Ansible – Playbook

Agenda

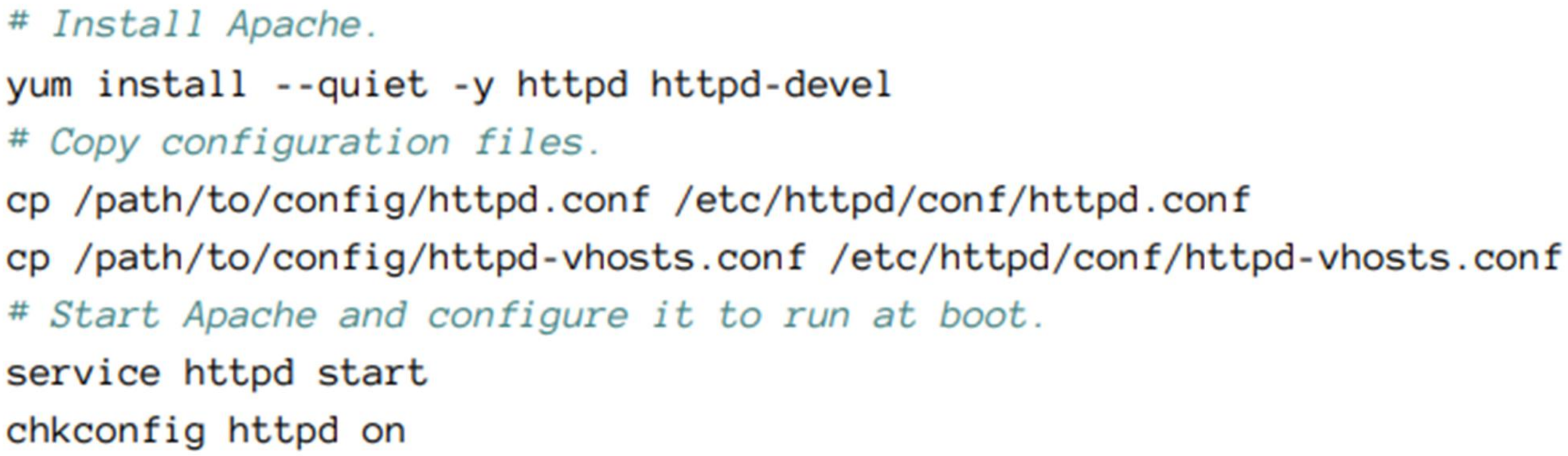
- Running Playbooks
- Real world examples
- Limiting playbooks to certain hosts
- Writing Apache Playbook

Playbooks – Ansible metaphor for configs

- Playbooks are set of tasks – or plays which will be run against a particular server or set of servers
- Playbook - a list of instructions describing the steps to bring your server to a certain configuration state
- What attracts most DevOps personnel to Ansible is the fact that it is easy to convert shell scripts directly into Ansible plays

Playbooks – Ansible metaphor for configs

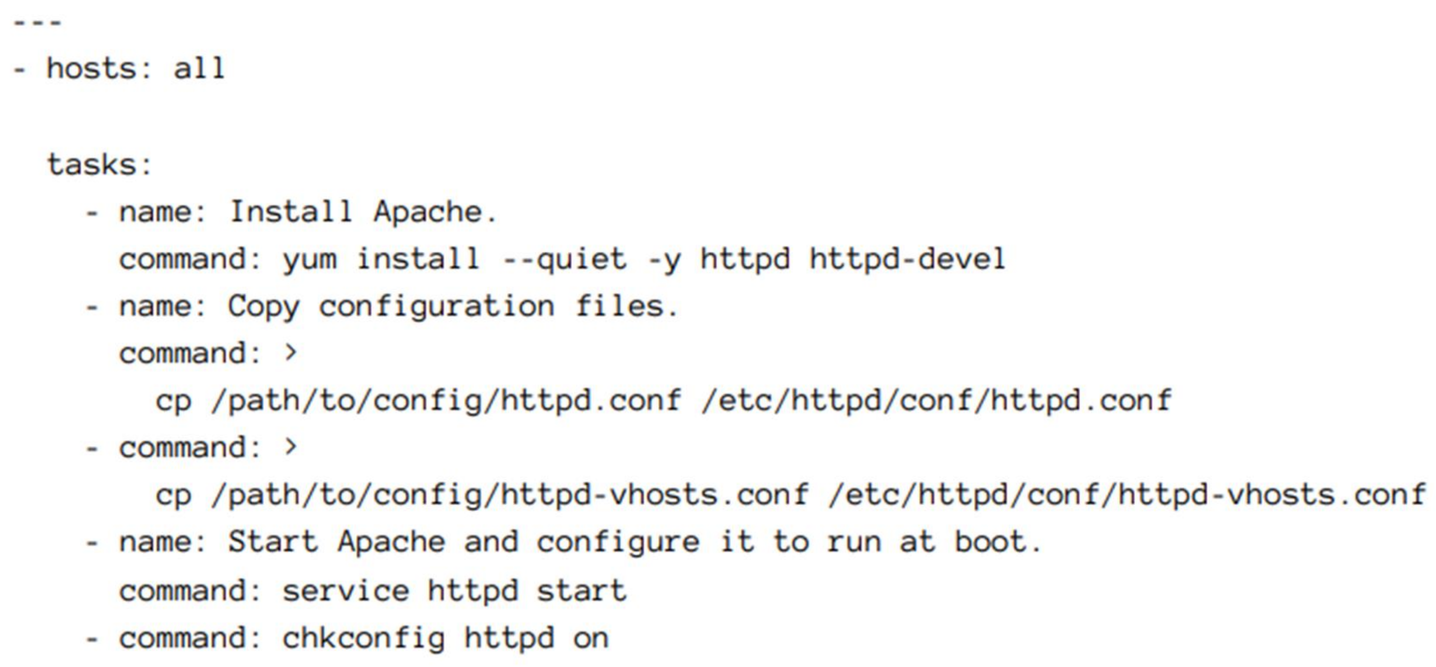
- Below is a shell script for installing Apache



```
# Install Apache.
yum install --quiet -y httpd httpd-devel
# Copy configuration files.
cp /path/to/config/httpd.conf /etc/httpd/conf/httpd.conf
cp /path/to/config/httpd-vhosts.conf /etc/httpd/conf/httpd-vhosts.conf
# Start Apache and configure it to run at boot.
service httpd start
chkconfig httpd on
```

Shell script to Plays - comparison

- You can see that the existing shell scripts can easily be converted to YAML playbook



```yaml
---
- hosts: all

  tasks:
    - name: Install Apache.
      command: yum install --quiet -y httpd httpd-devel
    - name: Copy configuration files.
      command: >
        cp /path/to/config/httpd.conf /etc/httpd/conf/httpd.conf
    - command: >
        cp /path/to/config/httpd-vhosts.conf /etc/httpd/conf/httpd-vhosts.conf
    - name: Start Apache and configure it to run at boot.
      command: service httpd start
    - command: chkconfig httpd on
```

- $ ansible-playbook playbook.yml
- You can quickly transition to writing playbooks if you are good at writing shell scripting

Limiting playbooks to particular hosts and groups

- You can limit a playbook to specific groups or individual hosts by changing the hosts: definition
- The value can be set to all hosts, a group of hosts defined in your inventory, multiple groups of hosts (e.g. webservers,dbservers), individual hosts or mixed (using wild card matches)

- $ ansible-playbook playbook.yml --limit webservers

- In this case (assuming your inventory file contains a webservers group), even if the playbook is set to hosts: all, or includes hosts in addition to what's defined in the webservers group, it will only be run on the hosts defined in webservers.

- $ ansible-playbook playbook.yml --limit xyz.example.com
- $ ansible-playbook playbook.yml --list-hosts

Setting user and sudo options with ansible-playbook

- If no user is defined alongside the hosts in a playbook, Ansible assumes you'll connect as the user defined in your inventory file for a particular host, and then will fall back to your local user account name.
-
  ```
  $ ansible-playbook playbook.yml --remote-user=ndtadmin
  ```
-
  ```
  $ ansible-playbook playbook.yml --sudo --sudo-user=ndtadmin --ask-sudo-pass
  ```

Other options for ansible-playbook

- --inventory=PATH (-i PATH): Define a custom inventory file (default is the default Ansible inventory file, usually located at /etc/ansible/hosts)
- --verbose (-v): Verbose mode (show all output, including output from successful options). You can pass in -vvvv to give every minute detail
- --extra-vars=VARS (-e VARS): Define variables to be used in the playbook, in "key=value,key=value" format
- --forks=NUM (-f NUM): Number for forks (integer). Set this to a number higher than 5 to increase the number of servers on which Ansible will run tasks concurrently
- --connection=TYPE (-c TYPE): The type of connection which will be used (this defaults to ssh; you might sometimes want to use local to run a playbook on your local machine
- --check: Run the playbook in Check Mode (‘Dry Run’); all tasks defined in the playbook will be checked against all hosts, but none will actually be run

Apache – Playbook – apacheplaybook.yml

```
---
- name: This sets up an httpd webserver
  hosts: all
  become: yes
  tasks:
    - name: Install the httpd rpm
      yum: name=httpd state=present
    - name: start the httpd service
      service: name=httpd state=started
    - name: Open port 80
      firewalld: service=http permanent=true state=enabled
    - name: start the firewalld service
      service: name=firewalld state=restarted
```

Stopping the Apache service

- Write the ad hoc command to stop apache service