Technote on ansible basics & Patching through ansible on Linux

ANSIBLE IT Automation TOOL & FEATURES

- Ansible is a free open source application
- Agent-less functionality
- Configuration of Servers
- Application Deployment
- Continuous testing of already installed application
- Automation of tasks, Provisioning and Orchestration
- Based on Python/YAML. There is no need to understand Python to use Ansible
- Tasks are written in codes, Eliminate Errors and Configuration inconsistency
- Highly flexible & Scalable
- Large number of ready to use modules for systems management
- Custom modules could be added if needed
- Configuration roll-back in case of error
- Simple and human readable, easy installation & configuration
- Self-documenting, Write once, Use multiple times
- Large number of ready to use modules for systems management
- Custom modules could be added if needed
- Simple and human readable, easy installation & configuration

ANSIBLE REQUIREMENTS:

- ANSIBLE package Avaibale on EPEL Repository
- SSh Client (OpenSSH), Password Less Authentication for non-root user (ansible) from Control System to Client Systems
- Sudo Previlliage: To execute all or selective commands without password prompt on client systems
- Python

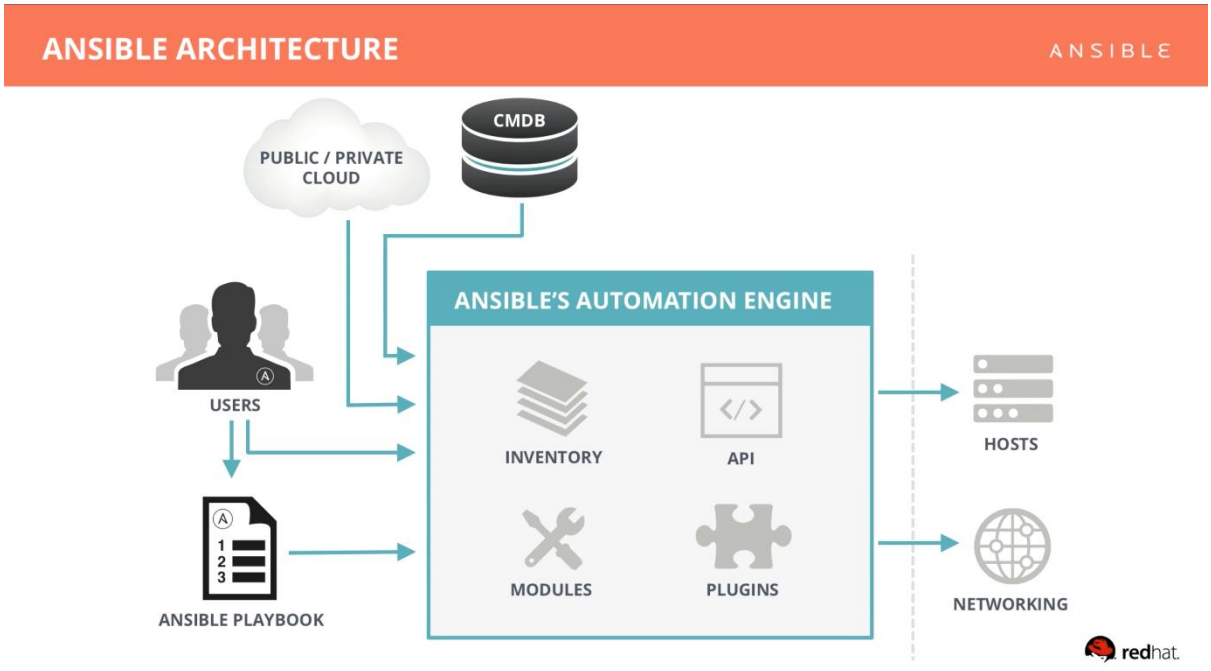
ANSIBLE COMPONENTS

- **Program**: ansible adhoc commands & ansible playbook are Programs of Ansible
- **Modules**: Set of instruction for a particular type configuration management
- Modules perform configuration and system management examples Modules Copy, Service, File, Yum, user, Group

ANSIBLE PLAYBOOK

- In Ansible terms, play is like a task/command. A set of commands grouped together to serve a defined purpose can be termed as a playbook. A playbook written in YAML is very sensitive to "space" and "indentation".
- YAML stands for Yet Another Markup Language. Ansible uses YAML to instruct the Automation Framework to perform tasks. YAML is very space sensitive. Hence, it will be better to setup indentation in text editors. Failing to do so will fail the syntax

ANSIBLE ARCHITECT



LAB SETUP

ANSIBLE CONTROLLER: RHEL7

[root@ansible-controller ~]# cat /etc/redhat-release

CentOS Linux release 7.4.1708 (Core)

[root@ansible-controller ~]# rpm -qa|grep ansible

ansible-2.4.2.0-1.el7.noarch

[root@ansible-controller ~]#

CLIENT SERVER: RHEL6

[root@cman1 ~]# cat /etc/redhat-release

CentOS release 6.9 (Final)

[root@cman1 ~]#

User "ansible" is created as ansible admin user (we can have of any name) with "/home/ansible" as home dir on Controller & across clients. Set for password less communication from Controller to all clients and permitted for sudo root access on clients

---

[root@ansible-controller ~]# ansible --version

ansible 2.4.2.0

config file = /etc/ansible/ansible.cfg

configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']

ansible python module location = /usr/lib/python2.7/site-packages/ansible

executable location = /usr/bin/ansible

python version = 2.7.5 (default, Aug 4 2017, 00:39:18) [GCC 4.8.5 20150623 (Red Hat 4.8.5-16)]

Ansible main config and binary files:

/etc/ansible/ansible.cfg ## All ansible related configuration

/etc/ansible/hosts ### Client Inventory Groups

Ansible binary/commands:

/usr/bin/ansible #### Run a single task against a set of hosts

/usr/bin/ansible-playbook ## Runs Ansible playbooks, executes defined tasks on the targeted hosts

/usr/bin/ansible-doc ## displays information on modules installed in Ansible libraries.

/usr/bin/ansible-galaxy ## command to manage Ansible roles in shared repostories, the default of which is Ansible Galaxy https://galaxy.ansible.com

/usr/bin/ansible-vault ## encryption/decryption utility for Ansible data files

/usr/bin/ansible-pull ## pulls playbooks from a Version Control System(VCS) repository

/usr/bin/ansible-inventory ## used to display or dump the configured inventory as Ansible sees it

/usr/bin/ansible-config ## View, edit, and manage ansible configuration

/usr/bin/ansible-console ##REPL console, that allows for running ad-hoc tasks against a chosen inventory

---

Invenstory file on ansible controller, Below as example

[ansible@ansible-controller ~]$ cat /etc/ansible/hosts|grep -v "#"

[NONRH]

ubuntu

[APPSERVER]

appserver

[DBSERVER]

dbserver

[NEWGROUP]

cman1

---

ansible adhoc commands: To achieve an task on remote machine

#examples adhoc commands

#ansible-doc –l |more ###list all available modules

#ansible-doc –s yum ## List detail/options available with module

#ansible appgroup –m ping ## Check ping response for hosts under appgroup inventory grpup

#ansible all –m ping ### Check ping response for all serevrs in ansible Server inventory

#ansible all –m shell –a “uname –a;df –h” ## shell commands o/p

ansible -m setup all ## Gathers facts about remote hosts

##More examples of adhoc command here –s is used to allow ansible user to use sudo access on remote machine for root commands

#ansible appgroup –m yum –a “name=httpd state =present” -s ## yum module to install pkg httpd

#ansible appgroup –m service –a “name=httpd state=started” -s # service module start httpd service

#ansible webgroup -m service -a "name=httpd enabled=yes" –s ## set service as enabled

#ansible appgroup –m copy –a “src=/tmp/file dest=/tmp/file” –s ##use of copy module

##Below example is to check hosts information(OS Distribution) for Servers under NEWGROUP ansible Inventory

```
[ansible@ansible-controller ~]$ ansible -m setup NEWGROUP |grep distribution
        "ansible_distribution": "CentOS",
        "ansible_distribution_file_parsed": true,
        "ansible_distribution_file_path": "/etc/redhat-release",
        "ansible_distribution_file_variety": "RedHat",
        "ansible_distribution_major_version": "6",
        "ansible_distribution_release": "Final",
        "ansible_distribution_version": "6.9",
```

####Example of shell Module: Here checked kernel version, uptime and OS Release of a client machine

```
[ansible@ansible-controller ~]$ ansible NEWGROUP -m shell -a "uname -a;uptime;cat /etc/redhat-release"
cman1 | SUCCESS | rc=0 >>
Linux cman1 2.6.32-696.3.1.el6.x86_64 #1 SMP Tue May 30 19:52:55 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
 07:21:53 up  4:06,  3 users,  load average: 0.00, 0.00, 0.00
CentOS release 6.9 (Final)
```

Playbook in ansible: It is group of adhoc commands/tasks to be executed in scripted way.

#Example of a playbook: To install vsftpd pkg, start vsftpd service & make it persistent on Reboot

1. **First task is to install vsftpd pkg**
2. **Second task is to start the Service & put chkconfig on**

Pkg installation Playbook snapshot:

```
---
- hosts: NEWGROUP      ###### hosts group from ansbile inventory
  user: ansible        #### ansible admin user
  become: yes
  become_method: sudo   ####ansible user will use sudo to run admin commands (ansible ALL=NOPASSWD: ALL)
  tasks:                ####tasks going to be achved as below
  - name: 1. Deploy FTP server    ####first task
    yum: name=vsftpd state=present
  - name: 2. Start ftp service && make it enabled on reboot     ### second task
    service: name=vsftpd state=started enabled=yes
```

##Below o/p of running the above playbook

Here we could see changes made on Client on execution of both tasks and no task failed. Colour has significance like green colour no changes made and orange colour states changed made. From /etc/ansible/ansible.cfg we can customise the colours for o/p display

```
[ansible@ansible-controller ~]$ ansible-playbook pkg.playbook.yml

PLAY [NEWGROUP] ****************************************************************

TASK [Gathering Facts] *********************************************************
ok: [cman1]

TASK [1. Deploy FTP server] ****************************************************
changed: [cman1]

TASK [2. Start ftp service && make it enabled on reboot] ***********************
changed: [cman1]

PLAY RECAP *********************************************************************
cman1                      : ok=3    changed=2    unreachable=0    failed=0

[ansible@ansible-controller ~]$
```

####Patchng Playbook####

1. First task is to verify if apps process stopped

2. Second task is to skip patching if apps process running

3. Third task is to start patching if application is stopped && OS Distribution is CentOS or RHEL

4. Fourth task is for Server Reboot if kernel update has happened

5. Fifth task to restart the server if above condition is met

6. Sixth task is to take pause till Server comes up after reboot

7. Seventh Task is to confirm system is up and responding to ssh after Reboot

====snapshots of patching playbook==========

```
---
####playbook for patching on CentoS & Rhel Servers
#####
- hosts: NEWGROUP           #### Inventry group from ansible Server
  user: ansible             ### user have passwordless communication with clients
  become: yes               ###Above user can become ADMIN(root) on client machine
  become_method: sudo
####Starting with tasks in order to achieve patching on Servers listed as per invebtory Group on ansible Server
#######Here we are defining/registering some value/variables as condition for next task to be performed

  tasks:
#1. Task to Check whether Apps process running or not here we are taking apache/http as apps process

  - name: 1. Verify Apps process running status
    shell: if ps -eaf|egrep 'http|apache'|grep -v grep > /dev/null ;then echo 'process_running';else echo 'process_not_run
ning';fi
    ignore_errors: true
    register: app_process_check   #### Registered varibale for next Steps/tasks

#2. Task is for decision ,task will fail/quit, if apps process is running;if No proces running will be skipped to move on
next task

  - name: 2. Decision whether to start with patching skip Fail message if Apps stopped
    fail: msg="{{ inventory_hostname }} have running Application. Pls stop apps first, then start with patchung."
    when: app_process_check.stdout == "process_running"
```

```
#3. This task will upgrade/patching of rpm/pkgs if application is stopped && if OS Distribution is CentOS or RHEL

  - name: 3. Upgrade all packages on the server/Start Patching
    yum:
     name="kernel"
     state=latest
    when: app_process_check.stdout == "process_not_running" and ansible_distribution == 'CentOS' or ansible_distribution =
= 'Red Hat Enterprise Linux'
    register: yum_update

#4. This task is if kernel update has happened and system needs reboot or not

  - name: 4. Check if Server reboot required after Patching/kerenel update.
    shell: KERNEL_NEW=$(rpm -q --last kernel |head -1 | awk '{print $1}' | sed 's/kernel-//'); KERNEL_NOW=$(uname -r); if
[[ $KERNEL_NEW != $KERNEL_NOW ]]; then echo "reboot_server"; else  echo "reboot_not_needed"; fi
    ignore_errors: true
    register: reboot_required   ## Registered variable for Next Step/Task

#5. This task is to restart the server

  - name: 5. Restart system if Kernel is upgraded to boot with new Kernel
    command: shutdown -r +1 "Rebooting System After Patching"  #### delaying command for 1mins & display message as well
    async: 0
    poll: 0
    when: reboot_required.stdout == "reboot_server"
    register: server_rebooted        ##### Registered variable for next steps/tasks
    ignore_errors: true
```

```
#6.  This task is to wait for 5 mins for system to come up after Reboot; We can set time as per Server type Physical/Virtu
al

  - name: 6. Since Server is undergoing Reboot So pause for 300 secs
    pause:
      minutes: 5
    when: reboot_required.stdout == "reboot_server"
#7.  This task is to confirm, system is up and responding to ssh after Reboot

  - name: 7. Confirmation if system responding to ssh/Up
    wait_for:
        host={{ inventory_hostname }}
        port=22
        delay=15
        timeout=300
        state=started
    when: server_rebooted
```

####Starting with Patching Playbook####

1. **Run patching Playbook when apps Process are running: It will through error after first task**

   [root@cman1 ~]# ps -eaf|grep http
   root 3401 1 0 13:57 ? 00:00:00 /usr/sbin/httpd
   apache 3404 3401 0 13:57 ? 00:00:00 /usr/sbin/httpd
   apache 3405 3401 0 13:57 ? 00:00:00 /usr/sbin/httpd

```
[ansible@ansible-controller ~]$ ansible-playbook patching.yml

PLAY [NEWGROUP] ****************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [cman1]

TASK [1. Verify Apps process running status] ***********************************************************************
changed: [cman1]

TASK [2. Decision whether to start with patching skip Fail message if Apps stopped] ********************************
fatal: [cman1]: FAILED! => {"changed": false, "msg": "cman1 have running Application. Pls stop apps first, then start with
 patchung."}
        to retry, use: --limit @/home/ansible/patching.retry

PLAY RECAP *********************************************************************************************************
cman1                      : ok=2    changed=1    unreachable=0    failed=1
```

2. **Now Run patching playbook when apps are stopped: All tasks will be completed, wherever modifications are made o/p will be displayed as changed. "OK" will be displayed where no changes is made. Fail message in-case process running will be skipped here**

```
[root@cman1 ~]# service httpd stop
Stopping httpd:                                [  OK  ]
[root@cman1 ~]# ps -eaf|grep http
root      3662  3366  0 14:01 pts/0    00:00:00 grep http
```

```
[ansible@ansible-controller ~]$ ansible-playbook patching.yml

PLAY [NEWGROUP] ****************************************************************

TASK [Gathering Facts] *********************************************************
ok: [cman1]

TASK [1. Verify Apps process running status] ***********************************
changed: [cman1]

TASK [2. Decision whether to start with patching skip Fail message if Apps stopped] ****
skipping: [cman1]

TASK [3. Upgrade all packages on the server/Start Patching] ********************
changed: [cman1]

TASK [4. Check if Server reboot required after Patching/kerenel update.] *******
changed: [cman1]

TASK [5. Restart system if Kernel is upgraded to boot with new Kernel] *********
changed: [cman1]

TASK [6. Since Server is undergoing Reboot So pause for 300 secs] **************
Pausing for 300 seconds
(ctrl+C then 'C' = continue early, ctrl+C then 'A' = abort)
ok: [cman1]

TASK [7. Confirmation if system responding to ssh/Up] **************************
ok: [cman1]

PLAY RECAP *********************************************************************
cman1                      : ok=7    changed=4    unreachable=0    failed=0
```

Here you could see No task failed and and 4 tasks made changes with their action

Below on Client Server we could see Server going down after Patching and came up with upgraded Kernel Version

```
[root@cman1 ~]# uname -a
Linux cman1 2.6.32-696.3.1.el6.x86_64 #1 SMP Tue May 30 19:52:55 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
[root@cman1 ~]#
Broadcast message from ansible@cman1
        (unknown) at 13:09 ...

The system is going down for reboot in 1 minute!
Rebooting System After Patching

Broadcast message from ansible@cman1
        (unknown) at 13:10 ...

The system is going down for reboot NOW!
Rebooting System After Patching
login as: root
root@192.168.121.132's password:
Last login: Thu Aug  2 13:03:28 2018 from 192.168.121.1
[root@cman1 ~]# uname -a
Linux cman1 2.6.32-754.2.1.el6.x86_64 #1 SMP Fri Jul 13 12:50:12 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
```

####Next starting patching playbook again on recent patched Server###

Since no Kernel/package will be upgraded Task of Server Reboot is skipped

```
[ansible@ansible-controller ~]$ ansible-playbook patching.yml

PLAY [NEWGROUP] ****************************************************************

TASK [Gathering Facts] *********************************************************
ok: [cman1]

TASK [1. Verify Apps process running status] ***********************************
changed: [cman1]

TASK [2. Decision whether to start with patching skip Fail message if Apps stopped] *******
skipping: [cman1]

TASK [3. Upgrade all packages on the server/Start Patching] ********************
ok: [cman1]

TASK [4. Check if Server reboot required after Patching/kerenel update.] *****************
changed: [cman1]

TASK [5. Restart system if Kernel is upgraded to boot with new Kernel] *********************
skipping: [cman1]

TASK [6. Since Server is undergoing Reboot So pause for 300 secs] **************************
skipping: [cman1]

TASK [7. Confirmation if system responding to ssh/Up] **************************************
ok: [cman1]

PLAY RECAP *********************************************************************
cman1                      : ok=5    changed=2    unreachable=0    failed=0
```

Devinder Singh Lalotra