Network Automation Tools

Ivan Pepelnjak (ip@ipSpace.net)
Network Architect

ipSpace.net AG

Revision history

2016-09-13 Added information about Puppet support on various data center switches

Who is Ivan Pepelnjak (@ioshints)

Past

- Kernel programmer, network OS and web developer
- Sysadmin, database admin, network engineer, CCIE
- Trainer, course developer, curriculum architect
- Team lead, CTO, business owner

Present

- Network architect, consultant, blogger, webinar and book author

Focus

- Network automation and SDN
- Large-scale data centers, clouds and network virtualization
- Scalable application design
- Core IP routing/MPLS, IPv6, VPN

CISCO CCIE EMERITUS

vmware vEXPERT

Traditional Network Automation Tools

SSH-based API

Configuration management

IPAM and configuration templates

What Others Are Using

Configuration/state management tools

- Puppet
- Chef
- Salt

Automation framework

- Ansible

Source code control tools

- Git
- Subversion (SVN)
- RCS, CVS, SCCS

Reviews

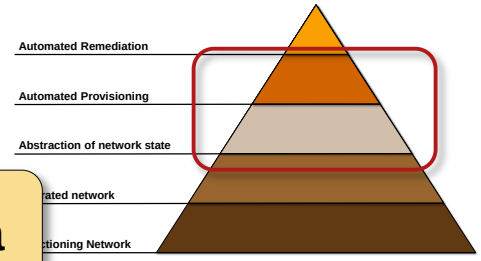
- Gerrit

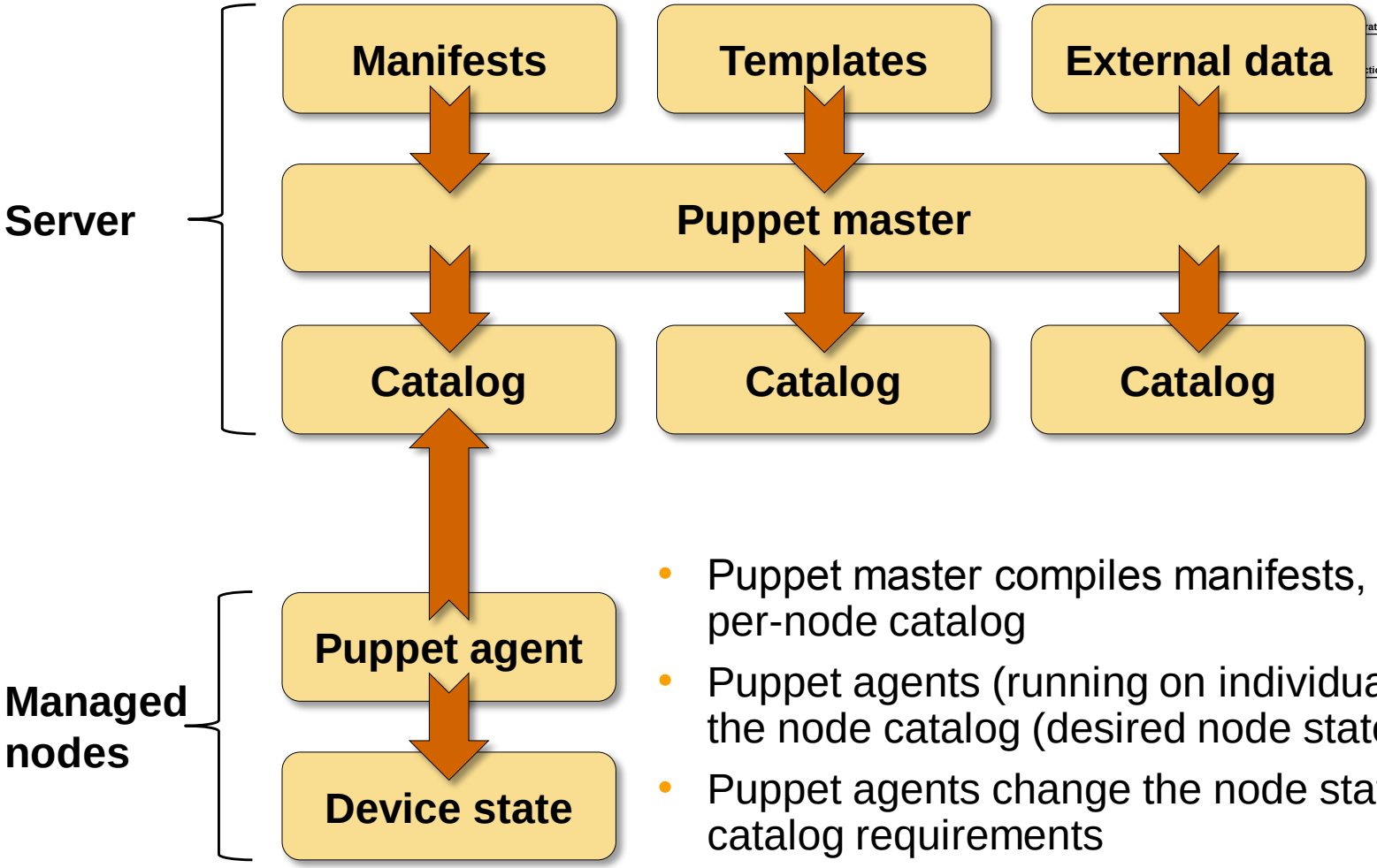
Continuous integration

- Jenkins

Configuration and State Management

Puppet and Chef





- Puppet master compiles manifests, templates, … into per-node catalog
- Puppet agents (running on individual nodes) pull down the node catalog (desired node state)
- Puppet agents change the node state to match the catalog requirements
- Changes to node state are *idempotent*

Puppet and Chef on Network Devices

Puppet/Chef agent must be running on the managed node

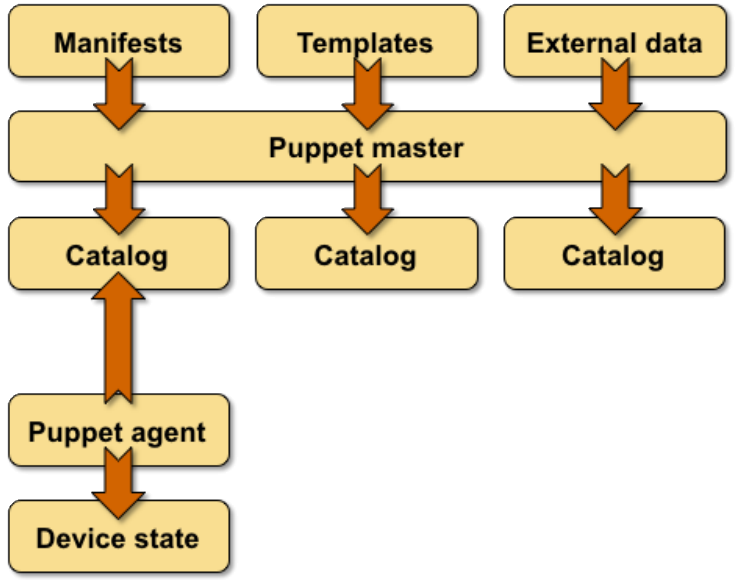
- The features you can manage on a node are limited by the capabilities of the node agent
- Most data center vendors support VLAN- and basic interface management
- Changes made by Puppet/Chef agent should be limited by RBAC

netdev framework

- Proxy agent on a Linux server
- Vendor-neutral network abstraction framework
- Usually VLAN, interface and LAG management

Junos implementation

- ERB templates for user-defined functionality
- Any aspect of Junos configuration can be managed with ERB template



Networking Vendor Puppet and Chef Support

	Puppet/Chef
Arista	✓
Brocade VDX	✓
Cisco IOS	✗
Cisco IOS-XR	✓
Cisco NX-OS	✓
Cumulus Linux	✓
Dell FTOS	✗
HP Comware	✗
Juniper Junos	✓

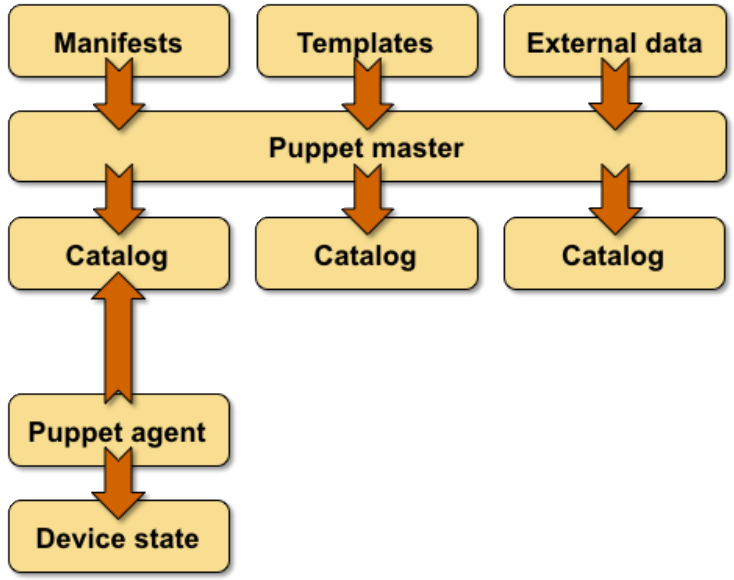
Puppet Support with Brocade NOS 7.0

Puppet agent for NOS 7.0

- Uses standard *netdev* model
- Puppet agent with Brocade provider runs on an external server
- NETCONF is used to manage VCS fabric configuration
- One manifest per VCS cluster

Supported objects

- Device (VCS ID, Rbridge ID)
- Interface (port) and LAG
- VLAN and L2 interface (VLAN-to-port mapping)



Puppet Support on Nexus OS

- Puppet agent running as a Linux container
- Nexus 3000, 5x00, 6000, 7000 and 9000

Standard resource types (netdev)

- DNS, NTP, RADIUS, SNMP, TACACS
- Interfaces, VLANs and port channels

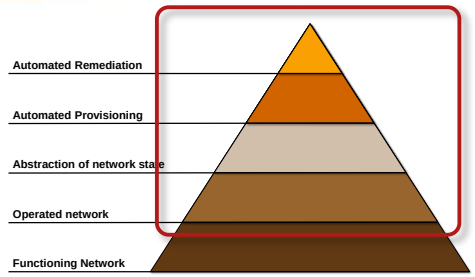
Cisco-specific resource types

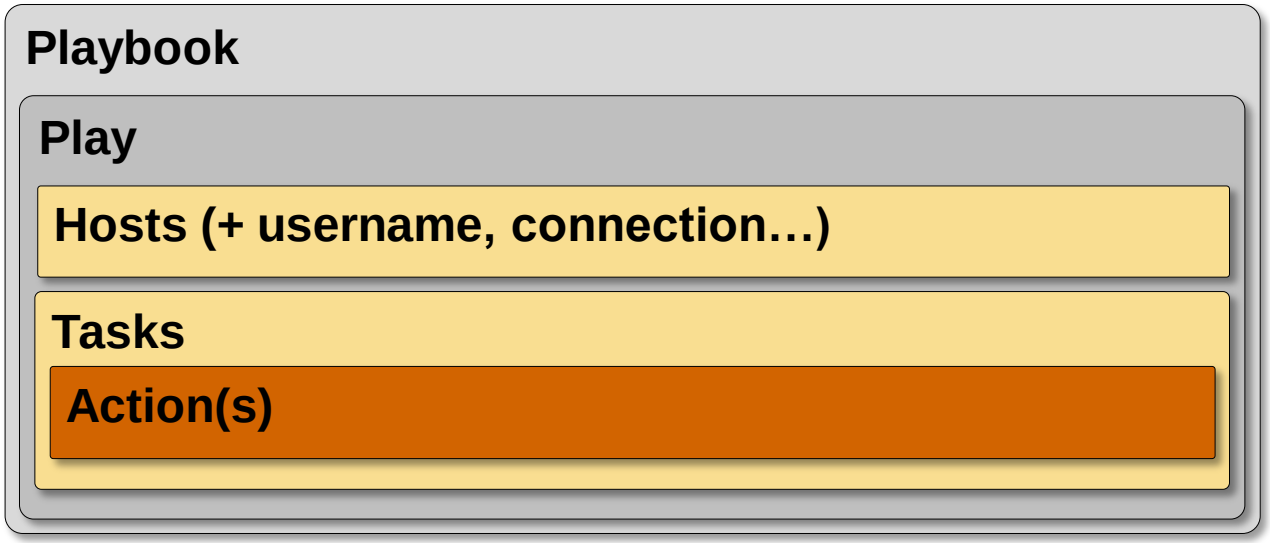
- BGP, OSPF, Multicast and STP
- Interfaces, Port Channel, vPC, FabricPath
- VLANs, VRFs, VNIs, VXLAN, EVPN
- AAA, RADIUS, TACACS
- DNS, NTP, SNMP, SYSLOG
- ACLs

✅ = Supported — = Not Applicable	N9k	N3k	N5k	N6k	N7k	Caveats
cisco_aaa_ authentication_login	✅	✅	✅	✅	✅	
cisco_aaa_ authorization_login_cfg_svc	✅	✅	✅	✅	✅	
cisco_aaa_ authorization_login_exec_svc	✅	✅	✅	✅	✅	
cisco_aaa_group_tacacs	✅	✅	✅	✅	✅	
cisco_acl	✅	✅	✅	✅	✅	
cisco_ace	✅	✅	✅*	✅*	✅*	*caveats
cisco_command_config	✅	✅	✅	✅	✅	
cisco_bgp	✅	✅	✅*	✅*	✅*	*caveats
cisco_bgp_af	✅*	✅*	✅	✅*	✅	*caveats
cisco_bgp_neighbor	✅	✅	✅	✅	✅	
cisco_bgp_neighbor_af	✅	✅	✅	✅	✅	
cisco_bridge_domain	—	—	—	—	✅	
cisco_bridge_domain_vni	—	—	—	—	✅	
cisco_encapsulation	—	—	—	—	✅	
cisco_evpn_vni	✅	—	✅	✅	✅	*caveats
cisco_fabricpath_global	—	—	✅	✅	✅*	*caveats
cisco_fabricpath_topology	—	—	✅	✅	✅	
cisco_interface	✅	✅	✅*	✅*	✅	*caveats
cisco_interface_channel_group	✅	✅	✅	✅	✅	
cisco_interface_ospf	✅	✅	✅	✅	✅	
cisco_interface_portchannel	✅*	✅*	✅*	✅*	✅*	*caveats

Automation Frameworks

Ansible

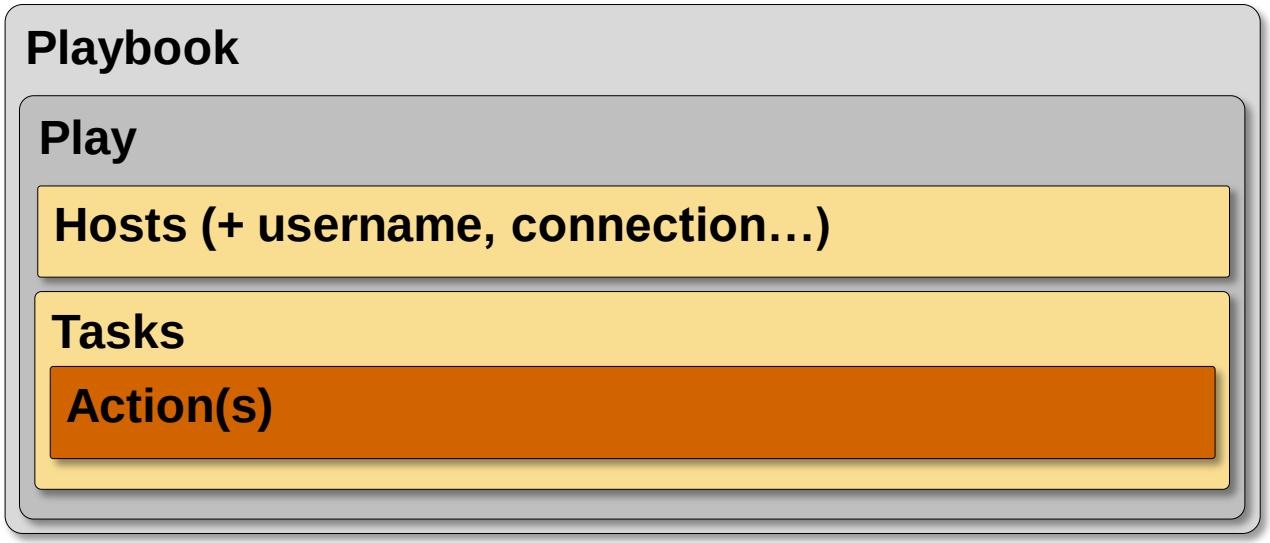




Ansible is a generic automation framework

- Plays in a playbook are executed in sequence every time a playbook is run
- Actions are executed by Ansible *modules* (plugins)
- No agent is running on the managed nodes
- Ansible modules download code to managed node and execute it, or use some other mechanism (API) to communicate with the managed node
- Each module might be idempotent… or not

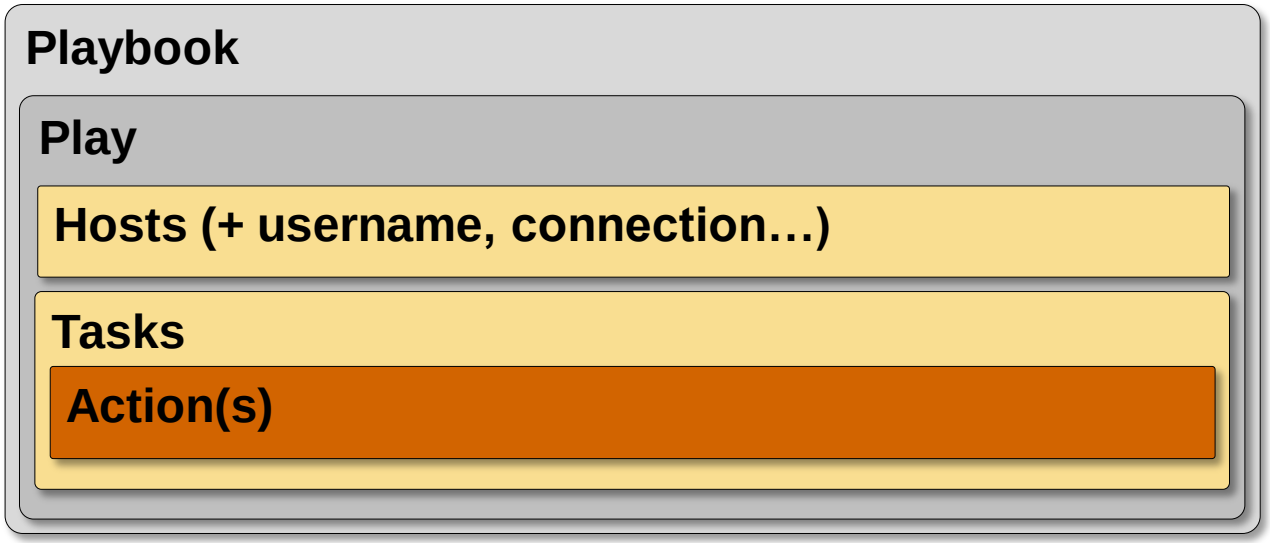
Network Automation with Ansible



Use cases

- Scripting (gather facts ➔ upgrade software ➔ reboot ➔ gather & verify facts)
- Configuration build from per-node variables and templates
- Configuration deployment
- Workflow automation (change ➔ approval ➔ version control ➔ deployment)
- Automated tests
- Automated troubleshooting

Typical Ansible Network Automation Scenario



Build new or modified configurations (standard Ansible modules)

Deploy configurations on network devices

- Vendor-supplied modules (Arista, Cumulus, Juniper, Palo Alto…)
- Community modules (NX-OS, Cisco IOS, Fortinet…)
- NAPALM (open-source framework)

Networking Vendor Ansible Support

	Ansible core	Third party	NAPALM
Arista	2.1	Vendor	✓
Brocade		User	
Cisco IOS	2.1	User	✓
Cisco IOS-XR	2.1		✓
Cisco NX-OS	2.1	Vendor	✓
Cumulus Linux		Vendor	
Dell FTOS			
HP Comware			
Juniper Junos	2.1	Vendor	✓

Simple Jinja2 Template

```
interface Loopback0
 ip address {{loopback.ip}} 255.255.255.255
 !
!
interface {{LAN.interface}}
 ip address {{LAN.ip}} 255.255.255.0
 !
!
interface {{WAN.0.interface}}
 description WAN uplink
 ip vrf forwarding Internet
 ip address {{WAN.0.ip}} {{WAN.0.subnet}}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
```

Slightly More Complex Jinja2 Template

```
{% for intf in WAN %}
interface {{WAN[intf].interface}}
 description WAN uplink
 ip vrf forwarding Internet
{% if WAN[intf].ip == 'DHCP' %}
 ip address dhcp
{% else %}
 ip address {{WAN[intf].ip}} {{WAN[intf].subnet|default(…)}}
{% endif %}
{% if WAN[intf].interface > 'Serial' %}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
{% endif %}
{% endfor %}
```

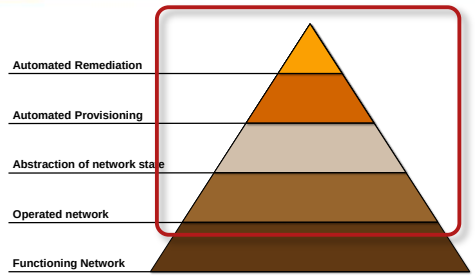
Sample Ansible Playbook

```
---
- name: Creating configurations for spoke routers
  hosts: spokes
  connection: local
  tasks:
    - name: build configurations
      template: src=spokes/main.conf dest={{inventory_hostname}}.conf

- name: Creating configurations for hub routers
  hosts: hubs
  connection: local
  tasks:
    - name: build configurations
      template: src=hubs/main.conf dest={{inventory_hostname}}.conf
```

Version Control and Reviews

Git: Code Repository



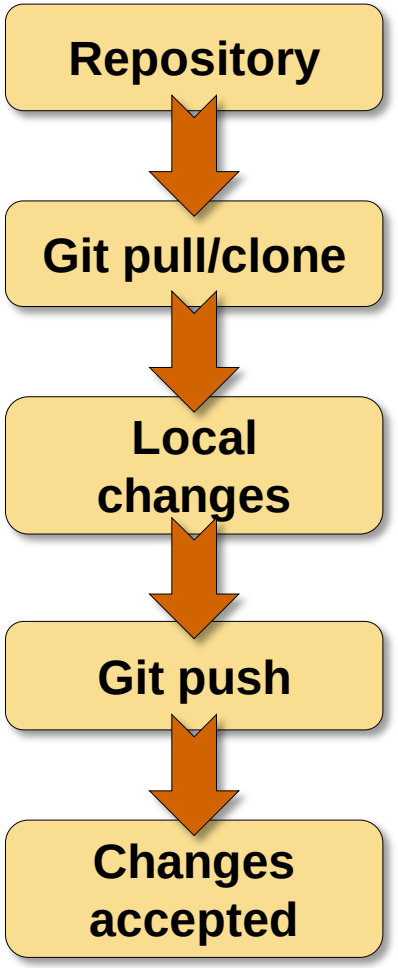
Git = Distributed revision control system

GitHub = Web-based Git repository hosting service

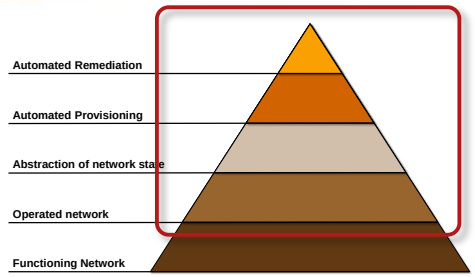
- Similar to SCCS, RCS, CVS, SVN
- Designed for Linux kernel development
- Widely used for all sorts of version control tasks
- **It doesn't matter which tool you use**

Use Git (or a similar tool) to:

- Track history of changes to device configurations
- Manage all text files related to your network
- Correlate file changes to requests, tickets, outages…



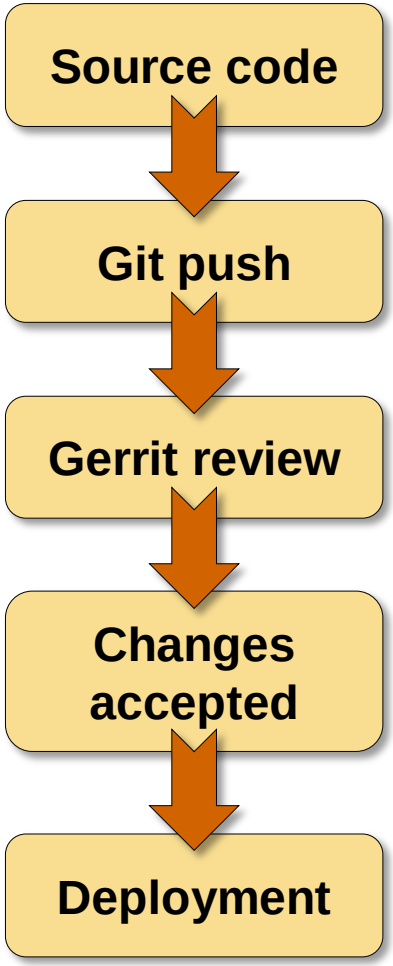
Gerrit: Code Review Framework



- Network configuration source code is modified (Ansible templates, node parameters, device configurations…)
- Changes are submitted to version control system
- **Gerrit** is used to automate the code review workflow
- Changes are accepted and incorporated into the version control system

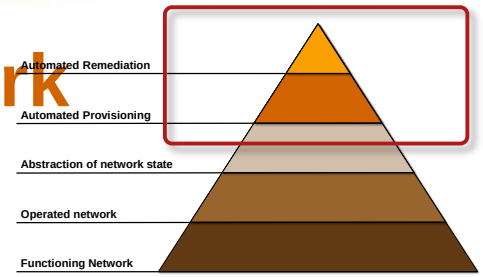
The new code version is deployed

- Ansible playbook builds new device configurations
- Changes between old and new configurations are reviewed and approved
- New configurations are deployed



Other Tools

Jenkins: Continuous Integration Framework

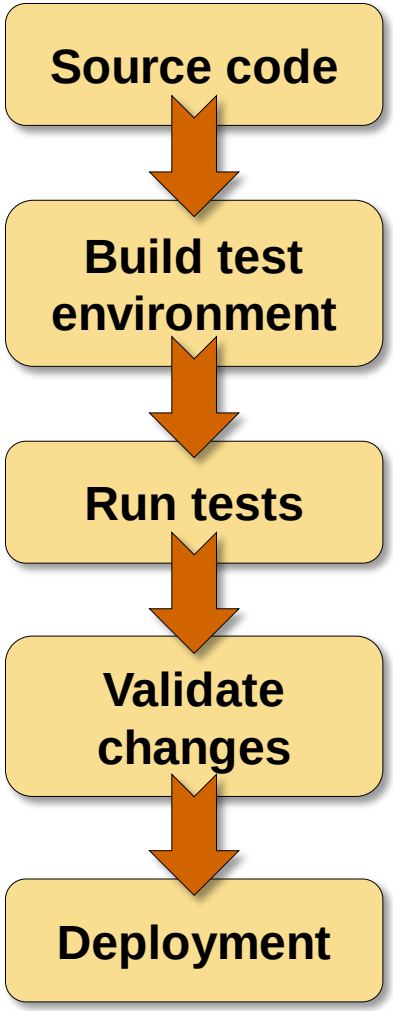


Validate changes to the source code by

- Building a test environment from the source code
- Run integration/deployment tests
- Validate tests results

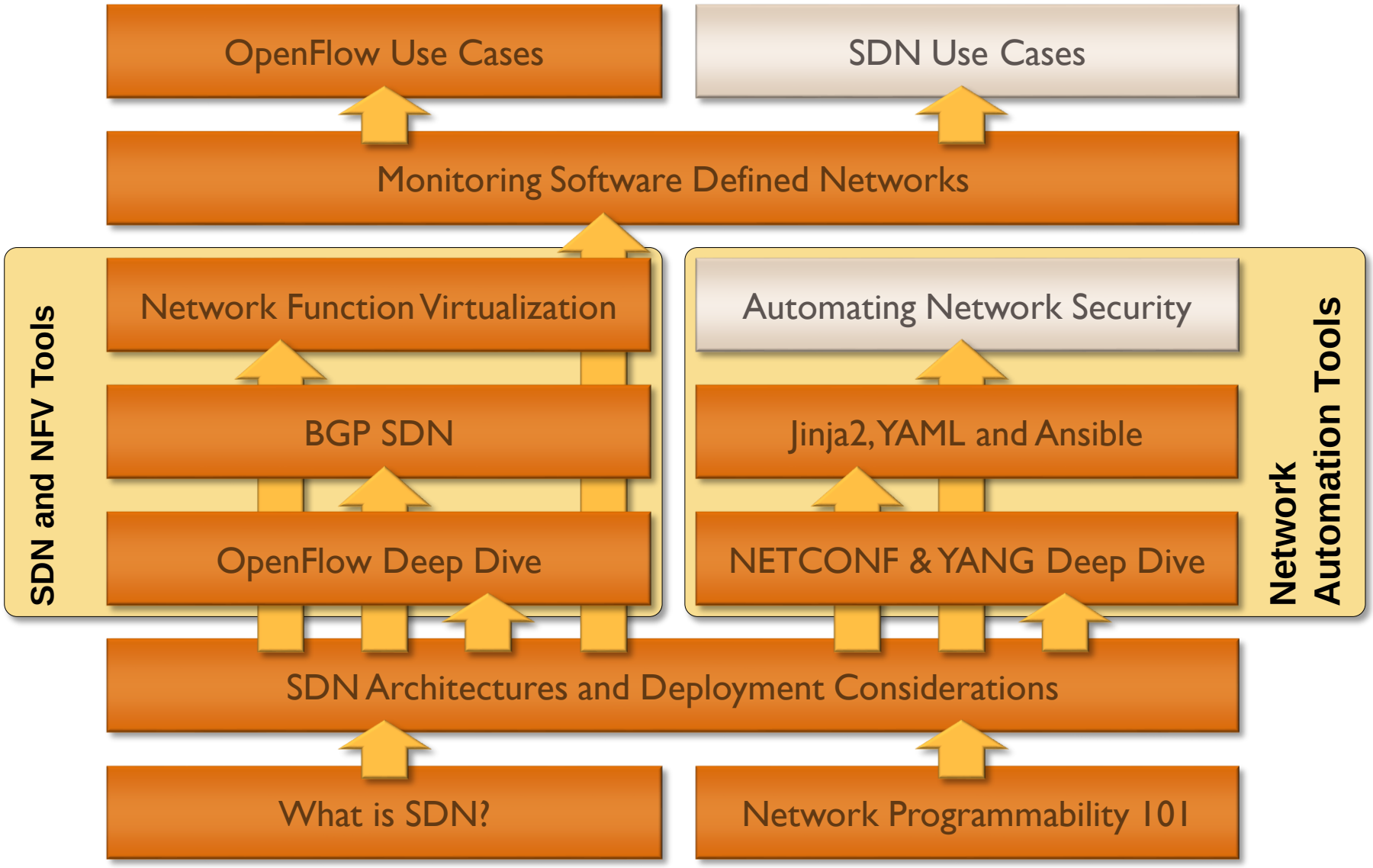
Network-focused usage

- Minimal: load new configuration on an actual device to verify its correctness
- Optimal: build a test environment using virtual devices and verify end-to-end connectivity



More Information

Advanced SDN and Network Automation Track



SUBSCRIBE to SDN mailing list

Get SDN tips, training announcements, presentations, videos and blog posts straight into your Inbox.

NAME:

EMAIL:

SIGN UP

SDN, OPENFLOW AND NFV RESOURCES ON IPSPACE.NET

Software-defined networking (SDN) can mean anything, from programmable network elements to architectures in which control- and forwarding planes reside on different devices.

The resources listed on this page will help you understand SDN, its implications and its applicability in your environment.

SDN TRAINING AND CONSULTING

- On-site and online consulting
- SDN, OpenFlow and NFV Workshop
- Software Defined Data Centers (SDDC) Workshop
- Advanced SDN Training
- Introduction to SDN
- Customized webinars and workshops

INDIVIDUAL SDN WEBINARS

- NETCONF and YANG
- Network Programmability 101
- SDN Architectures and Deployment Considerations
- VMware NSX Architecture

MORE SDN WEBINARS

SDN-RELATED BOOKS

- Overlay Virtual Networks in Software-Defined Data Centers

BUY NOW

- SDN and OpenFlow

BUY NOW

PRESENTATIONS

- SDN – 4 Years Later (video)
- What is SDN?
- Should I program my network? (video)
- Virtual Routers
- From Traditional Silos to SDDC (video)
- What Matters is Your Business (video)
- Automating Network Security, Troopers 15

MORE SDN PRESENTATIONS

MORE SDDC PRESENTATIONS

Stay in Touch

Web:	ipSpace.net
Blog:	blog.ipSpace.net
Email:	ip@ipSpace.net
Twitter:	@ioshints

SDN:	ipSpace.net/SDN
Webinars:	ipSpace.net/Webinars
Consulting:	ipSpace.net/Consulting

Questions?

Send them to ip@ipSpace.net or @ioshints