Runbook for AWS Systems Manager Capabilities

Summary:

This runbook aims to provide hands-on experience with key AWS Systems Manager (SSM) capabilities: **Session Manager**, **Hybrid Activation**, **Resource Groups**, **Patch Manager**, **Run Command, Automation**, and **State Manager**.

By the end of this runbook, you will understand how to configure and utilize some of these SSM features to manage EC2 instances efficiently.

---

1. Setting Up Instances:

Goal: Create and configure three EC2 instances to meet the prerequisites for becoming managed instances. A managed instance requires the SSM agent and an IAM role with the necessary permissions.

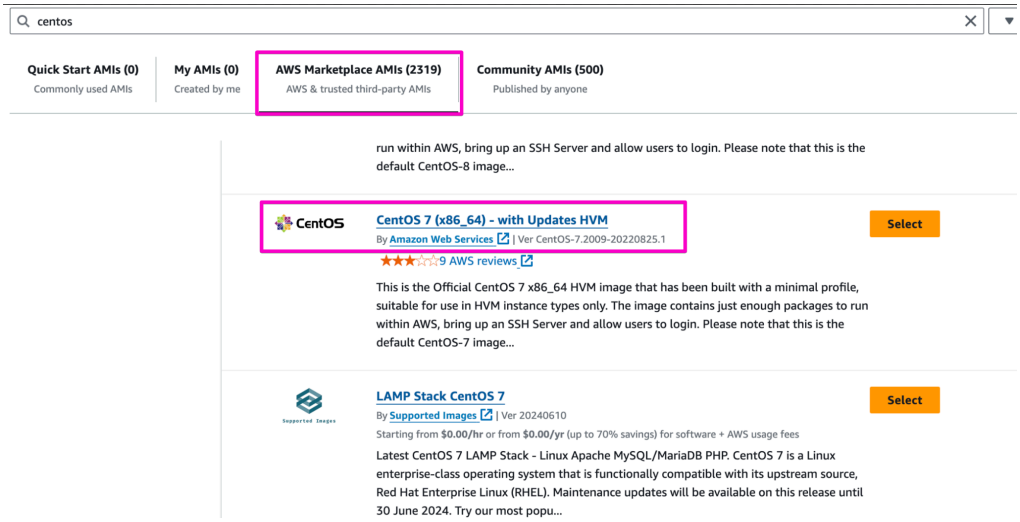
Note: Instances do not need port 22 open since we will use Session Manager for access. Open port 80 for HTTP traffic as we will install Apache via Run Command.

Prerequisite:

- A VPC (use VPC and more to create a VPC with prefix SSM, and 2 public subnets)
- create IAM role with necessary permissions for SSM

Step 1: Launch EC2 Instances

1. **CentOS Instance:**
    - **Name:** centos
    - **AMI:** CentOS 7



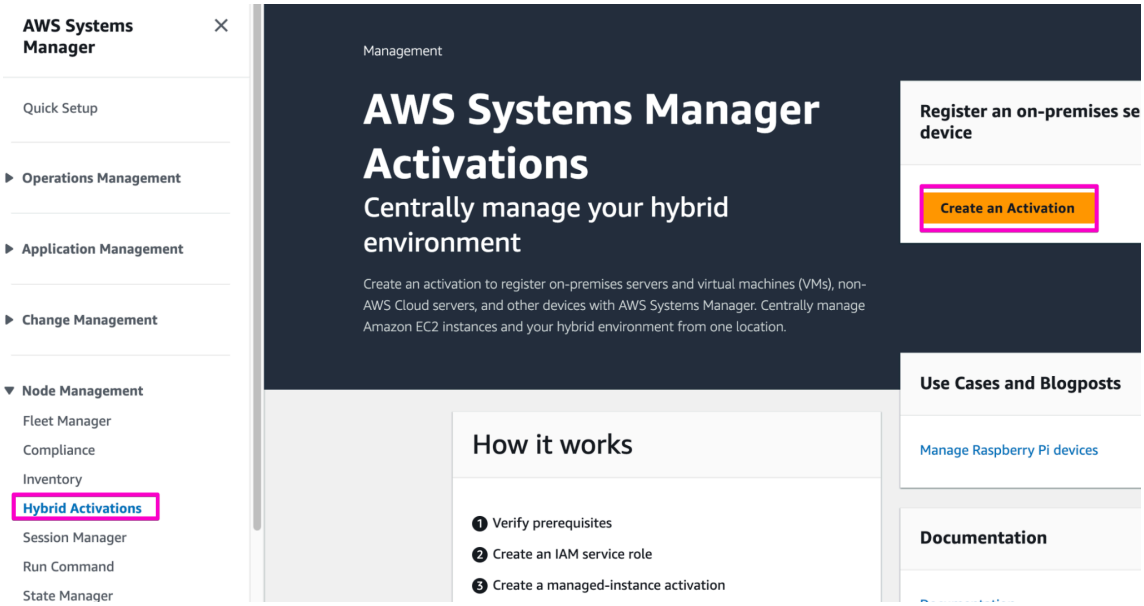
- **Instance Type:** t2.micro
- **Network:** Public Subnet
- **Security Group:** Allow port 80 (HTTP) No need for port 22
- **IAM Role:** Attach a role with `AmazonSSMManagedInstanceCore` policy.
- **User Data:** Installs the SSM agent to meet the managed instance prerequisite.

```
#!/bin/bash
yum install -y
https://s3.amazonaws.com/amazon-ssm-us-east-1/latest/linux_amd64/amazon-ssm-agent.rpm
systemctl start amazon-ssm-agent
systemctl enable amazon-ssm-agent
```

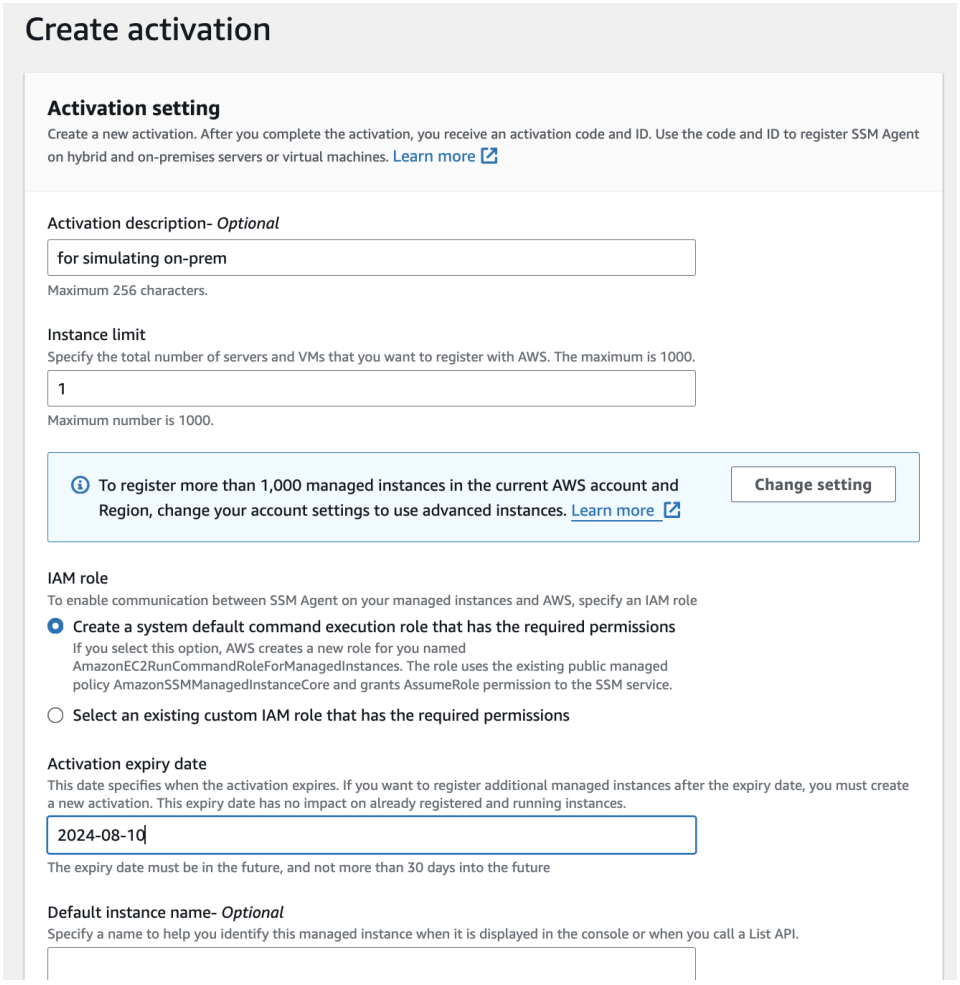
2. **Amazon Linux 2 Instance:**
   - **Name: amazonLinux**
   - **AMI:** Amazon Linux 2
   - **Instance Type:** t2.micro
   - **Network:** Public Subnet
   - **Security Group:** Allow port 80 (HTTP) No need for port 22
   - **IAM Role:** Attach a role with `AmazonSSMManagedInstanceCore` policy.
   - **Note:** The SSM agent is pre-installed on Amazon Linux 2 instances, so no user data is needed.

3. **RHEL 8 Instance:**

1. **Create Hybrid Activation:**
   - Navigate to the Systems Manager console.
   - Under "Hybrid Activations, "create a new activation.



- Provide necessary details and download the activation code and ID.



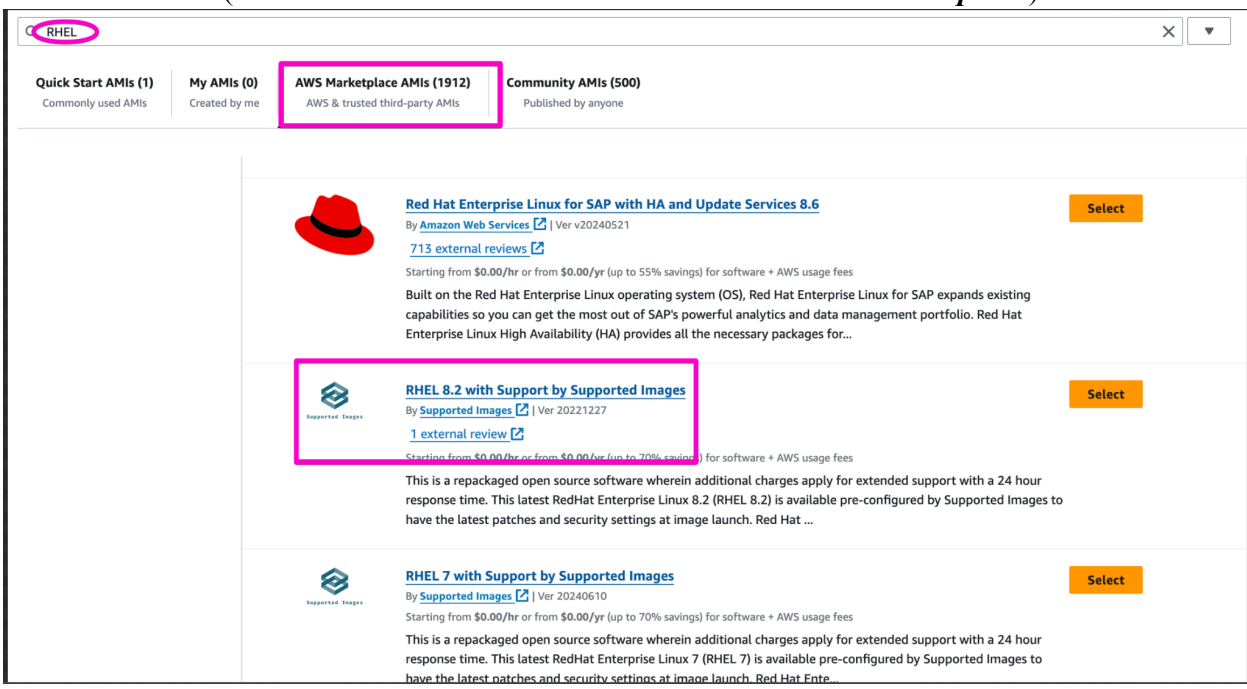
-

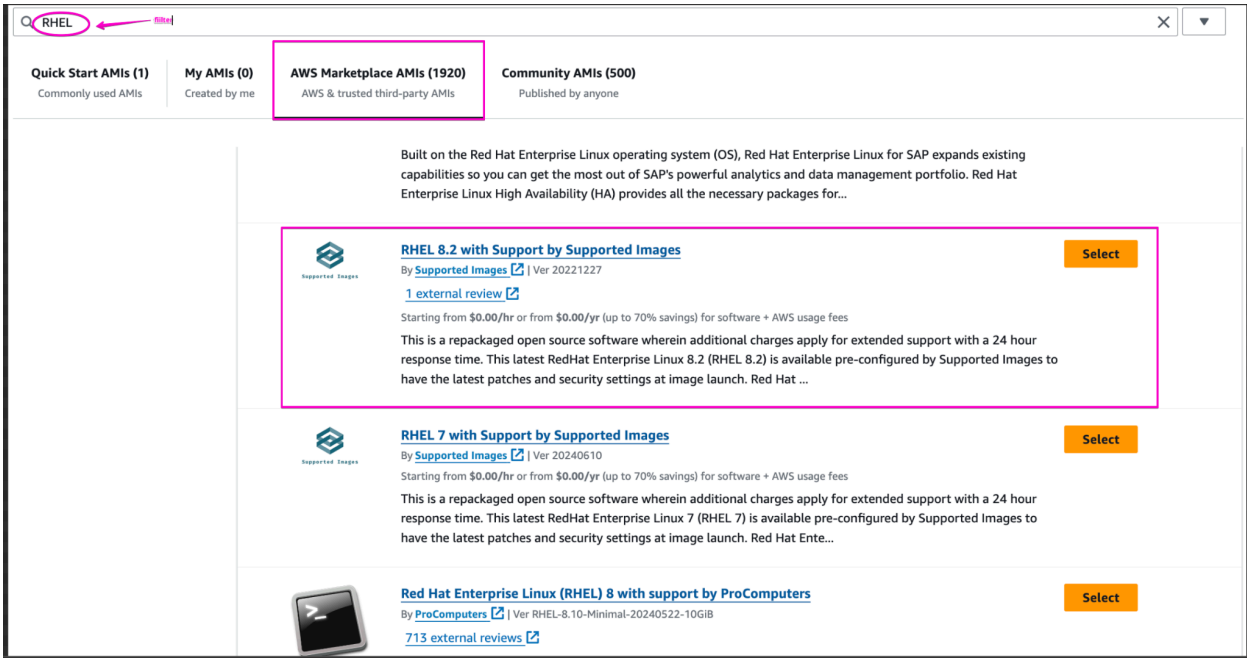
- click on create Activation

  NB copy the provided activation credentials before closing window

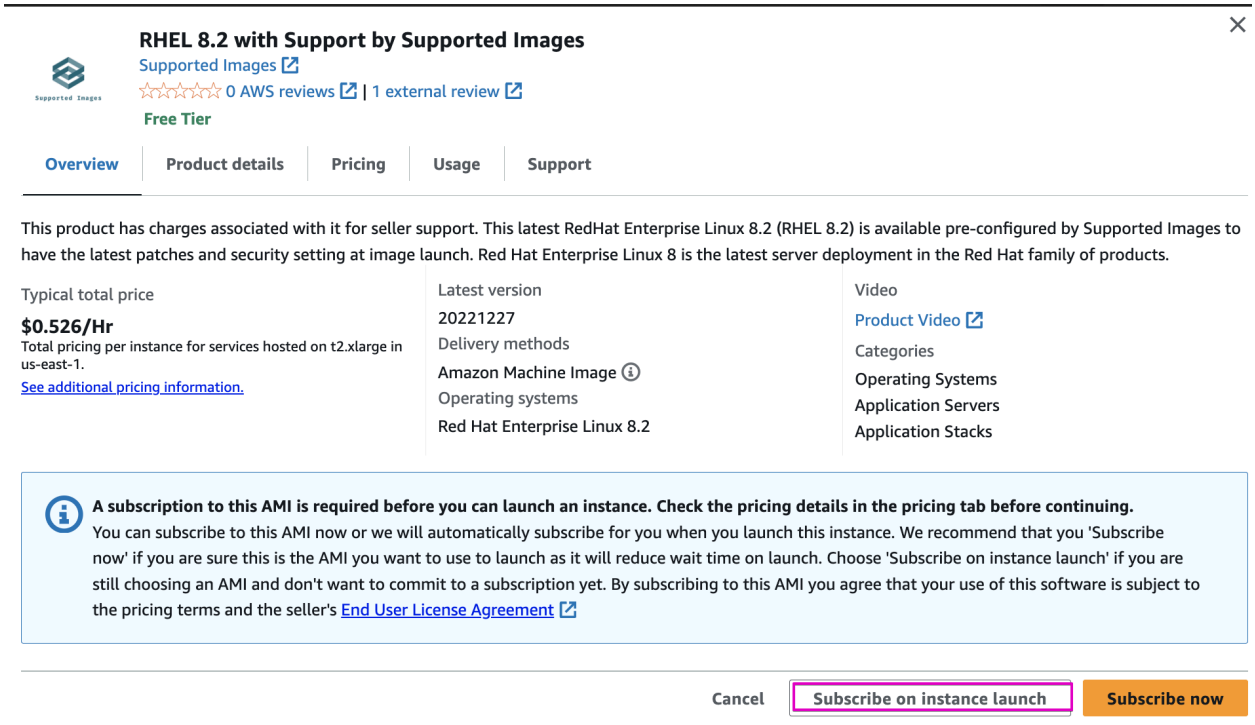
2. **Launch the RHEL Instance with User Data: (simulating Onprem machine)**

   - **Name:** RHEL
   - **AMI:** RHEL 8.2 (click on **Browse more AMIs** and search in ***AWS marketplace***)





- select the 8.2 the instance AMI and click on **subscribe on instance launch.** This will take you back to the (familiar) instance launch console, **confirm changes** and proceed.



- **Instance Type:** t2.micro
- **Network:** Public Subnet
- **Security Group:** Allow port 80 (HTTP) No need for port 22

- **User Data:** Installs the SSM agent and configures hybrid activation to simulate an on-premise environment. Ensure to pass your hybrid activation code, Id and region

```
#!/bin/bash
mkdir /tmp/ssm
sudo dnf install -y
https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_
amd64/amazon-ssm-agent.rpm
sudo systemctl stop amazon-ssm-agent
sudo amazon-ssm-agent -register -code "qMTYsh/a2Jb+URVyMpxO" -id
"eb791b57-d612-443a-93b9-5d4702ab9152" -region "us-east-1”
sudo systemctl start amazon-ssm-agent
sudo systemctl enable amazon-ssm-agent
```

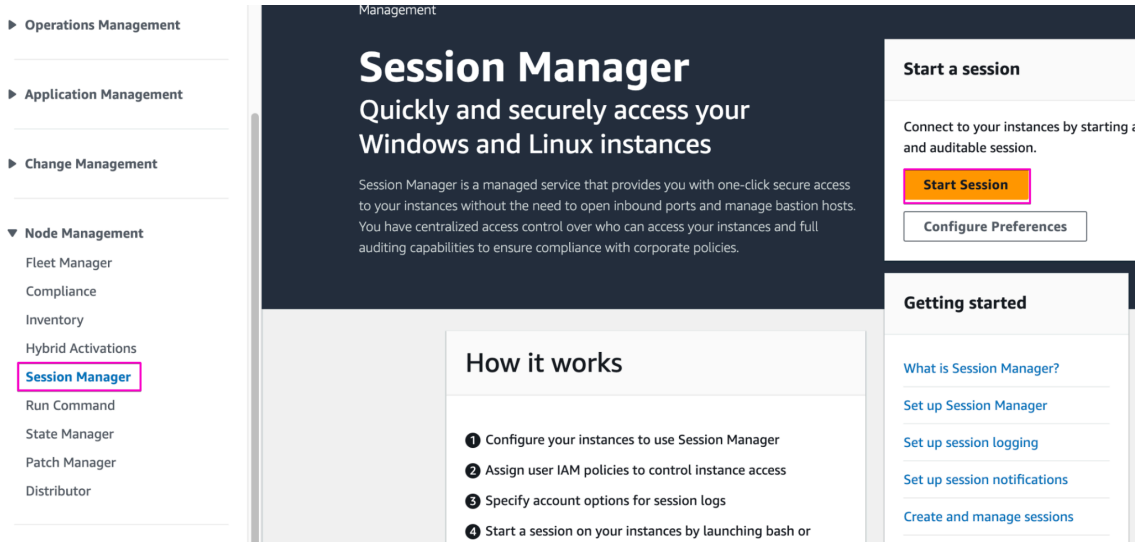
Step 3: Confirm Instances in Fleet Manager

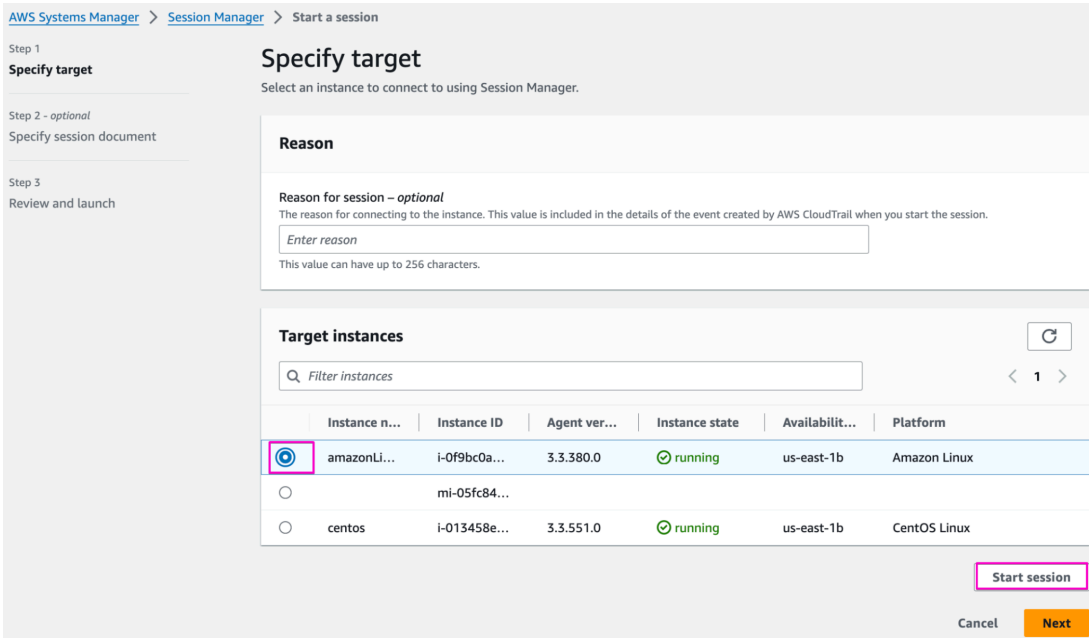
1. **Check Managed Instances:**
   - Navigate to Systems Manager > Fleet Manager.
   - Ensure all three instances appear as managed instances.

2. Session Manager:

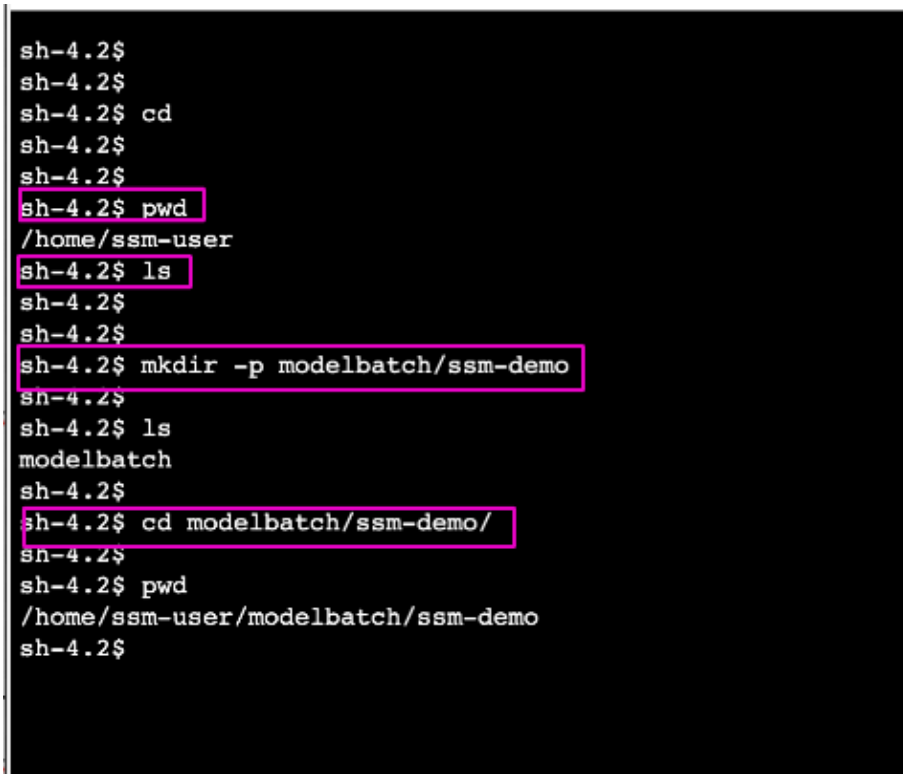
Goal: Use Session Manager to securely access and manage instances without needing SSH or bastion hosts.

1. **Access Instances via Session Manager:**
   - Navigate to Systems Manager > Session Manager.
   - Start a new session and select an instance (CentOS, Amazon Linux 2, RHEL 8).

- 

- 

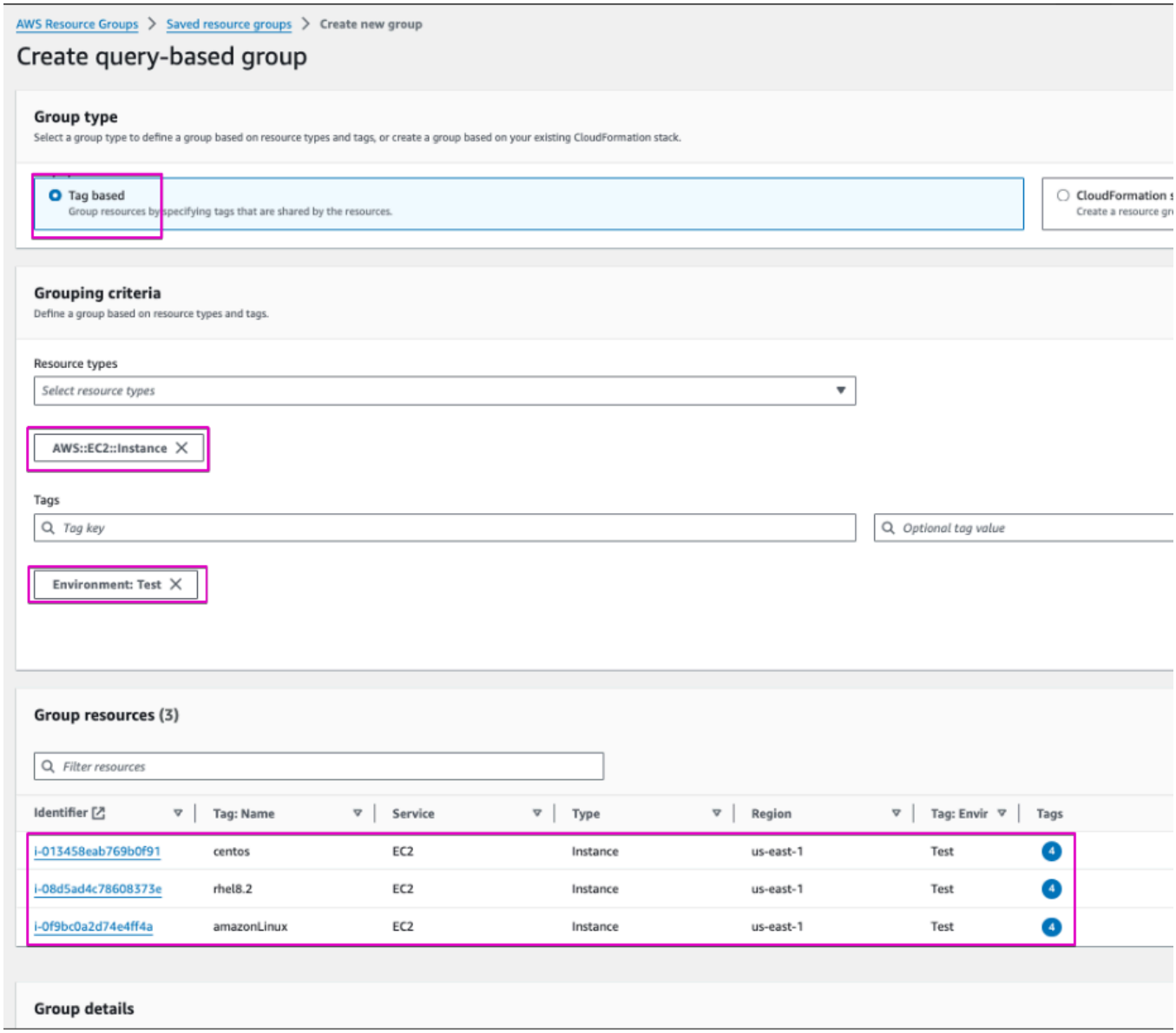
- this opens the terminal to the selected instance. Demonstrate basic command execution on each instance.

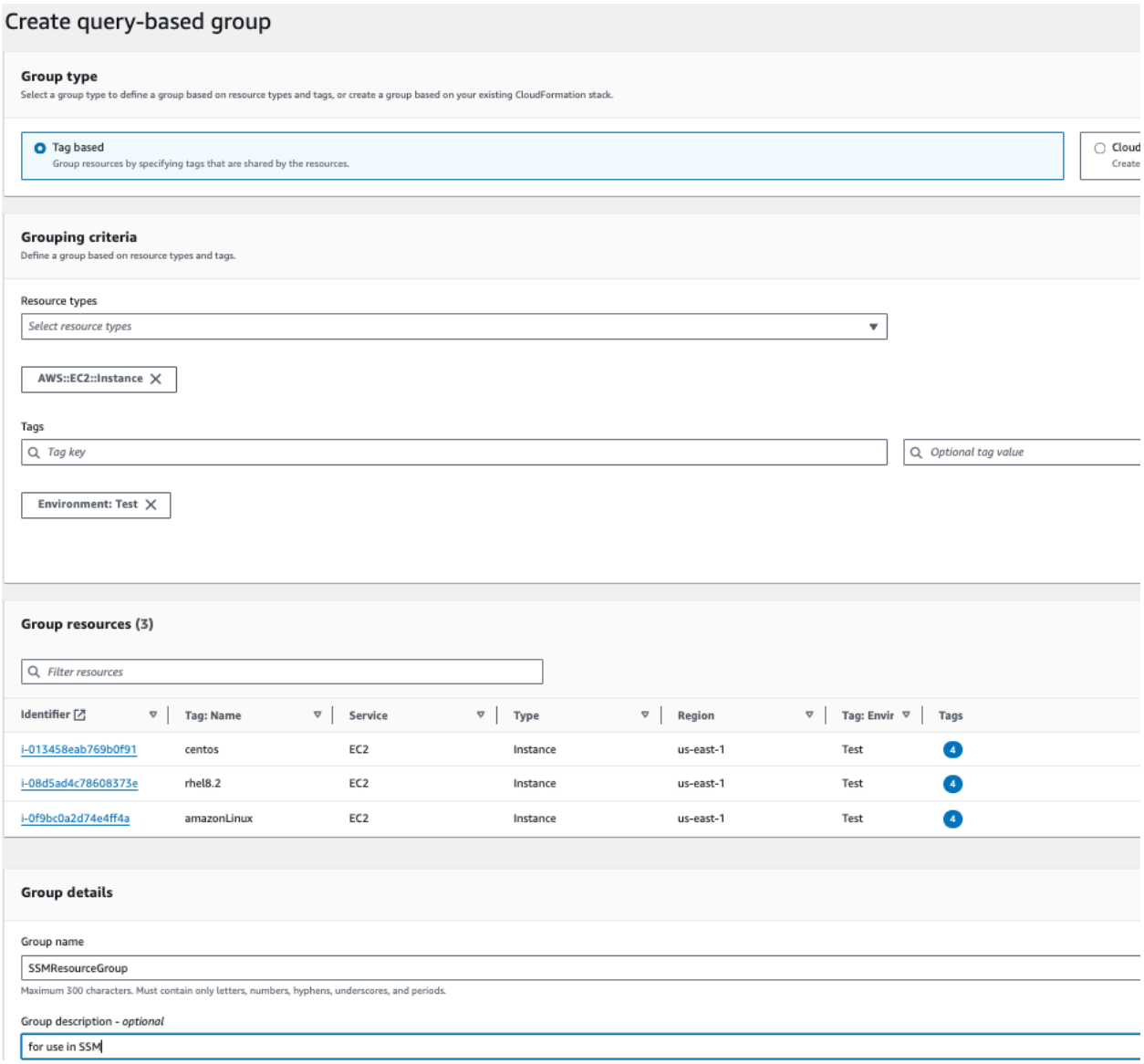


3. Resource Groups:

Goal: Use resource groups to organize and manage AWS resources more effectively. Resource groups allow you to group resources by specific criteria, such as tags, and manage them as a single entity.

1. **Create a Resource Group:**
    - **Tag Instances:**
        - Tag each instance with `Environment=Test`.
    - **Navigate to Resource Groups:**
        - Go to the AWS Management Console and select "Resource Groups."
    - **Create a New Group:**
        - Click on "Create a resource group."
        - Choose the "Tag-based group" option.
        - Define the tag (e.g., `Environment=Test`) to include the instances.
        - Name the group (e.g **SSMResourceGroup**) and review the resource list to ensure all instances are included.
        - **Create** the resource group.


AWS Resource Groups > Saved resource groups > Create new group
Create query-based group
Group type
Select a group type to define a group based on resource types and tags, or create a group based on your existing CloudFormation stack.
Tag based
Group resources by specifying tags that are shared by the resources.
CloudFormation
Grouping criteria
Define a group based on resource types and tags.
Resource types
Select resource types
AWS::EC2::Instance
Tags
Tag key
Optional tag value
Environment: Test
Group resources (3)
Filter resources
Identifier
Tag: Name
Service
Type
Region
Tag: Envir
Tags
i-013458eab769b0f91
centos
EC2
Instance
us-east-1
Test
4
i-08d5ad4c78608373e
rhel8.2
EC2
Instance
us-east-1
Test
4
i-0f9bc0a2d74e4ff4a
amazonLinux
EC2
Instance
us-east-1
Test
4
Group details

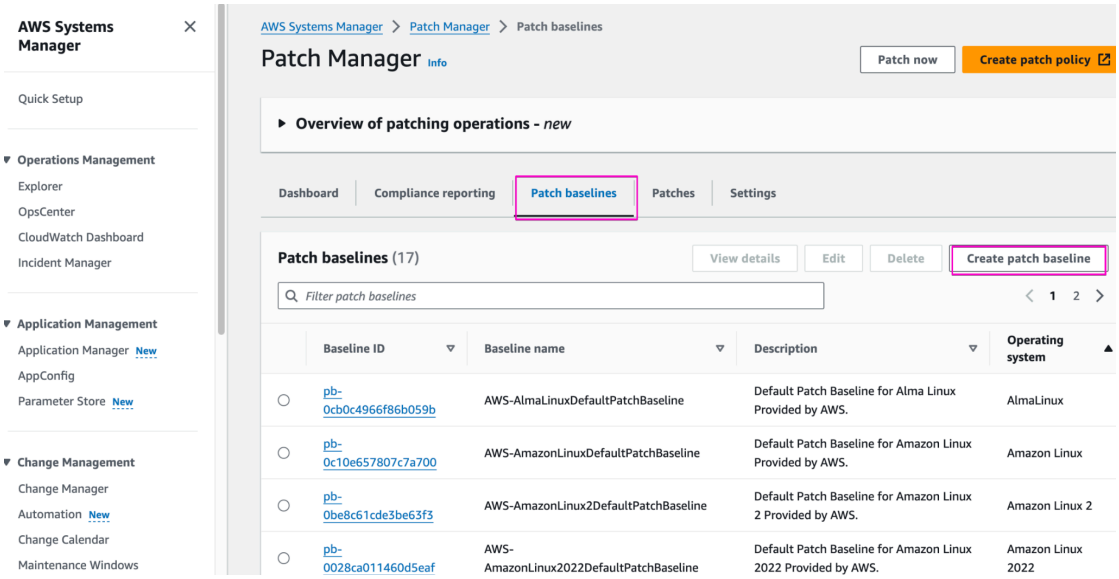


4. Patch Manager:

Goal: Automate patching of instances to ensure they are up-to-date with the latest security and software updates.

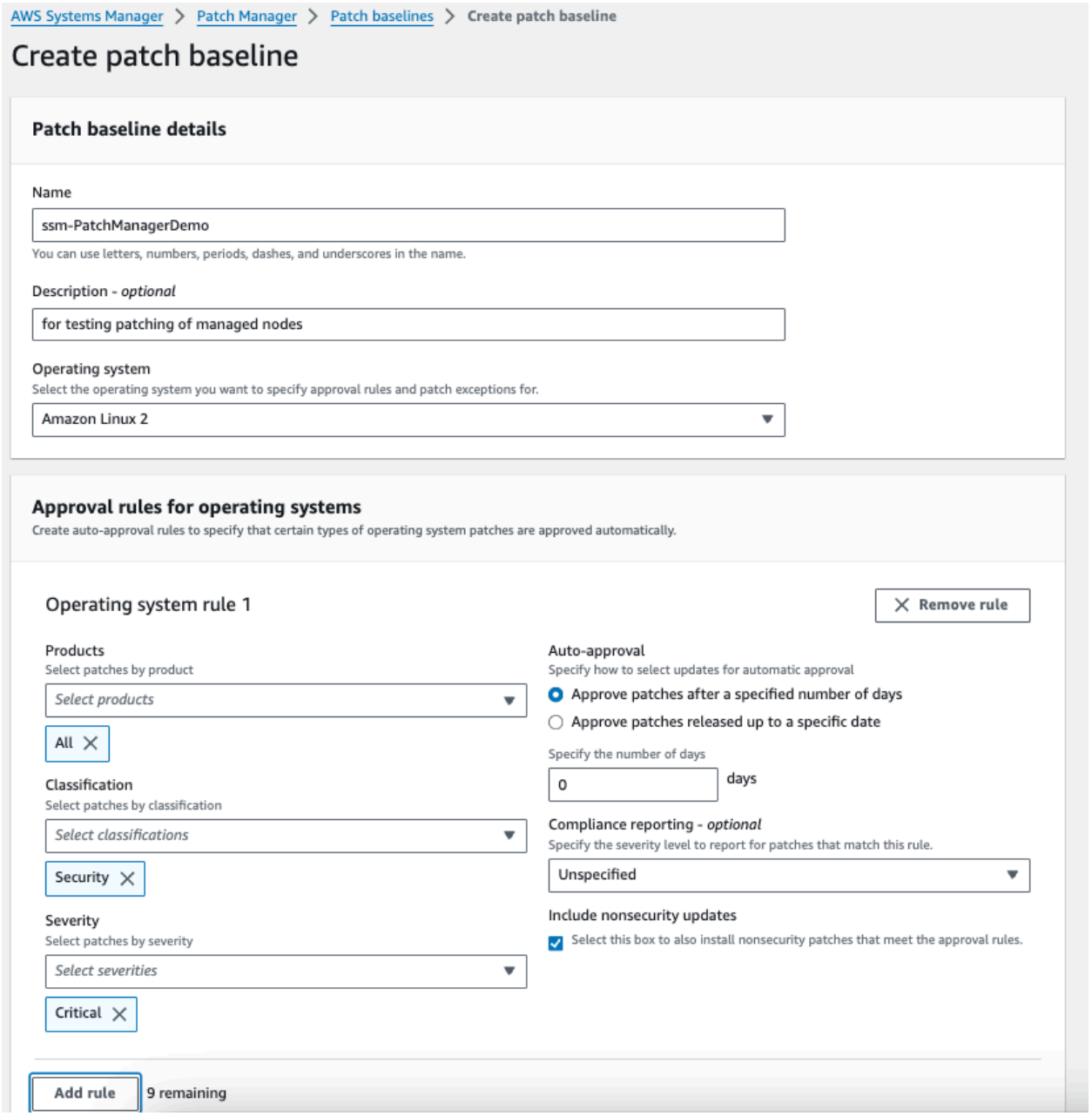
1. **Tag Instances:**
    - Ensure each instance is tagged with `Patch Group=TestGroup`.
2. **Create Patch Baseline:**
    - **Navigate to Patch Manager:**
        - Go to Systems Manager > Patch Manager.
    - **Create a Patch Baseline:**
        - Click on "Start with an overview"

- select **Patch baselines** and click on "Create patch baseline."



- Provide a name (e.g. ssm-PatchManagerDemo) and description for the baseline.
- Select the operating system (e.g., Amazon Linux, CentOS, RHEL).

- Define the patch rules, such as including all patches or only critical/security patches, etc.

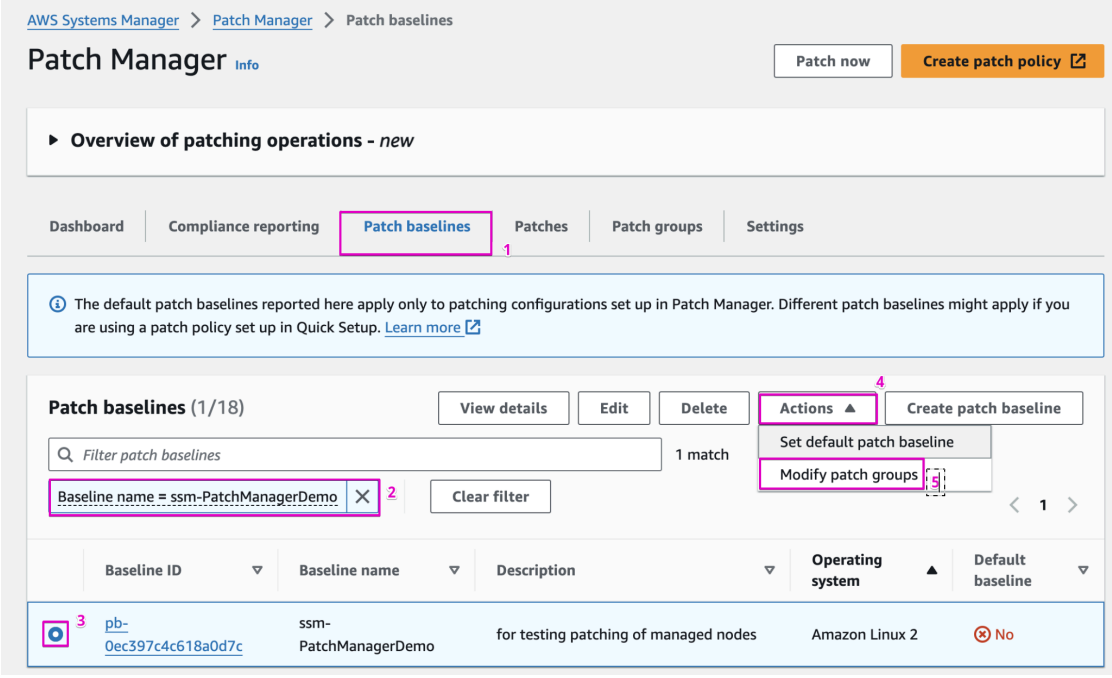


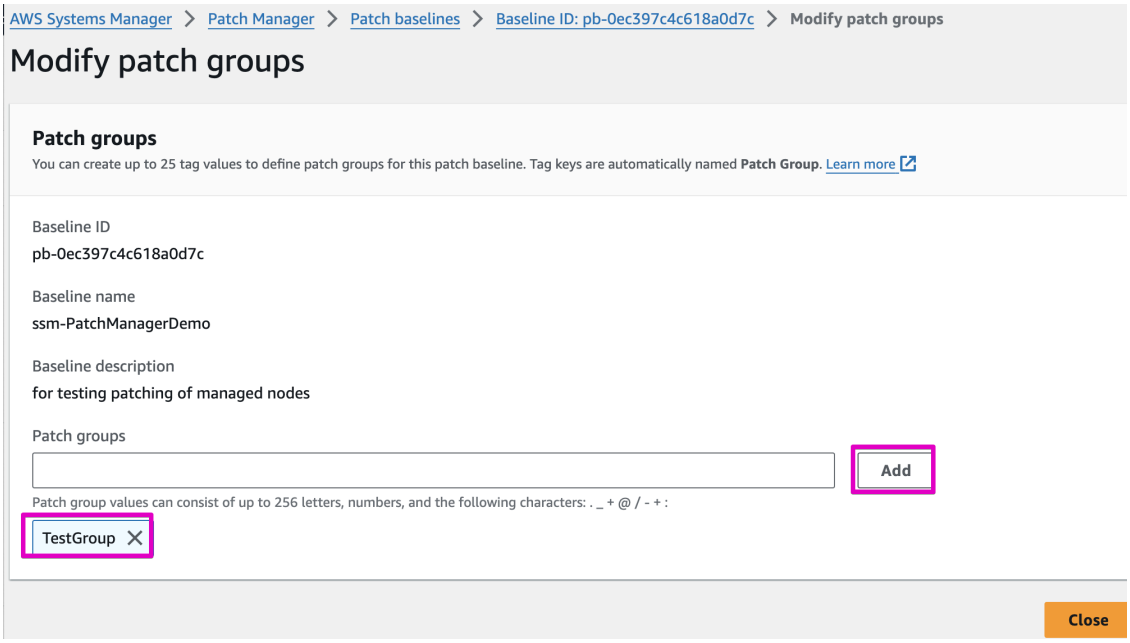
- Scroll down and create the baseline.

o **Set Baseline as Default Baseline**:
If Patch Manager was accessed in a region prior to December 2022, you can set a baseline as the default baseline during or after creation using both console and CLI
In regions accessed after Dec 2022, only the CLI option is possible using the command below.

aws ssm register-default-patch-baseline --baseline-id *baseline-id-or-ARN*

3. **Assign Patch Baseline to Patch Group:**
    - **Navigate to the Patch Baseline:**
        - In the Patch Manager console, select the patch baseline you created earlier.
    - **Assign to Patch Group:**
        - click on *Actions* and *modify the patch baseline* by adding the tag value `TestGroup` assigned to the tag key "PatchGroup"





If the patch **baseline ID** details page does not include an ***Actions*** menu, patch groups can't be configured in the console, use the CLI with the below command

```
aws ssm register-patch-baseline-for-patch-group  --baseline-id "value" --patch-group "value"
```

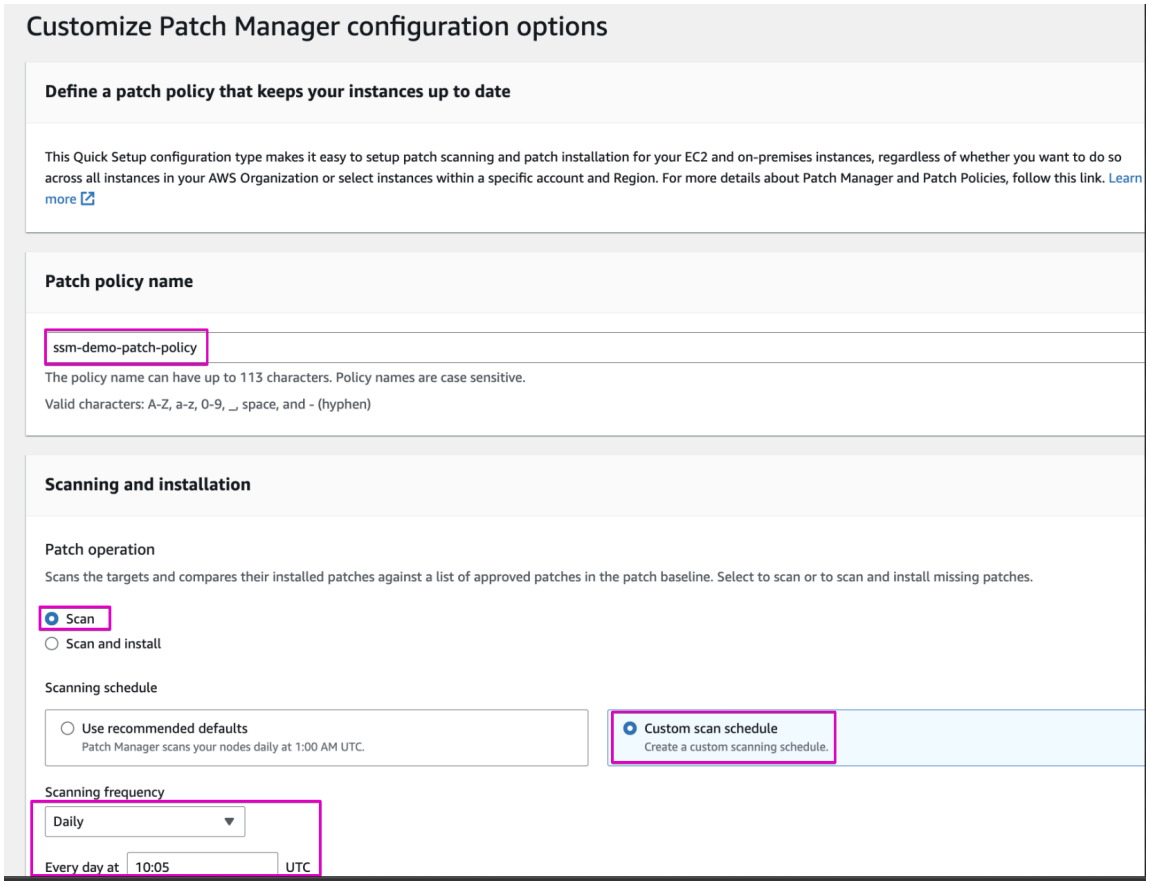
e.g

```
aws ssm register-patch-baseline-for-patch-group \
  --baseline-id "pb-0e2ba677964f29e91" \
  --patch-group "TestGroup" \
  --region us-east-1
```

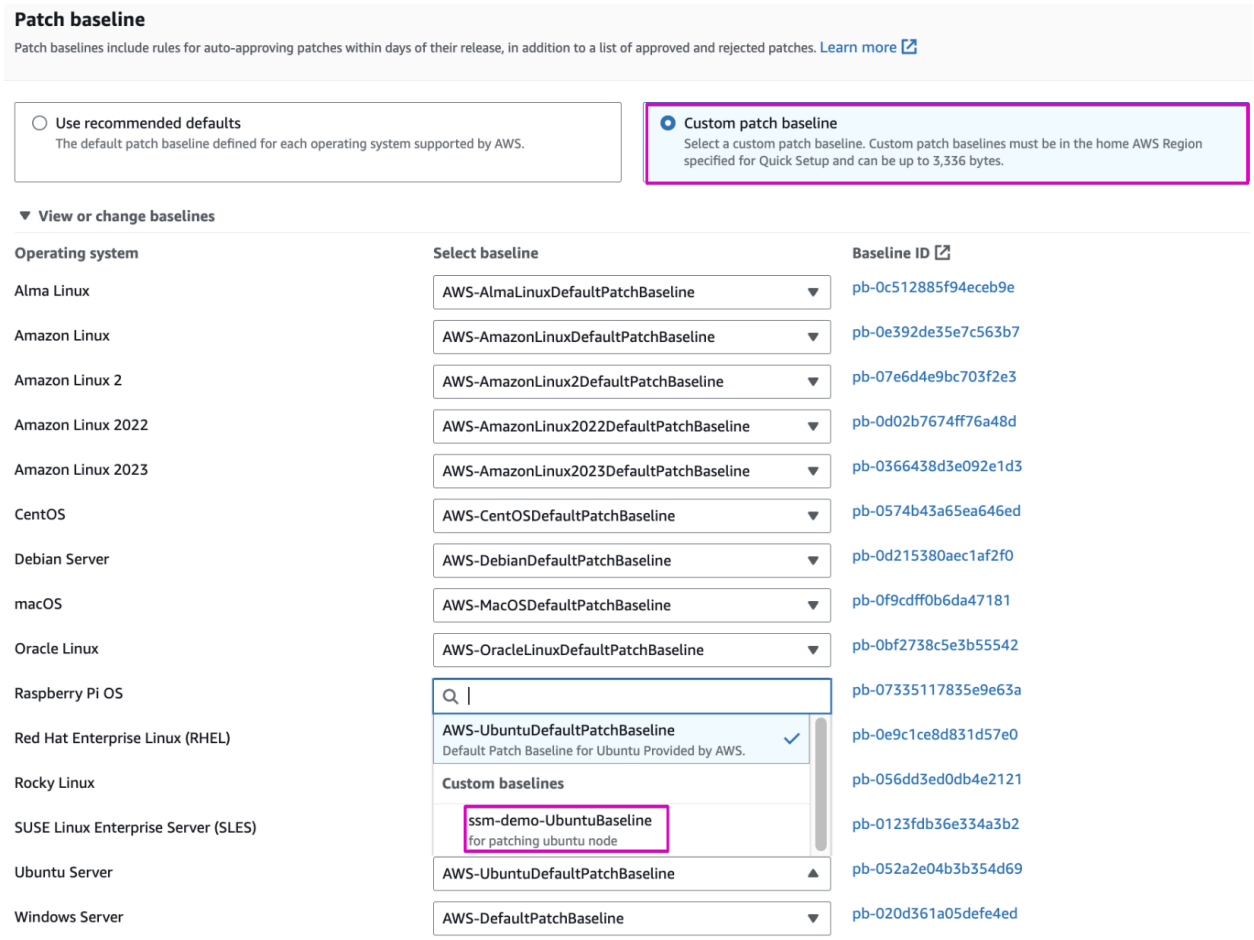
4. **Patch Using Patch Policy:**
   From DEC 2022 AWS recommends using **patch policies** to patch managed nodes.

   - **create patch policy**
     - Navigate the SSM console and select "Patch Manager"
     - select create patch policy
     - configure patching
     - Under Patch Baseline section, select the custom baseline created earlier

   - configure patching:
     1. name: ssm-demo-patch-policy
     2. patch operation: Scan or Scan and install
     3. scanning schedule: custom scan schedule
     4. scanning frequency: Daily, and enter suitable time in UTC

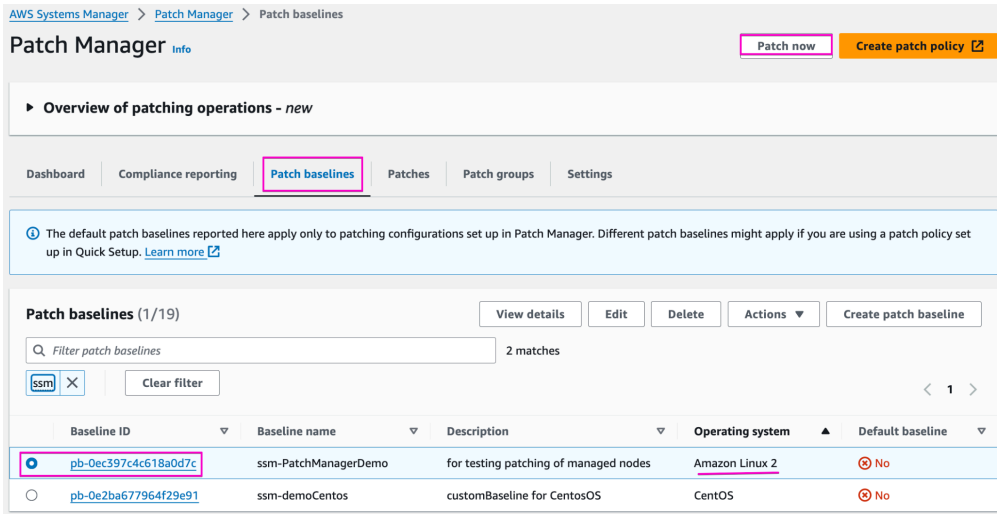


- Under Patch Baseline, select the custom baseline created earlier

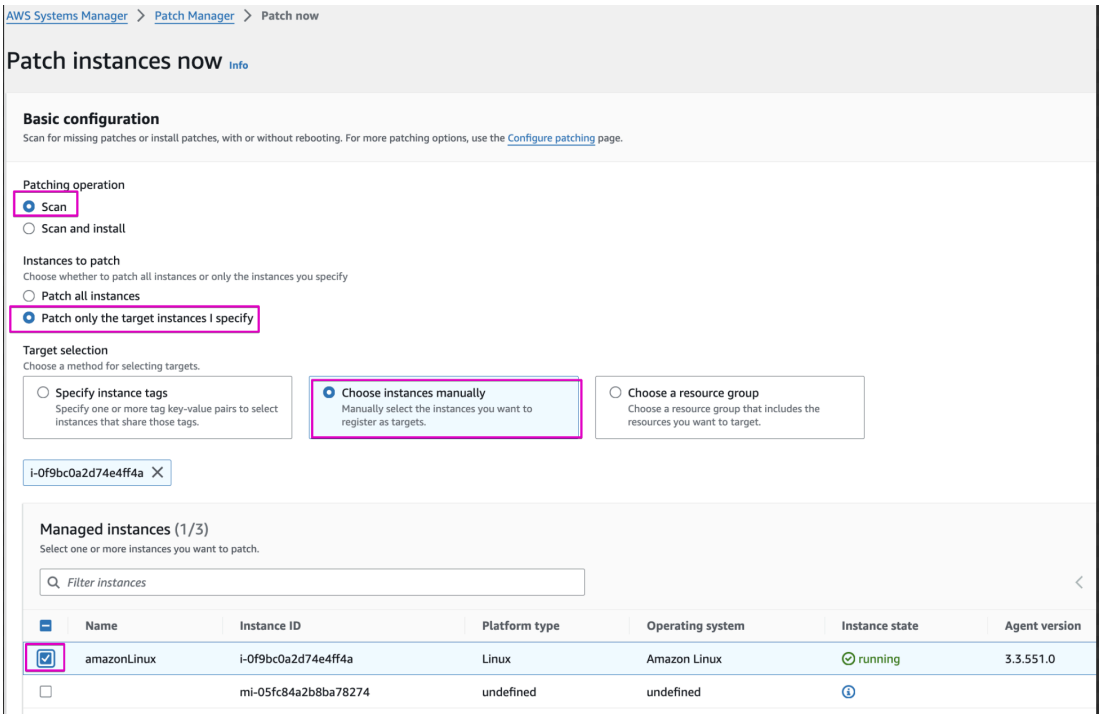


- Under Targets, select resource groups or use all managed nodes
- Review and create policy

5. **Run Patching:**
   - **Navigate to Patch Now:**
     - In the Patch Manager console, select "Patch now."
   - **Select the Patch Baseline:**
     - Choose the patch baseline you created.



- **Target Instances:**
  - Select the instances individually or use the tag-based patch group.
- **Run the Patching:**
  - Execute the patching process and monitor its progress to ensure patches are applied successfully.



click on Patch Now

6. **(Optional- legacy) Maintenance Window:**
    - Create a maintenance window to understand how it can be used with Patch Manager.
    - **Create a Maintenance Window:**
        - Navigate to Systems Manager > Maintenance Windows.
        - Click on "Create maintenance window."
        - Provide a name, schedule (e.g., weekly on Sunday at 2 AM), duration, and cutoff time.
        - Register the target instances using the `Patch Group=TestGroup` tag.
        - Register a task to apply patches using the patch baseline.

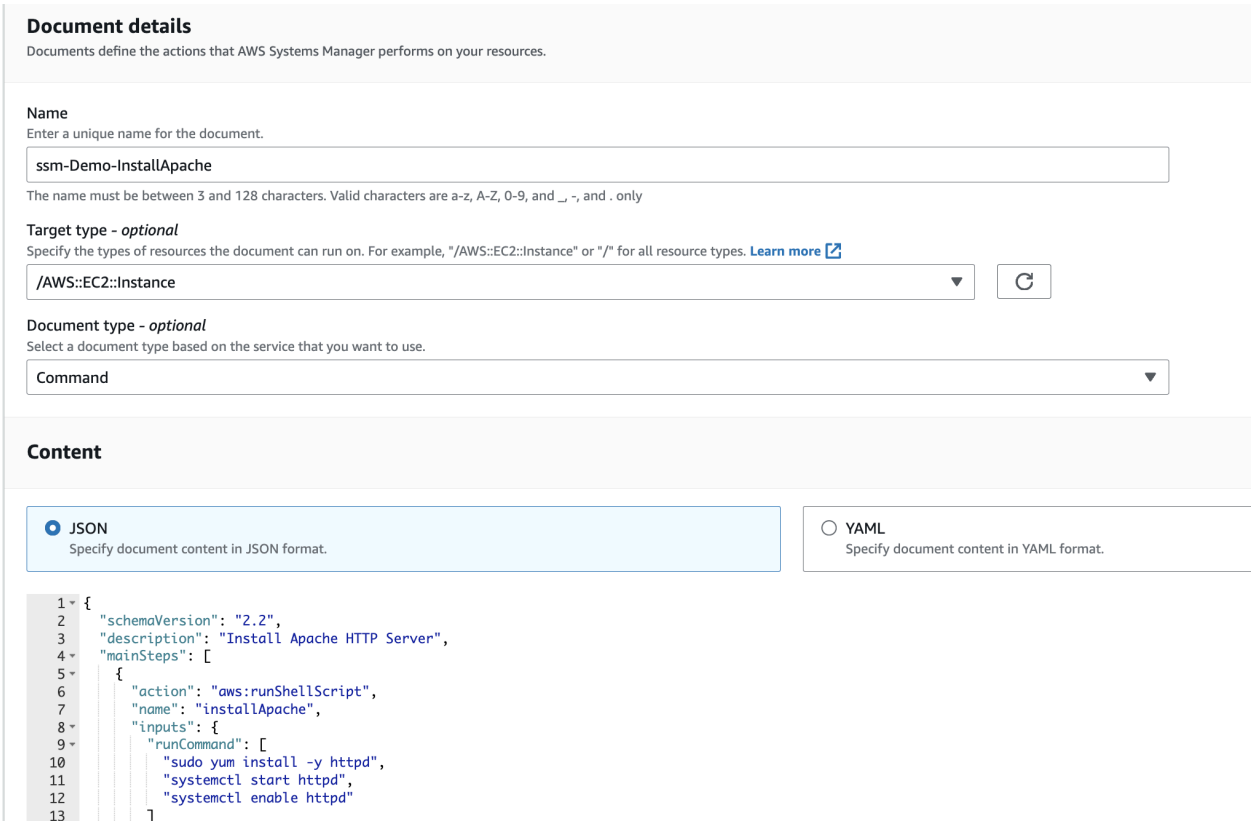
5. Run Command:

Goal: Use Run Command to execute commands across instances simultaneously.

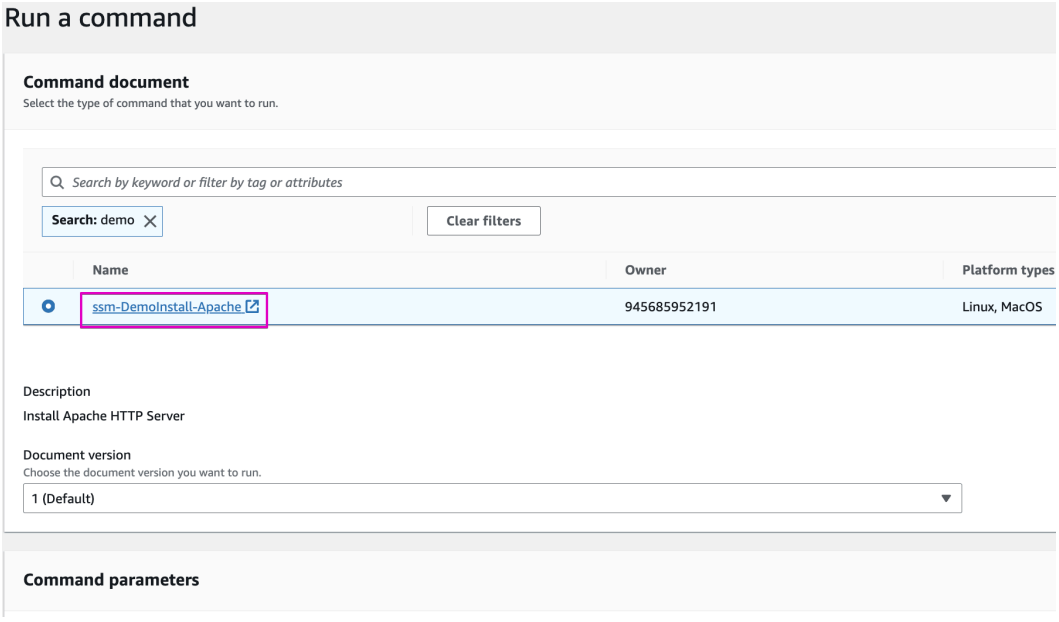
1. **Create a Run Command Document:**
    - **Navigate to Run Command:**
        - Go to Systems Manager > Shared Resources.
    - **Create a New Document:**
        - Click on "Create document."
        - Choose "Command document."
        - Provide a name (ssm-Demo-InstallApache) and description for the document.
        - for target type, select /**AWS::EC2::Instance**
        - document type(optional): select **command**
        - In the content section, use the following script to install Apache:

```
{
  "schemaVersion": "2.2",
  "description": "Install Apache HTTP Server",
  "mainSteps": [
    {
      "action": "aws:runShellScript",
      "name": "installApache",
      "inputs": {
        "runCommand": [
          "sudo yum install -y httpd",
          "systemctl start httpd",
          "systemctl enable httpd"
        ]
      }
    }
  ]
}
```

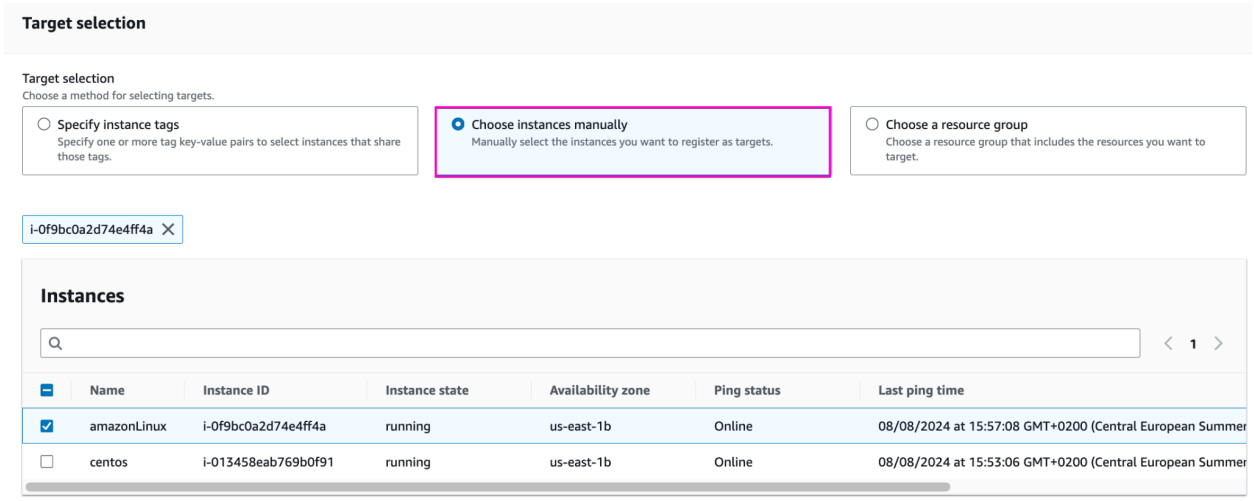
        - Save and create the document.



2. **Execute Run Command:**
    - **Select the Document:**
        - In Run Command, select the created document.



- **Specify Targets:**
  - Select Instance manually or Choose the resource group created earlier.

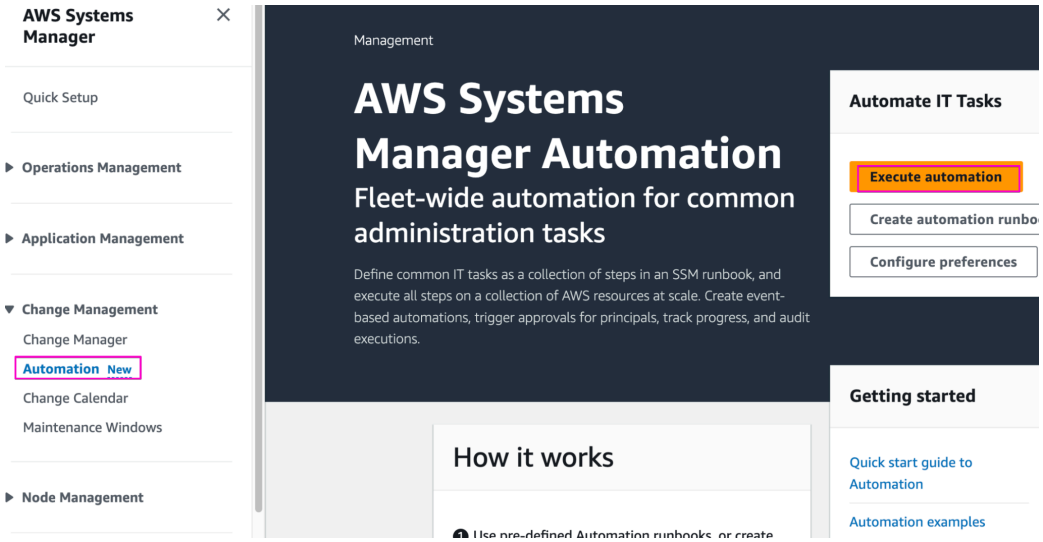


- **Run the Command:**
  - Click run to Execute the command and monitor its progress.

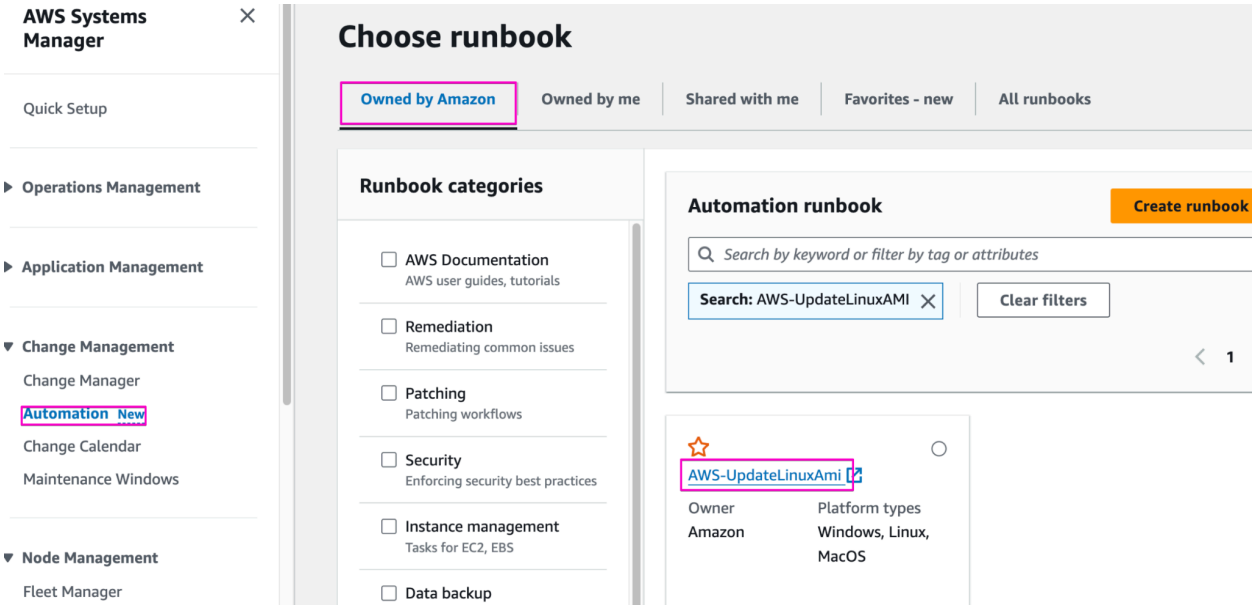
6. Automation:

Goal: Use SSM Automation to automate common tasks using AWS-provided automation documents (runbooks).

1. **Execute an Automation Document:**
   - **Navigate to Automation:**
     - Go to Systems Manager , and under Change Management, click on Automation.



- **Start Automation Execution:**
  - Click on "Execute automation."
  - Choose AWS-provided runbooks, and select `AWS-UpdateLinuxAMI` and click on Next.



  - In the Execution mode section, choose **Simple Execution**,
  - In the Input parameters section, provide the required parameters (
    1. source AMI id
    2. create AutomationRole for systems Manager service name **ssmAdmin** (create role with *Admin* or *AmazonSSMAutomationRole* policy)
    3. IamInstanceProfile Name: role with EC2-service that gives SSM necessary permissions.
    4. target AMI name etc

Input parameters

SourceAmiId
(Required) The source Amazon Machine Image ID.
ami-03972092c42e8c0ca

IamInstanceProfileName
(Required) The instance profile that enables Systems Manager (SSM) to manage the instance.
ManagedInstanceProfile

AutomationAssumeRole
(Required) The ARN of the role that allows Automation to perform the actions on your behalf.
ssmAdmin

TargetAmiName
(Optional) The name of the new AMI that will be created. Default is a system-generated string including the source AMI id, and the creation time and date.
DemoSSM-UpdateLinuxAmi_from_{{SourceAmiId}}_on_{{global:DATE_TIME}}

InstanceType
(Optional) Type of instance to launch as the workspace host. Instance types vary by region. Default is t3.micro.
t2.micro

SecurityGroupIds
(Optional) A comma separated list of security group IDs with the required Inbound and Outbound connectivity rules.
StringList

SubnetId
(Optional) Specify the SubnetId if you want to launch EC2 instance in a specific subnet.
String

PreUpdateScript
(Optional) URL of a script to run before updates are applied. Default ("none") is to not run a script.
none

PostUpdateScript
(Optional) URL of a script to run after package updates are applied. Default ("none") is to not run a script.
none

IncludePackages
(Optional) Only update these named packages. By default ("all"), all available updates are applied.
all

ExcludePackages
(Optional) Names of packages to hold back from updates, under all conditions. By default ("none"), no package is excluded.
none

MetadataOptions
(Optional) The metadata options for the instance.
{"HttpEndpoint":"enabled","HttpTokens":"optional"}

- Execute the automation and monitor its progress under **Executed Steps**

Execution status

Overall status	All executed steps	# Succeeded
In Progress	3	3
# Failed	**# Cancelled**	**# TimedOut**
0	0	0

Executed steps (6)

Find Steps

1

Step ID	Step #	Step name	Action	Status	Start time	End time
9932ff2b-abd8-41d5-90d4-bafadfdc6d69	1	launchInstance	aws:runInstances	Success	Thu, 08 Aug 2024 18:02:37 GMT	Thu, 08 Aug 2024 18:06:03 GMT
15921ef8-c7c2-4a7e-be4a-eff654512353	2	verifySsmInstall	aws:runCommand	Success	Thu, 08 Aug 2024 18:06:03 GMT	Thu, 08 Aug 2024 18:06:06 GMT
a2c52d87-3118-454b-a49e-82c607f4c67f	3	updateOSSoftware	aws:runCommand	Success	Thu, 08 Aug 2024 18:06:06 GMT	Thu, 08 Aug 2024 18:06:17 GMT
dd6c2d49-f6eb-4aad-b041-20216b215e18	4	stopInstance	aws:changeInstanceState	In Progress	Thu, 08 Aug 2024 18:06:17 GMT	-
ff0e81fa-a32e-4c35-808a-eb4cf2d85755	5	createImage	aws:createImage	Pending	-	-
373c147e-3651-4a9c-a1b5-e49c1fdf998a	6	terminateInstance	aws:changeInstanceState	Pending	-	-

7. State Manager:

Goal: Use State Manager to ensure instances are in a consistent state.

1. **Create a State Manager Association:**
    - **Navigate to State Manager:**
        - Go to Systems Manager > State Manager.
    - **Create Association:**
        - Click on "Create association."
        - Choose a document, such as `AWS-UpdateSSMAgent` to ensure the SSM agent is always running.
        - Provide association details, such as targets (e.g., instances tagged with `Environment=Test`).
        - Configure the schedule and parameters as needed.
        - Create the association.
2. **Monitor Compliance:**
    - **Check Compliance Status:**
        - In State Manager, monitor the compliance status of the instances.
        - Ensure that the SSM agent remains active and up-to-date.