Network Automation with Ansible

Frank Seesink

v1.0

West Virginia NETWORK
Enhancing Education through Technology

History of Network Management

- SNMP

“Simple” Network Management Protocol

- Oh, and “screen scraping”

DevOps

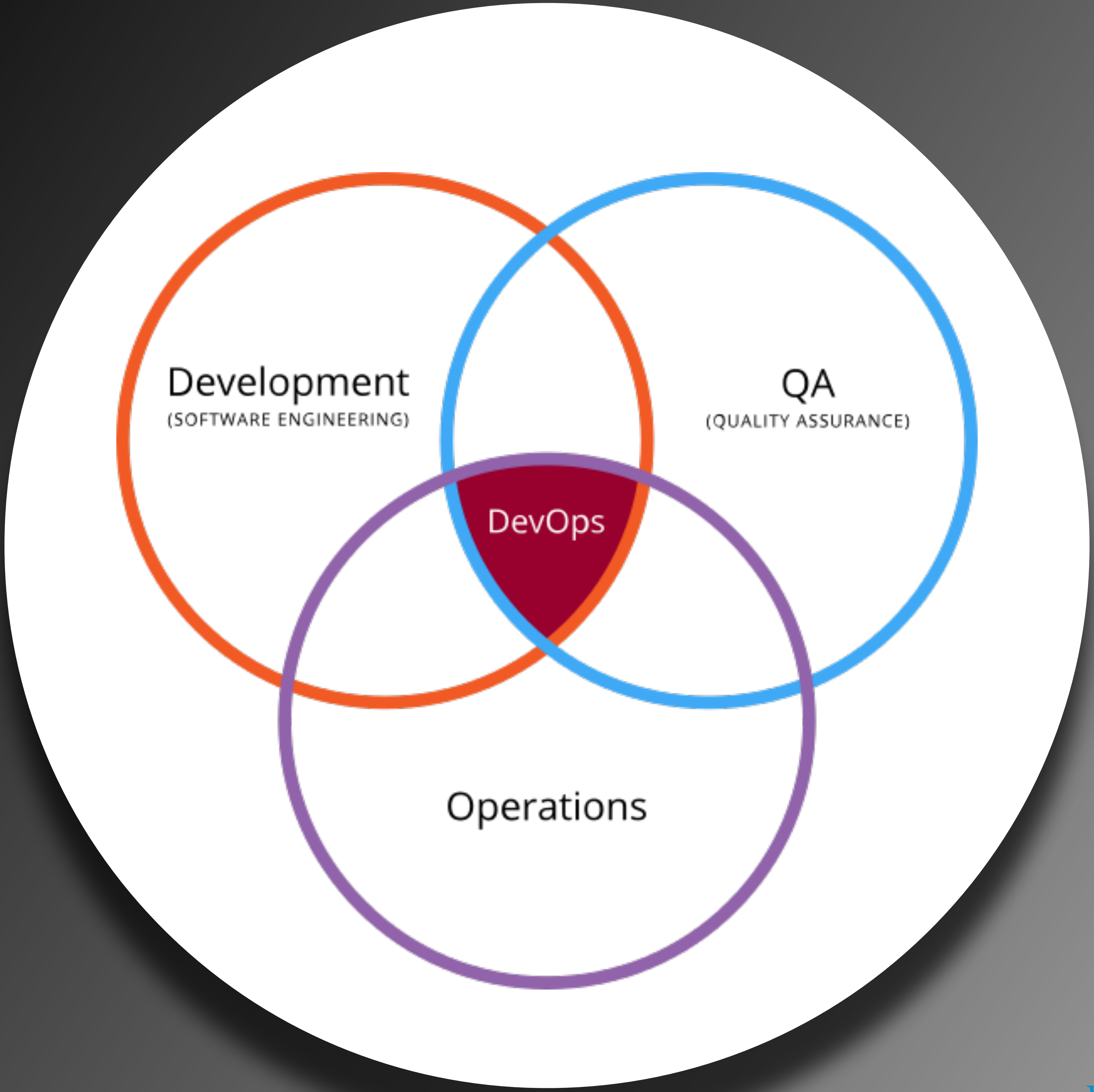
West Virginia NETWORK
Enhancing Education through Technology

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


Development
(SOFTWARE ENGINEERING)
QA
(QUALITY ASSURANCE)
DevOps
Operations

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®

CFEngine → THE C PROGRAMMING LANGUAGE

CHEF™ puppet labs® → Ruby

ANSIBLE munki SALTSTACK → python™

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)

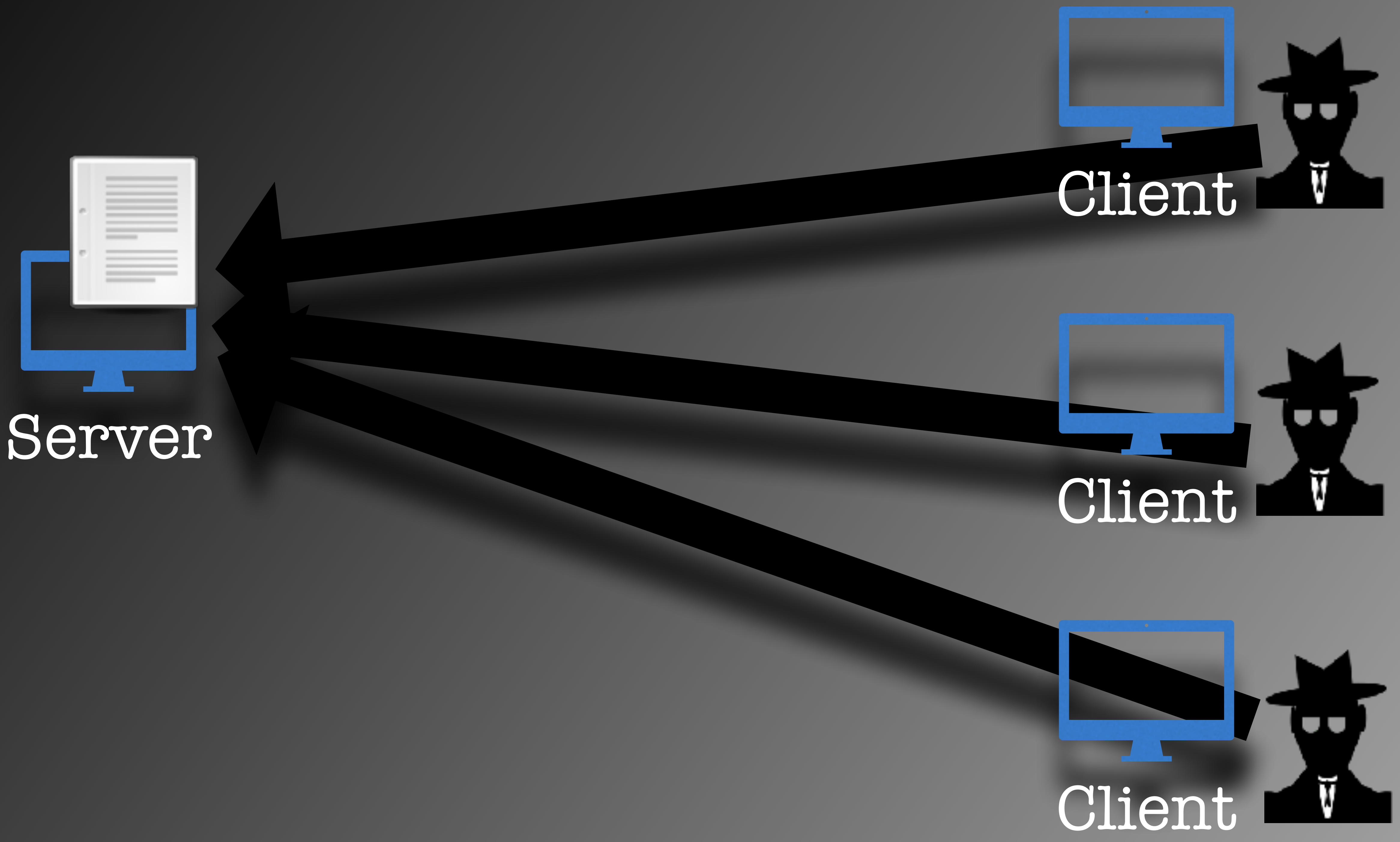
West Virginia NETWORK
Enhancing Education through Technology

Agent-based vs. Agent-less*

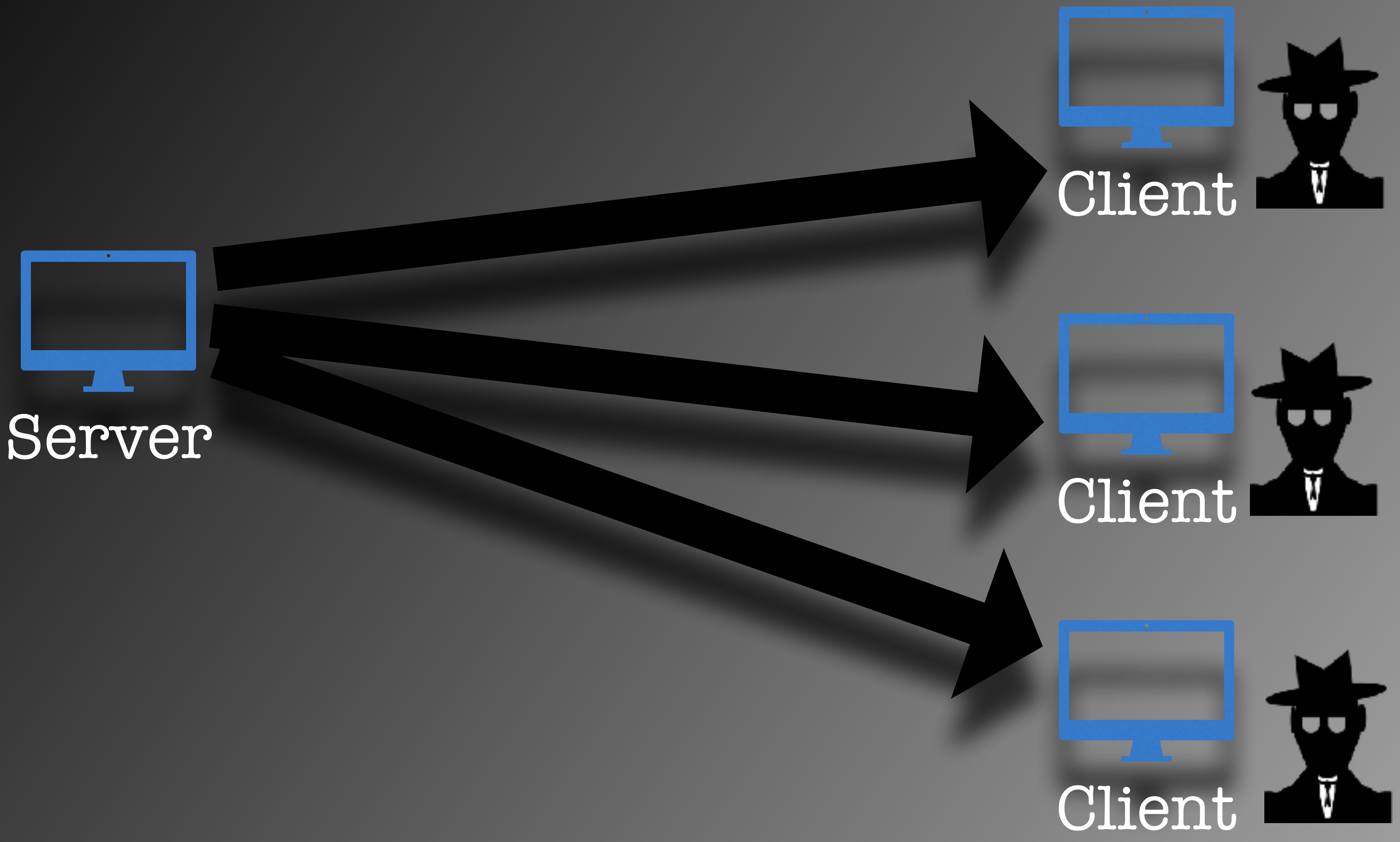
- CFEngine
- Chef
- Munki
- Puppet
- SaltStack

- Ansible

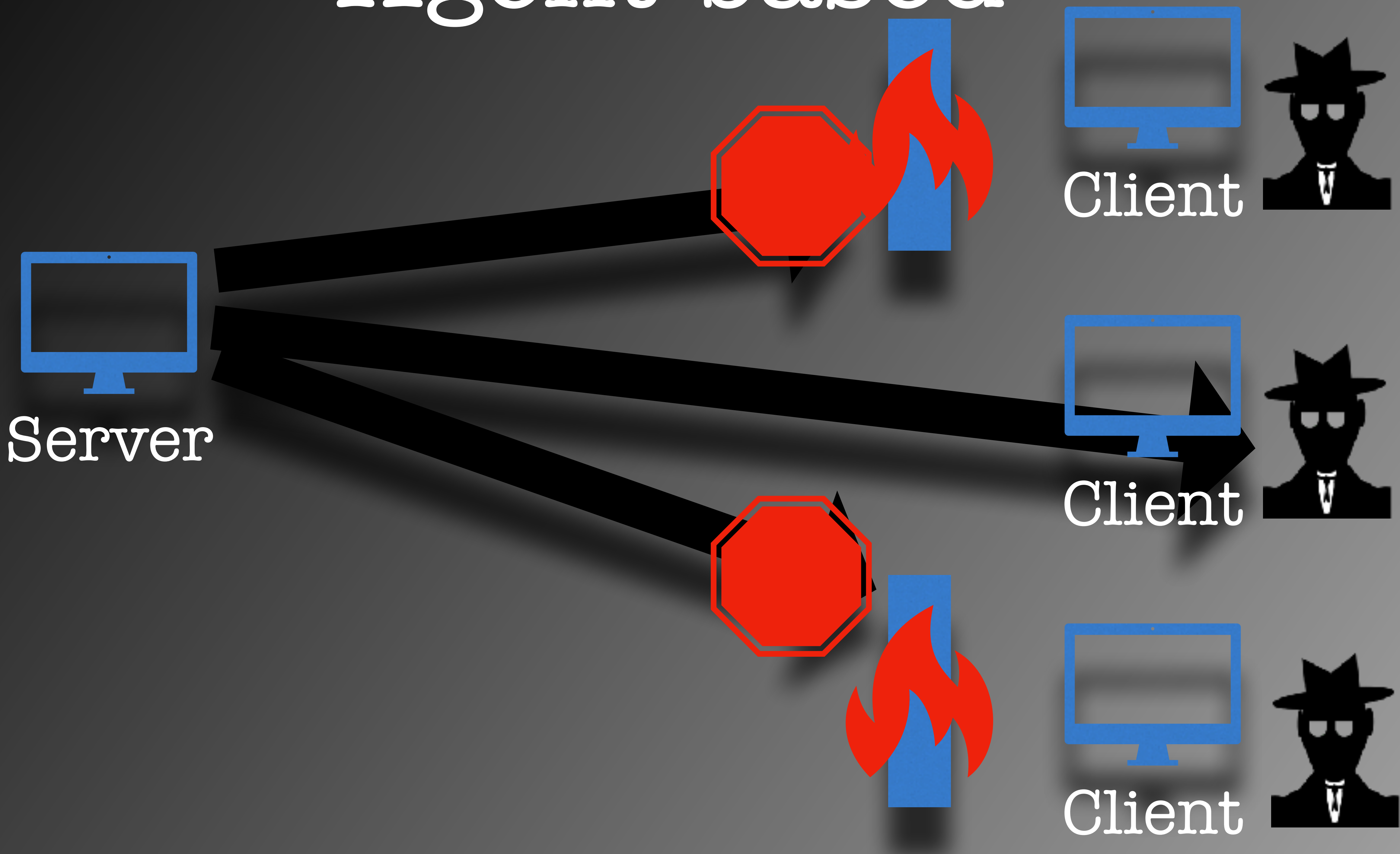
Agent-based



Agent-less

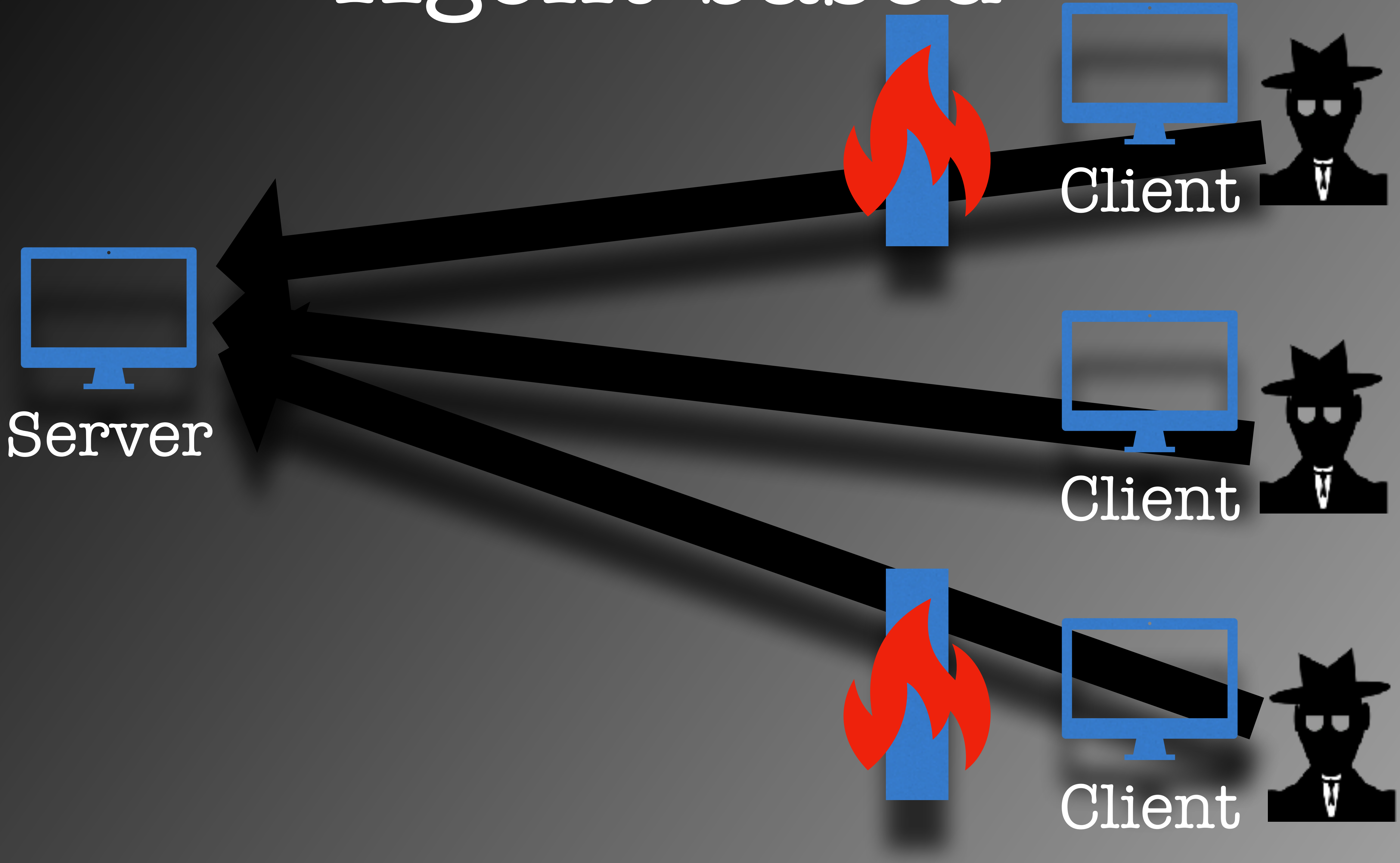


Advantages of Agent-based

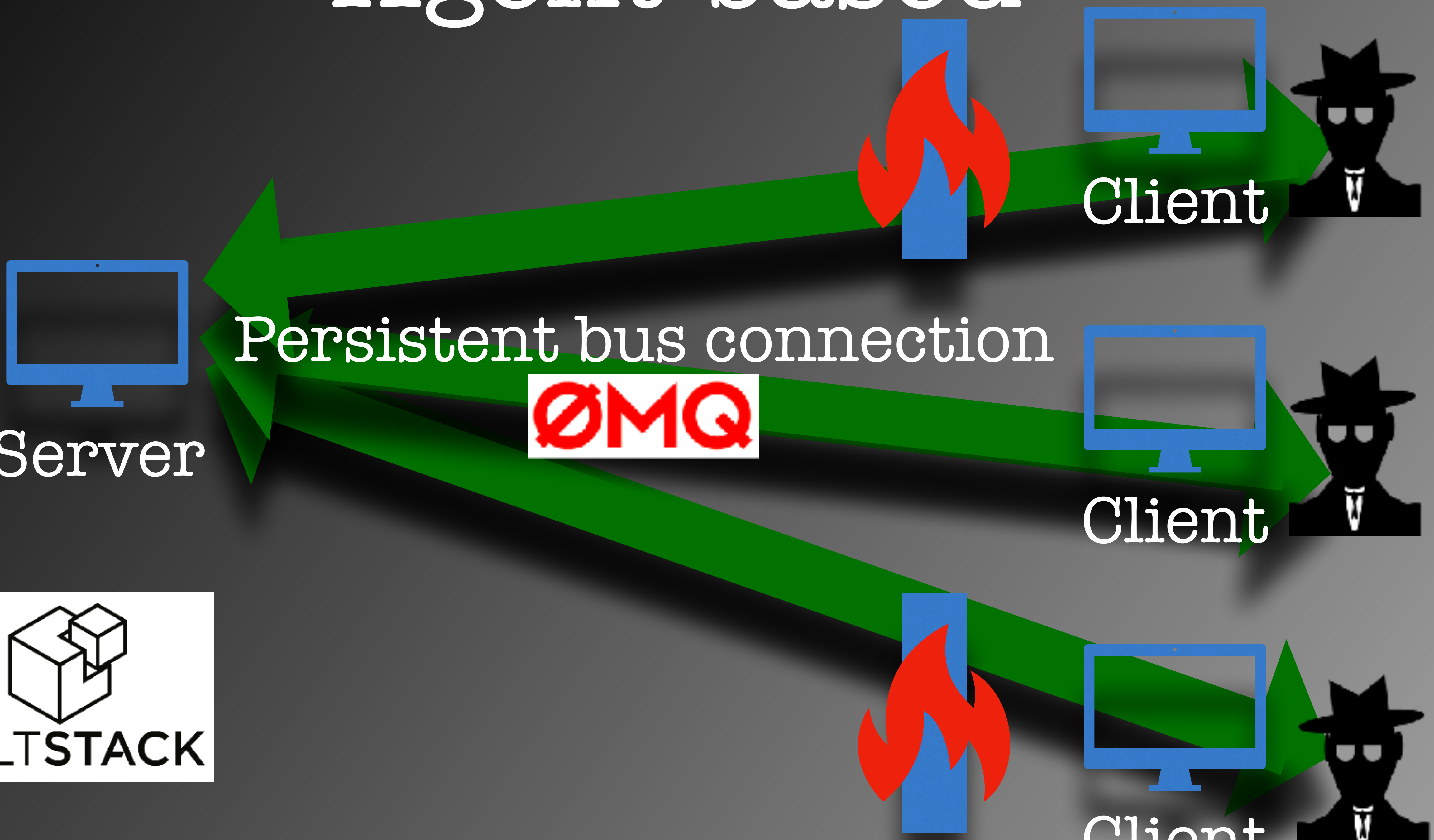


West Virginia NETWORK
Enhancing Education through Technology

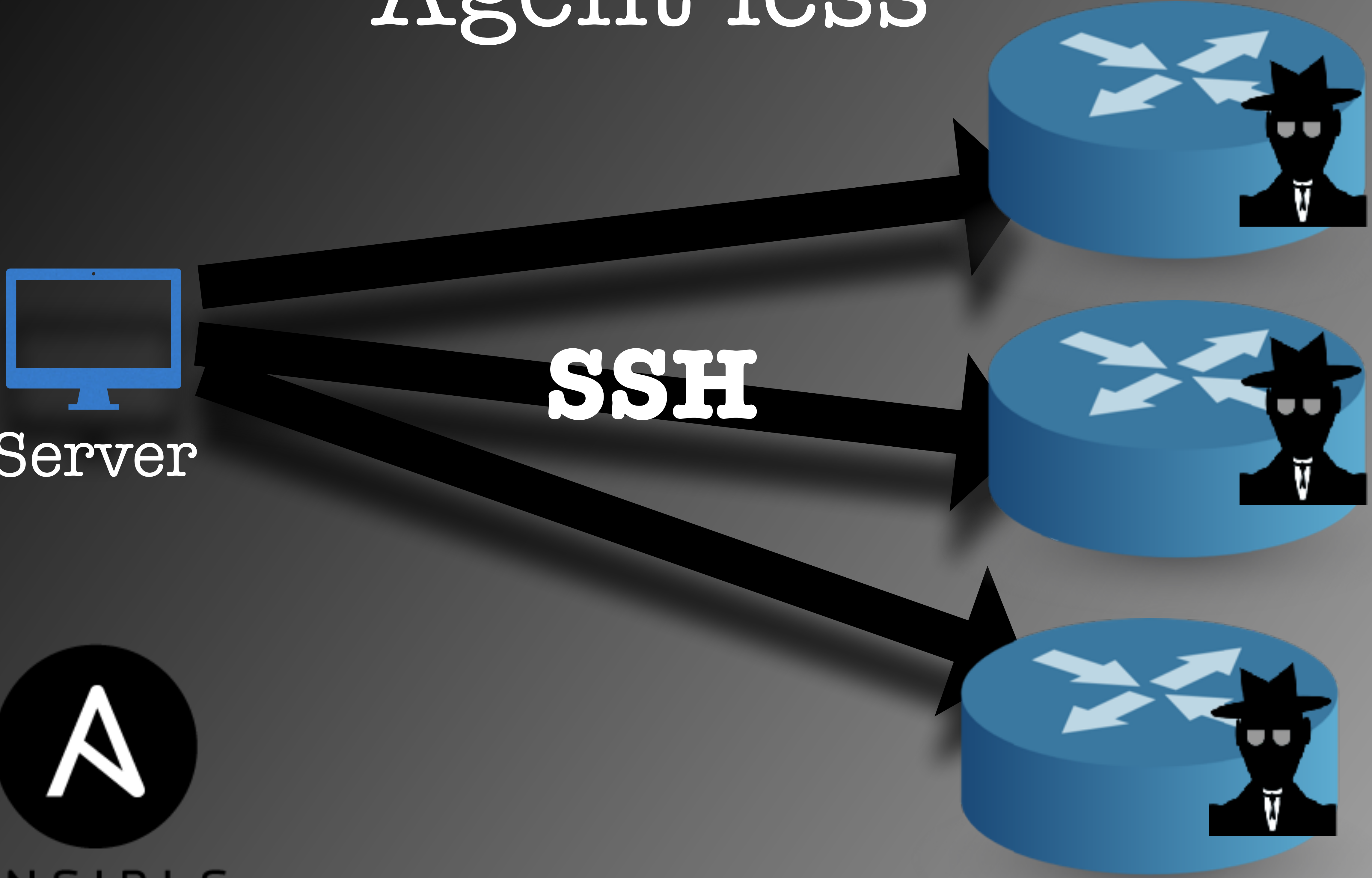
Advantages of Agent-based



Advantages of Agent-based



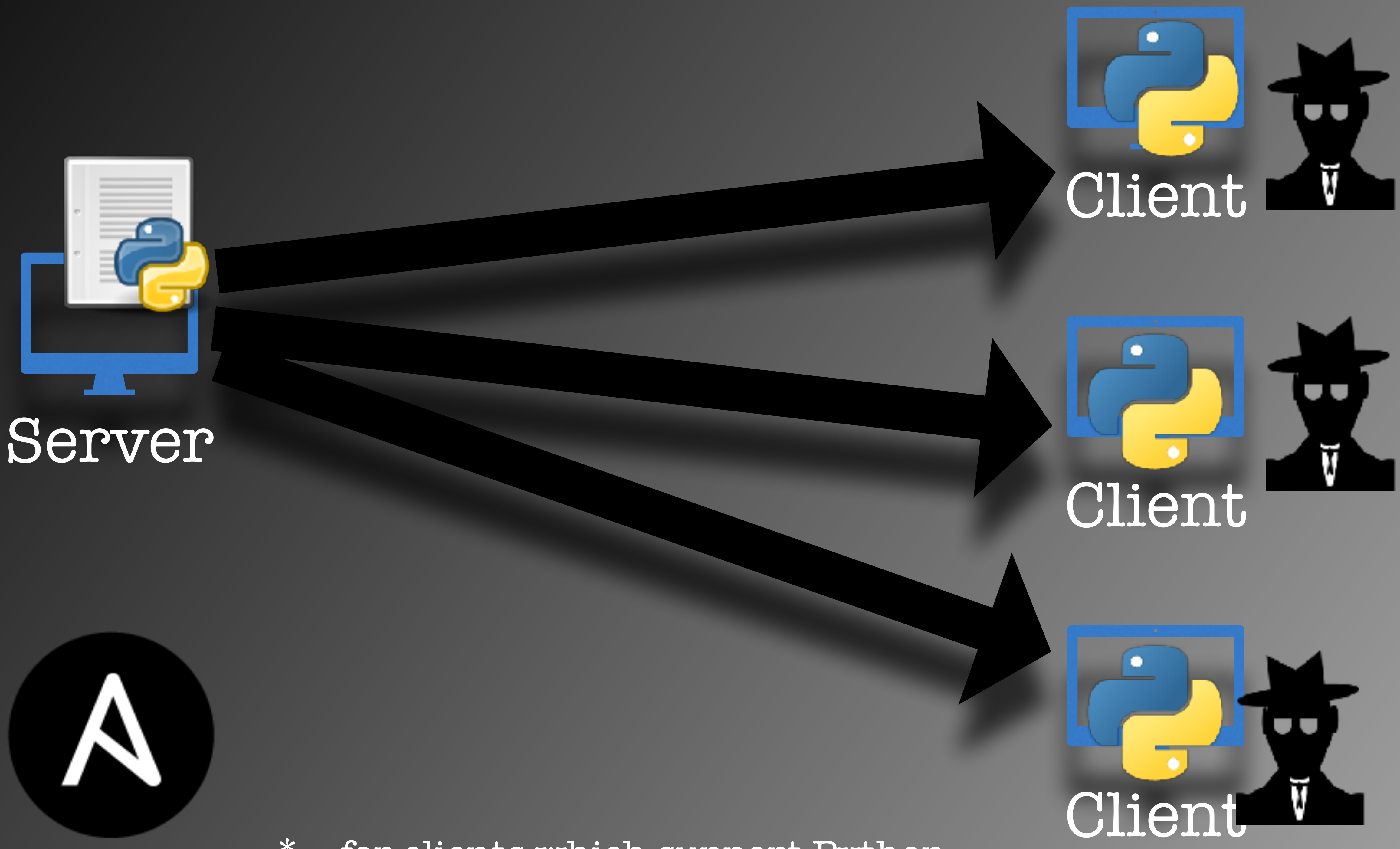
Advantages of Agent-less



ANSIBLE

West Virginia NETWORK
Enhancing Education through Technology

Agent-less*

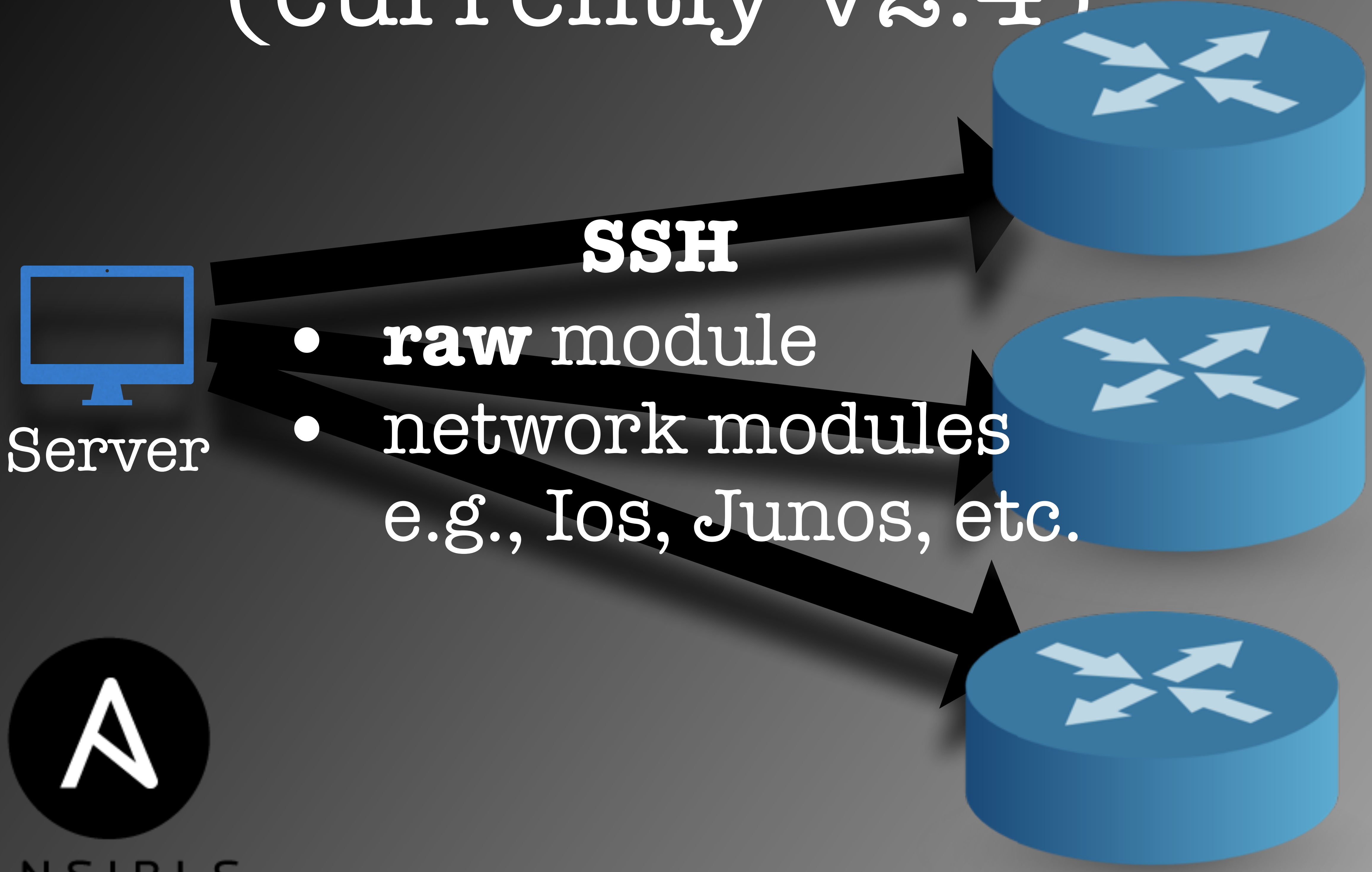


ANSIBLE

* 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)



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 (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

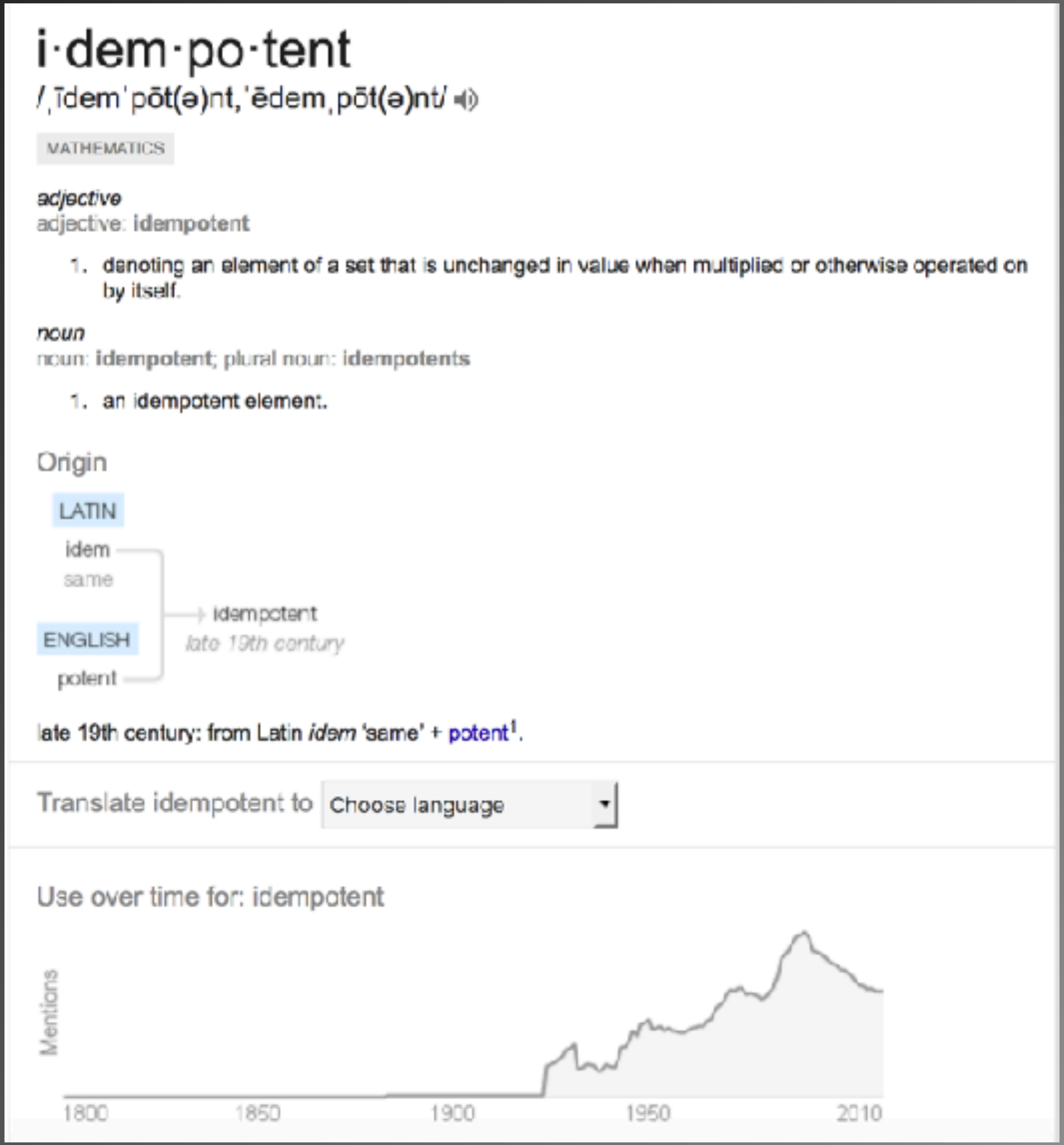
Source: http://docs.ansible.com/ansible/latest/list_of_network_modules.html

West Virginia NETWORK
Enhancing Education through Technology

I am NOT idempotent! Wait… what?

West Virginia NETWORK
Enhancing Education through Technology

Idempotent



i·dem·po·tent

/ˌīdemˈpōt(ə)nt,ˈēdemˌpōt(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

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

Translate idempotent to Choose language

Use over time for: idempotent

Source: "The Google"

redhat®

ANSIBLE

RED HAT® ANSIBLE® Automation

RED HAT ANSIBLE TOWER

Scale + operationalize your automation

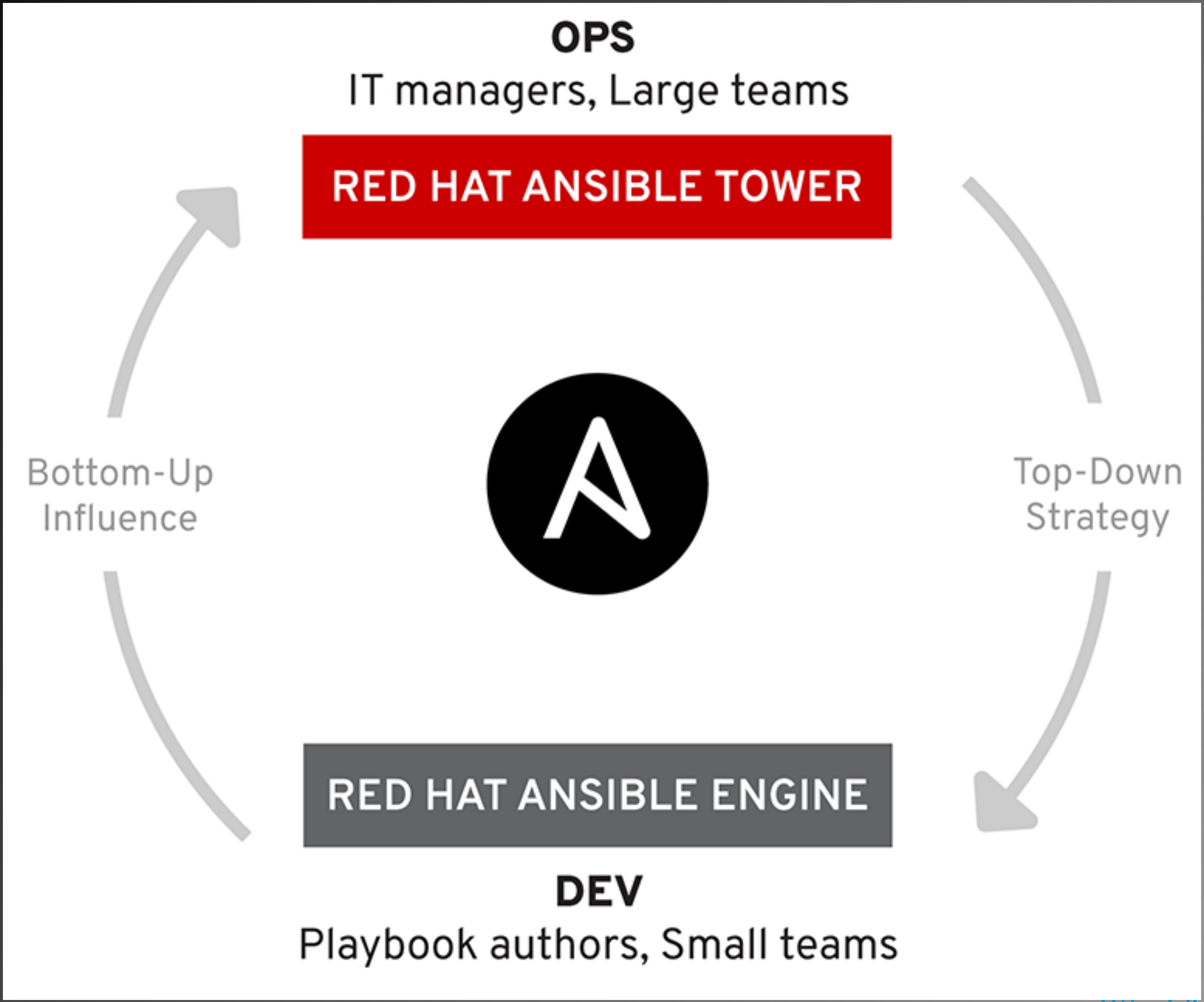
CONTROL | KNOWLEDGE | DELEGATION

RED HAT ANSIBLE ENGINE

Support for your Ansible automation

SIMPLE | POWERFUL | AGENTLESS

FUELED BY AN INNOVATIVE **OPEN SOURCE** COMMUNITY


OPS
IT managers, Large teams
RED HAT ANSIBLE TOWER
Bottom-Up Influence
Top-Down Strategy
RED HAT ANSIBLE ENGINE
DEV
Playbook authors, Small teams

Red Hat Ansible

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

So THAT's why Ansible

Live Demo

West Virginia NETWORK
Enhancing Education through Technology

Deeper Dive

West Virginia NETWORK
Enhancing Education through Technology

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

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
```

West Virginia NETWORK
Enhancing Education through Technology

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
```

Groups of Groups

Groups

Host variable

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

West Virginia NETWORK
Enhancing Education through Technology

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
```

West Virginia NETWORK
Enhancing Education through Technology

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

- 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
```

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"
```

West Virginia NETWORK Enhancing Education through Technology

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”
```

West Virginia NETWORK
Enhancing Education through Technology

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"
```

West Virginia NETWORK
Enhancing Education through Technology

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')
```

West Virginia NETWORK
Enhancing Education through Technology

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/

West Virginia NETWORK Enhancing Education through Technology

Questions?

http://frank.seesink.com

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