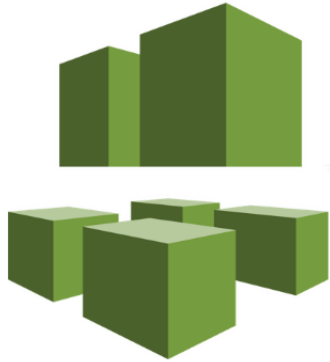


## Automation using AWS Systems Manager Service

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.

The screenshot shows the 'Create role' page in the AWS IAM console. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a user profile 'Prasad\_SMU'. The page title is 'Create role' with a progress indicator showing steps 1, 2, 3, and 4. Step 1 is 'Select type of trusted entity'. Below this, there are four options: 'AWS service' (selected), 'Another AWS account', 'Web identity', and 'SAML 2.0 federation'. The 'AWS service' option is highlighted with a blue border and includes a description: 'Allows AWS services to perform actions on your behalf. Learn more'. Below the options, there is a section 'Choose a use case' with 'Common use cases' listed: 'EC2' (selected) and 'Lambda'. The 'EC2' use case is highlighted in blue and includes a description: 'Allows EC2 instances to call AWS services on your behalf.' Below this, there is a section 'Or select a service to view its use cases' with a grid of service icons: API Gateway, CodeDeploy, EMR, KMS, RoboMaker, AWS Backup, CodeGuru, ElastiCache, Kinesis, and S3. At the bottom, there is a 'Required' label and two buttons: 'Cancel' and 'Next: Permissions'.

Select the below two Default IAM Policies:

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

The screenshot shows the 'Create role' page in the AWS IAM console, step 2: 'Attach permissions policies'. The top navigation bar is the same as the previous screenshot. The page title is 'Create role' with a progress indicator showing steps 1, 2, 3, and 4. Step 2 is 'Attach permissions policies'. Below this, there is a section 'Attach permissions policies' with a sub-section 'Choose one or more policies to attach to your new role.' There is a 'Create policy' button and a search bar. The search bar contains the text 'SSM' and shows 'Showing 13 results'. Below the search bar, there is a table with columns 'Policy name' and 'Used as'. The table lists several policies, with 'AmazonEC2RoleforSSM' and 'AmazonSSMFullAccess' selected (checked). The 'Used as' column shows 'Permissions policy (1)' for the selected policies. At the bottom, there is a 'Required' label and three buttons: 'Cancel', 'Previous', and 'Next: Tags'.

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

**Create role**

**Review**

Provide the required information below and review this role before you create it.

**Role name\***

Use alphanumeric and '+,=, @, \_' characters. Maximum 64 characters.

**Role description**

Maximum 1000 characters. Use alphanumeric and '+,=, @, \_' characters.

**Trusted entities** AWS service: ec2.amazonaws.com

**Policies**

- [AmazonEC2RoleforSSM](#)
- [AmazonSSMFullAccess](#)

**Permissions boundary** Permissions boundary is not set

\* Required

[Cancel](#) [Previous](#) [Create role](#)

[Feedback](#) [English \(US\)](#) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

## 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**.

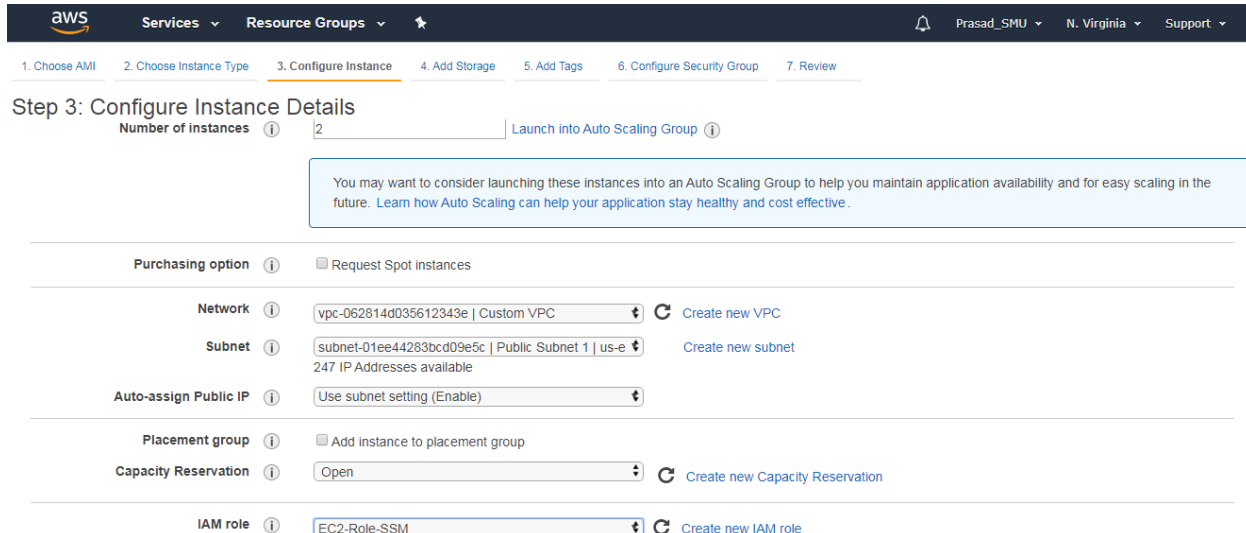
**Step 1: Choose an Amazon Machine Image (AMI)**

[Cancel and Exit](#)

☐ Free tier only ⓘ

AMI	AMI ID	Architecture	Root Device Type	Virtualization Type	ENA Enabled	Select
<b>Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type</b>	ami-0915e09cc7ceee3ab	64-bit (x86)	ebs	hvm	Yes	<a href="#">Select</a>
<b>Red Hat Enterprise Linux 8 (HVM), SSD Volume Type</b>	ami-098f16afa9edf40be / ami-029ba835ddd43c34f	64-bit (x86) / 64-bit (Arm)	ebs	hvm	Yes	<a href="#">Select</a>
<b>SUSE Linux Enterprise Server 15 SP1 (HVM), SSD Volume Type</b>	ami-0068cd63259e9f24c / ami-062c482b34db4e4e0	64-bit (x86) / 64-bit (Arm)	ebs	hvm	Yes	<a href="#">Select</a>

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.



Step 3: Configure Instance Details

Number of instances: 2 Launch into Auto Scaling Group

You may want to consider launching these instances into an Auto Scaling Group to help you maintain application availability and for easy scaling in the future. [Learn how Auto Scaling can help your application stay healthy and cost effective.](#)

**Purchasing option** ☐ Request Spot instances

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

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

**Auto-assign Public IP** Use subnet setting (Enable)

**Placement group** ☐ Add instance to placement group

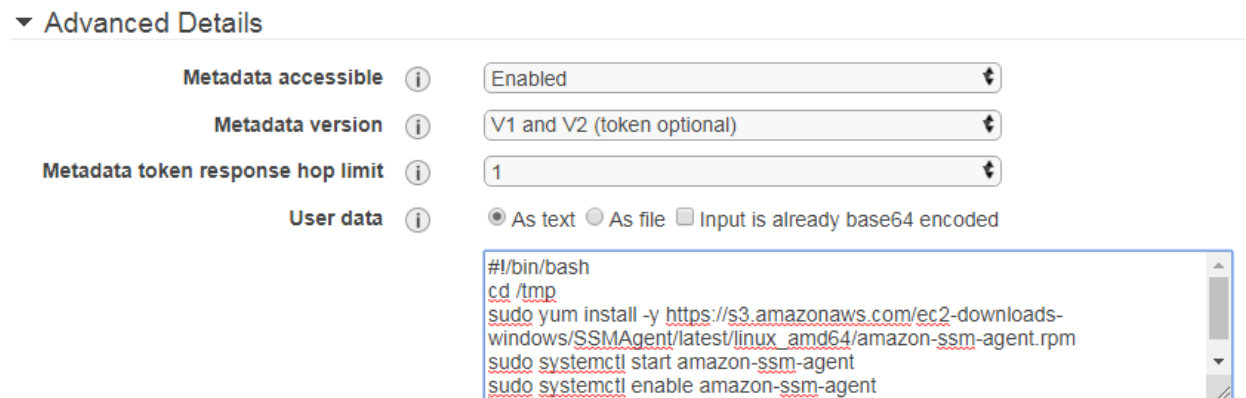
**Capacity Reservation** Open [Create new Capacity Reservation](#)

**IAM role** EC2-Role-SSM [Create new IAM role](#)

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.



**Advanced Details**

**Metadata accessible** Enabled

**Metadata version** V1 and V2 (token optional)

**Metadata token response hop limit** 1

**User data** ☒ As text ☐ As file ☐ Input is already base64 encoded

```
#!/bin/bash
cd /tmp
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-
windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm
sudo systemctl start amazon-ssm-agent
sudo systemctl enable amazon-ssm-agent
```

Click on Next.

You can mention Tags as per your choice.

**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	SSM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Env	Production	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
OS	RHEL8	<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.

**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](#)

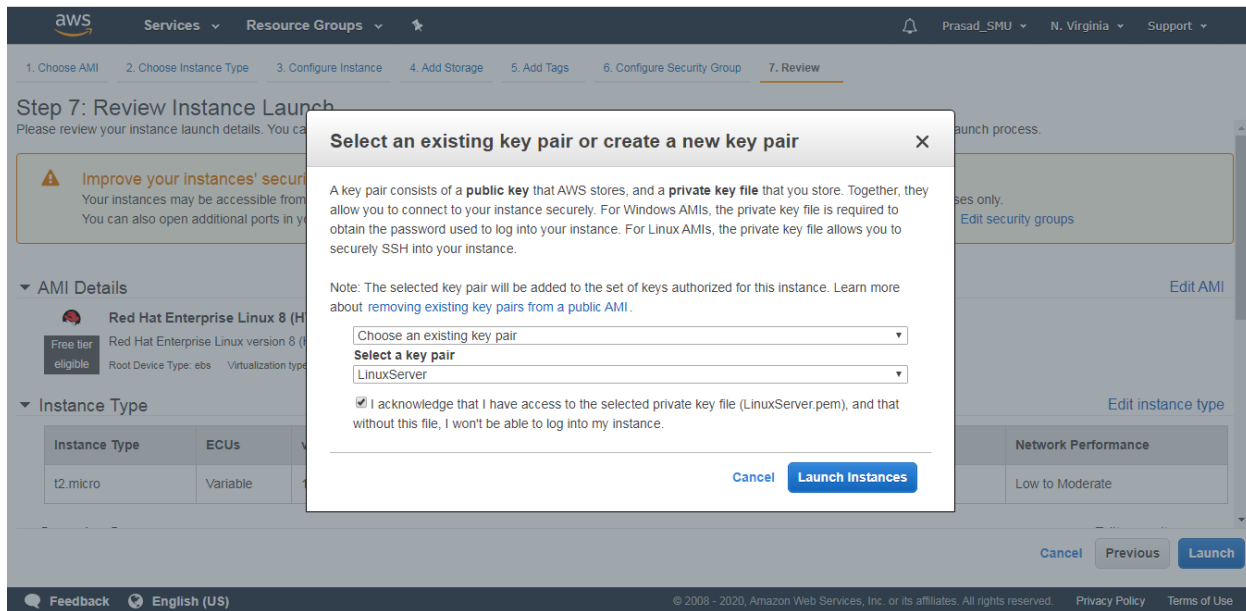
**Warning**

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

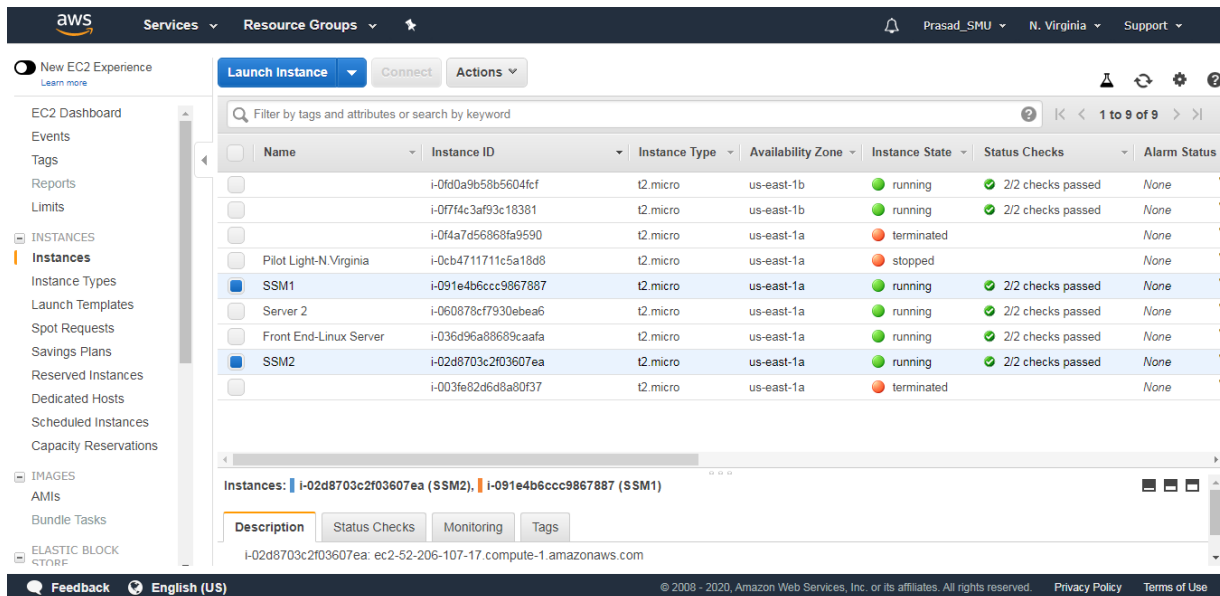
Click on Review and Launch.

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

Click on Launch Instances.



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



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

Navigate to AWS Systems Manager Service.

On 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.

The screenshot shows the AWS Systems Manager console. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information. The left sidebar lists navigation options: Quick Setup, Operations Management (Explorer, OpsCenter, CloudWatch Dashboard, Trusted Advisor & PHD), Application Management (Resource Groups, AppConfig, Parameter Store), Actions & Change (Automation, Change Calendar, Maintenance Windows), and Instances & Nodes. The main content area features a large header with the title 'AWS Systems Manager' and the subtitle 'Gain Operational Insight and Take Action on AWS Resources.' Below this is a 'Get Started with Systems Manager' button. A section titled 'How it works' includes icons representing various management tasks. A 'More resources' section with a 'Documentation' link is also present. The footer contains copyright information and links to Privacy Policy and Terms of Use.

This screenshot displays the 'Managed Instances' page within the AWS Systems Manager console. The left sidebar is updated to show 'Managed Instances' under the 'Instances & Nodes' section. The main content area has tabs for 'Managed Instances' and 'Settings'. Below the tabs, there are buttons for 'View details', 'Agent auto update', 'Configure Inventory', and 'Actions'. A search bar is provided. A table lists the managed instances:

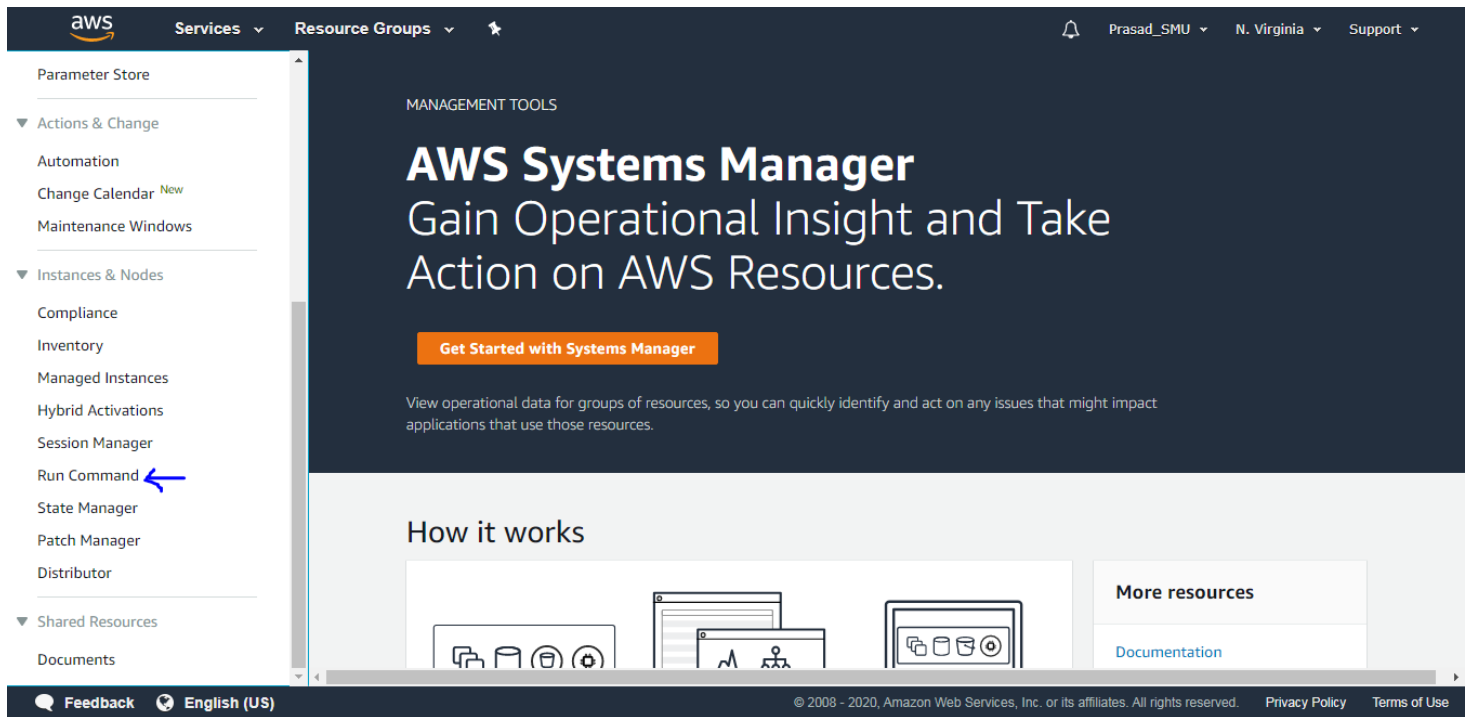
	Instance ID	Name	Ping status	Platform type	Platform name
<input type="radio"/>	i-02a89e653909f7a66		Online	Linux	Amazon Linux
<input type="radio"/>	i-0a25233378569135c		Online	Linux	Amazon Linux

On the right, a 'Managed Instance' panel provides a definition: 'A managed instance is any Amazon EC2 instance or on-premises server or virtual machine in your hybrid environment that has been configured for Systems Manager. [Learn More](#)'

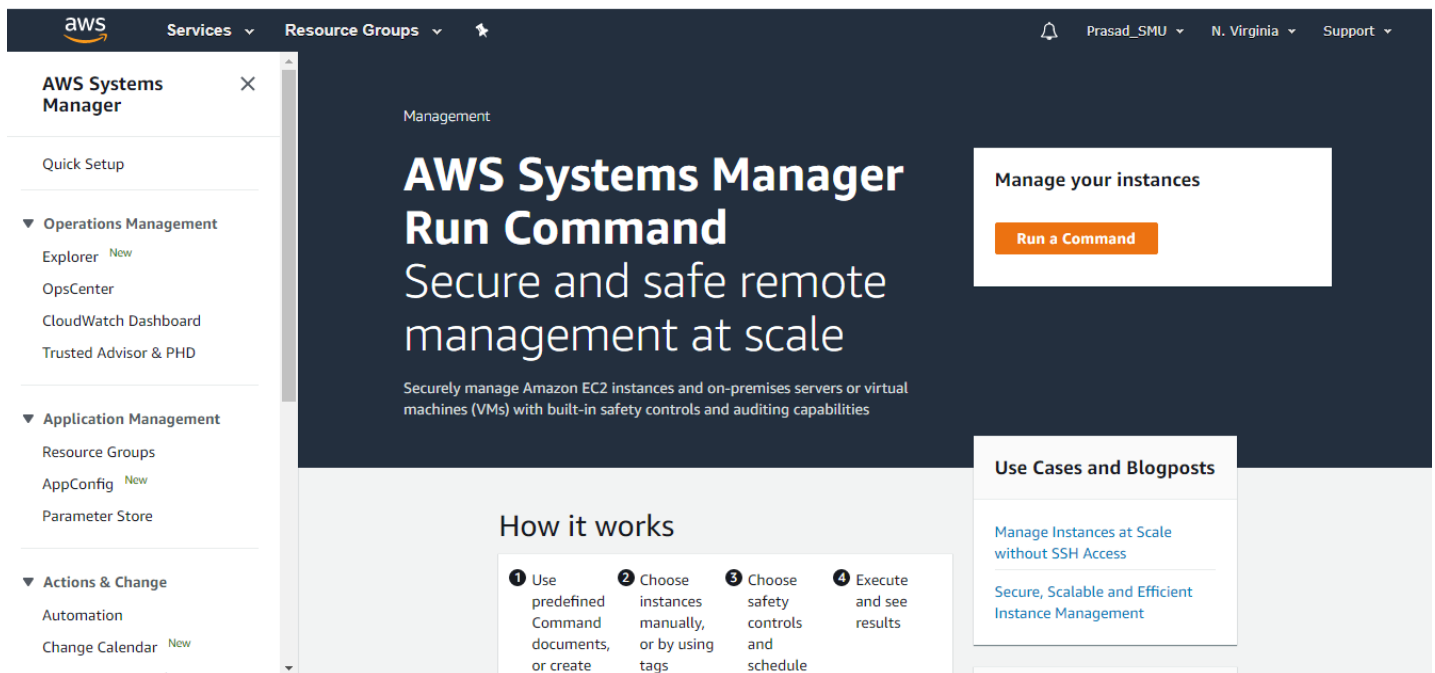
You can also verify the Instance IDs from EC2 Service Dashboard.

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

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



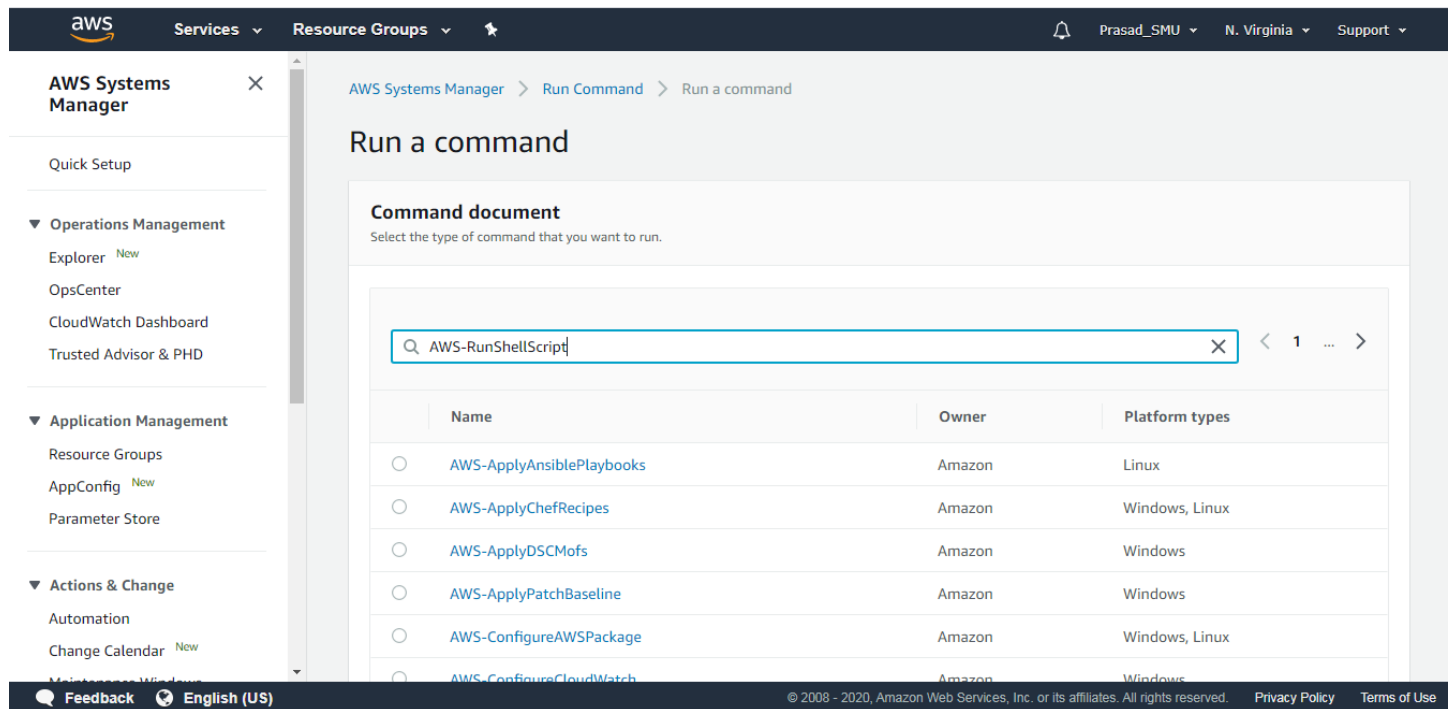
Click on Run a Command.





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

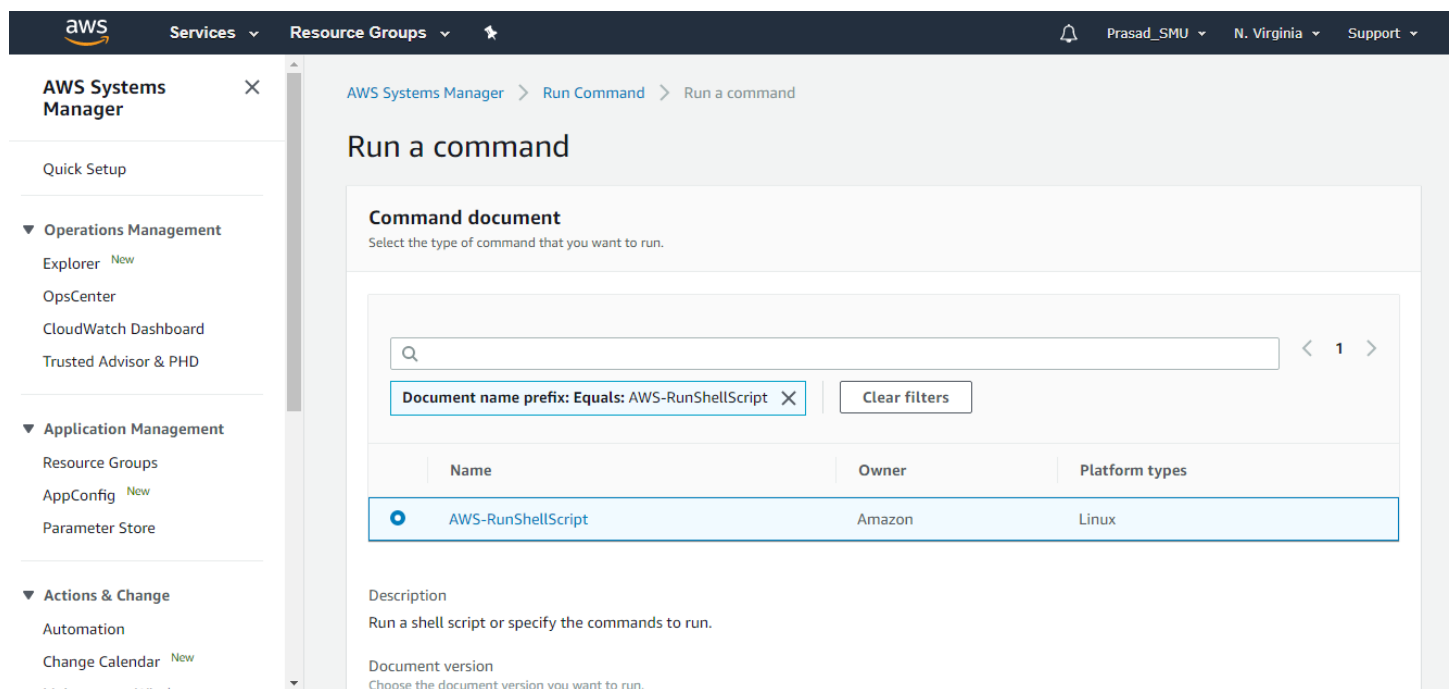
## AWS-RunShellScript



The screenshot shows the AWS Systems Manager console. The left sidebar contains navigation links for AWS Systems Manager, Quick Setup, Operations Management (Explorer, OpsCenter, CloudWatch Dashboard, Trusted Advisor & PHD), Application Management (Resource Groups, AppConfig, Parameter Store), and Actions & Change (Automation, Change Calendar). The main content area is titled 'Run a command' and shows a search bar with 'AWS-RunShellScript'. Below the search bar is a table of command documents.

	Name	Owner	Platform types
<input type="radio"/>	AWS-ApplyAnsiblePlaybooks	Amazon	Linux
<input type="radio"/>	AWS-ApplyChefRecipes	Amazon	Windows, Linux
<input type="radio"/>	AWS-ApplyDSCMofs	Amazon	Windows
<input type="radio"/>	AWS-ApplyPatchBaseline	Amazon	Windows
<input type="radio"/>	AWS-ConfigureAWSPackage	Amazon	Windows, Linux
<input type="radio"/>	AWS-ConfigureCloudWatch	Amazon	Windows

Select the Command Document.



The screenshot shows the AWS Systems Manager console. The left sidebar is the same as the previous screenshot. The main content area is titled 'Run a command' and shows a search bar. Below the search bar is a filter bar with 'Document name prefix: Equals: AWS-RunShellScript'. Below the filter bar is a table of command documents.

	Name	Owner	Platform types
<input checked="" type="radio"/>	AWS-RunShellScript	Amazon	Linux

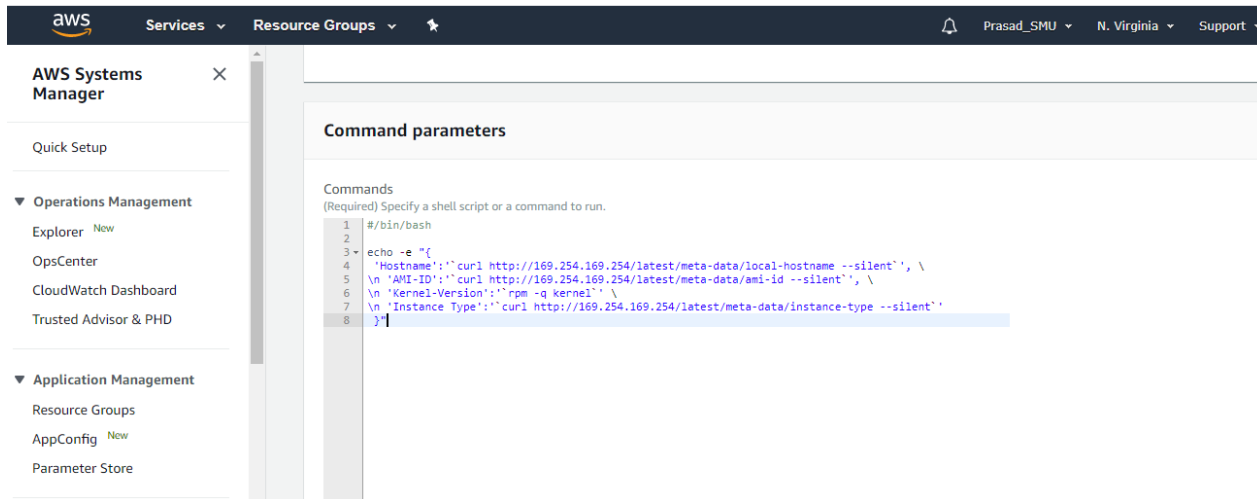
Description  
Run a shell script or specify the commands to run.

Document version  
Choose the document version you want to run.



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.



```
1 #/bin/bash
2
3 echo -e "{
4   'Hostname': `curl http://169.254.169.254/latest/meta-data/local-hostname --silent`, \
5   'AMI-ID': `curl http://169.254.169.254/latest/meta-data/ami-id --silent`, \
6   'Kernel-Version': `rpm -q kernel` \
7   'Instance Type': `curl http://169.254.169.254/latest/meta-data/instance-type --silent`
8 }
```

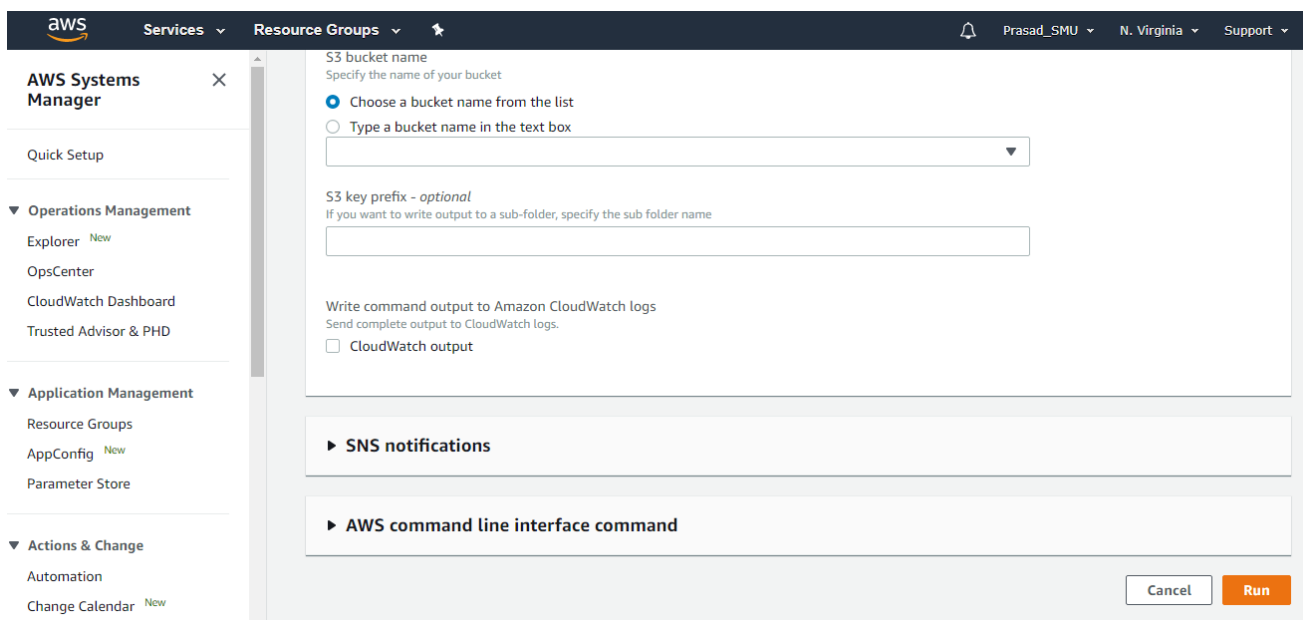
You can also specify the S3 Bucket Name wherein logs of the Run Command will be saved.

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.

The screenshot shows the AWS Systems Manager console. The left sidebar contains navigation options: Quick Setup, Operations Management (Explorer, OpsCenter, CloudWatch Dashboard, Trusted Advisor & PHD), Application Management (Resource Groups, AppConfig, Parameter Store), and Actions & Change (Automation, Change Calendar). The main content area displays the 'Run Command' page for Command ID: e586d47b-0298-4f51-bda6-4644b0038a8a. The command status is 'In Progress'. The 'Targets and outputs' table shows two targets in progress.

Instance ID	Instance name	Status	Detailed Status	Start time	Finish time
i-02d8703c2f03607ea	ip-10-192-10-203.ec2.internal	In Progress	In Progress		
i-091e4b6ccc9867887	ip-10-192-10-194.ec2.internal	In Progress	In Progress		

Status changes to **SUCCESS**.

The screenshot shows the AWS Systems Manager console with the same command ID: e586d47b-0298-4f51-bda6-4644b0038a8a. The command status is now 'Success'. The 'Targets and outputs' table shows two targets completed successfully.

Instance ID	Instance name	Status	Detailed Status	Start time	Finish time
i-02d8703c2f03607ea	ip-10-192-10-203.ec2.internal	Success	Success	Fri, 01 May 2020 23:59:25 GMT	Fri, 01 May 2020 23:59:25 GMT
i-091e4b6ccc9867887	ip-10-192-10-194.ec2.internal	Success	Success	Fri, 01 May 2020 23:59:25 GMT	Fri, 01 May 2020 23:59:25 GMT

Click on any Instance ID.

The screenshot shows the AWS Systems Manager console. A green notification bar at the top states: "Command ID: e586d47b-0298-4f51-bda6-4644b0038a8a was successfully sent!". The breadcrumb trail is: AWS Systems Manager > Run Command > Command ID: e586d47b-0298-4f51-bda6-4644b0038a8a > Output on: i-02d8703c2f03607ea. The main heading is "Output on i-02d8703c2f03607ea". Below it is a table titled "Step 1 - Command description and status".

Status	Detailed Status	Response code	Step name	Start time	Finish time
Success	Success	0	aws:runShellScript	Fri, 01 May 2020 23:59:25 GMT	Fri, 01 May 2020 23:59:25 GMT

Below the table is a section titled "Step 1 - Output" which is currently collapsed.

Expand **Step 1-Output**.

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

This screenshot is similar to the previous one, but the "Step 1 - Output" section is expanded. It contains the following text:

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'
}
```

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.

The screenshot shows the AWS Management Console interface. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information. The left sidebar shows the 'EC2 Dashboard' with various navigation options. The main content area displays a table of EC2 instances. The instance 'SSM2' is selected, and its details are shown below the table. A blue arrow points to the 'Private DNS' field, which displays 'ip-10-192-10-203.ec2.internal'.

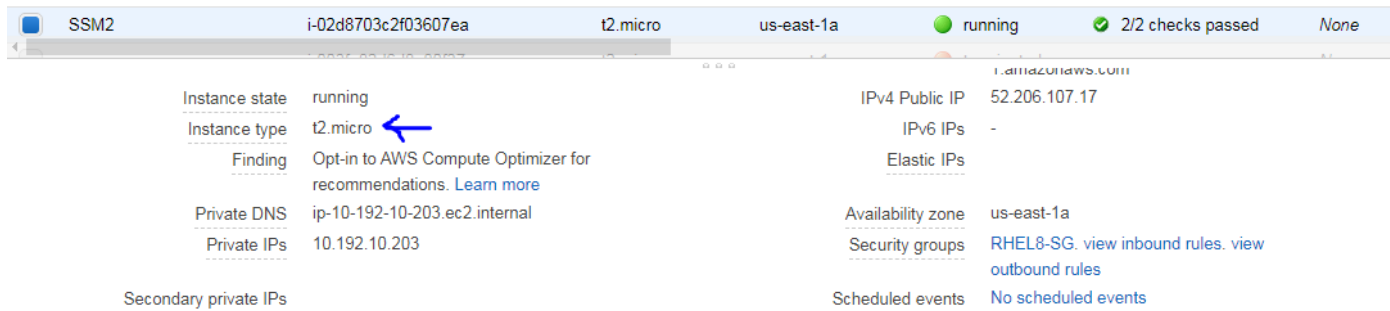
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
	i-0fd0a9b58b5604fcf	t2.micro	us-east-1b	running	2/2 checks passed	None
	i-0f7f4c3af93c18381	t2.micro	us-east-1b	running	2/2 checks passed	None
	i-0f4a7d56868fa9590	t2.micro	us-east-1a	terminated		None
Pilot Light-N.Virginia	i-0cb4711711c5a18d8	t2.micro	us-east-1a	stopped		None
SSM1	i-091e4b6ccc9867887	t2.micro	us-east-1a	running	2/2 checks passed	None
Server 2	i-060878cf7930e8ea6	t2.micro	us-east-1a	running	2/2 checks passed	None
Front End-Linux Server	i-036d96a88689caafa	t2.micro	us-east-1a	running	2/2 checks passed	None
SSM2	i-02d8703c2f03607ea	t2.micro	us-east-1a	running	2/2 checks passed	None

Instance state	running	IPv4 Public IP	52.206.107.17
Instance type	t2.micro	IPv6 IPs	-
Finding	Opt-in to AWS Compute Optimizer for recommendations. <a href="#">Learn more</a>	Elastic IPs	
Private DNS	ip-10-192-10-203.ec2.internal	Availability zone	us-east-1a
Private IPs	10.192.10.203	Security groups	RHEL8-SG. <a href="#">view inbound rules. view outbound rules</a>

Private IPs	10.192.10.203	Security groups	RHEL8-SG. <a href="#">view inbound rules. view outbound rules</a>
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



Instance state: running

Instance type: t2.micro

Finding: Opt-in to AWS Compute Optimizer for recommendations. [Learn more](#)

Private DNS: ip-10-192-10-203.ec2.internal

Private IPs: 10.192.10.203

Secondary private IPs

IPv4 Public IP: 52.206.107.17

IPv6 IPs: -

Elastic IPs

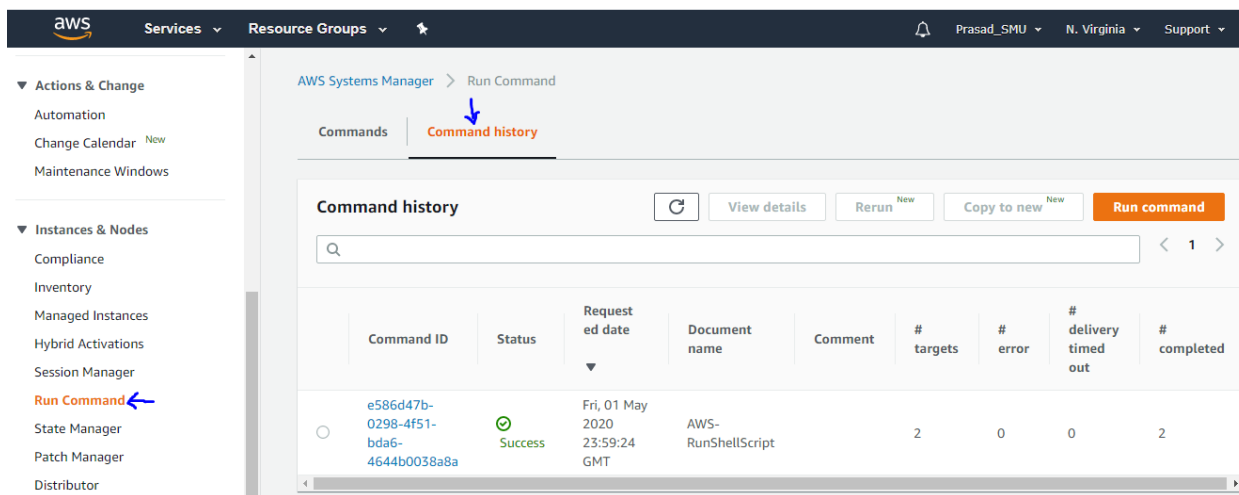
Availability zone: us-east-1a

Security groups: [RHEL8-SG](#) [view inbound rules](#) [view outbound rules](#)

Scheduled events: [No scheduled events](#)

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

Click on Command History.



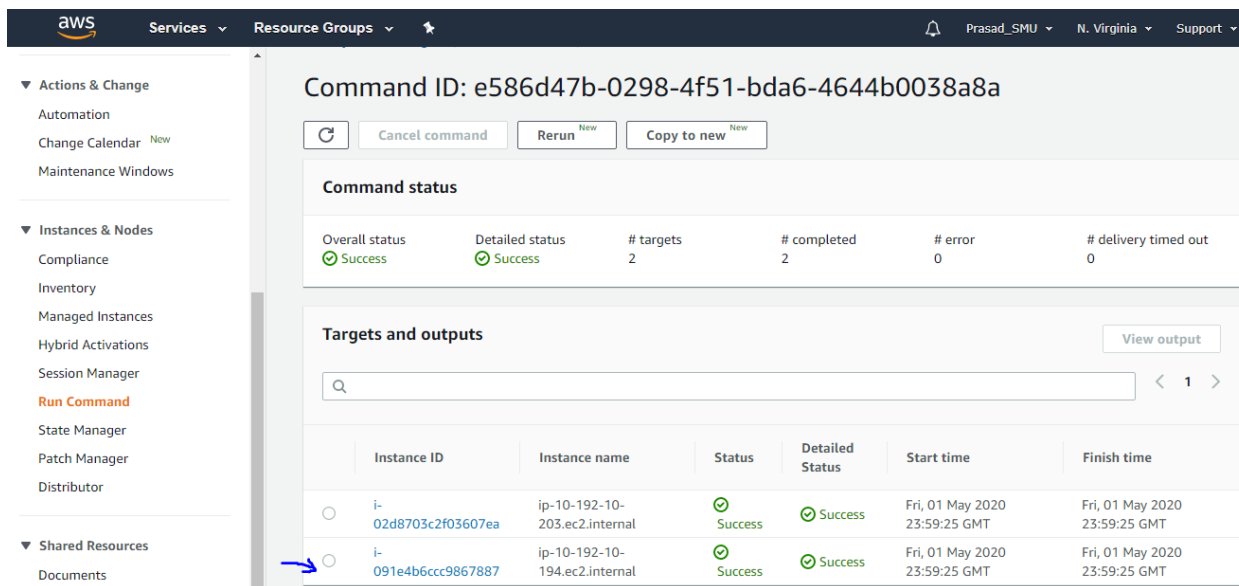
Commands | **Command history**

Command history

View details Rerun Copy to new Run command

Command ID	Status	Request ed date	Document name	Comment	# targets	# error	# delivery timed out	# completed
e586d47b-0298-4f51-bda6-4644b0038a8a	Success	Fri, 01 May 2020 23:59:24 GMT	AWS-RunShellScript		2	0	0	2

Now click on another Instance ID.



Command ID: e586d47b-0298-4f51-bda6-4644b0038a8a

Cancel command Rerun Copy to new

**Command status**

Overall status	Detailed status	# targets	# completed	# error	# delivery timed out
Success	Success	2	2	0	0

**Targets and outputs**

View output

Instance ID	Instance name	Status	Detailed Status	Start time	Finish time
i-02d8703c2f03607ea	ip-10-192-10-203.ec2.internal	Success	Success	Fri, 01 May 2020 23:59:25 GMT	Fri, 01 May 2020 23:59:25 GMT
i-091e4b6ccc9867887	ip-10-192-10-194.ec2.internal	Success	Success	Fri, 01 May 2020 23:59:25 GMT	Fri, 01 May 2020 23:59:25 GMT

Expand **Step 1-Output**.

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

**Output on i-091e4b6ccc9867887**

**Step 1 - Command description and status**

Status	Detailed Status	Response code	Step name	Start time	Finish time
Success	Success	0	aws:runShellScript	Fri, 01 May 2020 23:59:25 GMT	Fri, 01 May 2020 23:59:25 GMT

**▼ 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-10-192-10-194.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.


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

Instance: i-091e4b6ccc9867887 (SSM1) Public DNS: ec2-18-232-83-9.compute-1.amazonaws.com

**Description** Status Checks Monitoring Tags

Instance ID	i-091e4b6ccc9867887	Public DNS (IPv4)	ec2-18-232-83-9.compute-1.amazonaws.com
Instance state	running	IPv4 Public IP	18.232.83.9
Instance type	t2.micro	IPv6 IPs	-
Finding	Opt-in to AWS Compute Optimizer for recommendations. <a href="#">Learn more</a>	Elastic IPs	
Private DNS	ip-10-192-10-194.ec2.internal	Availability zone	us-east-1a
Private IPs	10.192.10.194	Security groups	RHEL8-SG: <a href="#">view inbound rules</a> , <a href="#">view outbound rules</a>



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. <a href="#">view inbound rules</a> . <a href="#">view outbound rules</a>		
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](mailto:pbhavsar@smu.edu) .