# INVENTORY AND PATCH MANAGEMENT

## Introduction

In the cloud, you can apply the same engineering discipline that you use for application code to your entire environment. You can define your entire workload (applications, infrastructure, etc.) as code and update it with code. You can script your operations procedures and automate their execution by triggering them in response to events. By performing operations as code, you limit human error and enable consistent execution of operations activities.

In this lab you will apply the concepts of *Infrastructure as Code* and *Operations as Code* to the following activities:

* Deployment of Infrastructure
* Inventory Management
* Patch Management

### **1.1 Deploy the Lab Infrastructure**

To deploy the lab infrastructure:

1. Use your administrator account to access the CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. Choose **Create Stack**.
3. On the **Select Template** page, select **Upload a template file** and select the OE\_Inventory\_and\_Patch\_Mgmt.json file you just downloaded.

**AWS CloudFormation Designer**

AWS [CloudFormation Designer](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/working-with-templates-cfn-designer.html) is a graphic tool for creating, viewing, and modifying AWS CloudFormation templates. With Designer you can diagram your template resources using a drag-and-drop interface. You can edit their details using the integrated JSON and YAML editor. AWS CloudFormation Designer can help you see the relationship between template resources.

1. On the **Select Template** page, in the lower-right corner, click the **View in Designer** button.
2. Briefly review the graphical representation of the environment we are about to create, including the template in the JSON and YAML formats. You can use this feature to convert between JSON and YAML formats.
3. **Choose the Create Stack icon** (a cloud with an arrow) to return to the **Select Template page**.
4. On the **Select Template** page, choose **Next**.

A CloudFormation template is a JSON or YAML formatted text file that describes your AWS infrastructure [containing both optional and required sections](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/template-anatomy.html) . In the next steps, we will provide a name for our stack and parameters that will be passed into the template to help define the resources that will be implemented.

1. In the **Specify Details** section, define a **Stack name**, such as OELabStack1.
2. In the **Parameters** section:
   1. Leave **InstanceProfile** blank as we have not yet defined an instance profile.
   2. Leave **InstanceTypeApp** and **InstanceTypeWeb** as the default free-tier-eligible t2.micro value.
   3. Select the EC2 **KeyName** you defined earlier from the list.
   4. In a browser window, go to <https://checkip.amazonaws.com/> to get your IP. Enter your IP address in **SSHLocation** in CIDR notation (i.e., ending in /32).
   5. Define the **Workload Name** as Test.
   6. Choose **Next**.
3. On the **Options** page under **Tags**, define a **Key** of **Owner**, with **Value** set to the username you choose for your administrator. You may define additional keys as needed. The CloudFormation template creates all the example tags given in the discussion on tagging above.
4. Leave all other sections unmodified. Scroll to the bottom of the page and choose **Next**.
5. On the **Review** page, review your choices and then choose **Create**.
6. On the CloudFormation console page
   1. **Check the box next to your Stack Name** to see its details.
   2. If your **Stack Name** is not displayed, click the **refresh** button (circular arrow) in the top right until it appears.
   3. If the details are not displayed, choose the refresh button until details appear.
7. Choose the **Events** tab for your selected workload to see the activity log from the creation of your CloudFormation stack.

When the **Status** of your stack displays **CREATE\_COMPLETE** in the filter list, you have just created a representation of a typical lift and shift 2-tier application migrated to the cloud.

1. Navigate to the [EC2 console](https://console.aws.amazon.com/ec2/) to view the deployed systems:
   1. Choose **Instances**.
   2. Select a server and review the details under its **Description** and **Tag** tabs.
   3. (Optional) choose **Security Groups** and select the Security Group whose name begins with the name of your stack. Examine the inbound rules.
   4. (Optional) navigate to the VPC console and examine the configuration of the VPC you just created.

## **The impact of Infrastructure as Code**

With infrastructure as code, if you can deploy one environment, you can deploy any number of copies of that environment. In this example we have created a Test environment. Later, we will repeat these steps to deploy a Prod environment.

The ability to dynamically deploy temporary environments on-demand enables parallel experimentation, development, and testing efforts. It allows duplication of environments to recreate and analyze errors, as well as cut-over deployment of production systems using blue-green methodologies. These practices contribute to reduced risk, increased operations effectiveness, and efficiency.

# INVENTORY MANAGEMENT USING OPERATIONS AS CODE

## **Management Tools: Systems Manager**

[AWS Systems Manager](https://aws.amazon.com/systems-manager/features/) is a collection of features that enable IT Operations that we will explore throughout this lab.

There are [set up tasks](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-setting-up.html) and [pre-requisites](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-prereqs.html) that must be satisfied prior to using Systems Manager to manage your EC2 instances or on-premises systems in [hybrid environments](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-managedinstances.html) .

* You must use a supported operating system
  + Supported operating systems include versions of Windows, Amazon Linux, Ubuntu Server, RHEL, and CentOS
* The SSM Agent must be installed
  + The SSM Agent for Windows also requires PowerShell 3.0 or later to run some [SSM documents](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-prereqs.html#prereqs-powershell)
* Your EC2 instances must have outbound internet access
* You must access Systems Manager in a supported region
* Systems Manager requires IAM roles
  + for instances that will process commands
  + for users executing commands

SSM Agent is installed by default on:

* Amazon Linux base AMIs dated 2017.09 and later
* Windows Server 2016 instances
* Instances created from Windows Server 2003-2012 R2 AMIs published in November 2016 or later

[There is no additional charge for AWS Systems Manager](https://aws.amazon.com/systems-manager/pricing/) . You only pay for your underlying AWS resources managed or created by AWS Systems Manager (e.g., Amazon EC2 instances or Amazon CloudWatch metrics). You only pay for what you use as you use it. There are no minimum fees and no upfront commitments.

### **2.1 Setting up Systems Manager**

1. Use your administrator account to access the Systems Manager console at <https://console.aws.amazon.com/systems-manager/>.
2. Choose **Managed Instances** from the navigation bar. If you have not satisfied the pre-requisites for Systems Manager, you will arrive at the **AWS Systems Manager Managed Instances** page.
   * As a user with AdministratorAccess permissions, you already have [User Access to Systems Manager](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-access-user.html) .
   * The Amazon Linux AMIs used to create the instances in your environment are dated 2017.09. They are [supported operating systems](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-manager-supported-oses.html) and have the [SSM Agent](https://docs.aws.amazon.com/systems-manager/latest/userguide/ssm-agent.html) installed by default.
   * If you are in a [supported region](https://docs.aws.amazon.com/general/latest/gr/rande.html#ssm_region) the remaining step is to configure the IAM role for instances that will process commands.
3. Create an Instance Profile for Systems Manager managed instances:
   * Navigate to the [IAM console](https://console.aws.amazon.com/iam/)
   * In the navigation pane, choose **Roles**.
   * Then choose **Create role**.
   * In the **Select type of trusted entity** section, verify that the default **AWS service** is selected.
   * In the **Choose the service that will use this role** section, scroll past the first reference to EC2 (**EC2 Allows EC2 instances to call AWS services on your behalf**) and choose **EC2** from within the field of services. This will open the **Select your use case** section further down the page.
   * In the **Select your use case** section, choose **EC2 Role for Simple Systems Manager** to select it.
   * Then choose **Next: Permissions**.
4. Under **Attached permissions policy**, verify that **AmazonEC2RoleforSSM** is listed, and then choose **Next: Review**.
5. In the **Review** section:
   * Enter a **Role name**, such as ManagedInstancesRole.
   * Accept the default in the **Role description**.
   * Choose **Create role**.
6. Apply this role to the instances you wish to manage with Systems Manager:
   * Navigate to the [EC2 Console](https://console.aws.amazon.com/ec2/) and choose **Instances**.
   * Select the first instance and then choose **Actions**, **Security**, and **Modify IAM Role**.
   * Under **Modify IAM Role**, select **ManagedInstancesRole** from the drop down list and choose **Save**.
   * Repeat this process, assigning **ManagedInstancesRole** to each of the 3 remaining instances.
7. Return to the [Systems Manager console](https://console.aws.amazon.com/systems-manager/) and choose **Managed Instances** from the navigation bar. Periodically choose **Managed Instances** until your instances begin to appear in the list. Over the next couple of minutes your instances will populate into the list as managed instances.

**Note** If desired, you can use a [more restrictive permission set](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-access-user.html) to grant access to Systems Manager.

### **2.2 Create a Second CloudFormation Stack**

1. Create a second CloudFormation stack using the procedure in 1.1 with the following changes(You can delete the previously created stack):
   * In the **Specify Details** section, define a Stack name, such as OELabStack2.
   * Specify the **InstanceProfile** using the ManagedInstancesRole you defined.
   * Define the **Workload Name** as Prod.

## **Systems Manager: Inventory**

You can use [AWS Systems Manager Inventory](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-inventory.html) to collect operating system (OS), application, and instance metadata from your Amazon EC2 instances and your on-premises servers or virtual machines (VMs) in your hybrid environment. You can query the metadata to quickly understand which instances are running the software and configurations required by your software policy, and which instances need to be updated.

### **2.3 Using Systems Manager Inventory to Track Your Instances**

1. Under **Instances & Nodes** in the AWS Systems Manager navigation bar, choose **Inventory**.
   1. Scroll down in the window to the **Corresponding managed instances** section. Inventory currently contains only the instance data available from the EC2
   2. Choose the **InstanceID** of one of your systems.
   3. Examine each of the available tabs of data under the **Instance ID** heading.
2. Inventory collection must be specifically configured and the data types to be collected must be specified
   1. Choose **Inventory** in the navigation bar.
   2. Choose **Setup Inventory** in the top right corner of the window
3. In the **Setup Inventory** screen, define targets for inventory:
   1. Under **Specify targets by**, select **Specifying a tag**
   2. Under **Tags** specify Environment for the key and OELabIPM for the value

**Note** You can select all managed instances in this account, ensuring that all managed instances will be inventoried. You can constrain inventoried instances to those with specific tags, such as Environment or Workload. Or you can manually select specific instances for inventory.

1. Schedule the frequency with which inventory is collected. The default and minimum period is 30 minutes
   1. For **Collect inventory data every**, accept the default **30** Minute(s)
2. Under parameters, specify what information to collect with the inventory process
   1. Review the options and select the defaults
3. (Optional) If desired, you may specify an S3 bucket to receive the inventory execution logs (you will need to [create a destination bucket for the logs](https://docs.aws.amazon.com/AmazonS3/latest/gsg/CreatingABucket.html) prior to proceeding):
   1. Check the box next to **Sync inventory execution logs to an S3 bucket** under the **Advanced** options.
   2. Provide an S3 bucket name.
   3. (Optional) Provide an S3 bucket prefix.
4. Choose **Setup Inventory** at the bottom of the page (it can take up to 10 minutes to deploy a new inventory policy to an instance).
5. To create a new inventory policy, from **Inventory**, choose **Setup inventory**.
6. To edit an existing policy, from **State Manager** in the left navigation menu, select the association and choose **Edit**.

**Note** You can create multiple Inventory specifications. They will each be stored as **associations** within **Systems Manager State Manager**.

## **Systems Manager: State Manager**

In State Manager, an [association](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-associations.html) is the result of binding configuration information that defines the state you want your instances to be in to the instances themselves. This information specifies when and how you want instance-related operations to run that ensure your Amazon EC2 and hybrid infrastructure is in an intended or consistent state.

An association defines the state you want to apply to a set of targets. An association includes three components and one optional set of components:

* A document that defines the state
* Target(s)
* A schedule
* (Optional) Runtime parameters.

When you performed the **Setup Inventory** actions, you created an association in State Manager.

### **2.4 Review Association Status**

1. Under **Actions** in the navigation bar, select **State Manager**. At this point, the **Status** may show that the inventory activity has not yet completed.
   1. Choose the single Association id that is the result of your **Setup Inventory** action.
   2. Examine each of the available tabs of data under the **Association ID** heading.
   3. Choose **Edit**.
   4. Enter a name under **Name - optional** to provide a more user friendly label to the association, such as InventoryAllInstances (white space is not permitted in an Association Name).

Inventory is accomplished through the following:

* The activities defined in the AWS-GatherSoftwareInventory command document.
* The parameters provided in the **Parameters** section are passed to the document at execution.
* The targets are defined in the **Targets** section.

**Important** In this example there is a single target, the wildcard. The wildcard matches all instances making them all targets.

* The schedule for this activity is defined under **Specify schedule** and **Specify with** to use a CRON/Rate expression on a 30 minute interval.
* There is the option to specify **Output options**.

**Note** If you change the command document, the **Parameters** section will change to be appropriate to the new command document.

1. Navigate to **Managed Instances** under **Instances and Nodes** in the navigation bar. An **Association Status** has been established for the inventoried instances under management.
2. Choose one of the **Instance ID** links to go to the inventory of the instance. The Inventory tab is now populated and you can track associations and their last activity under the Associations tab.
3. Navigate to **Compliance** under **Instances & Nodes** in the navigation bar. Here you can view the overall compliance status of your managed instances in the **Compliance Summary** and the individual compliance status of systems in the **Corresponding managed instances** section below.

**Note** The inventory activity can take up to 10 minutes to complete. While waiting for the inventory activity to complete, you can proceed with the next section.

## **Systems Manager: Compliance**

You can use AWS Systems Manager Configuration [Compliance](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-compliance.html) to scan your fleet of managed instances for patch compliance and configuration inconsistencies. You can collect and aggregate data from multiple AWS accounts and Regions, and then drill down into specific resources that aren’t compliant.

By default, Configuration Compliance displays compliance data about Systems Manager Patch Manager patching and **Systems Manager State Manager** associations. You can also customize the service and create your own compliance types based on your IT or business requirements. You can also port data to **Amazon Athena** and **Amazon QuickSight** to generate fleet-wide reports.

# PATCH MANAGEMENT

## **Systems Manager: Patch Manager**

AWS Systems Manager [Patch Manager](https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-patch.html) automates the process of patching managed instances with security related updates.

**Note** For Linux-based instances, you can also install patches for non-security updates.

You can patch fleets of Amazon EC2 instances or your on-premises servers and virtual machines (VMs) by operating system type. This includes supported versions of Windows, Ubuntu Server, Red Hat Enterprise Linux (RHEL), SUSE Linux Enterprise Server (SLES), and Amazon Linux. You can scan instances to see only a report of missing patches, or you can scan and automatically install all missing patches. You can target instances individually or in large groups by using Amazon EC2 tags.

**Warning**

* **AWS does not test patches for Windows or Linux before making them available in Patch Manager** .
* **If any updates are installed by Patch Manager the patched instance is rebooted**.
* **Always test patches thoroughly before deploying to production environments**.

## **Patch Baselines**

Patch Manager uses **patch baselines**, which include rules for auto-approving patches within days of their release, as well as a list of approved and rejected patches. Later in this lab we will schedule patching to occur on a regular basis using a Systems Manager **Maintenance Window** task. Patch Manager integrates with AWS Identity and Access Management (IAM), AWS CloudTrail, and Amazon CloudWatch Events to provide a secure patching experience that includes event notifications and the ability to audit usage.

**Warning** The [operating systems supported by Patch Manager](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-manager-supported-oses.html) may vary from those supported by the SSM Agent.

### **3.1 Create a Patch Baseline**

1. Under **Instances and Nodes** in the **AWS Systems Manager** navigation bar, choose **Patch Manager**.
2. Click the **View predefined patch baselines** link under the **Configure patching** button on the upper right.
3. Choose **Create patch baseline**.
4. On the **Create patch baseline** page in the **Provide patch baseline details** section:
   1. Enter a **Name** for your custom patch baseline, such as AmazonLinuxSecAndNonSecBaseline.
   2. Optionally enter a description, such as Amazon Linux patch baseline including security and non-security patches.
   3. Select **Amazon Linux** from the list.
5. In the **Approval rules** section:
   1. Examine the options in the lists and ensure that **Product**, **Classification**, and **Severity** have values of **All**.
   2. Leave the **Auto approval delay** at its default of **0 days**.
   3. Change the value of **Compliance reporting - optional** to **Critical**.
   4. Choose **Add another rule**.
   5. In the new rule, change the value of **Compliance reporting - optional** to **Medium**.
   6. Check the box under **Include non-security updates** to include all Amazon Linux updates when patching.

If an approved patch is reported as missing, the option you choose in **Compliance reporting**, such as Critical or Medium, determines the severity of the compliance violation reported in System Manager **Compliance**.

1. In the **Patch exceptions** section in the **Rejected patches - optional** text box, enter system-release.\* This will [reject patches](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-manager-approved-rejected-package-name-formats.html) to new Amazon Linux releases that may advance you beyond the [Patch Manager supported operating systems](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-manager-supported-oses.html) prior to your testing new releases.
2. For Linux operating systems, you can optionally define an [alternative patch source repository](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-manager-how-it-works-alt-source-repository.html) . Choose the **X** in the **Patch sources** area to remove the empty patch source definition.
3. Choose **Create patch baseline** and you will go to the **Patch Baselines** page where the AWS provided default patch baselines, and your custom baseline, are displayed.

## **Patch Groups**

A [patch group](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html) is an optional method to organize instances for patching. For example, you can create patch groups for different operating systems (Linux or Windows), different environments (Development, Test, and Production), or different server functions (web servers, file servers, databases). Patch groups can help you avoid deploying patches to the wrong set of instances. They can also help you avoid deploying patches before they have been adequately tested.

You create a patch group by using Amazon EC2 tags. Unlike other tagging scenarios across Systems Manager, a patch group must be defined with the tag key: Patch Group (tag keys are case sensitive). You can specify any value (for example, web servers) but the key must be Patch Group.

**Note** An instance can only be in one patch group.

After you create a patch group and tag instances, you can register the patch group with a patch baseline. By registering the patch group with a patch baseline, you ensure that the correct patches are installed during the patching execution. When the system applies a patch baseline to an instance, the service checks if a patch group is defined for the instance.

* If the instance is assigned to a patch group, the system checks to see which patch baseline is registered to that group.
* If a patch baseline is found for that group, the system applies that patch baseline.
* If an instance isn’t assigned to a patch group, the system automatically uses the currently configured default patch baseline.

### **3.2 Assign a Patch Group**

1. Choose the **Baseline ID** of your newly created baseline to enter the details screen.
2. Choose **Actions** in the top right of the window and select **Modify patch groups**.
3. In the **Modify patch groups** window under **Patch groups**, enter Critical, choose **Add**, and then choose **Close** to be returned to the **Patch Baseline** details screen.

## **AWS-RunPatchBaseline**

[AWS-RunPatchBaseline](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-manager-ssm-documents.html#patch-manager-ssm-documents-recommended-AWS-RunPatchBaseline) is a command document that enables you to control patch approvals using patch baselines. It reports patch compliance information that you can view using the Systems Manager **Compliance** tools. For example,you can view which instances are missing patches and what those patches are.

For Linux operating systems, compliance information is provided for patches from both the default source repository configured on an instance and from any alternative source repositories you specify in a custom patch baseline. AWS-RunPatchBaseline supports both Windows and Linux operating systems.

## **AWS Systems Manager: Document**

An [AWS Systems Manager document](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-ssm-docs.html) defines the actions that Systems Manager performs on your managed instances. Systems Manager includes many pre-configured documents that you can use by specifying parameters at runtime, including ‘AWS-RunPatchBaseline’. These documents use JavaScript Object Notation (JSON) or YAML (a recursive acronym for “YAML Ain’t Markup Language”), and they include steps and parameters that you specify.

All AWS provided Automation and Run Command documents can be viewed in AWS Systems Manager **Documents**. You can [create your own documents](https://docs.aws.amazon.com/systems-manager/latest/userguide/create-ssm-doc.html) or launch existing scripts using provided documents to implement custom operations as code activities.

### **3.3 Examine AWS-RunPatchBaseline in Documents**

To examine AWS-RunPatchBaseline in Documents:

1. In the AWS Systems Manager navigation bar under **Shared Resources**, choose **Documents**.
2. Click in the **search box**, select **Document name prefix**, and then **Equal**.
3. Type AWS-Run into the text field and press Enter on your keyboard to start the search.
4. Select AWS-RunPatchBaseline and choose **View details**.
5. Review the content of each tab in the details page of the document.

## **AWS Systems Manager: Run Command**

[AWS Systems Manager Run Command](https://docs.aws.amazon.com/systems-manager/latest/userguide/execute-remote-commands.html) lets you remotely and securely manage the configuration of your managed instances. Run Command enables you to automate common administrative tasks and perform ad hoc configuration changes at scale. You can use Run Command from the AWS Management Console, the AWS Command Line Interface, AWS Tools for Windows PowerShell, or the AWS SDKs.

### **3.4 Scan Your Instances with AWS-RunPatchBaseline via Run Command**

1. Under **Instances and Nodes** in the AWS Systems Manager navigation bar, choose **Run Command**. In the Run Command dashboard, you will see previously executed commands including the execution of AWS-RefreshAssociation, which was performed when you set up inventory.
2. (Optional) choose a Command ID from the list and examine the record of the command execution.
3. Choose **Run Command** in the top right of the window.
4. In the **Run a command** window, under **Command document**:
   * Choose the search icon and select Platform types, and then choose Linux to display all the available commands that can be applied to Linux instances.
   * Choose **AWS-RunPatchBaseline** in the list.
5. In the **Command parameters** section, leave the **Operation** value as the default **Scan**.
6. In the **Targets** section:
   * Under **Specify targets by**, choose **Specifying a tag** to reveal the **Tags** sub-section.
   * Under **Enter a tag key**, enter Workload, and under **Enter a tag value**, enter Test and click **Add**.

The remaining Run Command features enable you to:

* Specify **Rate control**, limiting **Concurrency** to a specific number of targets or a calculated percentage of systems, or to specify an **Error threshold** by count or percentage of systems after which the command execution will end.
* Specify **Output options** to record the entire output to a preconfigured **S3 bucket** and optional **S3 key prefix**.

**Note** Only the last 2500 characters of a command document’s output are displayed in the console.

* Specify **SNS notifications** to a specified **SNS Topic** on all events or on a specific event type for either the entire command or on a per-instance basis. This requires Amazon SNS to be preconfigured.
* View the command as it would appear if executed within the AWS Command Line Interface.

1. Choose **Run** to execute the command and return to its details page.
2. Scroll down to **Targets and outputs** to view the status of the individual targets that were selected through your tag key and value pair. Refresh your page to update the status.
3. Choose an **Instance ID** from the targets list to view the **Output** from command execution on that instance.
4. Choose **Step 1 - Output** to view the first 2500 characters of the command output from Step 1 of the command, and choose **Step 1 - Output** again to conceal it.
5. Choose **Step 2 - Output** to view the first 2500 characters of the command output from Step 2 of the command. The execution step for **PatchWindows** was skipped as it did not apply to your Amazon Linux instance.
6. Choose **Step 1 - Output** again to conceal it.

### **3.5 Review Initial Patch Compliance**

1. Under **Instances & Nodes** in the the AWS Systems Manager navigation bar, choose **Compliance**.
2. On the **Compliance** page in the **Compliance resources summary**, you will now see that there are 4 systems that have critical severity compliance issues. In the **Resources** list, you will see the individual compliance status and details.

### **3.6 Patch Your Instances with AWS-RunPatchBaseline via Run Command**

1. Under **Instances and Nodes** in the AWS Systems Manager navigation bar, choose **Run Command**.
2. Choose **Run Command** in the top right of the window.
3. In the **Run a command** window, under **Command document**:
   1. Choose the search icon, select Platform types, and then choose Linux to display all the available commands that can be applied to Linux instances.
   2. Choose **AWS-RunPatchBaseline** in the list.
4. In the **Targets** section:
   1. Under **Specify targets by**, choose **Specifying a tag** to reveal the **Tags** sub-section.
   2. Under **Enter a tag key**, enter Workload and under **Enter a tag value** enter Test.
5. In the **Command parameters** section, change the **Operation** value to **Install**.
6. In the **Targets** section, choose **Specify a tag** using Workload and Test.

**Note** You could have choosen **Manually selecting instances** and used the check box at the top of the list to select all instances displayed, or selected them individually.

**Note** There are multiple pages of instances. If manually selecting instances, individual selections must be made on each page.

1. In the **Rate control** section:
   1. For **Concurrency**, ensure that **targets** is selected and specify the value as 1.

**Tip** Limiting concurrency will stagger the application of patches and the reboot cycle, however, to ensure that your instances are not rebooting at the same time, create separate tags to define target groups and schedule the application of patches at separate times.

* 1. For **Error threshold**, ensure that **error** is selected and specify the value as 1.

1. Choose **Run** to execute the command and to go to its details page.
2. Refresh the page to view updated status and proceed when the execution is successful.

**Warning** Remember, if any updates are installed by Patch Manager, the patched instance is rebooted.

### **3.7 Review Patch Compliance After Patching**

1. Under **Instances & Nodes** in the the AWS Systems Manager navigation bar, choose **Compliance**.
2. The **Compliance resources summary** will now show that there are 4 systems that have satisfied critical severity patch compliance.

In the optional Scheduling Automated Operations Activities section of this lab you can set up Systems Manager Maintenance Windows and schedule the automated application of patches.

### **The Impact of Operations as Code**

In a traditional environment, you would have had to set up the systems and software to perform these activities. You would require a server to execute your scripts. You would need to manage authentication credentials across all of your systems.

Operations as code reduces the resources, time, risk, and complexity of performing operations tasks and ensures consistent execution. You can take operations as code and automate operations activities by using scheduling and event triggers. Through integration at the infrastructure level you avoid “swivel chair” processes that require multiple interfaces and systems to complete a single operations activity.