ANSIBLE - HOW TO SUCCESSFULLY REPEAT THE BORING STUFF EVERY TIME. MINIONS NOT INCLUDED.

GRETCHEN KUWAHARA, PENN STATE

OVERVIEW

- My Journey
- What is Ansible?
- Setup and Usage
- Demonstrations
- Wrap Up

MY JOURNEY

MY JOURNEY - OREGON STATE

- Systems Administrator at Oregon State
  - 1000 Macs around campus
  - 3 Linux Servers
  - Easy to maintain servers by hand

MY JOURNEY - PENN STATE

- Systems Administrator at Penn State
  - Support Linux servers for Mac services
  - 30+ Linux servers
  - Not so easy to maintain by hand

MY JOURNEY - FIRST SERVER

- Task: Create a new reposado server
- Opportunity to learn about Penn State's service
  - Kerbose, LDAP, Backups, etc
- Used team documentation… updated some too
  - Created a check list for next time

MY JOURNEY - NEXT TIME CAME

- Task: Create a webserver
- Needed the same basic services
- Realized check-list wasn't accurate
  - Time spent troubleshooting
- Not setup the same way as previous servers

"I need to automate this!"

MY JOURNEY - SHELL SHOCK

- Task: Patch all things TODAY
  - During maintenance hours
- Attempted to do it with too
- Inconsistent results

MY JOURNEY - THE SEARCH

- Possible solutions
  - Puppet, Salt, Chef, others…
- Evaluated on setup, use, and interactions with RedHat, etc
- Conference session demonstration of Ansible
  - JSS server in 8 minutes

WHAT IS ANSBILE?

WHAT IS ANSIBLE?

- An open source community project sponsored by Red Hat.
- Automation for everything from servers to clouds to containers and everything in between.
  - Macs, Linux and even Windows
- Uses: configuration management, package deployment, ad-hoc command execution

WHAT IS ANSIBLE?

- How is it different?
  - No demon required to be installed on the client
  - Uses existing transport mechanisms (SSH or SCP)
  - Start performing actions right away
  - Easy learning curve

SETUP AND USAGE

SETUP - REQUIREMENTS

- Controller
  - Linux based system
  - OpenSSH
  - Python 2.6
- Mac/Linux Remote Clients
  - SSH
  - Python
- Windows Remote Clients
  - PowerShell

SETUP AND USAGE - SETUP

- Package distributions available
- Configuration files
  - ansible.cfg
  - hosts file
- Copy SSH keys to remote clients

SETUP - HOSTS FILE

- INI Format File
- List of hosts managed by Ansible
  - Group Hosts
  - Host Aliases
  - Specify connection ports, usernames

```
[nodes]
node1 ansible_host=192.168.245.180
node2 ansible_host=192.168.245.179

[webservers]
dave ansible_host=192.168.245.150
stuart ansible_host=192.168.245.151
jerry ansible_host=192.168.245.152
```

USAGE - MODULES

- Modules do the actual work and get executed in each task
- Built in Modules
  - File - copy, replace, lineinfile,
  - Package - yum, apt, rpm_key, rhn_channel
  - System - user, group, ping, puppet, osx_defaults, services
  - Command - script, shell

USAGE - ANSIBLE COMMAND

- Great for one-liners
  - gather information, install packages, or copy files
- Run as your username, by default
- Can run with privileges
  - –ask-become-pass

```
$ ansible host -a "/usr/bin/foo" -u username –ask-become-pass
```

SETUP AND USAGE - ANSIBLE COMMAND

```
$ ansible alias -m module_name -a “arguments for module”
```

USAGE - FILE MANAGEMENT

- Create or delete a file or directory
- Set permissions and ownership

```
$ ansible nodes -m copy -a “src=/etc/file.cfg dest=/etc/file.cfg
  mode=644 owner=root group=wheel”
```

```
$ ansible nodes -m copy -a dest=/path/to/dir mode=755 owner=root
  group=wheel state=directory”
```

USAGE - MANAGING PACKAGES

- Install or update software packages

```
$ ansible nodes -m yum -a “name=package_name state=latest”
```

- State argument indicates desired package state
  - latest, present, or absent
- Can specify a version with the package name too

```
$ ansible nodes -m yum -a “name=php-5.4.16-36 state=present”
```

USAGE - MANAGING SERVICES

- Start or restart services

```
$ ansible nodes -m services -a “name=service state=started”
```

- Ensure service is stopped

```
$ ansible nodes -m service -a “name=service state=stopped”
```

USAGE - USERS AND GROUPS

- Automate user and group management
  - Create or delete accounts

```
$ ansible nodes -m user -a "name=username state=present
  groups=admin uid=504 shell='/bin/bash'"
```

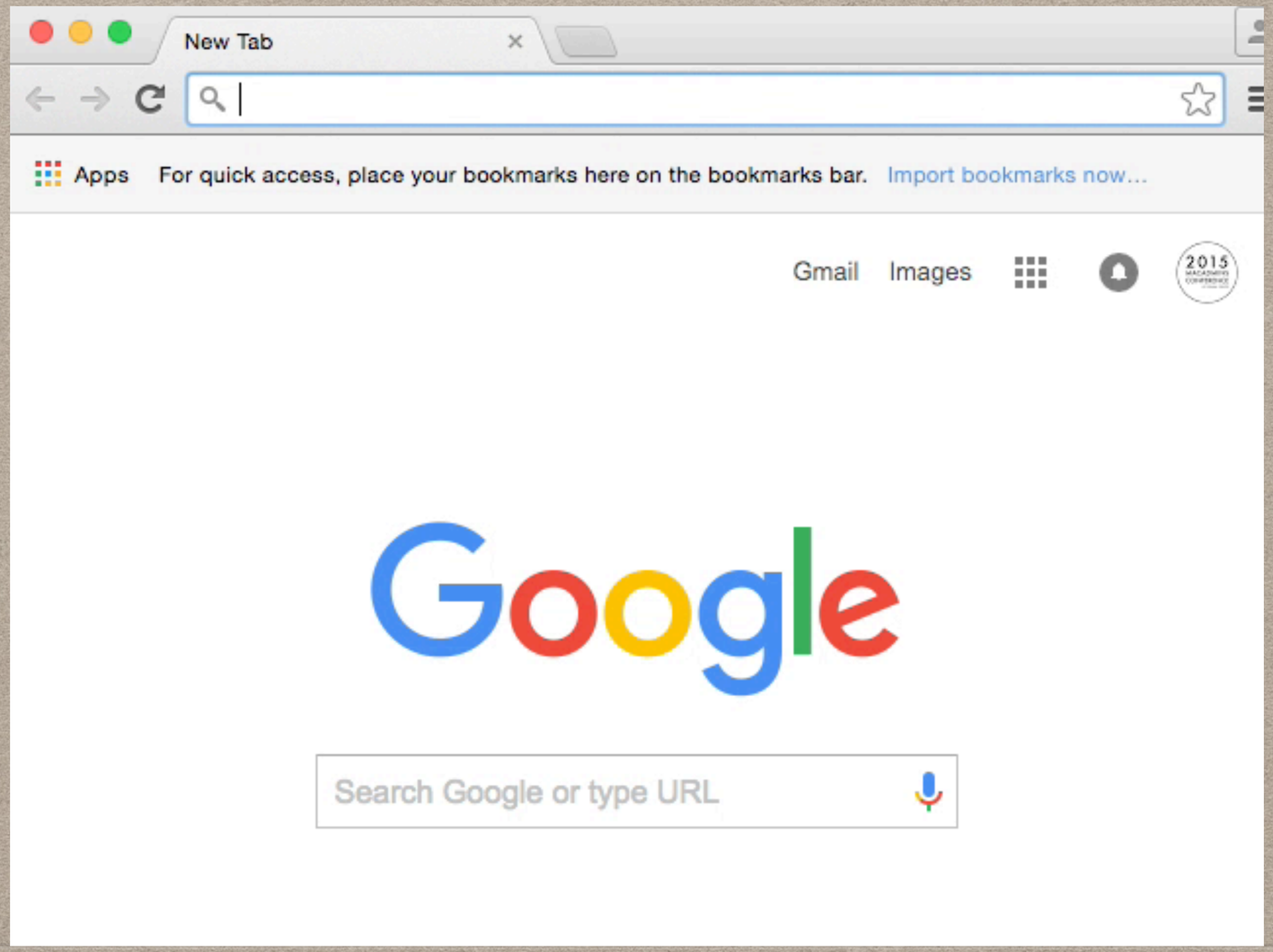
  - Create groups

```
$ ansible nodes -m group -a "name=admin state=present"
```

USAGE - DEMO

```
[root@controler ~]# ansible nodes -m ping
node2 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
node1 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh.",
    "unreachable": true
}
[root@controler ~]#
```

USAGE - CASE: WEBSERVER



USAGE - PLAYBOOKS

USAGE - PLAYBOOKS

- YAML file format
- Multiple commands in a file called “plays”
- Use variables
- Keep them in source control

USAGE - PLAYBOOKS

```
$ ansible nodes -m yum -a "name=httpd state=present"
```

```
$ ansible nodes -m copy -a "src=~/index.html
  dest=/var/www/html/index.html mode=644 owner=apache
  group=apache"
```

```
$ ansible nodes -m service -a "name=httpd
  state=started"
```

USAGE - PLAYBOOKS

```
$ ansible nodes -m yum -a "name=httpd state=present"
```

```
- name: ensure http is installed
  yum: name=httpd state=present
```

USAGE - PLAYBOOKS

```
$ ansible nodes -m copy -a "src=~/index.html
  dest=/var/www/html/index.html mode=644 owner=apache
  group=apache"
```

```
- name: write the index.html file
  copy: src=~/index.html dest=/var/www/html/index.html
    mode=644 owner=apache group=apache
```

USAGE - PLAYBOOKS

```
$ ansible nodes -m service -a "name=httpd
  state=started"
```

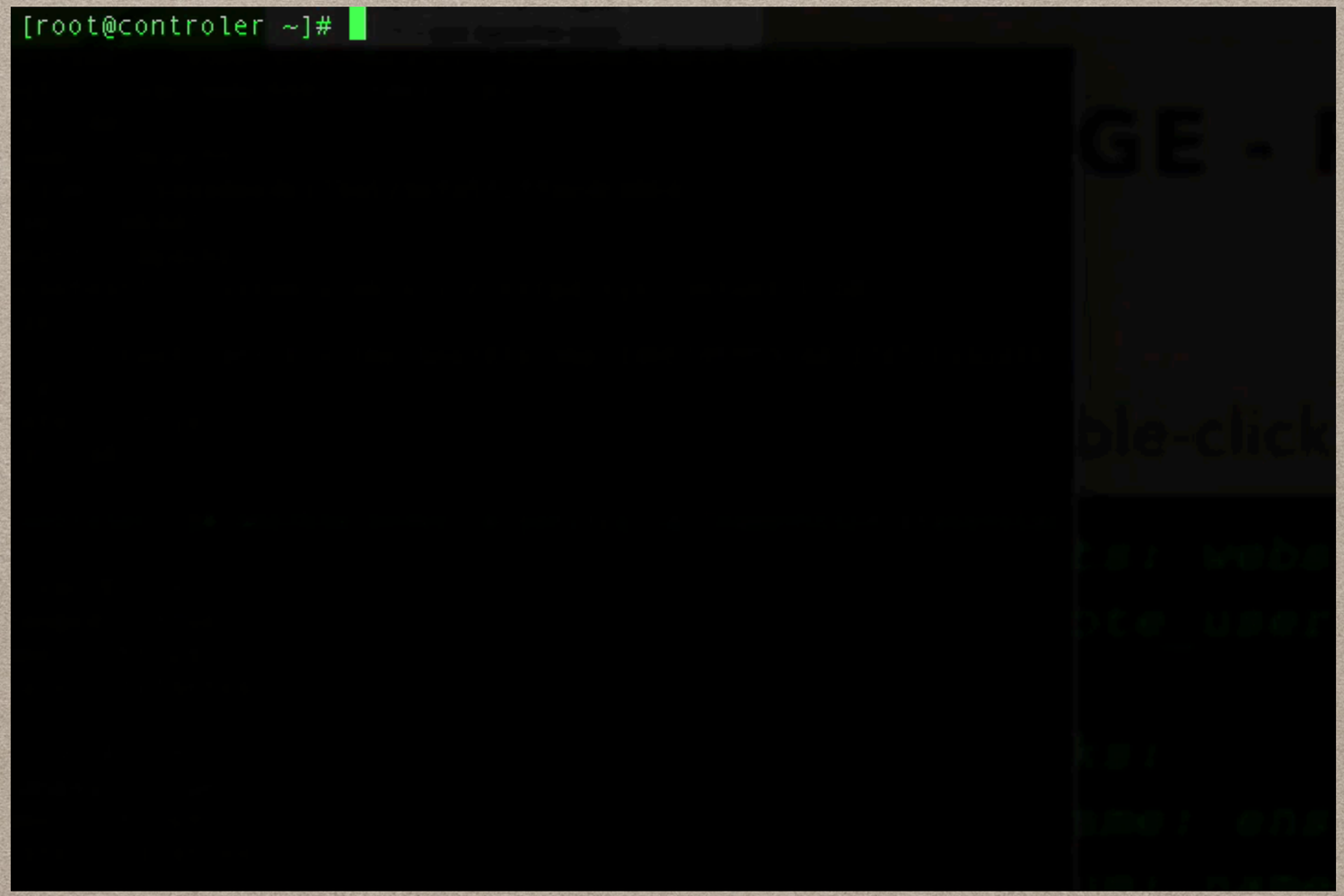
```
- name: ensure apache is running
  service: name=httpd state=started
```

USAGE - PLAYBOOKS

```
- hosts: nodes
  remote_user: root

  tasks:
  - name: ensure http is installed
    yum: name=httpd state=present
  - name: write the index.html file
    copy: src=~/index.html dest=/var/www/html/index.html
      mode=644 owner=apache group=apache
  - name: ensure apache is running
    service: name=httpd state=started
```

USAGE - PLAYBOOKS



USAGE - PLAYBOOKS

Setup an environment variable

Install selinux-python

Install openssl-dev

Copy over duo's repo file

Import Duo's security key

Install duo_unix

Copy over configuration file

Change sshd_config file

Restart SSH service

```
- name: Configure 2FA
  hosts: new
  environment:
    http_proxy: http://proxy.aset.psu.edu:8080
  tasks:
   - name: Install lib selinux-python
     yum: name=libselinux-python
   - name: install openssl-dev
     yum: name=openssl-devel
   - name: copy duorepo file
     copy: src=files/duo.repo dest=/etc/yum.repos.d/duo.repo
   - name: import securitykey
     rpm_key: state=present key=https://www.duosecurity.com/RPM-GPG-KEY-DUO
   - name: install duo
     yum: name=duo_unix
   - name: configuring login_duo
     copy: src=files/login_duo.conf dest=/etc/duo/login_duo.conf mode=0600 owner=sshd
   - name: Disable PermitTunnel and AllowTcpForwarding
     lineinfile:
       dest=/etc/ssh/sshd_config
       regexp=^{{ item.key }}
       line="{{ item.key }} {{ item.value }}"
       insertafter=EOF
       backup=yes
     with_items:
       - { key: 'PermitTunnel', value: 'no' }
       - { key: 'AllowTcpForwarding', value: 'no' }
       - { key: 'ForceCommand', value: '/usr/sbin/login_duo' }

   - name: restart sshd
     service: name=sshd state=restarted
```

USAGE - ADVANCED PLAYBOOKS

- Conditionals - If / Then

```
tasks:
  - name: "shutdown Debian flavored systems"
    command: /sbin/shutdown -t now
    when: ansible_os_family == "Debian"
```

- Roles - Set of playbooks used depending on server's role

WRAP UP

WRAP UP

- Commercial product - Ansible Tower
- Resources - http://docs.ansible.com
- Examples - https://github.com/ansible/ansible-examples
- Tutorials
  - Digital Ocean - https://www.digitalocean.com/community/tags/ansible?type=tutorials
  - Server Hackers - https://serversforhackers.com/an-ansible-tutorial

QUESTIONS?