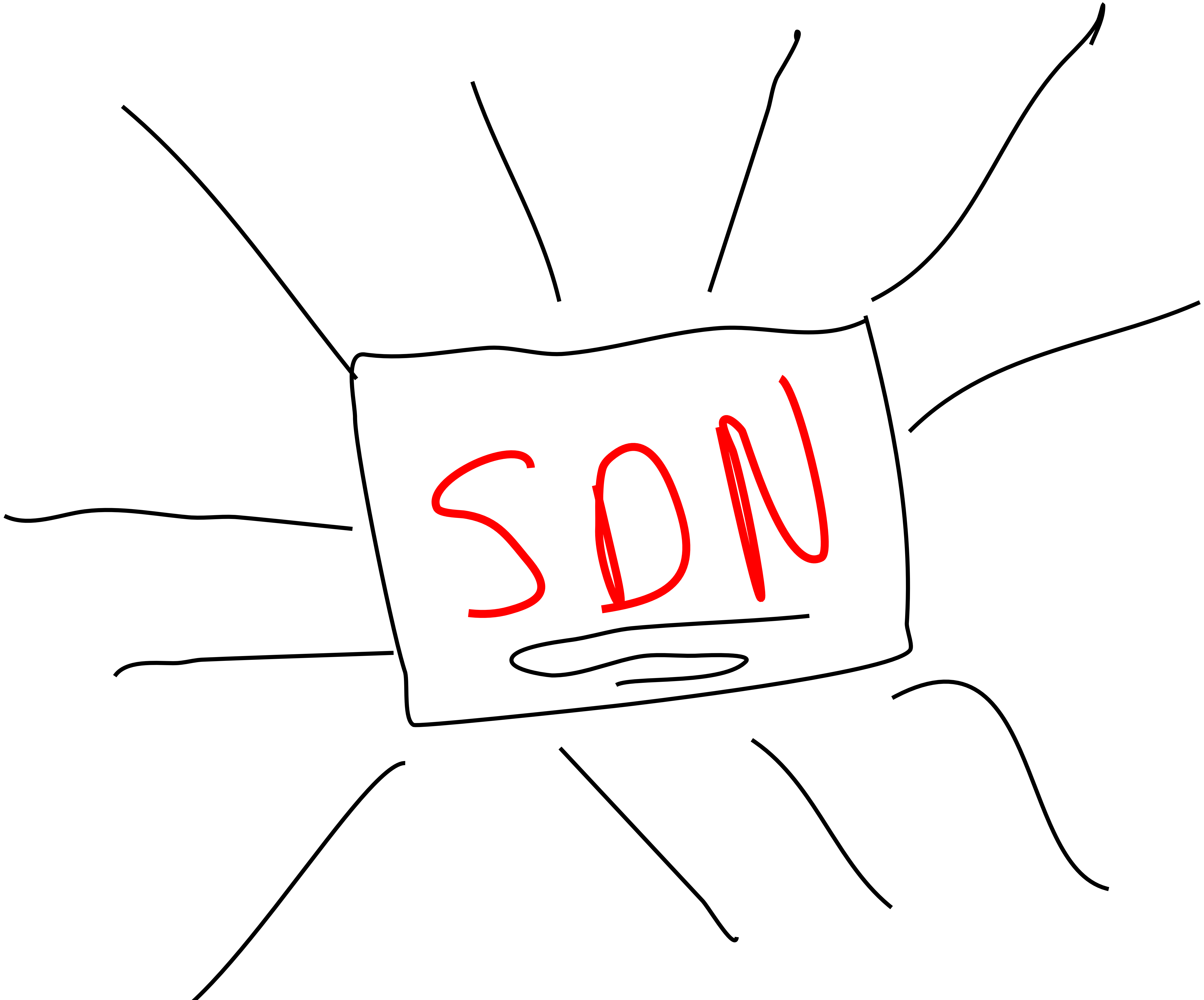
Network Automation Tools and Practices
October 15, 2017, 1:00-5:00
Pacific E on Pacific Concourse Level

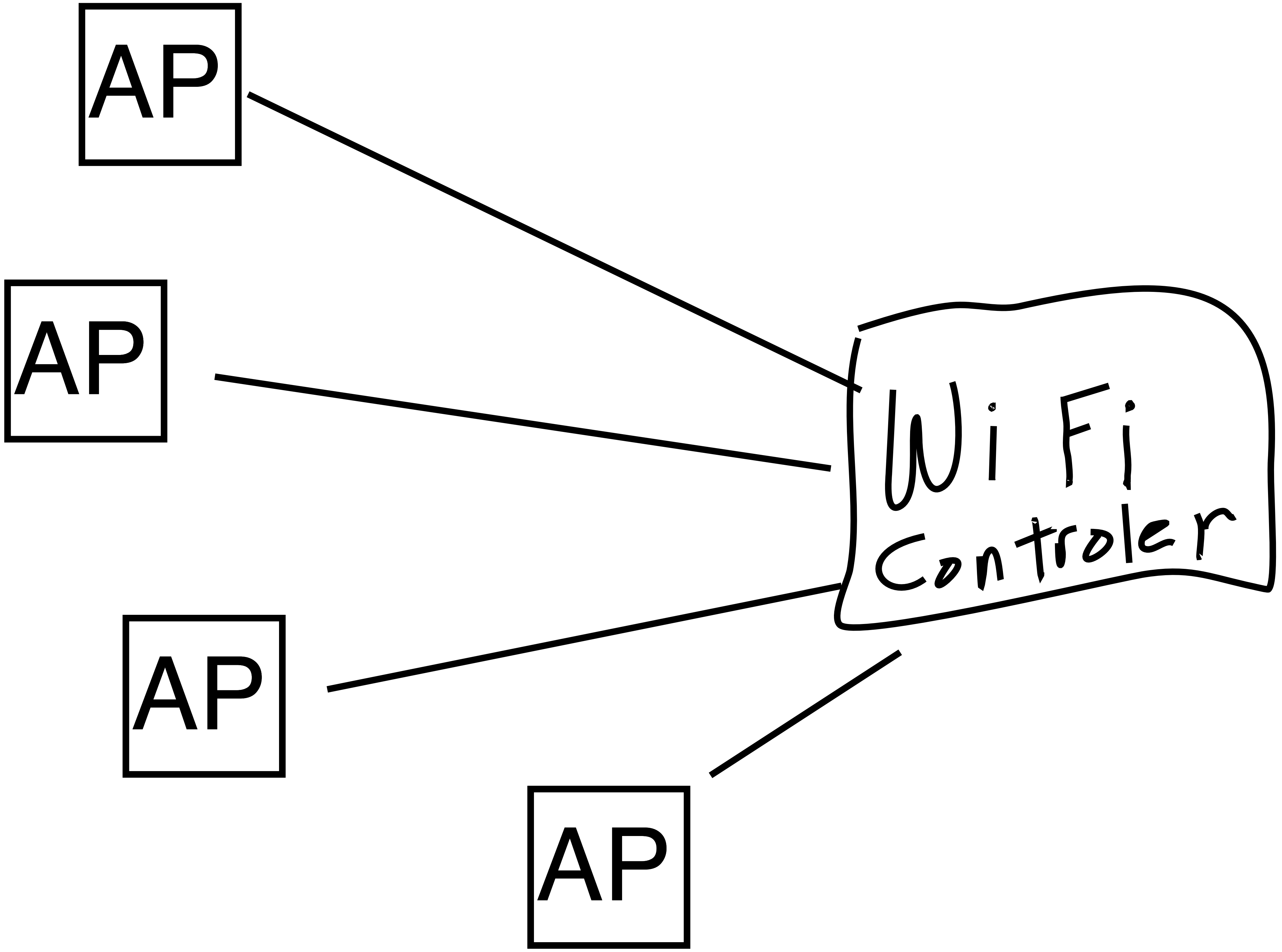
Agenda

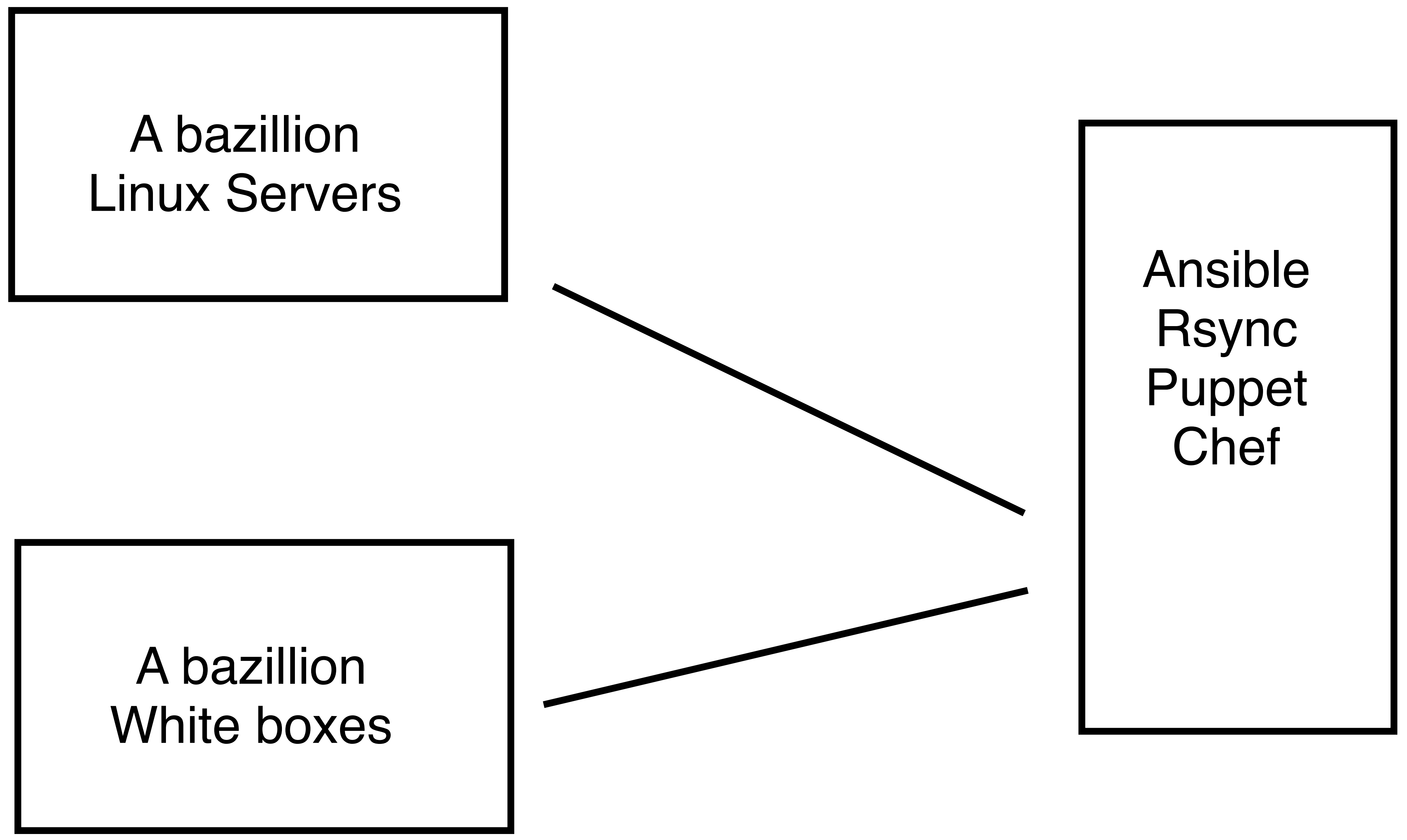
Welcome (Linda Roos, Internet2)
Brief Overview of Network Automation (Steven Wallace, Indiana University)
Openflow Retrospective (Ed Balas and A.J. Ragusa, Indiana University GlobalNOC)
Network Automation at KINBER (Mike Carey, KINBER)
Network Automation with Ansible (Frank Seesink, WVnet)
Finding or Creating Network Savvy Programmers (A.J. Ragusa and Ed Balas, Indiana University GlobalNOC)
Discussion
Next Steps

Brief Overview of Network Automation

Steven Wallace


SDN


AP
AP
AP
AP
WiFi
Controler


A bazillion Linux Servers
A bazillion White boxes
Ansible
Rsync
Puppet
Chef

Openflow Retrospective

Ed Balas and AJ Ragusa
Indiana University Global NOC

Automation pre SDN / OF buzz

- Does not require god products or big architectures
- config generation / push 1997
  - accept filters etc
- Autotriage using craft interfaces 1998
  - layer2 FDDI path discovery and audit
- VLAN provisioning portal
  - Sherpa

OpenFlow Considerations

(that apply to other tech too)

Looking back at a talk I gave 2 years ago on March 2, 2015 as part of the OIN workshop

Are you sure you ?

- **You are now the System Integrator**
    - control plane separation price
- vendor implementations are
    - buggy
    - incomplete
    - poor performing

~~when, not if~~ If, NOT when

- OpenFlow ideal for use cases like SciPass
    - 5-tuple based control
    - multiple output actions
- When depends on use case and evolution of ecosystem
    - apps
    - switches

going deeply programmable?

- Don’t reinvent the wheel
  - MPLS or BGP
  - pay attention to cost of integration
- Not a large number of apps / controllers out there
- Not a lot of interop testing
- Need your own test lab

Forklift upgrade of entire fabric unlikely

- follow incremental strategy
- look at hybrid or appliance mode
- identify unserved niches

What you need to know about OpenFlow Switches

- nobody supports the entire spec
- many details subject to interpretation
- controllers are trusted and can act as DoS vector
- dont assume data plane performs well
- control plane performance may disappoint
- 100g deep buffer rare

Testing

- ~~anytime~~ every time something changes
- have seen code fail in production due to libssl upgrade —> cypher support
- firmware bugs specific to type of line card
- yes yes you do need to test 100g and the 10g
- not all modules are supported in OpenFlow Mode

Testing
new switch code rev

- 30 hrs of engineer time if things go smooth
- 40+ hrs if an issue is discovered
- averaged about 1 test per month
- the amount of time to perform test has been growing

Testing apps we wrote

- 30 hrs of testing if things go smooth
- double if non-trival fix required
- 4 releases of OESS in last 6 months
- test time for this has been growing as well
  - going back up with shift to MPLS
- we are working to improve automation

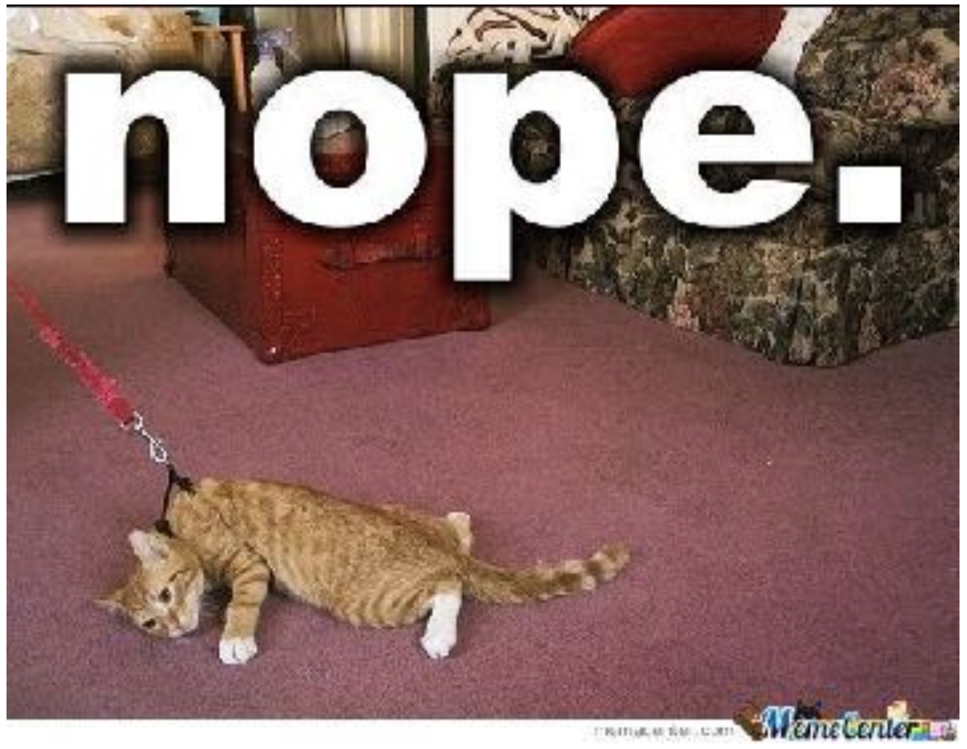
2 years later

What happened to OpenFlow on AL2S

- Lack of support from vendors
  - low quality hardware in one vendor
  - lack of OF support in another
- swap out the wing while in flight
  - now a single vendor network
  - moved from OF to MPLS and Netconf
  - slow and methodical process

Did Internet2 give up on SDN?

- some may have drawn the wrong conclusion here



What is Internet2 doing with SDN

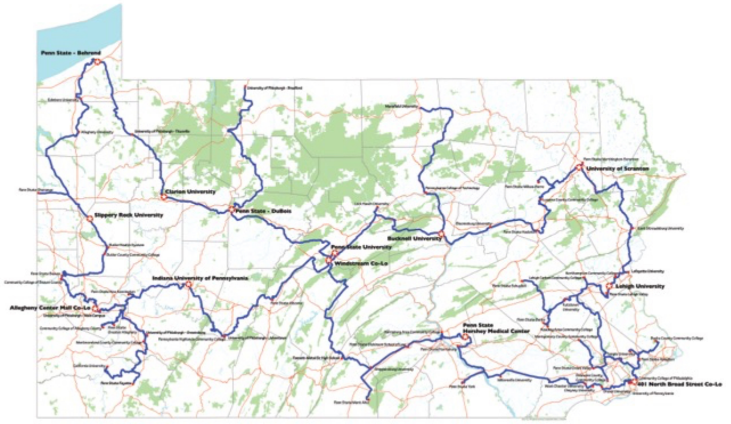
- SDN is a philosophy not a technology
- OESS still the way to provision AL2S
  - now with Netconf and MPLS
- OpenFlow supported with Corsa using overlay
  - the lack of advanced use cases is concerning
- cloud service orchestration
- tactical automation
- system disaggregation

Determining the Value of Network Automation for Small Networks

SPEAKER **Michael Carey** KINBER (Keystone Initiative for Network Based Education & Research)

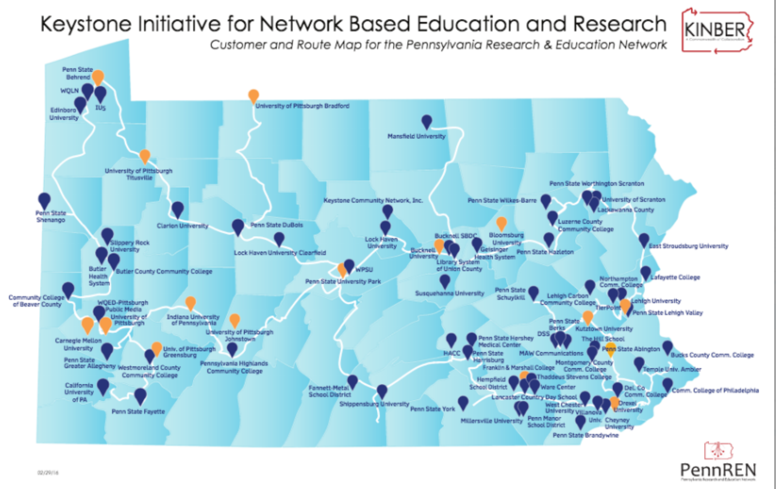
Determining the Value of Network Automation for Small Networks

- Who is KINBER and How Does that Effect our Network Automation Strategy
- KINBER's Automation Strategy
- Automation Tools & Partners
- Successes and Challenges



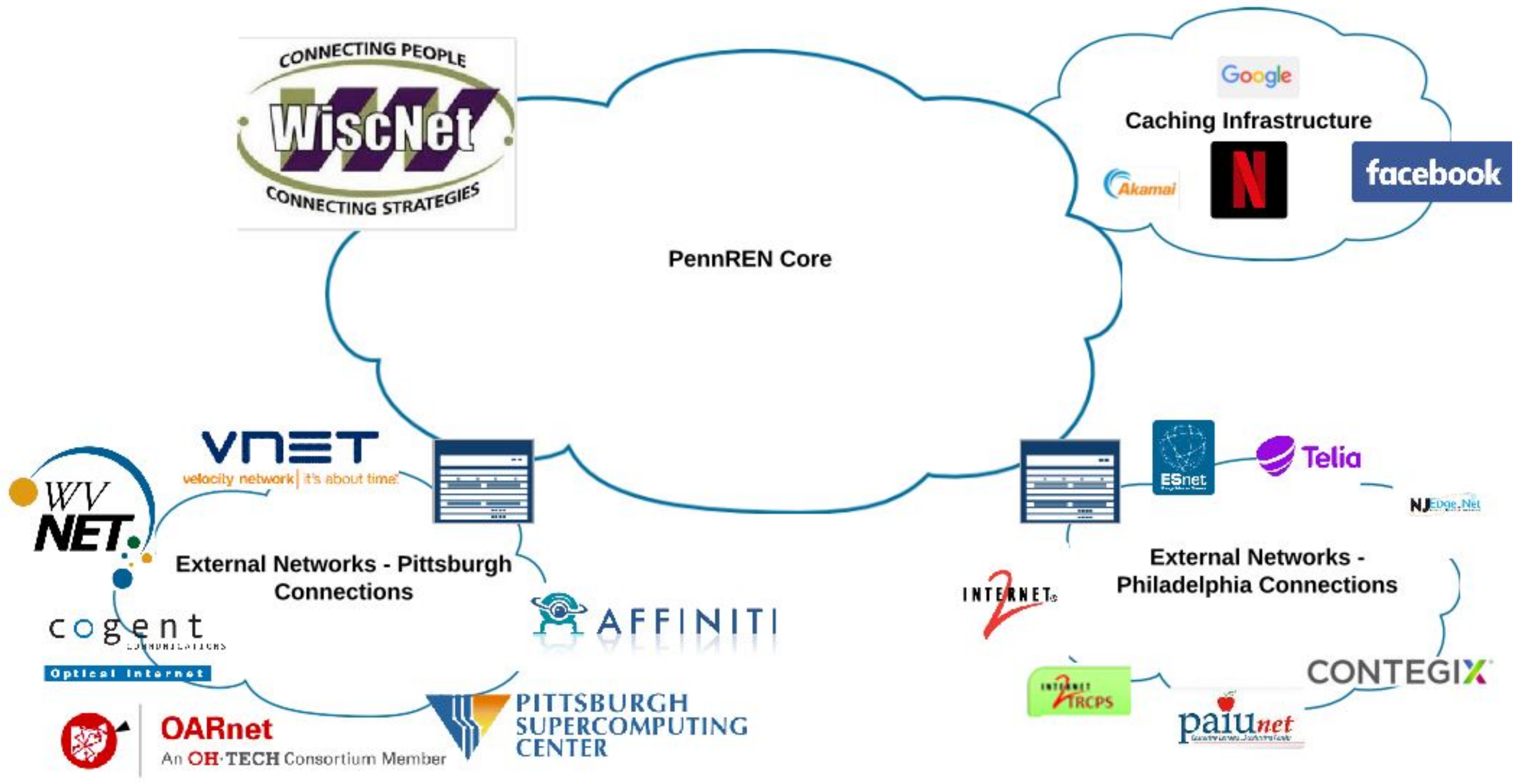
KINBER/PennREN

- 152 Connections
  - 2x 100G Connections
  - 27x 10G Connections
- 30-40G+ Egress Traffic
- Own 1800+ miles of fiber, 110 POPs
- Small Staff
  - 7 Full-Time
    - 3 Engineers including myself
  - Tier I (Service Desk) & II (1st Level Engineering) NOC Services contracted to GlobalNOC

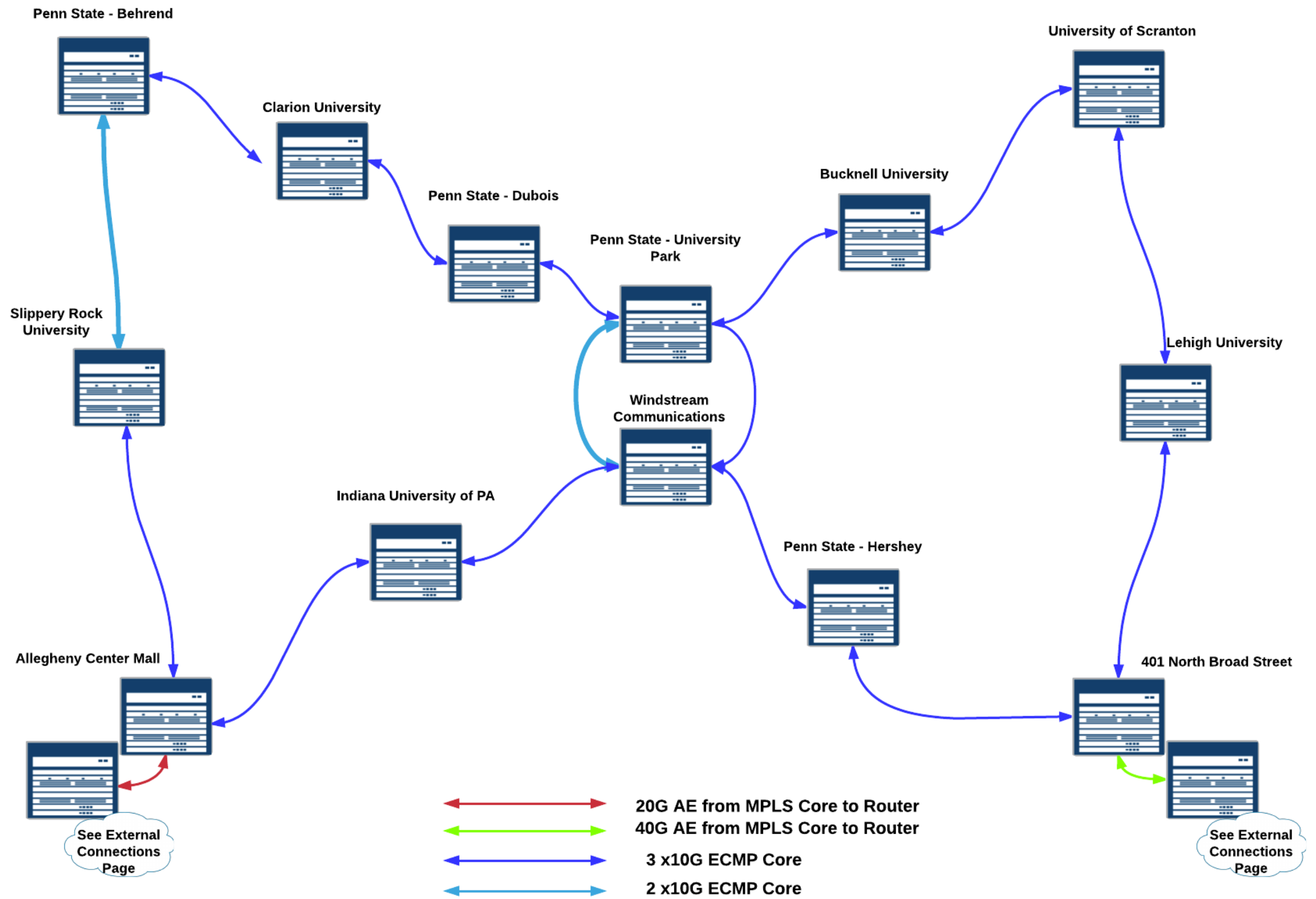


Keystone Initiative for Network Based Education and Research
Customer and Route Map for the Pennsylvania Research & Education Network
KINBER
Penn State Behrend
WQLN
Edinboro University
IUS
University of Pittsburgh Bradford
Mansfield University
University of Pittsburgh Titusville
Penn State Shenango
Clarion University
Penn State DuBois
Keystone Community Network, Inc.
Penn State Wilkes-Barre
Penn State Worthington Scranton
University of Scranton
Lackawanna County
Luzerne County Community College
Slippery Rock University
Butler Health System
Butler County Community College
Lock Haven University Clearfield
Lock Haven University
Bucknell SBDC
Bucknell University
Library System of Union County
Bloomsburg University
Geisinger Health System
Penn State Hazleton
East Stroudsburg University
WPSU
Penn State University Park
Community College of Beaver County
WQED-Pittsburgh Public Media
University of Pittsburgh
Indiana University of Pennsylvania
University of Pittsburgh Johnstown
Susquehanna University
Penn State Schuylkill
Lehigh Carbon Community College
Northampton Comm. College
Lafayette College
Lehigh University
TierPoint
Penn State Lehigh Valley
Penn State Berks
Kutztown University
DSS
The Hill School
Carnegie Mellon University
Penn State Greater Allegheny
Univ. of Pittsburgh Greensburg
Westmoreland County Community College
Pennsylvania Highlands Community College
Penn State Hershey Medical Center
HACC
Penn State Harrisburg
MAW Communications
Penn State Abington
Bucks County Comm. College
Montgomery County Comm. College
Temple Univ. Ambler
Franklin & Marshall College
Thaddeus Stevens College
Hempfield School District
Ware Center
California University of PA
Penn State Fayette
Fannett-Metal School District
Shippensburg University
Penn State York
Lancaster Country Day School
West Chester University
Villanova Univ.
Del. Co Comm. College
Comm. College of Philadelphia
Drexel University
Cheyney University
Millersville University
Penn Manor School District
Penn State Brandywine
02/29/16
PennREN

KINBER External Network Peering Connection Overview

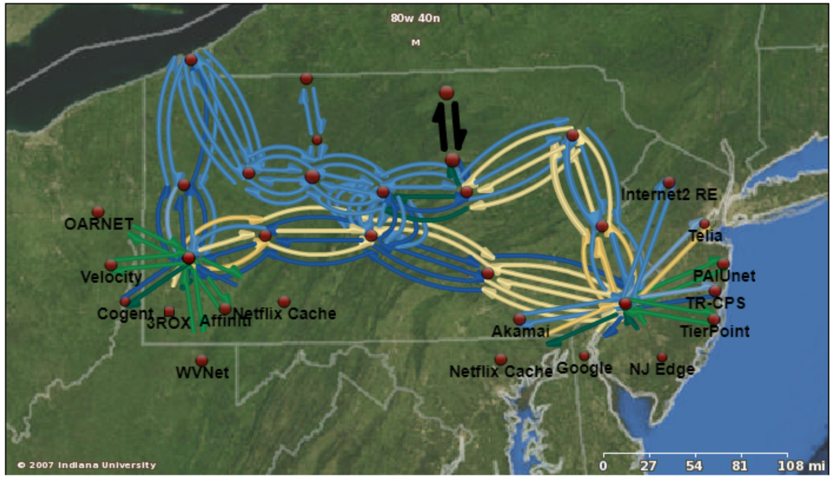


KINBER Statewide MPLS Core Backbone



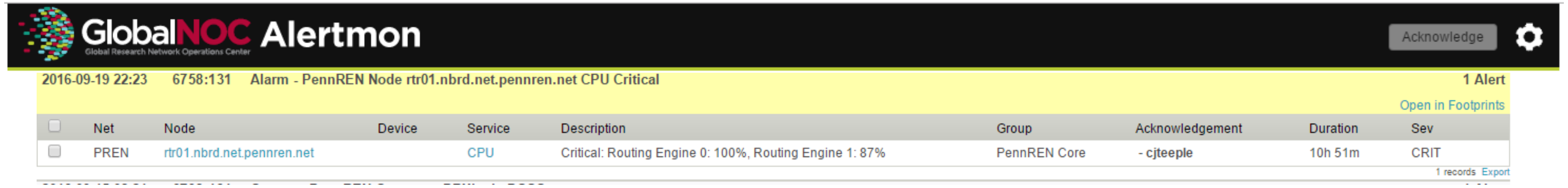
KINBER Business Lifecycle

- Seed ✓
- Startup ✓
- Growth ✓
  - Engineering Resource Focus
    - 80% of Engineering is Provisioning New Customers
    - 15% is Tier III Break/Fix Situations
    - 5% Network Enhancement Projects (Improvements, Better Services, Better Responses)
  - Constant range of issues bidding for time
  - R&R
    - Revenue
    - Reputation



KINBER Automation Strategy

- Relate to Business Life Cycle & Business Processes
  - **Provisioning**
    - Faster Provisioning = Faster Revenue
    - Faster Provisioning = Better Customer Experience
    - Less Errors = Better Customer Service
  - **Break/Fix**
    - Better Data Collection = Quicker Problem Identification



2016-09-19 22:23 6758:131 Alarm - PennREN Node rtr01.nbrd.net.pennren.net CPU Critical 1 Alert

Open in Footprints

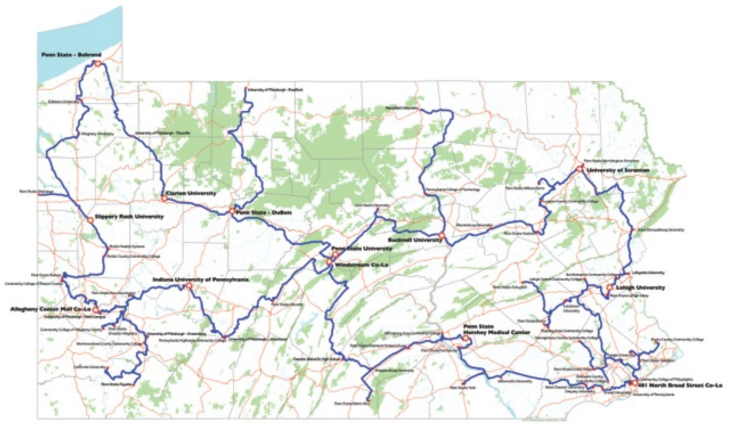
	Net	Node	Device	Service	Description	Group	Acknowledgement	Duration	Sev
☐	PREN	rtr01.nbrd.net.pennren.net		CPU	Critical: Routing Engine 0: 100%, Routing Engine 1: 87%	PennREN Core	- cjteeple	10h 51m	CRIT

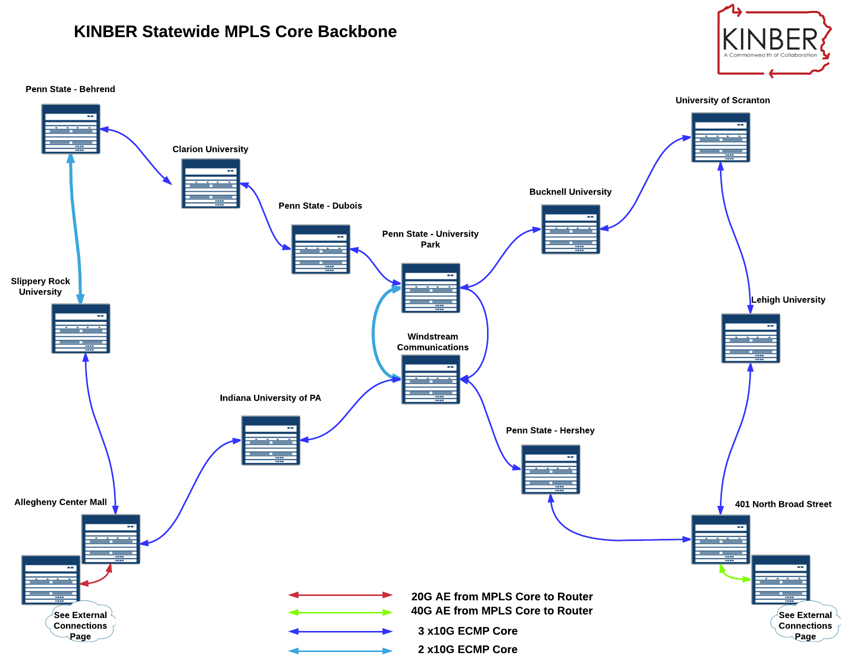
1 records Export

KINBER Automation Strategy

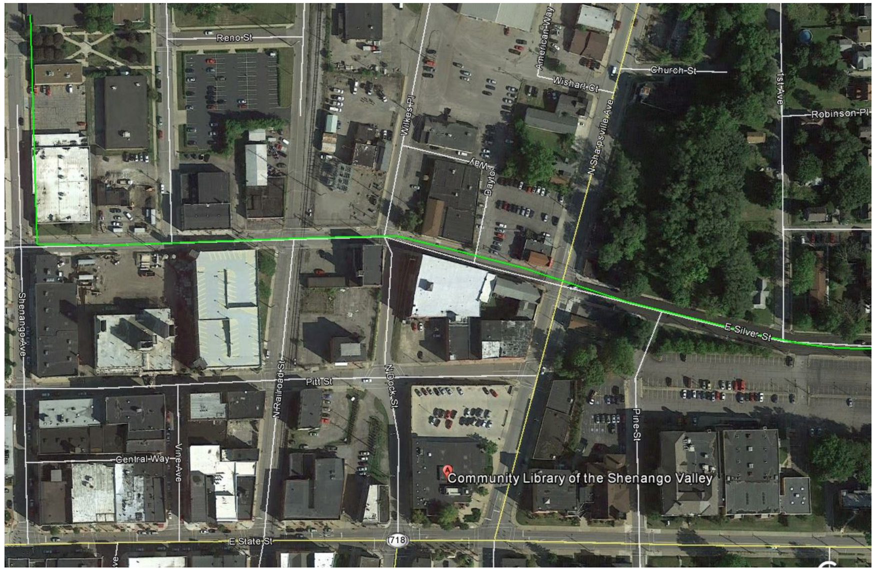
- **Provisioning**
    - Network Design
    - Procurement
    - **Network Configuration**
    - **Network Deployment**
    - Certification of Services
    - Certification of Database Records

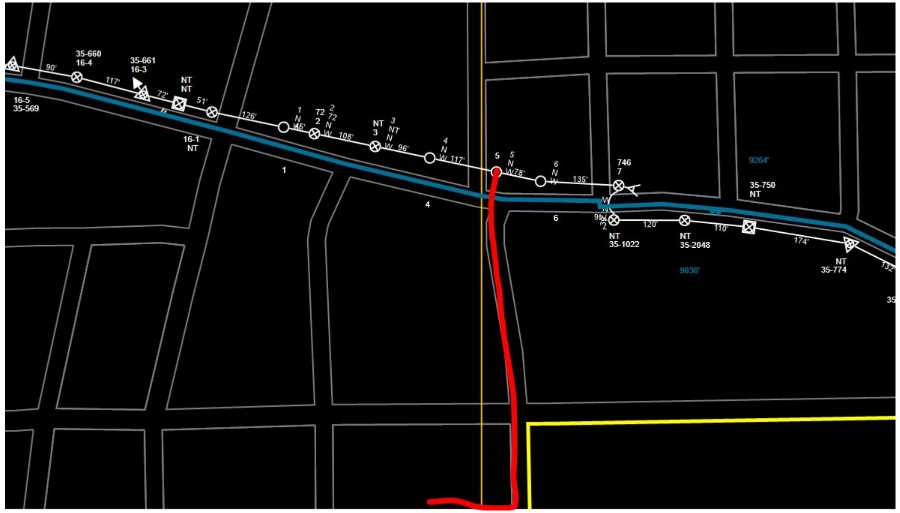
KINBER Automation Strategy – Provisioning/Design





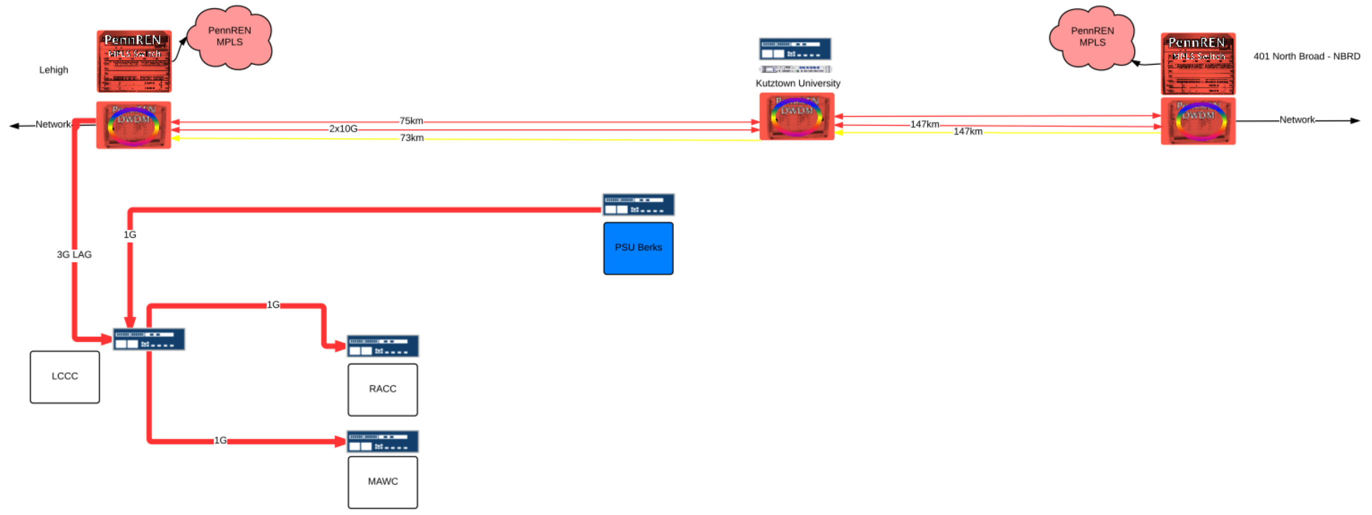
KINBER Automation Strategy - Design





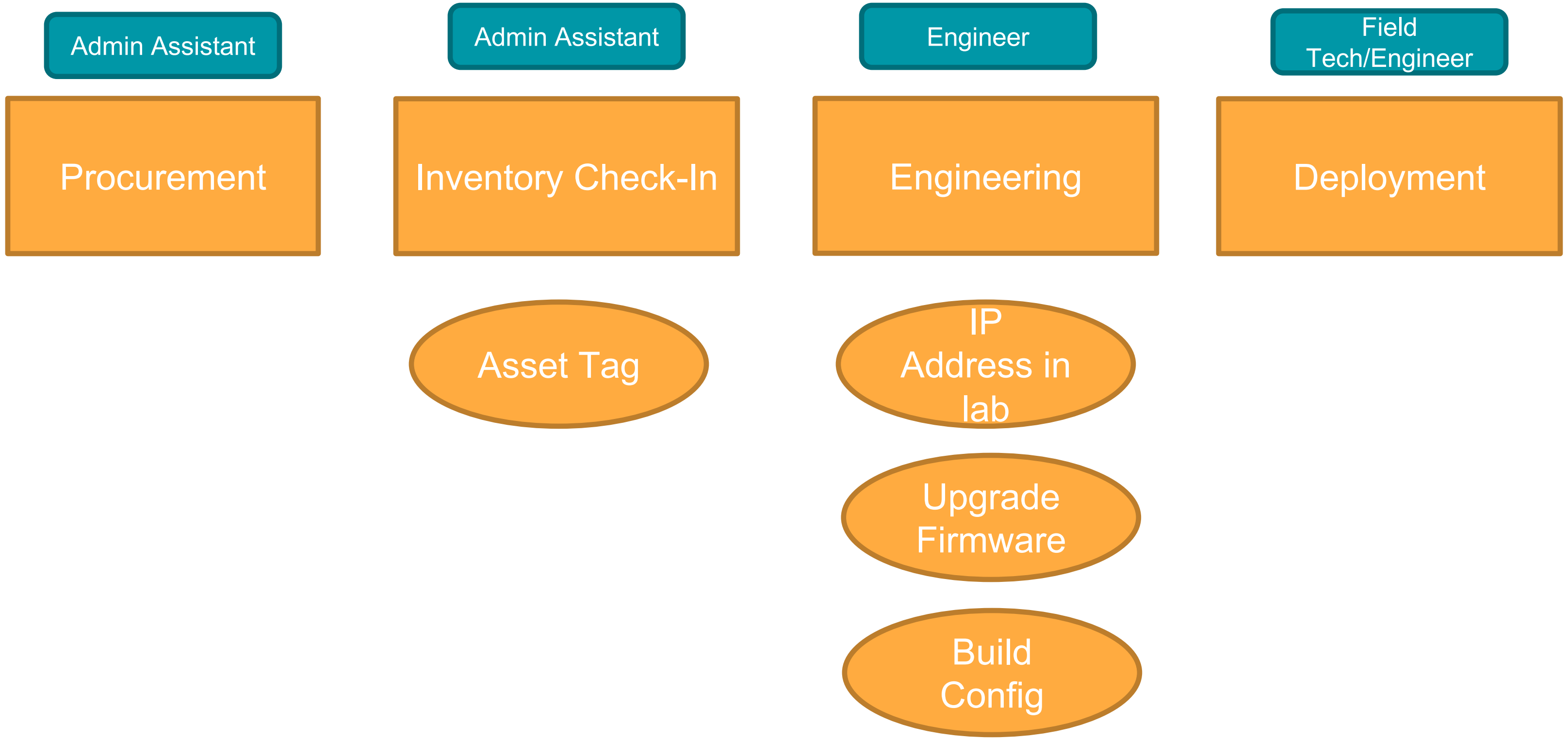
KINBER Automation Strategy - Provisioning

- Juniper EX3300/EX3400/MX-104
- 3rd Party Optics



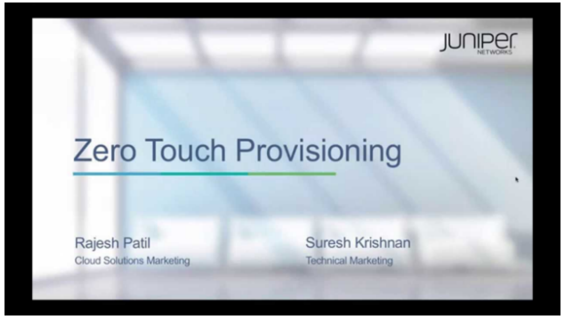
PennREN

KINBER Automation Strategy - Provisioning

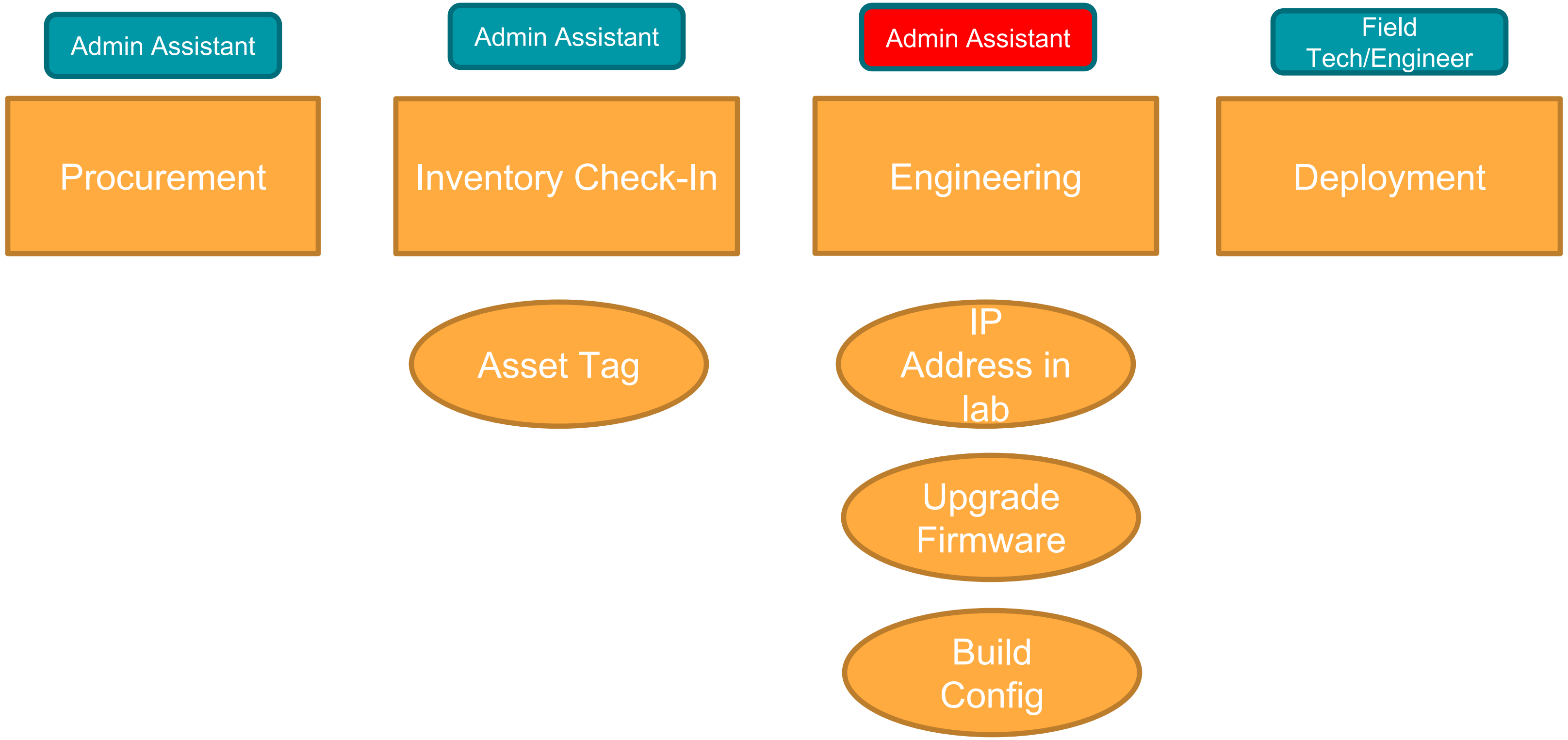


Network Automation – Zero Touch Provisioning

- **Partner**
  - Juniper
- **Tools**
  - Zero-Touch Provisioning (ZTP)
- **Process**
  - Majority of our Node Expansion is Juniper EX Models off a Core Node
  - Can involve **non-engineering staff** powering up unit and plugging into lab environment where ZTP performs an initial code upgrade and base configuration load



Network Automation – Zero Touch Provisioning



Network Automation – Zero Touch Provisioning

DHCP Server

Cables/Power Cords/Switch

TFTP Server

JUNOS Firmware

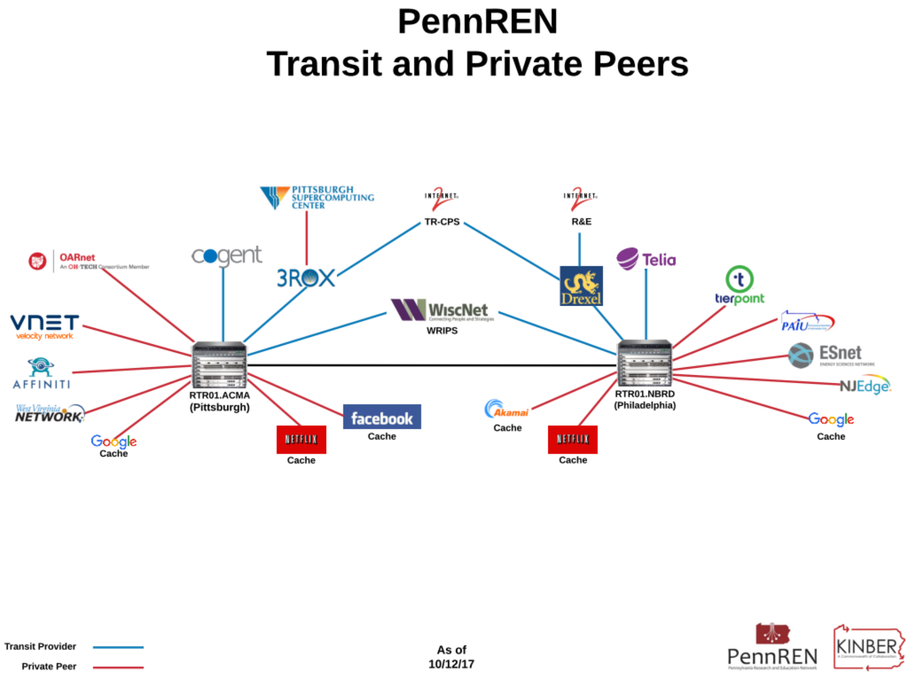
Base Config Files

Network Automation – Zero Touch Provisioning

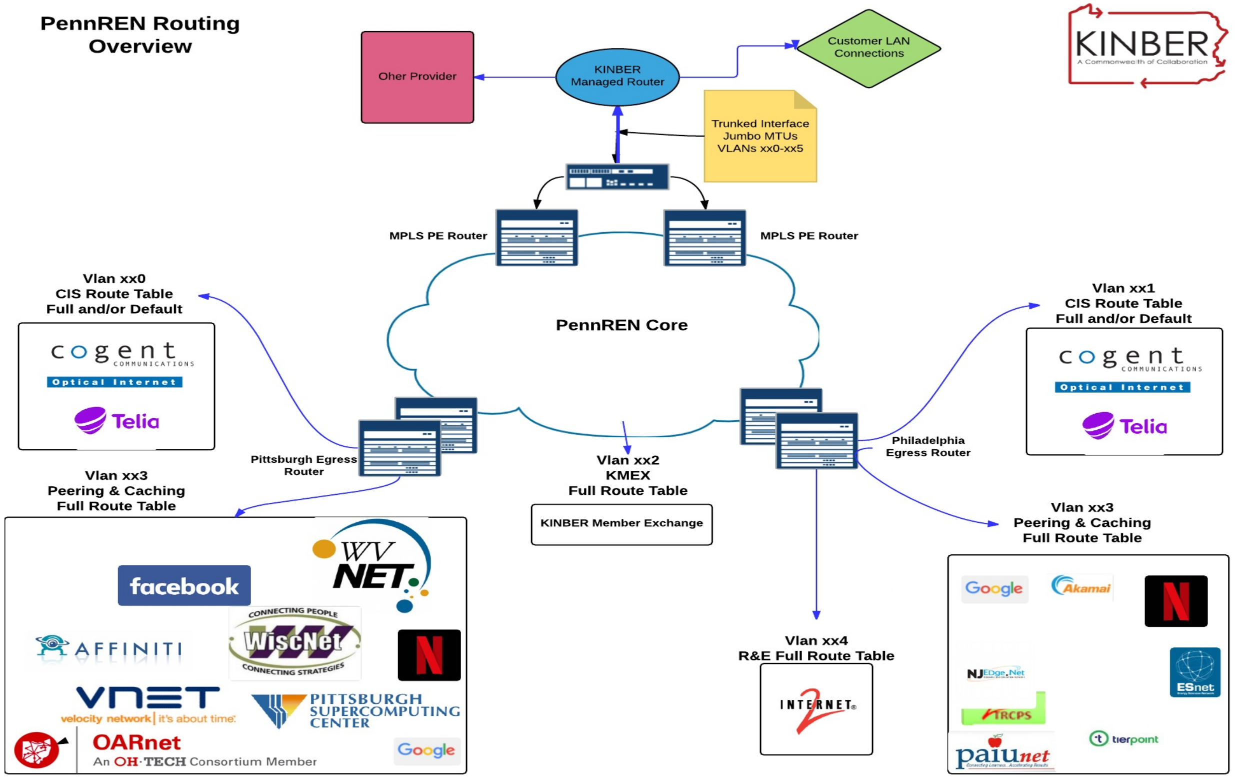
- Reallocated Engineering Time
- Reduced Configuration Time
- Faster Provisioning = Faster Revenue, Better Customer Experience

- Next Steps
  - Replicate in the field?
  - Improved Base Configuration
  - Email based alert – “Your Switch/Router is now ready”

Network Automation Tools and Implementation



Network Automation Tools and Implementation



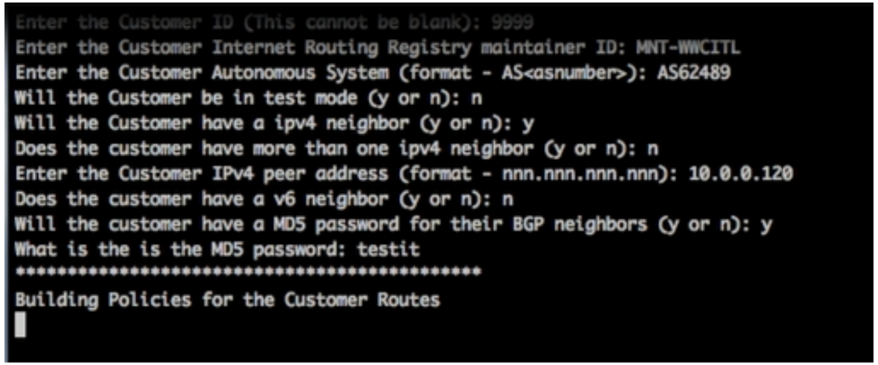
PennREN Routing Overview
Oher Provider
KINBER Managed Router
Customer LAN Connections
KINBER A Commonwealth of Collaboration
Trunked Interface Jumbo MTUs VLANs xx0-xx5
MPLS PE Router
MPLS PE Router
PennREN Core
Vlan xx0 CIS Route Table Full and/or Default
cogent communications Optical Internet
Telia
Vlan xx1 CIS Route Table Full and/or Default
cogent communications Optical Internet
Telia
Pittsburgh Egress Router
Philadelphia Egress Router
Vlan xx2 KMEX Full Route Table
KINBER Member Exchange
Vlan xx3 Peering & Caching Full Route Table
facebook
WV NET
AFFINITI
WiscNet
N
VNET velocity network | it's about time
PITTSBURGH SUPERCOMPUTING CENTER
OARnet An OH·TECH Consortium Member
Google
Vlan xx4 R&E Full Route Table
INTERNET2
Vlan xx3 Peering & Caching Full Route Table
Google
Akamai
N
NJEDge.Net
ESnet
TRCPS
tierpoint
paiunet

Network Automation Tools - Provisioning

- **Partner**
  - GlobalNOC
- **Tools**
  - Dist-Tool
- **Process**
  - **System-wide configuration changes**
    - dist-tool --template VPLS_Service.json --node-name swt01.psup.net.pennren.net
    - dist-tool --template RE_Service.json --node-name rtr01.nbrd.net.pennren.net
    - dist-tool --template CIS_Service.json --node-name rtr01.acma.net.pennren.net

Network Automation Tools – Routing Provisioning

- **Partner**
  - Integration Partners
- **Tools**
  - Provisioning Application for Peering
- **Process**
  - Streamline process of adding new peers
  - Automate AS-SET



```
Enter the Customer ID (This cannot be blank): 9999
Enter the Customer Internet Routing Registry maintainer ID: MNT-WWCITL
Enter the Customer Autonomous System (format - AS<asnumber>): AS62489
Will the Customer be in test mode (y or n): n
Will the Customer have a ipv4 neighbor (y or n): y
Does the customer have more than one ipv4 neighbor (y or n): n
Enter the Customer IPv4 peer address (format - nnn.nnn.nnn.nnn): 10.0.0.120
Does the customer have a v6 neighbor (y or n): n
Will the customer have a MD5 password for their BGP neighbors (y or n): y
What is the is the MD5 password: testit
*********************************************
Building Policies for the Customer Routes
```

KINBER Automation Strategy – Break/Fix

- **What break-fix steps can we improve on?**
  - Initial Data Collection
  - Troubleshooting Commands

- **BGP Session Drops…….**
  - What does Tier I do?
  - What does Tier II do?

Network Automation Tools – Break/Fix

- **Partner**
  - GlobalNOC
- **Tools**
  - Dist-Tool
- **Process**
  - **System-wide configuration changes**
    - dist-tool --template BGP_Alarm.json –Service-ID PREN-S05413
    - dist-tool --template CPU_Alarm.json --node-name rtr01.nbrd.net.pennren.net
    - dist-tool --template UPS_Alarm.json --node-name ups01.psup.net.pennren.net

KINBER Automation Strategy – Break/Fix

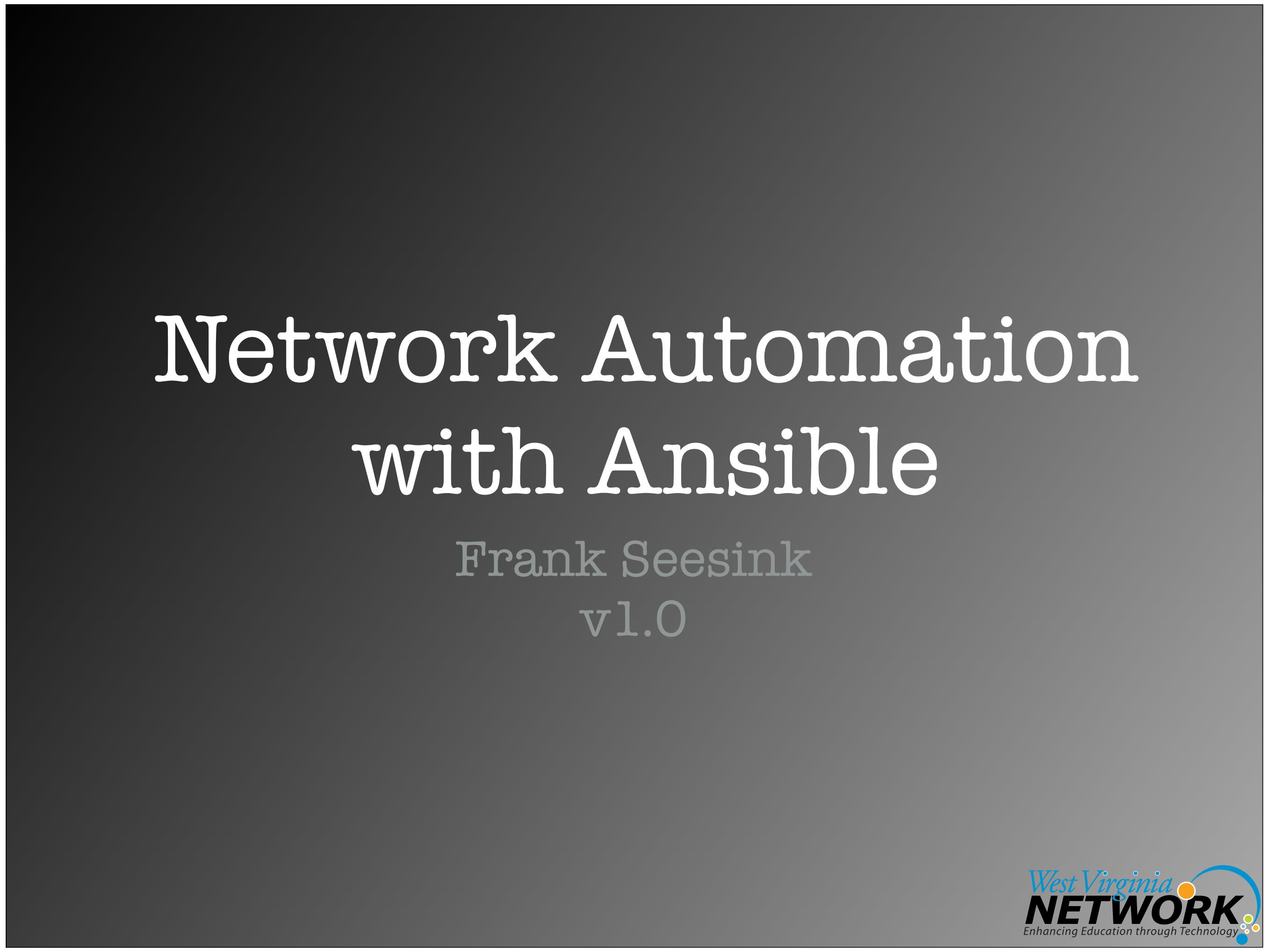
- Reduce MTTR
- Better Customer Experience
- Better Utilize Tier II personnel (on-call hours)
- Train Tier I staff

Successes and Challenges

- **Successes**
    - Continous theme of “Benefits in Automation”
    - Metrics are starting to form to detail reasons for automation
    - Translating Ideas to Solutions **for Business Purposes**
- **Challenges**
    - Resources availability in Growth-Stage our the Company
        - Translating Ideas to Solutions

Network Automation Working Group?

- Share ZTP Templates and Design (And other vendor equivalents)
- Establish an Peering/IRR Toolkit that is easily portable across members
- Establish and share Break/Fix Scripts



Network Automation with Ansible

Frank Seesink
v1.0

The greatest gift is that of time. This is my attempt to give you back some of yours.

History of Network Management

West Virginia NETWORK
Enhancing Education through Technology

History of Network Management

- SNMP

History of Network Management

- SNMP

“Simple” Network Management Protocol

History of Network Management

- SNMP

"Simple" Network Management Protocol

- Oh, and "screen scraping"

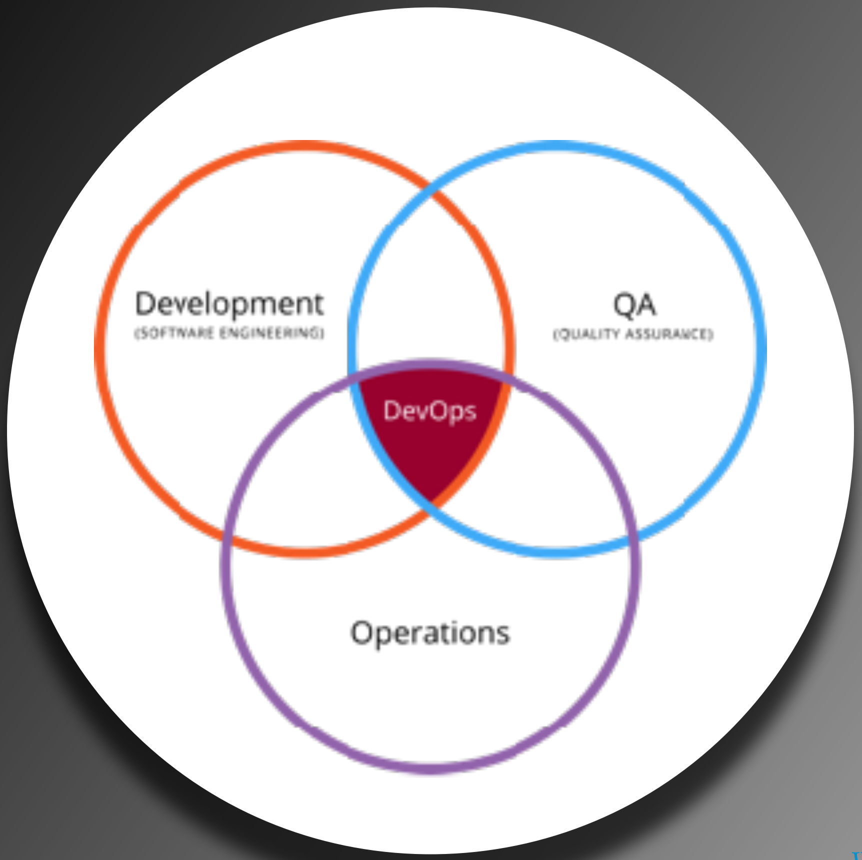
DevOps

West Virginia NETWORK
Enhancing Education through Technology

What is this DevOps of which you speak?

- “DevOps (a clipped compound of "development" and "operations") is a software engineering practice that aims at unifying software development (Dev) and software operation (Ops).”

Source: https://en.wikipedia.org/wiki/DevOps



West Virginia NETWORK
Enhancing Education through Technology

In Plain English?

In Plain English?

The love child between systems/network administrators and programmers

Configuration Management Tools

West Virginia NETWORK
Enhancing Education through Technology

CFEngine

CHEF™

Microsoft System Center Configuration Manager

ANSIBLE

munki

SALTSTACK

puppet labs

West Virginia NETWORK
Enhancing Education through Technology

CFEngine → THE C PROGRAMMING LANGUAGE

CFEngine → THE C PROGRAMMING LANGUAGE

CHEF™ puppet labs® → Ruby

CFEngine → THE C PROGRAMMING LANGUAGE

CHEF™ puppet labs → Ruby

ANSIBLE SALTSTACK → python™

West Virginia NETWORK
Enhancing Education through Technology

So Why Ansible?

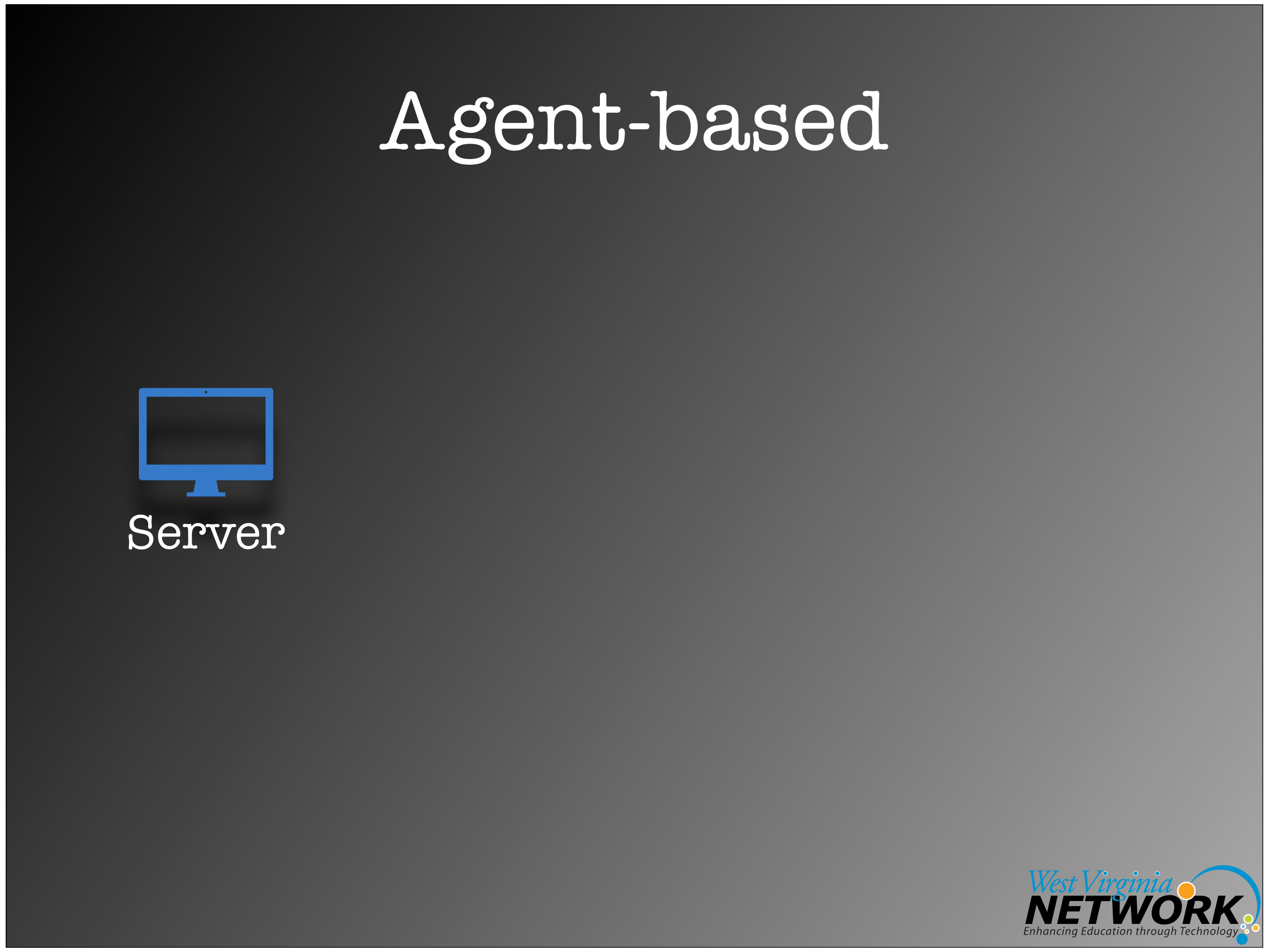
Ansible

The name "Ansible" references a fictional instantaneous hyperspace communication system (as featured in Orson Scott Card's **Ender's Game** (1985),[9][10] and originally conceived by Ursula K. Le Guin for her novel Rocannon's World (1966)).[11]

Source: https://en.wikipedia.org/wiki/Ansible_(software)

Agent-based vs. Agent-less*

Agent-based	Agent-less
• CFEngine	• Ansible
• Chef	
• Munki	
• Puppet	
• SaltStack	



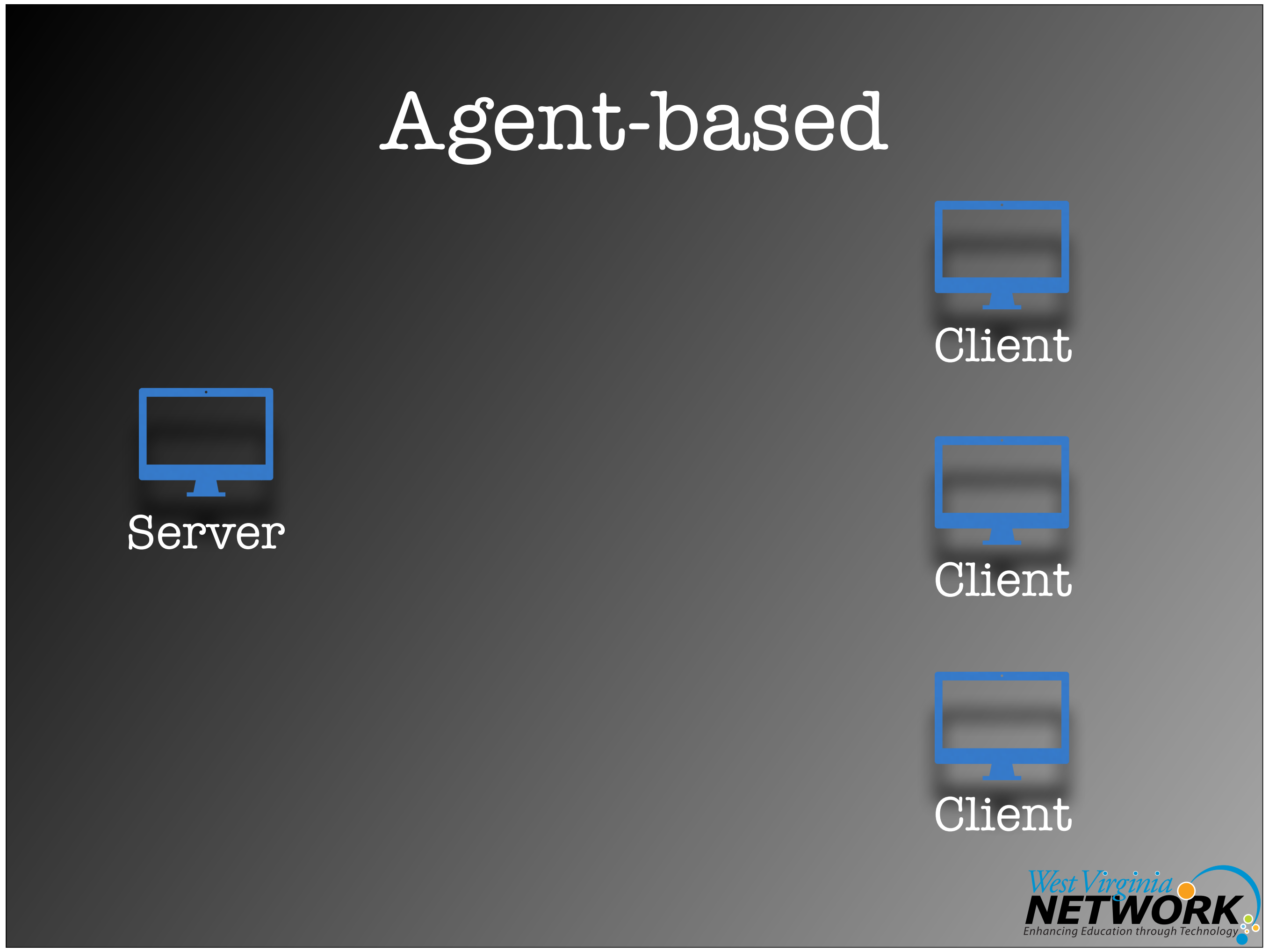
Agent-based

Server

Terms:
Server == Puppet Master, Salt Master, etc.
Client== Puppet Agent, Salt Minion, etc.
Configuration files == (Puppet) catalog, Salt States (SLS), etc.

Also have terms like grains, pillars, etc. for Salt, for example.

Typically agents check-in every so often—default for Puppet is every 15 minutes, for Munki is once every 4 hours—to make sure they are up-to-date.

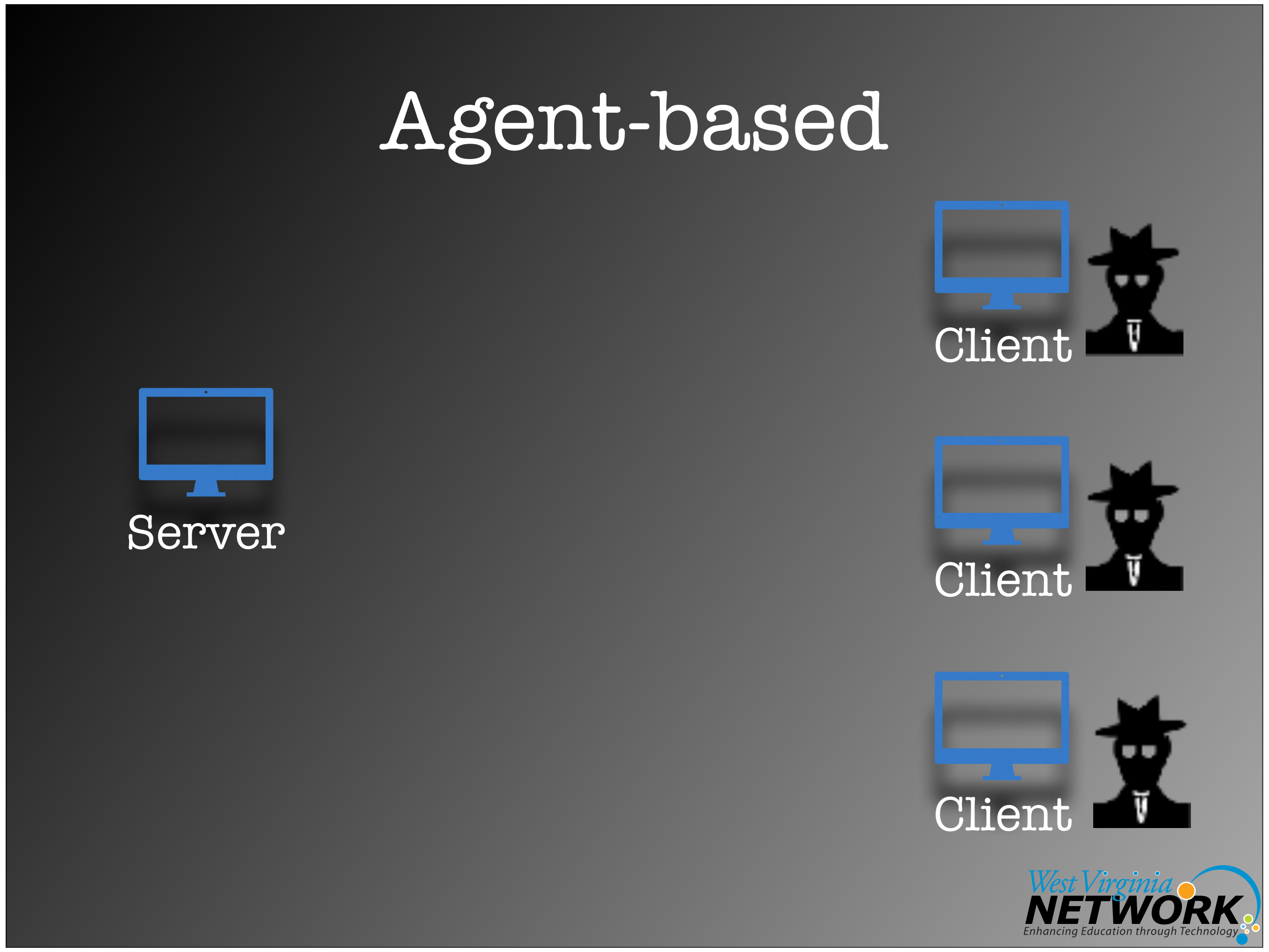


Agent-based

Terms:
Server == Puppet Master, Salt Master, etc.
Client== Puppet Agent, Salt Minion, etc.
Configuration files == (Puppet) catalog, Salt States (SLS), etc.

Also have terms like grains, pillars, etc. for Salt, for example.

Typically agents check-in every so often—default for Puppet is every 15 minutes, for Munki is once every 4 hours—to make sure they are up-to-date.

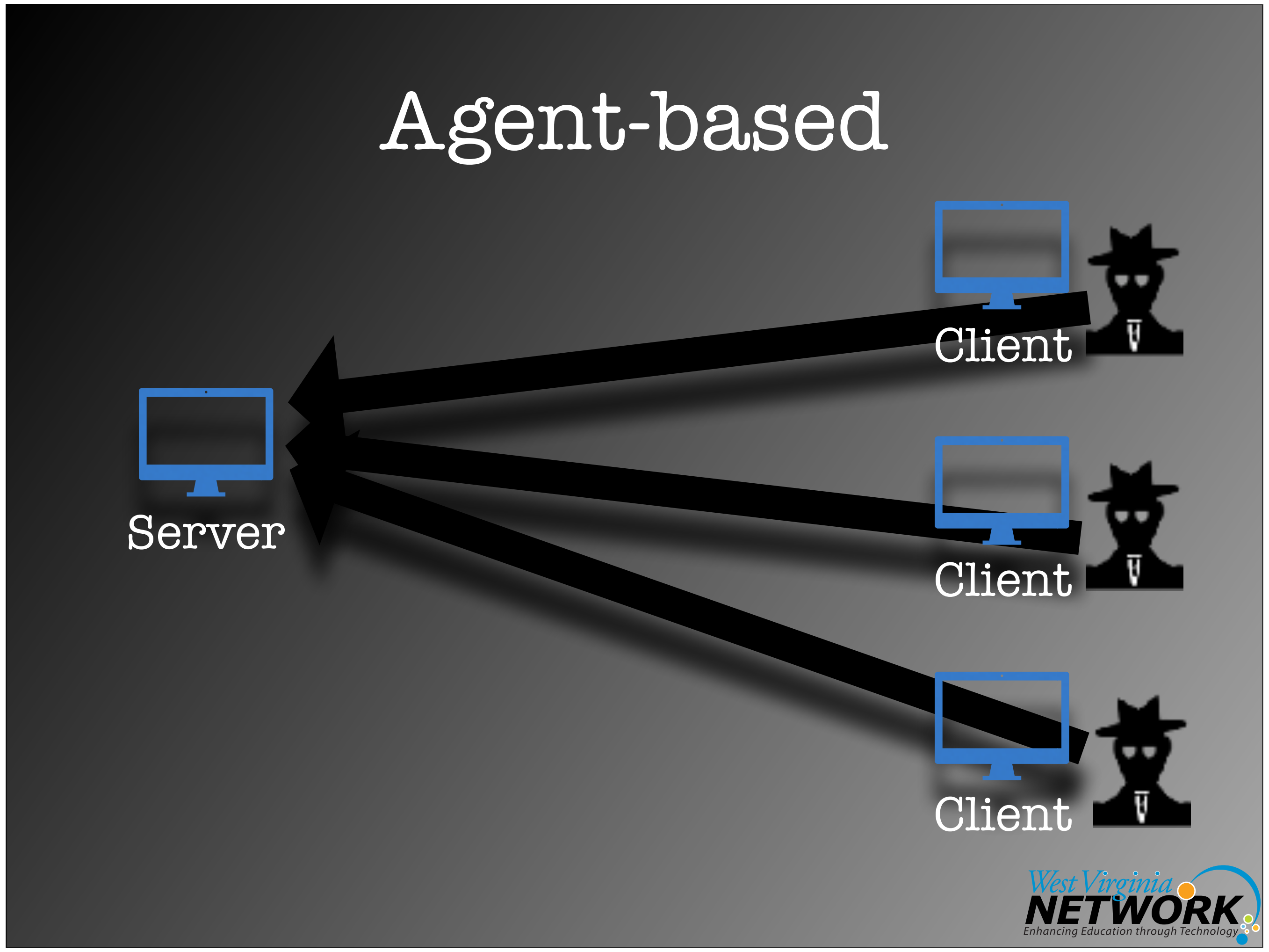


Agent-based

Terms:
Server == Puppet Master, Salt Master, etc.
Client== Puppet Agent, Salt Minion, etc.
Configuration files == (Puppet) catalog, Salt States (SLS), etc.

Also have terms like grains, pillars, etc. for Salt, for example.

Typically agents check-in every so often—default for Puppet is every 15 minutes, for Munki is once every 4 hours—to make sure they are up-to-date.



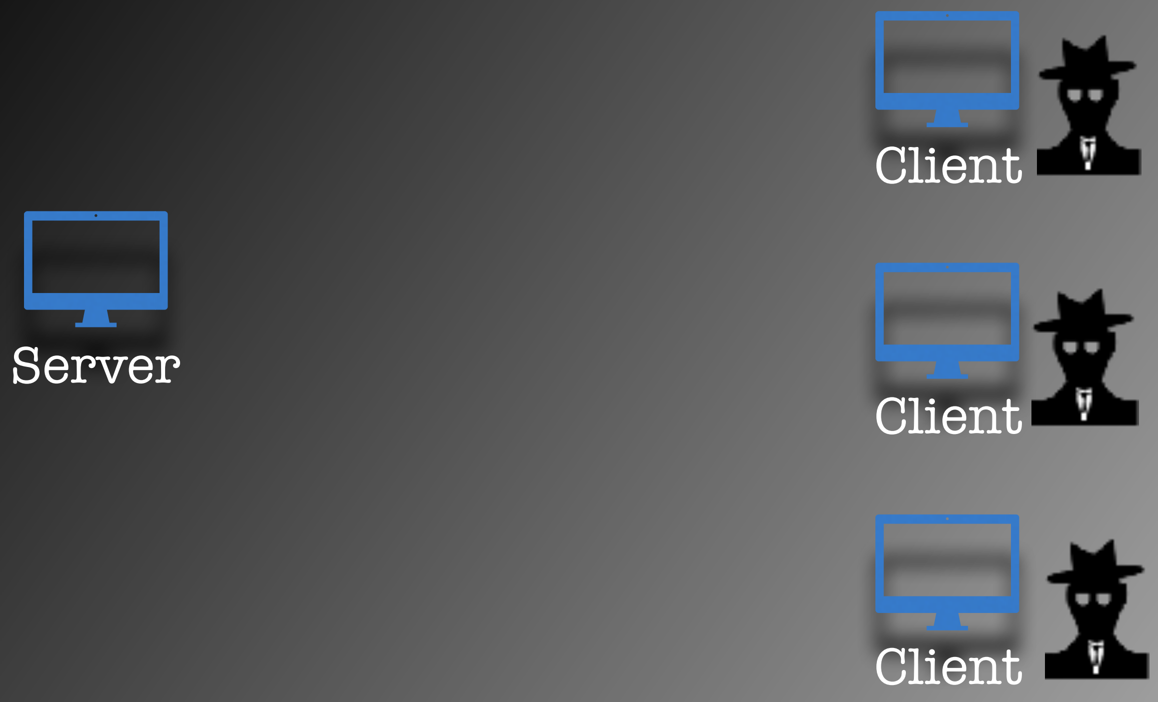
Agent-based

Terms:
Server == Puppet Master, Salt Master, etc.
Client== Puppet Agent, Salt Minion, etc.
Configuration files == (Puppet) catalog, Salt States (SLS), etc.

Also have terms like grains, pillars, etc. for Salt, for example.

Typically agents check-in every so often—default for Puppet is every 15 minutes, for Munki is once every 4 hours—to make sure they are up-to-date.

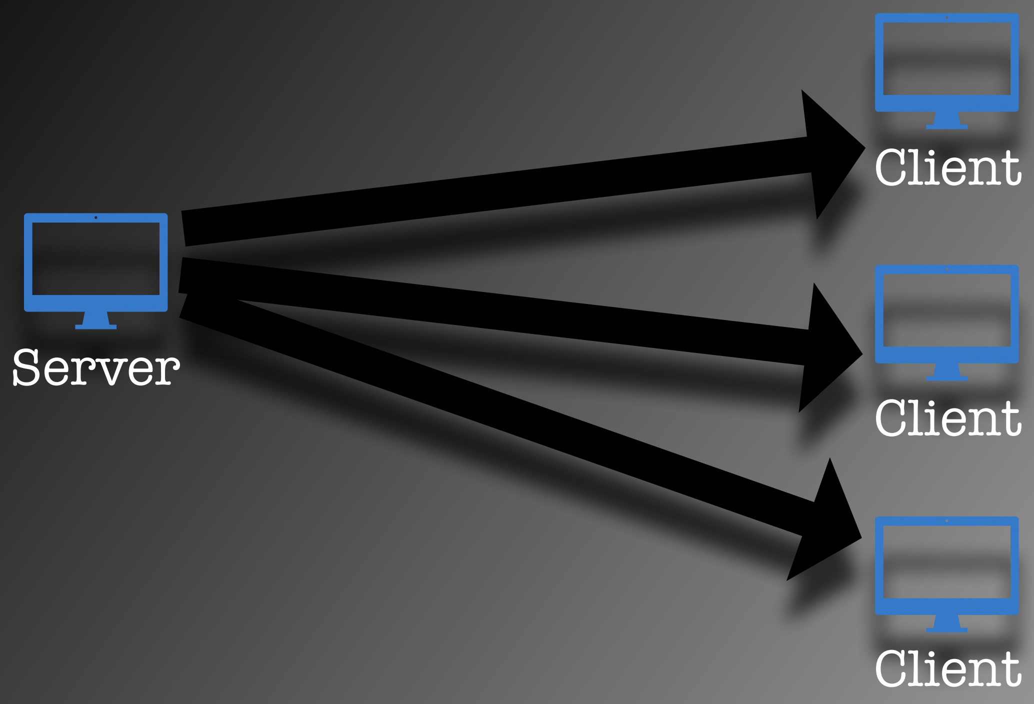
Agent-less



Agent-less

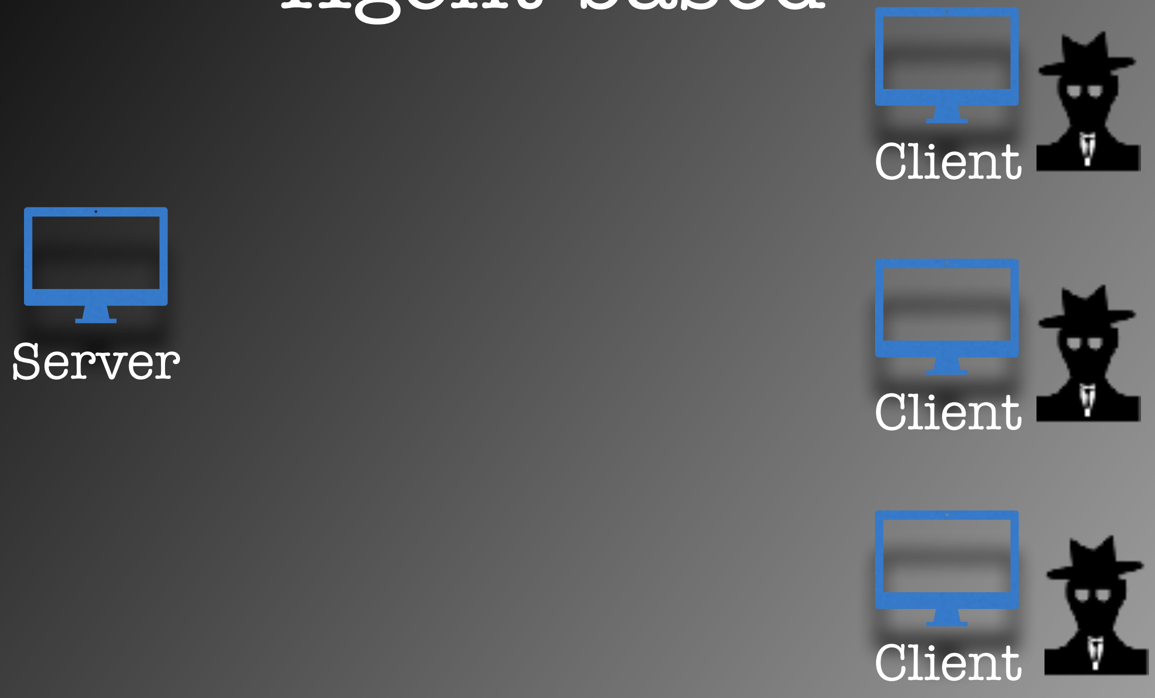
Client
Server
Client
Client

Agent-less



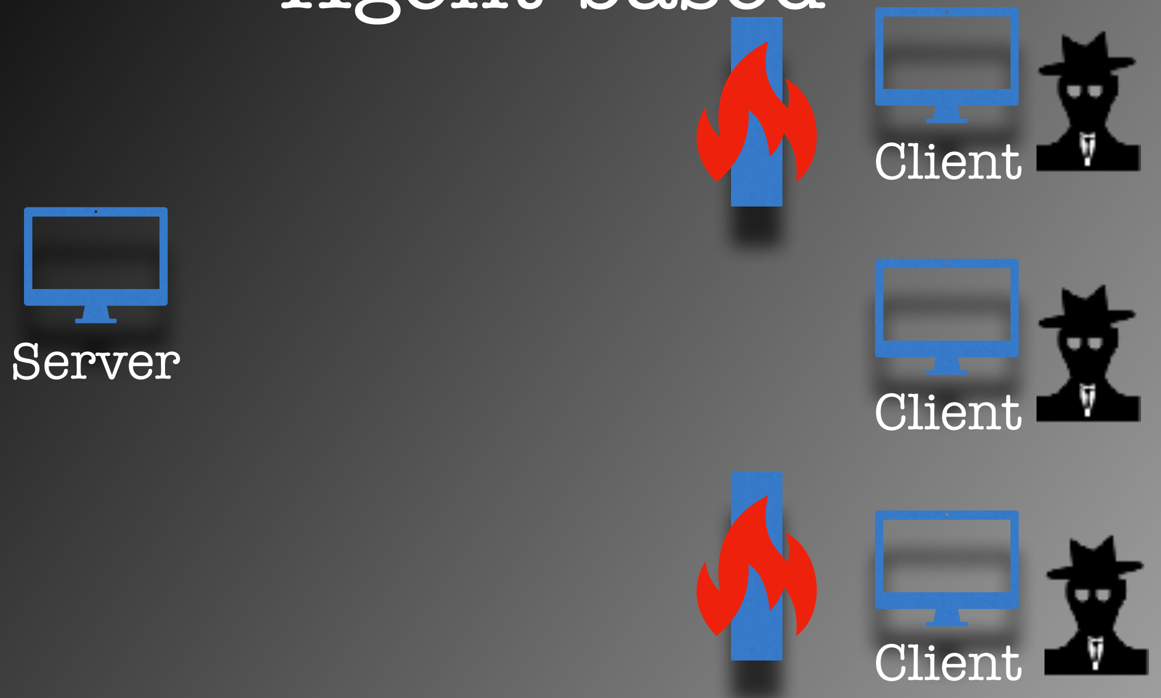
West Virginia NETWORK Enhancing Education through Technology

Advantages of Agent-based



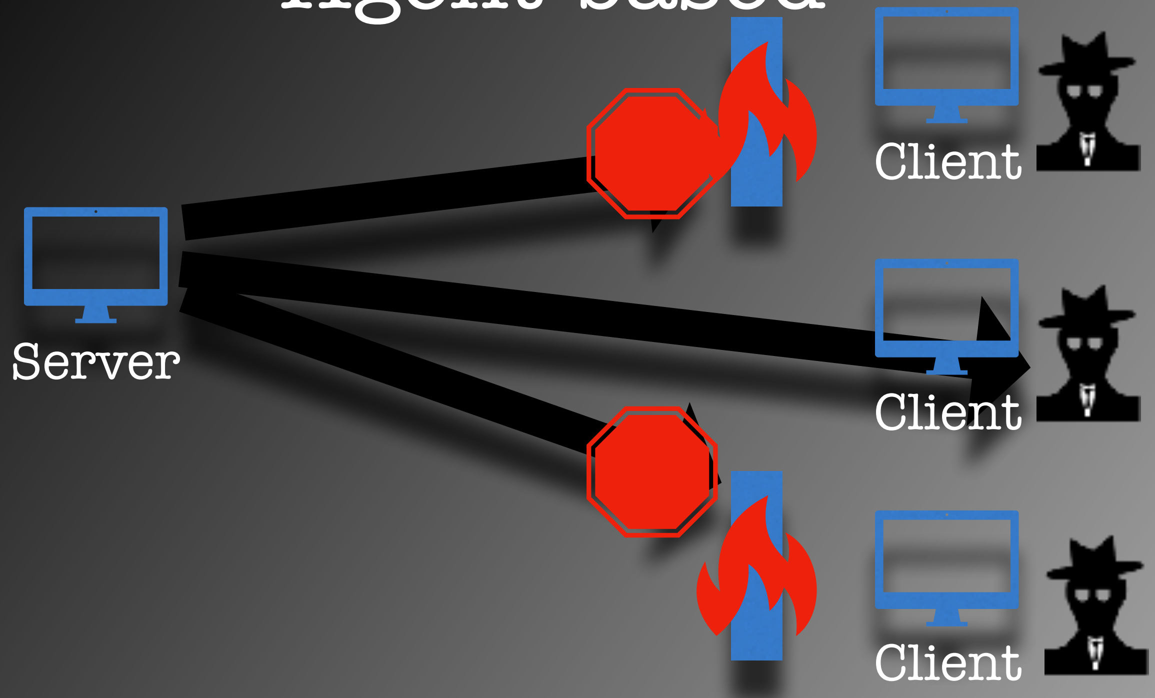
West Virginia NETWORK
Enhancing Education through Technology

Advantages of Agent-based

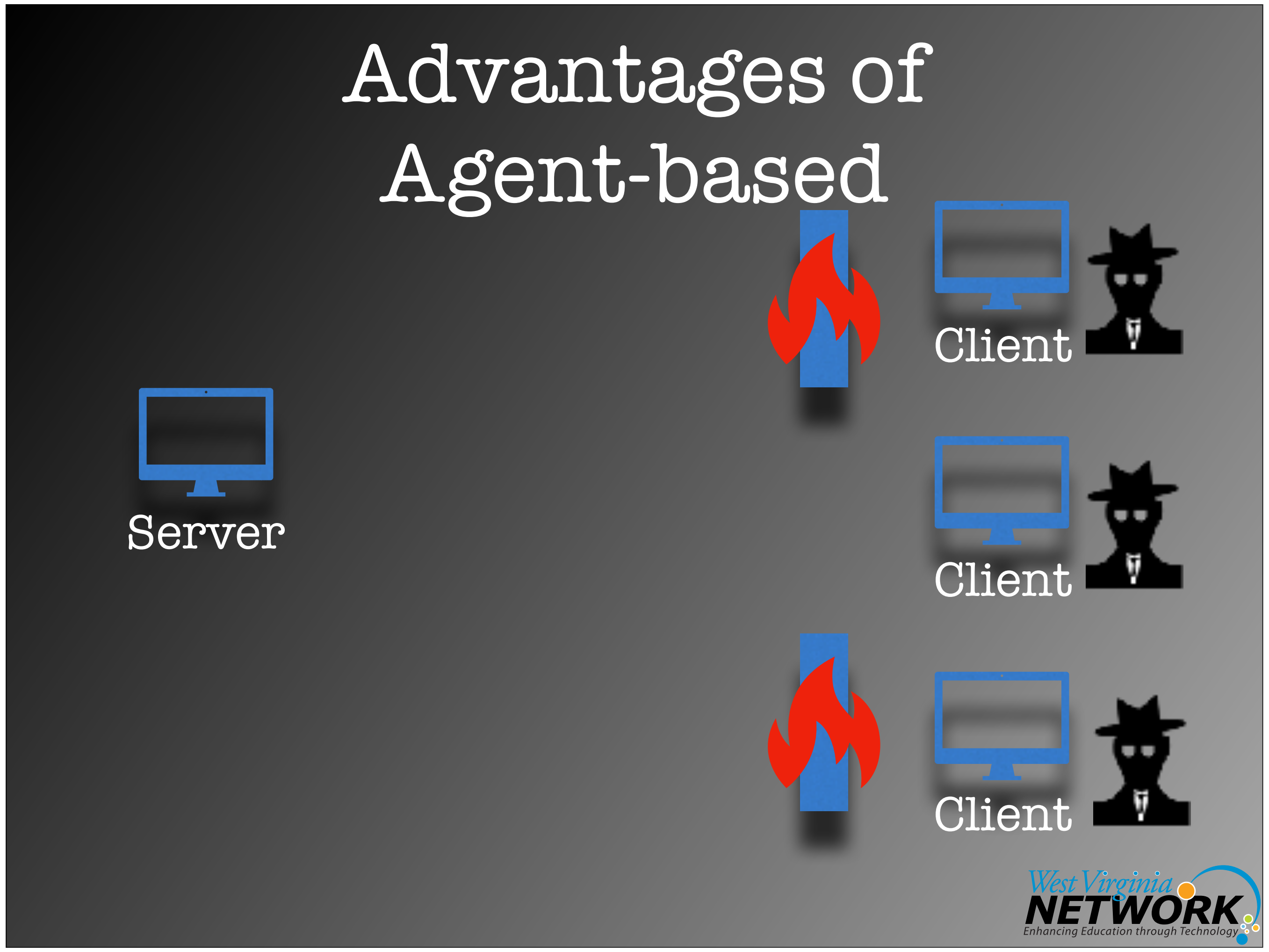


West Virginia NETWORK
Enhancing Education through Technology

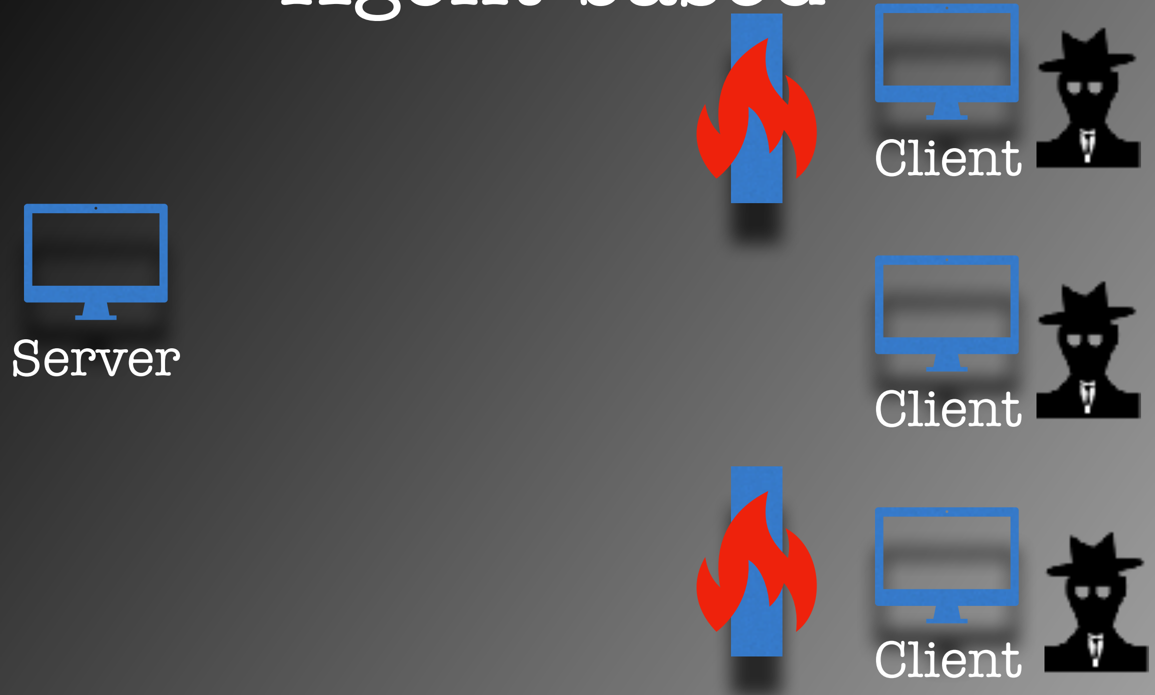
Advantages of Agent-based



West Virginia NETWORK
Enhancing Education through Technology



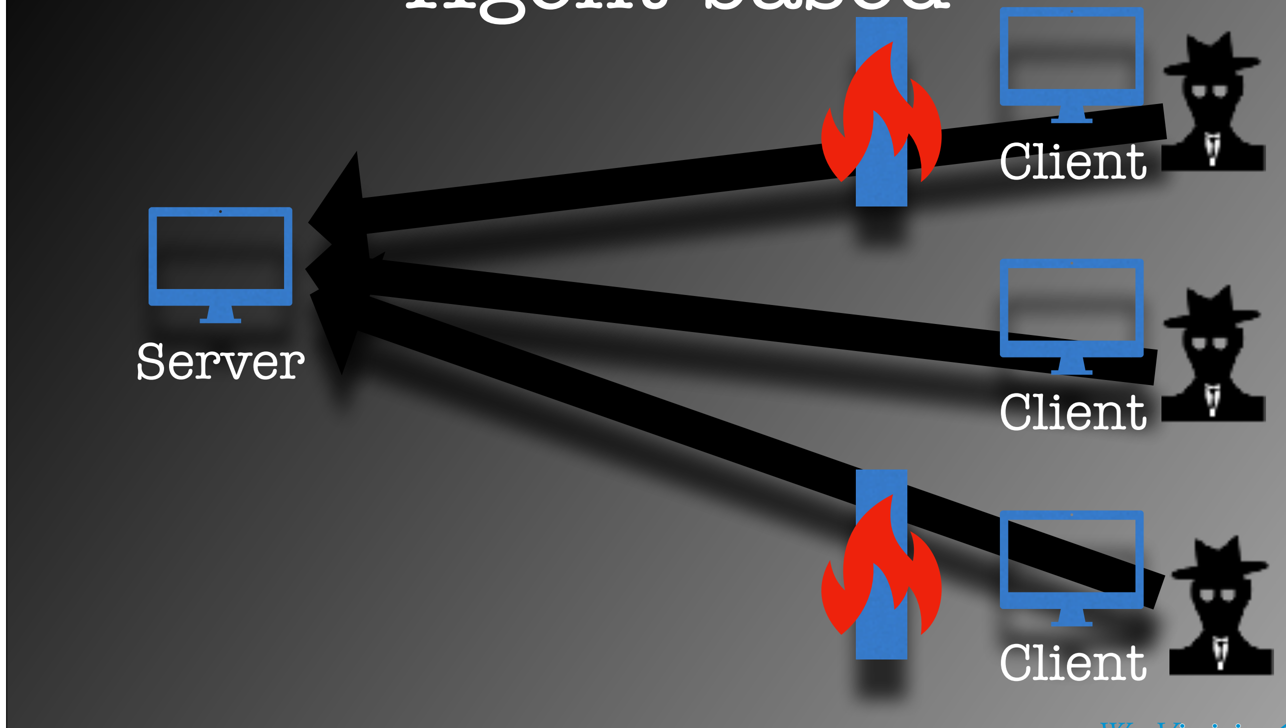
Advantages of Agent-based



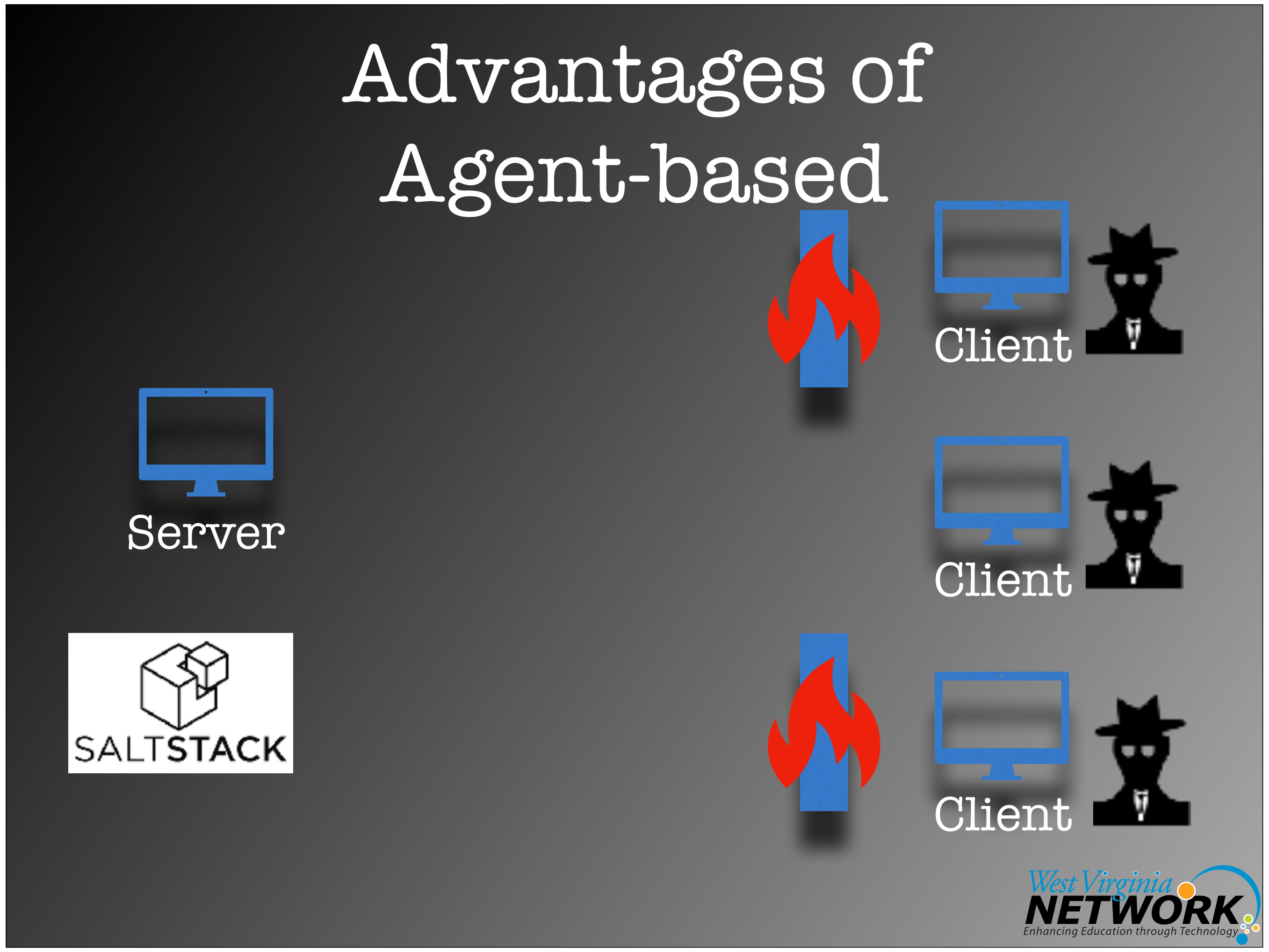
West Virginia NETWORK
Enhancing Education through Technology

Typically agents check in, thus coming out through any firewalls vs. the server trying to come in. Of course, in a tightly regulated environment with proxy servers, etc., this may require additional work, but often things “just work.”

Advantages of Agent-based



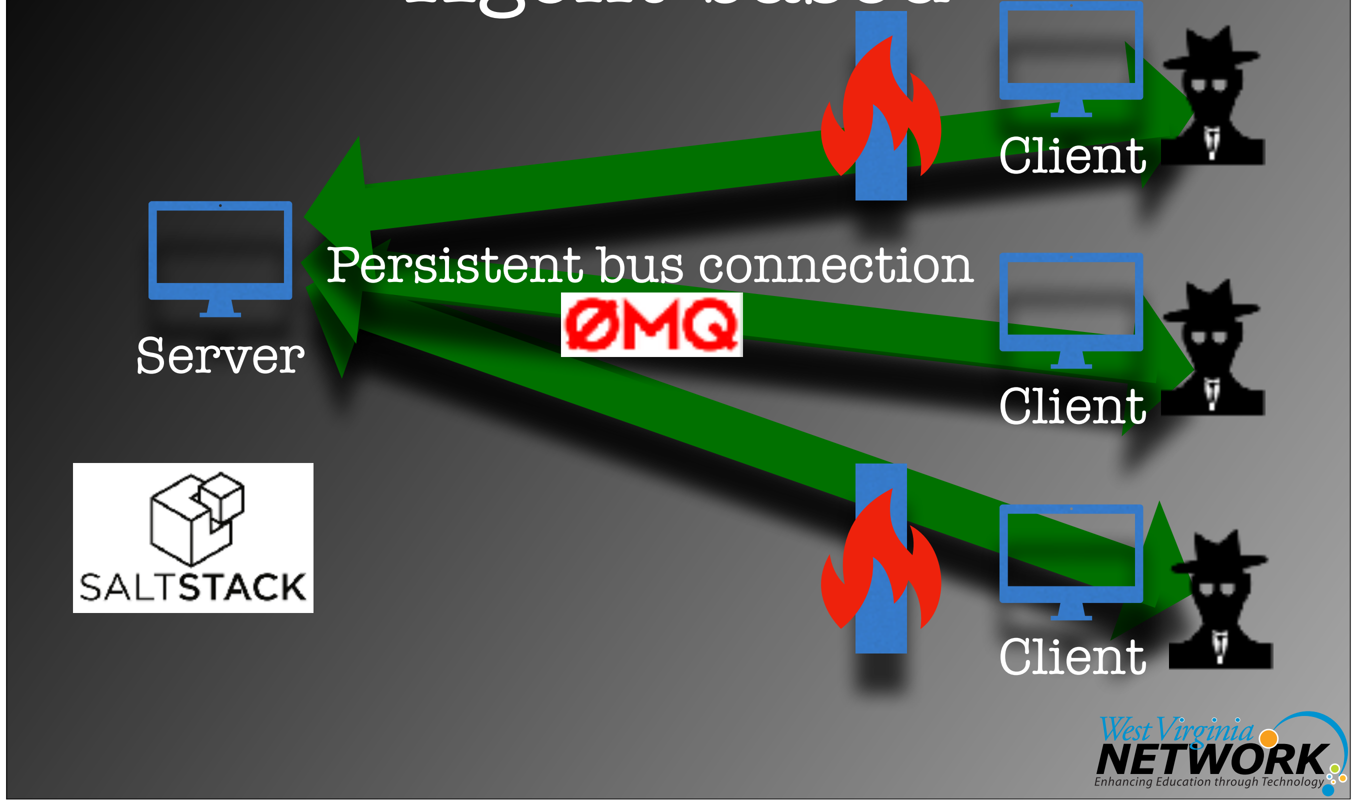
Typically agents check in, thus coming out through any firewalls vs. the server trying to come in. Of course, in a tightly regulated environment with proxy servers, etc., this may require additional work, but often things "just work."



Advantages of Agent-based

Salt Stack is different from other agent-based configuration management tools in that it creates a persistent connection back to the minions. This allows for immediate execution of commands.
For example, you get a call that some of your users are experiencing issues getting to Google. With Salt, you could tell all of your minions to ping Google's servers and to report back. This gives you insight from across your network (and also gives you a kind of botnet of your very own!).

Advantages of Agent-based


Server
Persistent bus connection
ØMQ
Client
Client
Client
SALTSTACK
West Virginia NETWORK
Enhancing Education through Technology

Salt Stack is different from other agent-based configuration management tools in that it creates a persistent connection back to the minions. This allows for immediate execution of commands.
For example, you get a call that some of your users are experiencing issues getting to Google. With Salt, you could tell all of your minions to ping Google's servers and to report back. This gives you insight from across your network (and also gives you a kind of botnet of your very own!).

Advantages of Agent-less

Client
Server
Client
Client
ANSIBLE

West Virginia NETWORK
Enhancing Education through Technology

Advantages of Agent-less

Server
ANSIBLE

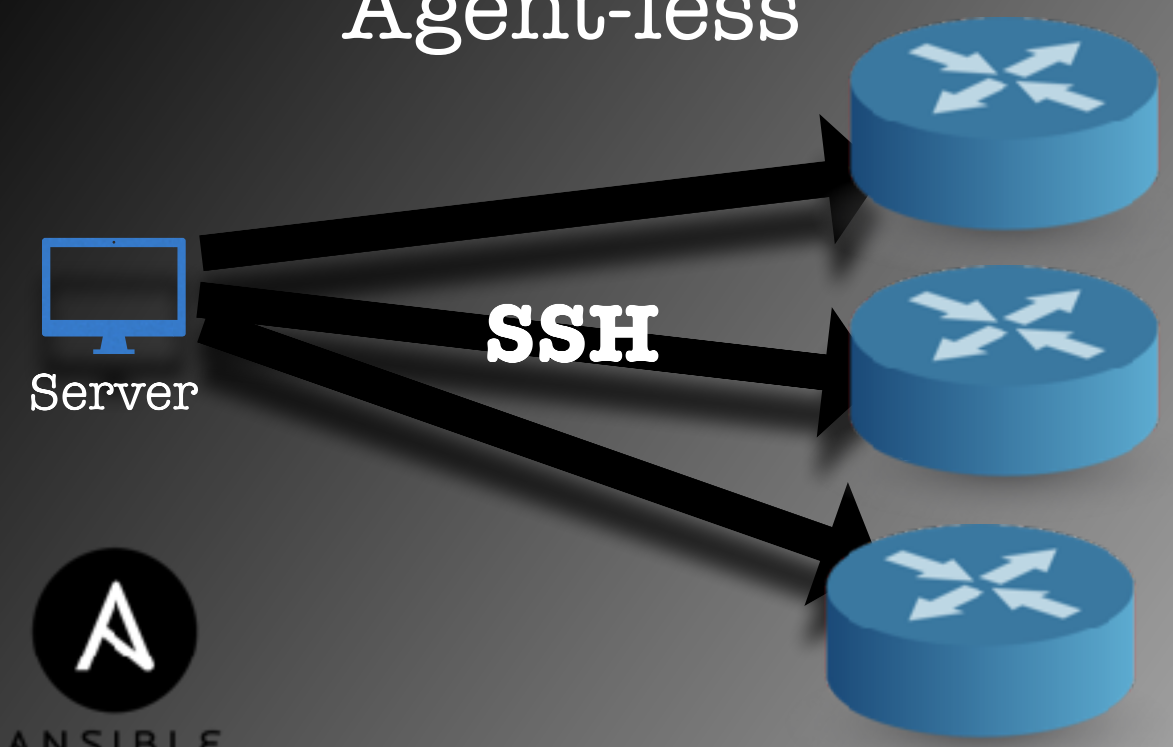
West Virginia NETWORK
Enhancing Education through Technology

Advantages of Agent-less

Server
ANSIBLE

West Virginia NETWORK
Enhancing Education through Technology

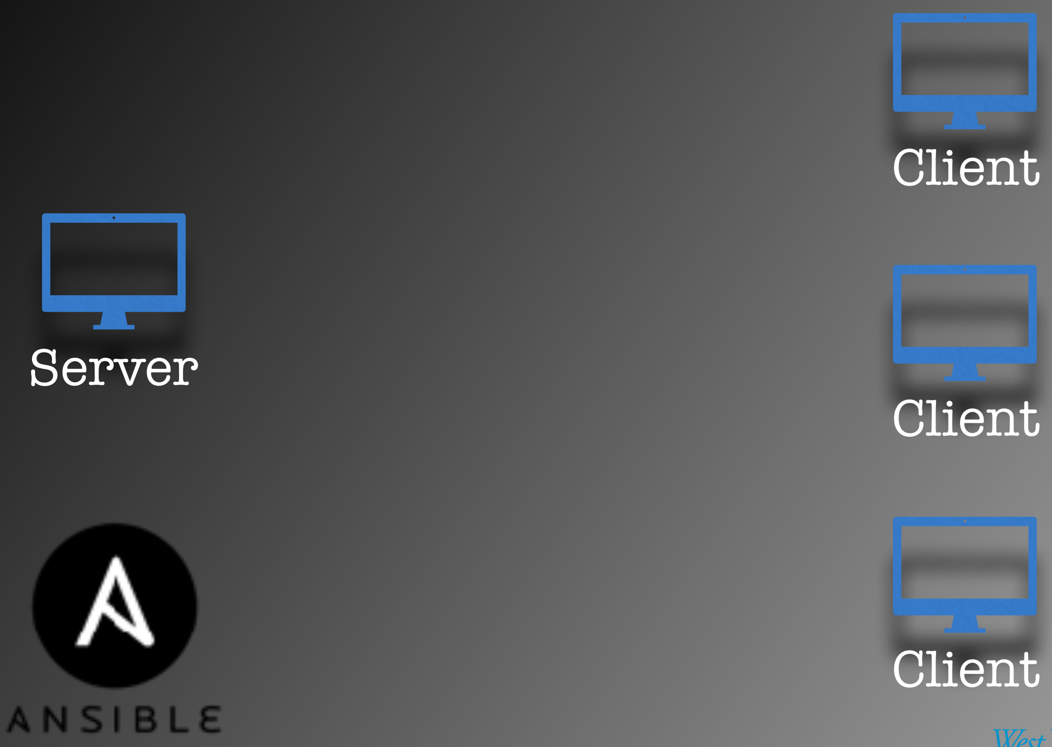
Advantages of Agent-less



ANSIBLE

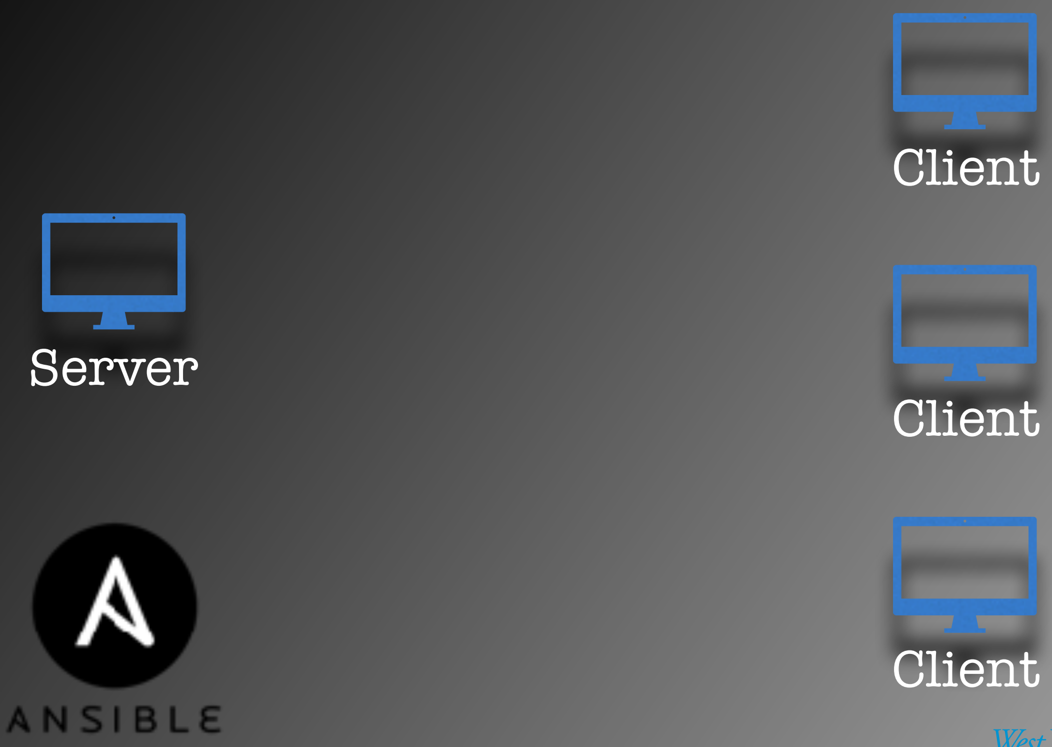
West Virginia NETWORK
Enhancing Education through Technology

Agent-less

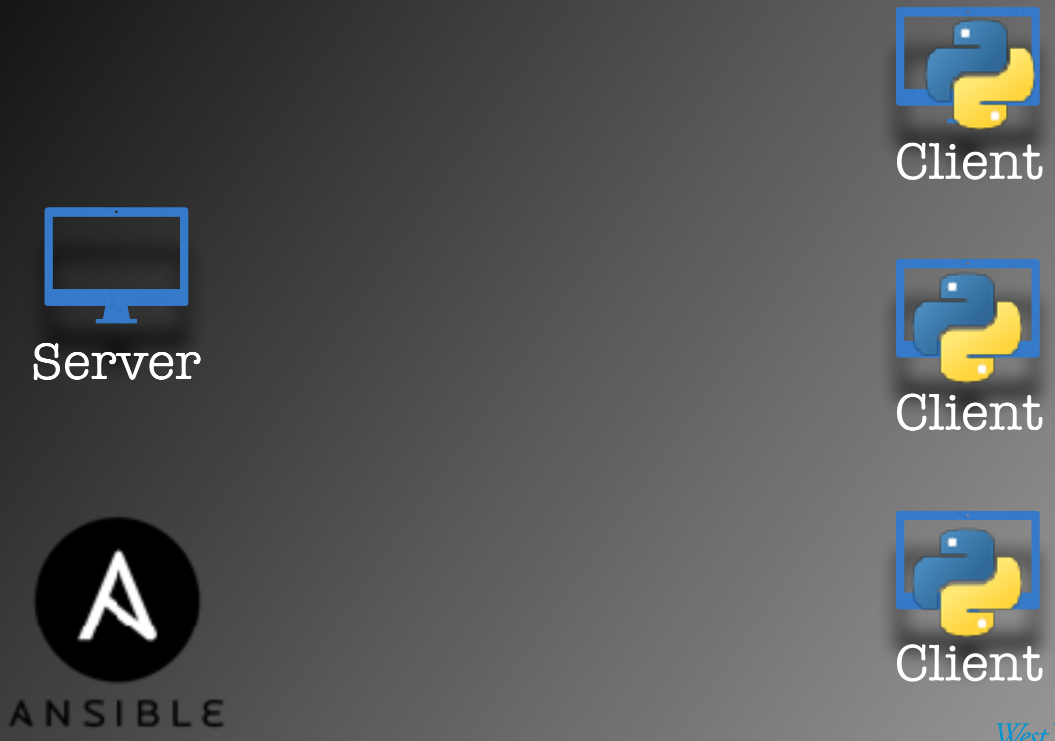


West Virginia NETWORK
Enhancing Education through Technology

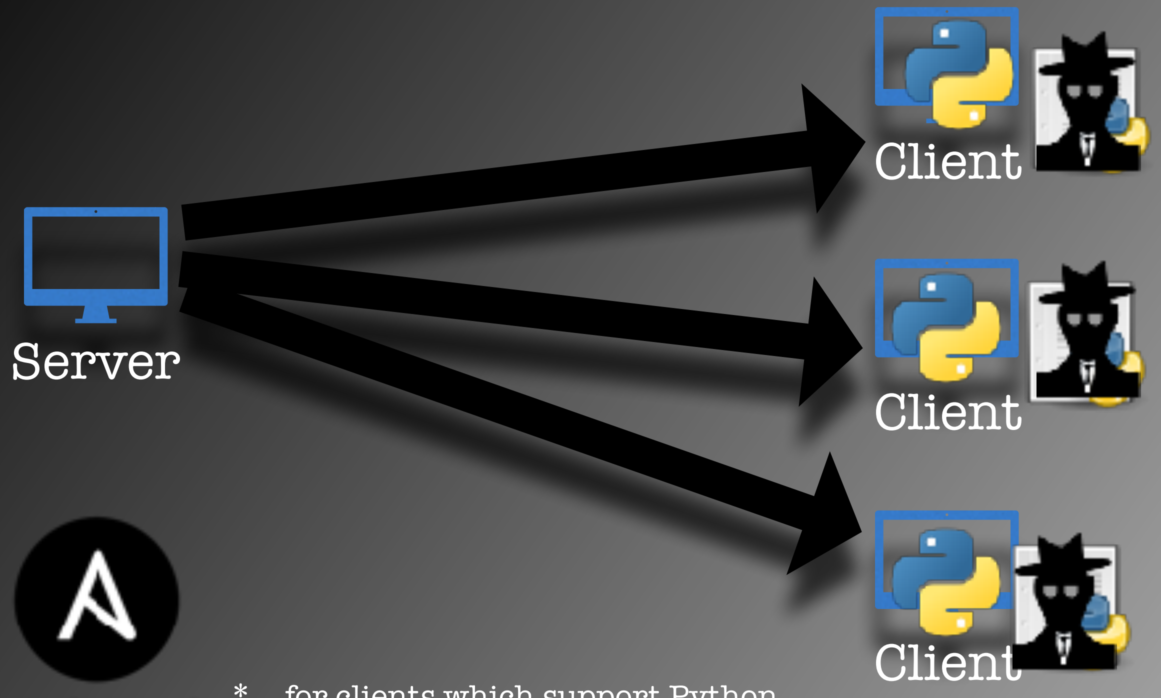
Agent-less*



Agent-less*



Agent-less*

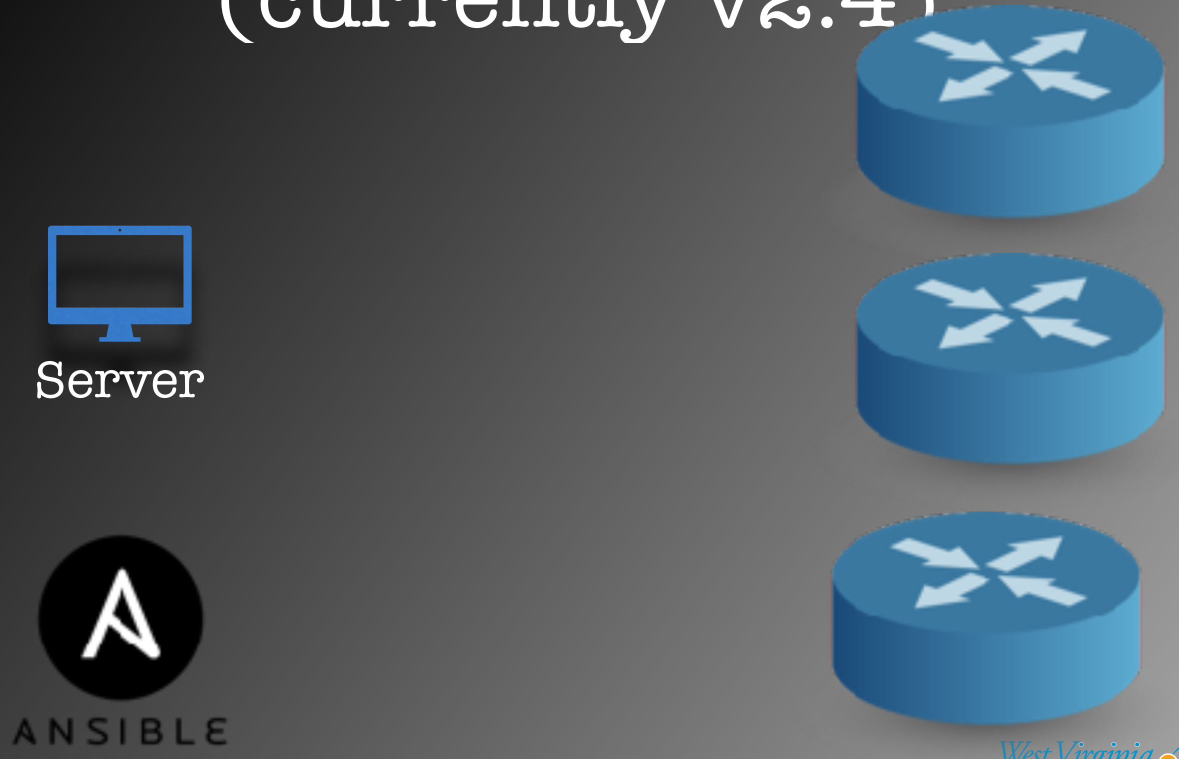


ANSIBLE

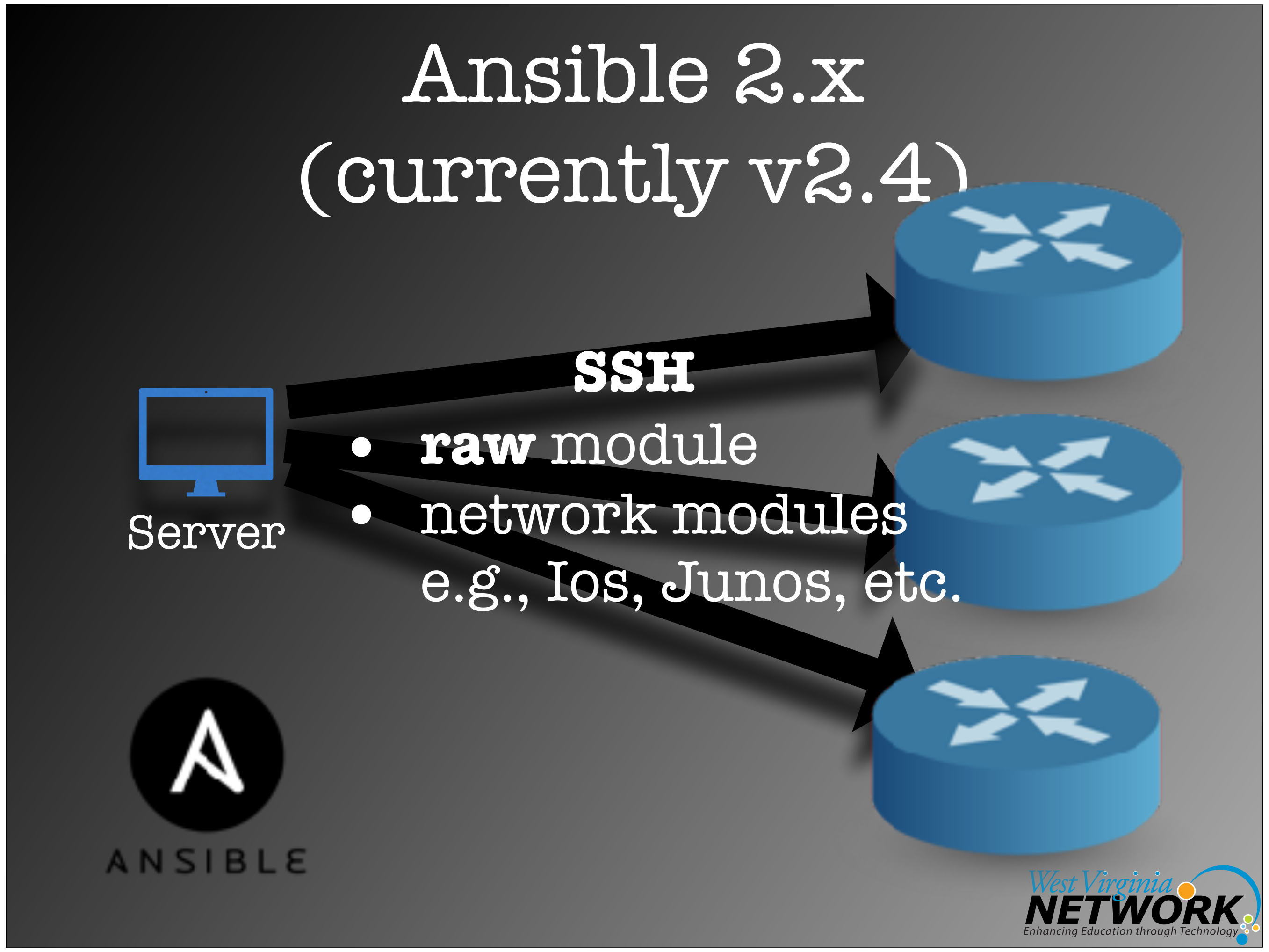
* for clients which support Python, agent script sent through SSH tunnel to run on far end

West Virginia NETWORK
Enhancing Education through Technology

Ansible 2.x
(currently v2.4)



Ansible 2.x (currently v2.4)



SSH

- **raw** module
- network modules e.g., Ios, Junos, etc.

Network Modules

- A10
- ACI (Cisco)
- Aireos (Cisco)
- Aos
- Aruba
- Asa (Cisco)
- Avi
- Bigswitch
- Citrix
- Cloudengine
- Cloudvision (Arista)
- Cumulus
- Dellos10
- Dellos6
- Dellos9
- Eos (Arista)
- F5
- Fortios
- Illumos
- Interface
- Ios (Cisco)
- Iosxr (Cisco)
- Junos
- **Layer2**
- **Layer3**
- Lenovo
- Netconf
- Netscaler
- Netvisor
- Nuage
- Nxos (Cisco)
- Ordnance
- Ovs
- Panos
- **Protocol**
- Radware
- **Routing**
- Sros
- **System**
- Vyos

Source: http://docs.ansible.com/ansible/latest/list_of_network_modules.html

Network Modules

- A10
- ACI (Cisco)
- Aireos (Cisco)
- Aos
- Aruba
- Asa (Cisco)
- Avi
- Bigswitch
- Citrix
- Cloudengine
- Cloudvision (Arista)
- Cumulus
- Dellos10
- Dellos6
- Dellos9
- Eos (Arista)
- F5
- Fortios
- Illumos
- Interface
- Ios (Cisco)
- Iosxr (Cisco)
- Junos
- **Layer2**
- **Layer3**
- Lenovo
- Netconf
- Netscaler
- Netvisor
- Nuage
- Nxos (Cisco)
- Ordnance
- Ovs
- Panos
- **Protocol**
- Radware
- **Routing**
- Sros
- **System**
- Vyos

Source: http://docs.ansible.com/ansible/latest/list_of_network_modules.html

Network Modules (cont.)

Cisco IOS

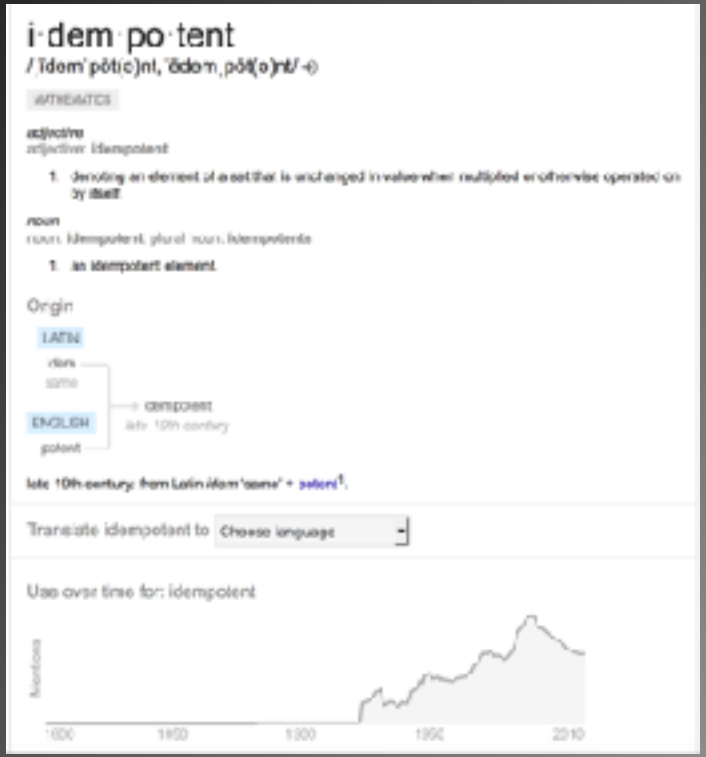
- Ios
  - **ios_banner** - Manage multiline banners on Cisco IOS devices
  - **ios_command** - Run commands on remote devices running Cisco IOS
  - **ios_config** - Manage Cisco IOS configuration sections
  - **ios_facts** - Collect facts from remote devices running Cisco IOS
  - **ios_interface** - Manage Interface on Cisco IOS network devices
  - **ios_logging** - Manage logging on network devices
  - **ios_ping** - Tests reachability using ping from IOS switch
  - **ios_static_route** - Manage static IP routes on Cisco IOS network devices
  - **ios_system** - Manage the system attributes on Cisco IOS devices
  - **ios_user** - Manage the aggregate of local users on Cisco IOS device
  - **ios_vrf** - Manage the collection of VRF definitions on Cisco IOS devices

Source: http://docs.ansible.com/ansible/latest/list_of_network_modules.html

I am NOT idempotent!
Wait... what?

West Virginia NETWORK
Enhancing Education through Technology

Idempotent



i·dem·po·tent
/ˈīdəmˈpōt(ə)nt, ˈēdəm,pōt(ə)nt/

MATHEMATICS

adjective
adjective: idempotent

1. denoting an element of a set that is unchanged in value when multiplied or otherwise operated on by itself.

noun
noun: idempotent; plural noun: idempotents

1. an idempotent element.

Origin

LATIN
idem
same

ENGLISH
potent

idempotent
late 19th century

late 19th century: from Latin *idem* 'same' + potent[1].

Translate idempotent to Choose language

Use over time for: idempotent

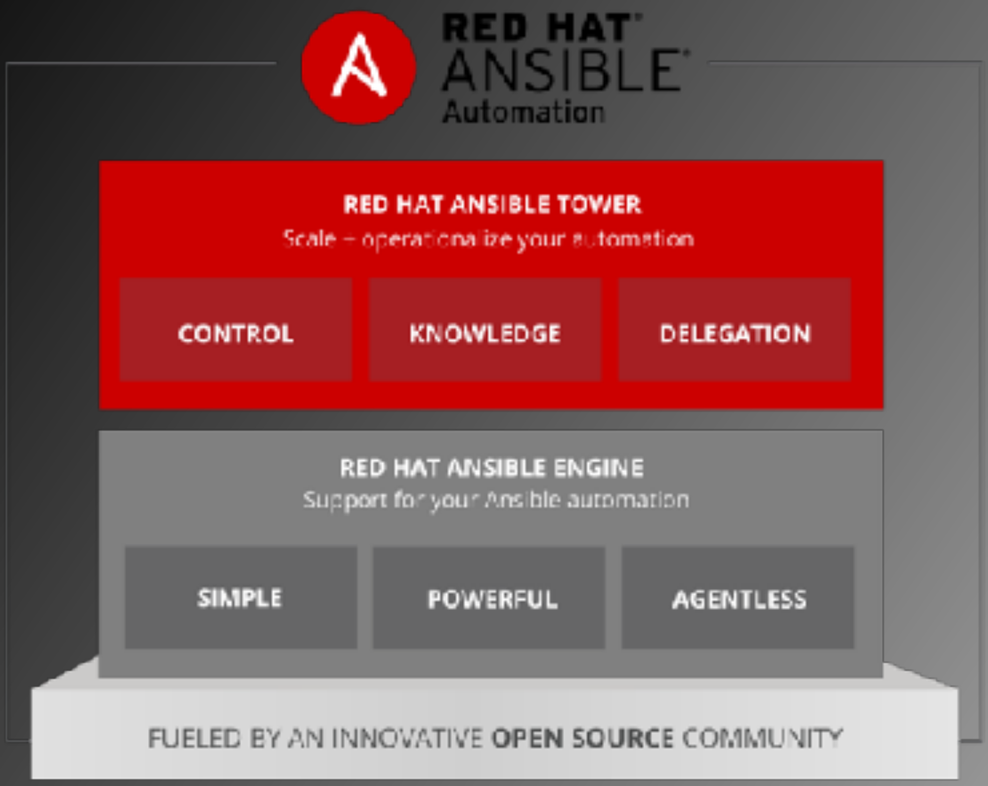
Source: "The Google"

ANSIBLE

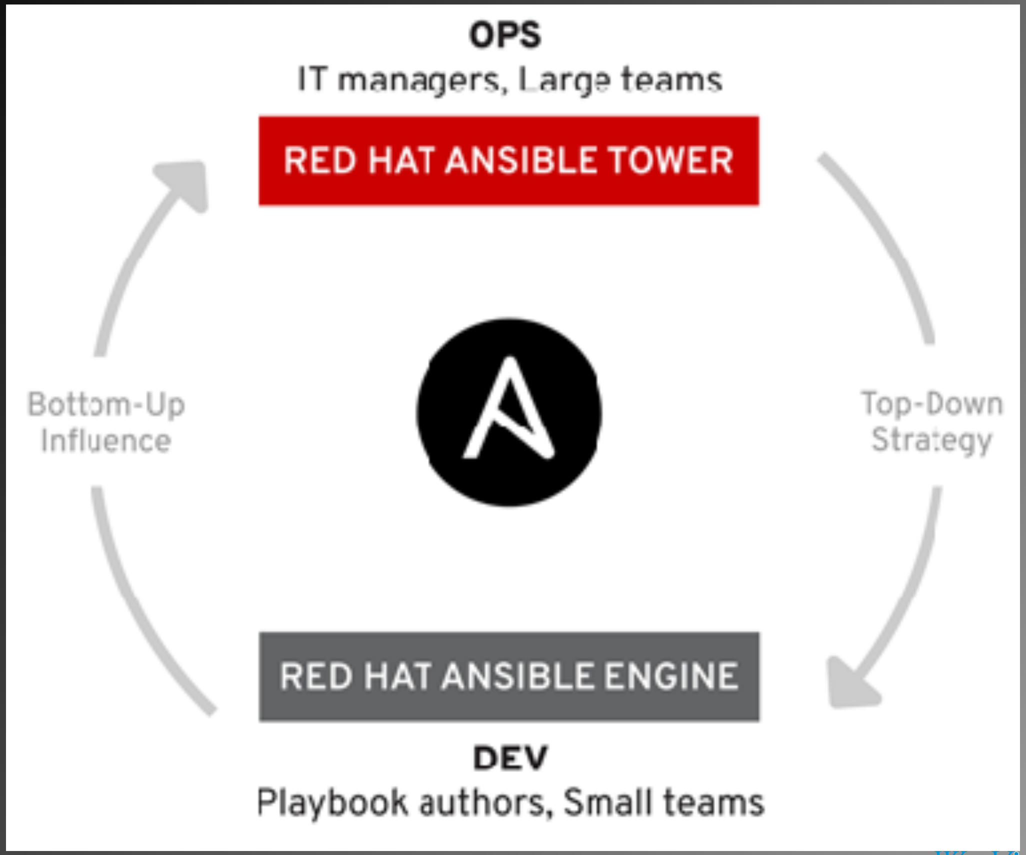
redhat

ANSIBLE

West Virginia NETWORK
Enhancing Education through Technology


RED HAT ANSIBLE Automation
RED HAT ANSIBLE TOWER
Scale – operationalize your automation
CONTROL
KNOWLEDGE
DELEGATION
RED HAT ANSIBLE ENGINE
Support for your Ansible automation
SIMPLE
POWERFUL
AGENTLESS
FUELED BY AN INNOVATIVE OPEN SOURCE COMMUNITY

West Virginia NETWORK
Enhancing Education through Technology



West Virginia NETWORK
Enhancing Education through Technology

Red Hat Ansible

Ansible (source)	Red Hat Ansible Engine
AWX	Red Hat Ansible Tower

West Virginia NETWORK
Enhancing Education through Technology

Red Hat Ansible

Ansible (source)	Red Hat Ansible Engine
AWX	Red Hat Ansible Tower
Fedora	RHEL

So THAT's why Ansible

Live Demo

Deeper Dive

West Virginia NETWORK
Enhancing Education through Technology

System Requirements

System Requirements

- **Control Machine Requirements**
  - Currently Ansible can be run from any machine with Python 2 (versions 2.6 or 2.7) or Python 3 (versions 3.5 and higher) installed (Windows isn’t supported for the control machine).

System Requirements

- **Control Machine Requirements**
  - Currently Ansible can be run from any machine with Python 2 (versions 2.6 or 2.7) or Python 3 (versions 3.5 and higher) installed (Windows isn't supported for the control machine).
- **Managed Node Requirements**
  - On the managed nodes, you need a way to communicate, which is normally ssh. By default this uses sftp. If that's not available, you can switch to scp in ansible.cfg. You also need Python 2.6 or later.

Source: http://docs.ansible.com/ansible/latest/intro_installation.html#control-machine-requirements

West Virginia NETWORK
Enhancing Education through Technology

Installing Ansible

Installing Ansible

- Yum (CENTOS/RHEL)
- Apt (Ubuntu/Debian)
- Pip

Installing Ansible

- Yum (CENTOS/RHEL)
- Apt (Ubuntu/Debian)
- Pip

```
$ sudo easy_install pip
$ sudo pip install ansible
```

Installing Ansible

- Yum (CENTOS/RHEL)
- Apt (Ubuntu/Debian)
- Pip

```
$ sudo easy_install pip
$ sudo pip install ansible


If for any reason you have issues, try:
$ sudo -H pip install --ignore-installed --upgrade ansible
```

Running Ansible

Running Ansible

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

Running Ansible

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k


$ ansible 10.1.1.1 -m raw -a "command" -u <user> -k
```

Running Ansible

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -m raw -a "command" -u <user> -k
```

FAILS.

Running Ansible

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -m raw -a "command" -u <user> -k
```

FAILS.

No inventory file. This is a minimum requirement.

Running Ansible

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -m raw -a "command" -u <user> -k
```

FAILS.

No inventory file. This is a minimum requirement.

So we need to create an inventory file.

Running Ansible

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -m raw -a "command" -u <user> -k
```

FAILS.

No inventory file. This is a minimum requirement.

So we need to create an inventory file.

Inventory files are plain text files which contain a list of devices which you intend to manage with Ansible. It can be as simple as a straight list of IP addresses. Inventory files can be formatted in different ways, but a common one is the Windows INI file format. The other common format is YAML, which is also the format used to write Ansible Playbooks.

Simple Inventory File

```
10.1.1.1
10.1.1.2
10.1.1.3
node1.domain.com
node2.domain.com
…
last.item.com
```

Inventory File

```
[routers:children]
backbone-routers
gateway-routers

[backbone-routers]
backbone1  ansible_host=10.1.1.1
backbone2  ansible_host=10.1.1.2
backbone3  ansible_host=10.1.1.3

[gateway-routers]
gateway1   ansible_host=10.1.2.1
gateway2   ansible_host=10.1.2.2

[switches]
switch1    ansible_host=10.1.3.1
switch2    ansible_host=10.1.3.2
switch3    ansible_host=10.1.3.3
10.1.4.1
10.1.5.1
```

Inventory File

```
[routers:children]
backbone-routers
gateway-routers

[backbone-routers]
backbone1  ansible_host=10.1.1.1
backbone2  ansible_host=10.1.1.2
backbone3  ansible_host=10.1.1.3

[gateway-routers]
gateway1   ansible_host=10.1.2.1
gateway2   ansible_host=10.1.2.2

[switches]
switch1    ansible_host=10.1.3.1
switch2    ansible_host=10.1.3.2
switch3    ansible_host=10.1.3.3
10.1.4.1
10.1.5.1
```

Host variable

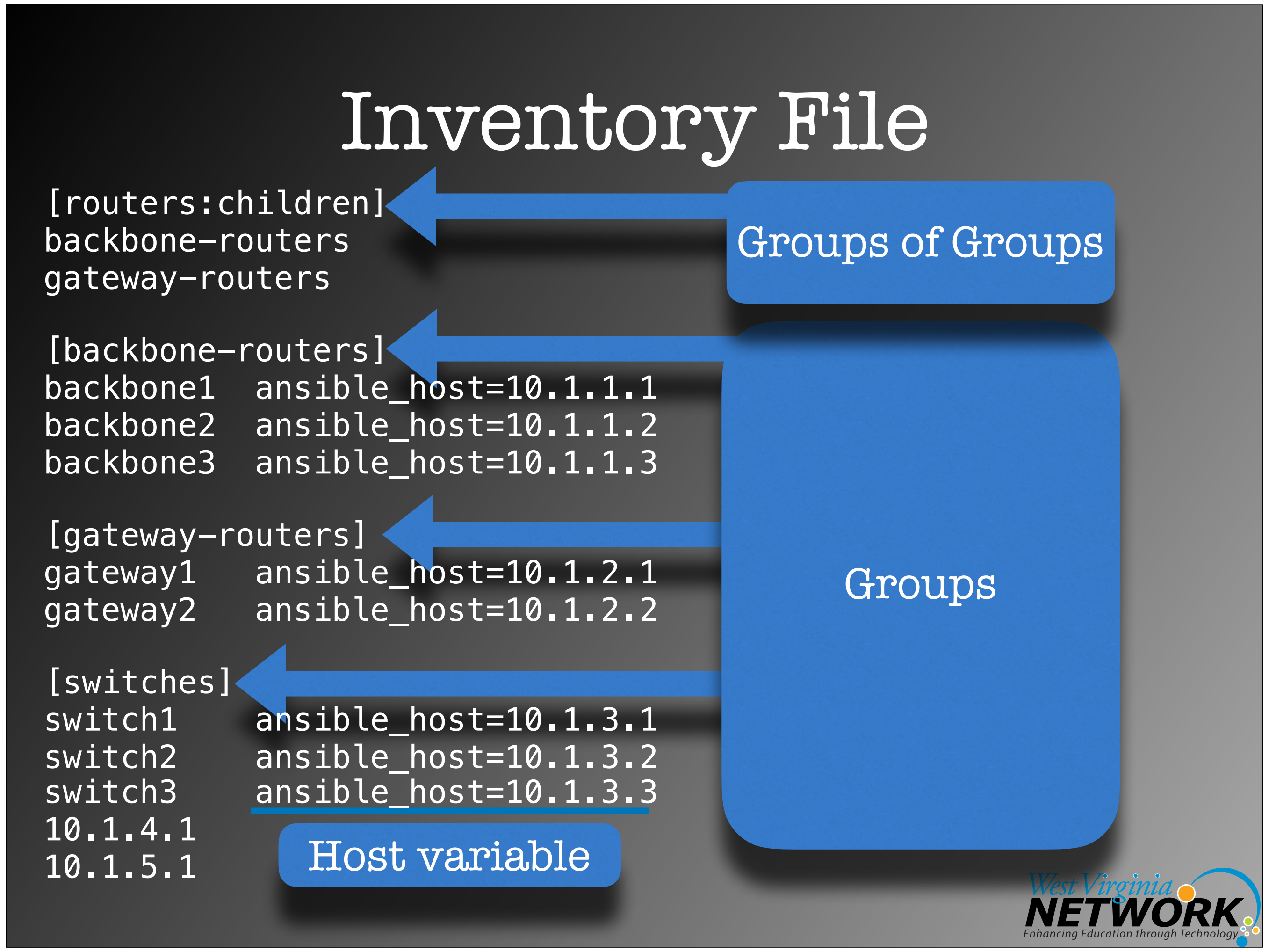
West Virginia NETWORK
Enhancing Education through Technology

Inventory File

```
[routers:children]
backbone-routers
gateway-routers

[backbone-routers]
backbone1  ansible_host=10.1.1.1
backbone2  ansible_host=10.1.1.2
backbone3  ansible_host=10.1.1.3

[gateway-routers]
gateway1   ansible_host=10.1.2.1
gateway2   ansible_host=10.1.2.2

[switches]
switch1    ansible_host=10.1.3.1
switch2    ansible_host=10.1.3.2
switch3    ansible_host=10.1.3.3
10.1.4.1
10.1.5.1
```

Groups

Host variable



Inventory File

```
[routers:children]
backbone-routers
gateway-routers

[backbone-routers]
backbone1  ansible_host=10.1.1.1
backbone2  ansible_host=10.1.1.2
backbone3  ansible_host=10.1.1.3

[gateway-routers]
gateway1   ansible_host=10.1.2.1
gateway2   ansible_host=10.1.2.2

[switches]
switch1    ansible_host=10.1.3.1
switch2    ansible_host=10.1.3.2
switch3    ansible_host=10.1.3.3
10.1.4.1
10.1.5.1
```

West Virginia NETWORK
Enhancing Education through Technology

Running Ansible (2)

Running Ansible (2)

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

Running Ansible (2)

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k


$ ansible 10.1.1.1 -i inventory.txt -m raw -a "command" -u <user> -k
```

Running Ansible (2)

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -i inventory.txt -m raw -a "command" -u <user> -k
```

Running Ansible (2)

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -i inventory.txt -m raw -a "command" -u <user> -k
```

It WORKS! But this is a lot of typing.

Running Ansible (2)

```
$ ansible <device_list> -m <module> -a <attributes> -u <username> -k
```

```
$ ansible 10.1.1.1 -i inventory.txt -m raw -a "command" -u <user> -k
```

It WORKS! But this is a lot of typing.

Let's create an ansible.cfg file to hold our default settings.

ansible.cfg

```
##################################################
# Default configuration values
##################################################

[defaults]
inventory = ./inventory.txt
host_key_checking = False  ;Disable checking SSH keys on remote nodes
record_host_keys = False   ;Disable recording newly discovered hosts in hostfile
timeout = 10               ;Specify how long to wait for responses
forks = 30                 ;Number of parallel processes to spawn
ask_pass = True            ;Playbooks should prompt for password by default
# ask_vault_pass = True
# The following is since we're dealing with Cisco IOS mostly
gathering = explicit       ;facts not gathered unless directly requested in play
# log_path = ./ansible.log ;log information about executions
module_name = raw          ;default module name (-m) value for /usr/bin/ansible
remote_user = frank_seesink
# vault_password_file = /path/to/vault_password_file
```

(Windows INI format)

ansible.cfg Locations

- ANSIBLE_CONFIG
  (an environment variable)
- ansible.cfg (in the current directory)
- .ansible.cfg (in the home directory)
- /etc/ansible/ansible.cfg

Running Ansible (3)

Running Ansible (3)

```
$ ansible <device_list> -i <inventory> -m <module> -a <attributes> -u
<username> -k
```

Running Ansible (3)

```
$ ansible <device_list> -i <inventory> -m <module> -a <attributes> -u
<username> -k
```

```
$ ansible 10.1.1.1 -a “command”
```

Running Ansible (3)

```
$ ansible <device_list> -i <inventory> -m <module> -a <attributes> -u
<username> -k
```

```
$ ansible 10.1.1.1 -a “command”
```

e.g.,

```
$ ansible 10.1.1.1 -a “show version”
```

Running Ansible (3)

```
$ ansible <device_list> -i <inventory> -m <module> -a <attributes> -u
<username> -k
```

```
$ ansible 10.1.1.1 -a “command”
```

e.g.,

```
$ ansible 10.1.1.1 -a “show version”
$ ansible routers  -a “show version”
```

Running Ansible (3)

```
$ ansible <device_list> -i <inventory> -m <module> -a <attributes> -u
<username> -k
```

```
$ ansible 10.1.1.1 -a “command”
```

e.g.,

```
$ ansible 10.1.1.1 -a “show version”
$ ansible routers  -a “show version”
$ ansible routers  -a “show version”          | grep “SUCCESS\|Version”
```

Running Ansible (3)

$ **ansible** *<device_list>* **-i** *<inventory>* **-m** *<module>* **-a** *<attributes>* **-u** <username> **-k**

```
$ ansible 10.1.1.1 -a “command”
```

e.g.,

```
$ ansible 10.1.1.1 -a “show version”
$ ansible routers  -a “show version”
$ ansible routers  -a “show version”          | grep “SUCCESS\|Version”
$ ansible switches -a “show run”              | grep “SUCCESS\|username”
```

Running Ansible (3)

```
$ ansible <device_list> -i <inventory> -m <module> -a <attributes> -u
<username> -k



$ ansible 10.1.1.1 -a “command”


e.g.,

$ ansible 10.1.1.1 -a “show version”
$ ansible routers  -a “show version”
$ ansible routers  -a “show version”           | grep “SUCCESS\|Version”
$ ansible switches -a “show run”               | grep “SUCCESS\|username”
$ ansible all      -a “show run | include ntp”| grep “SUCCESS\| ntp”
```

Example 1

(single file inventory)

```
~/
 ansible.cfg
 inventory.txt
 setup_router.yml
 vlan.yml
```

Example 2

(Using directories)

```
~/
  ansible.cfg
  group_vars/
    backbone-routers
    gateway-routers
    switches
  host_vars/
    backbone1
    backbone2
    …
    switch3
  inventory.txt
  setup_router.yml
  vlan.yml
```

Example 2

(Using directories)

```
~/
  ansible.cfg
  group_vars/
    backbone-routers
    gateway-routers
    switches
  host_vars/
    backbone1
    backbone2
    …
    switch3
  inventory.txt
  setup_router.yml
  vlan.yml
```

```
---
ansible_host: 10.1.1.1
```

```
---
ansible_host: 10.1.1.2
```

```
---
ansible_host: 10.1.3.3
```

Ansible Playbooks

Ansible Playbooks

- YAML files

Ansible Playbooks

- YAML files
- Starting with Ansible v2.4
  - Imperative (define each step) vs. Declarative (define end state)

Playbook (raw)

```
---
- name: Show version of IOS running on routers
  hosts: routers
  gather_facts: false

  tasks:
    - name: Use raw mode to show version
      raw: "show version"

      register: print_output

    - debug: var=print_output.stdout_lines
```

West Virginia NETWORK
Enhancing Education through Technology

Playbook (ios_command)

```
---
- name: Backup running-config on routers
  hosts: routers
  gather_facts: false
  connection: local

  tasks:
    - name: Backup the current config
      ios_command:
        authorize: yes
        commands: show run

      register: print_output

    - name: save output to a file
      copy: content="{{ print_output.stdout[0] }}" dest="./output/
{{ inventory_hostname }}.txt"
```

Playbook (ios_command)

```
---
- name: Show IOS version and interfaces on switches
  hosts: switches
  gather_facts: false
  connection: local

  tasks:
    - name: Run multiple commands and evaluate the output
      ios_command:
        authorize: yes
        commands:
          - show version
          - show interfaces
      register: print_output

    - debug: var=print_output.stdout
```

West Virginia NETWORK
Enhancing Education through Technology

Playbook (ios_command)

```
---
- name: Execute 'show version' on device(s)
  hosts: "{{ host }}"
  gather_facts: false
  connection: local

  tasks:
    - name: Run show version
      ios_command:
        authorize: yes
        commands:
          - show version

      register: print_output

    - debug: var=print_output.stdout

# ansible-playbook show-version.yml -e “host=newtarget(s)"
# ansible-playbook show-version.yml -e “host=routers”
```

Playbook (ios_config)

```
---
- name: Define a VLAN
  hosts: "{{ host | default('switches') }}"
  gather_facts: false
  connection: local

  tasks:
    - name: Define VLAN
      ios_config:
        timeout: 60
        authorize: yes
        parents: "vlan {{ vlan }}"
        lines: "name {{ vlanname }}"

    - name: List VLANs
      ios_command:
        commands: "show vlan | include {{ vlan }}.*active"
      register: print_output

    - debug: var=print_output.stdout

# ansible-playbook set-vlan.yml -e "vlan=250 vlanname=My-new-VLAN"
```

Playbook (ios_facts)

```
---
- name: Collect facts on an IOS device
  hosts: "{{ host | default('switches') }}"
  gather_facts: false
  connection: local

  tasks:
    - name: Collect facts
      ios_facts:
        # gather_subset: all

    - debug:
        msg:
          - "Router      {{ inventory_hostname }}"
          - "Hostname:   {{ ansible_net_hostname }}"
          - "S/N:        {{ ansible_net_serialnum }}"
          - "OS version: {{ ansible_net_version }}"
      when:
        - ansible_net_model | regex_search('3945')
```

Precedence

In 2.x, we have made the order of precedence more specific (with the last listed variables winning prioritization):

1. role defaults [1]
2. inventory file or script group vars [2]
3. inventory group_vars/all
4. playbook group_vars/all
5. inventory group_vars/*
6. playbook group_vars/*
7. inventory file or script host vars [2]
8. inventory host_vars/*
9. playbook host_vars/*
10. host facts
11. play vars
12. play vars_prompt
13. play vars_files
14. role vars (defined in role/vars/main.yml)
15. block vars (only for tasks in block)
16. task vars (only for the task)
17. role (and include_role) params
18. include params
19. include_vars
20. set_facts / registered vars
21. extra vars (always win precedence)

Source: http://docs.ansible.com/ansible/latest/playbooks_variables.html#variable-precedence-where-should-i-put-a-variable

Learning Materials

- https://www.ansible.com/
  - https://docs.ansible.com/
  - https://www.ansible.com/webinars-training
- https://www.udemy.com/ansible-for-network-engineers-cisco-quick-start-gns3-ansible/

Questions?

http://frank.seesink.com

http://frank.seesink.com/presentations/Ansible-Fall2017

West Virginia NETWORK
Enhancing Education through Technology