apfelwerk

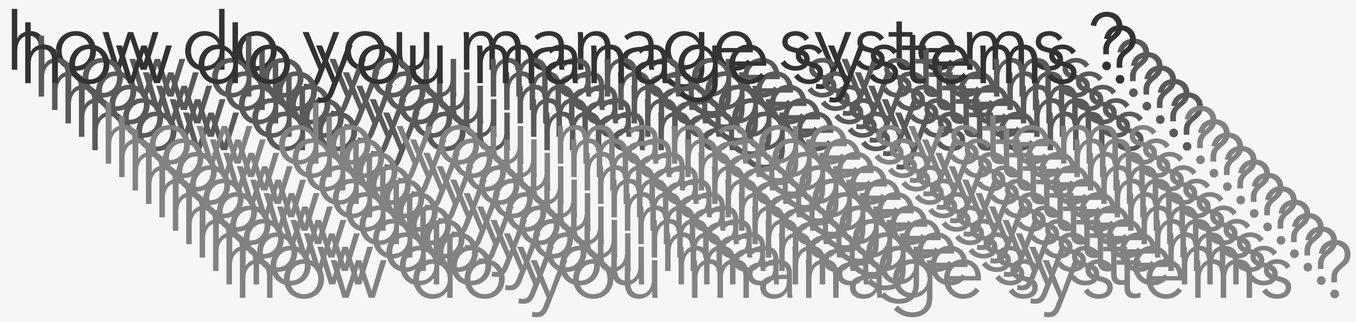
Henry Stamerjohann

Apfelwerk GmbH & Co. KG

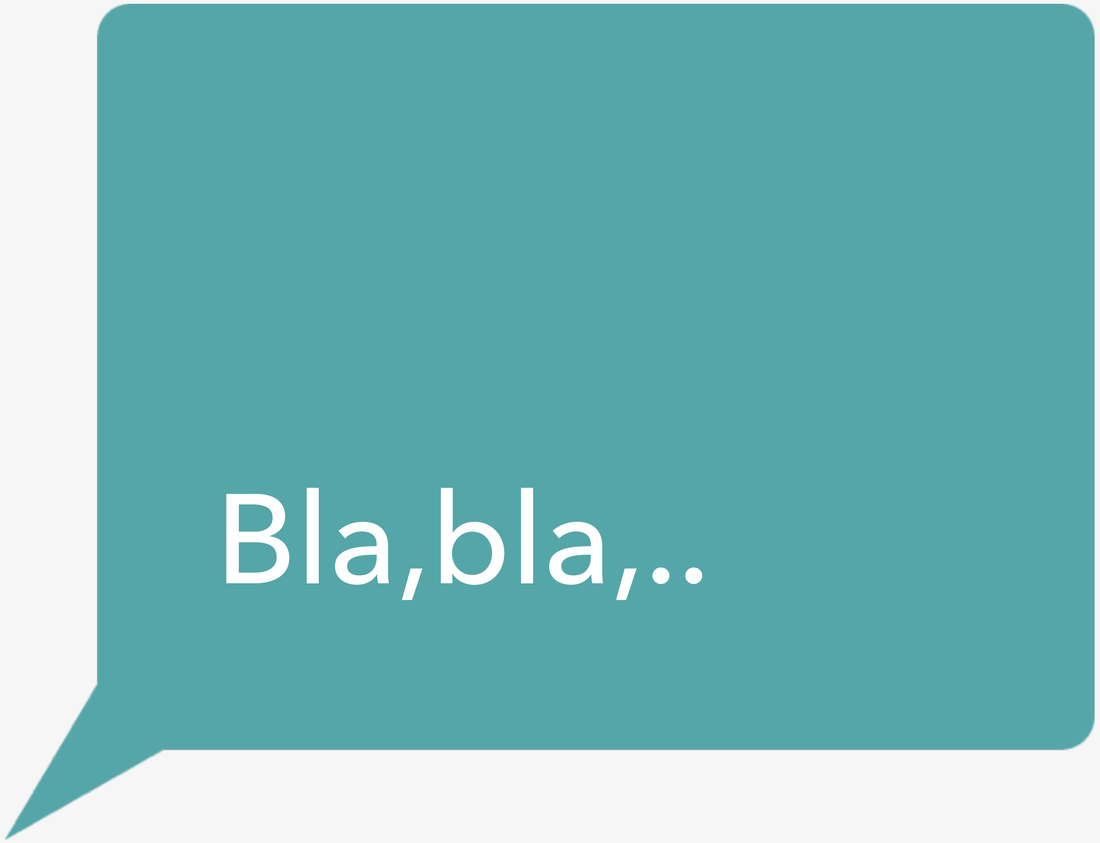
@head_min

#macadmins

Configuration Management


how do you manage systems ??

Why do cfgmgmt ?



Why do cfgmgmt ?

Infrastructure as Code

Documented Progress

Deterministic Process

Idempotency

Ansible

What is Ansible ?

What is Ansible ?

Radically **simple IT automation** system

Works via **SSH communication**

No extra software on servers required

Configure systems **without**

a daemon or central database

Will it Blend ?

Configuration management

Application deployment

Multi-node orchestration

Ad-hoc task-execution

Support cloud provisioning

Foundation

Foundation

Open Source

Based on Python 2.7

Strong community

Highly modular

Extendable

The Goals

The Goals

Simplicity

Maximum ease of use

Security

Reliability

Auditability by humans

More Facts

Use **simple YAML** based syntax

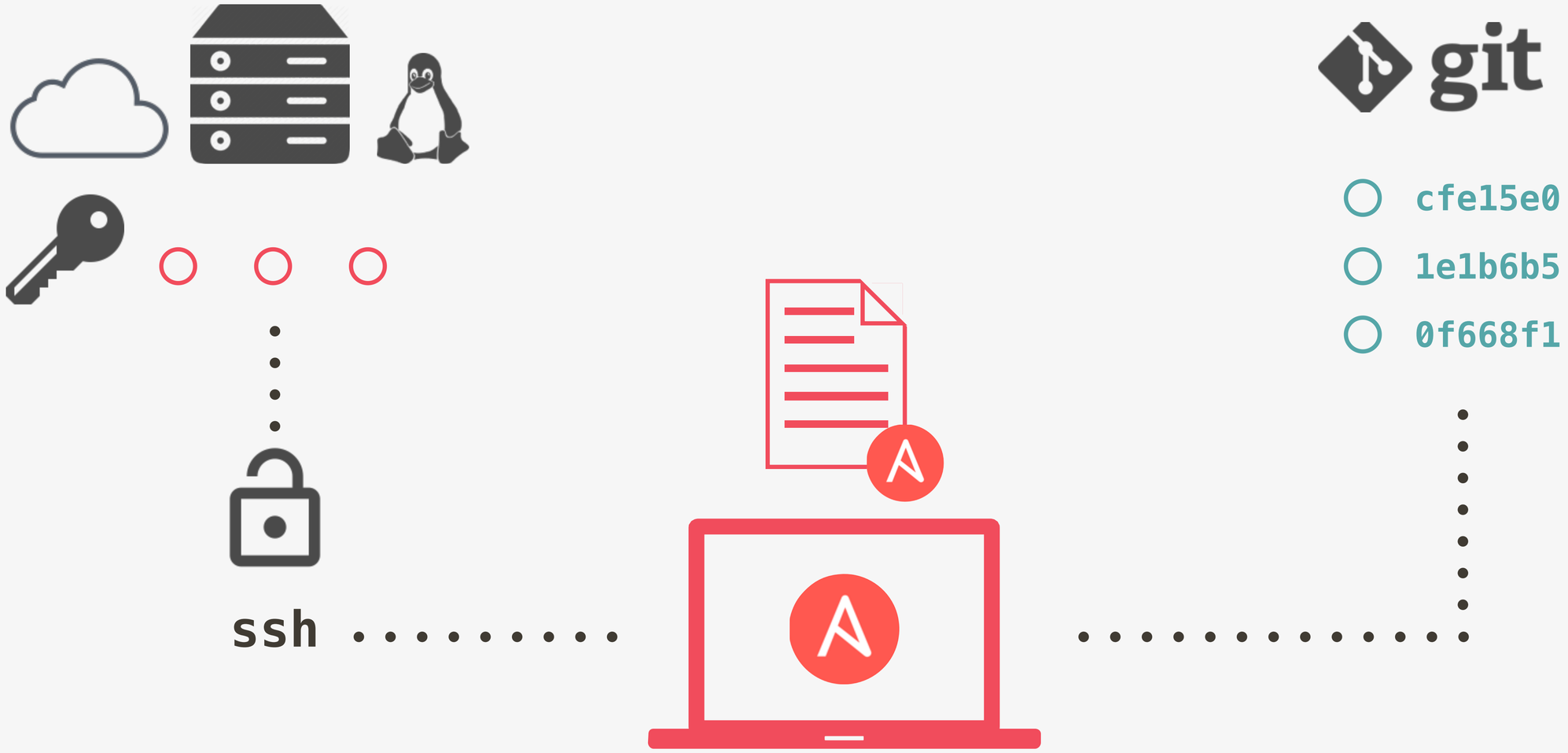
Simple execution order **top to bottom**

Agent-less - no client to install and maintain

supports powerful Jinja2 template

large library of **builtin modules**

Operation



Install Ansible

Requirements

Python on **admin** machine

Python on **nodes** to be managed

ssh-keys on nodes (recommended)

Module dependencies (Python)

Install Ansible


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

module requirements:

```
pip install <py-module_name>
```

Ansible binary

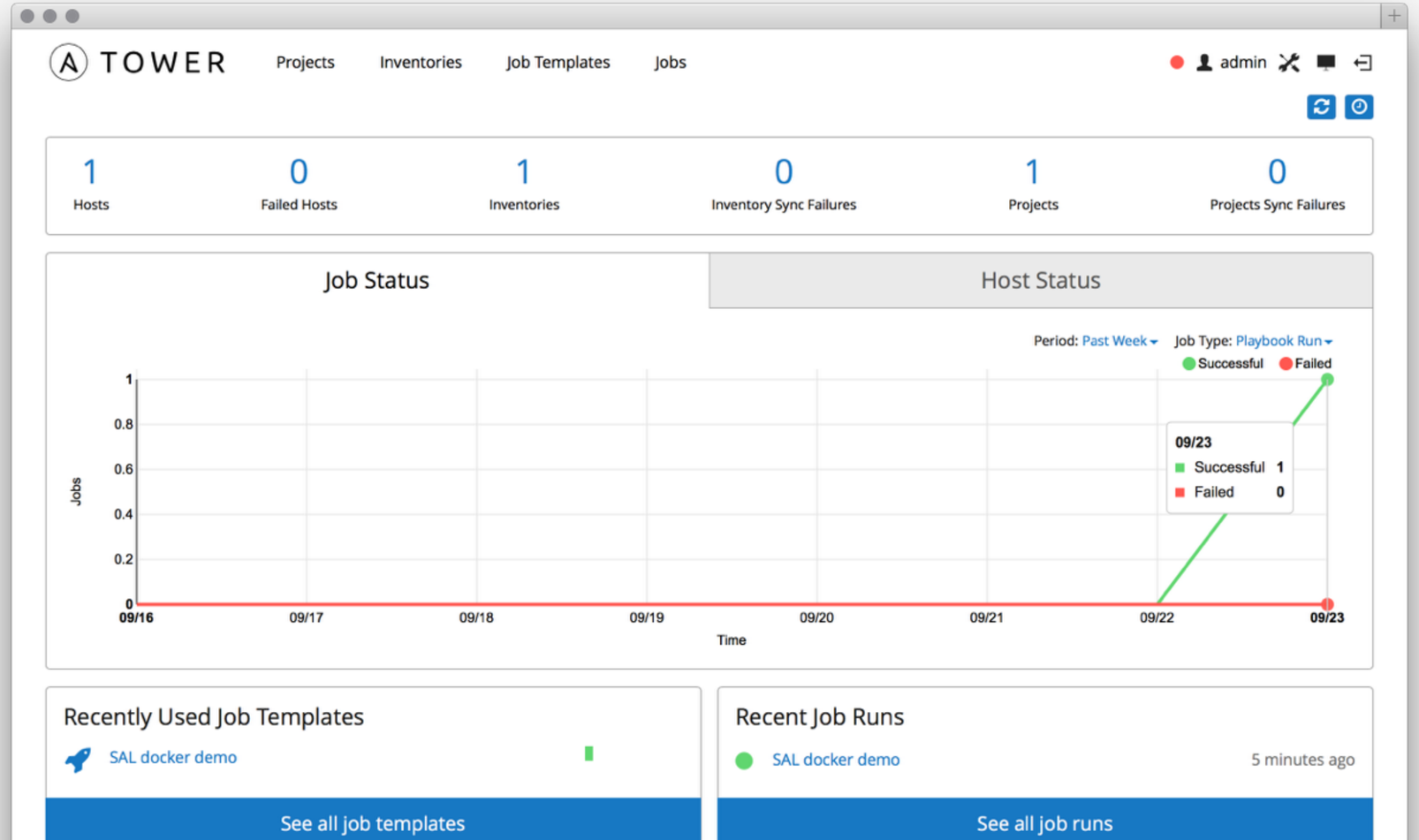
- ansible
- ansible-playbook
- ansible-doc
- ansible-galaxy
- ansible-pull
- ansible-vault

Ansible run options

- debug with **-vvvv**
- dry run mode with **--check**
- check playbooks with **--syntax-check**

Ansible Tower 2.3

http://www.ansible.com/tower



commercial version

Get closer with Ansible

Terminology

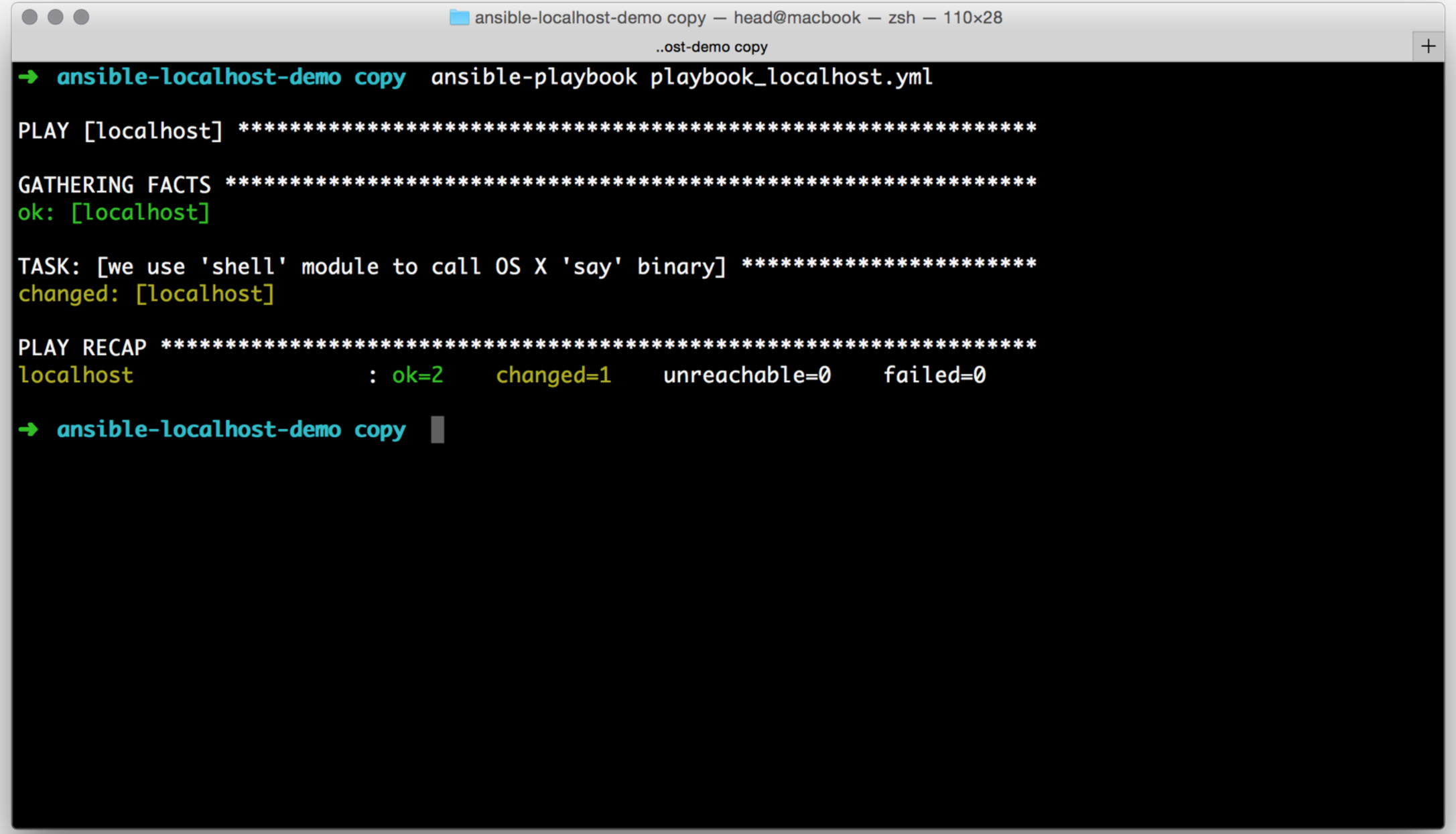
- **Inventory** — Hosts file to work with
- **Playbook** — Masterfile to execute
- **Task** — Call a module, execute command
- **Module** — Call for specific functionality

Playbook dissect

`---`	YAML, start of a playbook
`hosts:`	machine(s) you want to perform operations
`vars:`	variables for that play / include separate vars file
`tasks:`	tasks you want to perform on the host
`- name:`	display description for executed task/command
`shell:`	run a module that allows to perform shell commands
`- include:`	"run" external YAML file containing tasks, variables

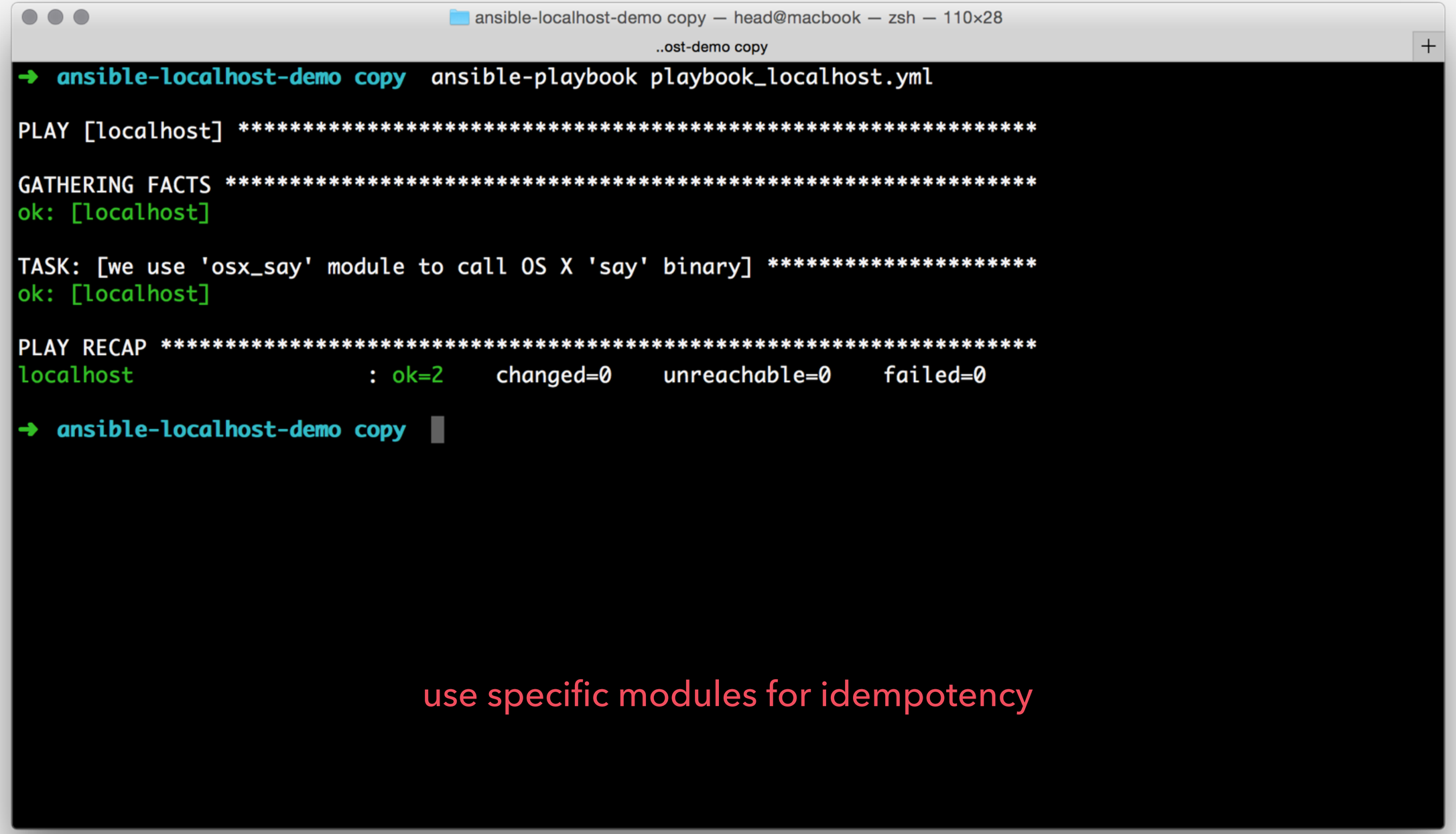
Playbook / shell module

```
1  ---
2  - hosts: localhost
3    vars:
4    - message: "Hello MacSysAdmin 2015!"
5
6    # we use the 'shell' module to call '/usr/bin/say'
7    tasks:
8    - name: we use 'shell' module to call OS X 'say' binary
9      shell: say "{{ message }}" --voice=Zarvox
```


ansible-localhost-demo copy — head@macbook — zsh — 110×28
..ost-demo copy
➜ ansible-localhost-demo copy ansible-playbook playbook_localhost.yml
#0
PLAY [localhost] ***
GATHERING FACTS **
ok: [localhost]
#1
TASK: [we use 'shell' module to call OS X 'say' binary] ************************
changed: [localhost]
#2
PLAY RECAP ***
localhost : ok=2 changed=1 unreachable=0 failed=0
➜ ansible-localhost-demo copy

Playbook / osx_say module

```
1  ---
2  - hosts: localhost
3    vars:
4    - message: "Hello MacSysAdmin 2015!"
5
6    # we use the 'osx_say' module to call '/usr/bin/say'
7    tasks:
8    - name: we use 'osx_say' module to call OS X 'say' binary
9      osx_say: msg="{{ message }}" voice=Zarvox
```


ansible-localhost-demo copy — head@macbook — zsh — 110×28
..ost-demo copy
➜ ansible-localhost-demo copy ansible-playbook playbook_localhost.yml
#0
PLAY [localhost] ***
GATHERING FACTS **
ok: [localhost]
#1
TASK: [we use 'osx_say' module to call OS X 'say' binary] **********************
ok: [localhost]
#2
PLAY RECAP ***
localhost : ok=2 changed=0 unreachable=0 failed=0
➜ ansible-localhost-demo copy
use specific modules for idempotency

Playbook dissect

--- ○ YAML, start of a playbook

hosts: ○ machine(s) you want to perform operations

vars: ○ variables for that play / include vars file

tasks: ○ tasks you want to perform on the host

name: ○ display description for executed task/command

osx_say: ○ a specific module that allows you to call 'say' on OS X

include: ○ "run" external YAML file containing tasks, variables

{{ parameter }} ○ in braces we reference parameters from vars/inventory

Inventory file: ○ referred in ansible.cfg, similar to a /etc/hosts file

Inventory

Inventory

Hosts file are simple Text files

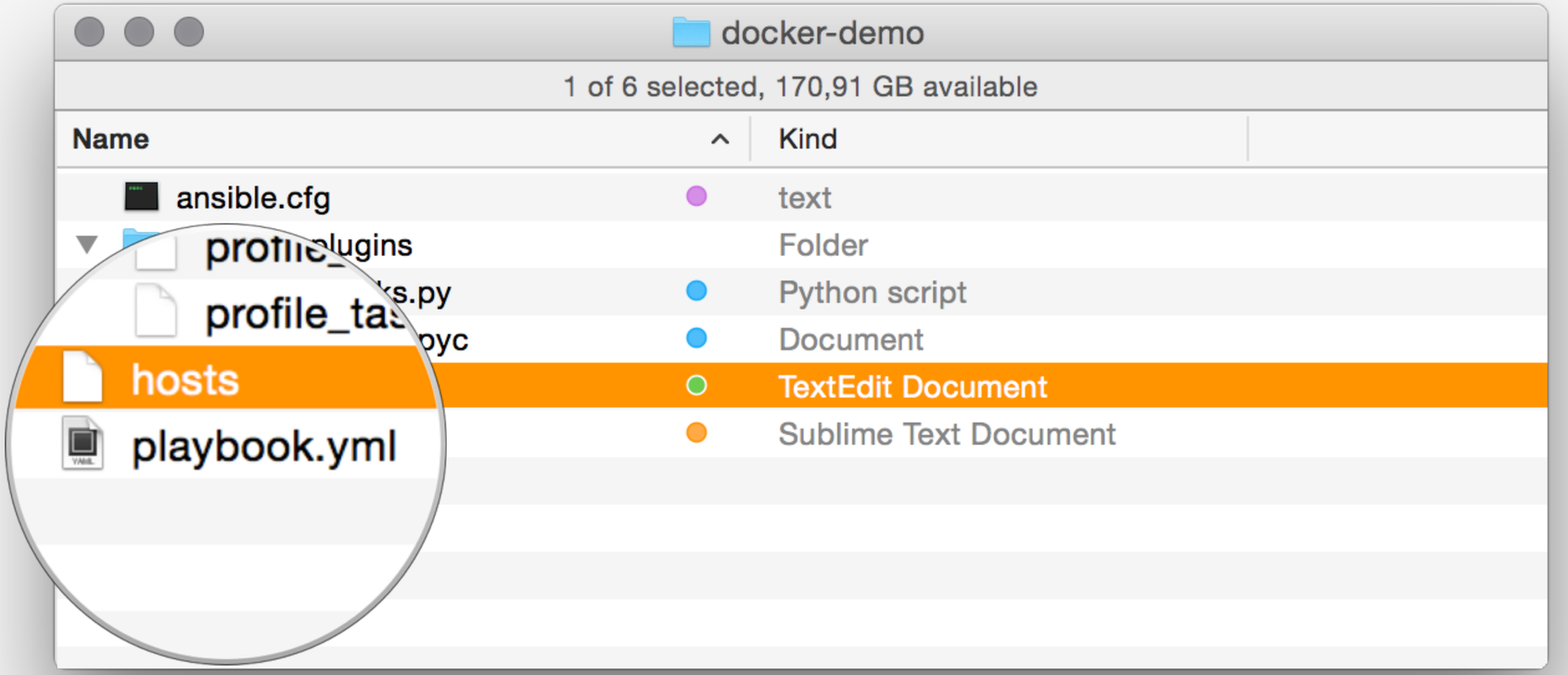
Describes your servers

IP Addresses or DNS Names

Grouped by Name

Vars for Groups

Inventory


docker-demo
1 of 6 selected, 170,91 GB available
Name
Kind
ansible.cfg
text
Folder
Python script
Document
hosts
TextEdit Document
playbook.yml
Sublime Text Document

Inventory

```
# simplest inventory entry is just an IP / hostname
[munki_host]
10.0.0.20

# hostname or FQDN with attach vars
[dockerhost]
dockerhost.macadmin.me ansible_ssh_host=192.168.191.167 ansible_ssh_user=root

# reference your local admin machine
[localhost]
localhost ansible_connection=local

```

Inventory

Syntax reference

# comments	comments begin with the '#' character
	Space and blank lines are ignored
[name]	Groups of hosts delimited by [header] elements
server / ip	Use hostname, FQDN or IP addresses
[multi:children]	Define a Groups of groups with child groups
[name:vars]	Define common vars for a group

Ad hoc calls

Ad hoc Ansible

```
# command pattern
ansible <group> -i <hosts> -m <module_name> -a <arguments> -s <optional:sudo>
```

```
# Run ad-hoc install and start of NTP/ntpd service
ansible munki_host -i hosts -m apt -a "pkg=ntp state=installed"
ansible munki_host -i hosts -m service -a "name=ntpd state=started enabled=yes"
```

Ad hoc Ansible

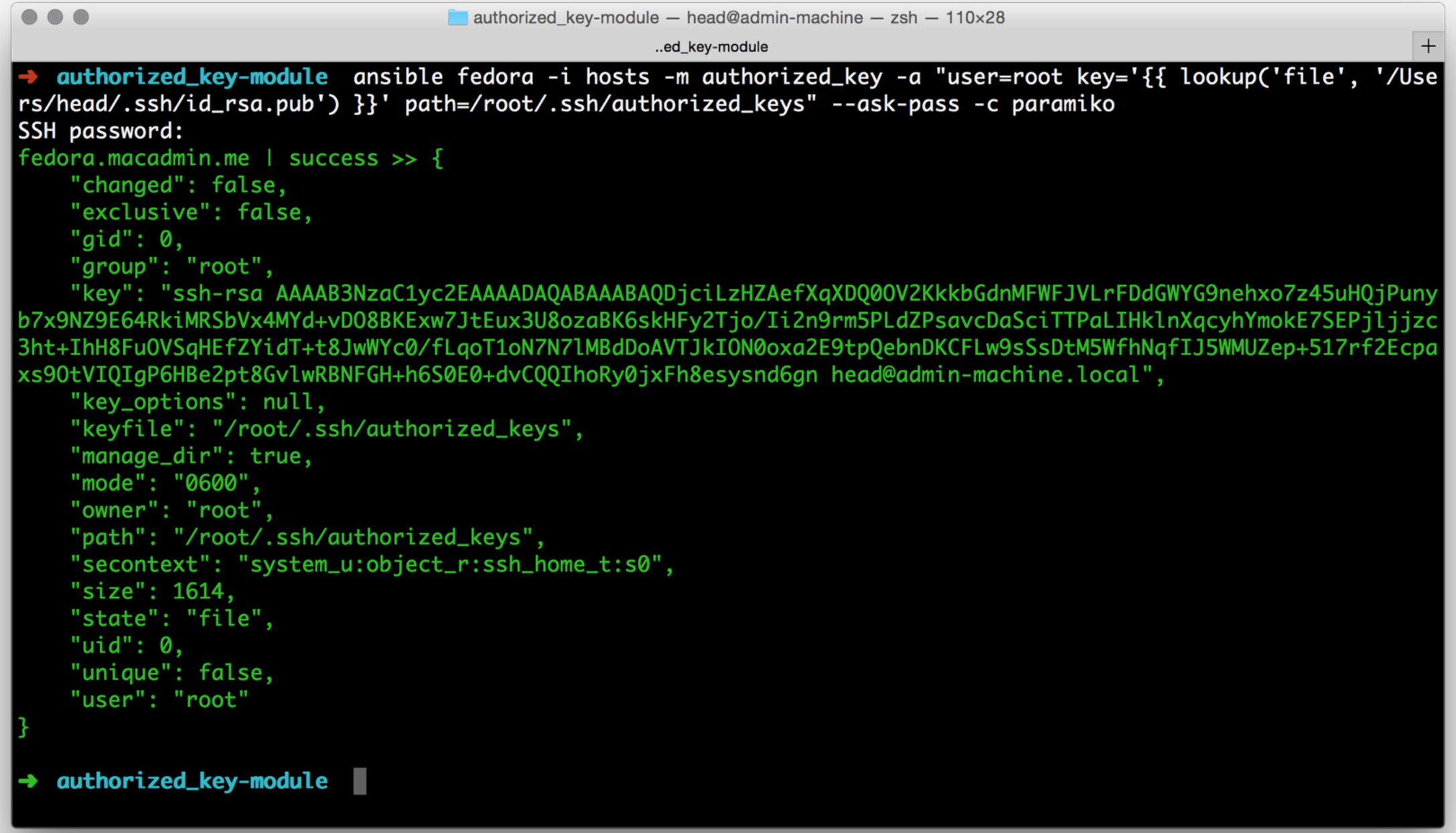
```
# restart jss tomcat server on linux
ansible jss-host -i hosts -m shell -a "/etc/init.d/jamf.tomcat7 restart"


# create munki repo folder structure
ansible munki_host -i hosts -m shell -a "mkdir -p /var/www/html/\
munki_repo/{catalogs,manifests,pkgs,pkgsinfo}"
```

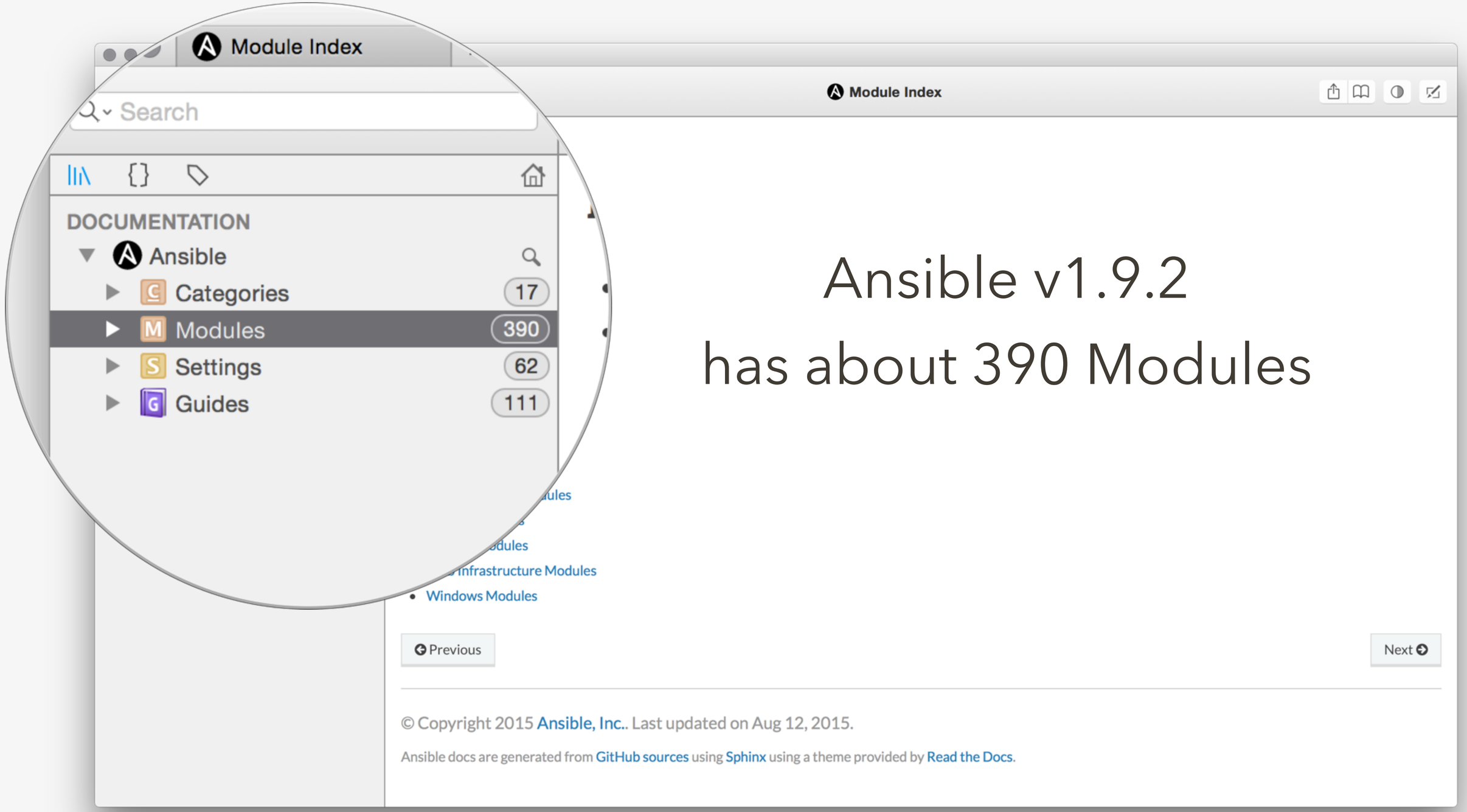
ssh-key bootstrap

```
# install ssh keys on remote host
ansible fedora -i hosts -m authorized_key -a "user=root \
key='{{ lookup('file', '/Users/macadmin/.ssh/id_rsa.pub') }}' \
path=/root/.ssh/authorized_keys" --ask-pass -c paramiko
```

*** Note: here we do a local lookup for read in the id_dsa.pub ssh-key and copy to remote host machine named "fedora"*


authorized_key-module — head@admin-machine — zsh — 110×28
..ed_key-module
#1
➜ authorized_key-module ansible fedora -i hosts -m authorized_key -a "user=root key='{{ lookup('file', '/Users/head/.ssh/id_rsa.pub') }}' path=/root/.ssh/authorized_keys" --ask-pass -c paramiko
#2
SSH password:
fedora.macadmin.me | success >> {
"changed": false,
"exclusive": false,
"gid": 0,
"group": "root",
"key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDjciLzHZAefXqXDQ0OV2KkkbGdnMFWFJVLrFDdGWYG9nehxo7z45uHQjPuny
#3
b7x9NZ9E64RkiMRSbVx4MYd+vDO8BKExw7JtEux3U8ozaBK6skHFy2Tjo/Ii2n9rm5PLdZPsavcDaSciTTPaLIHklnXqcyhYmokE7SEPjljjzc
3ht+IhH8FuOVSqHEfZYidT+t8JwWYc0/fLqoT1oN7N7lMBdDoAVTJkION0oxa2E9tpQebnDKCFLw9sSsDtM5WfhNqfIJ5WMUZep+517rf2Ecpa
xs9OtVIQIgP6HBe2pt8GvlwRBNFGH+h6S0E0+dvCQQIhoRy0jxFh8esysnd6gn head@admin-machine.local",
"key_options": null,
"keyfile": "/root/.ssh/authorized_keys",
"manage_dir": true,
"mode": "0600",
"owner": "root",
"path": "/root/.ssh/authorized_keys",
"secontext": "system_u:object_r:ssh_home_t:s0",
"size": 1614,
"state": "file",
"uid": 0,
"unique": false,
"user": "root"
}
➜ authorized_key-module

Ansible
modules & tasks



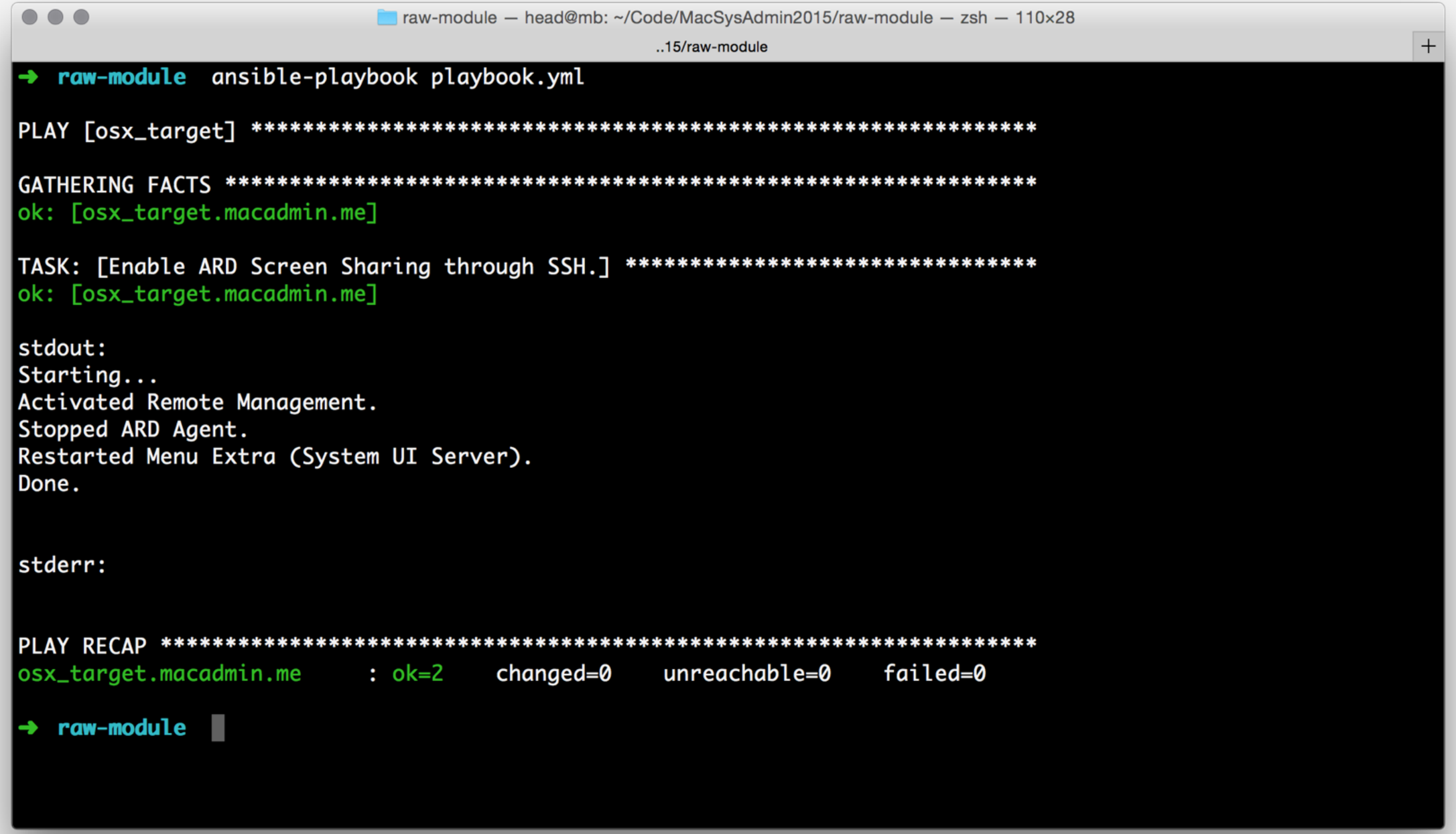
Ansible v1.9.2

has about 390 Modules

raw

Executes a low-down and dirty SSH command

```
1  tasks:
2  - name: Enable ARD Screen Sharing through SSH.
3    raw:  /System/Library/CoreServices/RemoteManagement/\
4          ARDAgent.app/Contents/Resources/kickstart \
5          -activate -configure -access -on -users admin \
6          -privs -all -restart -agent -menu
```

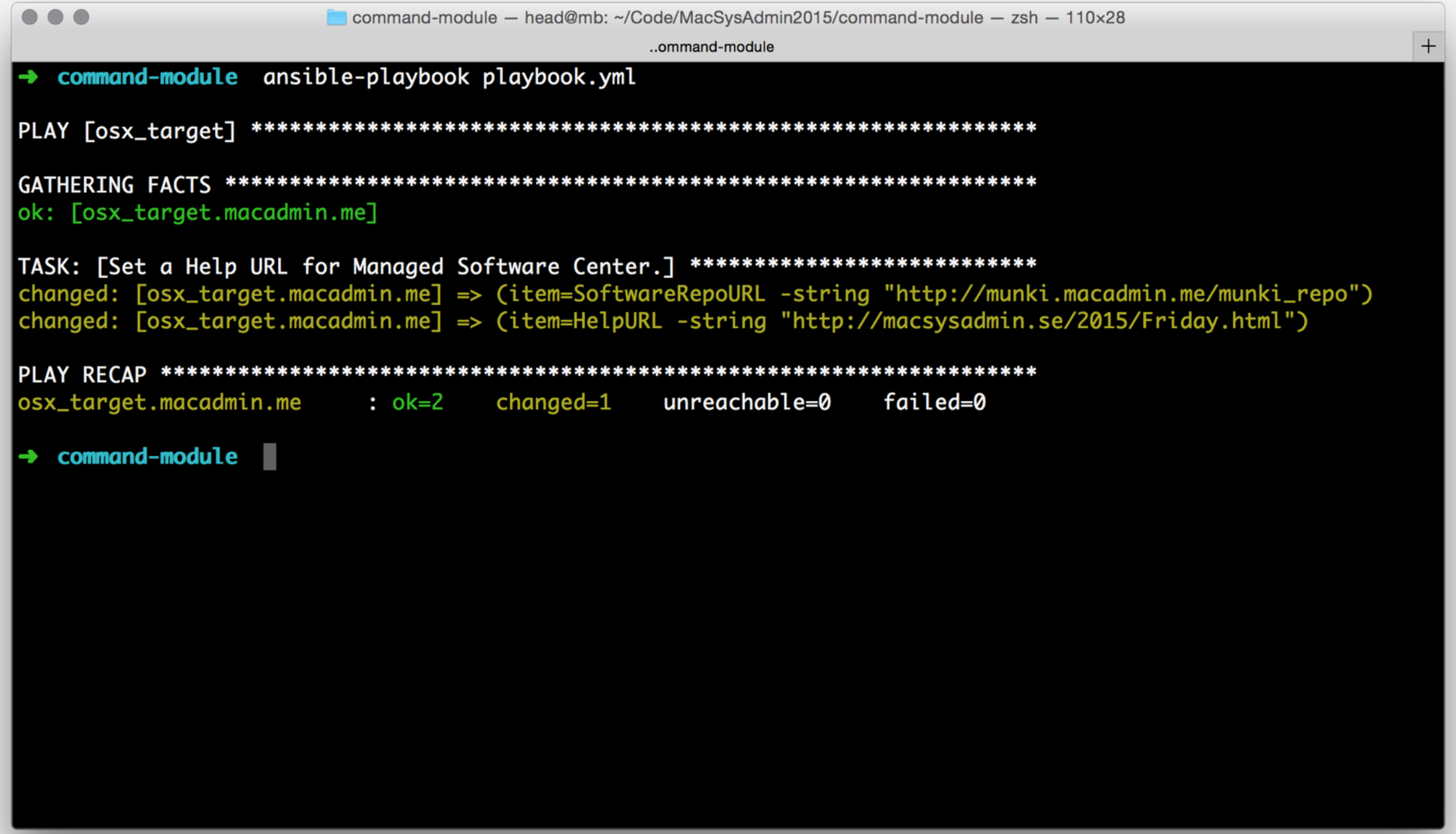

raw-module — head@mb: ~/Code/MacSysAdmin2015/raw-module — zsh — 110×28
..15/raw-module
➜ raw-module ansible-playbook playbook.yml
#0
PLAY [osx_target] **
GATHERING FACTS **
ok: [osx_target.macadmin.me]
#1
TASK: [Enable ARD Screen Sharing through SSH.] *********************************
ok: [osx_target.macadmin.me]
#2
stdout:
Starting...
Activated Remote Management.
Stopped ARD Agent.
Restarted Menu Extra (System UI Server).
Done.
stderr:
#3
PLAY RECAP **
osx_target.macadmin.me : ok=2 changed=0 unreachable=0 failed=0
➜ raw-module

command

Executes a command on remote node(s)

```
tasks:
- name: Set Repo and Help URLs for Munki / Managed Software Center.
  command: defaults write /Library/Preferences/ManagedInstalls {{ item }}
  with_items:
  - SoftwareRepoURL -string "http://munki.macadmin.me/munki_repo"
  - HelpURL -string "http://macsysadmin.se/2015/Friday.html"
```

*** Note: here we loop over the 'with_items:' list, inserted at {{item}} position*


command-module — head@mb: ~/Code/MacSysAdmin2015/command-module — zsh — 110×28
..ommand-module
➜ command-module ansible-playbook playbook.yml
#0
PLAY [osx_target] ***
GATHERING FACTS ***
ok: [osx_target.macadmin.me]
#1
TASK: [Set a Help URL for Managed Software Center.] ***************************
changed: [osx_target.macadmin.me] => (item=SoftwareRepoURL -string "http://munki.macadmin.me/munki_repo")
changed: [osx_target.macadmin.me] => (item=HelpURL -string "http://macsysadmin.se/2015/Friday.html")
#2
PLAY RECAP **
osx_target.macadmin.me : ok=2 changed=1 unreachable=0 failed=0
➜ command-module

apt / yum /dnf

Package managers for individual linux distributions

```
tasks:
- name: Install with apt for on Debian based OS
  apt:  name=apache2 state=latest
  when: ansible_os_family == 'Debian'

- name: Install with yum on CentOS distribution
  yum:  name=httpd state=latest
  when: ansible_distribution == "CentOS"

- name: Install with dnf package manager for Fedora 22
  dnf:  name=httpd state=latest
  when: ansible_distribution == "Fedora"
```

pip

Manages Python library dependencies

```
tasks:
- name: Install requests 2.7 HTTP library
  pip: name=requests version=2.7 virtualenv=/my_app/venv
```

mysql_db + mysql_user

Add or remove MySQL databases + users

```
tasks:
- name: Create a database for JSS
  mysql_db: name="jamfsoftware" state=present

- name: Create a user for our databases
  mysql_user: name="jamfsoftware" password=jamfsw03 \
  priv=*.*:ALL state=present
```

*** here we create a database and set user access*

mysql_db

Restore a MySQL database

```
tasks:
- name: Copy over the JSS dump file
  copy: src=jss_dump.sql.bz2 dest=/tmp

- name: Restore jss dump back into mysql_db
  mysql_db: name="jamfsoftware" state=import \
  target=/tmp/jss_dump.sql.bz2
```

```
** here we load a DB dump back into mysql
```

defaults

OS X defaults command module

```
tasks:
- name: Enable Safari Debug Menu in OS X.
  osx_defaults: domain=com.apple.Safari key=IncludeInternalDebugMenu \
  type=bool state=present
```

*** Note: Apple OS X caches defaults. You may need to logout and login to apply the changes.*

New
Ansible 2.0

mac_pkg

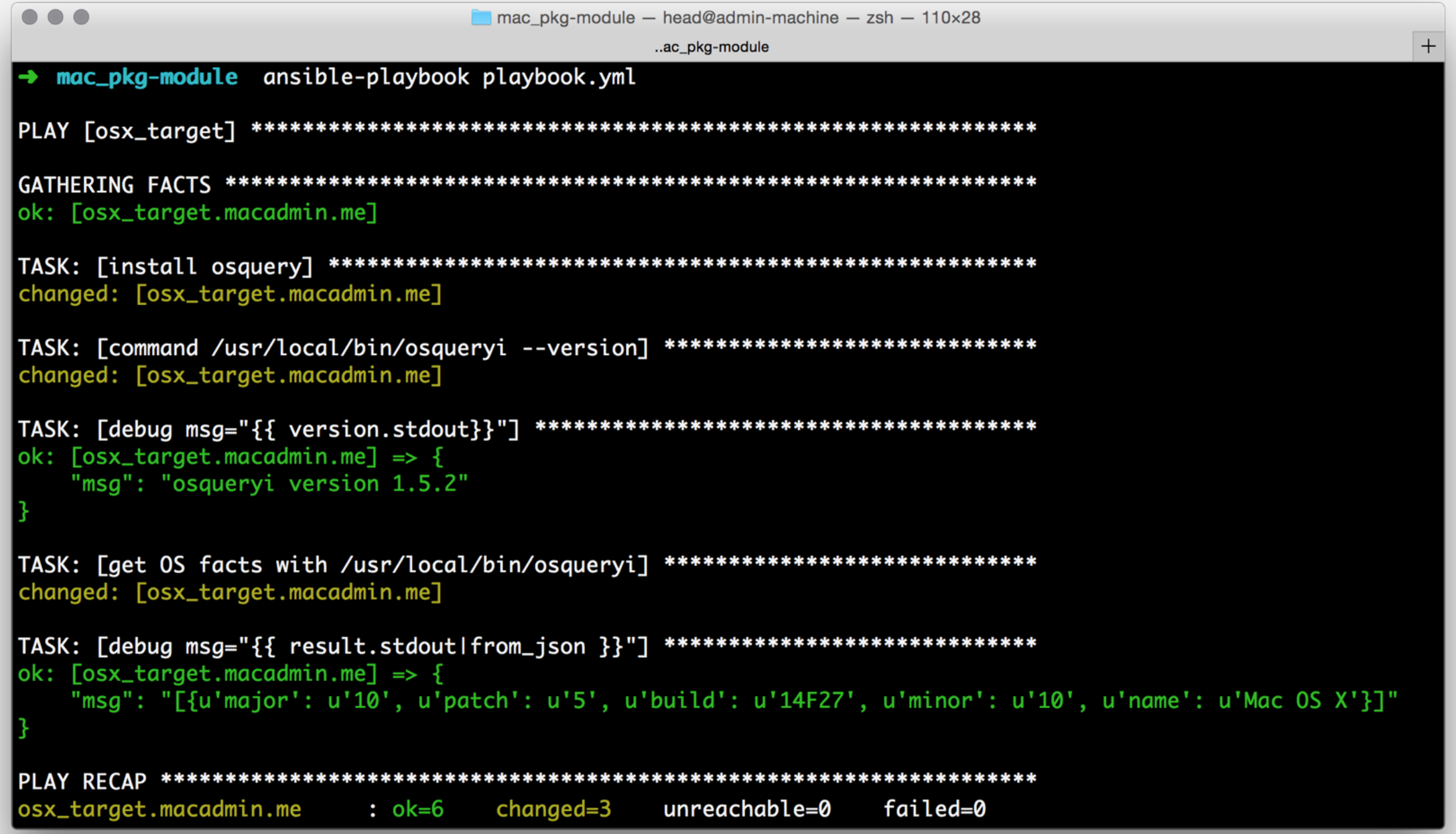
Executes installer or copy the bare .app from DMG or zip archive

```
vars:
- osquery_vers: 1.5.2
tasks:
- name: install osquery
  mac_pkg: pkg_name=osquery-{{osquery_vers}}
           url=https://osquery-packages.s3.amazonaws.com/darwin/osquery-\
           {{osquery_vers}}.pkg
           archive_path='osquery-{{osquery_vers}}.pkg'
           state=present
```

** *Note: this is a custom written Module* **

http://spencer.gibb.us/blog/2014/02/03/introducing-battleschool/

Custom module


mac_pkg-module — head@admin-machine — zsh — 110×28
..ac_pkg-module
➜ mac_pkg-module ansible-playbook playbook.yml
#0
PLAY [osx_target] ***
GATHERING FACTS ***
ok: [osx_target.macadmin.me]
#1
TASK: [install osquery] ***
changed: [osx_target.macadmin.me]
#2
TASK: [command /usr/local/bin/osqueryi --version] ***************************
changed: [osx_target.macadmin.me]
#3
TASK: [debug msg="{{ version.stdout}}"] *************************************
ok: [osx_target.macadmin.me] => {
"msg": "osqueryi version 1.5.2"
}
#4
TASK: [get OS facts with /usr/local/bin/osqueryi] ***************************
changed: [osx_target.macadmin.me]
#5
TASK: [debug msg="{{ result.stdout|from_json }}"] ***************************
ok: [osx_target.macadmin.me] => {
"msg": "[{u'major': u'10', u'patch': u'5', u'build': u'14F27', u'minor': u'10', u'name': u'Mac OS X'}]"
}
#6
PLAY RECAP **
osx_target.macadmin.me : ok=6 changed=3 unreachable=0 failed=0


Docker

Thoughts on Docker

Keep your Docker setup **minimal**

Automate everything possible

Consider complexity

Check if you really need to run
Kubernertes, etcd2....

Ansible and Docker

three main scenarios:

Use **docker** & **docker_image** modules *

Build & run Docker with the **shell** module

Running a **playbook** inside a container

Ansible docker

Manage docker containers - with shell module

```
tasks:
- shell: docker pull grahamgilbert/postgres
- name: run postgres-sal in docker container
  shell: >
      docker run -d --name="postgres-sal" \
      -v /db:/var/lib/postgresql/data \
      -e DB_NAME=sal \
      -e DB_USER=admin \
      -e DB_PASS=password \
      --restart="always" \
      grahamgilbert/postgres
```

Ansible docker

Manage docker containers - with shell module

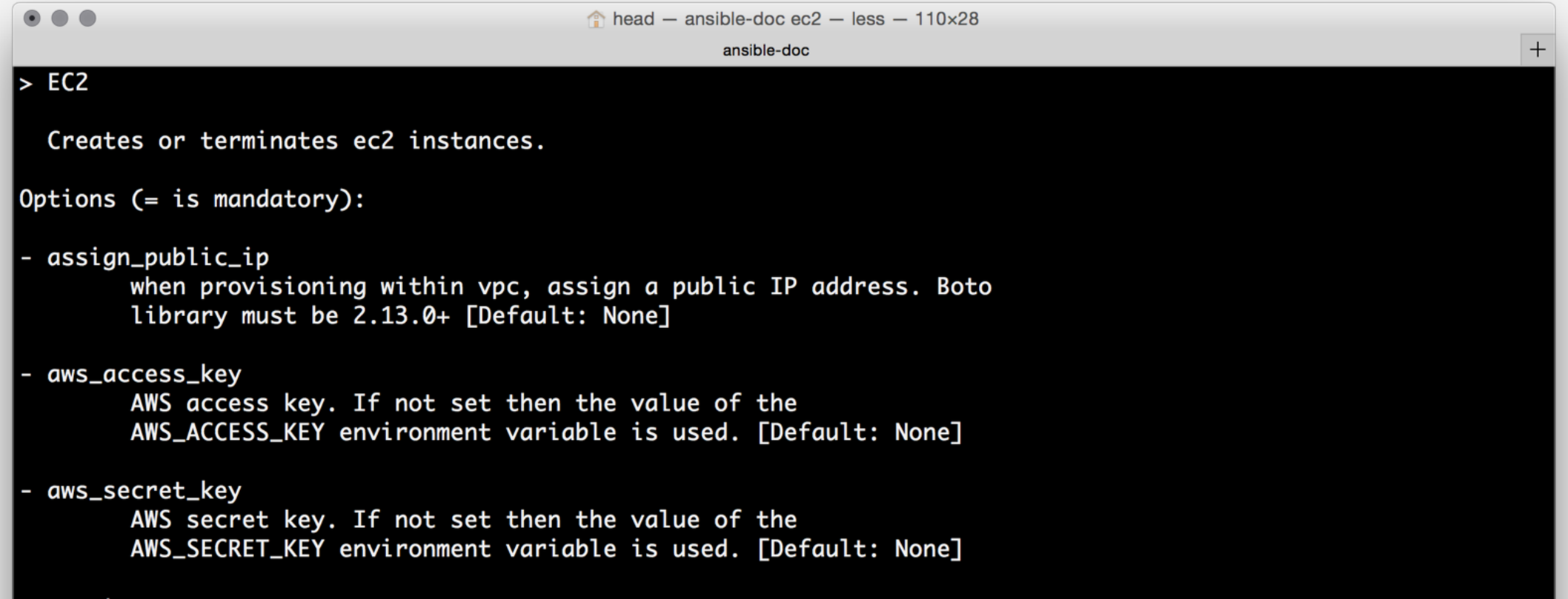
```
# tasks: continued…
- name: run sal 2.0.1 in docker container
  shell: >
      docker run -d --name="sal" \
      -p 80:8000 \
      --link postgres-sal:db \
      -e ADMIN_PASS=pass \
      -e DB_NAME=sal \
      -e DB_USER=admin \
      -e DB_PASS=password \
      --restart="always" \
      macadmins/sal:2.0.1
```

Demo

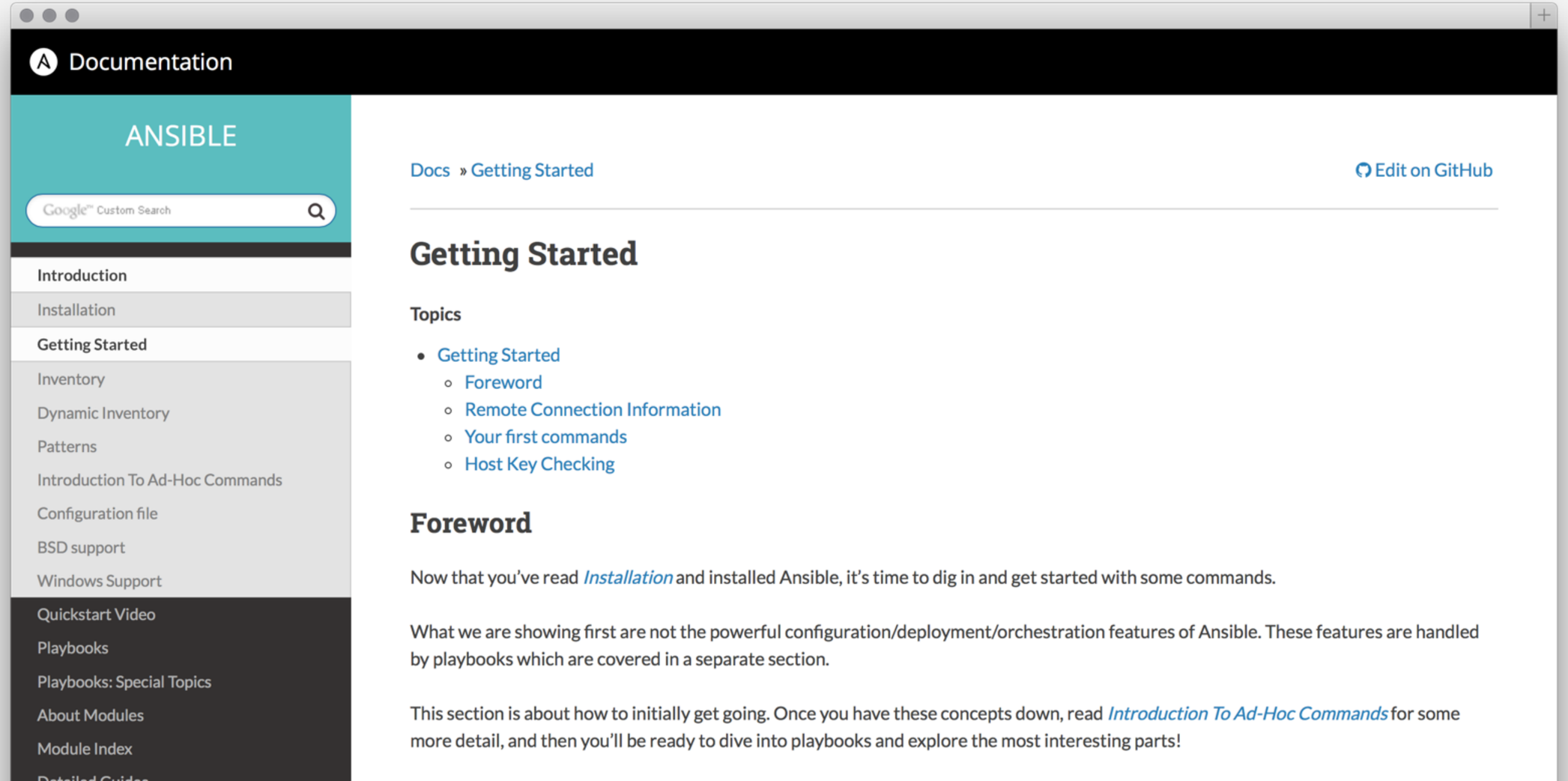
Documentation

ansible-doc

/usr/local/bin/ansible-doc <module_name>



docs.ansible.com



Roles & Galaxy

Roles

Roles

Organize tasks into **groups**

Call them **by name**

Run related tasks as a package

Apply variations of Roles to single machine or group of servers

Playbook + Roles

```
1  ---
2  - hosts: munki_server
3    roles:
4    - init
5
6  ---
7  - hosts: munki_server
8    vars:
9     - htpasswd_var=passW0rd
10 - roles:
11   - common
12   - nginx
13   - samba
14   - munkihost
```

 # First run - init / prepare

- create a user

- enable ssh-key login

 # Second run - deploy app

- setup & harden the server

- install web server

- install smb sharing

- setup munki_repo

Playbook + Roles

```
1  ---
2  - hosts: munki_server
3    roles:
4    - init
5
6  ---
7  - hosts: munki_server
8    vars:
9     - htpasswd_var=NEwPASS%H3Re
10 - roles:
11   - common
12   - nginx
13   - samba
14   - munkihost
```

 # First run - init / prepare

~~*- create a user*~~

~~*- enable ssh key login*~~

 # Second run - deploy app

~~*- setup & harden the server*~~

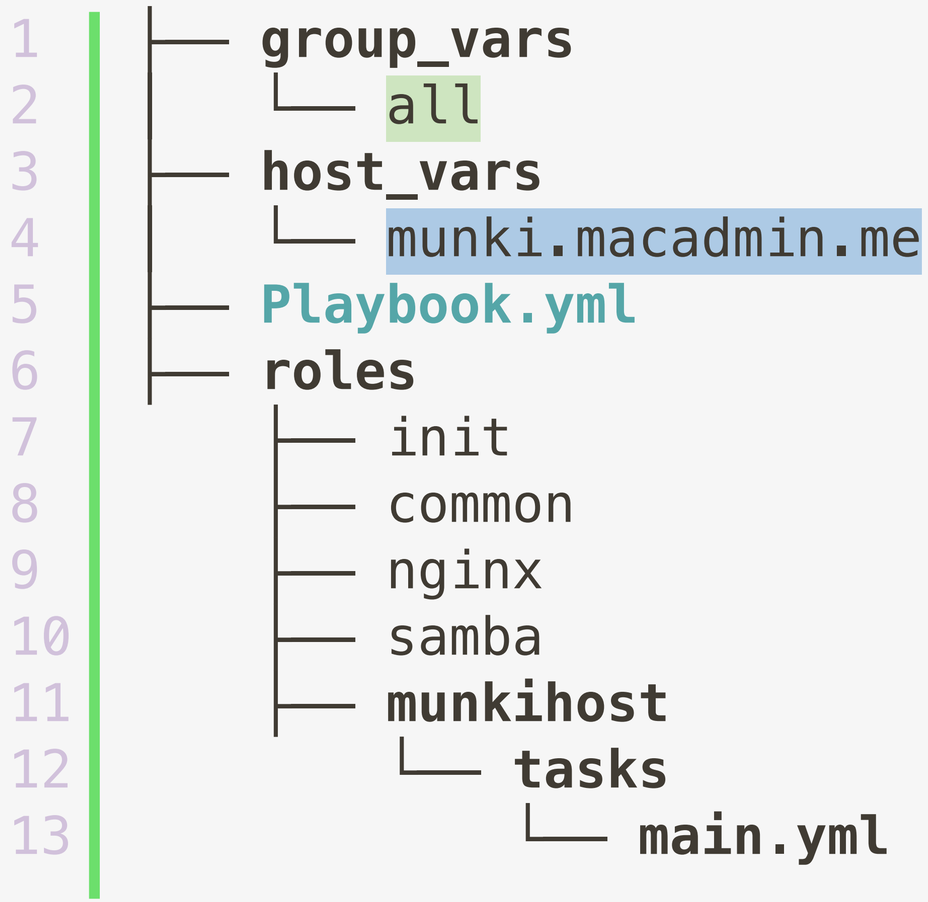
- update web server

idempotency applies here

~~*- install smb sharing*~~

~~*- setup munki_repo*~~

Playbook + Roles + Vars




```
# Group variable file for all hosts

# FQDN specific variable file
```

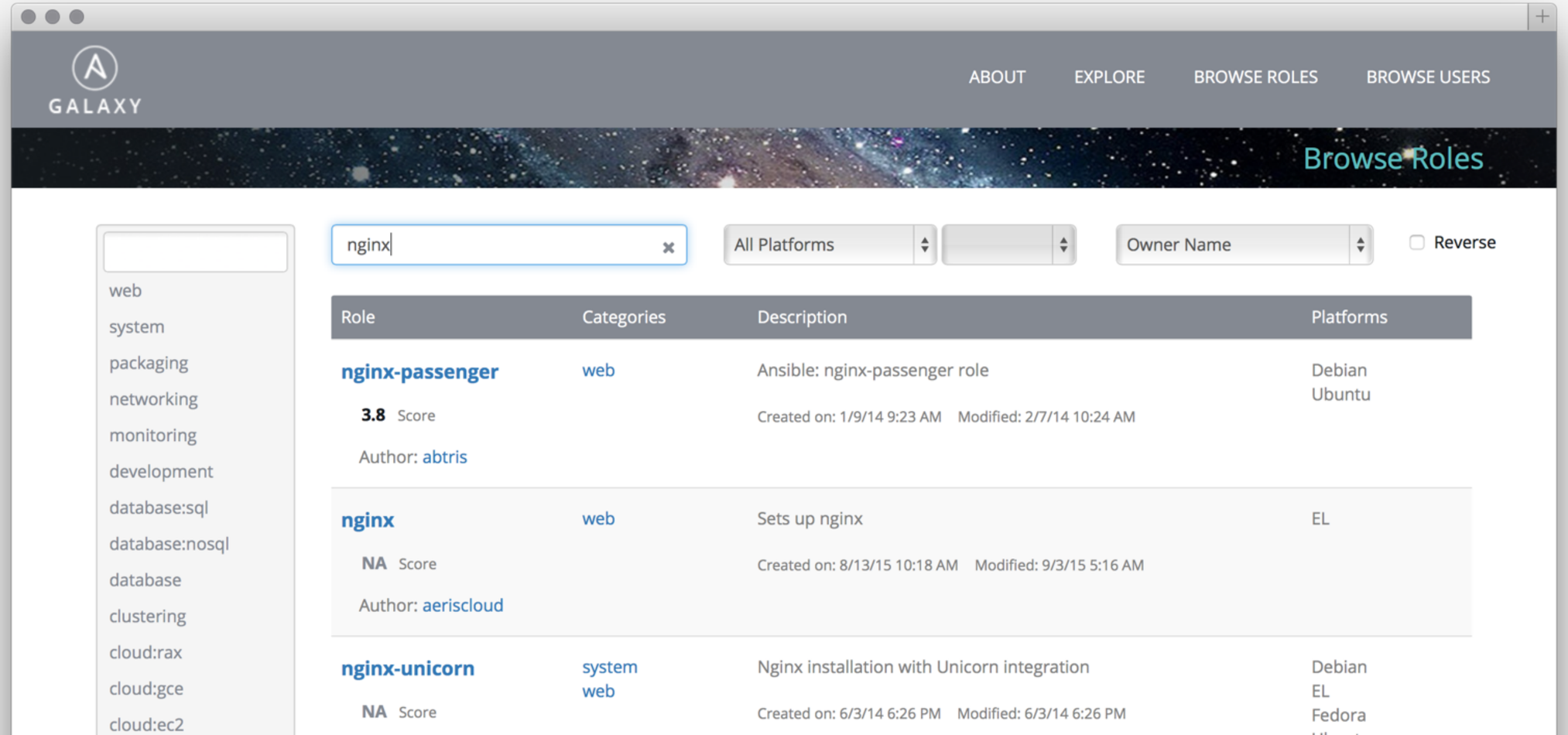
Playbook & Roles

group_vars: Group variable files for all hosts (all, debian, redhat)
host_vars: Hostname/FQDN specific variable files for specific hosts

```
---          # First run - init / prepare a system
hosts:       machine(s) you want to perform operations
vars:        variables for that play / include vars file
roles:       call the roles you want to perform on the host(s)
---          # Second run  - deploy an application
hosts:       machine(s) you want to perform operations
vars:        variables for that play / include vars file
roles:       call the roles you want to perform on the host(s)
```

Ansible Galaxy

https://galaxy.ansible.com



Ansible Galaxy

https://galaxy.ansible.com

Repository of existing Roles

Install common roles, reuse and learn

Share your own roles with

the community

Ansible 2.0

expected
late 2015

What's new in 2.0?

Refactored code base

Improved **vars handling**

Dynamic includes during execution

Blocks on execution

Improved exception handling

Blocks

```
1  tasks:
2    - block:
3        - name: Shell script to connect the app to a monitoring service.
4          docker:
5            name: postgres-sal
6            image: grahamgilbert/postgres
7            state: restarted
8      rescue:
9        - shell: docker pull grahamgilbert/postgres
10       - name: run postgres-sal in docker container
11         shell: docker run -d --name="postgres-sal" \ (continued..)
12     always:
13       - docker: name: sal image: macadmins/sal:2.0.1 state: absent
14       - name: run sal 2.0.1 in docker container
15         shell: docker run -d --name="sal" -p 80:8000 \ (continued..)
```

Wrap up

What you'll need to start

Know-how to use **Terminal**

Basic knowledge in **scripting**

Know-how to handle **ssh-keys**

Capability to **read a book** and/or the great written Ansible **documentation**

Alternative CfgMgmt Tools

BASH based: http://waffles.terrarum.net

Python based: http://saltstack.com

Ruby based: https://www.chef.io/chef/

https://puppetlabs.com

How to learn Ansible

Install Ansible, try out a tutorial:

http://docs.ansible.com

http://www.ansible.com/get-started

Handout for this talk:

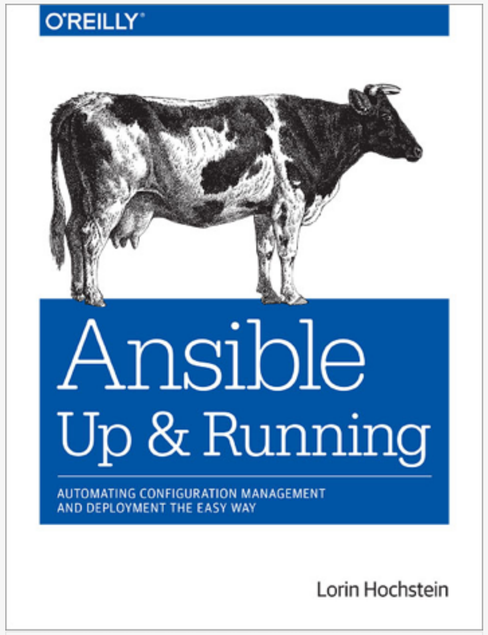
github.com/apfelwerk/macsysadmin2015

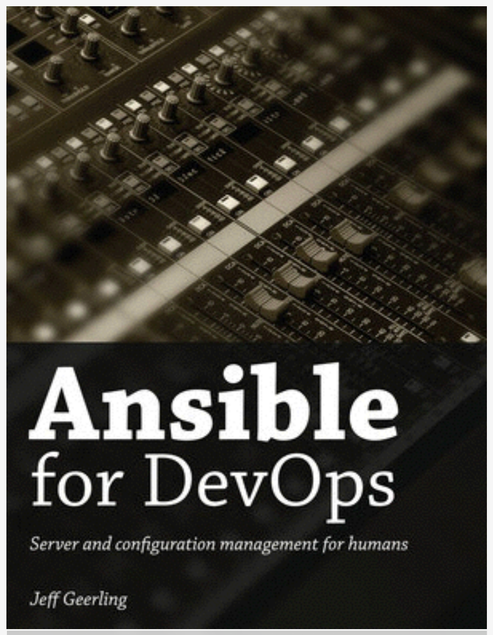
Books

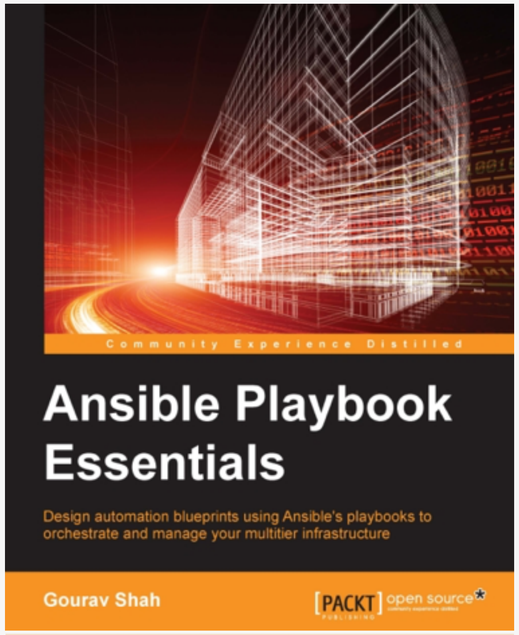
O'REILLY®
computer books • conferences • online publishing

Leanpub

[PACKT
PUBLISHING





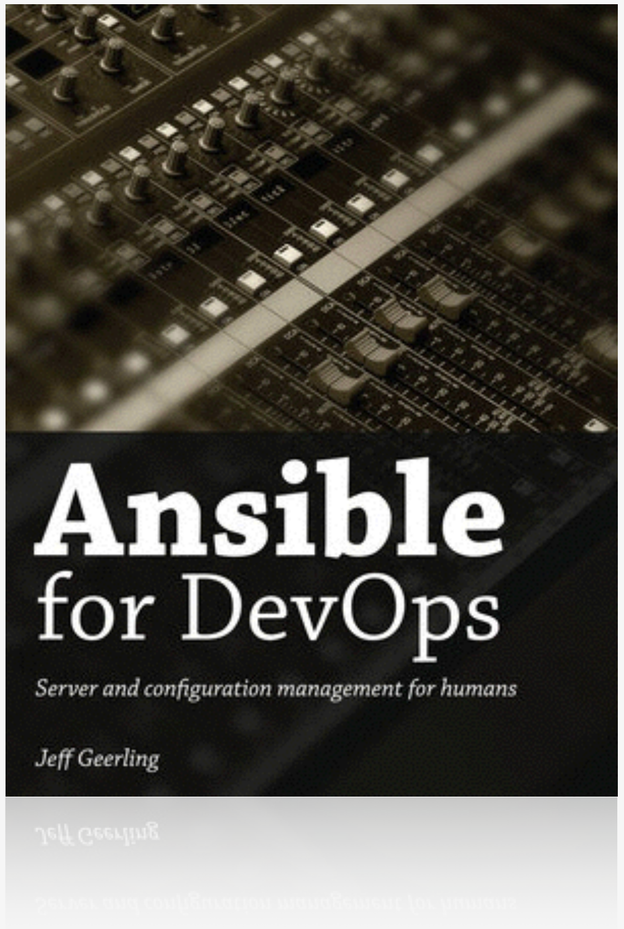


Recommended eBook

Ansible for DevOps

by Jeff Geerling

@geerlingguy



http://goo.gl/A2mubA

thank you!

sources

http://www.orangefreesounds.com/old-clock-ringing-short/

https://cdn1.iconfinder.com/data/icons/database-3/96/Big-Data-512.png

Feature image: “Hamburg Harbor” by Jasmin Bauomy is licensed under CC BY-SA 2.0.