

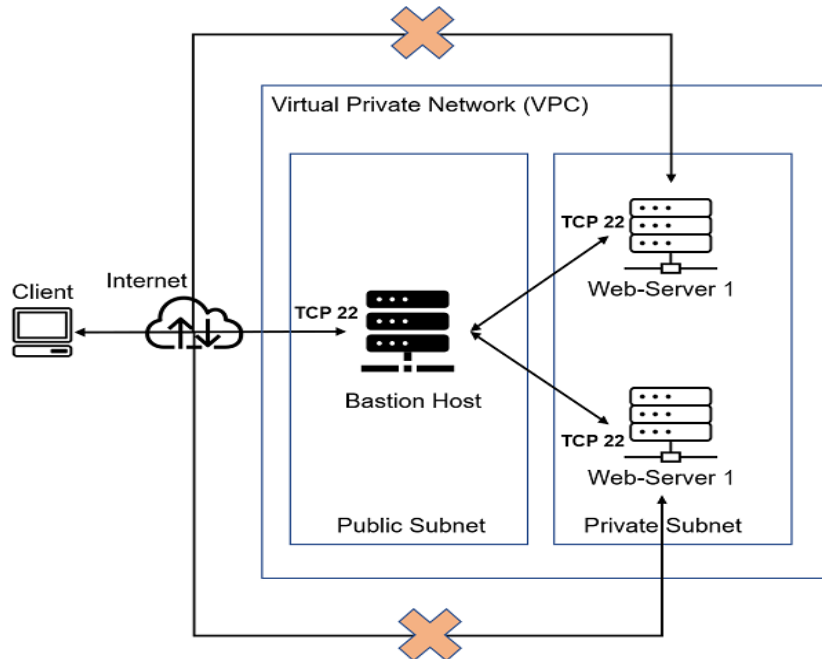
PROJECT 11: ANSIBLE CONFIGURATION MANAGEMENT - AUTOMATE PROJECT 7 - 10

Projects 7 to 10 required a lot of manual operations to set up virtual servers, install and configure the desired software, and the deployment of web applications. This Project will require automating routine tasks with Ansible Configuration Management. Ansible is an open-source, command-line IT automation software application written in Python. It can configure systems, deploy software, and orchestrate advanced workflows to support application deployment, system updates, and more. Ansible's main strengths are simplicity and ease of use. It also has a strong focus on security and reliability, featuring minimal moving parts.

Ansible Client as a Jump Server (Bastion Host)

SSH bastion host is a jump server (or gateway server) that gives access to instances within a private network using the SSH protocol. You can log into the instances of your private network via a single, centralized, entry point. You can also monitor user access to the infrastructure. With the current architecture, the web servers would be inside a secured network which cannot be reached directly from the Internet.

Once this virtual machine is in place, you will need an SSH configuration for each server within the private network. I will however install and configure Ansible to simulate the use of a Jump server or Bastion host to access our Web Servers and create a simple Ansible playbook to automate servers configuration. The diagram below shows the Virtual Private Network (VPC) is divided into two subnets – The public subnet has public IP addresses and the Private subnet is only reachable by private IP addresses.



Step 1 - Launch an EC2 Instance

Create an AWS EC2 server based on Ubuntu Server 20.04 LTS. We will use this server to run playbooks. Edit the inbound rule to allow TCP port 22 as Ansible needs it to ssh into target servers from the Jenkins-Ansible host.

Instances (1) Info									
Find instance by attribute or tag (case-sensitive)									
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
<input type="checkbox"/>	Jenkins-Ansible	i-04bb48e760a699d72	Running	t2.micro	Initializing	No alarms	eu-west-2b	ec2-13-40-81-70.eu-we...	13.40.81.70


Step 2 - Configure Ansible

1. Create a new repository in GitHub and name it ansible-config.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)



Required fields are marked with an asterisk (*).

Owner *  enemg / Repository name *

✓ ansible-config is available.

Great repository names are short and memorable. Need inspiration? How about [reimagined-lamp](#) ?

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: **None**

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **None**

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  **main** as the default branch. Change the default name in your [settings](#).

2. Install Ansible by running these commands.

- `sudo apt update`
- `sudo apt install ansible`

3. Check your Ansible version by running the `ansible --version`.

```
ubuntu@ip-172-31-40-224:~$ ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.10 (default, May 26 2023, 14:05:08) [GCC 9.4.0]
ubuntu@ip-172-31-40-224:~$
```

4. Configure Jenkins build job to save your repository content every time you change it – this will solidify your Jenkins configuration skills acquired in Project 9.

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts) ?
- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/enyemg/ansible-config.git

Credentials ?

- none -

Add

Advanced

Add Repository

5. I created a new Freestyle project ansible in Jenkins and point it to your 'ansible-config' repository and configured Webhook in GitHub to trigger the ansible build.

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Rules Beta

Actions

Webhooks

Environments

Webhooks / Manage webhook

Settings Recent Deliveries

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *
http://13.40.81.70:8080/github-webhook/

Content type
application/json

6. Configure a Post-build job to save all (**) files, similar to Project 9 and then apply and change.

Post-build Actions

Archive the artifacts ?

Files to archive ?

xx

Advanced ▾

Add post-build action ▾

Save

Apply

- Click on build now to see if the setup works okay.

Build #2 (8 Jul 2023, 19:38:05)



Build Artifacts

 README.md 16 B  view



No changes.



Started by user [Nenye](#)



Revision: fbb1a07d0dc249fe4344e7852af1264d1d623d3d

Repository: <https://github.com/enyemg/ansible-config.git>

- refs/remotes/origin/main

- I tested my setup by making some changes to the README.md file in the main branch to confirm that builds start automatically and Jenkins saves the files (build artifacts) in the following folder: `ls /var/lib/jenkins/jobs/ansible/builds/<build_number>/archive/`
- The console output of this change on Jenkins should appear below.

Build #3 (8 Jul 2023, 19:40:52)



Build Artifacts

 [README.md](#) 42 B  [view](#)



Changes

1. Update README.md ([details](#) / [githubweb](#))



[Started by GitHub push by enyemg](#)



Revision: 96177b8e061c5eb43edd93b3bd9d470bfe384063

Repository: <https://github.com/enyemg/ansible-config.git>

- refs/remotes/origin/main



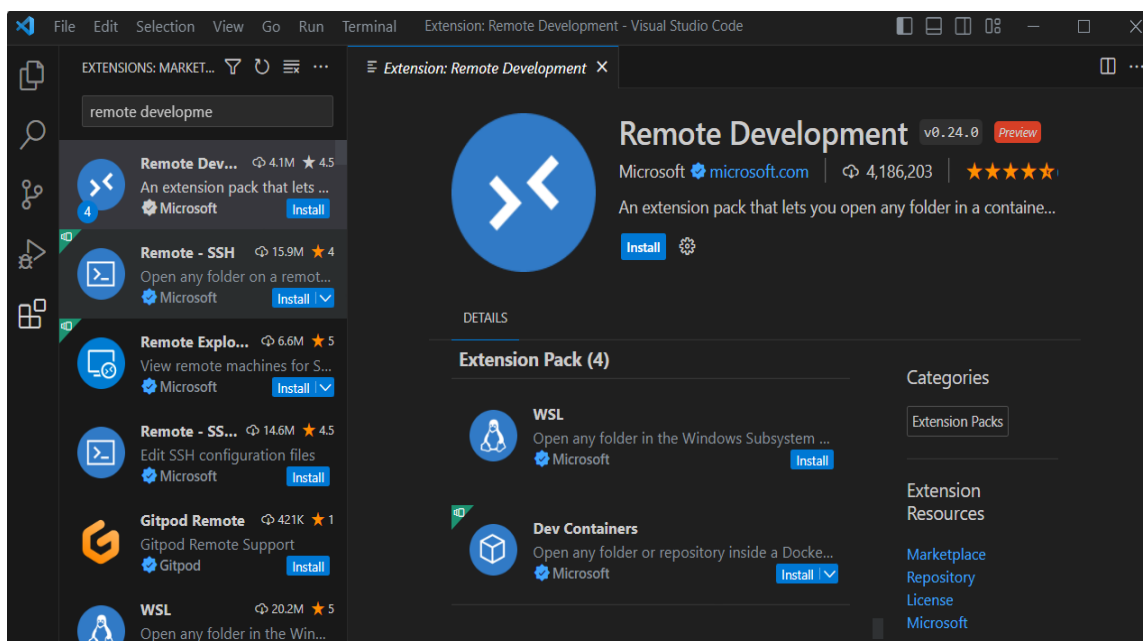
Console Output

```
Started by GitHub push by enyemg
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/ansible
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/ansible/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/enyemg/ansible-config.git # timeout=10
Fetching upstream changes from https://github.com/enyemg/ansible-config.git
> git --version # timeout=10
> git --version # 'git version 2.25.1'
> git fetch --tags --force --progress -- https://github.com/enyemg/ansible-config.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 96177b8e061c5eb43edd93b3bd9d470bfe384063 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 96177b8e061c5eb43edd93b3bd9d470bfe384063 # timeout=10
Commit message: "Update README.md"
> git rev-list --no-walk fbb1a07d0dc249fe4344e7852af1264d1d623d3d # timeout=10
Archiving artifacts
Finished: SUCCESS
```

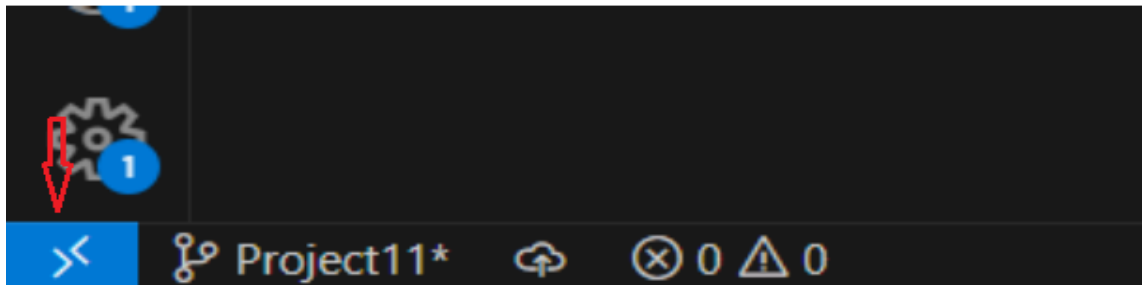
Note: Trigger Jenkins project execution only for /main branch. Every time you stop/start your Jenkins-Ansible server – you have to reconfigure GitHub webhook to a new IP address, to avoid this, you can allocate an Elastic IP to your Jenkins-Ansible server (you maybe charged for this).

Step 3 – Prepare the development environment using Visual Studio Code

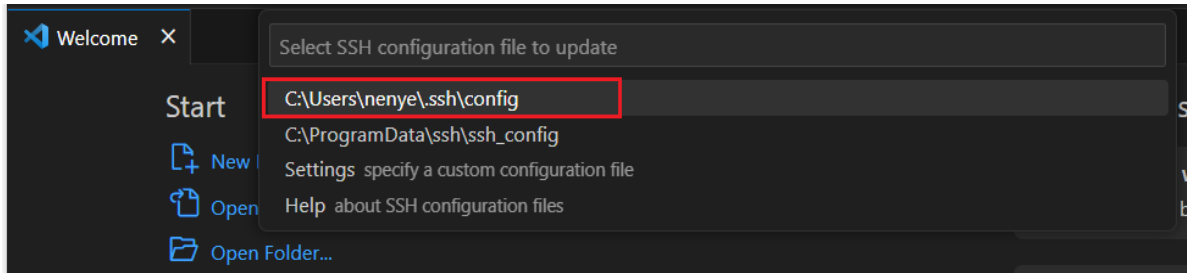
1. Visual Studio Code (VSC) is a code editor redefined and optimized for building and debugging modern web and cloud applications. I installed VSC but you can decide to use any other Source-code Editor of your choice and then configure it to connect to the newly created GitHub repository.
2. Launch VS Code & go to Extensions and search ssh. Select the extension as shown below and install it.



3. Click on the blue icon (Open a Remote Window) on the bottom-left corner.



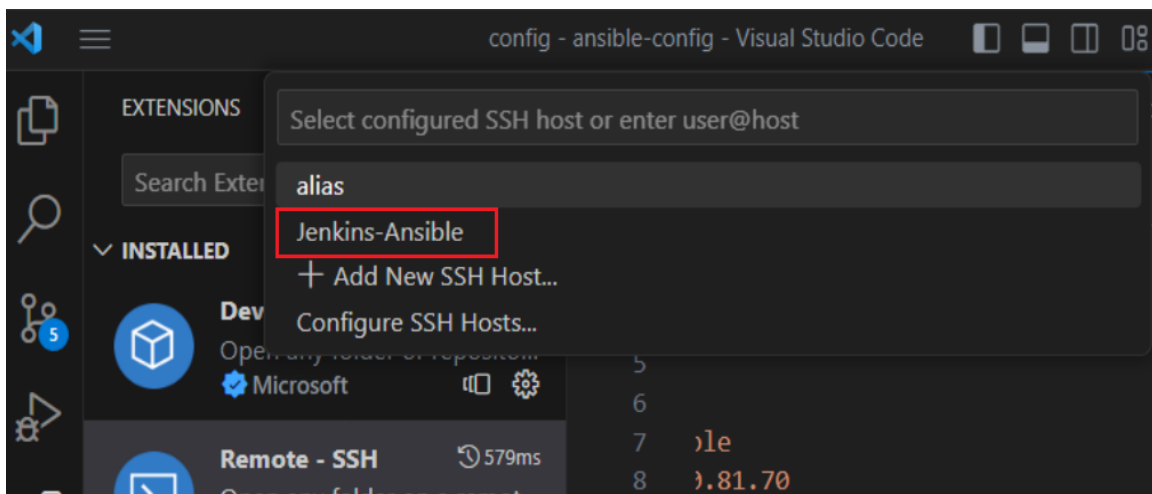
4. After clicking on the above icon, the pop-up will appear. Click on Open SSH Configuration File and select the file where you want to configure the remote host (EC2). In my case, I will select /Users/username/.ssh/config

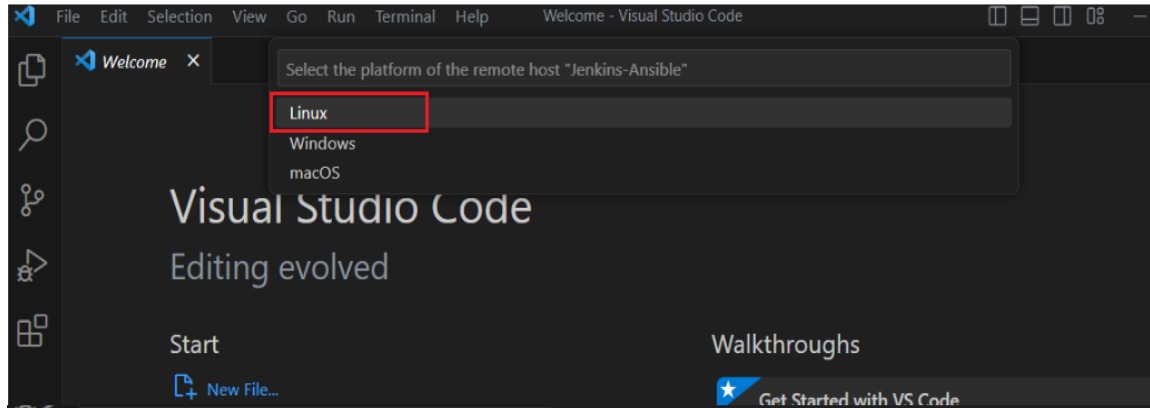


5. In this file, add the hosts with which you would like to connect for remote development in the below format and save.

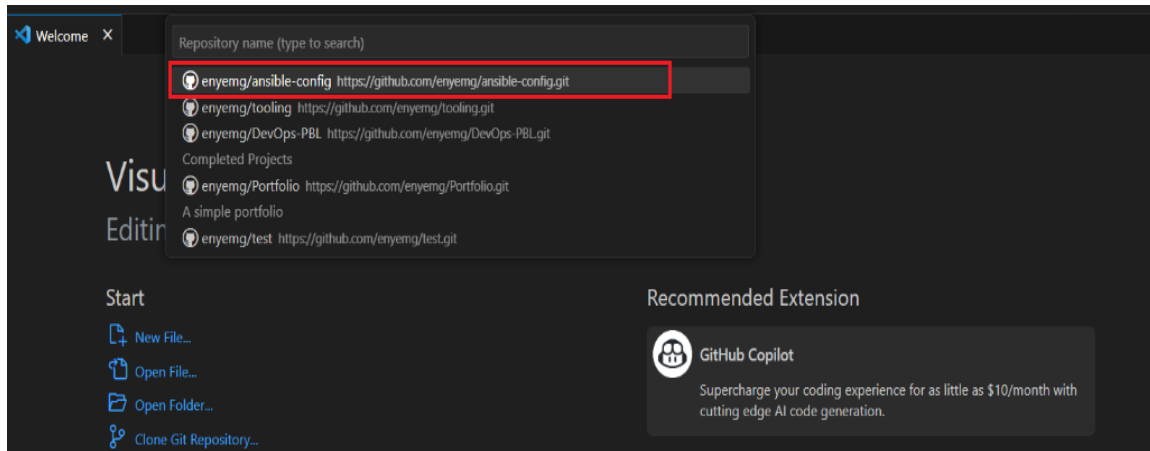
```
C: > Users > nenye > .ssh > config
1 # Read more about SSH config files: https://linux.die.net/man/5/ssh\_config
2 Host alias
3     HostName hostname
4     User user
5
6
7 Host Jenkins-Ansible
8     Hostname 13.40.81.70
9     User ubuntu
10    IdentityFile C:\Users\nenye\Downloads\LinuxKeyPair.pem
```

6. Click on the blue icon (bottom-left corner) to connect to the host Jenkins-Ansible and it will open a new window. As soon as it is connected, you will be able to see the alias name in the bottom-left corner.
7. Select the configured host and then the platform.





8. Clone down your ansible-config repository to your Jenkins-Ansible instance
git clone <ansible-config repo link>



```
ubuntu@ip-172-31-40-224:~$ git clone https://github.com/enyemg/ansible-config.git
Cloning into 'ansible-config'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), 1.22 KiB | 311.00 KiB/s, done.
```

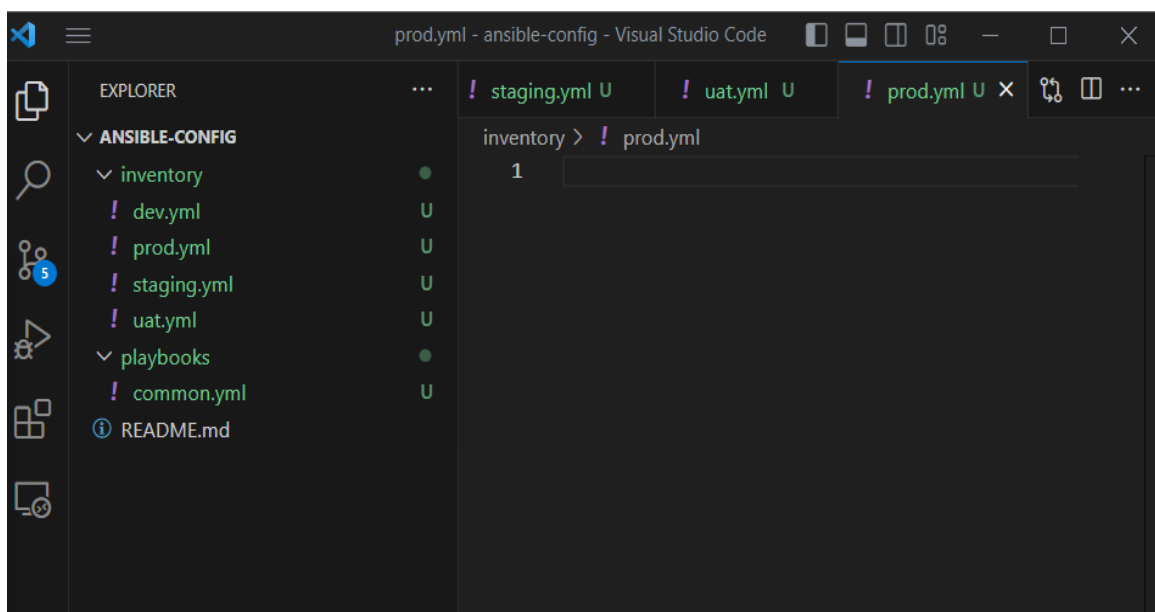
Step 4 - Begin Ansible Development

1. I created a new branch called proj45 that will be used for the development of a new feature in my ansible-config GitHub repository.

```
ubuntu@ip-172-31-40-224:~/ansible-config$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
ubuntu@ip-172-31-40-224:~/ansible-config$
```

2. Checkout the newly created feature branch to your local machine and start building your code and directory structure
3. Create a directory and name it playbooks – it will store all your playbook files.
 - Create a directory and name it inventory – it will be used to keep your hosts organised.
 - Within the playbooks folder, create your first playbook, and name it common.yml.
 - Within the inventory folder, create an inventory file (.yml) for each environment (Development, Staging Testing and Production) dev, staging, uat, and prod respectively.



Step 5 – Set up an Ansible Inventory

An Ansible inventory file defines the hosts and groups of hosts upon which commands, modules, and tasks in a playbook operate. This will be used to execute Linux commands on remote hosts and ensure that it is the intended configuration on a particular server that occurs. It is vital to have a way to organize our hosts in such an Inventory.

1. Import your key into ssh-agent:

```
eval `ssh-agent -s`
```

```
ssh-add <path-to-private-key>
```

```
ubuntu@ip-172-31-40-224:~$ eval `ssh-agent -s`  
Agent pid 6402  
ubuntu@ip-172-31-40-224:~$ ssh-add /home/ubuntu/.ssh/id_rsa  
Identity added: /home/ubuntu/.ssh/id_rsa (ubuntu@ip-172-31-40-224)
```

2. Confirm the key has been added with the command below, you should see the name of your key

```
ssh-add -l
```

```
ubuntu@ip-172-31-40-224:~$ ssh-add -l  
3072 SHA256:3fnWsd/9NfCXNk7NewRG21MApNycadq5ju5qRJLEgAs ubuntu@ip-172-31-40-224 (RSA)
```

3. Now, ssh into your Jenkins-Ansible server using ssh-agent

```
ssh -A ubuntu@public-ip
```

```
ubuntu@ip-172-31-40-224:~$ ssh -A ec2-user@18.130.26.135  
The authenticity of host '18.130.26.135 (18.130.26.135)' can't be established.  
ECDSA key fingerprint is SHA256:2SM3g3osJyVUdZx8Vjzr2nQt09FA0ltMDVyETM8b7Vs.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '18.130.26.135' (ECDSA) to the list of known hosts.  
Register this system with Red Hat Insights: insights-client --register  
Create an account or view all your systems at https://red.ht/insights-dashboard  
Last login: Sun Jul 9 14:12:01 2023 from 83.137.6.240  
[ec2-user@ip-172-31-26-195 ~]$
```

4. Update the inventory/dev.yml with this code:

```
[nfs]
```

```
<NFS-Server-Private-IP-Address> ansible_ssh_user='ec2-user'
```

[webservers]

<Web-Server1-Private-IP-Address> ansible_ssh_user='ec2-user'

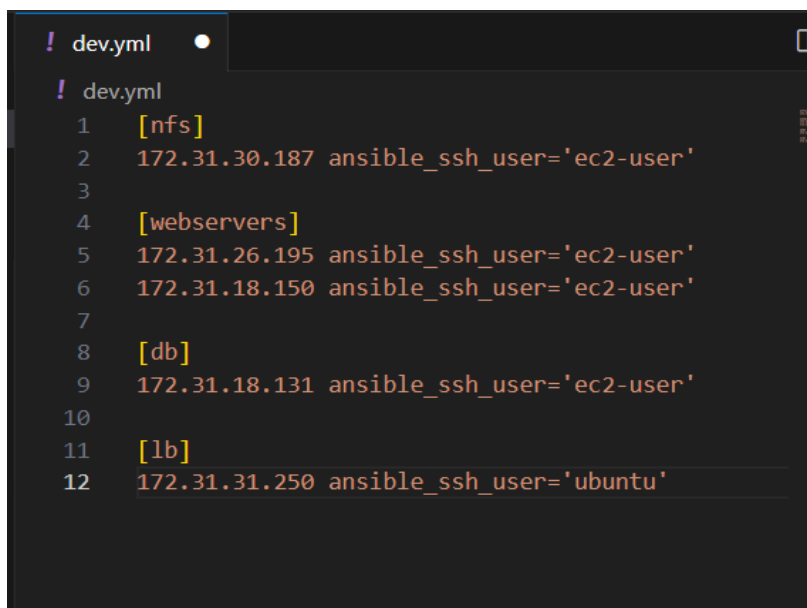
<Web-Server2-Private-IP-Address> ansible_ssh_user='ec2-user'

[db]

<Database-Private-IP-Address> ansible_ssh_user='ec2-user'

[lb]

<Load-Balancer-Private-IP-Address> ansible_ssh_user='ubuntu'

A screenshot of a code editor with a dark background. The file name 'dev.yml' is shown in the top left corner. The code is as follows:

```
! dev.yml
1  [nfs]
2  172.31.30.187 ansible_ssh_user='ec2-user'
3
4  [webservers]
5  172.31.26.195 ansible_ssh_user='ec2-user'
6  172.31.18.150 ansible_ssh_user='ec2-user'
7
8  [db]
9  172.31.18.131 ansible_ssh_user='ec2-user'
10
11 [lb]
12 172.31.31.250 ansible_ssh_user='ubuntu'
```

Step 6 – Create a Common Playbook

It is important to give Ansible the instructions on what needs to be performed on all servers listed in dev.yml and common.yml.

1. Update playbooks/common.yml file with the following code:

```

! common.yml
1  ---
2  - name: update web, nfs and db servers
3    hosts: webserver, nfs, db
4    remote_user: ec2-user
5    become: yes
6    become_user: root
7    tasks:
8      - name: ensure wireshark is at the latest version
9        yum:
10         name: wireshark
11         state: latest
12
13 - name: update LB server
14   hosts: lb
15   remote_user: ubuntu
16   become: yes
17   become_user: root
18   tasks:
19     - name: Update apt repo
20       apt:
21         update_cache: yes
22
23     - name: ensure wireshark is at the latest version
24       apt:
25         name: wireshark
26         state: latest

```

2. The code above is divided into two parts, each of which is intended to perform the same task: install
 - Wireshark - utility (or make sure it is updated to the latest version) on your RHEL 8 and Ubuntu servers.
 - Root: user to perform this task and respective package manager:
 - Yum: for RHEL 8 and
 - Apt: for Ubuntu

Step 7 – Update GIT with the latest code

Now all of your directories and files live on my machine, I will need to push changes made locally to GitHub. Since I have a separate branch, changes will be made on this branch, raise a Pull Request, and merge to the main branch.

1. I used git commands to add, commit and push your branch to GitHub.

```
git status
```

```
git add <selected files>
```

```
git commit -m "commit message"
```

```

● ubuntu@ip-172-31-40-224:~/ansible-config/inventory$ git commit -m "made changes"
[feature/proj45 6c56f91] made changes
Committer: Ubuntu <ubuntu@ip-172-31-40-224.eu-west-2.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

4 files changed, 12 insertions(+)
create mode 100644 inventory/dev.yml
create mode 100644 inventory/prod.yml
create mode 100644 inventory/staging.yml
create mode 100644 inventory/uat.yml
○ ubuntu@ip-172-31-40-224:~/ansible-config/inventory$ git push origin feature/proj45


```


```

● ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ git push origin feature/proj45
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 543 bytes | 108.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'feature/proj45' on GitHub by visiting:
remote:   https://github.com/enyemg/ansible-config/pull/new/feature/proj45
remote:
To https://github.com/enyemg/ansible-config.git
 * [new branch]      feature/proj45 -> feature/proj45


```

2. On Github, compare and pull down the latest changes on the feature branch into the main.


ansible-config
Public
Pin
Unwatch 1

 feature/proj45 had recent pushes 2 minutes ago
 Compare & pull request

main
3 branches
0 tags
Go to file
Add file
Code


Your main branch isn't protected

Protect this branch from force pushing or deletion, or require status checks before merging. [Learn more](#)

Protect this branch
×

feature/proj45

ansible-config

Go to file

Add file

Ubuntu

made changes

6:56/91 · 3 minutes ago

History

This branch is 1 commit ahead of, 1 commit behind main.

Contribute

Name	Last commit message	Last commit date
inventory	made changes	3 minutes ago
playbooks	made changes	10 minutes ago
README.md	Update README.md	4 hours ago

3. Once the code changes appear in the main branch – Jenkins will do its job and save all the files (build artifacts) to `/var/lib/jenkins/jobs/ansible/builds/<build_number>/archive/` directory on Jenkins-Ansible server.

```
ubuntu@ip-172-31-40-224:~$ sudo ls /var/lib/jenkins/jobs/ansible/builds/
1 2 3 legacyIds permalinks
```

4. The Console output on Jenkins should appear as this.

✓ Build #9 (9 Jul 2023, 13:49:21)

Build Artifacts

dev.yml

240 B

view

prod.yml

0 B

view

staging.yml

0 B

view

uat.yml

0 B

view

common.yml

530 B

view

README.md

42 B

view

Changes

1. made changes (details / githubweb)

Started by user Neny

Revision: 2c11a70be7ae10085f7a74326b17214453628beb

Repository: <https://github.com/enyemg/ansible-config.git>

refs/remotes/origin/main

✓ Console Output

```
Started by user Neny
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/ansible
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/ansible/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/enyemg/ansible-config.git # timeout=10
Fetching upstream changes from https://github.com/enyemg/ansible-config.git
> git --version # timeout=10
> git --version # 'git version 2.25.1'
> git fetch --tags --force --progress -- https://github.com/enyemg/ansible-config.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 34066df7dad48ea8b5b3872b2cf47dccc1390008 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 34066df7dad48ea8b5b3872b2cf47dccc1390008 # timeout=10
Commit message: "Merge pull request #1 from enyemg/feature/proj45"
> git rev-list --no-walk 34066df7dad48ea8b5b3872b2cf47dccc1390008 # timeout=10
Archiving artifacts
Finished: SUCCESS
```

Step 8 – Run the first Ansible test

From each of the servers and check if wireshark has been installed by running **which wireshark** or **wireshark --version**

```
ubuntu@LB:~$ which wireshark
/usr/bin/wireshark
ubuntu@LB:~$ wireshark --version
Wireshark 3.2.3 (Git v3.2.3 packaged as 3.2.3-1)
```

```
[ec2-user@webserver1 ~]$ which wireshark
/usr/bin/wireshark
[ec2-user@webserver1 ~]$ wireshark --version
Wireshark 3.4.10 (Git commit 733b3a137c2b)
```

After confirming, it is time to execute the ansible-playbook command and verify that the playbook works by running the below command.

```
cd ansible-config
```

```
ansible-playbook -i inventory/dev.yml playbooks/common.yml
```

```
ubuntu@ip-172-31-40-224:~/ansible-config$ ansible-playbook -i inventory/dev.yml playbooks/common.yml
```

```
PLAY [update web, nfs and db servers] *****

TASK [Gathering Facts] *****
[DEPRECATION WARNING]: Distribution rhel 9.2 on host 172.31.26.195 should use /usr/libexec/platform-python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible release will default to using the discovered platform python for this host. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
ok: [172.31.26.195]
[DEPRECATION WARNING]: Distribution rhel 9.2 on host 172.31.18.150 should use /usr/libexec/platform-python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible release will default to using the discovered platform python for this host. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
ok: [172.31.18.150]
[DEPRECATION WARNING]: Distribution rhel 9.2 on host 172.31.30.187 should use /usr/libexec/platform-python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible release will default to using the discovered platform python for this host. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
ok: [172.31.30.187]
[DEPRECATION WARNING]: Distribution rhel 9.2 on host 172.31.18.131 should use /usr/libexec/platform-python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible release will default to using the discovered platform python for this host. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
ok: [172.31.18.131]

TASK [ensure wireshark is at the latest version] *****
ok: [172.31.26.195]
ok: [172.31.18.150]
ok: [172.31.30.187]
ok: [172.31.18.131]
```

```
PLAY [update LB server] *****

TASK [Gathering Facts] *****
ok: [172.31.31.250]

TASK [Update apt repo] *****
changed: [172.31.31.250]

TASK [ensure wireshark is at the latest version] *****
changed: [172.31.31.250]

PLAY RECAP *****
172.31.18.131      : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.18.150      : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.26.195      : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.30.187      : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.31.250      : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu@ip-172-31-40-224:~/ansible-config$
```


The above result confirms that our Ansible playbook has been set up correctly and my updated Ansible architecture should now look like this.

