# Automation using AWS Systems Manager Service

In this Lab, we are going to discover AWS Systems Manager Service. AWS Systems Manager Service lets you remotely and securely manage the configuration of your EC2 Instances. It also helps you to Automate the management tasks.

When you launch an EC2 Instance, you need to login every time on the EC2 Instance to install the required packages or perform other required tasks. However, you can Automate such activities by using AWS Systems Manager Service. It is required to have Systems Manager agent installed on the Target Instances to perform the required tasks; hence we can conclude that the AWS Systems Manager service is not Agentless like Ansible.

AWS Systems Manager Service includes variety of features to support Automation. In this lab, we are going to focus on AWS Systems Manager-Run Command.

Below is the list of tasks:

**Task 1:** Create IAM Role

**Task 2:** Launch & Configure EC2 Instances (Red Hat Enterprise Linux 8) with SSM Agent

**Task 3:** AWS-Systems Manager: Managed Instances

**Task 4:** AWS-Systems Manager: Run Command

**Task 5:** AWS-Systems Manager Verification

# Task 1: Create IAM Role

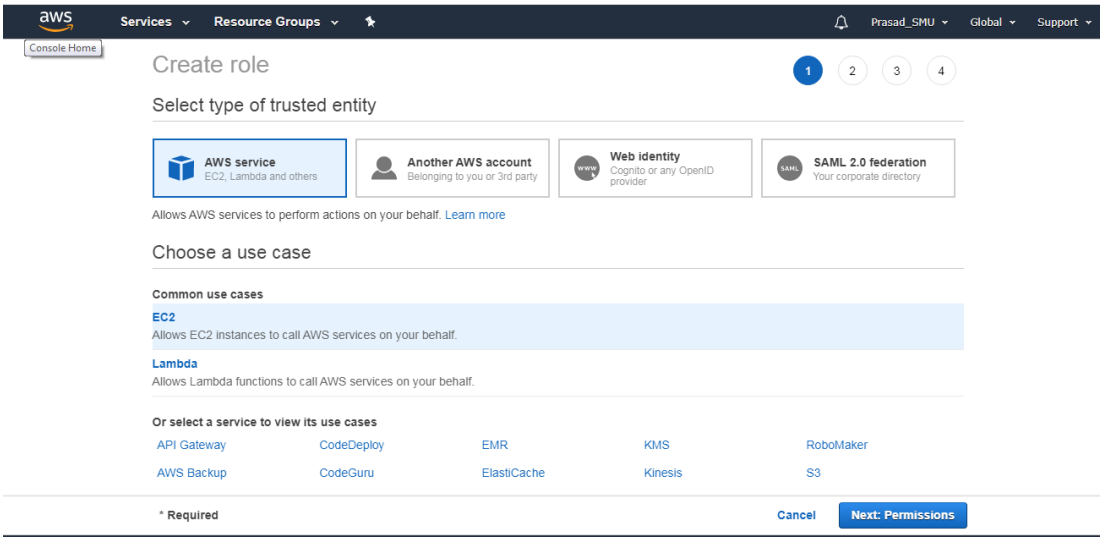
Login to the AWS Management Console.

Navigate to IAM Service and click on Roles.

Click on Create Role.

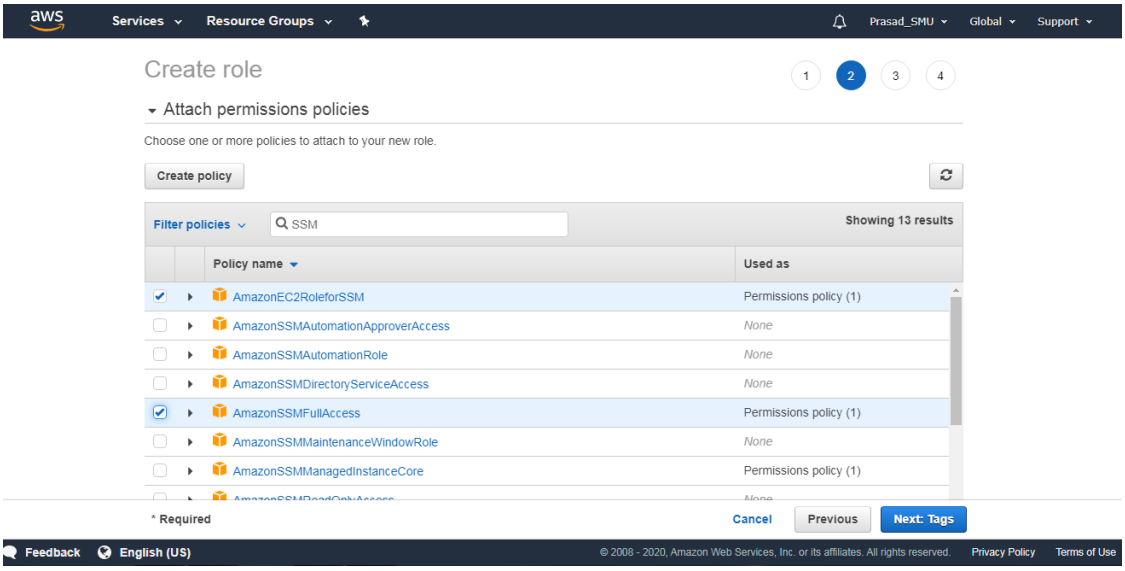
Make sure to select the Use Case as **EC2**.

Click Next: Permissions.

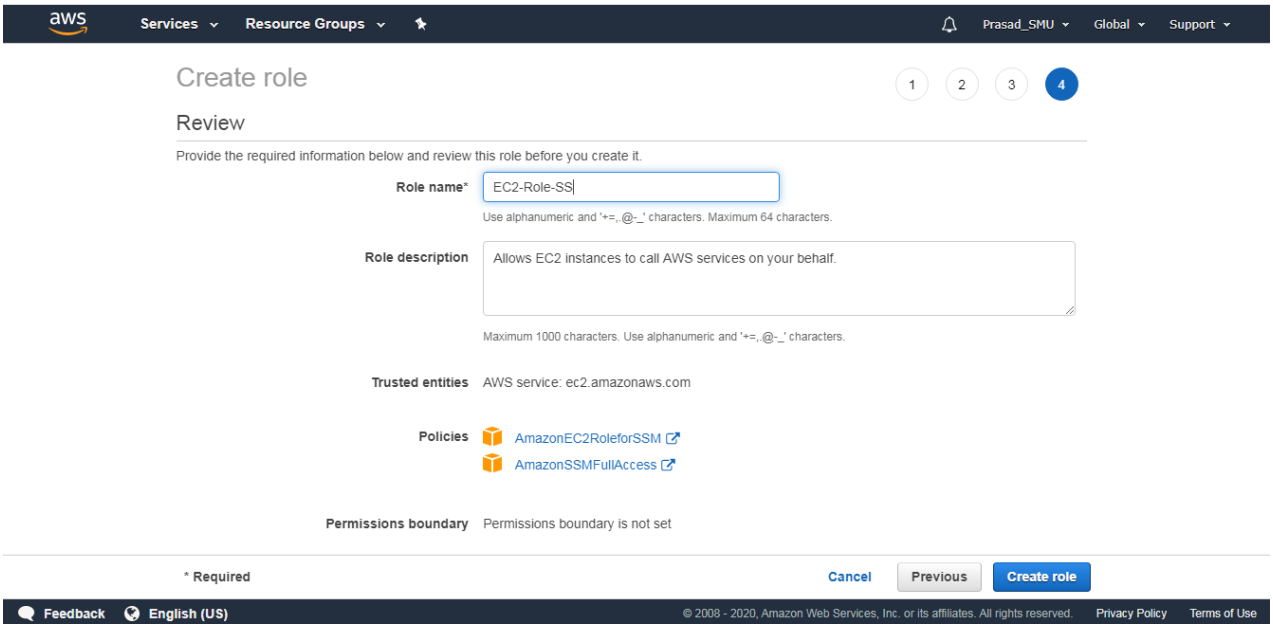



Select the below two Default IAM Policies:

1. **AmazonEC2RoleforSSM**
2. **AmazonSSMFullAccess**



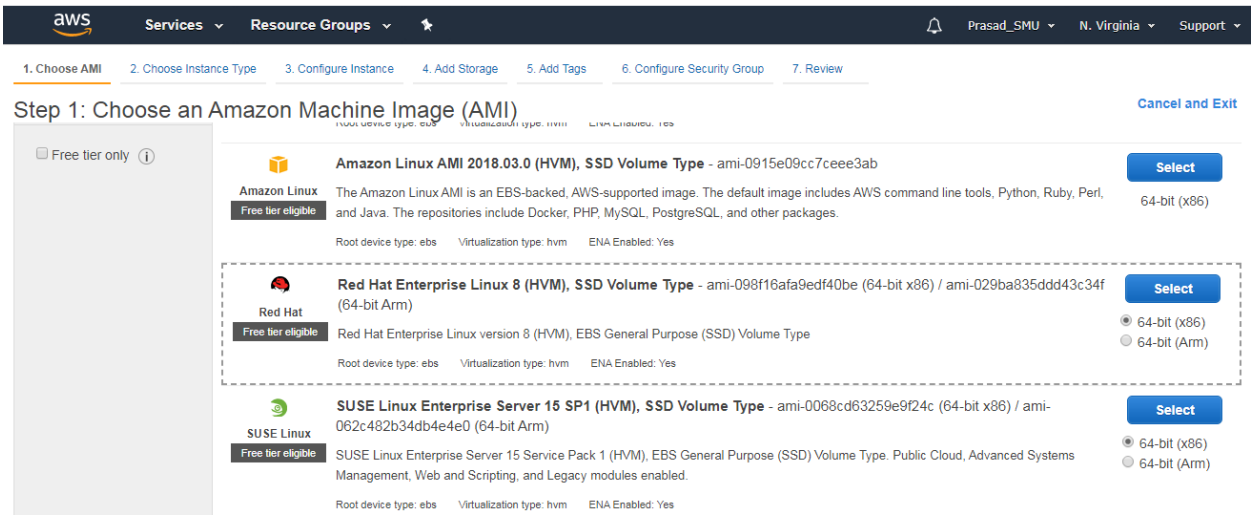
Give the Role Name as per your Choice and click on Create Role.



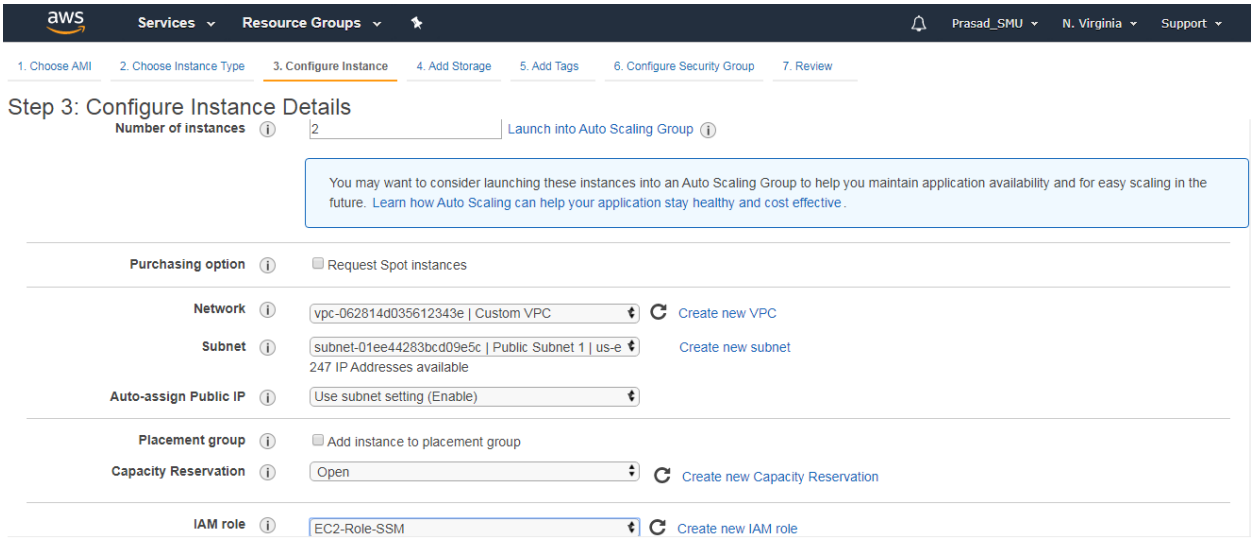

## Task 2: Launch & Configure EC2 Instances (Red Hat Enterprise Linux 8) with SSM Agent

Navigate to EC2 Service and click on Launch Instance.

Select the **Red Hat Enterprise Linux 8 AMI**.



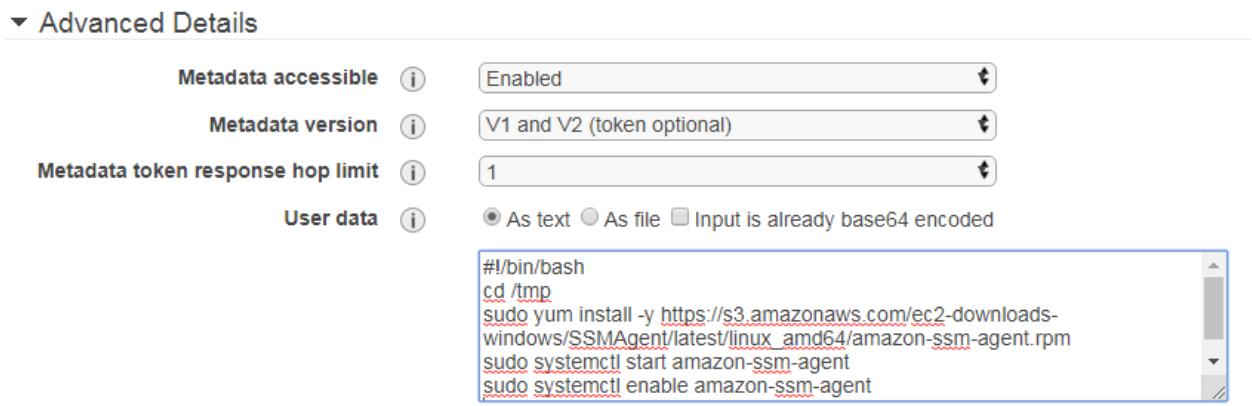

Select the Number of Instances as 2, select the Network as our Custom VPC, Select Subnet as Public Subnet 1 and select the IAM Role which you configured in the Task 1.




Since AWS Systems Manager is AGENTLESS, we need to install Packages for Systems Manager (SSM) to connect with Target Instances.

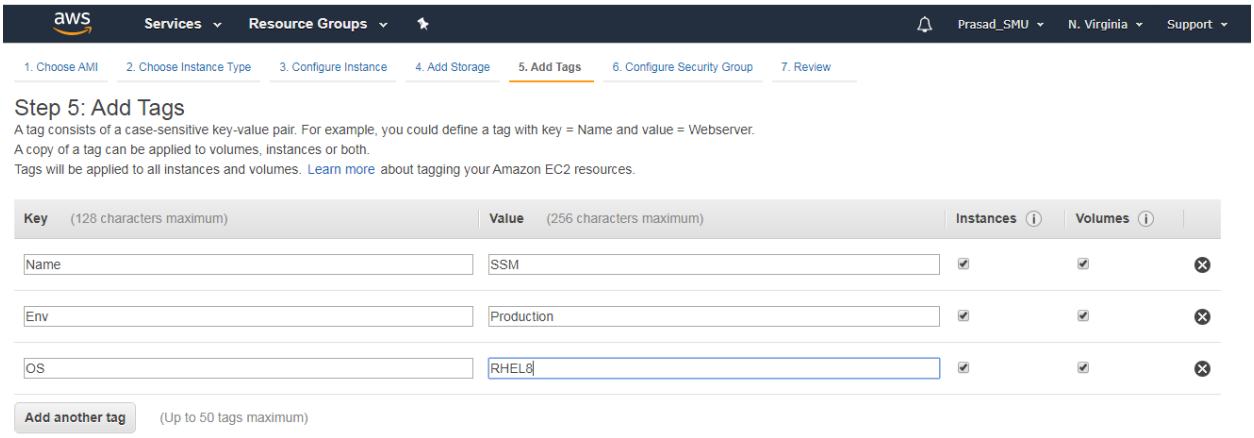
Scroll down on the same page, click on Advanced Details and in User Data field bootstrap the below commands.

I've provided the Commands in text file.



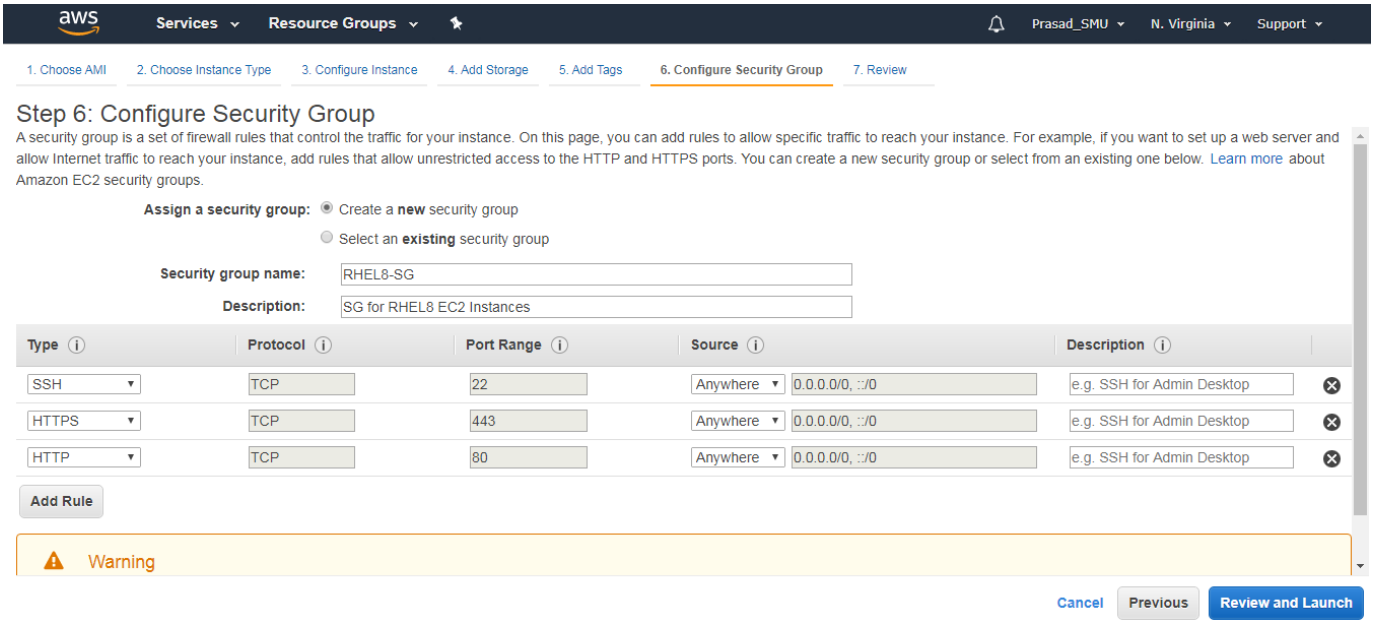

Click on Next.

You can mention Tags as per your choice.




Click Next: Security Groups.

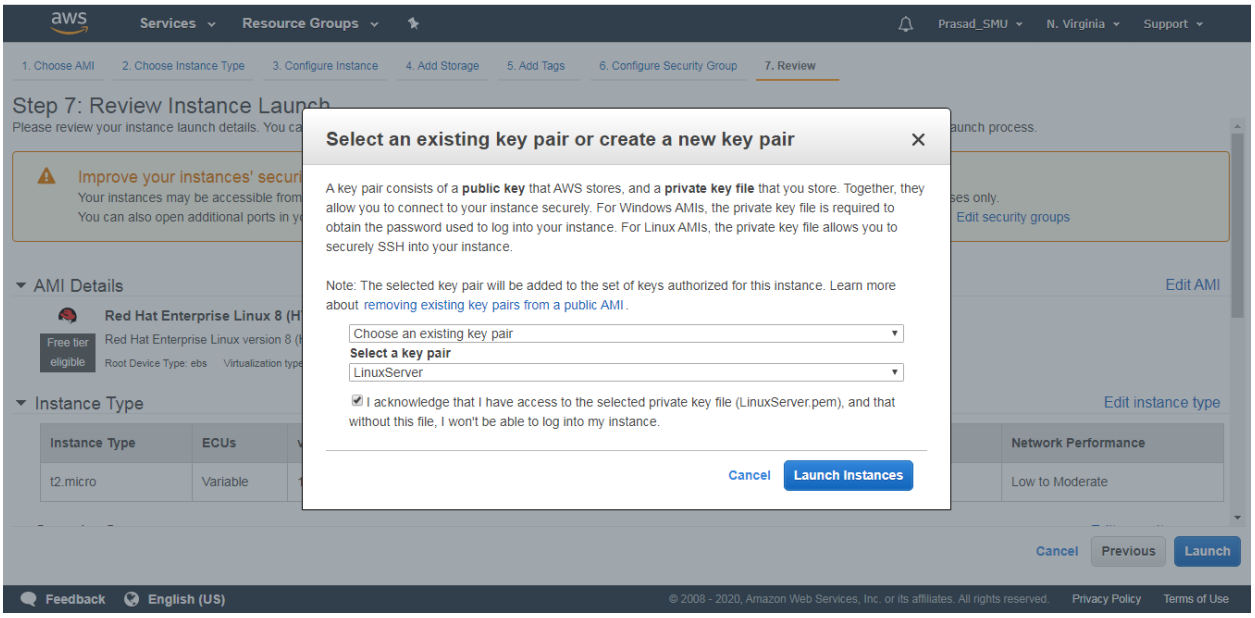
Create a new Security Group. Give the Name & Discription as per your choice. Allow SSH, HTTPS, HTTP Inbound traffic from Anywhere.



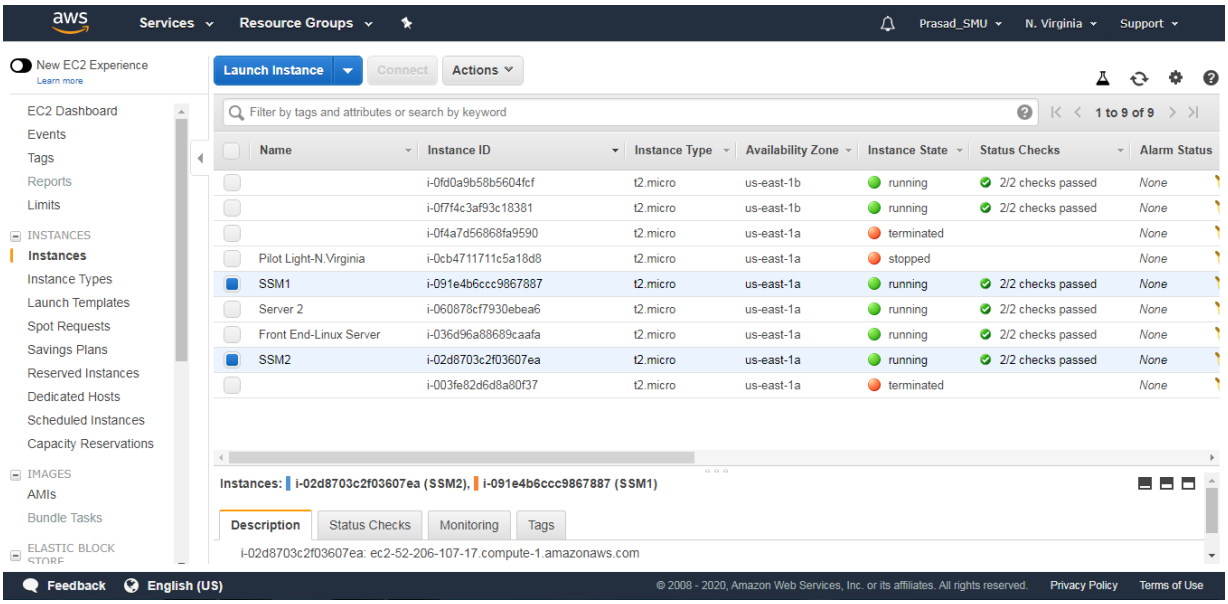

Click on Review and Launch.

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

Click on Launch Instances.




You can see that the Highlighted two Instances have been launched Successfully!!!!!



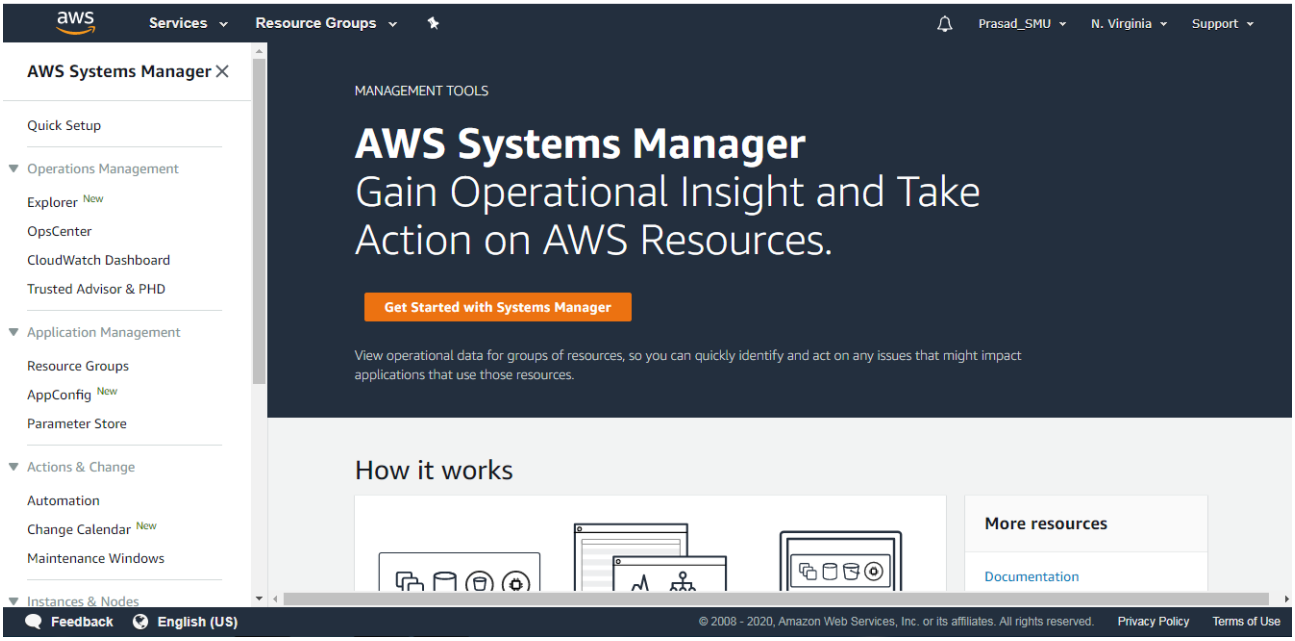

# Task 3: AWS-Systems Manager: Managed Instances

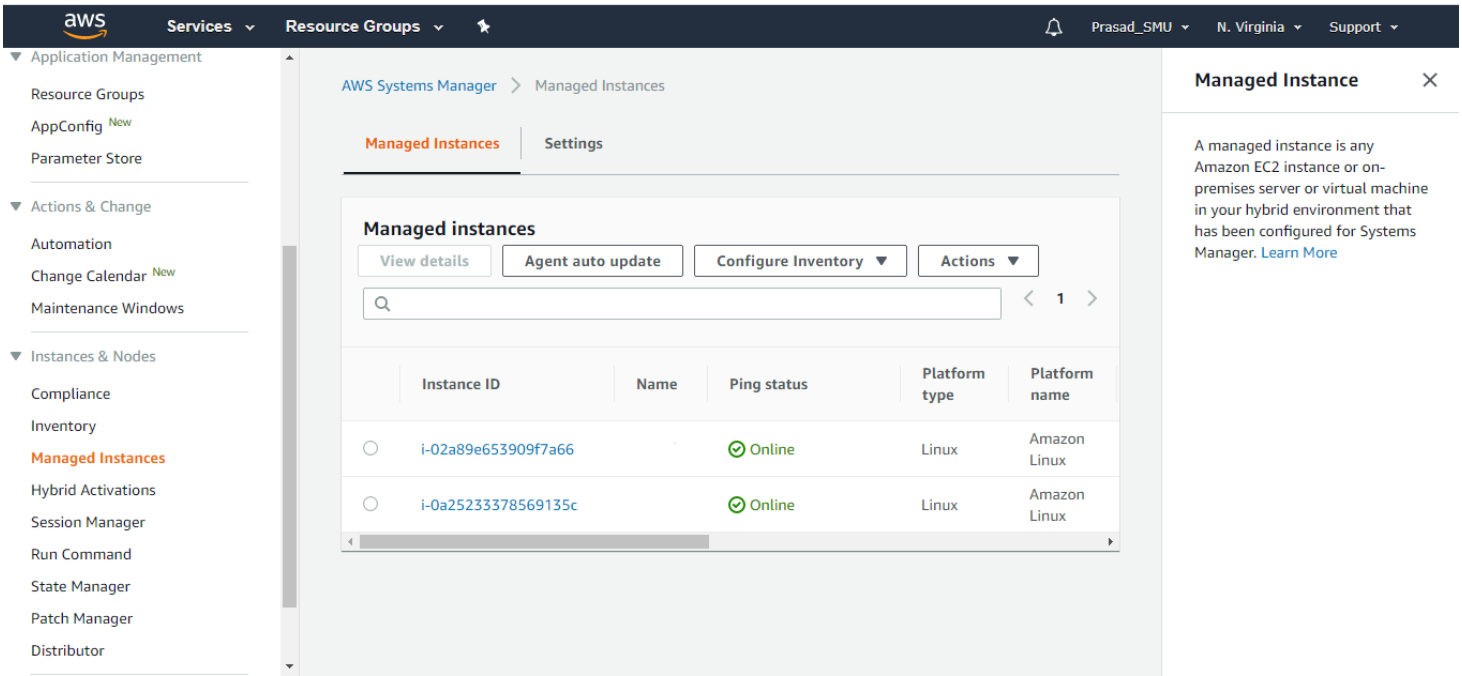
Navigate to AWS Systems Manager Service.

One the left-hand side, click on managed Instances.

You should see, both the Instances which we launched in previous task.

If you do not see any Instance in Managed Instances tab, it means Systems Manager Agent is not Installed on the EC2 Instances.

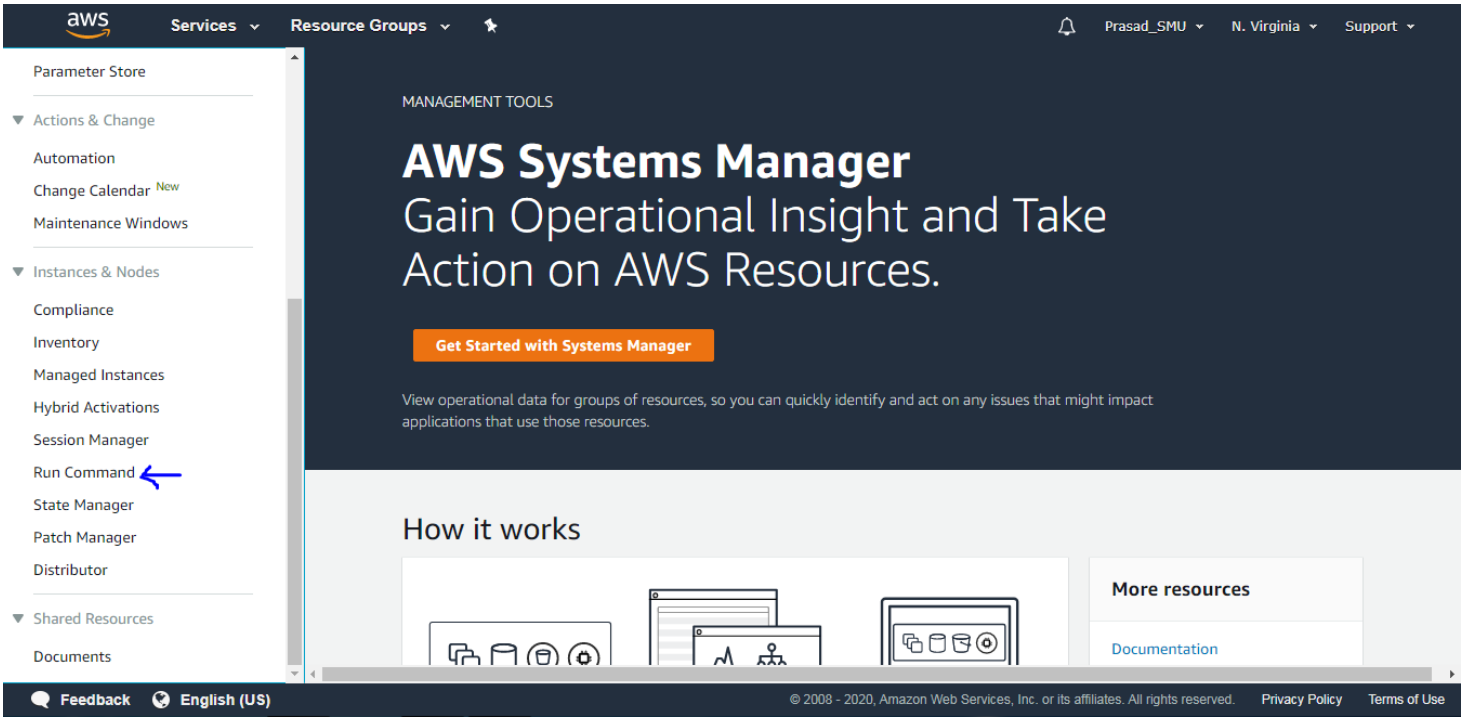


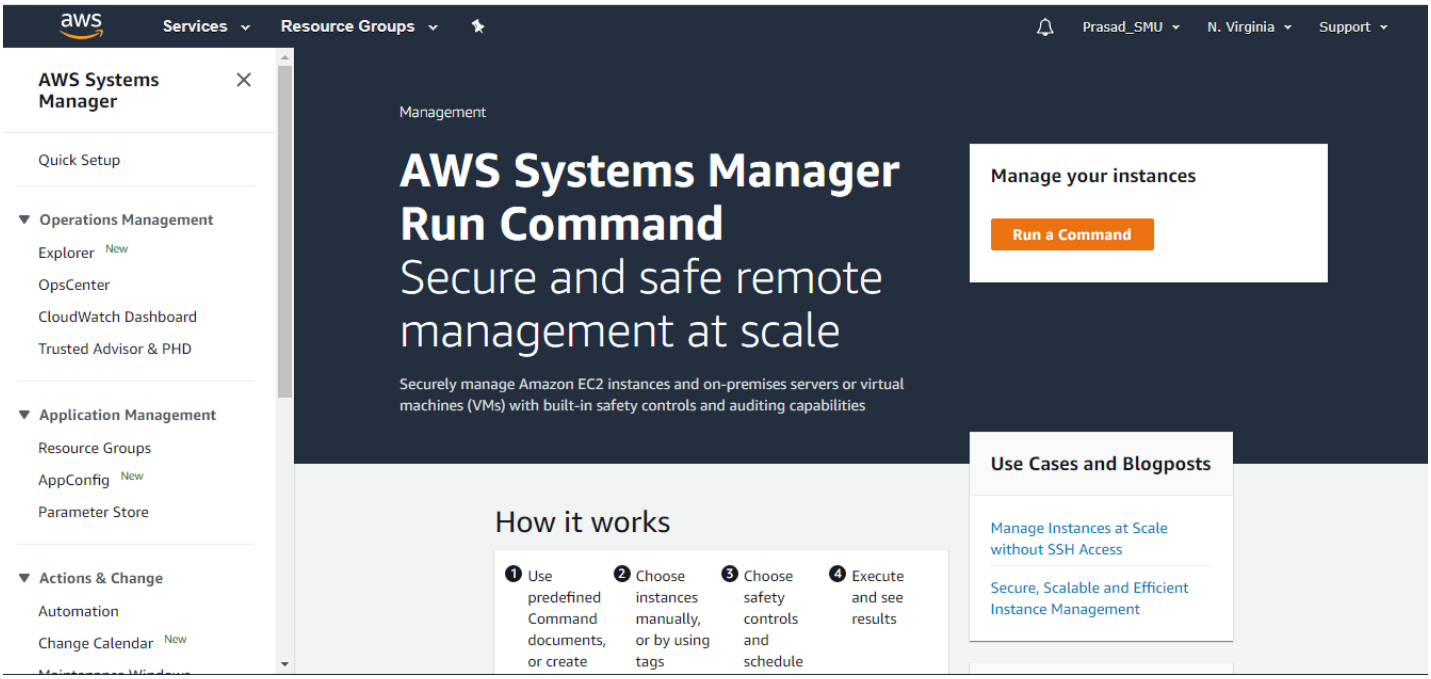

You can also verify the Instance IDs from EC2 Service Dashboard.

# Task 4: AWS-Systems Manager: Run Command

Now under the same Service, on the left-hand side, click on Run Command.

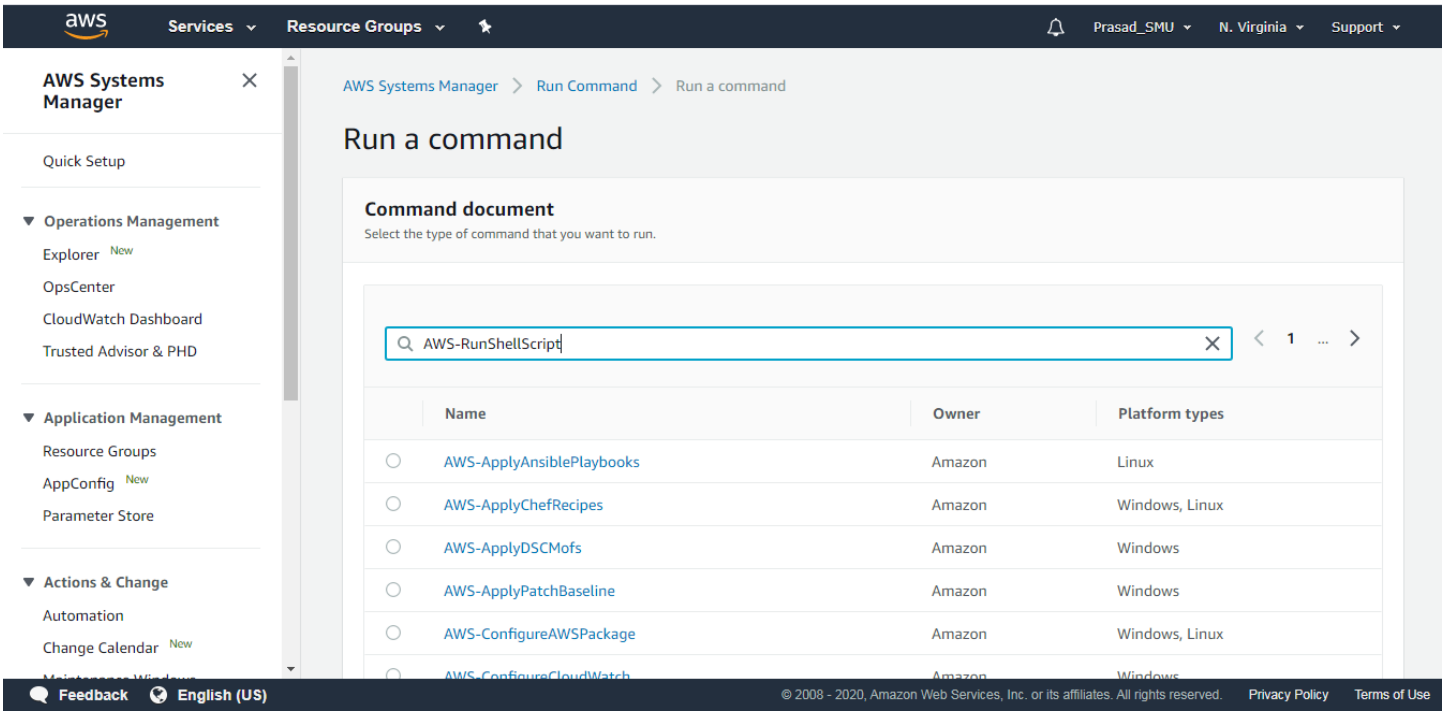



Click on Run a Command.

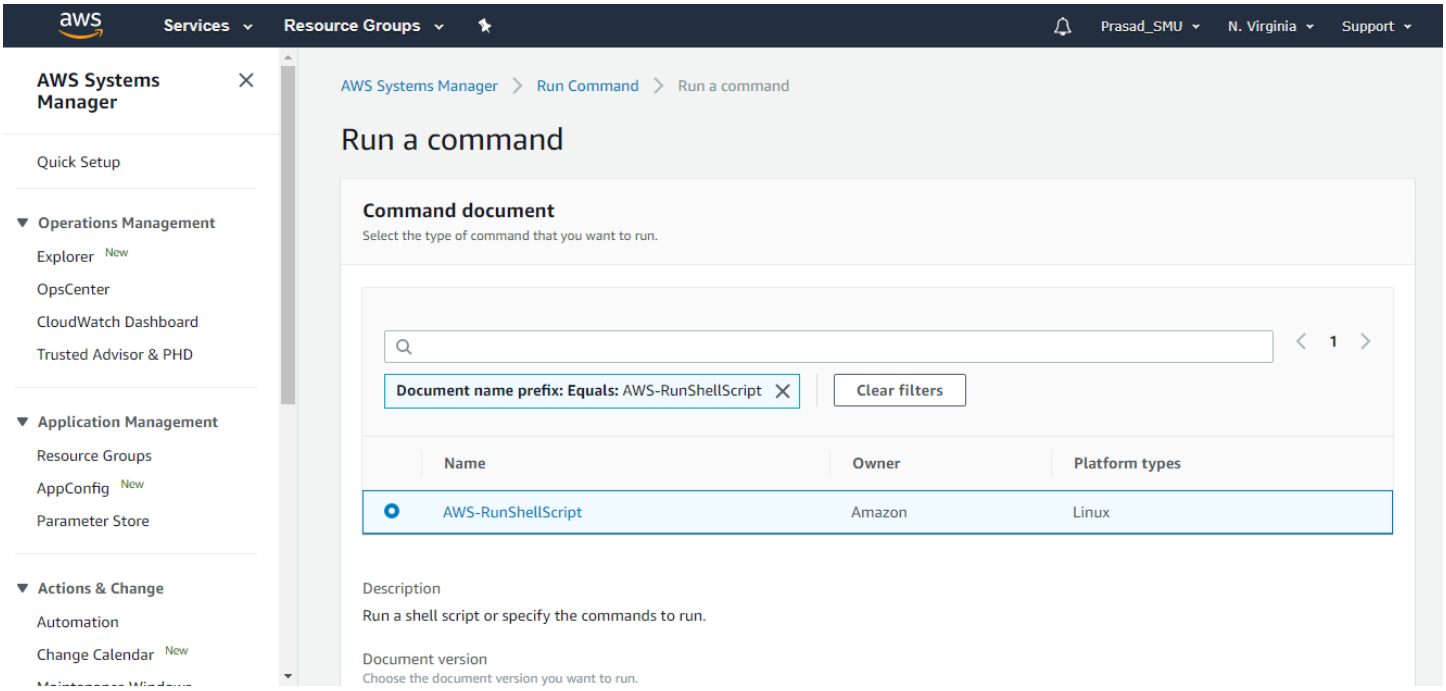


Under Command Document, search for the below AWS Managed Document.

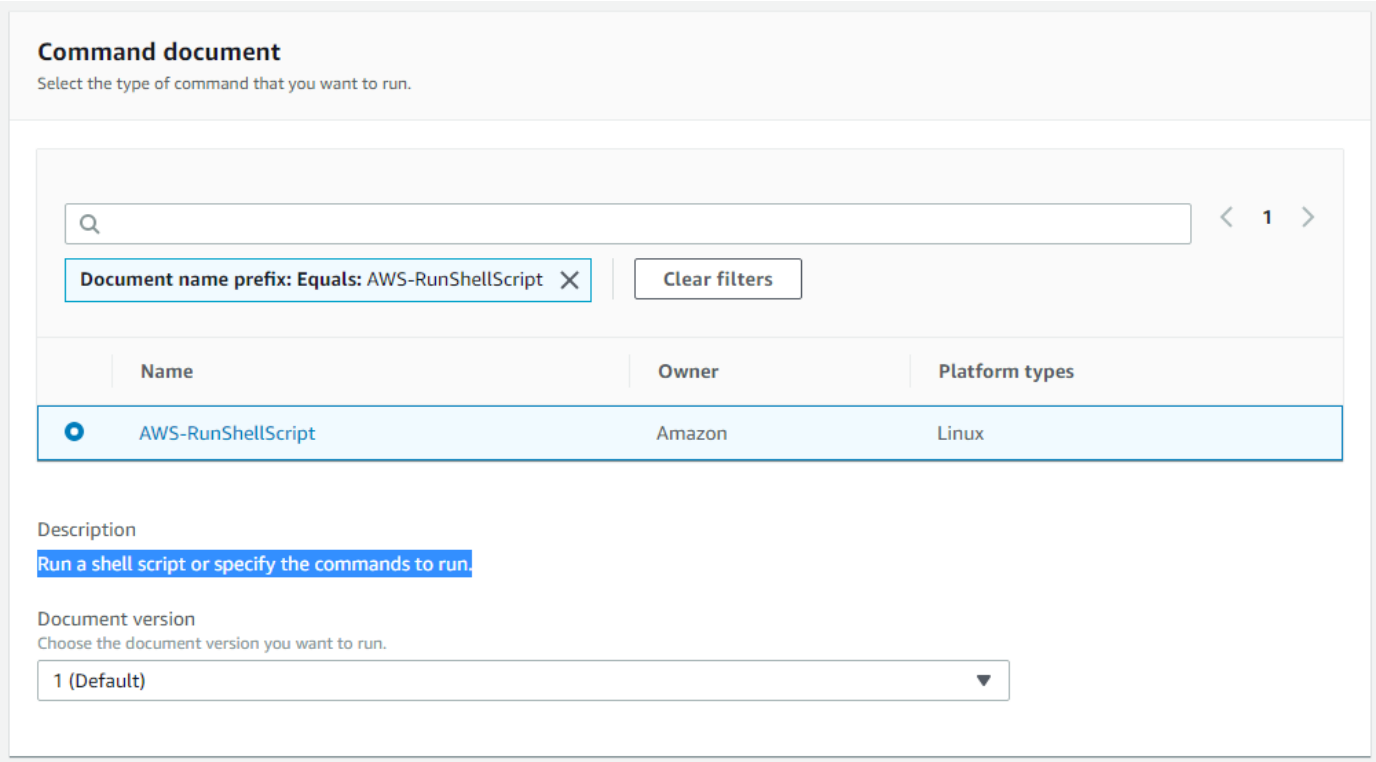
**AWS-RunShellScript**




Select the Command Document.

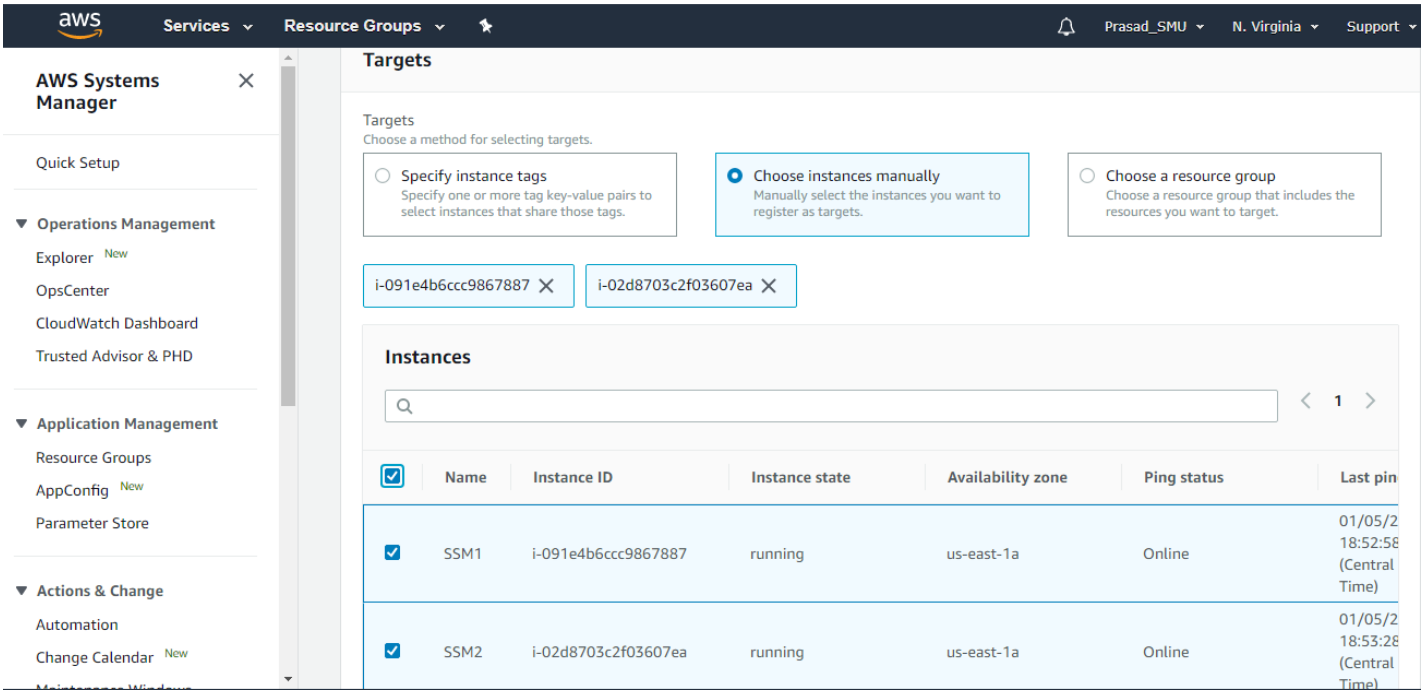


Read the highlighted Description.



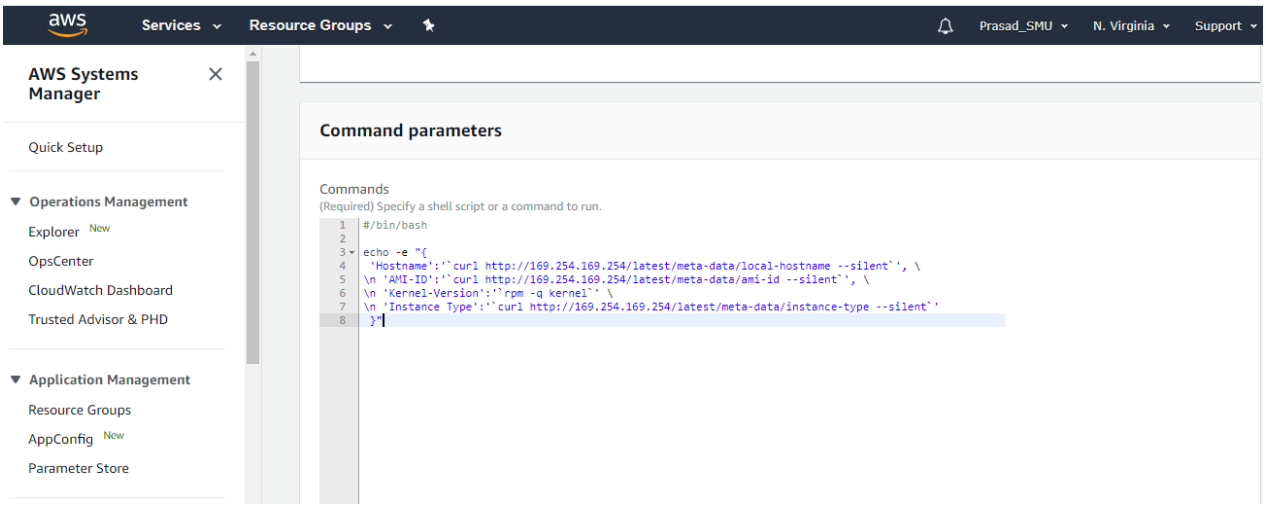

Under Targets, click on Choose Instances Manually and select both the EC2 Instances.

You can also select Instances using Tags.



Type the below Script under Command Parameters. I've provided the Script in Text File.

This script gets the Hostname, AMI-ID, Kernel-Version and Instance-Type of the EC2 Instance.



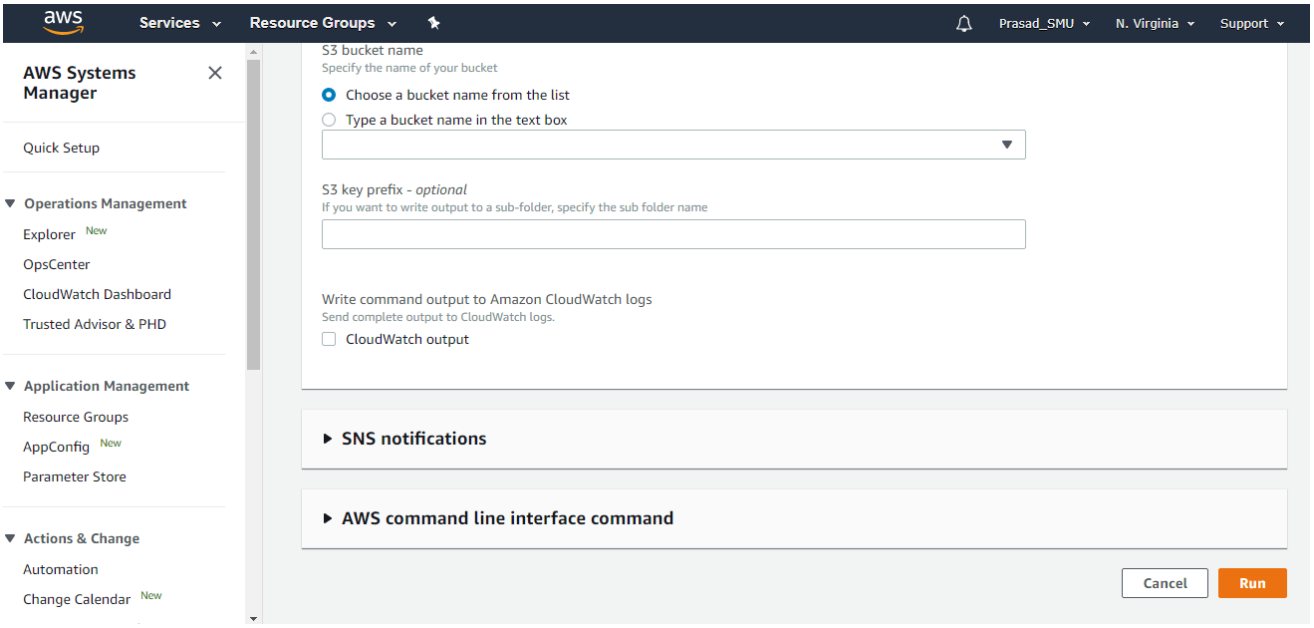

You can also specify the S3 Bucket Name wherein logs of the Run Command will be saved.

Logs in the S3 Bucket will be saved in stdout.txt and stderr.txt format.

Stderr.txt file is quite useful if the Run Command fails.

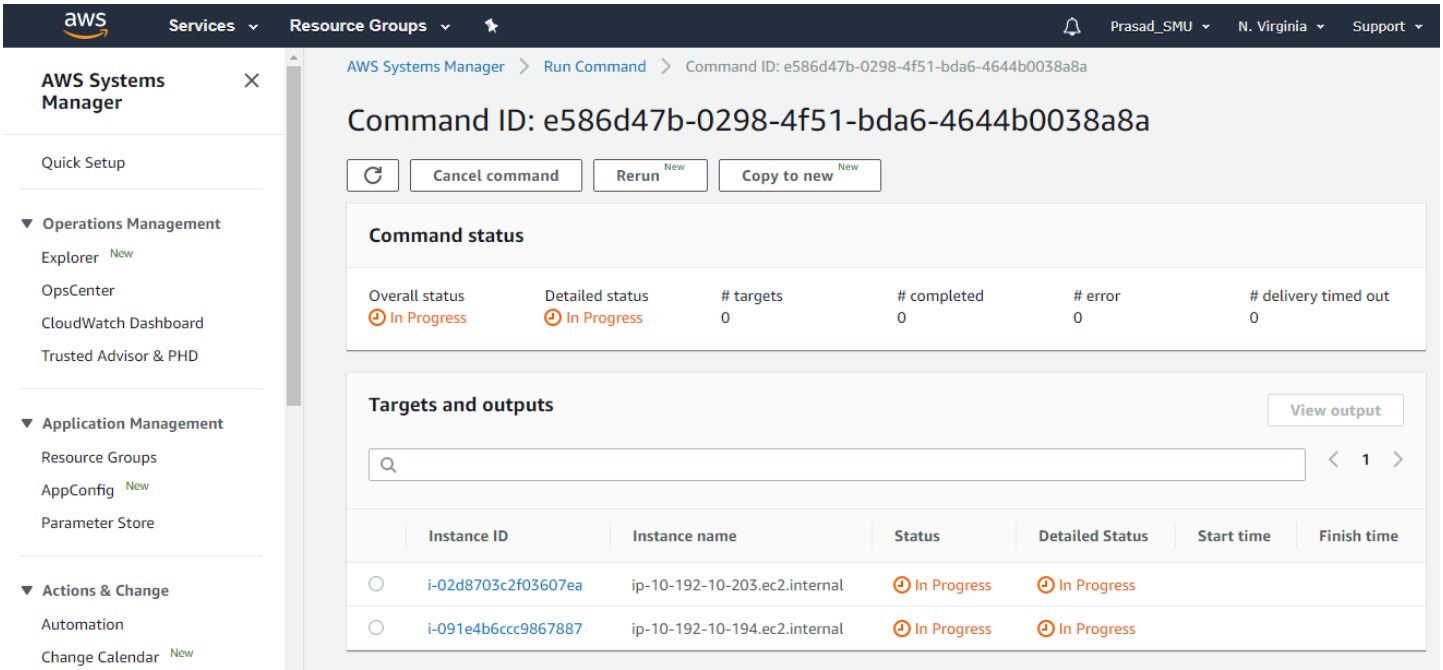
For this lab, we are not going to specify any S3 Bucket as we are running Shell Scripts using Systems Manager-Run Command. We'll specify S3 Bucket Name and observe the Logs while doing Ansible Automation on AWS.

Click RUN.

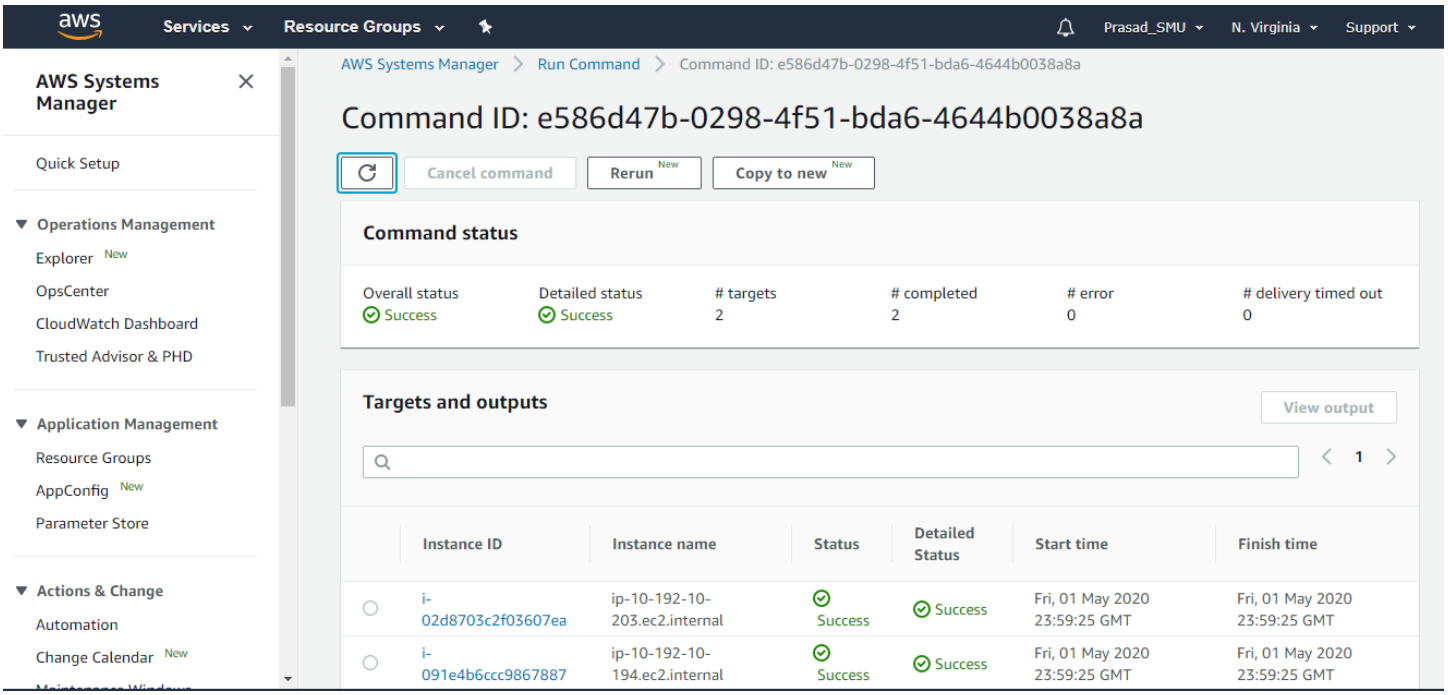


# Task 5: AWS-Systems Manager Verification

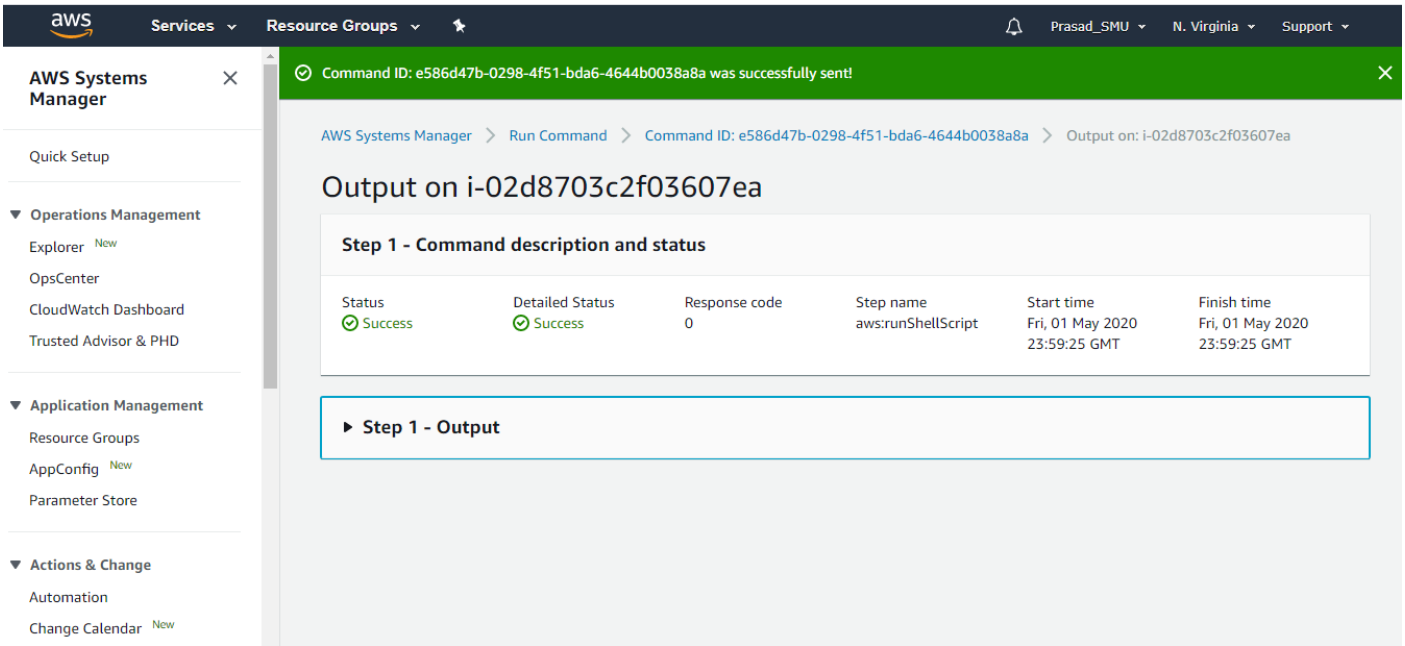
Make a note of Command ID and keep observing Overall Status.




Status changes to **SUCCESS.**

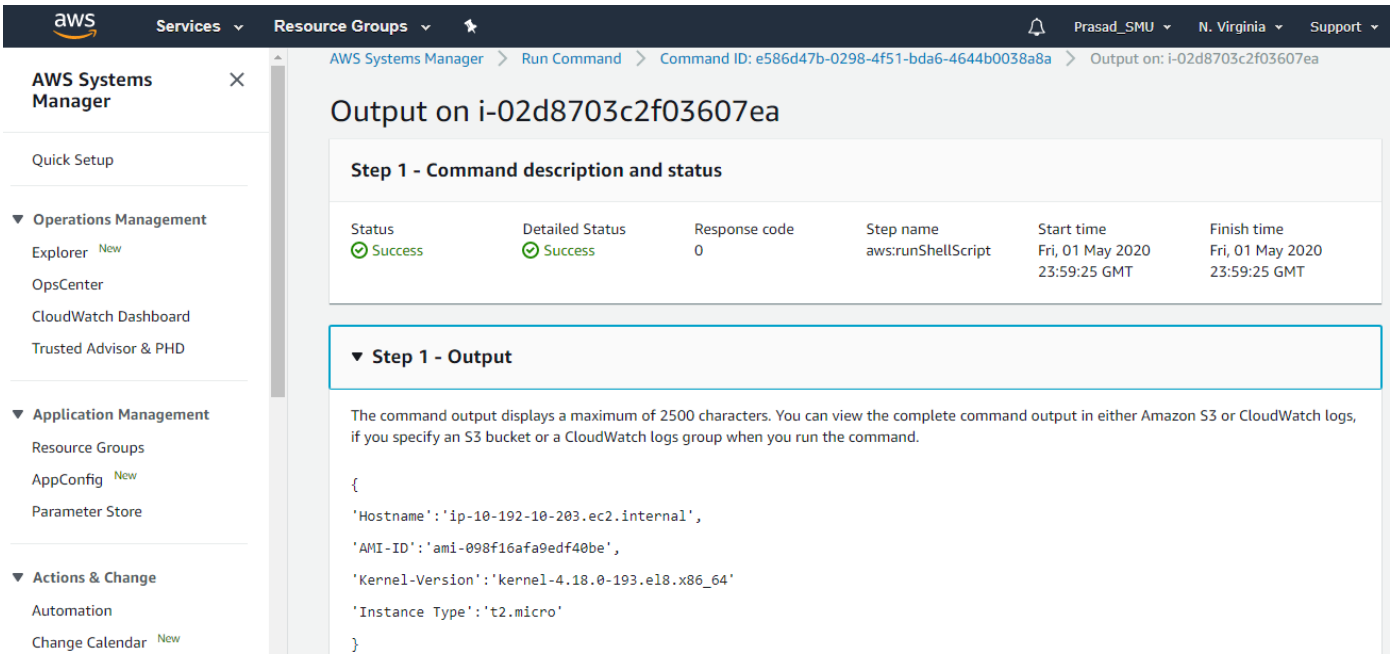


Click on any Instance ID.




Expand **Step 1-Output**.

You'll observe the Hostname, AMI-ID, Kernel-Version, Instance-Type of the EC2 Instance.



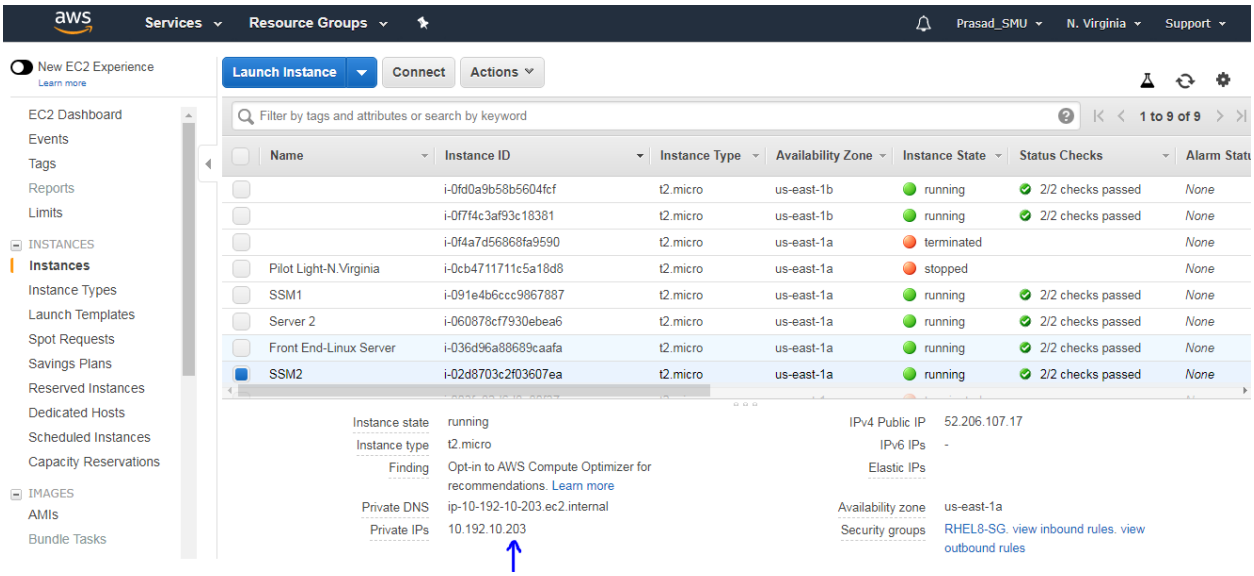
The command output has limitation of 2500 Characters. You can make use of S3 Bucket to store the Output if it is greater than 2500 Characters.

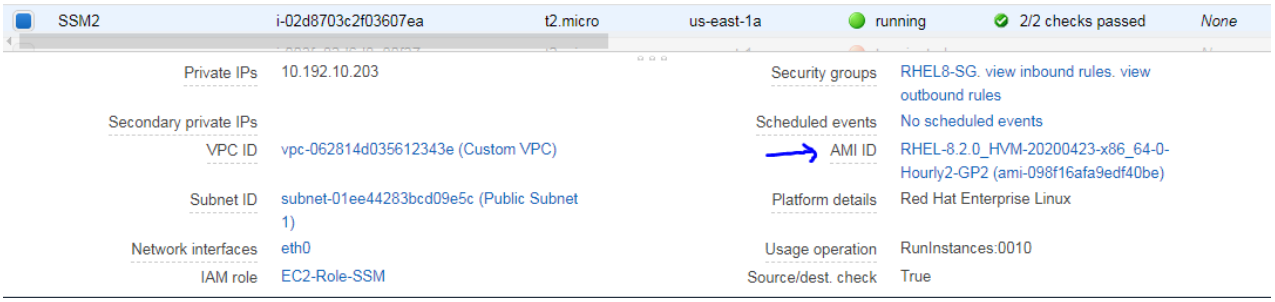
**Step 1 - Output**

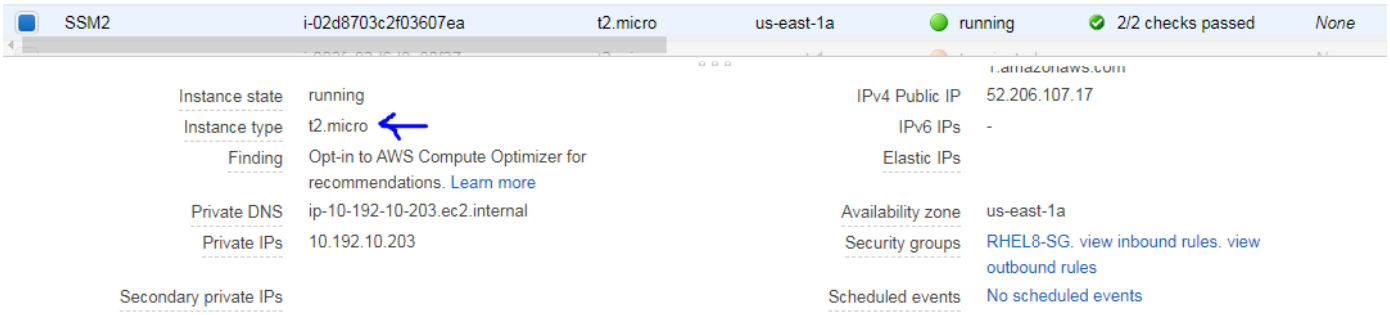
The command output displays a maximum of 2500 characters. You can view the complete command output in either Amazon S3 or CloudWatch logs, if you specify an S3 bucket or a CloudWatch logs group when you run the command.

```
{
'Hostname':'ip-10-192-10-203.ec2.internal',
'AMI-ID':'ami-098f16afa9edf40be',
'Kernel-Version':'kernel-4.18.0-193.el8.x86_64'
'Instance Type':'t2.micro'
}
```

Now verify the Hostname, AMI-ID, Kernel-Version, Instance-Type of the EC2 Instance from EC2 Service Dashboard.

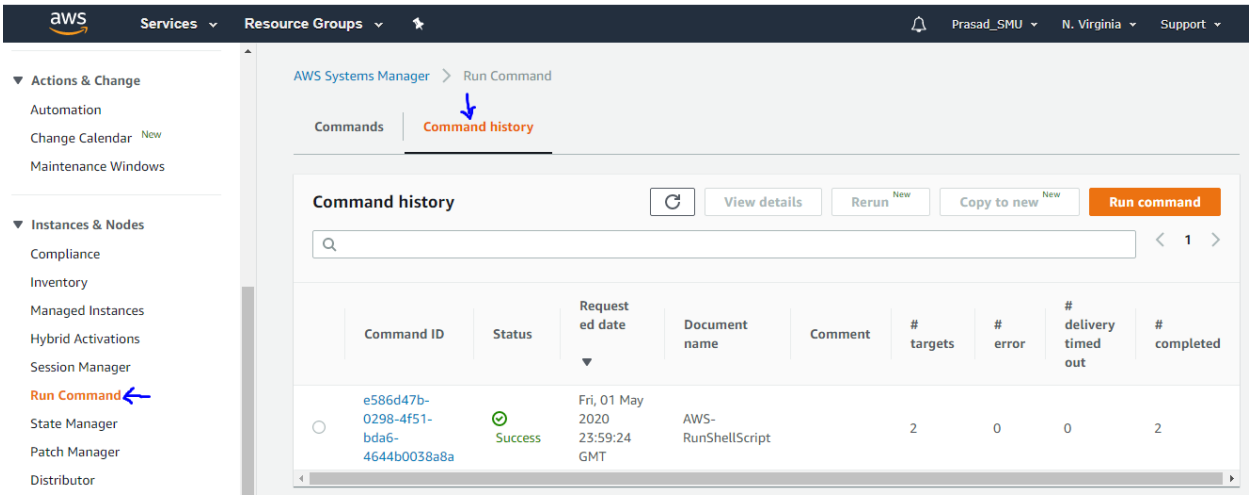


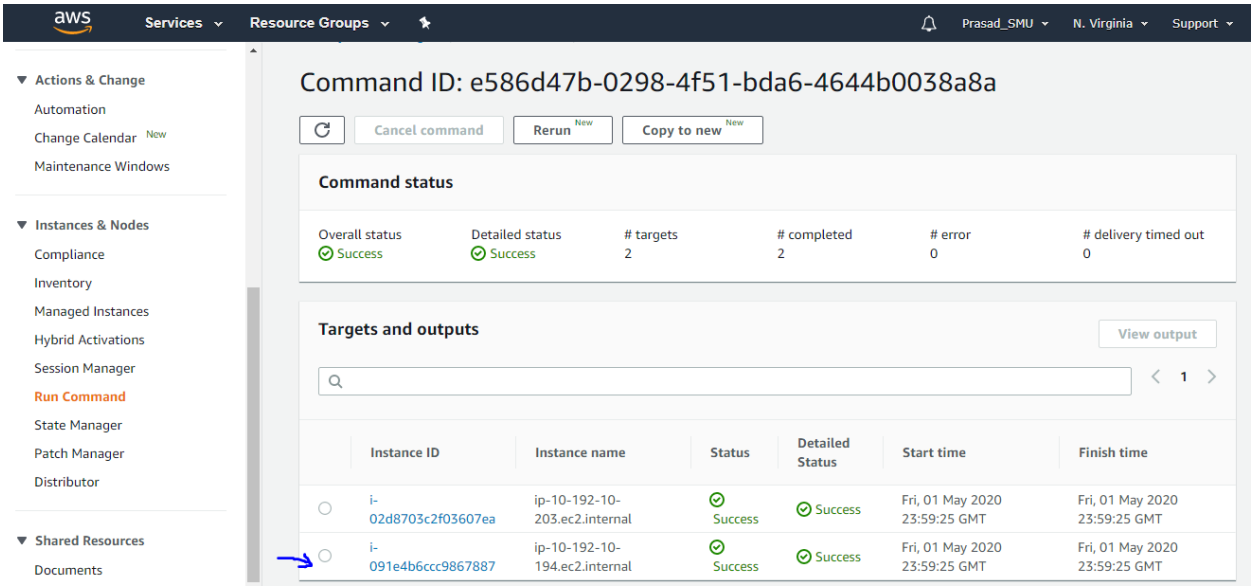



Now navigate to AWS-Systems Manager Service again and click on Run Command.

Click on Command History.

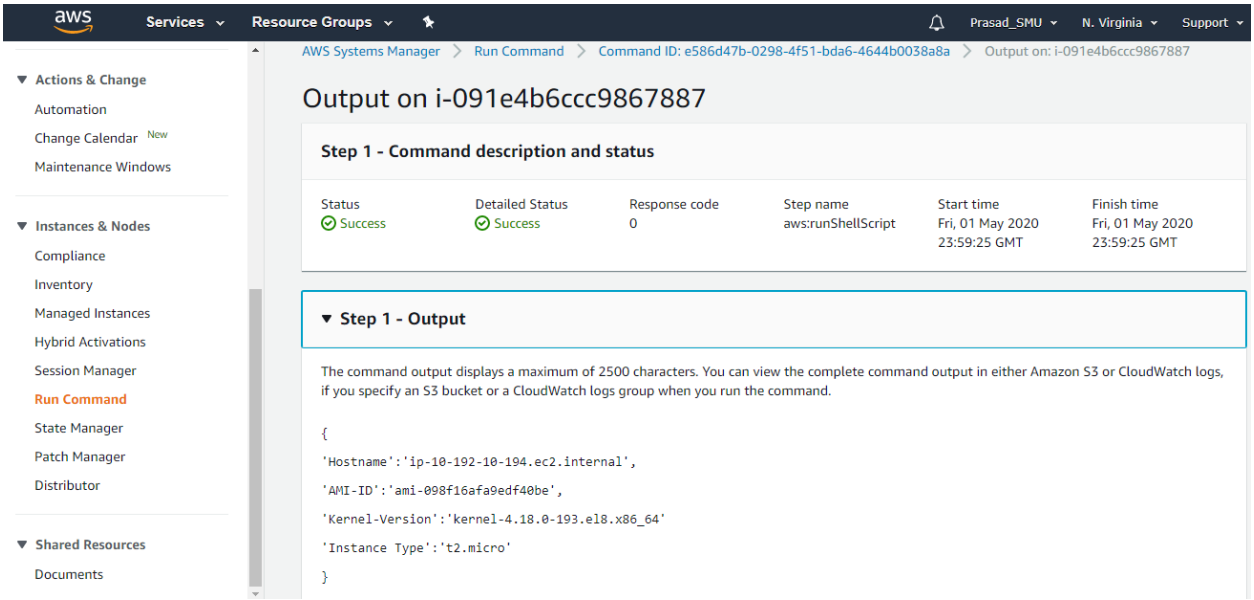



Now click on another Instance ID.

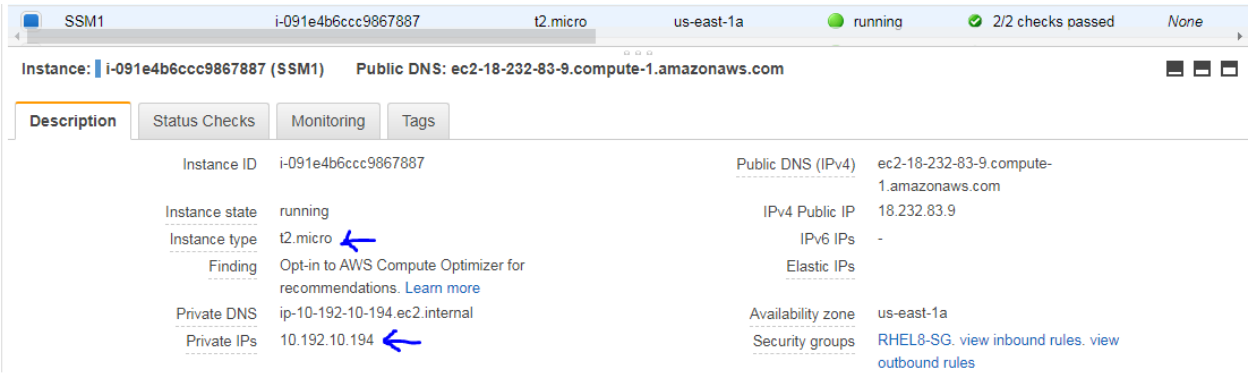


Expand **Step 1-Output**.

You'll observe the Hostname, AMI-ID, Kernel-Version, Instance-Type of the EC2 Instance.




Now verify the Hostname, AMI-ID, Kernel-Version, Instance-Type of the EC2 Instance from EC2 Service Dashboard.



| | | | | | | |
|---|---|---|---|---|---|---|
| SSM1 | i-091e4b6ccc9867887 | t2.micro | us-east-1a | running | 2/2 checks passed | None |

| | | | |
|---|---|---|---|
| Private DNS | ip-10-192-10-194.ec2.internal | Availability zone | us-east-1a |
| Private IPs | 10.192.10.194 | Security groups | RHEL8-SG. view inbound rules. view outbound rules |
| Secondary private IPs | | Scheduled events | No scheduled events |
| VPC ID | vpc-062814d035612343e (Custom VPC) | AMI ID | RHEL-8.2.0_HVM-20200423-x86_64-0-Hourly2-GP2 (ami-098f16afa9edf40be) |
| Subnet ID | subnet-01ee44283bcd09e5c (Public Subnet 1) | Platform details | Red Hat Enterprise Linux |
| Network interfaces | eth0 | Usage operation | RunInstances:0010 |
| IAM role | EC2-Role-SSM | Source/dest. check | True |
| Key pair name | LinuxServer | T2/T3 Unlimited | Disabled |
| Owner | 616399057974 | EBS-optimized | False |

This completes the lab on Automation using AWS-Systems Manager Service.

For questions, contact me on pbhavsar@smu.edu .