RH294

DATE: 30 Oct.
PAGE NO.:

SESSION: 1 ANSIBLE

- Used for Automation + Intelligence
- Python, Perl are imperative languages so we have to tell them: what to do & how to do.
- Intelligence: when the version | software/os changes some of the commands may change as Ansible is intelligent to change it accordingly.

Os.	RHEL 4 & 6	RHEL 7 & 8
Install	yum	yum/dnf
services	service	systemctl

- In Ansible we only have to tell: what to do. These are known as declarative languages

DATE: 4/NOV
PAGE NO.:

SESSION: 2 ANSIBLE

- Configuration management (CM)
- We do not have to install agents in the target computers. or It is agentless.

pip3 install ansible

- database of ip addresses in ansible is known as an Inventory.

ansible --version
ansible all --list-hosts
  ↳ lists the inventory
vim ip.txt
  192.168.0.199

- First we have to create a conf file.

mkdir /etc/ansible
vim /etc/ansible/ansible.cfg
  [defaults]
  inventory = /root/ip.txt
# ansible all httpd start
# ansible all -m service name=httpd state=start
  -> # name=httpd state=running
yum install sshpass

DATE: 5 NOV.
PAGE NO.:

SESSION: 3 ANSIBLE

cat /etc/ansible/ansible.cfg

inventory (ip.txt)
192.168.0.162 ansible_user=root
ansible_ssh_pass=redhat
ansible-connection=ssh

ansible all --list-hosts
ansible all -m ping
↳ checks the connectivity with target.

- Ansible standardize (abstract) all the cmd of different OSs into its packages/modules.
- RAL (Resource Abstraction Layer)
- ansible package module (URL)

ansible all -m package -a "name=httpd state=present"
↳ Installs httpd
(name = argument, present = Installs)

- If the package is already installed it won't run the cmd

ansible all -m service -a "name=httpd state="started"
(service = ex. systemctl in RHEL) OR module name
started = Starts

DATE : / /
PAGE NO. :

ansible all -m copy -a " src=web.html dest=/var/www/html"

vim /etc/ansible/ansible.cfg
↳ ~~host_key_checking = false~~

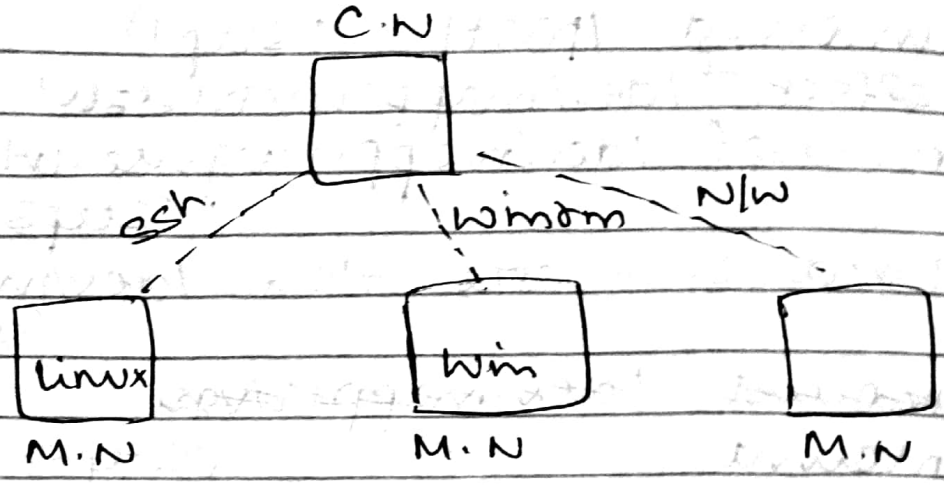
[defaults]
inventory = /root/ip.txt
host_key_checking = false

mkdir /cd /home/student/deploy-inventory

ansible <group> -i inventory --list-hosts
↳ we can use our custom inventory and do not have to update it in .cfg file.

DATE: 24 NOV
PAGE NO.:

SESSION: 4
SESSION: 3 ANSIBLE



C.N - Controller Node
M.N - Managed Node

ansible all -m ping
ansible all --list-hosts
ansible all -m package -a "name=
(module, argument)
httpd state=present".

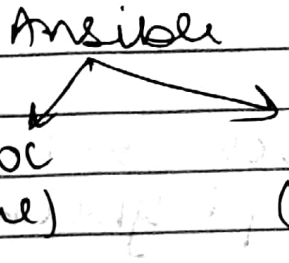
gedit web.yml
ansible all ——→
ansible all -m ——→
——————→

- Correct way for playbook.

key → hosts: all ← value
tasks:
- package: "name ____"
- copy: "src = ___ dest = ___"
- service: "name ____".

DATE: / /
PAGE NO.:

- We use json/yml format to ~~write~~ write the scripts of tools in key-value format.



- key value format

```
name : "nimal"
phone: "1111"
email :
  - "v@lw.com"
  - "j@lw.com"
  - "p@lw.com"
address: "LW"
```

Indentation should be same

- yml starts with 3 hyphen (---) & ends with 1 (-)

```
# For certain hosts
- hosts : 192.168.0.9
  tasks:
    - package: ____
    - copy : ____
- hosts : 1.2.3.4 , 192.168.129.4
  tasks :
    - package : ____
    - service : ____
```

DATE : / /
PAGE NO. :

- yml is indent sensitive and we have to give one space after the hyphen. lists should be indented accordingly.

ansible-playbook web.yml
↳ runs the playbook.

vim web.yml
```
-hosts: all
 tasks:
 -package:
      name: "httpd" ← string in quotes
      state: present ← key word
 - copy:                without quotes
      src: "____"
      dest: "____"
 - service:
      name: "httpd"
      state: started
```

ansible-playbook --syntax-check web.yml
↳ checks the syntax.
↳ advisable to run it before running the code.

DATE : / /
PAGE NO. :

TASK : i) configure appache web server with playbook.

ii) Playbook for docker configuration

St 1. install docker

St 2. service docker

St 3. docker download image

St 4. launch the os (containers)

DATE 25 Nov
PAGE NO.:

SESSION: 38 ANSIBLE

- To use ansible to install a package, yum should be pre-configured.

#

```
- hosts: all
  tasks:
  - file:
      state: directory
      path: "/dvd1"
```
} Creates a dir

↑ dir_name

```
  - mount:
      src: "/dev/cdrom" or "/dev/sr0"
      path: "/dvd1"
      state: mounted
      fstype: iso9660
```
{ format type of iso 9660

```
  - yum_repository:
      baseurl: "/dvd1/AppStream"
      name: "mydvd1"
      description: "my yum dvd1 for package"
  - yum_repository: → gpgcheck: no
      baseurl: "/dvd1/BaseOS"
      name: "mydvd2"
      description: "                    ".
  - package: → gpgcheck: no
      name = "httpd"
      state = present
  - ~~service~~
  - copy:
      dest: "/var/www/html/index.html"
      content: "this is my website"
```

DATE : / /
PAGE NO. :

- service :
  name : "httpd"
  state : started
  enabled : yes
+ firewalld :
  port : "80/tcp"
  state : enabled
  permanent : yes
  immediate : yes

firewall-cmd --list-port
↳ shows the ports ~~disabled~~ enabled

DATE: 26 NOV
PAGE NO.:

SESSION: 39 ANSIBLE

ansible all -m ping

cd /etc/httpd/conf.d/
↳ document root of apache web server

vim lwl.conf → create a new file
Listen 8080 — new port — ip of web server
<VirtualHost 192.168.0.174:8080>
DocumentRoot /var/www/lw — changes the doc root.
</VirtualHost>

copy module
↳ src
↳ content

```
-copy:
  dest: "____"
  src: "lwl.conf"
```

#

```
-file:
  state: directory
  path: "/var/www/lw"
```

- we have to restart the services after changing the conf file to see the changes.

```
-service:
  state: restarted
```

DATE : / /
PAGE NO. :

- change the firewall for port 8080.

#

```
- hosts : all
  d : /var/www/xxx  ← variable
  tasks :
    -
```

- debug ← prints in ansible

```
- debug :
    msg : "hi hello"
```

#

```
- hosts : all
  vars :                          } keyword to initialize
    - x : "hello"                   variables
    - y : 5
    - z : 78
  tasks :
    - package :
    - debug :                     ← variable
        msg : & " {{ n }} "
```

DATE: / /
PAGE NO.:

prompts → It will always ask for the value for the variable

- hosts: all
  vars_prompt:
    - name: dns_die
      private: no
      prompt: "enter dns name pls"

DATE: 27 NOV
PAGE NO.:

SESSION: 40 ANSIBLE

- It is compulsory to make the ip the part of inventory. otherwise, it won't run.

- ansible_facts is the module that will gather all the info abt. the OS and will tell other modules.

```
- debug:
    msg: "{{ ansible_facts }}"
```

```
msg: "{{ ansible_facts['system'] }}"
```
↳ It will only print the system (Linux)

- setup is a module; it will go to a certain ip and retrive all the facts abt it and store it in a variable called ansible_facts.

```
- setup:
    filter: "ansible_architecture"
```

- In copy module if we want to upload multiple lines in content we can use (greaterthan symbol)

```
>
" abcd efg \n
hijk "
```

DATE : / /
PAGE NO. :

- template is exactly like the copy module but the only difference is that template module will go inside the src file and look for variables and replace its value.

```
- template:
    dest: "____"
    src: "____"
```

pipe

```
# ansible 192.168.0.165 -m setup | less
```

- In newer version of ansible we do not have to write "ansible_" before fact.
- {{ ansible_facts["dynamic_ipv4"]["address"] }}

```
# task:
  - name: "doc name"
    file:
      ____
      ____
      ____

  - name: "Mt dir"
    file:
      ____
      ____
```

DATE : / /
PAGE NO. :

vars_files:
- http_var.yml
  ↳ creates a seperate yml file for variables.

http_var.yml
dvd_dir: "/dvd5"
doc_root: "/var/www/lw123"
http_port: 8080

TASK: Playbook for Hadoop using ansible.

ANSIBLE

vim /etc/httpd/conf/httpd.conf
↳ config file of httpd

AllowOverride AuthConfig
*. line 154

htpasswd -c /etc/www.passwd jinal
↳ create users

cat /etc/www.passwd

htpasswd /etc/www.passwd jack
↳ create 2nd user in www.passwd.

cd /var/www/html/
vim .htaccess
authtype basic → authname "lw website"
authuserfile /etc/www.passwd
require valid-user

DATE: / /
PAGE NO.:

```
- hosts : all
  tasks:
   - package:
       name : "httpd"
       state : present
   - copy :
       dest : /var/www/html/index.html"
       content : " Secure page !! "
   - replace :
       path : "/etc/httpd/conf/httpd.conf"
       regexp "AllowOverride None"
       replace : "AllowOverride AuthConfig"
     - copy :
       dest : "/var/www/html/.htaccess"
       src : ".htaccess"
```

ansible-doc -l
↳ lists all the module available

ansible-doc -l | grep package
ansible-doc package
↳ shows the doc for package
ansible-doc -l | grep htpasswd

ansible-doc htpasswd

```
- package :
    name : "python36"
- pip :
    name : "passlib"
```

↑ If pip doesn't work

DATE: / /
PAGE NO.:

```
-htpasswd:
    path: "/etc/www.passwd"
    name: "jack"
    password: "redhat"
-service:
    name: "httpd"
    state: ~~restarted~~ restarted
```

- htpasswd module will be failed. To resolve this error we need to solve its dependency.

pip3 install passlib
↳ Install this in controller node & managed node

```
-yum:
    name:
```

ANSIBLE

- Reverse Proxy: They recreate the request and send them to the server. This also create a security. i.e The client will never ~~the~~ hit the server only the proxy server will and the client won't know the ip of server.

- Reverse Proxy also provides the load balancer service. So that it will divide the request among all the servers in order to prevent the overloading of servers.

- HAproxy is a software / package to configure a reverse proxy.

DATE : / /
PAGE NO. :

yum install haproxy

vim /etc/haproxy/haproxy.cfg

68 is. bind *:8080
87,88. ~~bact~~ appl 172.168.0.120:80 check
app2 172.168.0.155:80 check

systemctl start haproxy

journalctl -xe
↳ If the haproxy and doesn't starts

- hosts: myweb
  tasks:
  - package:
      name: "httpd"

- hosts: myweb
  tasks:
  - package:
      name: "haproxy"

xOx

ANSIBLE

- make haproxy.cfg as template file

- hosts : ~~all~~ ip
  tasks :
  - yum_repo for docker
  - package:
    name : "docker-ce" } FAILED
    state : present

ansible all -m command -a date
↳ we can run shell cmds.

- command : "yum install docker-ce --nobest -y"

DATE : / /
PAGE NO. :

ANSIBLE

- {{ }}
  ↳ variable
- {% %}
  ↳ Loops
- {# #}
  ↳ comment

- we can only do this in template module

```
# vars:
  -db:
    -"nimal"
    -"pop"
    -"harry"
```

It will print it as a list

```
#
  {% for i in db %}
    this is {{ i }}
  {% endfor %}
```

They will run 3 times

```
#. {% endif %}
```

```
# {% for i in groups['myweb'] %}
    this is ip {{ i }}
  {% endfor %}
```

↑ lists all the ip in ip##

DATE : / /
PAGE NO. :

± vim haproxy.cfg

{%- for i in groups['myweb'] %}
server app{{i}}:80 check
{%- endfor %}

↓ we use counter
app{{ loop.index }}
↓
Internal variable

DATE : 9/ Dec
PAGE NO. :

SESSION : 50 ANSIBLE

- Module for ec2.aws
  - ec2_instance

pip3 install boto3
  - SDK: software development

aws_access_key: "username"
  aws_secret_key: "password"

- services → IAM → users
  - SA → AdministratorAccess
  - Create user if not created.
  - TO create accesskey and secret key.

- In ansible, localhost group is by default created for controller node.
- Inside the module the developer has written the URL of the API

TASK: (i) ec2 instance provisioning
(ii) Config it for webserver.

DATE: / /
PAGE NO.:

ANSIBLE

- In ansible, instead of if-else, switch we have when keyword

```
- hosts: myweb
  tasks:
  - package:
      name: "httpd"
    when: x == "redhat"
```

vars:
  - x : "redhat"

- os_name "{{ ansible_facts ['distribution'] }}"

when: os_name == "redhat"
(skipped)
↳ case sensitive

when: os_name == "RedHat"

- debug:
    var: ans _____ .dist or os_name

- Dif. bet var and msg in debug module is that msg only prints string whereas var prints the variable.

- In ad-hoc, they will print the output but in playbook they don't print it. We can use verbose (-v) option to see the o/p

DATE: / /
PAGE NO.:

#. tasks:

```
- command: date
  register : x
- debug:
    msg : "hi test"
  when: x.rc != 0
- debug:
    var: x.rc
```

↳ returned code

```
- service:
    name: "httpd"
    state : started
  register : y
- debug:
    var : y
```

- and → logical AND
  or → logical OR
- when only requires boolean values.

DATE : 11/ Dec
PAGE NO. :

Session : 52 ANSIBLE

TASK : docker.yml
- pull image.
- launch the centos image. using register option
- configure web server.

- ~~By default a task fails~~

- By a task fails, by default ansible terminates the playbook.

ignore_errors: yes
↳ If a task fails and we want to run the playbook for further tasks. we use this option.

2nd
```
tasks:
handlers: - name: "webservices"
  service:
    name: "httpd"
```

1st
```
- template:
    dest:
    src:
  notify: webservices
```

- we can create dif. var-files for different versions / OS.

DATE : 17 Dec
PAGE NO. :

SESSION: 55 ANSIBLE

- Exception handling:
  block - rescue are the keywords instead of try - except.

- In ignore-error if an exception arises we skip the task and run the next task.

- uri is a module that helps download images, videos using playbook.

- get_url module downloads file from http, https ~~servers~~.

```
- get_url:
    url: "http://<<—>>.jpg"
    dest: "/root/photo.jpg"
```

↳ if it fails for some reason.

```
task:
  ~~task~~:
- ~~block~~:
  ~~block~~:
  - package:
      ——
      ——
  - get_url:
      —
      —
      —
```

DATE: / /
PAGE NO.:

```
rescue:
  - debug:
      msg: "for rescue"
```

- If a task fails then instead of terminating the playbook, ansible runs the rescue tasks.
- always is the keyword in which the block of code always runs whether an exception occurs or not.

```
always:
  - debug:
      msg: " Thanks, bye "
```

- We use hyphen (-) only before the block keyword not with rescue & always.
- We can also use ansible to mail. by using mail module.

ARTIFICIAL

SESSION: 56 ML

ANSIBLE

DATE: 17 Dec
PAGE NO.:

- Feature Elimination
- parameter = (b, w) [bias, weight]
- string → Encoder → Number
  (state/Gender/Branch)

```
# from sklearn.preprocessing import
                        LabelEncoder
~~LabelEncoder()~~
state_le = LabelEncoder()
state = state_le.fit_transform
```

ANSIBLE

- command module do not have idempotance by default.

```
- command : "date"
  changed_when : false
```

↳ Now it will make the O/P in green { OK = 1 }

```
- command : "ls /lw3"
  register : x
  ignore_error : true
- command : "mkdir /lw3"
  when : "x.rc != 0"
```

DATE : / /
PAGE NO. :

- If we don't want a code of block to run
  - # when: false
- shell module features all the bash commands.
- shell is comparatively slower than the command module.

- In mail module we use

```
- hosts: 127.0.0.1
  tasks:
    - name: "mail"
      mail:
        host: ——
        port: 587
        username: ——@gmail.com
        password:
        to: ——@gmail.com
        subject: ——
        body: ——
```

- gmail secure app (on web)
  Allow less secure apps: ON

ARTICLE

DATE : / /
PAGE NO. :

ansible-vault -h
↳ encrypt

ansible-vault encrypt mysecret.yml
New vault password: ——— mys.yml
Confirm : ———

ansible-vault encrypt mysecret.yml
--output mys.yml

ansible-vault create mys.yml
↳ creates an encrypted file.

ansible-vault view mys.yml
↳ we can open it using this cmd.
Here, cat & vim doesn't work.

ansible-playbook --ask-vault-password
m.yml
↳ it will ask the password (vault) so that it can retrive its content.

DATE: 18 Dec 20
PAGE NO.:

SESSION: 57 ML ANSIBLE

# from sklearn.preprocessing OneHotEncoder
state_ohe = OneHotEncoder()
state_ohe.fit_transform (state)
↳ Failed 2D required.

{{ x = x.reshape(-1,1) }}
state = state.reshape(-1,1)

ANSIBLE

```
-package:
    name : {{item}}
    state : present
  loop:
    - "httpd"
    - "php"
```

or

we can create a variable

```
-vars:
  - x :
      - "httpd"
      - "php"
      - "abc"

loop: {{ x }}
name: "{{ item }}"
```

ARTICLE

DATE : / /
PAGE NO. :

- We can create & manage the users by using user module.

- user :
  name : "____"
  password : "____"
  state : present.
  groups : "____"
  wsp : usg.

vars :
- user :
  - "vimal", 1111
  - "jack", 222

⇓

vars :
- user :
  - "vimal"
    1111
  - "jack"
    2222

⇓

- user :
  - name : "vimal"
    phone : 1111
  - name : "jack"
    phone : 2222

DATE : / /
PAGE NO. :

- debug :
  var: user[2].phone

- user :
  name: "{{ item.name }}"
  password: "{{ item.phone }}"
loop: "{{user}}"

↓

'item.p | password_hash('sha512')

ANSIBLE:

- Packages in Python is similar to Role in Ansible
  - when the playbooks are big ~~lengthy~~ then we manage it.

ansible-galaxy -h
  - gives all the roles.

ansible-galaxy role init apache
  - creates a dir/role.

ansible-galaxy role list --roles-path /ws20

roles:
- role: "myapache"
- role: "akg".

DATE : / /
PAGE NO. :

ARTICLE

ansible-playbook setup.yml --role-path /ws20

vim /etc/ansible/ansible.cfg

roles_path = /ws20

(here the roles folder/dir)

cd myapache/
ls
(every task has dif. file)

cd tasks/
vim main.yml

- package:
  name : "httpd"
  state : present

cd vars/
P : "——"

⇒ Create a role for haproxy, apache.

ANSIBLE.

nim /etc/sudoers

vimal ALL=(ALL) ~~jtsm~~ ALL

ansible 192.168.0.155 -m command -a id

↳ Error, if login with user
↳ { ansible_user = vimal }

DATE : / /
PAGE NO. :

ansible 192.168.0.155 -m command -a id --become --become-user root
↳ { ansible_user = root }
↳ Failed !! Missing sudo password.

-- become-method sudo --ask-become-pass

vim /etc/ansible/ansible.cfg

[privilege_escalation]
become = true
become_method = sudo
become_user = root
become_ask_pass = false

↳ Now we don't have to write the syntax for become in adhoc.

vim /etc/sudoers
vimal ALL=(ALL) NOPASSWD: ALL
↳ NO it won't ask the passwd.

ssh-keygen
↳ creates a public key & private key.

ssh vimal@192.168.0.155

DATE: / /
PAGE NO.:

FIG 7

ssh-copy-id vimal@192.168.0.15
↳ uploads the key.

cat /root/.ssh/id_rsa
↳ location where private key is stored.

* C:/Users/vimal9.20Dage/Downloads/2h294-8.0-student-guide.pdf

DATE: 30 Dec
PAGE NO.:

SESSION:63 ANSIBLE

mkdir mydb
cat a.~~txt~~ → don't create any ~~ext.~~ extension.
1.1.1.1 or .txt.
cat b.txt ↳ supports: .json, .yml, .py
2.2.2.2

# ansible.cfg
inventory = /mydb ← dir

vim ip.py

#! /usr/bin/python3

* URL for Dynamic Inventory
# wget URL ⇒ .py, .ini
# chmod +x hosts.py
# python3 hosts.py ~~--list~~ --list

PAGE NO.:
DATE: 4/Jan/2

SESSION 05 ANSIBLE

ansible-galaxy list
↳ lists all the roles.
- If we want we can download roles from galaxy.

yum install rhel-system-roles
ansible-galaxy list
↳ No new role come up
Bec. role path is not set.
cd /usr/share/doc/rhel-system-roles/
ls ↳ They download the roles here.

cd /usr/share/ansible/roles
vim /etc/ansible/ansible.cfg
↳ Comment role path
#roles_path = ——

cd timesync
vim README.md
cd timesync

- Time server uses NTP protocol is used to sync time to all the clients/OS.

- To use a role we use role module.
roles:
- ——— role ———

mkdir group_vars/
↳ vars for roles
mkdir <<group-name>>

PAGE NO. :
DATE : / /

- Now we do not have to include var-files. We only need to create a dir (group_vars) then a file with the name similar to group name. Inside it we can give any file name with extension .yml

ansible-doc lineinfile

```
# - lineinfile :
      path : "____"
      regexp: "Listen 80"
      line : "Listen 81"

  → regexp: "[Ll]isten 80"
              └ wild card / condition
```

- lineinfile module can be used in hadoop cluster (hdfs-site, core-site)
- replace module can also be used in the place of lineinfile. But the diff is lineinfile replace the whole line & replace will only replace one block.

PAGE NO. :
DATE : / /

* Process for certificate (Training)
  - For RH294
    - Already received (check mail)
  - After exams. will receive certificate.
    - Applicable for 3yrs.
  - After all the tasks.
    - ~~test~~ and reviewed
      - receive the certificates.