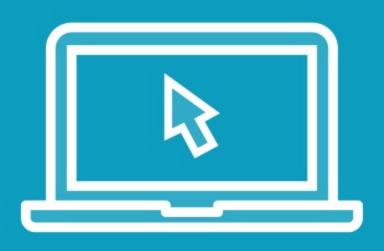
DNAC exposes the API to third-party programs

Northbound API: Controller ↔ program

Southbound API: Controller ↔ devices

# Intent API Authentication

### Demo



**Install Python library** 

**Authenticate to Cisco DNA Center** 

**Enumerate devices** 

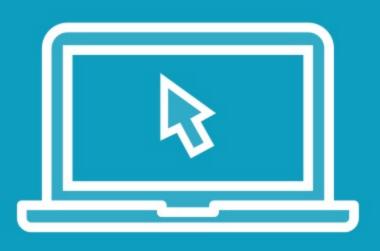
# HTTP Response Codes

# HTTP Response Codes

Code	Status	Uses	
200	Successful GET or PUT	Making requests or pushing configurations	
201	Successful POST	Creating new resources	
204	Successful DELETE	Deleting resources	
30x	Redirect	Redirecting HTTP to HTTPS	
400	Failure	Indicates improperly formatted request	
401	Not authenticated	Indicates invalid token	
403	Forbidden	Indicates unauthorized request	
404	Not found	Indicates incorrect URL	
405	Method not allowed	Indicates wrong verb (e.g. GET instead of POST)	
500	Server failure	Indicates internal server problem	

# vManage REST API

### Demo



Authenticate to vManage controller View OMP routes

# NETCONF, YANG, and RESTCONF

# NETCONF (RFC 6241)



Standardized API for setting and getting configuration and state information

**TCP/830** 

**Uses SSH (not HTTPS)** 

Represents data in XML format

#### netconf.xml

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <edit-config>
       <target>
            <running/>
       </target>
       <config>
            <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
                <interface>
                    <name>GigabitEthernet1
                   <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
                       <address>
                           <ip>10.98.76.54</ip>
                            fix-length>24</prefix-length>
                        </address>
                    </ipv4>
                </interface>
           </interfaces>
       </config>
   </edit-config>
</rpc>
```

### YANG



"Yet Another Next Generation..."

Structured data modeling language for NETCONF

YANG models define valid configuration parameters

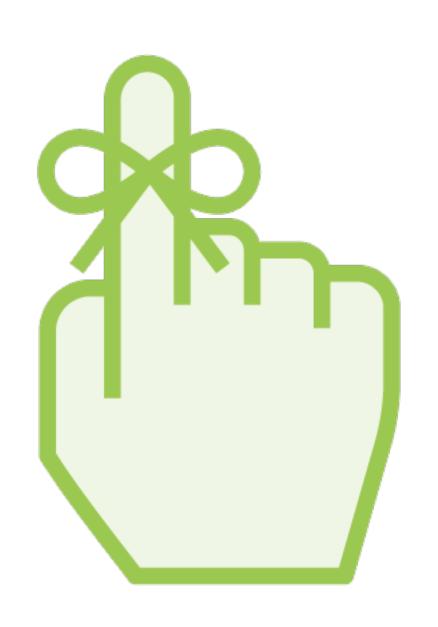
#### netconf.xml

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <edit-config>
       <target>
            <running/>
       </target>
       <config>
            <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
                <interface>
                    <name>GigabitEthernet1
                   <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
                       <address>
                           <ip>10.98.76.54</ip>
                            fix-length>24</prefix-length>
                        </address>
                    </ipv4>
                </interface>
           </interfaces>
       </config>
   </edit-config>
</rpc>
```

#### example.yang

```
list address {
    key "ip";
    description
    "List of IPv4 addresses on the interface";
    leaf ip {
        type inet:ipv4-address-no-zone;
        description
        "IPv4 address on the interface";
    choice subnet {
        mandatory true;
        description
        "Prefix length or netmask";
        leaf prefix-length {
            type uint8 {
                range "0..32";
        description
            "Length of the subnet prefix";
        leaf netmask {
            if-feature ipv4-non-contiguous-netmasks;
            type yang:dotted-quad;
            description
                "Subnet specified as a netmask";
```

### YANG Models for Cisco Devices



https://github.com/yangmodels/yang/tree/master/vendor/cisco

# RESTCONF (RFC 8040)



#### **Uses HTTP verbs**

- GET, POST, PUT, PATCH, DELETE

Configuration data in JSON or XML format

Model represented in the URL

- https://sw1.example.com:9443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet1

#### restconf.json

```
"name": "GigabitEthernet1",
"ipv4": {
    "address": {
        "ip": "10.98.76.54",
        "netmask": "255.255.255.0"
    }
}
```

# Embedded Event Manager

# Embedded Event Manager (EEM)



**Event-driven automation tool** 

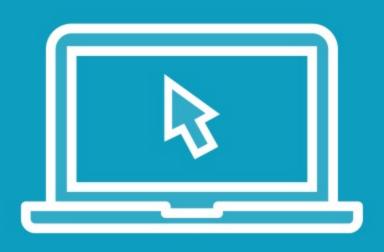
**Uses Tcl scripting language** 

**EEM** scripts are called applets

# EEM Applet Example

```
event manager applet WatchGig0/1
event syslog pattern "Line protocol on Interface GigabitEthernet0/1, changed state to
down" period 1
action 1.0 cli command "enable"
action 2.0 cli command "configure terminal"
action 3.0 cli command "interface gi0/1"
action 4.0 cli command "shut"
action 5.0 cli command "no shut"
exit
```

# Demo



### **Create EEM applet**

- watch-interface.tcl

# Configuration Management Platforms

# Configuration Management

# Infrastructure-as-code (IaC) approach to enforcing consistent configurations

#### Extensible

### Popular platforms

- Ansible
- Chef
- Puppet
- SaltStack

# Agent-based vs. Agentless

**Agent-based** 

Requires an installed agent

**Puppet** 

Chef

SaltStack

**Agentless** 

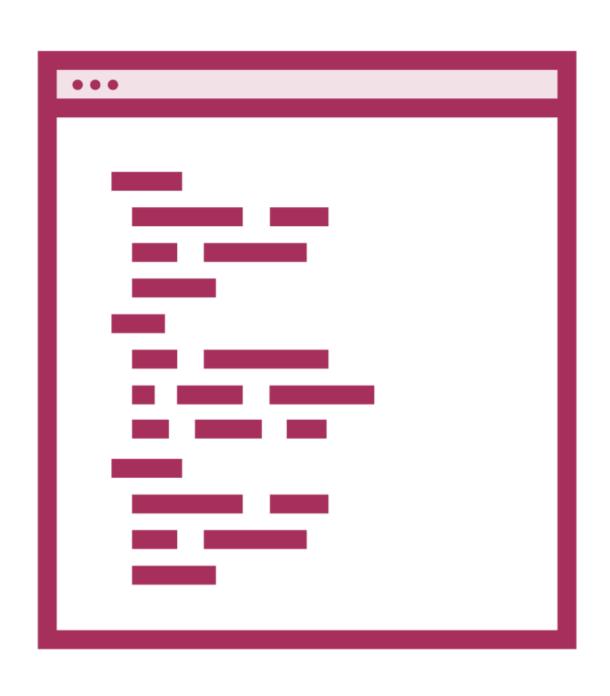
Uses well known protocols (e.g. SSH)

**Puppet Bolt** 

**Ansible** 

SaltStack SSH

# Languages



#### Ruby and YAML are the most common

#### **YAML**

- Resembles YANG
- Easy to read

#### Ruby

- Resembles Python
- Implemented as a domain-specific language (DSL) for simplicity

# YAML vs. Ruby

#### example.yaml

```
---
- name: Enable OSPF
   nclu:
       commands:
       - add ospf router-id {{ rid }}
       - add ospf network {{ prefix }}
       area {{ area }}
       atomic: true
       description: "Enable OSPF"
```

#### example.rb

```
class webserver::apache {
  $apache = $operatingsystem ? {
    centos => 'httpd',
    ubuntu => 'apache2',
  package { $apache:
    ensure => 'installed',
  service { "$apache":
    enable => true,
    ensure => running,
```

# Platform Comparison

Platform	Agent-based or agentless	Language	Configuration repository term
Ansible	Agentless	YAML	Playbook
Chef	Agent-based	Ruby	Recipe
Puppet	Agent-based	Ruby	Manifest
Puppet Bolt	Agentless	Ruby	Manifest
SaltStack	Agent-based	YAML	Formula
SaltStack SSH	Agentless	YAML	Formula



### Application programming interfaces

- Cisco DNA Center Intent API
- vManage REST API
- NETCONF
- RESTCONF



#### **NETCONF**

- Standardized API for setting and getting configuration and state information
- Uses SSH
- Configurations represented in XML
- YANG is the structured modeling language for NETCONF



### **RESTCONF**

- Uses HTTP actions
- Configurations represented in JSON or XML



### Embedded event manager (EEM)

- Event-driven scripts called applets
- Tcl language

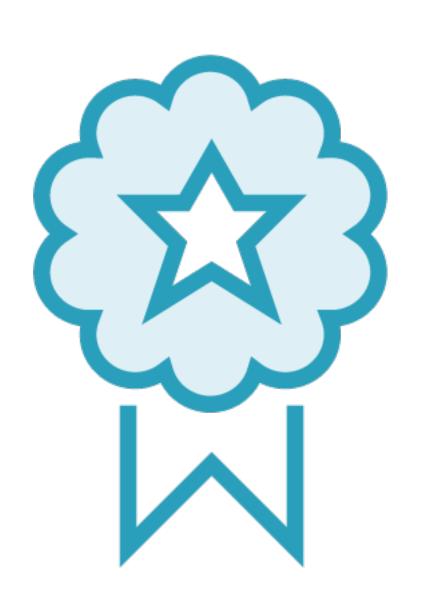


### Configuration management platforms

- Ansible
- Chef
- Puppet
- Puppet Bolt
- SaltStack
- SaltStack SSH

**Agent-based or agentless** 

# Thanks for Watching!



For more study resources, visit <a href="https://benpiper.com/books">https://benpiper.com/books</a>