Setup For Ansible in RHEL 8

1. Epel repository Configure
   sudodnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
2. yum install ansible -y
   a. Here we are installing ansible in System 1.
3. List all ansible Hosts
   ansible all --list-hosts

```
[WARNING]: provided hosts list is empty, only localhost is available. Not
e that
the implicit localhost does not match 'all'
  hosts (0):
```

4. Ansible Inventory File
   a. vi /etc/ansible/hosts

i. Here type the IP Address of Management Node (Sys 2)

```
#Ip Address of Managed Nodes
192.168.29.15
```

   ii. If you have more IP then Type in this file.

5. Now we can see we have 1 host in this Inventory File.
   So, Our Inventory has been configured.

```
[root@localhost ~]# ansible all --list-hosts
  hosts (1):
    192.168.29.15
```

6. Now From Controller Node I want to install Httpd package in my Management node –
   a. For this we need to use some keyword and ansible have different keyword available for every thing that is known as module
      Syntax: ansible all -m package “name=pkg-name state=install”
      ansible all -m package “name=httpd state=install”

If You got this Error

```
[root@localhost ~]# ansible all -m package -a "name=httpd state=installed"
192.168.29.15 | FAILED! => {
    "msg": "Using a SSH password instead of a key is not possible because
Host Key checking is enabled and sshpass does not support this.  Please ad
d this host's fingerprint to your known_hosts file to manage this host."
}
```

Then Go to this file- /etc/ansible/ansible.cfg

```
[root@localhost ~]# vim /etc/ansible/ansible.cfg
[root@localhost ~]#
```

And you will find #host_key_checking = False is commented in line 7. Just Uncomment this line

```
host_key_checking = False
```

Or

If You got another error like this

```
[root@localhost ~]# ansible all -m package -a "name=firefox state=present"
root@192.168.198.229 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: Permission denied (publickey,
gssapi-keyex,gssapi-with-mic,password).",
    "unreachable": true
}
```

Then First try to check the connectivity using ping command. If it is not connected then solve the issue first. Or if you are getting this error. Then try to ping the OS by using this command: **ansible all -m ping -u user --ask-pass**

```
[root@localhost ~]# ansible all -m ping -u user --ask-pass
SSH password:
root@192.168.198.229 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
```

If it is connected then it will return pong as a message.

But still our problem is not solved yet. So, if we want to run the any command so every time the ask the password. So, better to update this thing inside the **/etc/ansible/hosts** file

Syntax: IP-ADDR ansible_ssh_pass=XXXXX ansible_ssh_user=USER-Name

After that when you run this command then you will get another error. And here -m is used for module & -a is used for argument.

➢ **ansible all -m package -a “name=httpd state=install”**

```
[root@localhost ~]# ansible all -m package -a "name=firefox state=install
"
192.168.29.7 | FAILED! => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "msg": "value of state must be one of: absent, installed, latest, pre
sent, removed, got: install"
}
```

So for this you need to write

➢ **ansible all -m package -a “name=httpd state=installed”**

```
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
```

Now this will installed the software and you can verify it by one more time by running the same command. This time they show “**Nothing to do**”

```
[root@localhost ~]# ansible all -m package -a "name=firefox state=install
ed"
192.168.29.7 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "msg": "Nothing to do",
    "rc": 0,
    "results": []
}
```

The Best part of the ansible when we got any error it shows in : red color.
And when some part is changed then it will show in : yellow Color.
And when nothing is changed then it will show in : green color.

For removing the software, we use:

➢ **ansible all -m package -a "name=firefox state=absent"**

```
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "msg": "",
    "rc": 0,
    "results": [
        "Removed: firefox-78.9.0-1.el8_3.x86_64"
    ]
}
```

Webserver Setup by Ansible

For this we need to install a software "**httpd**" through which we can deploy our webpage to real world.

- **ansible all -m package -a "name=httpd state=present"**

```
[root@localhost ~]# ansible all -m package -a "name=httpd state=present"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "msg": "",
    "rc": 0,
    "results": [
        "Installed: mod_http2-1.15.7-3.module+el8.4.0+8625+d397f3da.x86_6
4",
        "Installed: redhat-logos-httpd-84.4-1.el8.noarch",
        "Installed: httpd-2.4.37-39.module+el8.4.0+9658+b87b2deb.x86_64",
        "Installed: httpd-filesystem-2.4.37-39.module+el8.4.0+9658+b87b2d
eb.noarch",
        "Installed: httpd-tools-2.4.37-39.module+el8.4.0+9658+b87b2deb.x8
6_64",
        "Installed: apr-1.6.3-11.el8.x86_64",
        "Installed: apr-util-1.6.1-6.el8.x86_64",
        "Installed: apr-util-bdb-1.6.1-6.el8.x86_64",
        "Installed: apr-util-openssl-1.6.1-6.el8.x86_64",
        "Installed: mailcap-2.1.48-3.el8.noarch"
    ]
}
```

And after installing the software we need to start the webserver services which is not started we can verify it by : "**systemctl status httpd"**

```
[root@localhost /]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd.service(8)
```

And here we need to start the service for this we use "**service**" module

- **ansible all -m service -a "name=httpd state=started"**

```
[root@localhost ~]# ansible all -m service -a "name=httpd state=started"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "name": "httpd",
    "state": "started",
```

We can also verify the service is started by **"systemctl status httpd"**

```
[root@localhost /]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2022-09-19 20:28:48 IST; 2min 33s ago
     Docs: man:httpd.service(8)
```

Now we need to deploy our webpages in the managed node & in managed node the webpages are not available. So, for this we transfer our webpage from our master node to managed node.

Here I created a web page in my master node. With the name of Index.html and now I am going to deploy this web page to my slave node which is also my managed node

```
[root@localhost ~]# ifconfig enp0s3
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.29.126  netmask 255.255.255.0  broadcast 192.168.29.
255
        inet6 fe80::a00:27ff:fec5:f42f  prefixlen 64  scopeid 0x20<link>
        inet6 2405:201:a404:d83c:a00:27ff:fec5:f42f  prefixlen 64  scopei
d 0x0<global>
        ether 08:00:27:c5:f4:2f  txqueuelen 1000  (Ethernet)
        RX packets 30539  bytes 6975674 (6.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5241  bytes 2732793 (2.6 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ls
AWS_CSA_Training.pem  Epel_Config.txt   Menu
C++                   LiME              MERN
DockerFiles           Machine_Learning  Privatekey.ppk
[root@localhost ~]# cat >> Index.html
Hello World!
```

And for this we need "**copy"** module and we transfer this file from our master node which is source to managed node which is our destination.

- **ansible all -m copy -a "dest=/var/www/html src=/root/Index.html"**

```
[root@localhost ~]# ansible all -m copy -a "dest=/var/www/html/ src=/root/Index.html"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "checksum": "a0b65939670bc2c010f4d5d6a0b3e4e4590fb92b",
    "dest": "/var/www/html/Index.html",
    "gid": 0,
    "group": "root",
    "md5sum": "8ddd8be4b179a529afa5f2ffae4b9858",
    "mode": "0644",
    "owner": "root",
    "secontext": "system_u:object_r:httpd_sys_content_t:s0",
    "size": 13,
    "src": "/root/.ansible/tmp/ansible-tmp-1663602112.2490346-6389-41146967597633/source",
    "state": "file",
    "uid": 0
}
```

we can also verify this in our managed node

```
[root@localhost html]# ls
Index.html
[root@localhost html]# cat Index.html
Hello World!
[root@localhost html]#
```

We can see the file is transferred but we are not able to see this webpage by using curl command.

```
[root@localhost ~]# curl 192.168.29.7:80/Index.html
curl: (7) Failed to connect to 192.168.29.7 port 80: No route to host
```

So, for solving this issue we need to write some firewall rules which help us to solve this challenge and for setting up the rules of firewall we need to import the firewall module which is "**firewalld"**

- **ansible all -m firewalld -a "immediate=yes port=80/tcp state=enabled"**

```
[root@localhost ~]# ansible all -m firewalld -a "immediate=yes port=80/tcp state=enabled"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "msg": "Non-permanent operation, Changed port 80/tcp to enabled"
}
```

The rule is set. Now we can see our webpage by the curl command.

```
[root@localhost ~]# curl 192.168.29.7:80/Index.html
Hello World!
```

The one thing for the apache webserver is that, when we reboot the OS then our services is stopped.

```
[root@localhost html]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; [redacted]; vendor preset: disabled)
   Active: active (running) since Mon 2022-09-19 20:28:48 IST; 1h 1min ago
     Docs: man:httpd.service(8)
 Main PID: 12100 (httpd)
```

```
[root@localhost ~]# ansible all -m service -a "name=httpd state=started enabled=yes"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "enabled": true,
    "name": "httpd",
    "state": "started",
```

So for make it permanent we need to enable it.

```
[root@localhost html]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; [redacted]; vendor preset: disabled)
   Active: active (running) since Mon 2022-09-19 20:28:48 IST; 1h 10min ago
```

- **ansible all -m firewalld -a "permanent=yes immediate=yes port=80/tcp state=enabled"**

```
[root@localhost ~]# ansible all -m firewalld -a "permanent=yes immediate=yes port=80/tcp state=enabled"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "msg": "Permanent and Non-Permanent(immediate) operation, Changed port 80/tcp to enabled"
}
```

Now we clean the environment.

First, we removing the Http web server and for this we use the following command.

- **ansible all -m package -a "name=httpd state=absent"**

```
[root@localhost ~]# ansible all -m package -a "name=httpd state=absent"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "msg": "",
    "rc": 0,
    "results": [
        "Removed: mod_http2-1.15.7-3.module+el8.4.0+8625+d397f3da.x86_64",
        "Removed: httpd-2.4.37-39.module+el8.4.0+9658+b87b2deb.x86_64"
    ]
}
```

And removing the Index.html file by our ansible command.

- **ansible all -m file -a "name=/var/www/html/Index.html state=absent"**

```
[root@localhost ~]# ansible all -m file -a "name=/var/www/html/Index.html state=absent"
192.168.29.7 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "path": "/var/www/html/Index.html",
    "state": "absent"
}
```

Webserver Setup by Ansible playbook

We use ansible to setup multiple OS in one go. And there we don't use command one by one to setup the OS and for this we use a concept of playbook. We create a playbook through which we can setup the entire OS by just a single playbook.

For this we need to write the playbook in a language that is "**YML (**yet another markup language**)**". So, create a file with any name with extension of "**.yml**"

- **vim webserver.yml**

```
- hosts: all
  tasks:
  - package: "name=httpd state=present"
  - copy: "dest=/var/www/html src=/root/Index.html"
  - service: "name=httpd state=started enabled=yes"
  - firewalld: "immediate=yes port=80/tcp state=enabled permanent=yes"
```

Write the script in the file and save this file. To run this file as a playbook we use command:

- **ansible-playbook webserver.yml**

```
[root@localhost Ansible]# ansible-playbook 01_web_setup.yml

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.29.7]

TASK [package] *****************************************************************
changed: [192.168.29.7]

TASK [copy] ********************************************************************
ok: [192.168.29.7]

TASK [service] *****************************************************************
changed: [192.168.29.7]

TASK [firewalld] ***************************************************************
ok: [192.168.29.7]

PLAY RECAP *********************************************************************
192.168.29.7               : ok=5    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

But we don't write the playbook in this way.

The Correct way to write the playbook is this: and then by this way we can run it ⬇

```
- hosts: all
  tasks:
  - name: "Installing Webserver"
    package:
      name: "httpd"
      state: "present"
  - copy:
      dest: "/var/www/html"
      src: "/root/Index.html"
  - service:
      name: "httpd"
      state: "started"
      enabled: "yes"
  - firewalld:
      immediate: "yes"
      port: "80/tcp"
      state: "enabled"
      permanent: "yes"
```

```
[root@localhost Ansible]# ansible-playbook 01_web_setup.yml

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

TASK [Gathering Facts] *********************************************************
**********
ok: [192.168.29.7]

TASK [Installing Webserver] ****************************************************
**********
ok: [192.168.29.7]

TASK [copy] ********************************************************************
**********
ok: [192.168.29.7]

TASK [service] *****************************************************************
**********
ok: [192.168.29.7]

TASK [firewalld] ***************************************************************
**********
ok: [192.168.29.7]

PLAY RECAP *********************************************************************
**********
192.168.29.7               : ok=5    changed=0    unreachable=0    fa
iled=0    skipped=0    rescued=0    ignored=0
```