

EC2 Instance BootStrapping

This is a lab done within the acloudguru, same steps apply to an EC2 instance you have created in the past or create a new one.

1. First step would be to go to the EC2 instance and select it.
2. Once selected click connect > and on the EC2 instance connect click connect.
3. You then want to connect to the instance by typing this command

a. `ssh cloud_user@3.82.246.2`

b. Type yes


c. You then would input the password to that webserver

```
ubuntu@ip-10-0-2-100:~$ ssh cloud_user@3.82.246.2
The authenticity of host '3.82.246.2 (3.82.246.2)' can't be established.
ECDSA key fingerprint is SHA256:A7Lz2aXFajxi+RLbOb3p+nEyJLJReE1TA4j6cWvWJCI.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.82.246.2' (ECDSA) to the list of known hosts.
Password: █
```

4. So now we need to get this up to company standards, so we need to ensure it is fully up to date. We need to install an apache web server, AWS CLI tool, my SQL installed, and setup an HTML file that gives detail about the server.
5. Before we get into this, we want to check for new updates so run `sudo apt update` and press enter
6. Then you would type in `sudo apt-get upgrade -y`
7. Or you can do this in a single command, which would be `sudo apt-get update && sudo apt-get upgrade -y`

But having it under one line may or may not have issues. If it does, you would have to separate them. For the sake of the lab, it was having issues under 1 line.

- a. First part of the command is to make sure we have the up to date packages in our repository
 - b. Second part is to get those updates to installed and -y will allow us to install those updates without asking us to confirm throughout the process
8. It may or may not ask for the password again, if so input the password
 9. Once the update is completed, we are going to install apache, the command would be `sudo apt-get install apache2 -y`
 10. To confirm all of this has been installed go back to the AWS console > Instances page
 11. In the details section of the instance, you will see the public IPv4 address, copy that address and paste in another tab
 - a. You don't want to click the hyperlink as it would open up in HTTPS where we want this in HTTP



ubuntu

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

12. So once we have verified we have this installed lets head back in the terminal
13. Next thing we will install the AWS CLI tool
 - a. We need to download a ZIP file, so we need to unzip it by installing the unzip tool
 - b. The command to use is `sudo apt-get install unzip -y`
14. So here we would need to pull down the file for AWS CLI, so here we would use the curl command
15. Download the AWS CLI tool: `curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
16. Unzip the file: `unzip awscliv2.zip`
17. Install: `sudo ./aws/install`
18. And lastly verify: `aws --version`

```
cloud_user@ip-10-0-2-100:~$ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
cloud_user@ip-10-0-2-100:~$ aws --version
aws-cli/2.7.21 Python/3.9.11 Linux/5.11.0-1028-aws exe/x86_64.ubuntu.20 prompt/off
cloud_user@ip-10-0-2-100:~$
```

19. So now we have AWS Cli setup, so now we have to get our HTML side of things taken care of.
20. So to edit our web page html file we need to grant normal user access to the file by typing in this command: `sudo chmod 777 /var/www/html/index.html`
 - a. Now in a production environment you would never want to pull permissions like this, but for a demo environment it is okay
21. Now we want to put information in the file
22. This command is to get instance metadata about our servers AZ: `curl http://3.82.246.2/latest/meta-data/placement/availability-zone`
23. The command above shows up that the html file is empty...?
24. Add AZ, instance ID, public IP, and local IP instance metadata to our website

```
echo '<html><h1>Bootstrap Demo</h1><h3>Availability Zone: ' > /var/www/html/index.html
```

```
curl http://169.254.169.254/latest/meta-data/placement/availability-zone >>  
/var/www/html/index.html
```

```
echo '</h3> <h3>Instance Id: ' >> /var/www/html/index.html
```

```
curl http://169.254.169.254/latest/meta-data/instance-id >> /var/www/html/index.html
```

```
echo '</h3> <h3>Public IP: ' >> /var/www/html/index.html
```

```
curl http://169.254.169.254/latest/meta-data/public-ipv4 >> /var/www/html/index.html
```

```
echo '</h3> <h3>Local IP: ' >> /var/www/html/index.html
```

```
curl http://169.254.169.254/latest/meta-data/local-ipv4 >> /var/www/html/index.html
```

```
echo '</h3></html>' >> /var/www/html/index.html
```

25. So once those commands are in, let's go back to our apache webpage and click refresh

Bootstrap Demo

Availability Zone: us-east-1c

Instance Id: i-07c90343432c10892

Public IP: 3.82.246.2

Local IP: 10.0.2.100

26. So, everything seems to be working so far. But we still have 1 more thing to do... Which is to install my SQL

27. To install my SQL: `sudo apt-get install mysql-server -y`

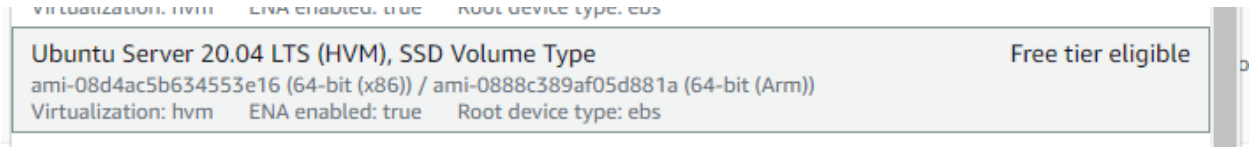
Once that is installed, you're finished. We are up to company standards which may seem like a lot and it is. Every single time an instance is created, you have to set up the instance just as we did above.

BUT

There is much more efficient way in doing this and that is with bootstrapping

BOOTSTRAPPING

1. Go to the EC2 instance you want and select it and click launch instances
2. We are naming this instance webserver-02 as we already have an 01
3. For the image we have been using Ubuntu, so we want to use Ubuntu Server 20.04 LTS



4. For instance type we would want to use t3.Micro
5. For the key pair we don't need one since we are not logging in from the terminal.
6. On the network settings click edit
7. For the public IP > Enable
8. For security groups, select the security group you created (which the security group is what we need to access the server)
9. Lastly before we launch instance, open up the advanced detail and scroll all the way to the bottom and find User data
10. Here this is where we want to put our script, which it will do everything we did manually, but it will do this while the instance is being created.

```
#!/bin/bash
sudo apt-get update -y
sudo apt-get install apache2 unzip -y
sudo systemctl enable apache2
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
echo '<html><h1>Bootstrap Demo</h1><h3>Availability Zone: ' > /var/www/html/index.html
curl http://169.254.169.254/latest/meta-data/placement/availability-zone >>
/var/www/html/index.html
echo '</h3> <h3>Instance Id: ' >> /var/www/html/index.html
curl http://169.254.169.254/latest/meta-data/instance-id >> /var/www/html/index.html
echo '</h3> <h3>Public IP: ' >> /var/www/html/index.html
curl http://169.254.169.254/latest/meta-data/public-ipv4 >> /var/www/html/index.html
echo '</h3> <h3>Local IP: ' >> /var/www/html/index.html
curl http://169.254.169.254/latest/meta-data/local-ipv4 >> /var/www/html/index.html
echo '</h3></html>' >> /var/www/html/index.html
sudo apt-get install mysql-server
sudo systemctl enable mysql
```

11. So it may take some time for this instance to have a status check of 2/2 checks passed so give it some time
12. Select the new instance we created and click connect > EC2 instance connect
13. Verify if Apache was installed by typing in `sudo systemctl status apache2`

```
ubuntu@ip-10-0-2-16:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2022-08-07 22:23:53 UTC; 4min 10s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2081 (apache2)
    Tasks: 55 (limit: 1116)
   Memory: 5.0M
   CGroup: /system.slice/apache2.service
           └─2081 /usr/sbin/apache2 -k start
             └─2082 /usr/sbin/apache2 -k start
               └─2084 /usr/sbin/apache2 -k start

Aug 07 22:23:53 ip-10-0-2-16 systemd[1]: Starting The Apache HTTP Server...
Aug 07 22:23:53 ip-10-0-2-16 systemd[1]: Started The Apache HTTP Server.
ubuntu@ip-10-0-2-16:~$
```

14. We see it running but we want to double check by typing in `ps aux | grep apache`

```
ubuntu@ip-10-0-2-16:~$ ps aux | grep apache
root        2081    0.0  0.4  6676  4536 ?        Ss   22:23   0:00 /usr/sbin/apache2 -k start
www-data    2082    0.0  0.4 1211564 4116 ?        Sl   22:23   0:00 /usr/sbin/apache2 -k start
www-data    2084    0.0  0.4 1211564 4116 ?        Sl   22:23   0:00 /usr/sbin/apache2 -k start
ubuntu      3189    0.0  0.0   8168   724 pts/0    S+   22:29   0:00 grep --color=auto apache
ubuntu@ip-10-0-2-16:~$
```

15. So now that we have verified that apache is indeed running on the server, now lets check on MySQL

16. The command to check on the SQL is: `sudo systemctl status mysql`

17. If nothing comes up use the ps aux command. So type in `ps aux | grep mysql`

```
ubuntu@ip-10-0-2-16:~$ sudo systemctl status mysql
Unit mysql.service could not be found.
ubuntu@ip-10-0-2-16:~$ sudo systemctl status mysqld
Unit mysqld.service could not be found.
ubuntu@ip-10-0-2-16:~$ ps aux | grep mysql
ubuntu      3210    0.0  0.0   8168   656 pts/0    R+   22:35   0:00 grep --color=auto mysql
ubuntu@ip-10-0-2-16:~$
```

18. So it looks like it actually didn't get installed. So we can try one more thing by starting the service by typing in `sudo systemctl start mysql`

19. So this command still couldn't find it

20. So use the curl command to get user data. So type in `curl http://54.205.80.59/latest/user-data`

a. Remember the public IP is different on this instance, don't get confused with the other instance (01)

21. So looking back at the script... it was missing one piece and it was at the end of the script. We forgot to include the -y at the end of mysql-server

22. So simply install it by typing in `sudo apt-get install mysql-server -y`

23. To enable the sql server type in: `sudo systemctl enable mysql`

24. So we will create another instance

25. This instance are the same steps 1-12

26. While we wait for 03 to complete we are gonna check and see if 02 html file is working properly

Bootstrap Demo

Availability Zone: us-east-1c

Instance Id: i-0ff648d07364a77c8

Public IP: 54.205.80.59

Local IP: 10.0.2.16

27. So, we see it is working just fine
28. Once we see the checks have passed click on the webserver and click connect > EC2 instance connect
29. We are going to repeat steps 13-17
30. Don't forget to type in `aws --version`
31. Once that is done you go into the dashboard, copy the IP address and go to a new tab and see if the bootstrap demo appears which it does.

Complete!!