Ansible

A 5 Minute into to server automation

What is Ansible?

- A simple way to automate repetitive tasks
- It remembers what has been done (unlike shell scripts) – it is state driven.
- No central server needed (just run from a workstation).
- No agents need to be run on the server, it does it all via SSH.
- Even I can get a basic setup going in an hour or so

Installing and setup

- Simply use your favourite package manager to install from repos. Or follow instructions on www.ansible.com to install from their repo.
- Then you need to list your servers in /etc/ansible/hosts

```
#   - Groups of hosts are delimited by [header] elements
#   - You can enter hostnames or ip addresses
#   - A hostname/ip can be a member of multiple groups
# Ex 1: Ungrouped hosts, specify before any group headers.
green.example.com
192.168.100.10
# Ex 2: A collection of hosts belonging to the 'webservers' group
[webservers]
alpha.example.org
192.168.1.110
```

Server installation

- Probably nothing. All you need is python 2.4 or above. Which is included in most distros.
- If you're running Python 2.4 (!?!?!) then you'll need to have python-simplejson
- But python 2.5 or above shouldn't need anything.
- You also need to have passwordless SSH keys setup. Also if you want to use sudo, passwordless sudo.
- Eg in /etc/sudoers:
- ansible   ALL=(ALL)NOPASSWD: ALL

A Simple Example

- Lets just show you how to install one program on a server.
- Firstly your 'playbook' (aka script) is written in YAML (Yet Another Markup Language).
- So here is a small example which installs two programs.

```
- hosts: servers
  user: ansible
  sudo: True
  tasks:
  - name: install apache2
    action: apt pkg=apache2 state=installed
  - name: install postgresql
    action: apt pkg=postgresql-server state=installed
```

Output from a simple example

- Here is the output:

```
PLAY [grid *************************************

GATHERING FACTS ******************************************************************
ok: [grid.]

TASK: [install apache2] ***********************************************************
ok: [grid]

TASK: [install postgresql] *********************************************************
changed: [grid]

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

Some more complexity

- You can add variables to a playbook.
- You can use loops and conditionals
- Reference other playbooks, eg have a playbook to install a LAMP stack, a playbook to setup backups etc
- Roles – extra abstraction for playbooks – reusable across playbooks
- Then mix and match to build a server (or group of servers) how you want to.
- It's possible to get fairly complex.
- But generally Ansible is a simple system and it's best to keep it simple.

Modules

- There are around 200 modules which you can use to build your playbook.
- You can write your own (it's Python)
- Example modules include, apt, yum, bzr, cpanm, django_manage, docker, git, lvg (lvm), mail, script, service
- They do what they say on the tin, eg service, can be used to start and stop a service.

Module Examples

- Here are some examples:

```
# Install (Bottle) python package on version 0.11.
- pip: name=bottle version=0.11
# Example action to stop service httpd, if running
- service: name=httpd state=stopped
# Create a new database with name 'bobdata'
- mysql_db: name=bobdata state=present
# Add a user to a password file and ensure permissions are set
- htpasswd: path=/etc/nginx/passwdfile name=janedoe
password=9s36?;fyNp owner=root group=www-data mode=0640
# Create a new Droplet
- digital_ocean: >
      state=present      command=droplet      name=mydroplet
      client_id=XXX      api_key=XXX      size_id=1      region_id=2
      image_id=3      wait_timeout=500
  register: my_droplet
```

Finally...

- Lots of examples out there on the web.
- Ansible Galaxy – like CPAN (or similar) but for Ansible.
- Documentation on the website – www.ansible.com - is really good.
- There's also a good IRC channel on Freenode
- Any questions?