RIPE NCC
RIPE NETWORK COORDINATION CENTRE

Ansible

A brief introduction

Anand Buddhdev | 22 Jun 2016 | SINOG 3

What is Ansible?

- A fictional machine, capable of **instantaneous** communication
- Ursula le Guin created this word in 1966, from the word “answerable”
- Star Trek communicators are examples of ansibles

Ansible in IT

- An simple and lightweight automation tool
- Execute one-time tasks, such as rebooting servers
- Perform system administration tasks such as adding users, installing packages, etc
- Configure servers and routers
- Gather information about devices

Features of Ansible

- Written in Python
  - easy to read and extend
- Open source
  - source code on GitHub
- Easy to install and run
  - get started in just a few minutes
- Scales from a handful of systems to hundreds
  - 585 hosts being managed at RIPE NCC

Installing Ansible

Requirements

- Python 2
- OpenSSH
- Python modules
  - Jinja2, MarkupSafe
  - PyYAML
  - paramiko, pycrypto (optional)
  - dnspython (optional)
  - netaddr (optional)

Installation

- CentOS / Fedora
  - enable the EPEL repository
  - yum install ansible
- Ubuntu / Debian
  - apt-add-repository ppa:ansible/ansible
  - apt-get install ansible
- Any operating system
  - create a virtualenv
  - pip install ansible

Components

Section subtitle

Modules

- The workhorses of Ansible
- Perform all manner of tasks on systems, eg:
  - copy files
  - start, stop or restart services
  - install or remove packages
  - configure firewalls
  - create, modify or remove cron jobs
  - create, modify or remove users and groups

Executables

- ansible (one-off commands)
- ansible-playbook (many tasks)
- ansible-console (interactive one-off tasks)
- ansible-doc (module documentation)
- ansible-galaxy (install third-party roles)
- ansible-pull (agent, for pull-mode)
- ansible-vault (encrypted data)

Inventory

- A file containing all managed hostnames
- Allows arbitrary grouping of hosts
- Can be a directory
  - file names are group names
  - file contents are concatenated
- Can also be an executable program
  - should output lists of hosts and groups

Example inventory

```
loner.example.com

[web_servers]
www1.example.com
www2.example.com

[dns_servers]
ns1.ch-gva.k.ripe.net
ns2.kw-kwi.k.ripe.net

[imap_servers]
postboy.ripe.net
postgirl.ripe.net

[hdfs_data_nodes]
node[00:19].bigdata.cloud
```

Running Ansible

How it works

- Ansible connects to host over ssh
- Creates a temporary directory
- Copies the module and its parameters into the temporary directory
- Runs the module from the temporary directory
- Gathers the result to report back
- Deletes the temporary directory

One-off commands

```
$ ansible -m <module> -a <paramaters> <hosts>

$ ansible -m ping all
loner.example.com | success >> {“changed”: false, “ping”: “pong”}
www1.example.com | success >> {“changed”: false, “ping”: “pong”}
www2.example.com | success >> {“changed”: false, “ping”: “pong”}
ns1.ch-gva.k.ripe.net | success >> {“changed”: false, “ping”: “pong”}
ns2.kw-kwi.k.ripe.net | success >> {“changed”: false, “ping”: “pong”}
postboy.ripe.net | success >> {“changed”: false, “ping”: “pong”}
postgirl.ripe.net | success >> {“changed”: false, “ping”: “pong”}
node00.bigdata.cloud | success >> {“changed”: false, “ping”: “pong”}
node01.bigdata.cloud | success >> {“changed”: false, “ping”: “pong”}
…
node19.bigdata.cloud | success >> {“changed”: false, “ping”: “pong”}
```

One-off commands

```
$ ansible -m command -a whoami postboy.ripe.net
postboy.ripe.net | SUCCESS | rc=0 >>
anandb


$ ansible -m command -a 'ls -l /etc/passwd' dns_servers
ns1.ch-gva.k.ripe.net | SUCCESS | rc=0 >>
-rw-r--r-- 1 root root 1563 Apr  5 15:18 /etc/passwd

ns2.kw-kwi.k.ripe.net | SUCCESS | rc=0 >>
-rw-r--r-- 1 root root 1563 Apr 20 12:43 /etc/passwd


$ ansible -a 'ls /etc/group' -o dns_servers
ns1.ch-gva.k.ripe.net | SUCCESS | rc=0 | (stdout) /etc/group
ns2.kw-kwi.k.ripe.net | SUCCESS | rc=0 | (stdout) /etc/group
```

Privilege escalation

```
$ ansible -m <module> -a <params> -b -K

# -b (become root, using sudo)
# -K (prompt for sudo password)


$ ansible -m yum -a name=tcpdump -b -K www2.example.com
SUDO password:
www2.example.com | success >> { "changed": true, "msg": "", "rc": 0,
"results": [ .. ] }
```

```
$ ansible -m yum -a name=tcpdump -b -K web_servers
SUDO password:
www1.example.com | success >> { "changed": true, "msg": "", "rc": 0,
"results": [ .. ] }
www2.example.com | success >> { "changed": false, "msg": "", "rc": 0,
"results": [ .. ] }
```

A common sequence

```
$ ansible -m yum -a name=nsd -bK dns_servers

$ ansible -m file -a 'path=/var/nsd owner=root group=nsd mode=0775' -bK
  dns_servers

$ ansible -m copy -a 'src=nsd.conf dest=/etc/nsd/nsd.conf' -bK dns_servers

$ ansible -m service -a 'name=nsd state=started' -bK dns_servers
```

Batching tasks

Playbooks

- Recipes of what to do, and on which hosts
- Written in YAML
  - human-readable
- Variable definitions
- Handlers
  - take action upon changes
- Reusable
  - Save for the future, share with colleagues, etc

A simple playbook

```
$ vi manage_nsd.yml


- hosts: dns_servers
  tasks:
  - name: install nsd
    yum: name=nsd
  - name: create database directory
    file: path=/var/nsd state=directory owner=root group=nsd mode=0775
  - name: copy config file
    copy: src=nsd.conf dest=/etc/nsd/nsd.conf
    notify: restart nsd
 handlers:
 - name: restart nsd
   service: name=nsd state=restarted
```

Running a playbook

```
$ ansible-playbook -bK manage_nsd.yml

PLAY [dns_servers] ************************************************************

GATHERING FACTS ***************************************************************
ok: [ns1.ch-gva.k.ripe.net]
ok: [ns1.kw-kwi.k.ripe.net]

TASK: [install nsd] ***********************************************************
changed: [ns1.ch-gva.k.ripe.net]
changed: [ns1.kw-kwi.k.ripe.net]

TASK: [create database directory] *********************************************
changed: [ns1.ch-gva.k.ripe.net]
changed: [ns1.kw-kwi.k.ripe.net]

TASK: [copy config file] ******************************************************
changed: [ns1.ch-gva.k.ripe.net]
changed: [ns1.kw-kwi.k.ripe.net]

NOTIFIED: [restart nsd] *******************************************************
changed: [ns1.ch-gva.k.ripe.net]
changed: [ns1.kw-kwi.k.ripe.net]

PLAY RECAP ********************************************************************
ns1.ch-gva.k.ripe.net  : ok=5  changed=4  unreachable=0  failed=0
ns1.kw-kwi.k.ripe.net  : ok=5  changed=4  unreachable=0  failed=0
```

Idempotence

```
$ ansible-playbook -bK manage_nsd.yml

PLAY [dns_servers] ************************************************************

GATHERING FACTS ***************************************************************
ok: [ns1.ch-gva.k.ripe.net]
ok: [ns1.kw-kwi.k.ripe.net]

TASK: [install nsd] ***********************************************************
ok: [ns1.ch-gva.k.ripe.net]
ok: [ns1.kw-kwi.k.ripe.net]

TASK: [create database directory] *********************************************
ok: [ns1.ch-gva.k.ripe.net]
ok: [ns1.kw-kwi.k.ripe.net]

TASK: [copy config file] ******************************************************
ok: [ns1.ch-gva.k.ripe.net]
ok: [ns1.kw-kwi.k.ripe.net]

PLAY RECAP ********************************************************************
ns1.ch-gva.k.ripe.net  : ok=4  changed=0  unreachable=0  failed=0
ns1.kw-kwi.k.ripe.net  : ok=4  changed=0  unreachable=0  failed=0
```

Templates

- Ansible uses the Jinja template engine
  - variable substitution
  - conditionals and loops (if, for)
  - filters to transform text
- Templates are just text files
  - text between {{ … }} undergoes variable substitution
  - text between {% … %} marks conditionals and loops
  - everything else is passed through

Playbook with variables

```
$ vi manage_nsd.yml


- hosts: dns_servers
  vars:
    nsd_procs: 8
    zones:
    - arpa.
    - root-servers.net.
  tasks:
  - name: nsd config
    template: src=nsd.conf.j2 dest=/etc/nsd/nsd.conf
    notify: restart nsd
  handlers:
  - name: restart nsd
    service: name=nsd state=restarted
```

nsd.conf.j2

```
# my nsd configuration
server:
    server-count: {{ nsd_procs }}
    identity: ascii_{{ ansible_fqdn }}
{% for x in range(5) %}
    ip-address: 193.0.9.{{ x }}
{% endfor %}

{% for z in zones %}
zone:
    name: {{ z }}
    request-xfr: 1.2.3.4
{% endfor %}
```

nsd.conf on ns1.ch-gva.k.ripe.net

```
# my nsd configuration
server:
  server-count: 8
  identity: ascii_ns1.ch-gva.k.ripe.net
  ip-address: 193.0.9.0
  ip-address: 193.0.9.1
  ip-address: 193.0.9.2
  ip-address: 193.0.9.3
  ip-address: 193.0.9.4

zone:
  name: arpa.
  request-xfr: 1.2.3.4
zone:
  name: root-servers.net.
  request-xfr: 1.2.3.4
```

Conditional task execution

```
$ ansible-playbook -K reboot.yml


- hosts: all
  serial: 1
  tasks:
  - name: reboot
    command: /sbin/reboot
    become: yes
    when: "ansible_kernel != '2.6.32-642.1.1.el6.x86_64'"
  - name: wait for host to return
    local_action: wait_for delay=30 host={{ inventory_hostname }} port=22
    when: "ansible_kernel != '2.6.32-642.1.1.el6.x86_64'"
```

Take-home summary

- Ansible is easy to learn, yet powerful
- Install it on your laptop NOW
  - very light requirements
- Use it for automating common tasks
  - use ansible to run cf-agent or puppet agent ;-)
- Head over to http://docs.ansible.com
  - excellent documentation

Questions

anandb@ripe.net
@aabdnn