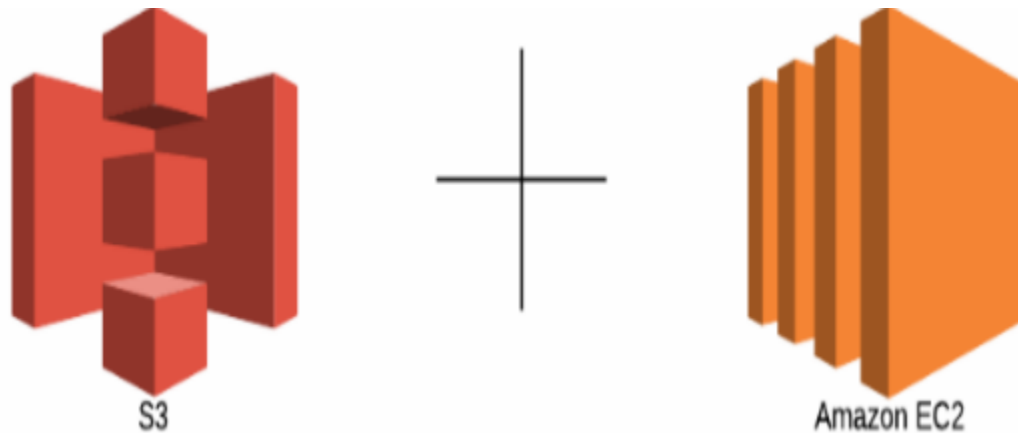


S3 Bucket Administration from EC2 Instance using AWS CLI



In this Lab, we are going to access and administer S3 Buckets from EC2 Instance using AWS CLI. We are also going to learn how to mount S3 Buckets to an EC2 Instance using S3fs file system. This is a best practice in industries if you want to archive data from local server or an EC2 Instance to S3 Buckets.

Below is the list of Tasks:

Task 1: Create IAM Role

Task 2: Launch & Configure EC2 Instance

Task 3: Connect to the EC2 Instance

Task 4: S3 Bucket Administration using AWS CLI

Task 5: Mount S3 Buckets to EC2 Instance using AWS CLI

Task 1: Create IAM Role

Login to the AWS Management Console.

Navigate to IAM Service and click on Roles.

Click on Create Role.


Make sure to select the Use Case as **EC2**.

Click Next: Permissions.

The screenshot shows the AWS IAM 'Create role' console page. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information 'Prasad_SMU'. The main heading is 'Create role' with a progress indicator showing four steps, with the first step being active. The first step is 'Select type of trusted entity', which has four options: 'AWS service' (selected), 'Another AWS account', 'Web identity', and 'SAML 2.0 federation'. Below this, there is a section 'Choose a use case' with 'Common use cases' where 'EC2' is selected. At the bottom, there are buttons for 'Cancel' and 'Next: Permissions'.

Select the below Default IAM Policies:

AmazonS3FullAccess

 Services ▾ Resource Groups ▾ ⭐

Prasad_SMU ▾ Global ▾ Support ▾


1 2 3 4

Filter policies ▾

Showing 1 result

* Required Cancel Previous Next: Tags

Give the Role Name as per your Choice and click on Create Role.

 Services ▾ Resource Groups ▾ ⭐

Prasad_SMU ▾ Global ▾

1 2 3 4

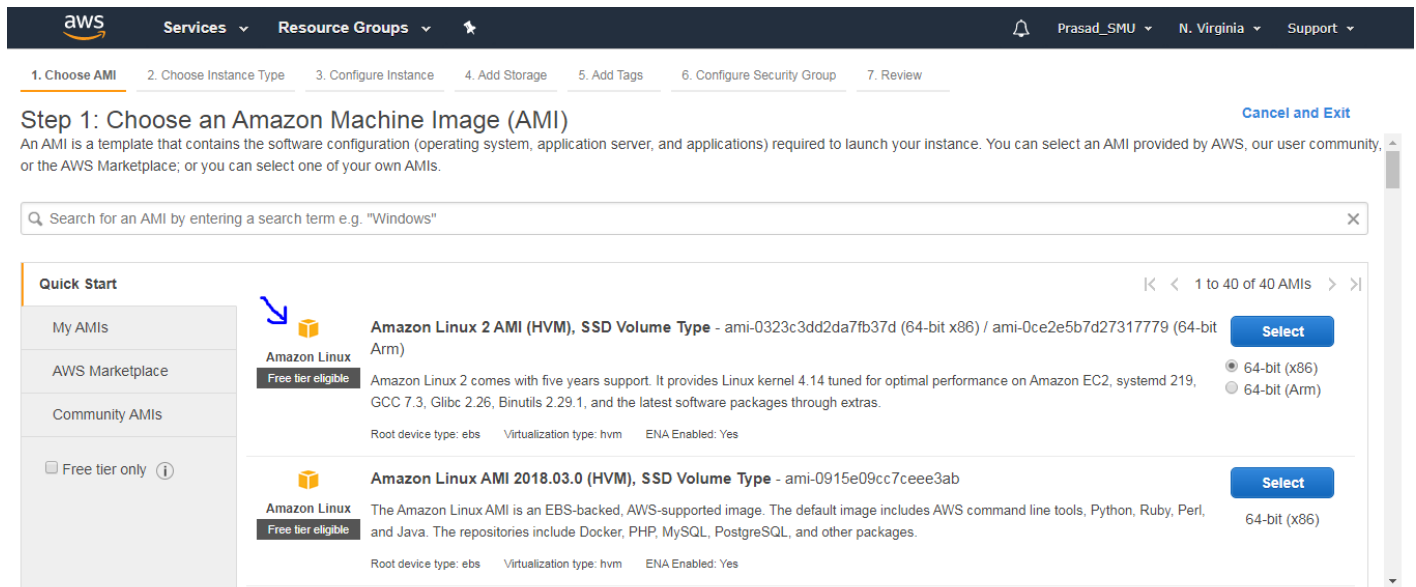
* Required Cancel Previous Create role

Task 2: Launch & Configure EC2 Instance

Login to the AWS Management Console.

Navigate to EC2 Service and click on Launch Instance.

Select the **Amazon Linux AMI**.



Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search for an AMI by entering a search term e.g. "Windows"

Quick Start

- My AMIs
- AWS Marketplace
- Community AMIs
- ☐ Free tier only

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0323c3dd2da7fb37d (64-bit x86) / ami-0ce2e5b7d27317779 (64-bit Arm)

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.

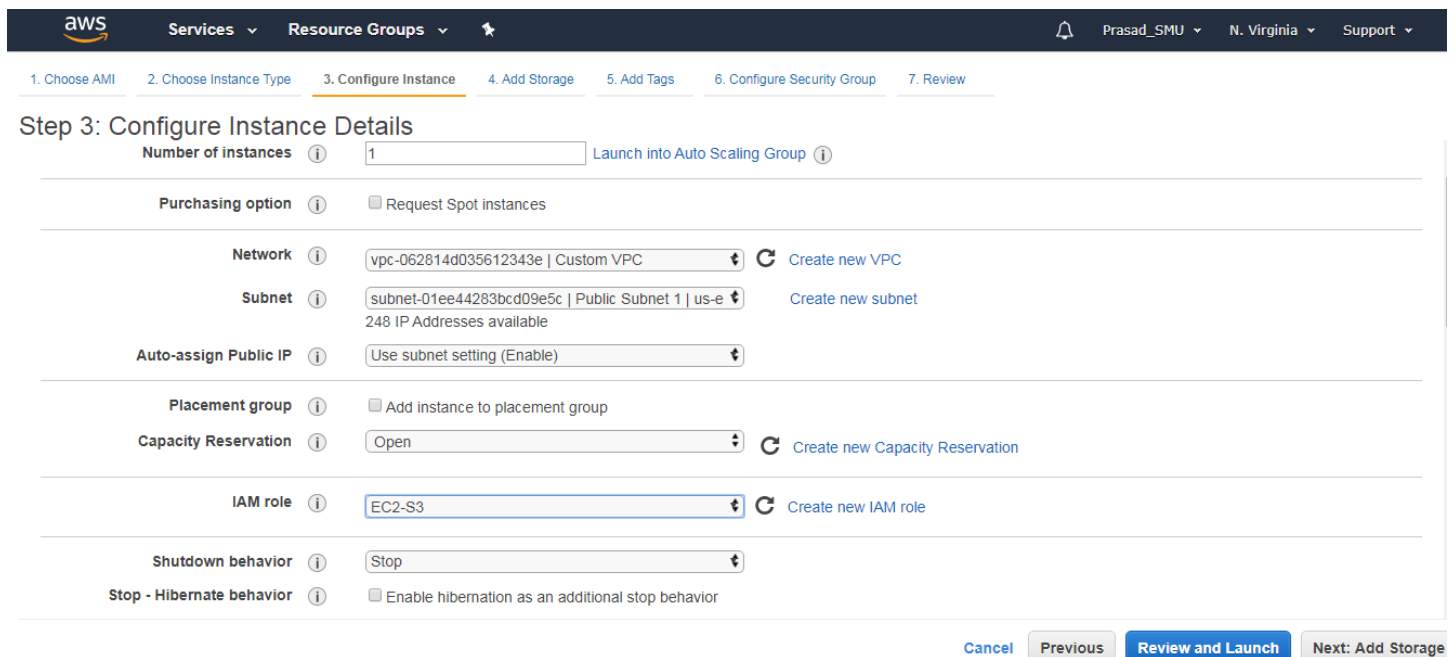
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0915e09cc7ceee3ab

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select the Number of Instances as 1, select the Network as our Custom VPC, Select Subnet as Public Subnet 1 and select the IAM Role which you configured in the Task 1.



Step 3: Configure Instance Details

Number of instances 1 [Launch into Auto Scaling Group](#)

Purchasing option ☐ Request Spot instances

Network vpc-062814d035612343e | Custom VPC [Create new VPC](#)

Subnet subnet-01ee44283bcd09e5c | Public Subnet 1 | us-e [Create new subnet](#)
248 IP Addresses available

Auto-assign Public IP Use subnet setting (Enable)

Placement group ☐ Add instance to placement group

Capacity Reservation Open [Create new Capacity Reservation](#)

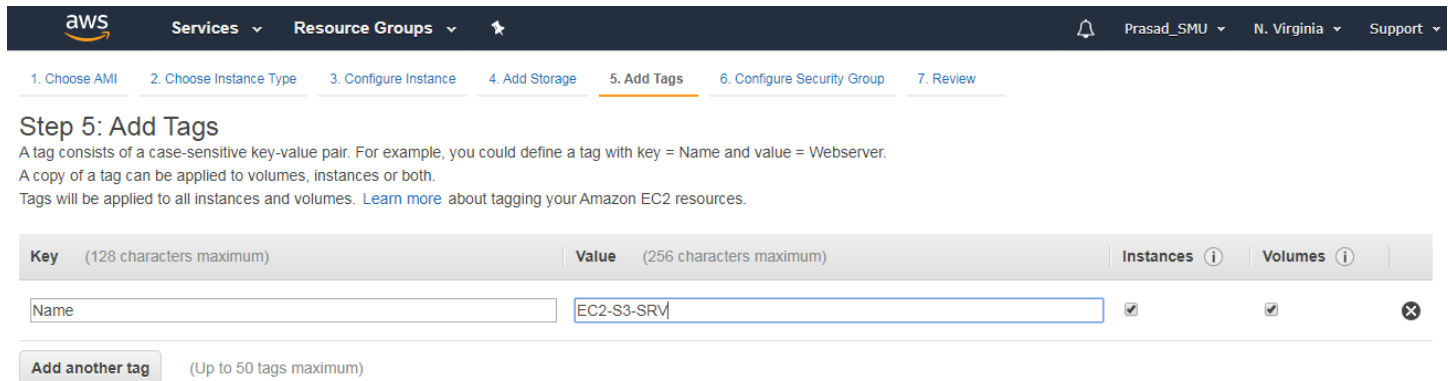
IAM role EC2-S3 [Create new IAM role](#)

Shutdown behavior Stop

Stop - Hibernate behavior ☐ Enable hibernation as an additional stop behavior

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

You can mention Tags as per your choice.



The screenshot shows the 'Step 5: Add Tags' page in the AWS Management Console. The breadcrumb trail at the top indicates the current step is '5. Add Tags'. Below the breadcrumb, there is a description of tags and a link to learn more. The main form has two columns: 'Key' and 'Value'. The 'Key' column has a text input field with 'Name' entered. The 'Value' column has a text input field with 'EC2-S3-SRV' entered. There are checkboxes for 'Instances' and 'Volumes', both of which are checked. At the bottom, there is a button 'Add another tag' and a note '(Up to 50 tags maximum)'.

aws Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 5: Add Tags

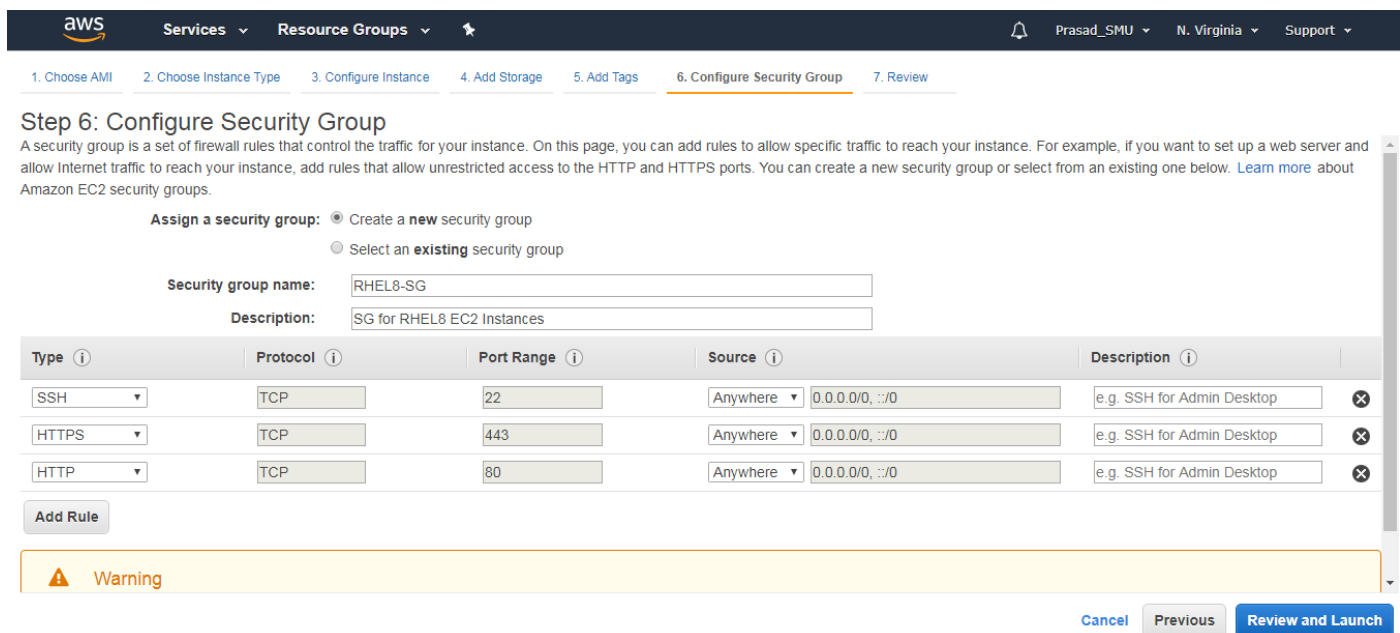
A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key (128 characters maximum)	Value (256 characters maximum)	Instances	Volumes
Name	EC2-S3-SRV	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

[Add another tag](#) (Up to 50 tags maximum)

Click Next: Security Groups.

Create a new Security Group. Give the Name & Discription as per your choice. Allow SSH, HTTPS, HTTP Inbound traffic from Anywhere.



The screenshot shows the 'Step 6: Configure Security Group' page in the AWS Management Console. The breadcrumb trail at the top indicates the current step is '6. Configure Security Group'. Below the breadcrumb, there is a description of security groups and a link to learn more. The main form has a section 'Assign a security group:' with two radio buttons: 'Create a new security group' (selected) and 'Select an existing security group'. Below this, there are text input fields for 'Security group name:' (RHEL8-SG) and 'Description:' (SG for RHEL8 EC2 Instances). Below these fields is a table with columns: 'Type', 'Protocol', 'Port Range', 'Source', and 'Description'. There are three rows of rules: SSH (TCP, 22, Anywhere, 0.0.0.0/0, ::/0), HTTPS (TCP, 443, Anywhere, 0.0.0.0/0, ::/0), and HTTP (TCP, 80, Anywhere, 0.0.0.0/0, ::/0). At the bottom, there is a button 'Add Rule' and a warning box. At the bottom right, there are buttons 'Cancel', 'Previous', and 'Review and Launch'.

aws Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name: RHEL8-SG

Description: SG for RHEL8 EC2 Instances

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Anywhere 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Anywhere 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Anywhere 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

[Add Rule](#)

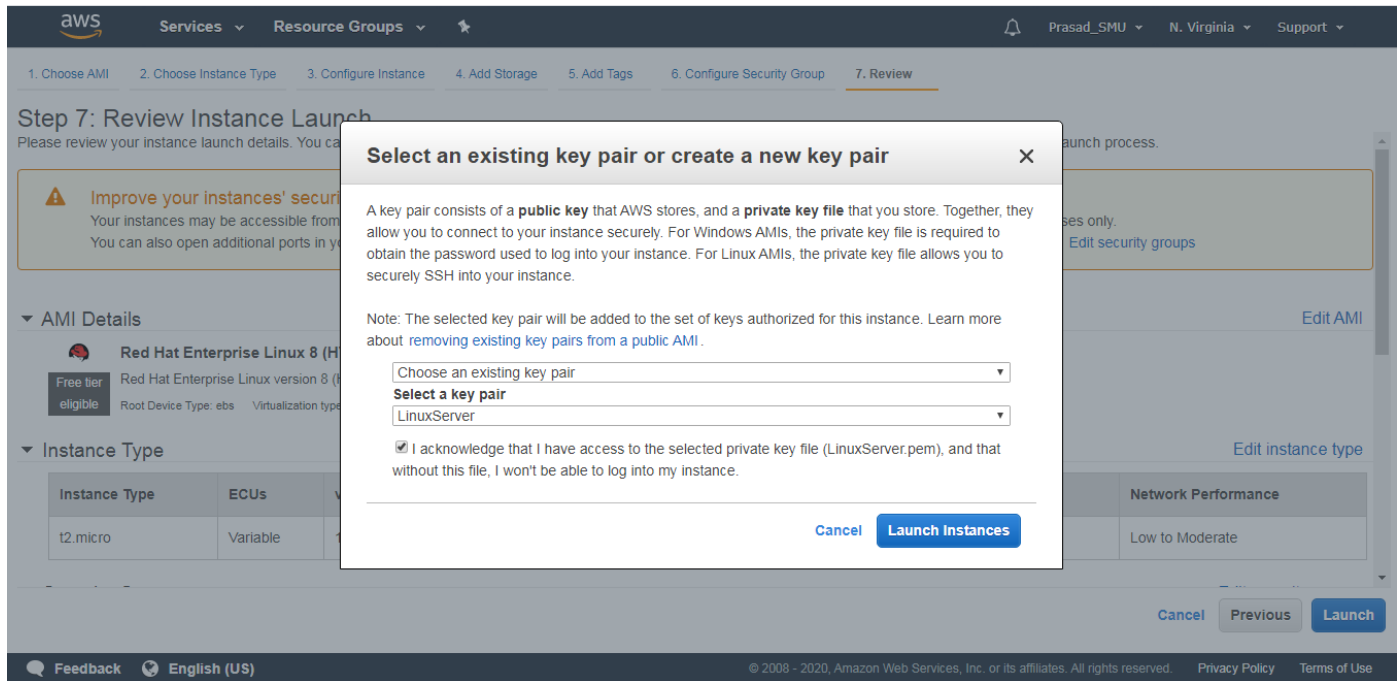
Warning

[Cancel](#) [Previous](#) [Review and Launch](#)

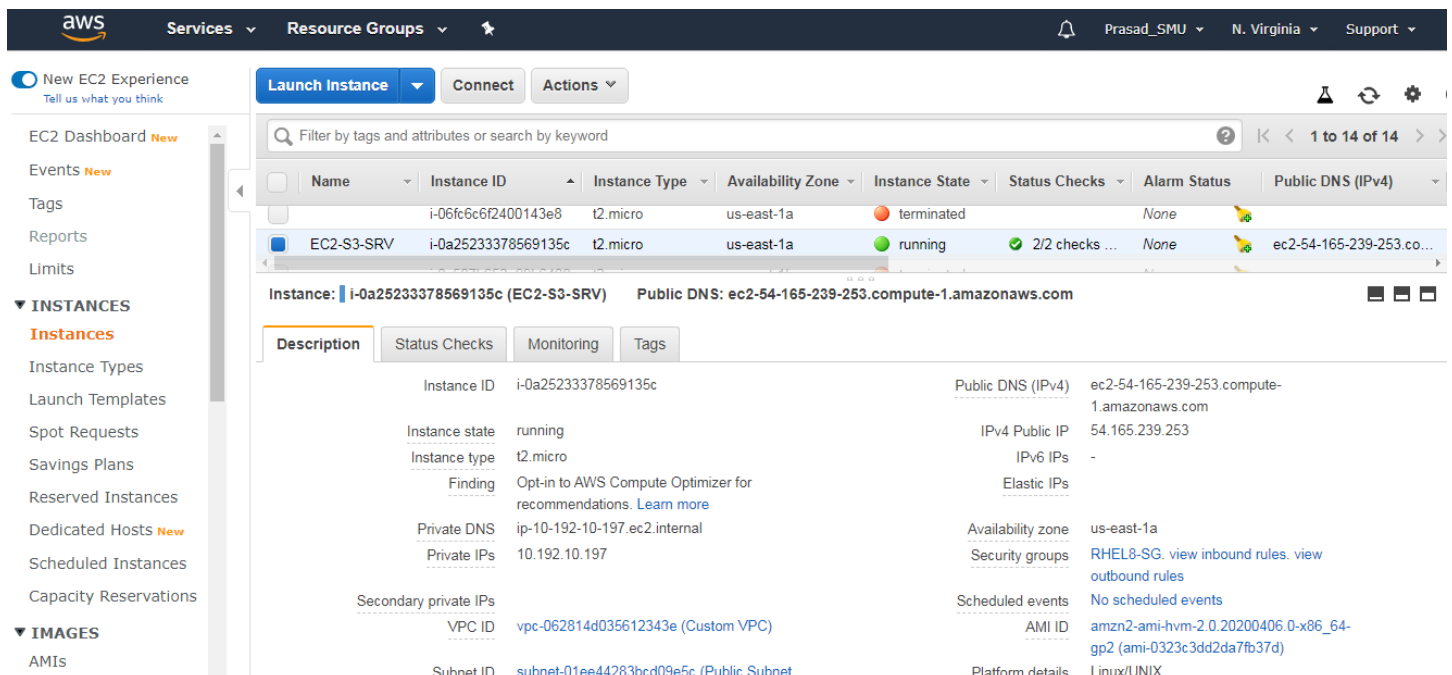
Click on Review and Launch.

Select the existing Key Pair which you've using for previous labs.

Click on Launch Instances.



You can see that the Highlighted Instance has been launched Successfully!!!!



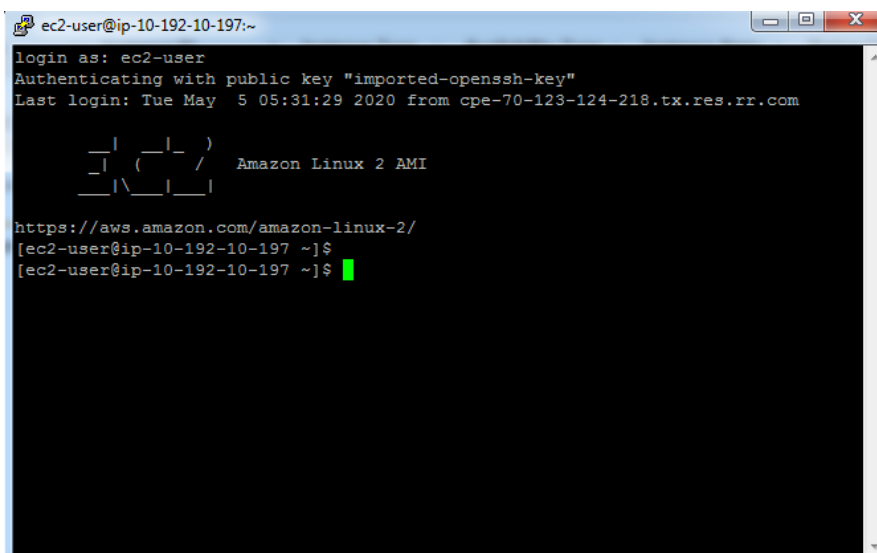
Task 3: Connect to the EC2 Instance

First, using the PuTTYgen, convert the downloaded. pem key to. ppk format.

- Open puttygen.exe
- In the PuTTY Key Generator panel, choose File > Load private key.
- At the bottom of the Load private key panel, click on the drop-down menu that displays *PuTTY Private Key Files (.ppk) and choose All Files**.
- Still in same panel, browse to the directory where you downloaded the .pem file (for example the Downloads directory).
- Select. pem and click Open.
- A PuTTYgen Notice screen should display, indicating that the key was successfully imported. Click OK.
- Click Save private key, then click Yes to save it without a passphrase.
- Give it the filename and click Save.
- Click the X at the top right of the PuTTY Key Generator to close it.

To connect to your Linux EC2 Instance, follow the below steps.

- Open PuTTY software.
- Give the Hostname as the Public IP Address of the Linux EC2 Instance.
- Click on Connections, then click on SSH, then click on Auth.
- Browse the .ppk file and hit Open.
- For certificates validation, click on YES.
- Provide the username as "ec2-user".



```
ec2-user@ip-10-192-10-197:~
login as: ec2-user
Authenticating with public key "imported-openssh-key"
Last login: Tue May  5 05:31:29 2020 from cpe-70-123-124-218.tx.res.rr.com

  _ | _ | _ |
  _ | ( _ | /
  _ | \ _ | _ |
                Amazon Linux 2 AMI

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-192-10-197 ~]$
[ec2-user@ip-10-192-10-197 ~]$
```

Task 4: S3 Bucket Administration using AWS CLI

By default, AWS packages are included in Linux based EC2 Instance.

Command: aws (double tab)

```
[root@ip-10-192-10-197 ~]# aws
aws
aws_completer
[root@ip-10-192-10-197 ~]# aws
```

Switch to ROOT user.

Command: sudo su -

```
[ec2-user@ip-10-192-10-197 ~]$ sudo su -
Last login: Tue May 5 05:31:56 UTC 2020 on pts/0
[root@ip-10-192-10-197 ~]#
```

To list the existing S3 Buckets in your account, run the below command.

Command: aws s3 ls

```
[root@ip-10-192-10-197 ~]# aws s3 ls
2020-04-19 08:18:12 cf-templates-umsgutrdp0mt-us-east-1
2020-05-02 04:43:40 prasadbhavsarlambofgod
2020-05-04 22:41:56 prasadsmu
[root@ip-10-192-10-197 ~]#
```

You can verify the S3 Buckets from AWS Management Console.

Amazon S3

Buckets (3) Copy ARN Empty Delete Create bucket

Buckets are the fundamental container in Amazon S3 for data storage. For others to access the objects in your buckets, you'll need to explicitly grant them permissions. [Learn more](#)

Find bucket by name

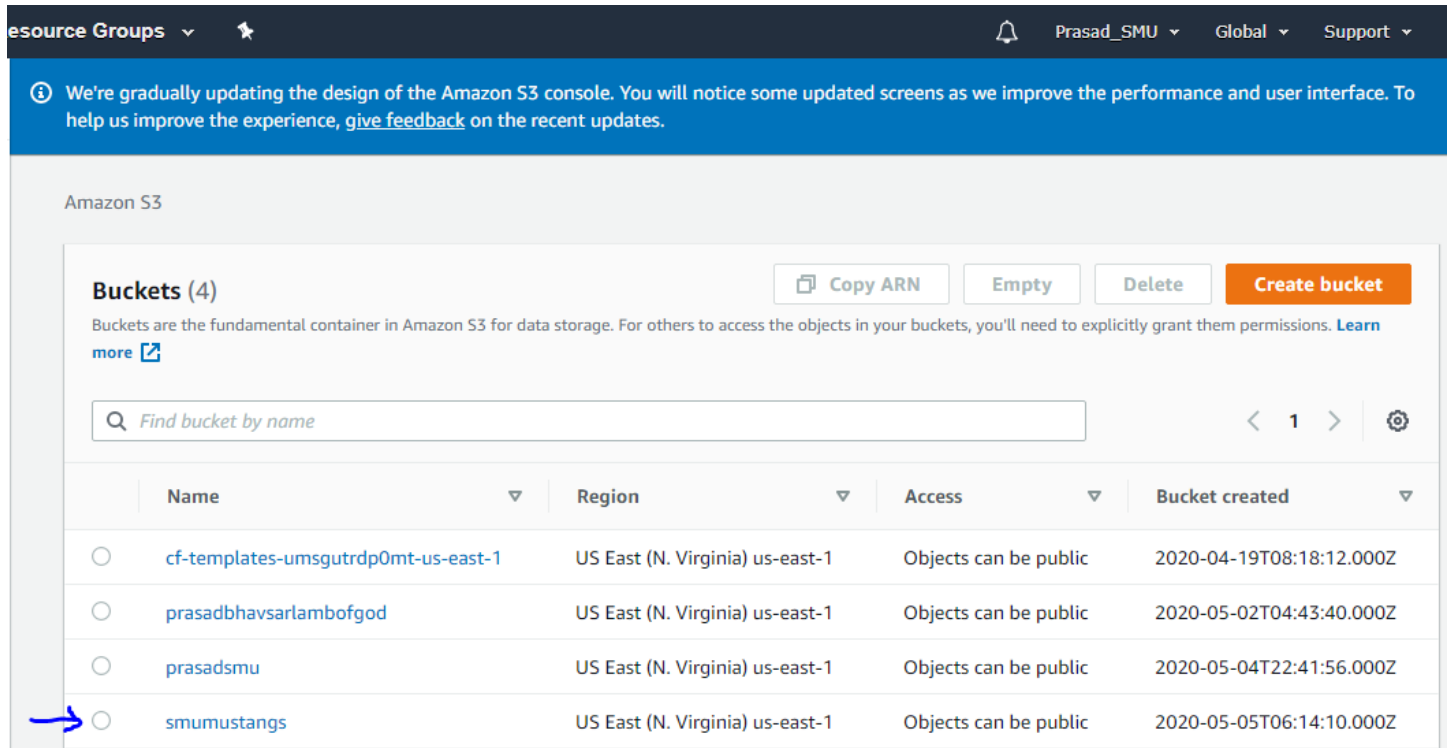
	Name	Region	Access	Bucket created
<input type="radio"/>	cf-templates-umsgutrdp0mt-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	2020-04-19T08:18:12.000Z
<input type="radio"/>	prasadbhavsarlambofgod	US East (N. Virginia) us-east-1	Objects can be public	2020-05-02T04:43:40.000Z
<input type="radio"/>	prasadsmu	US East (N. Virginia) us-east-1	Objects can be public	2020-05-04T22:41:56.000Z

To create a new Bucket, run the below commands.

Command: `aws s3 mb s3://smumustangs`

```
[root@ip-10-192-10-197 ~]# aws s3 mb s3://smumustangs
make_bucket: smumustangs
[root@ip-10-192-10-197 ~]#
```

You can verify the new S3 Bucket from AWS Management Console.



Amazon S3

Buckets (4) Copy ARN Empty Delete Create bucket

Buckets are the fundamental container in Amazon S3 for data storage. For others to access the objects in your buckets, you'll need to explicitly grant them permissions. [Learn more](#)

Find bucket by name

	Name	Region	Access	Bucket created
<input type="radio"/>	cf-templates-umsgutrdp0mt-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	2020-04-19T08:18:12.000Z
<input type="radio"/>	prasadbhavsarlambofgod	US East (N. Virginia) us-east-1	Objects can be public	2020-05-02T04:43:40.000Z
<input type="radio"/>	prasadsmu	US East (N. Virginia) us-east-1	Objects can be public	2020-05-04T22:41:56.000Z
<input checked="" type="radio"/>	smumustangs	US East (N. Virginia) us-east-1	Objects can be public	2020-05-05T06:14:10.000Z

To remove a S3 Bucket, run the below commands.

Command: `aws s3 rb s3://smumustangs`

```
[root@ip-10-192-10-197 ~]# aws s3 rb s3://smumustangs
remove_bucket: smumustangs
[root@ip-10-192-10-197 ~]#
[root@ip-10-192-10-197 ~]# aws s3 ls
2020-04-19 08:18:12 cf-templates-umsgutrdp0mt-us-east-1
2020-05-02 04:43:40 prasadbhavsarlambofgod
2020-05-04 22:41:56 prasadsmu
[root@ip-10-192-10-197 ~]#
```

You can verify the S3 Bucket from AWS Management Console.

Resource Groups

Prasad_SMU Global Support

We're gradually updating the design of the Amazon S3 console. You will notice some updated screens as we improve the performance and user interface. To help us improve the experience, [give feedback](#) on the recent updates.

Amazon S3

Buckets (3)

Buckets are the fundamental container in Amazon S3 for data storage. For others to access the objects in your buckets, you'll need to explicitly grant them permissions. [Learn more](#)

	Name	Region	Access	Bucket created
<input type="radio"/>	cf-templates-umsgutrdp0mt-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	2020-04-19T08:18:12.000Z
<input type="radio"/>	prasadbhavsarlambogod	US East (N. Virginia) us-east-1	Objects can be public	2020-05-02T04:43:40.000Z
<input type="radio"/>	prasadsmu	US East (N. Virginia) us-east-1	Objects can be public	2020-05-04T22:41:56.000Z

Task 5: Mount S3 Buckets to EC2 Instance using AWS CLI

Let's Install necessary packages to mount s3 Bucket.

First, list all the Mount Points.

Command: df -h

```
[root@ip-10-192-10-197 ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        475M   0   475M   0% /dev
tmpfs           492M   0   492M   0% /dev/shm
tmpfs           492M 436K   492M   1% /run
tmpfs           492M   0   492M   0% /sys/fs/cgroup
/dev/xvda1      8.0G  1.7G   6.4G  21% /
tmpfs           99M   0    99M   0% /run/user/1000
[root@ip-10-192-10-197 ~]#
```

To install FUSE packages, run the below set of commands one-by-one.

Command: yum -y install automate fuse fuse-devel gcc-c++ libcurl-devel libxml2-devel make openssl-devel

```
[root@ip-10-192-10-197 ~]# yum -y install automate fuse fuse-devel gcc-c++ libcurl-devel libxml2-devel make openssl-devel
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core | 2.4 kB 00:00:00
amzn2extra-docker | 1.8 kB 00:00:00
epel/x86_64/metalink | 19 kB 00:00:00
```

Command: git clone https://github.com/s3fs-fuse/s3fs-fuse

```
[root@ip-10-192-10-197 ~]# git clone https://github.com/s3fs-fuse/s3fs-fuse
Cloning into 's3fs-fuse'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (37/37), done.
remote: Total 5995 (delta 24), reused 29 (delta 12), pack-reused 5945
```

Commands:

cd s3fs-fuse/

./autogen.sh

```
[root@ip-10-192-10-197 ~]# cd s3fs-fuse/
[root@ip-10-192-10-197 s3fs-fuse]# ./autogen.sh
--- Make commit hash file -----
--- Finished commit hash file ---
--- Start autotools -----
configure.ac:26: installing './config.guess'
configure.ac:26: installing './config.sub'
configure.ac:27: installing './install-sh'
configure.ac:27: installing './missing'
src/Makefile.am: installing './depcomp'
parallel-tests: installing './test-driver'
```

Command: ./configure

```
[root@ip-10-192-10-197 s3fs-fuse]# ./configure
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking target system type... x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
```

Command: make

```
[root@ip-10-192-10-197 s3fs-fuse]# make
make all-recursive
make[1]: Entering directory `/root/s3fs-fuse'
Making all in src
make[2]: Entering directory `/root/s3fs-fuse/src'
g++ -DHAVE_CONFIG_H -I. -I.. -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -I/usr/include/libxml2
  -g -O2 -Wall -D_FILE_OFFSET_BITS=64 -D_FORTIFY_SOURCE=2 -MT s3fs.o -MD -MP -MF .deps/s3fs.Tpo -
c -o s3fs.o s3fs.cpp
mv -f .deps/s3fs.Tpo .deps/s3fs.Po
g++ -DHAVE_CONFIG_H -I. -I.. -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -I/usr/include/libxml2
  -g -O2 -Wall -D_FILE_OFFSET_BITS=64 -D_FORTIFY_SOURCE=2 -MT curl.o -MD -MP -MF .deps/curl.Tpo -
c -o curl.o curl.cpp
```

Command: make install

```
[root@ip-10-192-10-197 s3fs-fuse]# make install
Making install in src
make[1]: Entering directory `/root/s3fs-fuse/src'
make[2]: Entering directory `/root/s3fs-fuse/src'
  /usr/bin/mkdir -p '/usr/local/bin'
  /usr/bin/install -c s3fs '/usr/local/bin'
```

FUSE Package has been successfully installed.

Now to create a Mountpoint run the below command.

Command: mkdir -p /var/s3fs-demo-fs

```
[root@ip-10-192-10-197 s3fs-fuse]# mkdir -p /var/s3fs-demo-fs
[root@ip-10-192-10-197 s3fs-fuse]#
```

Let's now again create a S3 Bucket with the name "smumustangs".

```
[root@ip-10-192-10-197 s3fs-fuse]#
[root@ip-10-192-10-197 s3fs-fuse]# aws s3 mb s3://smumustangs
make_bucket: smumustangs
[root@ip-10-192-10-197 s3fs-fuse]#
```

Mount the S3 Bucket "smumustangs" on the mount point of EC2 Instance "s3fs-demo-fs" by specifying the IAM Role.

Command: s3fs smumustangs /var/s3fs-demo-fs -o iam_role=EC2-S3

```
[root@ip-10-192-10-197 s3fs-fuse]# s3fs smumustangs /var/s3fs-demo-fs -o iam_role=EC2-S3
[root@ip-10-192-10-197 s3fs-fuse]#
[root@ip-10-192-10-197 s3fs-fuse]#
```

To verify the Mount Points, run the below command. You'll notice that the S3 Bucket has been mounted on the EC2 Instance's mount point.

Command: df -h

```
[root@ip-10-192-10-197 s3fs-fuse]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        475M   0  475M   0% /dev
tmpfs           492M   0  492M   0% /dev/shm
tmpfs           492M 440K  492M   1% /run
tmpfs           492M   0  492M   0% /sys/fs/cgroup
/dev/xvda1      8.0G 1.9G  6.2G  24% /
tmpfs           99M   0   99M   0% /run/user/1000
→ s3fs          256T   0  256T   0% /var/s3fs-demo-fs
[root@ip-10-192-10-197 s3fs-fuse]#
[root@ip-10-192-10-197 s3fs-fuse]#
```

Go to the Mount Point.

Command: cd /var/s3fs-demo-fs

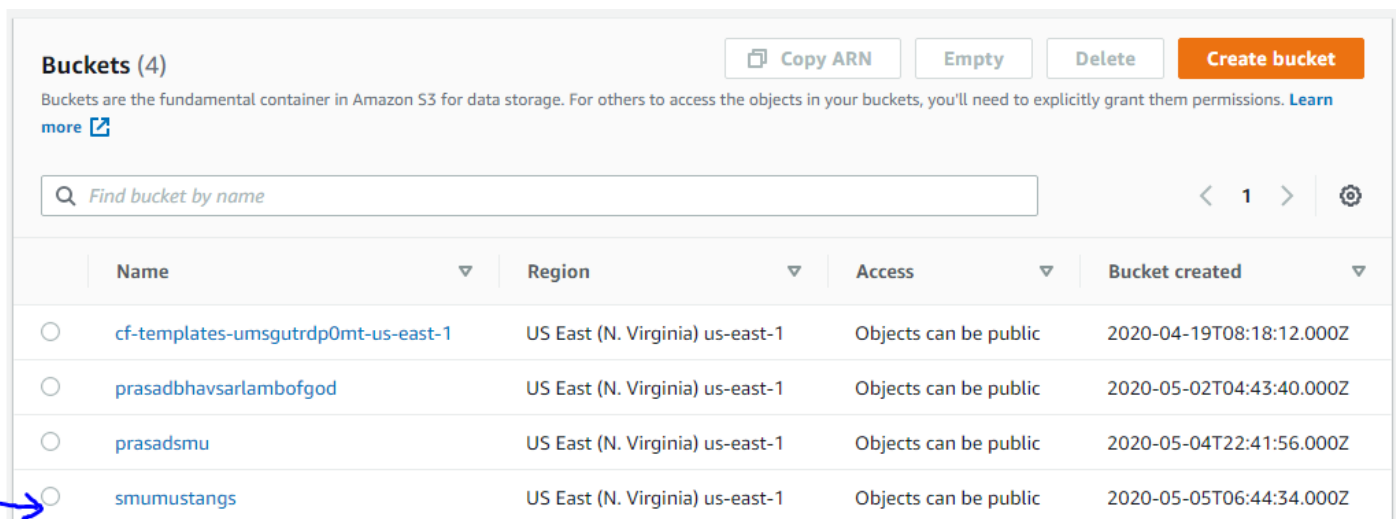
```
[root@ip-10-192-10-197 s3fs-fuse]# cd /var/s3fs-demo-fs
[root@ip-10-192-10-197 s3fs-demo-fs]#
```

Create a dummy file inside the Directory “/var/s3fs-demo-fs”.

Command: touch Prasad_Data

```
[root@ip-10-192-10-197 s3fs-demo-fs]#
[root@ip-10-192-10-197 s3fs-demo-fs]# touch Prasad_Data
```

Now go to AWS Management Console and go to S3 Service. You'll notice the Bucket is created.



Buckets (4) Copy ARN Empty Delete Create bucket

Buckets are the fundamental container in Amazon S3 for data storage. For others to access the objects in your buckets, you'll need to explicitly grant them permissions. [Learn more](#)

Find bucket by name

	Name	Region	Access	Bucket created
<input type="radio"/>	cf-templates-umsgutrdp0mt-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	2020-04-19T08:18:12.000Z
<input type="radio"/>	prasadbhavsarlambofgod	US East (N. Virginia) us-east-1	Objects can be public	2020-05-02T04:43:40.000Z
<input type="radio"/>	prasadsmu	US East (N. Virginia) us-east-1	Objects can be public	2020-05-04T22:41:56.000Z
<input checked="" type="radio"/>	smumustangs	US East (N. Virginia) us-east-1	Objects can be public	2020-05-05T06:44:34.000Z

Get inside the Bucket and you'll notice the dummy file which we created is stored in S3 Bucket as S3 Object.

aws Services Resource Groups

Amazon S3 > smumustangs

smumustangs

Overview Properties Permissions Management Access points

Q Type a prefix and press Enter to search. Press ESC to clear.

Upload Create folder Download Actions

US East (N. Virginia)

Viewing 1 to 1

<input type="checkbox"/>	Name ▼	Last modified ▼	Size ▼	Storage class ▼
<input type="checkbox"/>	Prasad_Data	May 5, 2020 1:54:25 AM GMT-0500	0 B	Standard

Viewing 1 to 1

This completes the Lab on S3 Bucket Administration from EC2 Instance using AWS CLI.

For Questions, contact me one pbhavsar@smu.edu .