Name: Reddy Hemanth
Reg.no:RA2011028010102

EXPERIMENT 2

Aim :- Creating Amazon EC2 Instances - Creating a LAMP Instance in the AWS CLI

A] Creating Amazon EC2 Instance

Procedure :

Steps to get the Amazon AWS access key ID and secret key

1. Go to the IAM Console and click on Users.
2. Click on the User that you want to create the access key for. Click on the actual row and not the check box.
3. On the next screen, click on "Create Access Key". If you have created keys before you should be able to see them (but can't download them again)
4. You will see a popup that allows you to download the access ID and key. The keys can be downloaded only once so make sure you save it in a safe place. You can, however, create another key later on.
5. The downloaded CSV has both the access id and key.

Configure Amazon CLI (Command Line Interface)

1. Type in "aws configure" on the command line.
2. Enter the Access ID, key and the default region.
3. That configures the CLI. This creates a directory called .aws in home. This directory has the credentials and the config file.
4. To test the configuration we will create a security group and then delete it from the AWS console. To create the security group type in

aws ec2 create-security-group --group-name my-sg --description "My security group"

This will create a new security group. Logon to AWS console to double check if you can see the security group (under EC2). You can then delete the group.

create AWS EC2 instance using CLI

We now finally look at how to create the EC2 instance using CLI. The CLI command for creating instance is called `run-instances`. When you create an instance from the console, you go through seven steps of configuration. All of that can be done using specific parameters on the CLI. While creating the instance we want to be able to select the AMI (machine image); select the instance type (hardware); set the VPC, IAM role, and other configuration parameters; configure additional block storage; add tags; add security groups and then launch one or multiple instances.

aws ec2 create-security-group --group-name EC2SecurityGroup --description "Security Group for EC2 instances to allow port 22"

aws ec2 authorize-security-group-ingress --group-name EC2SecurityGroup -protocol tcp --port 22 --cidr 0.0.0.0/0 aws ec2 describe-security-groups --group-names EC2SecurityGroup

Commands for EC2 :

1) AWS Configure

```
C:\Users\Admin>aws configure
AWS Access Key ID [****************FL4K]: AKIA2TKXESQCM3L3OVFF
AWS Secret Access Key [****************HQE0]: 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
Default region name [None]: us-east-1
Default output format [None]: json
```

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 create-key-pair --key-name demokeypair --query 'KeyMaterial' --output text >demokeypair.pem
```

2) Create – key value - pair

3) Describe key pair

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 describe-key-pairs --key-name demokeypair
{
    "KeyPairs": [
        {
            "KeyPairId": "key-098bf943f116ba623",
            "KeyFingerprint": "cd:f2:d6:4e:cb:97:8f:d4:19:c2:22:a3:65:fd:b6:8f:ce:ed:37:3e",
            "KeyName": "demokeypair",
            "KeyType": "rsa",
            "Tags": [],
            "CreateTime": "2022-08-26T18:24:31+00:00"
        }
    ]
}
```

4) Create security group

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 create-security-group --group-name demo-sg --description "Demo CLI Security Group"  --vpc-id vpc-042b52670a9bac
{
    "GroupId": "sg-0f94d19f489ddf564"
}
```

5) Authorize security group

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 authorize-security-group-ingress --group-id sg-0f94d19f489ddf564 --protocol tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0c14c2eacd6e6a450",
            "GroupId": "sg-0f94d19f489ddf564",
            "GroupOwnerId": "728716579844",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

6) Create EC2

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 run-instances --image-id ami-05fa00d4c63e32376 --instance-type t2.micro --subnet-id subnet-0e6f00f98e52d0971 --count 1 --security-group
-ids sg-0f94d19f489ddf564 --key-name demokeypair
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-05fa00d4c63e32376",
            "InstanceId": "i-003b25d102610cc17",
            "InstanceType": "t2.micro",
            "KeyName": "demokeypair",
            "LaunchTime": "2022-08-26T19:07:22+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1c",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-31-94-42.ec2.internal",
            "PrivateIpAddress": "172.31.94.42",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-0e6f00f98e52d0971",
            "VpcId": "vpc-042b52670a9bac0cb",
-- More --
```

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 run-instances --image-id ami-05fa00d4c63e32376 --instance-type t2.micro --subnet-id subnet-0e6f00f98e52d0971 --count 1 --security
-ids sg-0f94d19f489ddf564 --key-name demokeypair
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-05fa00d4c63e32376",
            "InstanceId": "i-003b25d102610cc17",
            "InstanceType": "t2.micro",
            "KeyName": "demokeypair",
            "LaunchTime": "2022-08-26T19:07:22+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1c",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-31-94-42.ec2.internal",
            "PrivateIpAddress": "172.31.94.42",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-0e6f00f98e52d0971",
            "VpcId": "vpc-042b52670a9bac0cb",
-- More --
```

7) Describe Instance EC2

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 describe-instances --instance-ids i-003b25d102610cc17
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-05fa00d4c63e32376",
                    "InstanceId": "i-003b25d102610cc17",
                    "InstanceType": "t2.micro",
                    "KeyName": "demokeypair",
                    "LaunchTime": "2022-08-26T19:07:22+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1c",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-172-31-94-42.ec2.internal",
                    "PrivateIpAddress": "172.31.94.42",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-3-87-190-140.compute-1.amazonaws.com",
                    "PublicIpAddress": "3.87.190.140",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
-- More  --
```

8) Terminate Instances

```
D:\clg_\sem 5\DCN\lab\ss\exp2>aws ec2 terminate-instances --instance-ids i-003b25d102610cc17
{
    "TerminatingInstances": [
        {
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "InstanceId": "i-003b25d102610cc17",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

B] Creating a LAMP Instance in the AWS CLI

Procedure :

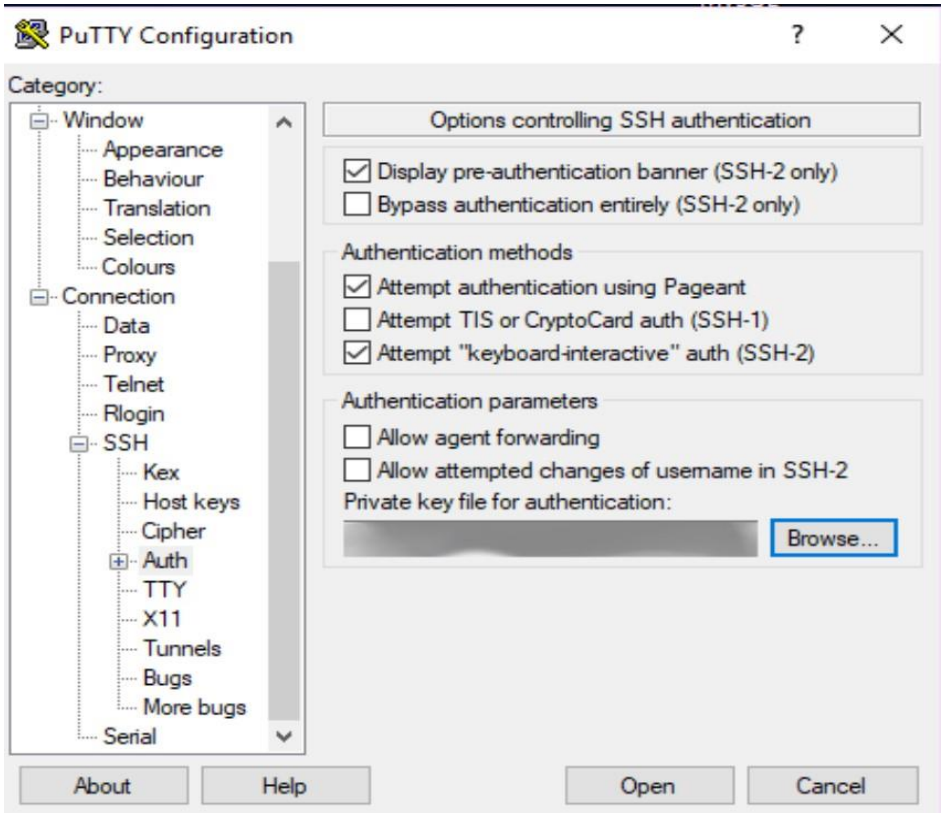
We will add four total rules. So go ahead and add three other rules. Set the types for all of the rules to:

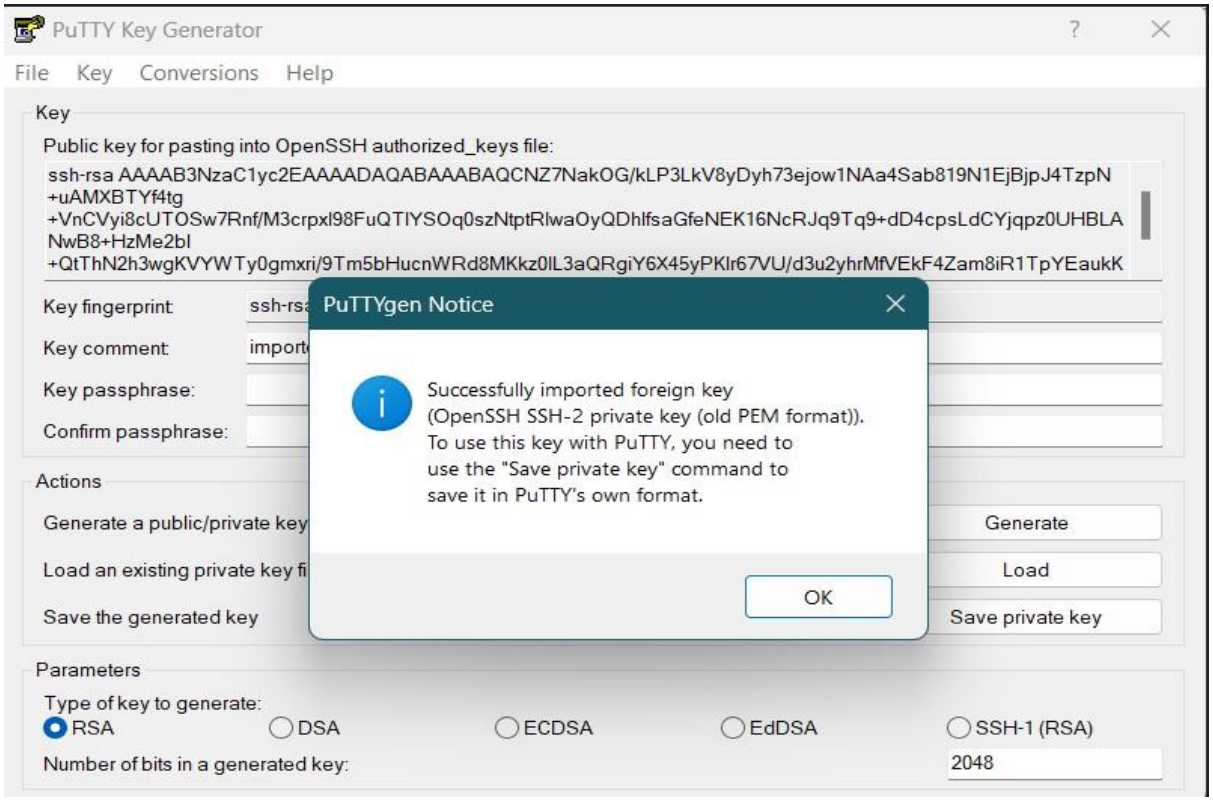
- SSH
- MySQL/Aurora
- HTTP
- HTTPS

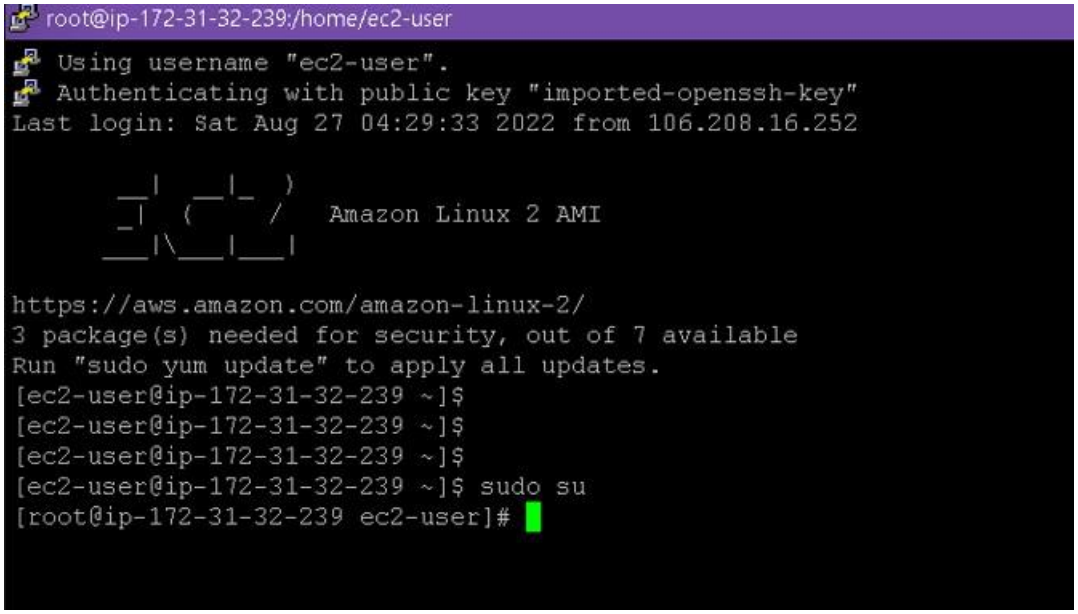
If you're on Windows, download the latest version of PuTTY to save yourself some time and heartache. If you're using MacOS, you're more than welcome to follow along, but you can SSH into your instance from the native command line.

1. Hit the windows Key, and search for PuTTYgen. Run this program and hit "Load".
2. Search for the Private Key File you just downloaded.
3. Click "Save Private Key", give it a name, and save it in a secure location.
4. You do not need a username and password attached to the file.
5. Now, open up PuTTY, and on the left hand pane, expand the tab "SSH", and then click on "Auth".
6. Set the "Private key file for authentication" to the private key you saved from PuTTYgen.
7. Scroll back up to "Session", and enter the host name for the new instance. Just copy and paste the IP address that we allocated with Elastic IP, enter a name in the "Saved Sessions" box, and click save, so that you can SSH in with just two clicks!
8. You can now double click the saved session, just confirm the next dialog box, and the command line will now prompt you for a user, type "ec2user" and hit enter

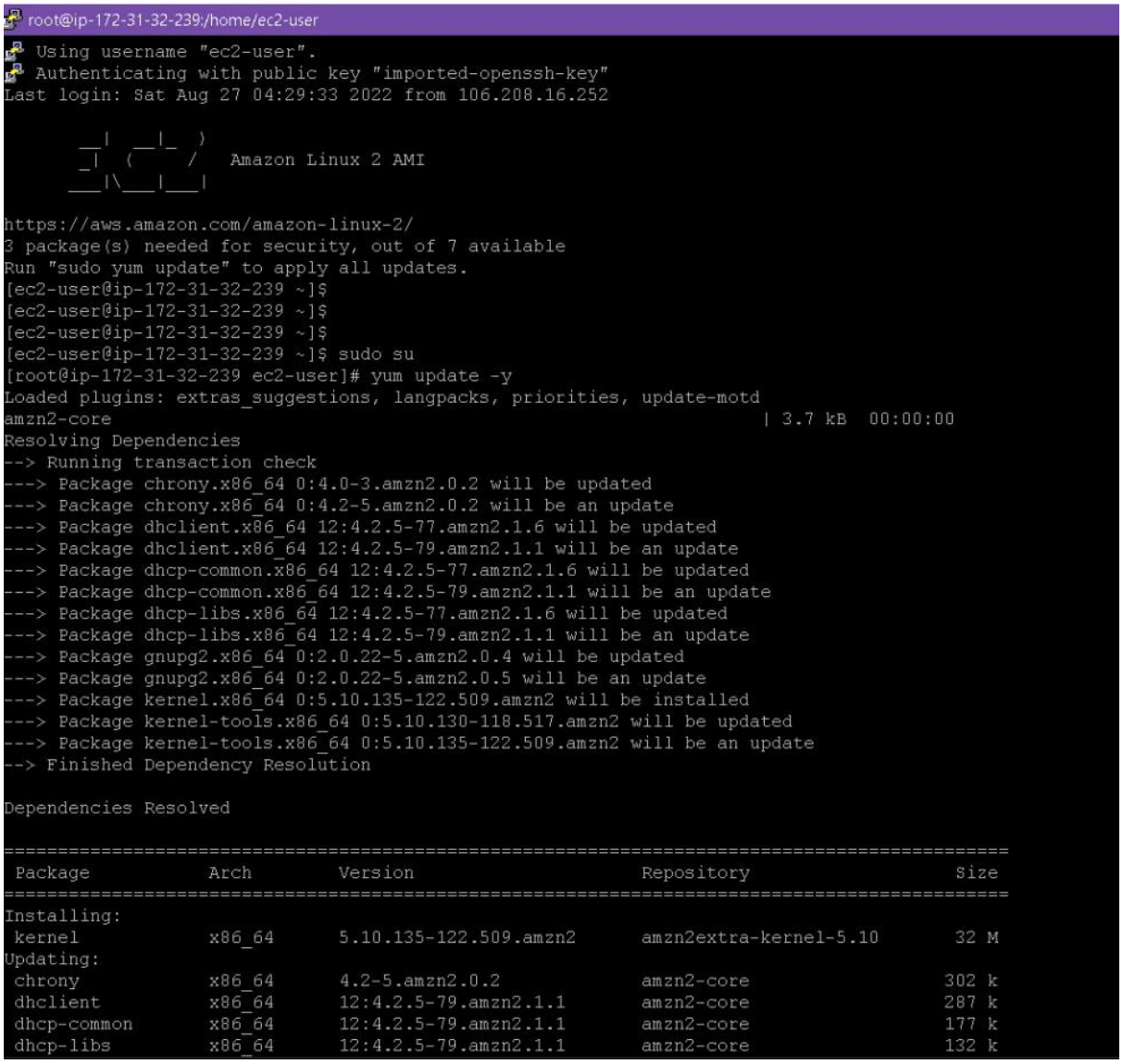
Verification :







- Ec2 instance is connected. 10. Type the commands to intall LAMP (linux, apache, mysql, php) server.
- Firstly, type sudo su to become the root user.
- To update all the packages in your instance type “yum update -y”



- To install Apache server in linux, type “yum install httpd”

```
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]# yum install httpd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.54-1.amzn2 will be installed
--> Processing Dependency: httpd-tools = 2.4.54-1.amzn2 for package: httpd-2.4.54-1.amzn2.x86
_64
--> Processing Dependency: httpd-filesystem = 2.4.54-1.amzn2 for package: httpd-2.4.54-1.amzn
2.x86_64
--> Processing Dependency: system-logos-httpd for package: httpd-2.4.54-1.amzn2.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.54-1.amzn2.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.54-1.amzn2.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.54-1.amzn2.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.54-1.amzn2.x86_6
4
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.54-1.amzn2.x86_64
--> Running transaction check
---> Package apr.x86_64 0:1.7.0-9.amzn2 will be installed
---> Package apr-util.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) = 1.6.1-5.amzn2.0.2 for package: apr-util-1.6
.1-5.amzn2.0.2.x86_64
---> Package generic-logos-httpd.noarch 0:18.0.0-4.amzn2 will be installed
---> Package httpd-filesystem.noarch 0:2.4.54-1.amzn2 will be installed
---> Package httpd-tools.x86_64 0:2.4.54-1.amzn2 will be installed
---> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
---> Package mod_http2.x86_64 0:1.15.19-1.amzn2.0.1 will be installed
--> Running transaction check
---> Package apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================
 Package                  Arch          Version                    Repository          Size
================================================================================================
```

- To install mysql or mariadb type “yum install mariadb mariadb-server”.

```
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]# yum install mariadb mariadb-server
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package mariadb.x86_64 1:5.5.68-1.amzn2 will be installed
---> Package mariadb-server.x86_64 1:5.5.68-1.amzn2 will be installed
--> Processing Dependency: perl-DBI for package: 1:mariadb-server-5.5.68-1.amzn2.x86_64
--> Processing Dependency: perl-DBD-MySQL for package: 1:mariadb-server-5.5.68-1.amzn2.x86_64
--> Processing Dependency: perl(Data::Dumper) for package: 1:mariadb-server-5.5.68-1.amzn2.x8
6_64
--> Processing Dependency: perl(DBI) for package: 1:mariadb-server-5.5.68-1.amzn2.x86_64
--> Running transaction check
---> Package perl-DBD-MySQL.x86_64 0:4.023-6.amzn2 will be installed
---> Package perl-DBI.x86_64 0:1.627-4.amzn2.0.2 will be installed
--> Processing Dependency: perl(RPC::PlServer) >= 0.2001 for package: perl-DBI-1.627-4.amzn2.
0.2.x86_64
--> Processing Dependency: perl(RPC::PlClient) >= 0.2000 for package: perl-DBI-1.627-4.amzn2.
0.2.x86_64
---> Package perl-Data-Dumper.x86_64 0:2.145-3.amzn2.0.2 will be installed
--> Running transaction check
---> Package perl-PlRPC.noarch 0:0.2020-14.amzn2 will be installed
--> Processing Dependency: perl(Net::Daemon) >= 0.13 for package: perl-PlRPC-0.2020-14.amzn2.
noarch
```

- To install php, type “yum install php php-mysql”.

```
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]# yum install php php-mysql
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package php-mysql is obsoleted by php-mysqlnd, trying to install php-mysqlnd-5.4.16-46.amzn2.
0.2.x86_64 instead
Resolving Dependencies
--> Running transaction check
---> Package php.x86_64 0:5.4.16-46.amzn2.0.2 will be installed
--> Processing Dependency: php-cli(x86-64) = 5.4.16-46.amzn2.0.2 for package: php-5.4.16-46.a
mzn2.0.2.x86_64
--> Processing Dependency: php-common(x86-64) = 5.4.16-46.amzn2.0.2 for package: php-5.4.16-4
6.amzn2.0.2.x86_64
---> Package php-mysqlnd.x86_64 0:5.4.16-46.amzn2.0.2 will be installed
--> Processing Dependency: php-pdo(x86-64) = 5.4.16-46.amzn2.0.2 for package: php-mysqlnd-5.4
.16-46.amzn2.0.2.x86_64
--> Running transaction check
---> Package php-cli.x86_64 0:5.4.16-46.amzn2.0.2 will be installed
---> Package php-common.x86_64 0:5.4.16-46.amzn2.0.2 will be installed
--> Processing Dependency: libzip.so.2()(64bit) for package: php-common-5.4.16-46.amzn2.0.2.x
86_64
```

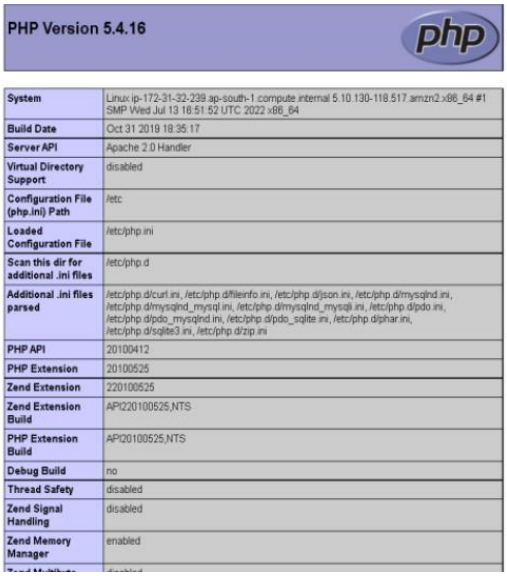
- Type “yum search php” to see all the packages installed in the server.

```
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]# yum search php
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
================================== N/S matched: php ==================================
graphviz-php.x86_64 : PHP extension for graphviz
php.x86_64 : PHP scripting language for creating dynamic web sites
php-bcmath.x86_64 : A module for PHP applications for using the bcmath library
php-cli.x86_64 : Command-line interface for PHP
php-common.x86_64 : Common files for PHP
php-dba.x86_64 : A database abstraction layer module for PHP applications
php-devel.x86_64 : Files needed for building PHP extensions
php-embedded.x86_64 : PHP library for embedding in applications
php-enchant.x86_64 : Enchant spelling extension for PHP applications
php-fpm.x86_64 : PHP FastCGI Process Manager
php-gd.x86_64 : A module for PHP applications for using the gd graphics library
php-intl.x86_64 : Internationalization extension for PHP applications
php-ldap.x86_64 : A module for PHP applications that use LDAP
php-mbstring.x86_64 : A module for PHP applications which need multi-byte string handling
php-mysql.x86_64 : A module for PHP applications that use MySQL databases
php-mysqlnd.x86_64 : A module for PHP applications that use MySQL databases
php-odbc.x86_64 : A module for PHP applications that use ODBC databases
php-pdo.x86_64 : A database access abstraction module for PHP applications
php-pear.noarch : PHP Extension and Application Repository framework
php-pgsql.x86_64 : A PostgreSQL database module for PHP
php-process.x86_64 : Modules for PHP script using system process interfaces
php-pspell.x86_64 : A module for PHP applications for using pspell interfaces
php-recode.x86_64 : A module for PHP applications for using the recode library
php-snmp.x86_64 : A module for PHP applications that query SNMP-managed devices
php-soap.x86_64 : A module for PHP applications that use the SOAP protocol
php-xml.x86_64 : A module for PHP applications which use XML
php-xmlrpc.x86_64 : A module for PHP applications which use the XML-RPC protocol
rrdtool-php.x86_64 : PHP RRDtool bindings
uuid-php.x86_64 : PHP support for Universally Unique Identifier library
php-pecl-memcache.x86_64 : Extension to work with the Memcached caching daemon
```

- Enabling the mariadb server.

```
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]# systemctl start mariadb
[root@ip-172-31-32-239 ec2-user]# systemctl enable mariadb
Created symlink from /etc/systemd/system/multi-user.target.wants/mariadb.service to /usr/lib/
systemd/system/mariadb.service.
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
[root@ip-172-31-32-239 ec2-user]#
```

- After enabling httpd (apache server) , go to the directory where cd /var/www/html/ .



Result:

LAMP server is successfully created using AWS CLI