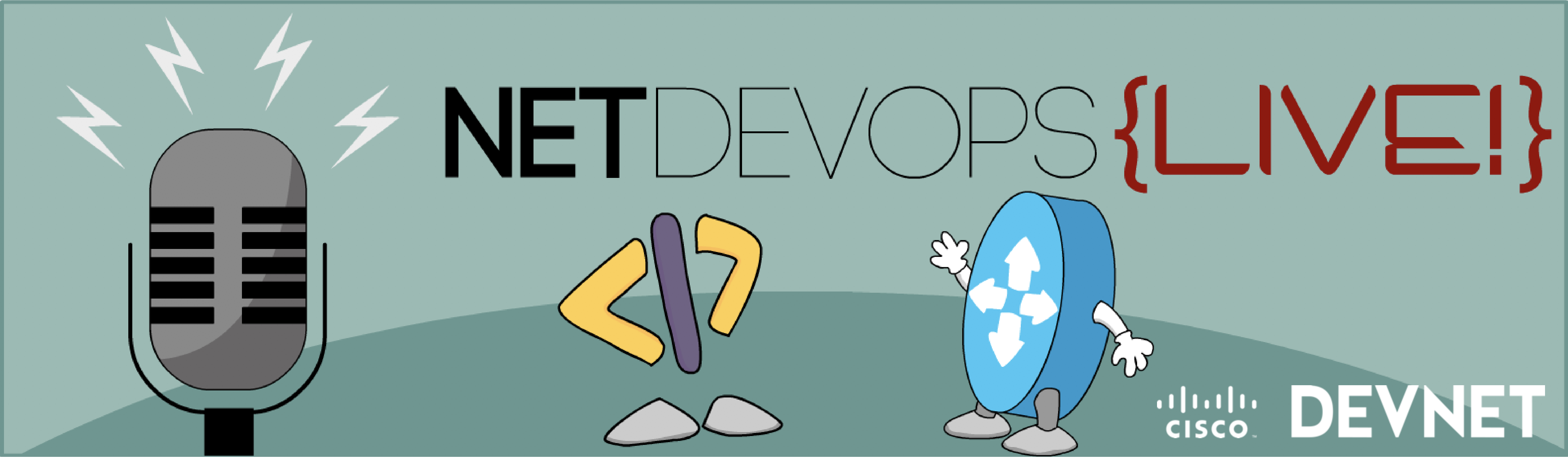


DevOps Style Configuration Management for the Network with Open Source.

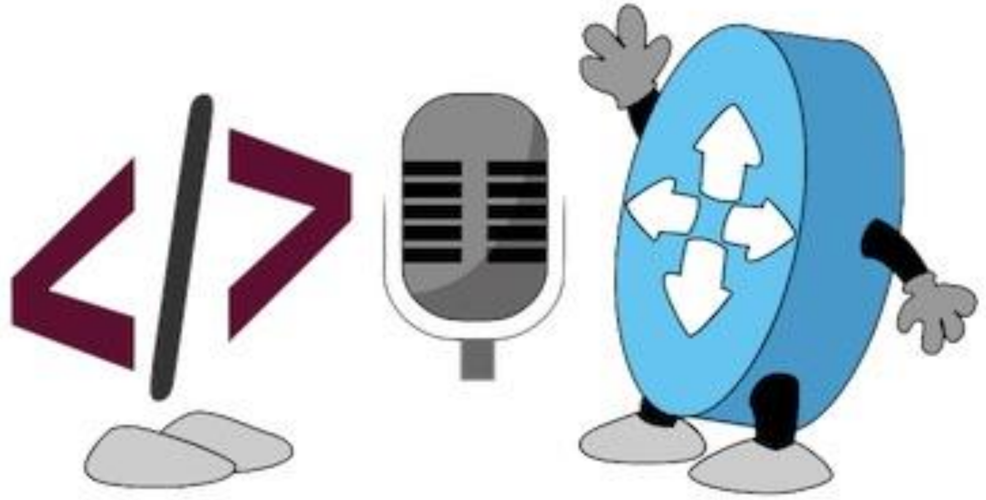
Stuart Clark
Network Automation Evangelist
Twitter: @bigevilbeard

Season 1, Talk 4

https://developer.cisco.com/netdevops/live

What are we going to talk about?

- What are Infrastructure as Code and Configuration Management?
- Benefits of Configuration Management
- Recipes, Manifests, Playbooks, Oh My! The Tools
- Configuration Management with Ansible Example



What are Infrastructure as Code and Configuration Management?

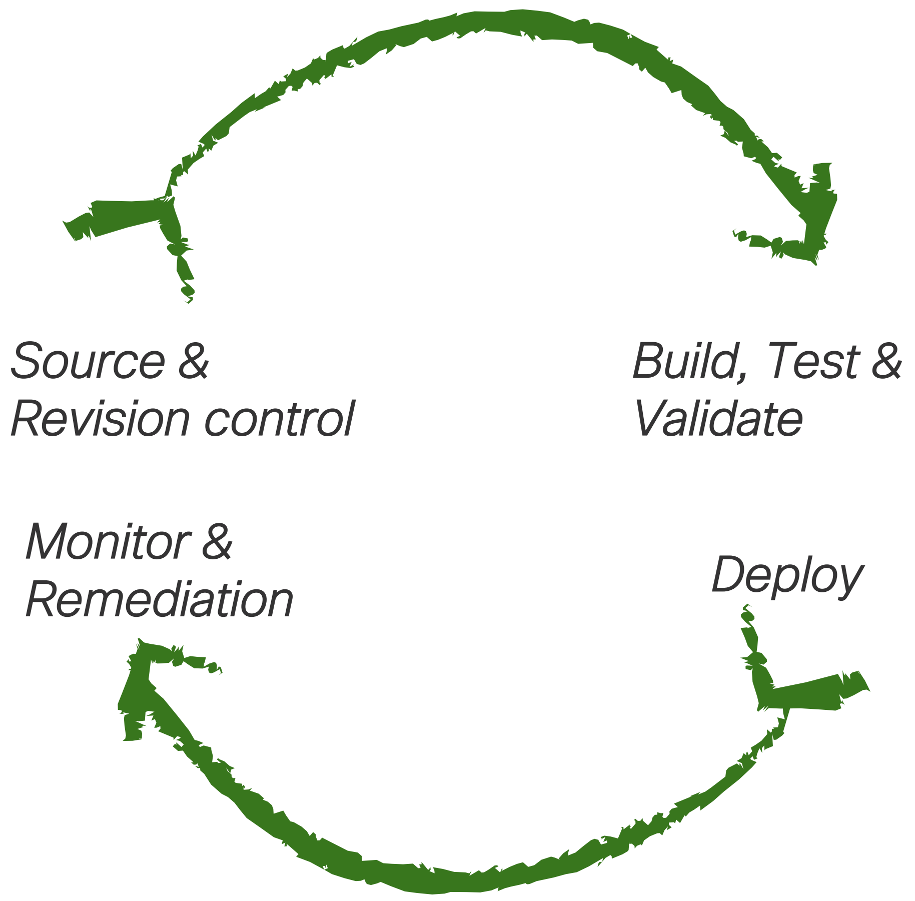
Infrastructure as Code… huh?

"Infrastructure as Code (IaC) is the process of managing and provisioning computer data centers through machine-readable definition files…"

https://en.wikipedia.org/wiki/Infrastructure_as_Code

Some Principals of ”Network as Code”

- Store network configuration in source control systems (ie git)
  - Use ”machine readable” formats like YAML, JSON, XML
- Treat the source control as single source of truth
  - Develop, test, and deploy to prod from same source
- Deploy configuration using programmatic APIs and tooling
  - Limit manual network configuration
  - Explore “Configuration Management” tooling.



Configuration Management: A mechanism for maintaining the characteristics of a system.

A definition…

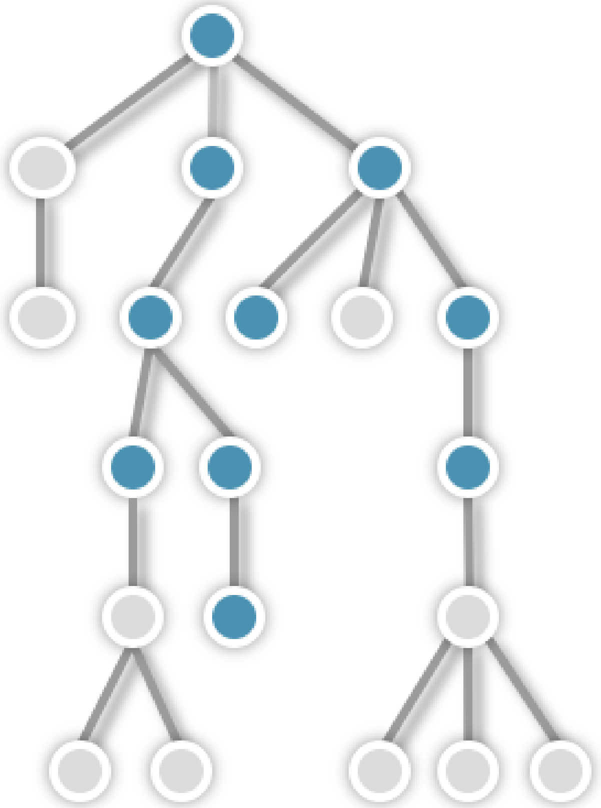
Mechanism = Automation

- No more hand to hand combat configuration management
- Configuration Management today is about the “tools”

Consistency

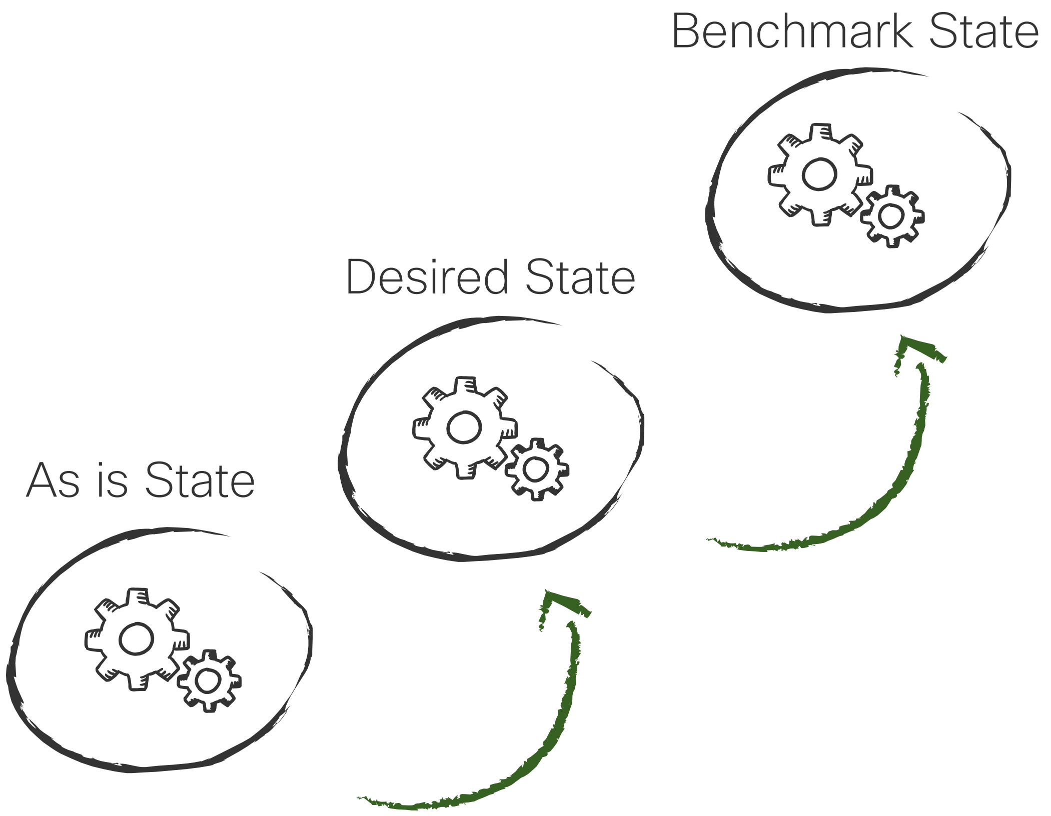
+ Scale

Success!



Characteristics = Desired State

- The software and version installed
- System attributes like name, address, ownership, etc
- Feature specific configurations

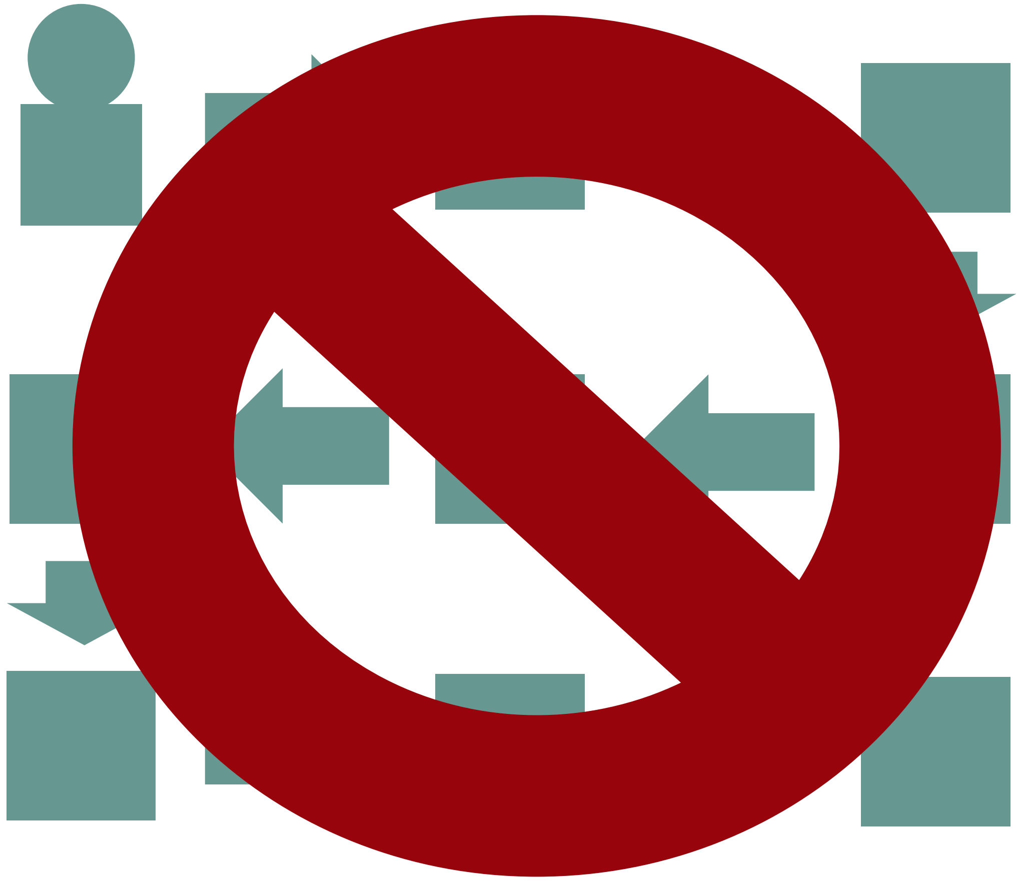


Configuration Management: A mechanism for maintaining the characteristics of a system.

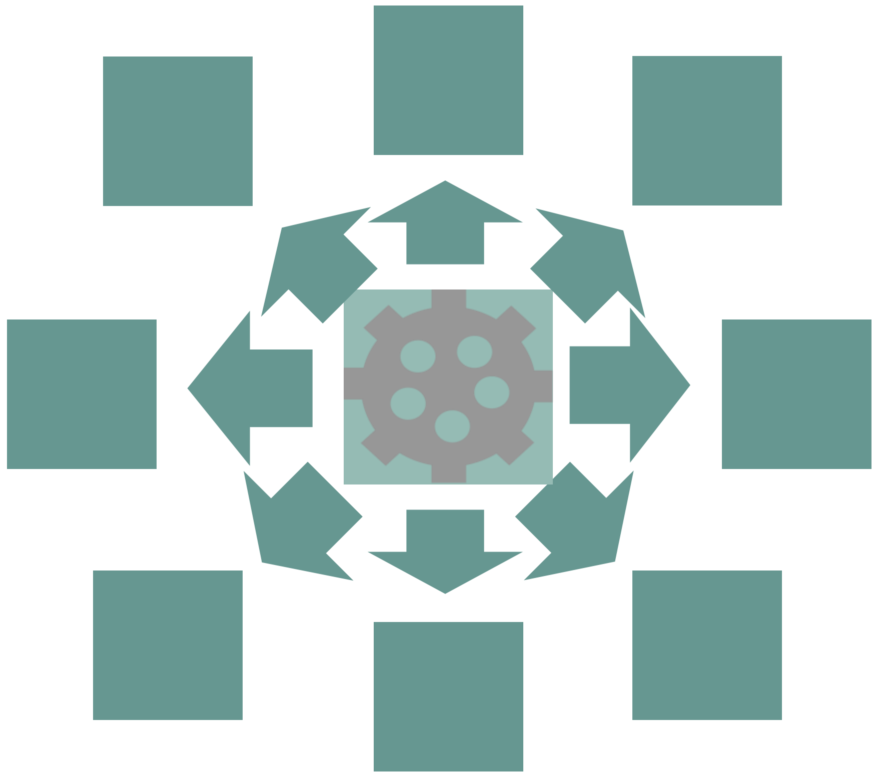
A definition…

Benefits of Configuration Management

Quickly Provision Infrastructure

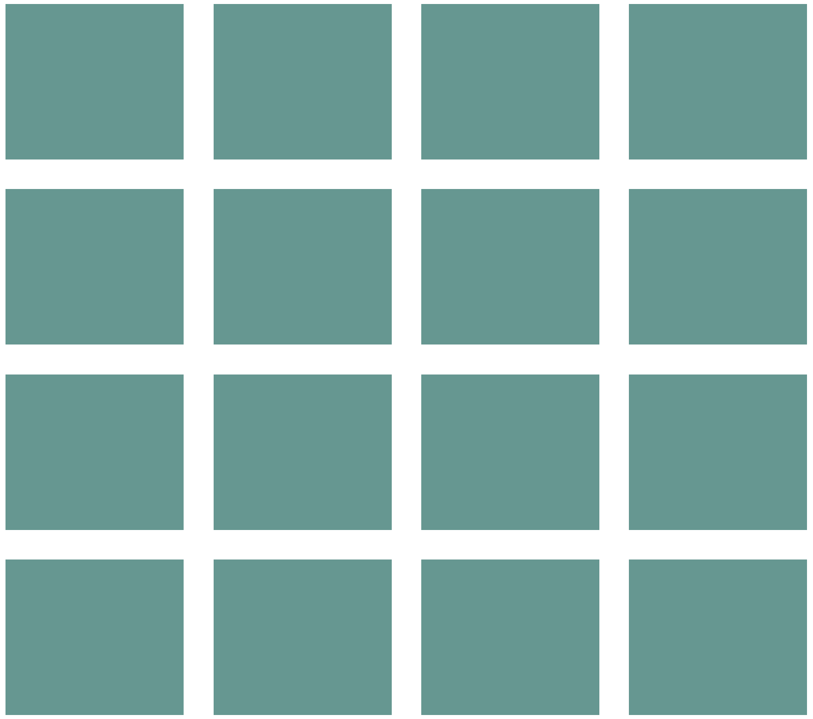


Quickly Provision Infrastructure



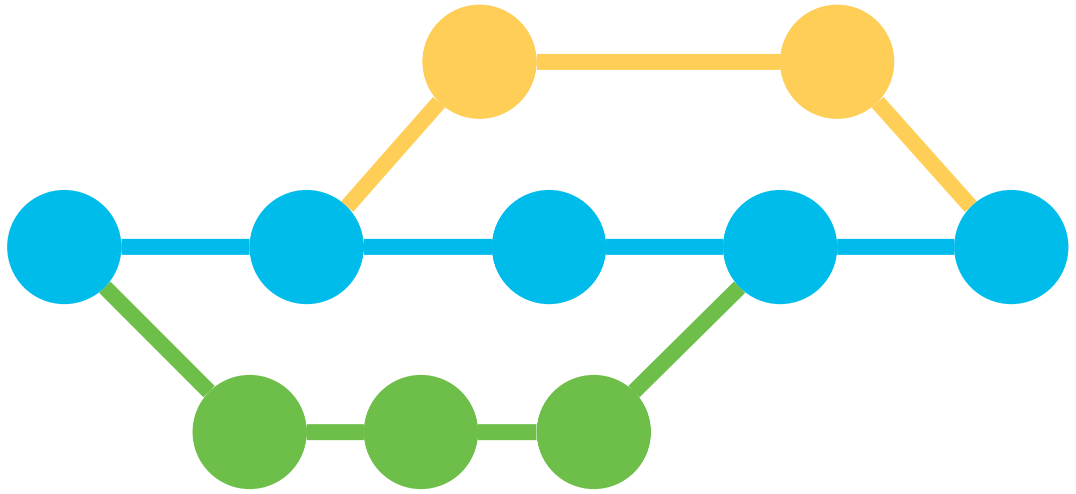
No More Snowflakes

No More Snowflakes



Version Controlled Infrastructure

Version Controlled Infrastructure



Recipes, Manifests, Playbooks, Oh My! The Tools

Commonalities of Configuration Management Tools

- Open Source Foundation
- Automation and Orchestration
- Idempotent Behavior
- Facts, lots of facts
- Modules and Libraries

Ansible

Puppet

Chef

SaltStack

Matrix of Common Info and Terms

	Ansible	Puppet	Chef	SaltStack
Language	Python + YAML	Ruby Based	Ruby	Python
Managed Node Requirements	Agentless	Traditionally Agent Based	Agent Based	Agent Based “minions”
Centralized Management	Any computer can be “controller” *Optional “Tower”*	Puppet Master	Chef Server	Salt Master
What you create	Playbook / Roles	Manifest / Module	Recipe / Cookbook	Pillar / Include

Why Ansible for the Network?

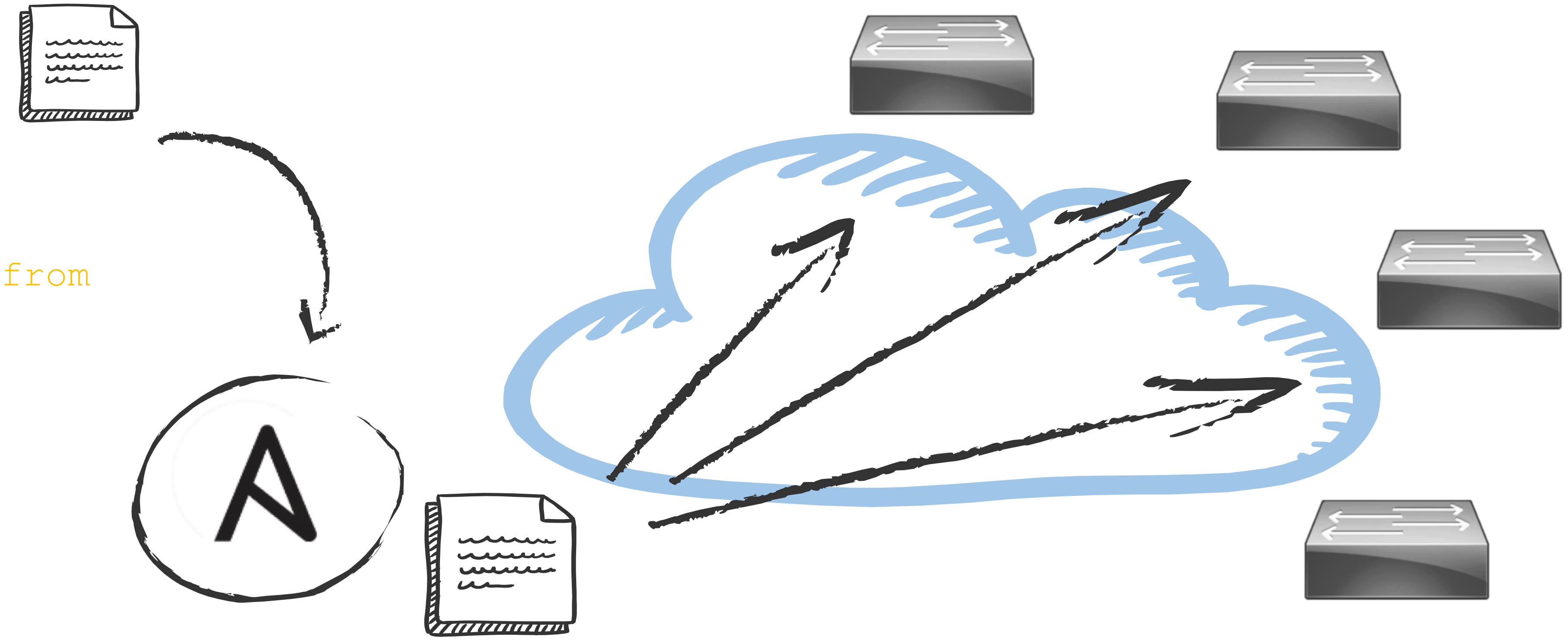
- Agentless
- Currently popular in network community
  - ie Lots of examples!
- Written in Python
- Simple to install and get started!
- But explore other options as well!

Configuration Management with Ansible Example

Ansible and Networking

1. Engineers deploy Ansible playbooks, roles, and modules

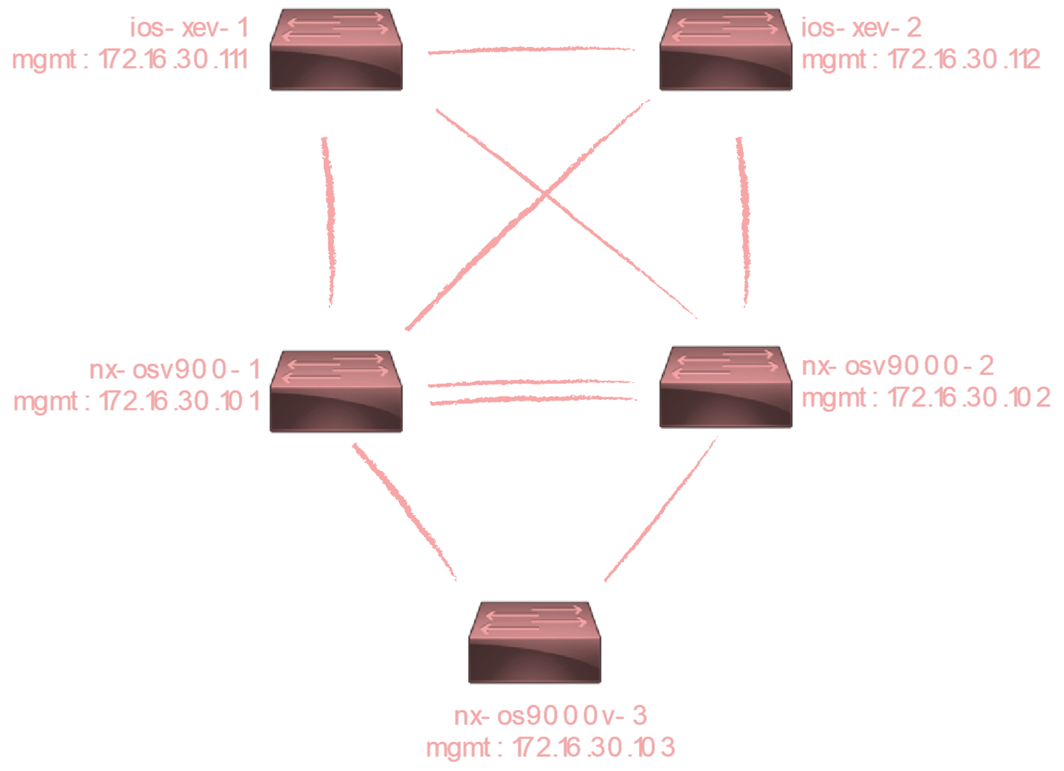
```
---
- name: Retrieve facts from
  switch
    hosts: switches
    connection: local
```



2. Ansible executes modules locally using APIs to interface with devices

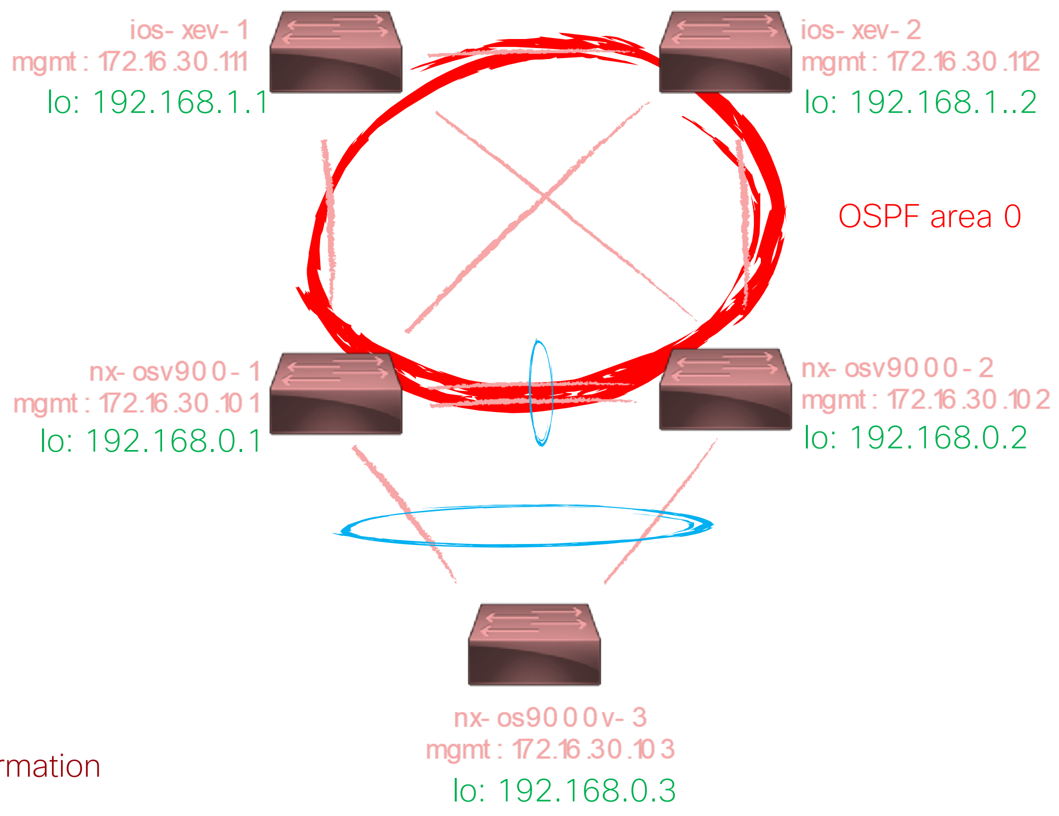
Starting Network Topology

- Physical Topology
  - "Core" – IOS XE Routers
  - "Distribution" – NX-OS Switches
  - "Access" – NX-OS Switches
- Network has been cabled already
- Management access to devices enabled
  - No other configuration completed



Desired Network Configuration

- Layer 3 Links between Core/Dist
  - OSPF Area 0 Routing Configured
- Distribution configured for VPC Domain
- Layer 2 port-channel trunk to access
- Set of VLANs Configured
  - SVIs at Distribution with HSRP Configured



VLAN Information

vlan 100 Management 172.16.100.0/24
vlan 101 Private 172.16.101.0/24
vlan 102 Guest 172.16.102.0/24
vlan 103 Security 172.16.103.0/24

”Network as Code” with Ansible for Configuration Management

- Ansible Playbook
  - Run roles against relevant groups
- Ansible Roles
  - Align to network roles
- Inventory File
  - List network devices
  - Logically group for configuration
- Variable Files
  - Device specific details
  - General group details

Playbook to Define the Orchestration and Intent

- Design the workflow needed to configure the network
- Link inventory devices and groups to particular "roles"
- Order of operation and dependencies for configuration

```
- name: Configure Distribution Switches
  hosts: distribution
  connection: local
  gather_facts: false

  roles:
    - nxos_vlans
    - nxos_vpc
    - nxos_vpc_trunks
    - nxos_l3_interfaces
    - nxos_hsrp
    - nxos_ospf

- name: Configure Access Switches
  hosts: access
  connection: local
  gather_facts: false

  roles:
    - nxos_vlans
    - nxos_po_trunks
```

Content edited for presentation brevity and clarity

Ansible Roles Per Feature

- Reusable roles target specific network configuration
- Different groups will get different roles

```
$ ls roles/

netconf_l3_interfaces < Configure
Interfaces
netconf_ospf          < Configure Routing

nxos_vlans            < Add VLAN
nxos_vpc              < Setup VPC
nxos_vpc_trunks       < Create VPC Trunk
nxos_po_trunks        < Create Po Trunk
nxos_l3_interfaces    < Configure
Interfaces
nxos_hsrp             < Setup HSRP
nxos_ospf             < Configure Routing
```

Content edited for presentation brevity and clarity

Network Inventory

- Groups for core / distribution / access tiers
- Group to identify network operating systems

```
[core]
172.16.30.111
172.16.30.112
[distribution]
172.16.30.101
172.16.30.102

[access]
172.16.30.103

[iosxe:children]
core

[nxos:children]
distribution
access
```

Content edited for presentation brevity and clarity

Configuration Details Maintained in Variable Files

- Separate from the automation and orchestration instructions
- Configuration details
  - VLAN List and Details
  - Layer 3 Interfaces
  - Router Id
  - etc
- Host and group collections possible
- Easily manage network configuration by updating variables
  - Example: Configure additional layer 3 interfaces by adding to list in file

Host Specific Details

```
ospf_router_id: 192.168.0.1
l3_interfaces:
  - interface_type: Loopback0
    description: Default Loopback
    ip_address: 192.168.0.1
    prefix: 32
- interface_type: Ethernet1/5
    description: L3 Link to ios-xev-1
    ip_address: 172.16.0.2
    prefix: 30
- interface_type: vlan100
    description: VLAN Interface - Management
    ip_address: 172.16.100.2
    prefix: 24
```

Group Details

```
vlans:
  - id: 100
    name: Management
  - id: 101
    name: Private
  - id: 102
    name: Guest
```

Content edited for presentation brevity and clarity

Idempotent Network Configuration with Ansible

- Run playbook at anytime to verify configuration still as desired
- Add interfaces, vlans, or networks by updating variables and re-running playbook
- Add new switches (ie access switches) into inventory and re-run playbook
- Only update playbook or roles when features added or changed

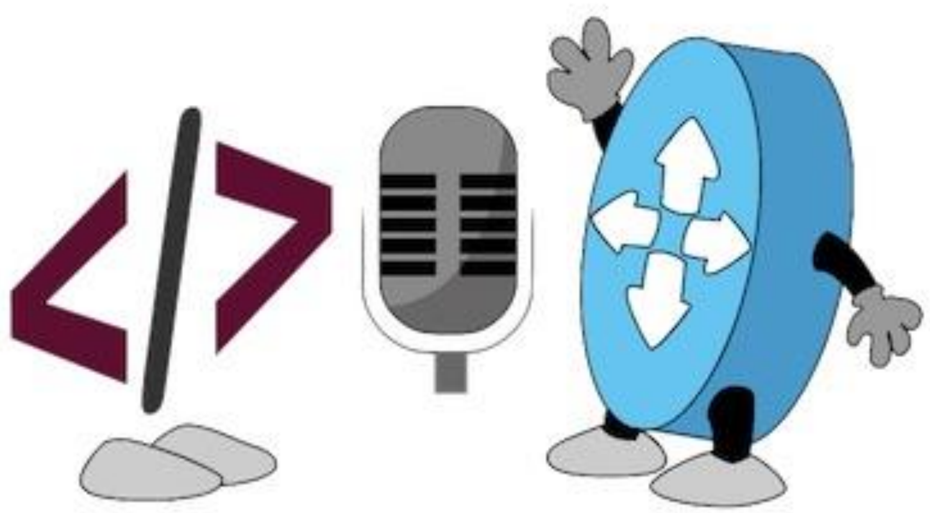
```
PLAY [Configure Distribution Switches] ****************************************************

TASK [nxos_vlans : Configure VLANs] *******************************************************
ok: [172.16.30.102] => (item={'id': 103, 'name': 'Security', 'gateway': '172.20.103.1'})
ok: [172.16.30.101] => (item={'id': 103, 'name': 'Security', 'gateway': '172.20.103.1'})
changed: [172.16.30.102] => (item={'id': 203, 'name': 'Demo3', 'gateway': '172.20.203.1'})
changed: [172.16.30.101] => (item={'id': 203, 'name': 'Demo3', 'gateway': '172.20.203.1'})

PLAY RECAP ********************************************************************************
172.16.30.101              : ok=9    changed=5    unreachable=0    failed=0
172.16.30.102              : ok=9    changed=5    unreachable=0    failed=0
```

Content edited for presentation brevity and clarity

Summary

What did we Talk about?

- What are Infrastructure as Code and Configuration Management?
- Benefits of Configuration Management
- Recipes, Manifests, Playbooks, Oh My! The Tools
- Configuration Management with Ansible Example



Webinar Resource List

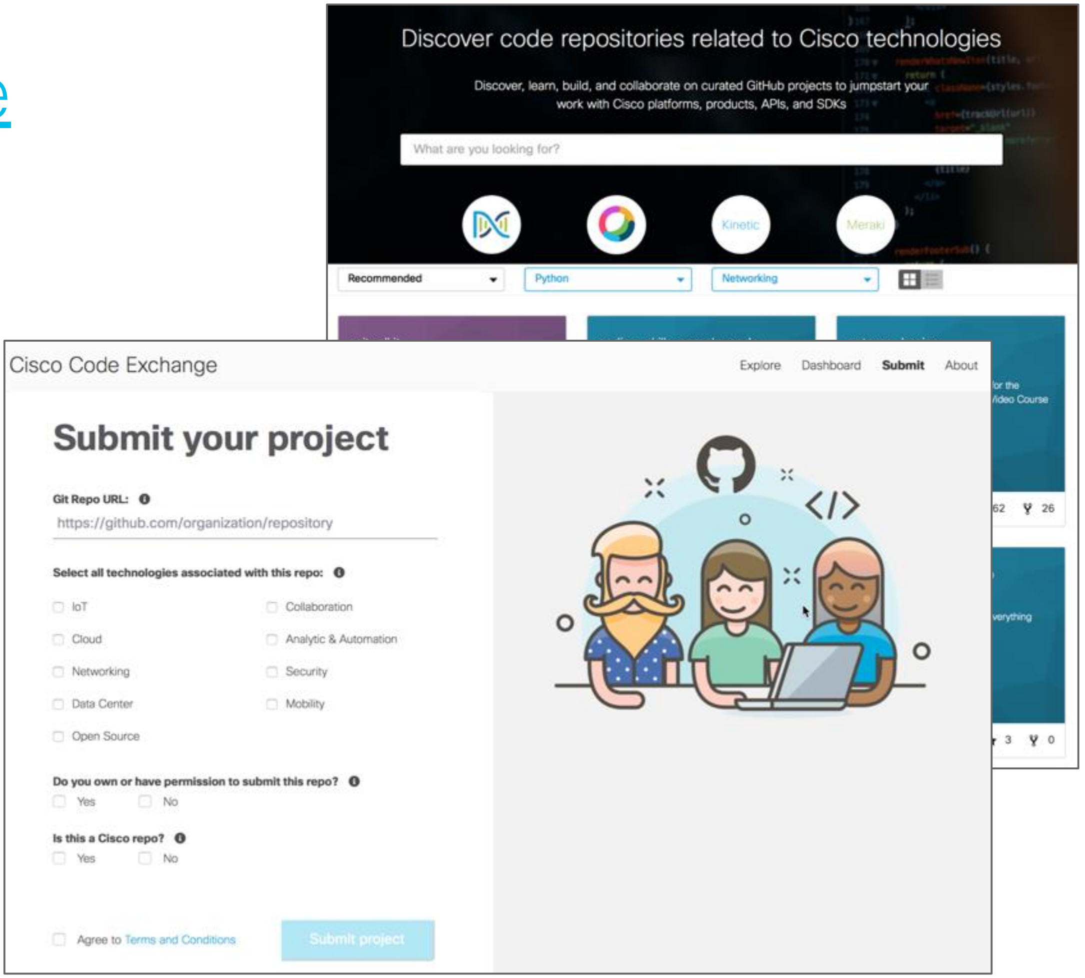
- Docs and Links
  - https://developer.cisco.com/netdevops
- Learning Labs
  - Laptop Setup http://cs.co/lab-dev-setup
  - Introduction to Ansible http://cs.co/lab-intro-ansible
  - Introduction to Ansible for IOS XE Configuration Management http://cs.co/lab-ansible-iosxe
- DevNet Sandboxes
  - IOS Always On http://cs.co/sbx-iosxe
  - NX-OS Always On http://cs.co/sbx-nxos
- Code Samples
  - https://github.com/hpreston/netdevops_demos

NetDevOps Live! Code Exchange Challenge

developer.cisco.com/codeexchange

Create an Ansible Playbook that ensures some network feature is configured as intended.

Example: SNMP, NTP, TACACS, VLANs, Routing



Discover code repositories related to Cisco technologies

Discover, learn, build, and collaborate on curated GitHub projects to jumpstart your work with Cisco platforms, products, APIs, and SDKs

What are you looking for?

Cisco Code Exchange

Submit your project

Git Repo URL:
https://github.com/organization/repository

Select all technologies associated with this repo:

☐ IoT	☐ Collaboration
☐ Cloud	☐ Analytic & Automation
☐ Networking	☐ Security
☐ Data Center	☐ Mobility
☐ Open Source	

Do you own or have permission to submit this repo?
☐ Yes ☐ No

Is this a Cisco repo?
☐ Yes ☐ No

☐ Agree to Terms and Conditions

Looking for more about NetDevOps?

- NetDevOps on DevNet
  developer.cisco.com/netdevops
- NetDevOps Live!
  developer.cisco.com/netdevops/live
- NetDevOps Blogs
  blogs.cisco.com/tag/netdevops
- Network Programmability Basics Video Course
  developer.cisco.com/video/net-prog-basics/

Got more questions? Stay in touch!

Stuart Clark

developer.cisco.com

stuaclar@cisco.com

@bigevilbeard

http://github.com/bigevilbeard

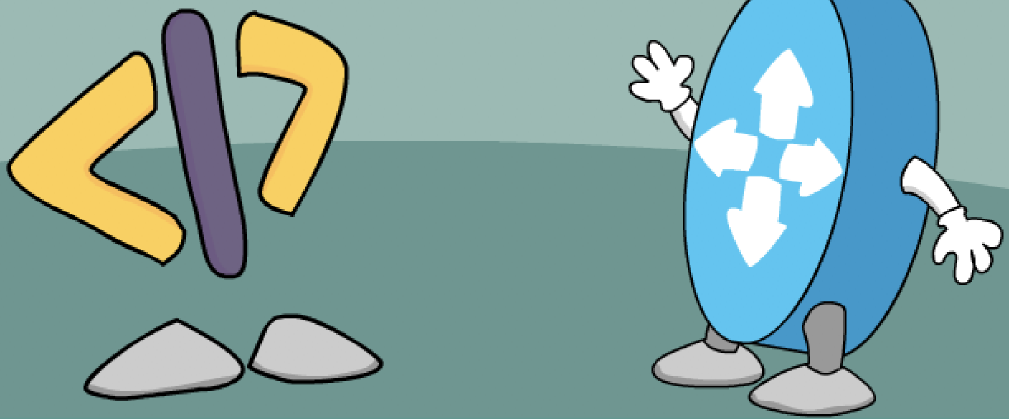
@CiscoDevNet

facebook.com/ciscodevnet/

http://github.com/CiscoDevNet

NETDEVOPS{LIVE!}

CISCO DEVNET



https://developer.cisco.com/netdevops/live

@netdevopslive