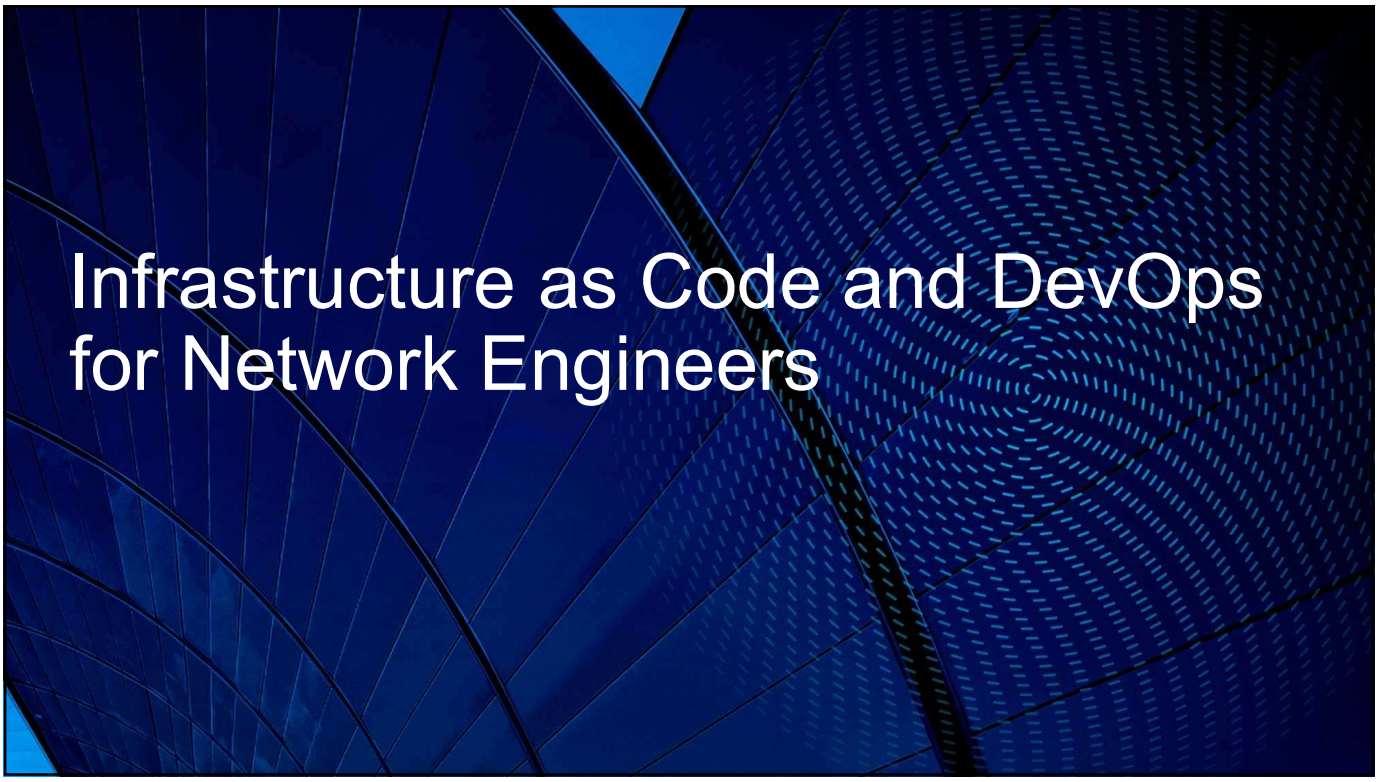


1

Abstract

Many organisations are trying to be more agile in their application delivery and are looking to practices such as “DevOps” and “Infrastructure as Code” to achieve this. But how does the network fit into these new models?

This session will introduce the concepts of DevOps and Infrastructure as Code, as well as the commonly used toolsets. The session will cover examples of how these tools can be used together to create rapid application delivery workflows, and practical use cases in managing a network. Finally, we will cover how these techniques can be applied to the network infrastructure, either on a device-by- device basis, on a network systems basis, or to the entire infrastructure with private cloud management platforms.

2

3

Configuration Management

- Today
  - Canonical configuration on individual devices
  - Many touch points for configuration changes
  - Difficult rollback
- Configuration Management
  - Centrally define desired state/configuration of all devices or subsets of devices

2

4

Declarative vs Imperative

Puppet (declarative)

```
user { 'cgascoig' :
  ensure => present,
  gid => 'admin',
}

group { 'admin' :
  ensure => present,
}
```

Shell script (imperative)

```
#!/bin/bash

if ! getent group sysadmin >/dev/null
then
        echo "Group sysadmin does not exist, creating"
        groupadd sysadmin
fi

if ! getent passwd chris >/dev/null
then
        echo "User chris does not exist, creating"
        useradd --gid sysadmin chris
fi

USERGROUPID=`getent passwd chris | awk -F: '{print $4}'`
USERGROUPNAME=`getent group $USERGROUPID| awk -F:'{print
$1}'`


if [ "$USERGROUPNAME"!= "sysadmin" ]
then
        echo "Primary group of user chris is not
sysadmin, updating"
        usermod --gid sysadmin chris
fi
```

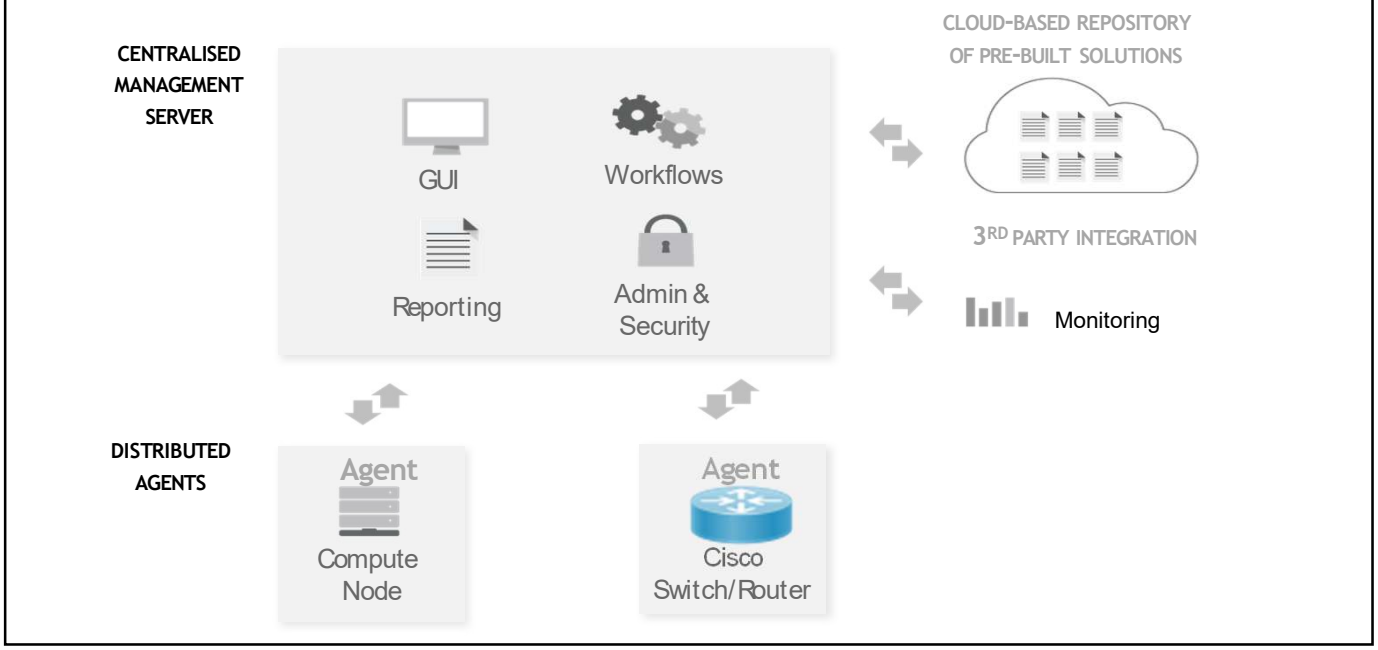
5

Configuration Management Tools


3

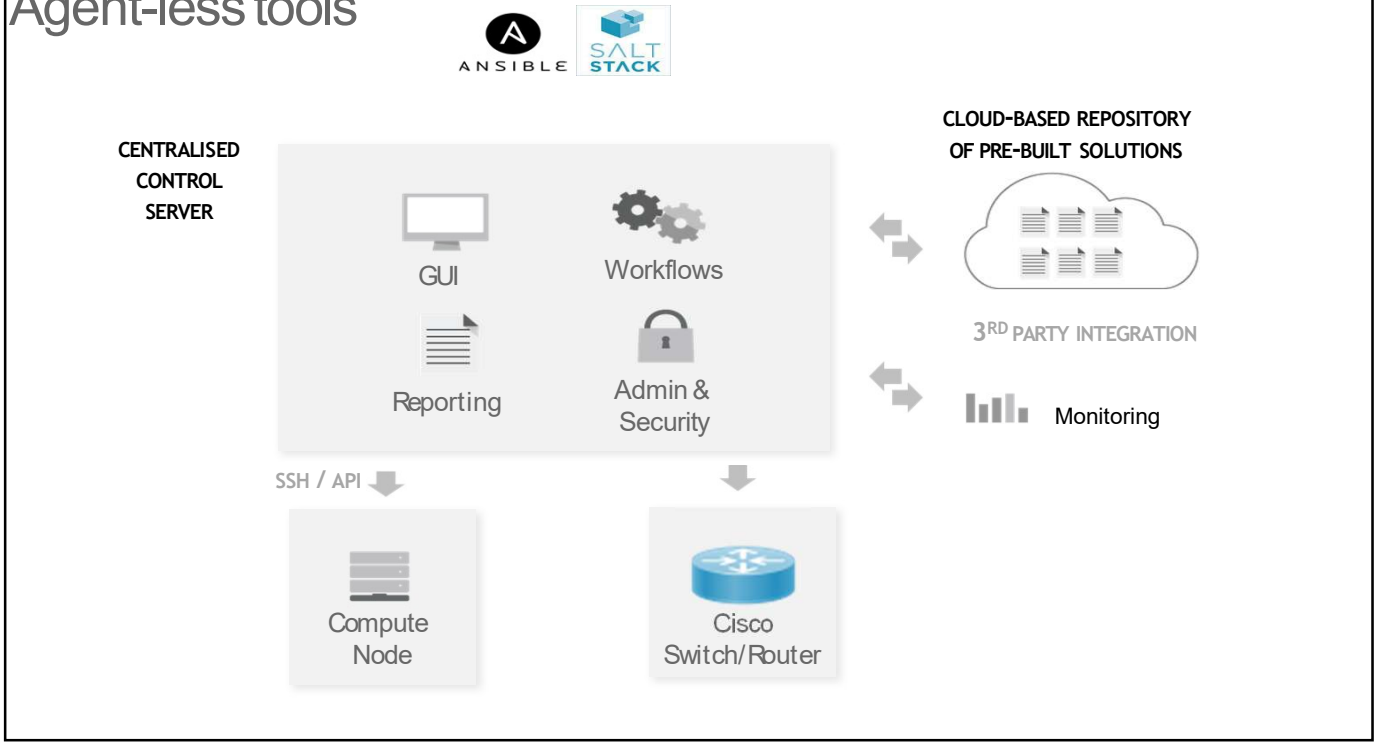
6

Agent-based tools



7

Agent-less tools



8

What is Puppet?

- Next-generation server automation tool
- Declarative language for expressing system configuration
- Client and server for distributing configuration
- Library for realising the configuration.

9

What is Chef?

- Configuration management tool written in Ruby and Erlang
- Pure-Ruby, domain-specific language (DSL) for writing system configuration "recipes"
- Streamlines the task of configuring & maintaining a company's servers
- Integrates with cloud-based platforms such as Rackspace and Amazon EC2 to automatically provision and configure new machines.

5

10

What is Ansible?

- IT automation engine for:
  - cloud provisioning
  - configuration management
  - application deployment
  - intra-service orchestration
- No agents or additional custom security infrastructure
- Very simple language

11

12

Puppet Workflow

1. Define Using Declarative language

Re-usable infrastructure-as-code
Define Desired State of Nodes

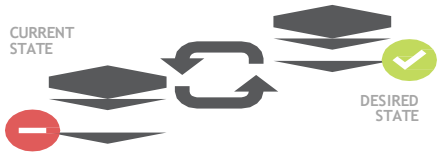
2. Simulate deployment

Before deploying changes, put node into simulation state

4. Report

Insight into changes

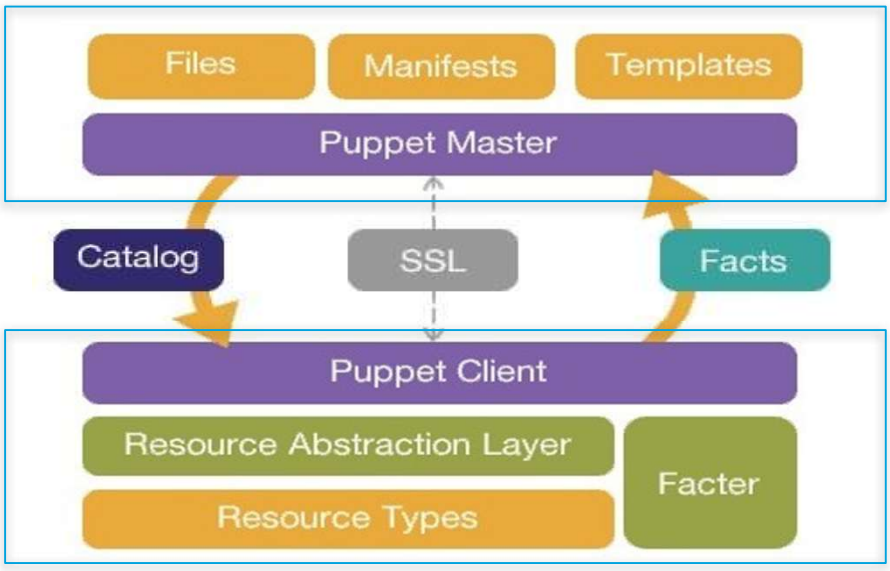
3. Enforce System to desired state



Automatically and reliably

13

Puppet Architecture



7

14

Puppet Pieces (Terminology)

- Puppet Master
  - Central “controller” software which orchestrates configuration deployment for one or more agents. Configuration expressed as a “manifest”.
- Puppet Agent
  - Software which interacts with a single Puppet Master to obtain configuration (desired state) in terms of Puppet Resources. Uses Puppet Resource Providers to carry out tasks to achieve configuration (desired state).
- Puppet Resources
  - Term used for grouping of managed objects/attributes and one or more corresponding implementations of management tasks. The 2 layers of a resource:
    - Resource Type: Definition of managed objects.
    - Resource Provider: Implementation of management tasks on objects.
- Puppet Manifest

  Collection of configuration settings in terms of resource type instances. Often referred to in puppet world as “code”. Manifests are commonly organised in sections that are mostly generic for many nodes (sections apply to specific types of nodes) using conditional logic.
- Facter

  Software which discovers runtime state on an agent node
- Facter Facts

  Runtime state for an agent node. Values can be strings, values, and arrays.

  Facts used as variables in most puppet contexts, ie. in puppet master manifests, resource types, and resource providers.

15

Manifest Sample

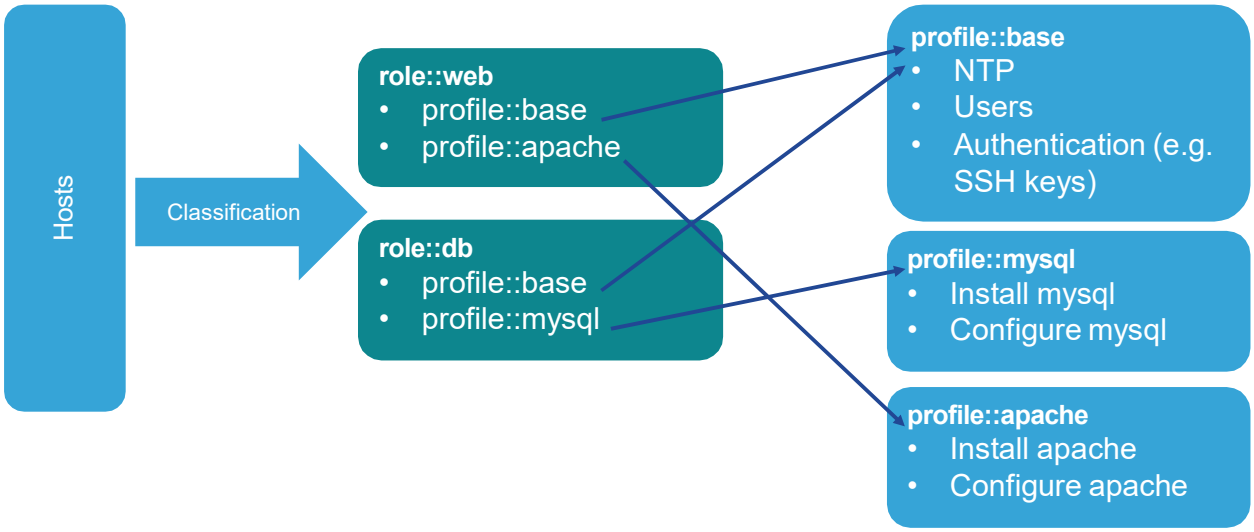
```
class ciscopuppet::demo_ntp {
  ntp_config { 'default':
    source_interface => 'ethernet2/1',
  }

  ntp_server { '5.5.5.5':
    ensure => present,
  }
}
```

8

16

Roles and Profiles Pattern



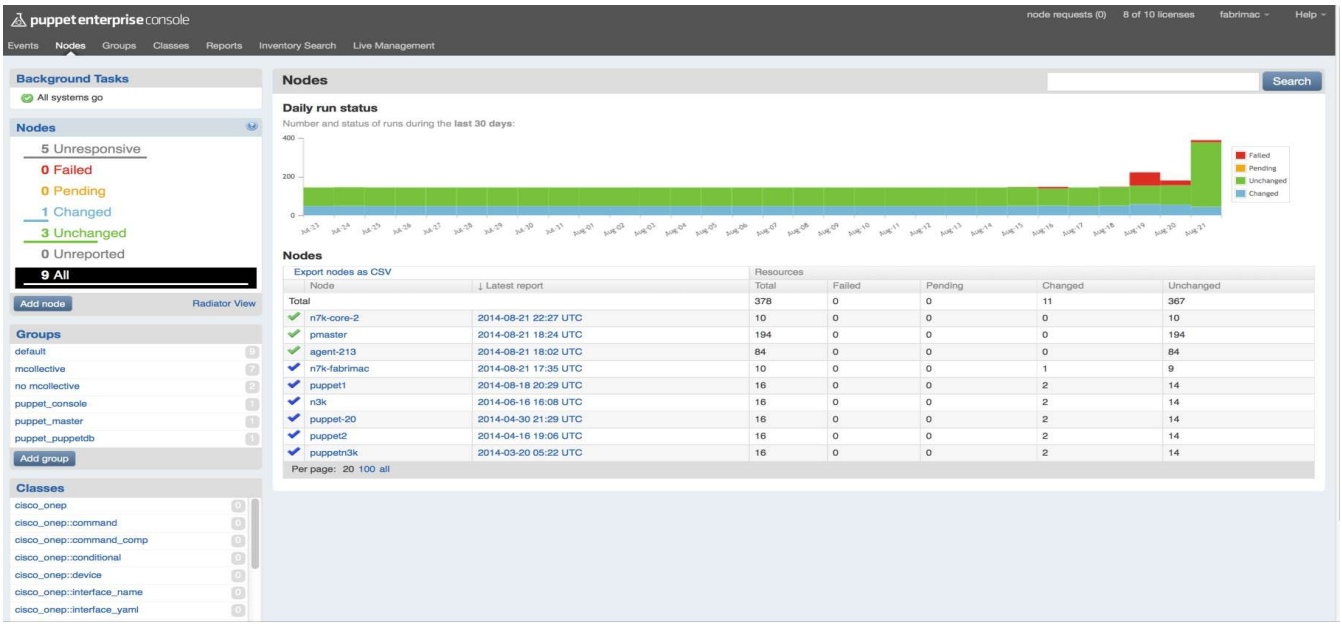
17

Puppet Run Modes

- Oneshot mode
  - Single run of puppet agent to request configuration from puppet master and take action to put resources in desired state.
- No-op mode
  - Single run of puppet agent to request configuration from puppet master BUT DO NOT take action on resources.
    - Used in scenarios where puppet master user wants to validate/inspect the catalogue being compiled for a node and understand the delta with current state.
- Daemon mode
  - Recurring periodic runs of puppet agent requesting configuration from puppet master and take action to put resources in desired state.

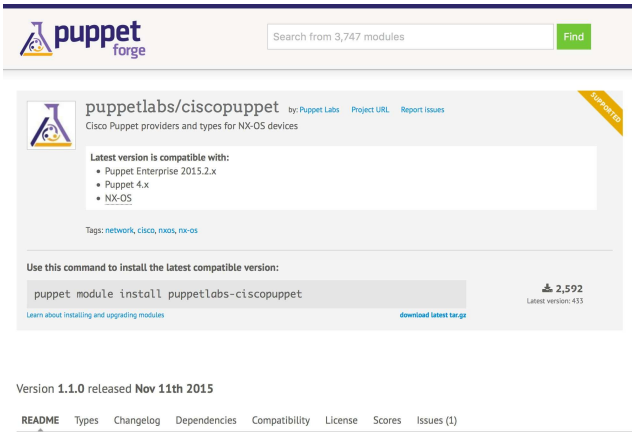
18

Puppet Enterprise Console

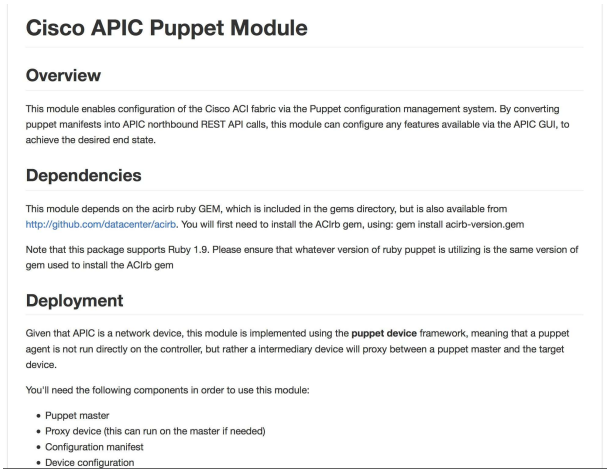


19

Cisco and Puppet

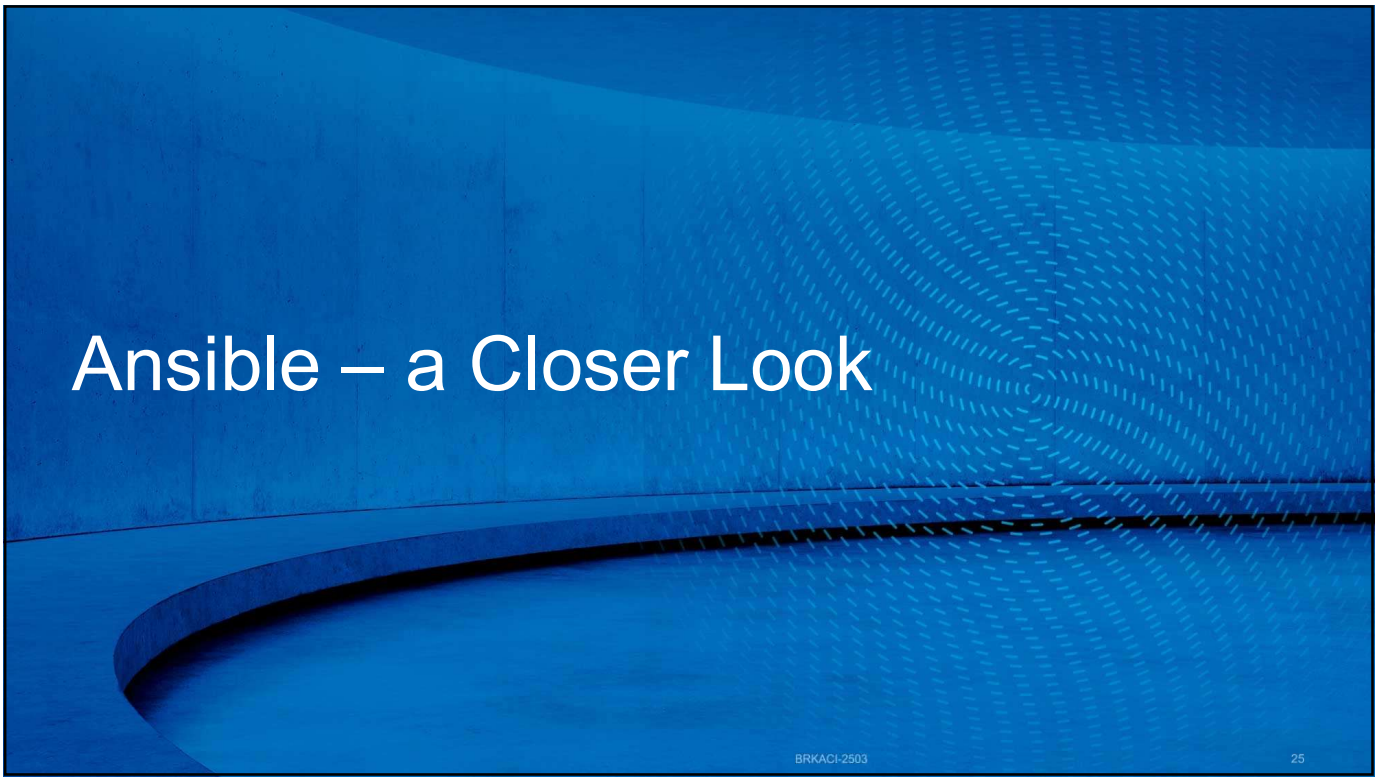


https://forge.puppetlabs.com/puppetlabs/ciscopuppet



https://github.com/datacentre/puppet-aci

20



21

YAML

- YAML Ain't Markup Language (Yet Another Markup Language)
- Data serialisation language
- Similar use-cases as JSON and XML
- Simple syntax

```
---
key1: value1
  key1-1: value1-1
  key1-2: value1-2

list:
  - item1
  - item2
  - item3
```

11

22

Tasks

- Execute module (e.g. yum, service) with specified parameters

```
- name: install Apache
  yum:
    name: httpd
    update_cache: true
    state: latest

- name: ensure Apache is running
  service:
    name: httpd
    state: started
```

23

Playbooks

- Map set of hosts and variables to a set of tasks to be executed

```
- hosts: webservers
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root
  tasks:
  - name: ensure apache is at the latest version
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
    notify:
    - restart apache
  - name: ensure apache is running (and enable it at boot)
    service: name=httpd state=started enabled=yes
  handlers:
    - name: restart apache
      service: name=httpd state=restarted
```


24

25

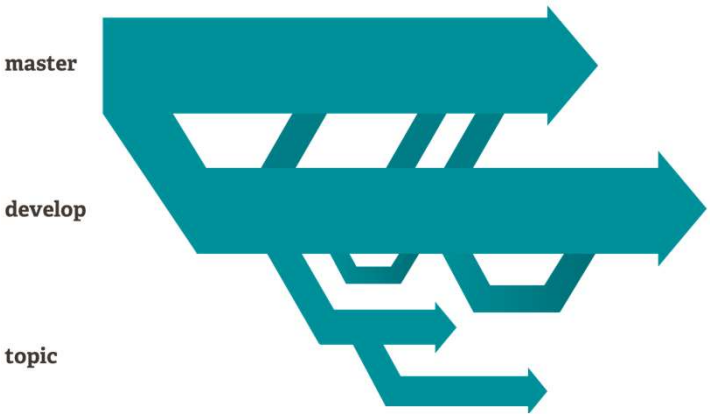
Git

- Originally developed by Linus Torvalds
- Fast
- Distributed Version Control System
- Repositories
- Multiple repositories (e.g. each developer's laptop, central/shared)
- Push/pull changes between repositories

26

Branching

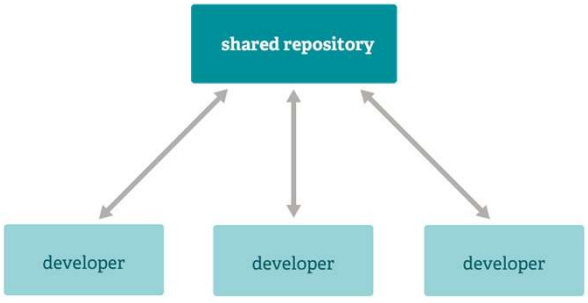
- Context switching
- Disposable experimentation
- Role based codelines (e.g. prod / test / dev)



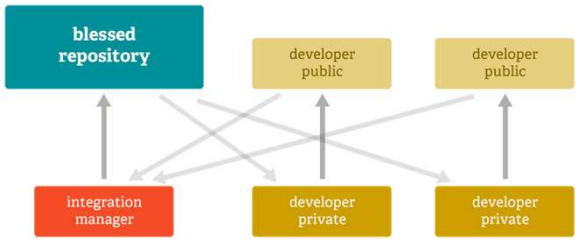
27

Distributed Workflows

Centralised workflow



Integration manager workflow



"Any" workflow…

28

Data Assurance

- Commit identifiers are the cryptographic checksum of the entire project

```
af4ff31 Merge pull request #170 from tomdee/patch-1
8688320 Fix typo
f226b3c Merge pull request #168 from golisai/master
f9d642b Fixed golint error
29194b9 Merge pull request #163 from contiv/demo3
13e0fcb godep import webui
654c35c Merge pull request #165 from unclejack/readme_mkdir_p
8ad910e rebase to master
14f58dd Merge pull request #161 from contiv/demo2
4ad8a1e README: use mkdir -p to create directory
f6a841b Merge pull request #162 from contiv/shaleman-patch-1
a3ddb63 set mtu for ovs internal port too
86a8c9a Update README.md
bfa36c3 script to test policy scale
1ddec0a fix container mtu; fix setup script
4ee8278 godep import latest libopenflow and ofnet
ea87c76 add tcp flags; reverse proxy for containers
8764d52 # This is a combination of 2 commits. # The first commit's message is:
838a9cb Merge pull request #159 from contiv/demo
eb608b8 fix sanity failure
93f4192 Merge branch 'demo' of https://github.com/contiv/netplugin into demo
```

29

Demo

- Initialise a new repository (“git init”)
- Add some files
- Stage and commit (“git add …” / “git commit”)
- History (“git log”)
- Branch (“git checkout –b”)
- Clone an existing repository (“git clone”)

15

30

31

Code Review

- Manage workflows
  - Access Control
- Discuss proposed changes

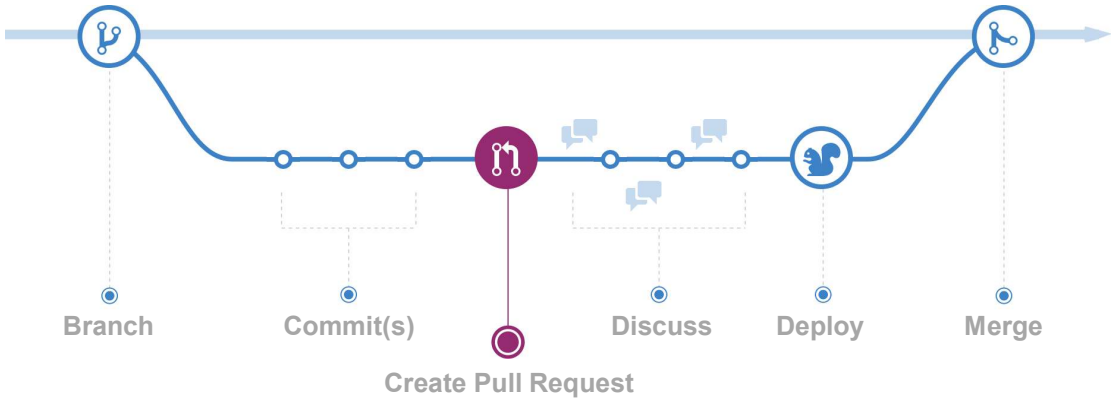
GitLab

16

Atlassian Bitbucket

32

GitHub / GitLab Flow



33

Infrastructure as Code

Continuous Integration / Continuous Delivery

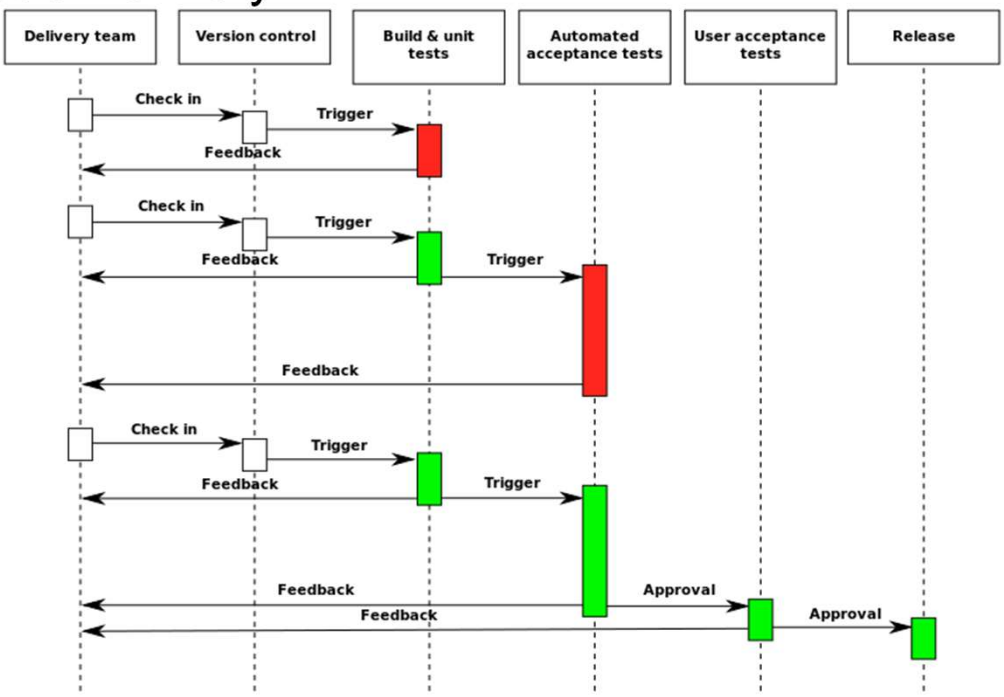
34

Continuous Integration

- Software development practice to automatically run unit and integration tests many times per day
  - Use source code management
  - Automate builds
  - Make builds self testing
  - Keep builds fast
  - Test in a clone of production
  - Publicise test results

35

Continuous Delivery

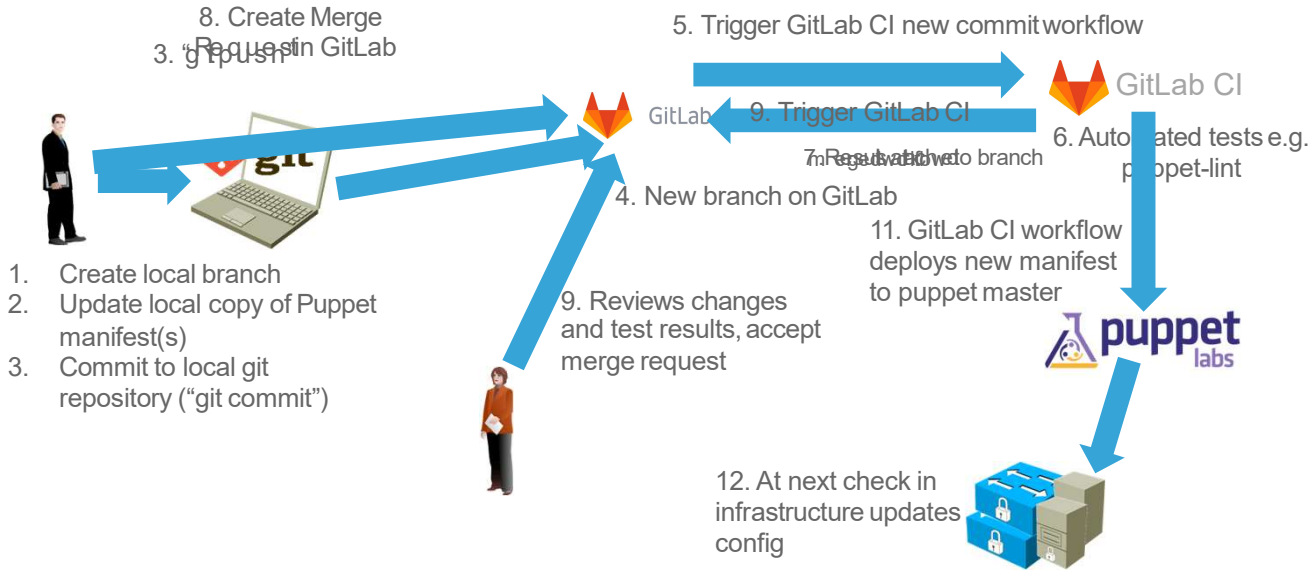


36

Putting It All Together

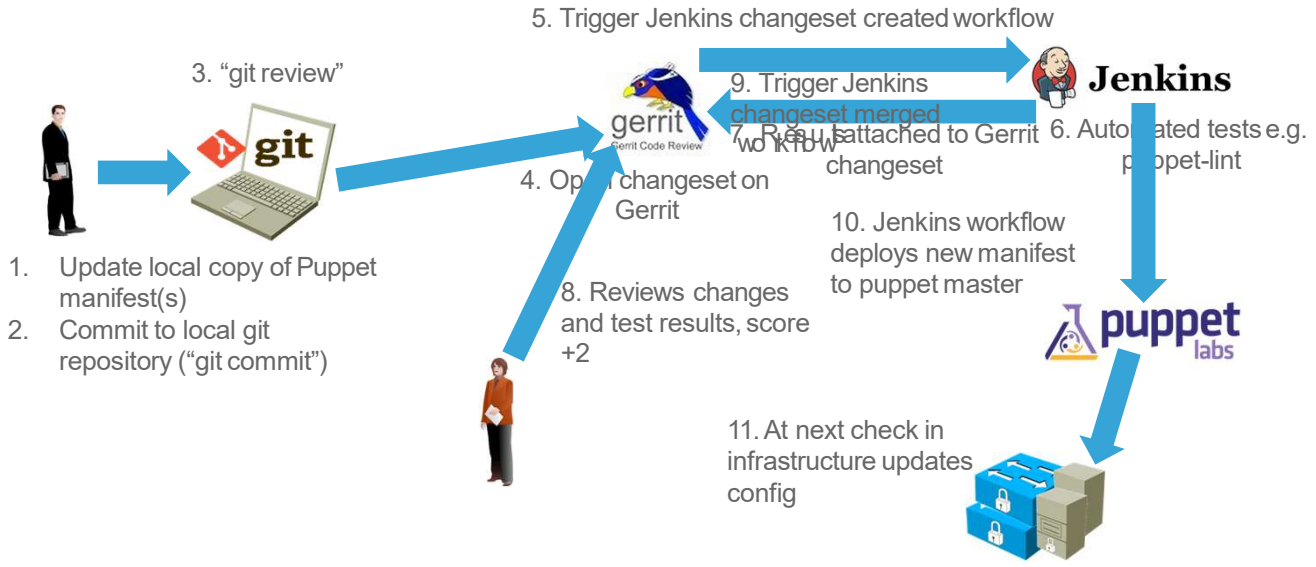
37

Example Workflow



38

Example Workflow

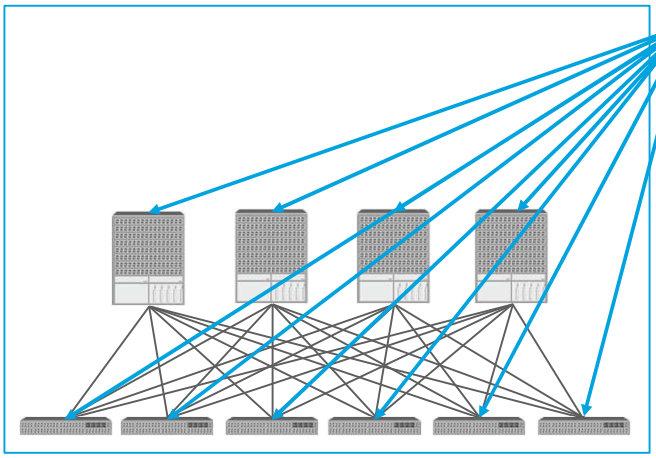


39

20

40

Abstraction

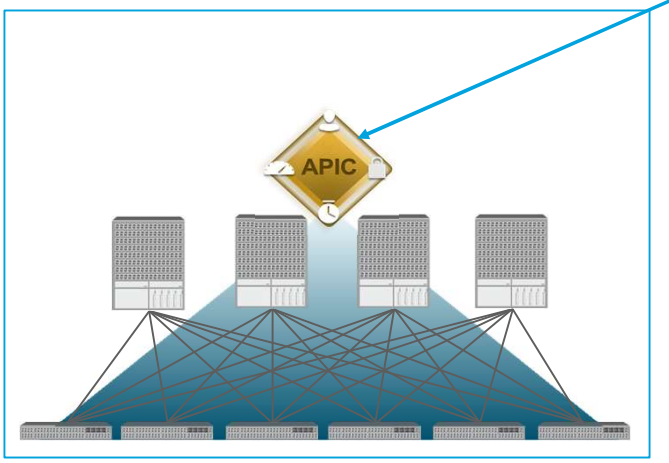


Automation

- Device level configuration
- Topology-dependent
- Network-centric constructs

41

Abstraction

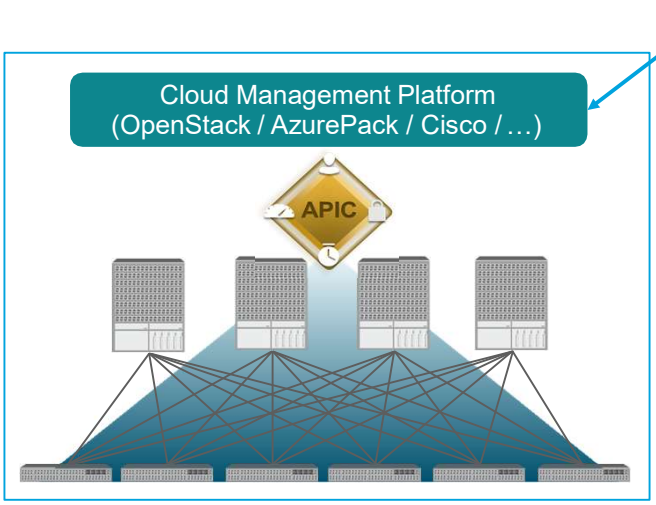


Automation

- Network level configuration
- Topology-independent
- Application- or network-centric constructs

42

Abstraction



- Infrastructure level configuration
- Topology-independent
- Application- or network-centric constructs

43

Conclusion

44

Conclusion

- Start small – use configuration management to ensure baseline configuration consistency (NTP, Syslog, etc)
- Think about who the user will be (network team, systems team, application team) – they will want different abstractions
- Work in other teams tools (e.g. server teams already using puppet)

45

Getting started

- Use NX-OSv (virtual Nexus 9000)
  - Vmware: ESXi or Fusion
  - KVM
  - VirtualBox
  - http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/7-x/nx-osv/configuration/guide/b_NX-OSv_9000.html
- Ansible / Puppet demos:
  - https://github.com/cgascoig/vagrant-nxosv-demos
- Other examples:
  - https://github.com/ndelecro/Nexus-9K-Programmability/tree/master/Ansible

46