AUTOMATION MADE SIMPLE

with

ansible

a talk by @erikaheidi
What to expect from this talk

1. Ansible Overview
2. Inventories / ad-hoc commands
3. Using Playbooks
4. Playbook crash-course
ANSIBLE
OVERVIEW
Ansible Overview

- Simple and Straightforward
- Human-readable automation language
- Agentless - needs only SSH
- Extensive list of built-in modules
- Used by Twitter, Atlassian, EA, Spotify, even NASA!

Automation made simple with Ansible
@erikaheidi
Installation

Mac OSX

$ brew update
$ brew install ansible

Ubuntu

$ sudo apt-add-repository -y ppa:ansible/ansible
$ sudo apt-get update
$ sudo apt-get install -y ansible

Detailed installation instructions: do.co/ansible-docs

*Windows is not officially supported as controller machine.
Setting up SSH access

- Servers should be accessible via SSH using keypair authentication
- It's recommended to have a user with sudo NOPASSWD permission to run the tasks in the server

How to configure your SSH access for running Ansible: bit.ly/ansible-ssh
Inventory file

# /etc/ansible/hosts
[webservers]
erikaheidi.com
dev-human.com

[testservers]
178.62.192.53
95.85.35.248
ad-hoc commands

ansible group [-i inventory] [-m module]

$ ansible all -m ping
$ ansible webservers -a "php -v"
$ ansible all -i staging -a "sudo apt-get update"
ad-hoc commands

```bash
+ ~ ansible all -m ping
95.85.35.248 | success >> {
  "changed": false,
  "ping": "pong"
}

dev-human.com | success >> {
  "changed": false,
  "ping": "pong"
}

erikaheidi.com | success >> {
  "changed": false,
  "ping": "pong"
}

178.62.192.53 | success >> {
  "changed": false,
  "ping": "pong"
}
```
ad-hoc commands

→ ~ ansible webservers -a "php -v"

dev-human.com | success | rc=0 >>
PHP 5.5.11-3+deb.sury.org~trusty+1 (cli) (built: Apr 23 2014 12:15:16)
Copyright (c) 1997-2014 The PHP Group
Zend Engine v2.5.0, Copyright (c) 1998-2014 Zend Technologies
    with Zend OPcache v7.0.4-dev, Copyright (c) 1999-2014, by Zend Technologies
    with Xdebug v2.2.3, Copyright (c) 2002-2013, by Derick Rethans

erikaheidi.com | success | rc=0 >>
PHP 5.5.17-2+deb.sury.org~trusty+1 (cli) (built: Sep 25 2014 09:06:49)
Copyright (c) 1997-2014 The PHP Group
Zend Engine v2.5.0, Copyright (c) 1998-2014 Zend Technologies
    with Zend OPcache v7.0.4-dev, Copyright (c) 1999-2014, by Zend Technologies

Automation made simple with Ansible
@erikaheidi
DEMO 1
Running ad-hoc commands
RUNNING PLAYBOOKS
---
- hosts: all
  sudo: true
  tasks:
    - name: Update apt-cache
      apt: update_cache=yes
    - name: Install Nginx
      apt: pkg=nginx state=latest
Running playbooks

ansible-playbook [-i inventory] [-l group] playbook.yml

$ ansible-playbook -i staging -l webservers playbook.yml
$ ansible-playbook playbook.yml --list-hosts
$ ansible-playbook playbook.yml --list-tasks
Running playbooks

```
→ demo ansible-playbook -l testservers playbook.yml

PLAY [all] *****************************************************

GATHERING FACTS *****************************************************
ok: [178.62.192.53]
ok: [95.85.35.248]

TASK: [Update apt-cache] *************************************
ok: [178.62.192.53]
ok: [95.85.35.248]

TASK: [Install Nginx] **************************************
ok: [178.62.192.53]
changed: [95.85.35.248]

PLAY RECAP ****************************************************
178.62.192.53        : ok=3  changed=0  unreachable=0  failed=0
95.85.35.248         : ok=3  changed=1  unreachable=0  failed=0
```

Automation made simple with Ansible
@erikaheidi
DEMO 2
ansible-playbook
PLAYBOOK CRASH-COURSE
---
- hosts: all
  sudo: yes
vars:
  web_server: nginx
tasks:
  - name: Install {{ web_server }}
    apt: pkg={{ web_server }} state=latest
Variables (facts)

- Information discovered from systems
- Globally available
- Example: `ansible_default_ipv4.address`

```bash
$ ansible erikaheidi.com -m setup
erikaheidi.com | success >> {
  "ansible_facts": {
    "ansible_all_ipv4_addresses": [ "178.62.160.46" ],
    "ansible_all_ipv6_addresses": [ "fe80::601:26ff:fe7a:a101" ]
  }
}
```
Loops (with_items)

tasks:
  - name: Install Packages
    apt: pkg=\{\{ item \}\} state=latest
  with_items:
    - nginx
    - php5-fpm
    - git
Loops (with_items)

```yaml
- hosts: all
  sudo: yes
  vars:
    sys_packages: [ 'nginx', 'php5-fpm', 'git' ]
  tasks:
    - name: Install Packages
      apt:
        pkg: {{ item }}
        state: latest
      with_items: sys_packages
```
- name: "shutdown Debian flavored systems"
  command: /sbin/shutdown -t now
  when: ansible_os_family == "Debian"

- name: check if bar is defined
  fail: msg="This play requires 'bar'"
  when: bar is not defined
Conditionals

- **name**: Check if PHP is installed
  
  `register: php_install`
  
  `command: php -v`
  
  `ignore_errors: true`

- **name**: Do something if PHP is installed
  
  `debug: var=php_install`
  
  `when: php_install|success`

- **name**: Do something if PHP is NOT installed
  
  `debug: msg='PHP is NOT installed!'
  
  `when: php_install|failed`
DEMO 3

conditionals
<VirtualHost *:80>
  ServerAdmin webmaster@localhost
  DocumentRoot {{ doc_root }}

  <Directory {{ doc_root }}>
    AllowOverride All
    Require all granted
  </Directory>

</VirtualHost>
Templates - Usage

- name: Change default apache vhost
template: src=templates/apache.tpl
dest=/etc/apache2/sites-available/000-default.conf
Handlers (services)

vars:
  - doc_root: /vagrant

tasks:
  - name: Change default apache vhost
    template: src=templates/apache.tpl
    dest=/etc/apache2/sites-available/000-default.conf
    notify: restart apache

handlers:
  - name: restart apache
    service: name=apache2 state=restarted
ORGANIZING YOUR PLAYS
Roles

```yaml
# playbook.yml
---
- hosts: all
  sudo: true
  vars:
    doc_root: /vagrant/web
  roles:
    - init
    - nginxphp
```
Includes

- playbook.yml
- init/tasks/main.yml
- nginxphp/handlers/main.yml
- nginxphp/tasks/main.yml
- nginxphp/tasks/php.yml
- nginxphp/templates/vhost.tpl

#roles/nginxphp/tasks/main.yml

- name: Install Nginx
  sudo: yes
  apt: pkg=nginx state=latest

- include: php.yml
# playbook.yml
---
- hosts: all
  sudo: true
  vars_files:
  - vars/all.yml
  roles:
    - init
    - nginxphp
    - vars/
OTHER COOL STUFF
- hosts: webservers
  sudo: true
  roles:
  - { role: server, tags: ['server'] }
  - { role: nginx, tags: ['nginx'] }
  - { role: php, tags: ['php'] }

$ ansible-playbook playbook.yml --tags "php"
- include: sharedfolders.yml
  when: app.env == 'prod'

---

- hosts: webservers
  sudo: true
  roles:
    - { role: firewall, when: app.env == 'prod' }

---

- hosts: webservers
  sudo: true
  vars_files:
    - [ "vars/{{ ansible_os_family }}.yml", "vars/os_defaults.yml" ]
Group Vars

```
# playbook.yml
---
- hosts: webservers
  sudo: true
  roles:
    - init
    - nginxphp
```
RESOURCES
```bash
$ ansible-galaxy install vendor.role [-p path]
```

**Automation made simple with Ansible**

@erikaheidi
Using Phansible with remote servers: bit.ly/phansible-remote

Automation made simple with Ansible
@erikaheidi
QUESTIONS?
THANKS!
@erikaheidi

Ansible Tutorials:
http://do.co/ansible