Patching Controls with Systems Manager

In this activity we will explore using AWS Systems Manager to gain visibility of the environment and automate patching. This activity is intended to introduce you to another opportunity to use automation to manage what is often a problem area - patch management.

In a cloud environment there are two different approaches to patching based on whether the architecture includes instances that are immutable or non-immutable. This might seem a confusing statement, but lets break it down. First, instances are the equivalent of servers in cloud speak. An immutable instance is one that is never changed, it just gets replaced. When there is a patch for an immutable instance, the image called an Amazon Machine Image (AMI), used to create the instance is update. The instances can be then be replaced with a new instance created from the new AMI.

Immutable infrastructure has some big benefits including greater infrastructure consistency, a more predictable deployment process, and the ability to easily scale up and down to meet capacity requirements. This immutable approach cannot be applied to every instance as some legacy applications don't allow for it. Where instances cannot adopt this immutable approach they can be managed using AWS AWS Systems Manager - Patch Manager.

In this activity you will use Patch Manager to ensure instances meet patch baselines requirements.

For more information see our resources on Automated patching for non-immutable instances in the hybrid cloud using AWS Systems Manager

Learning Outcomes

Learn to apply automated controls to patching and about the different approaches for immutable and non-immutable infrastructure.

In this activity you will learn how to move to automated patching controls, and how to use automation to gather information about the environment rather than relying on sample testing.

You will get an introduction to the concept of immutable infrastructure, and an understanding of how traditional patching approaches should be replaced by maintaining an AMI, rather than maintaining many separate instances. Again the idea is not to turn you into an engineer but to introduce new ways of approaching old problems.

Pre-provisioned Environment

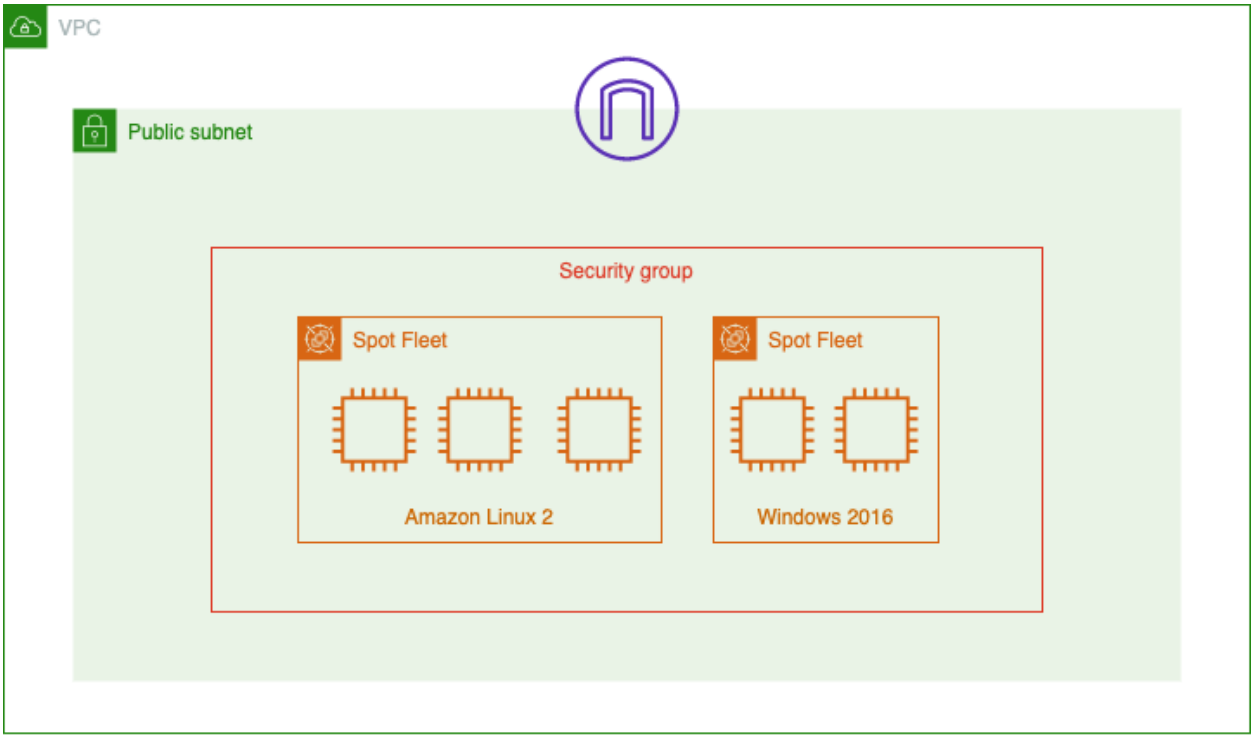
In this activity you will start with the following pre-provisioned AWS resources.

- an Amazon Virtual Private Cloud

- (VPC) with a single public subnet,
- a security group, and
- two EC2 fleets; the first with three EC2 instances running the Amazon Linux 2 operating system and the second with two instances running Windows
- **Note**: You may see additional EC2 instances deployed that will be used in future labs

An EC2 Fleet is a way of grouping and managing EC2 instances and provisioning using the lowest price combination of instances available. You can learn more about Fleets on the Introducing Amazon EC2 Fleet page.

Don't worry if not all of this is clear to you. The important thing to know is that this CloudFormation template will create five instances (or virtual servers) three running Amazon Linux 2, and two running Windows Server.



Target Audience

This activity has been developed specifically with the risk, compliance, and controls assurance community in mind, but anyone interested in understanding how to evidence AWS controls will benefit.

Prerequisites

There are no prerequisites and no assumed AWS knowledge for this activity.

Steps

- Determine the current OS versions
- Set patch baselines
- Review compliance
- Create a maintenance window
- Create Patch Manager configuration

Determine the current OS versions

First, we'll use AWS Systems Manager - **Inventory** to determine how many instances are running, and what operating system are installed.

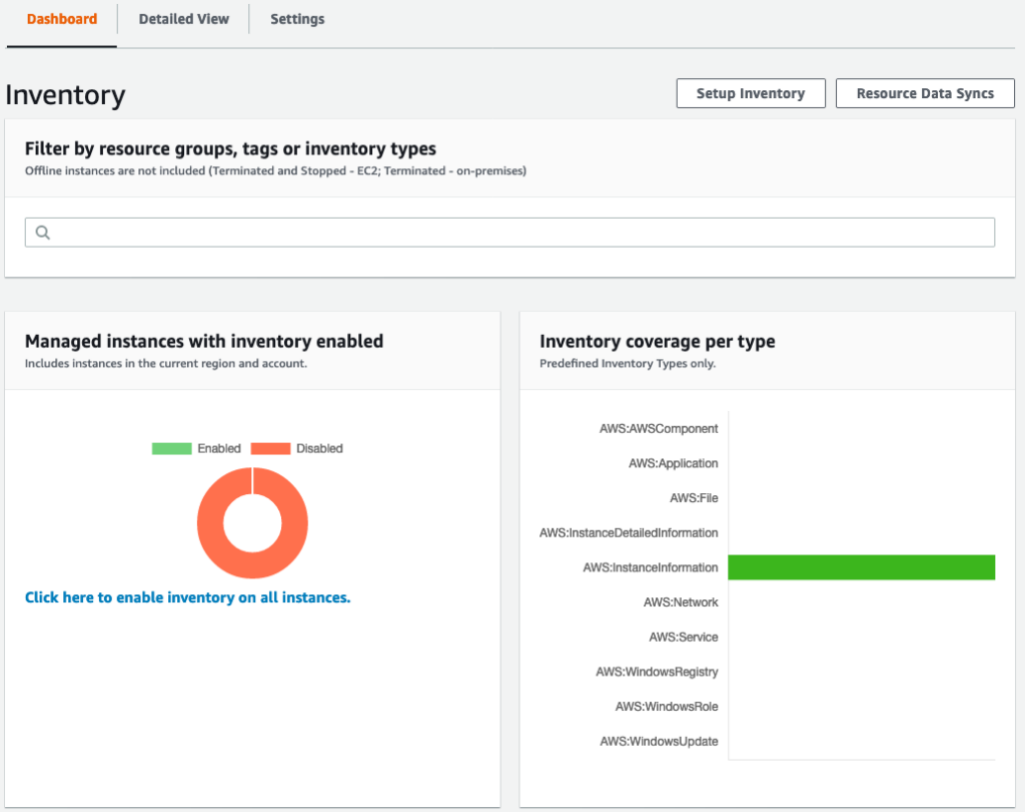
1. Go to AWS Systems Manager - Inventory

Select or Search for Systems Manager from the AWS Console.

Once in the Systems Manager page, you may need to click on the menu icon (☰) in the top left to open the navigation pane.

In the navigation pane, choose **Inventory** under **Node Management**.

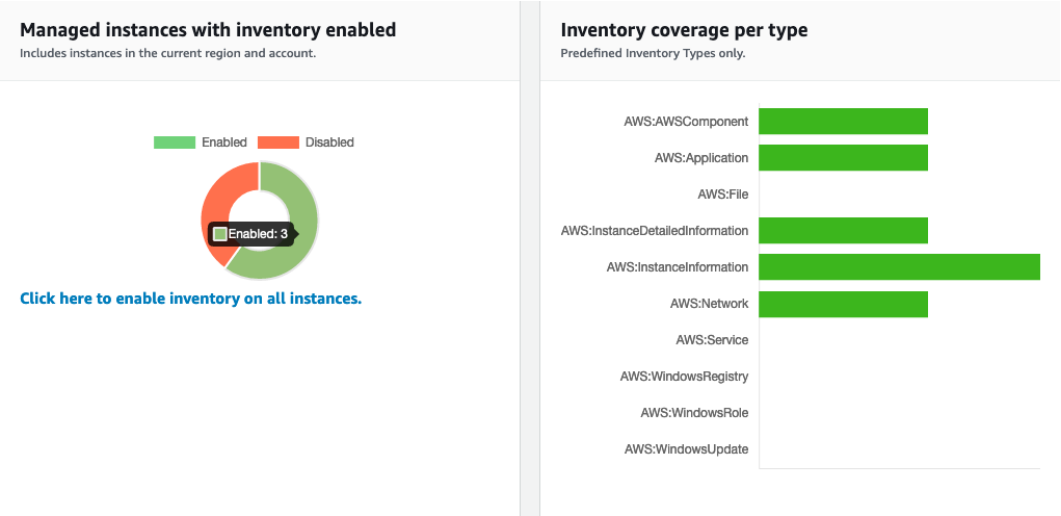
You will see something similar to the below.



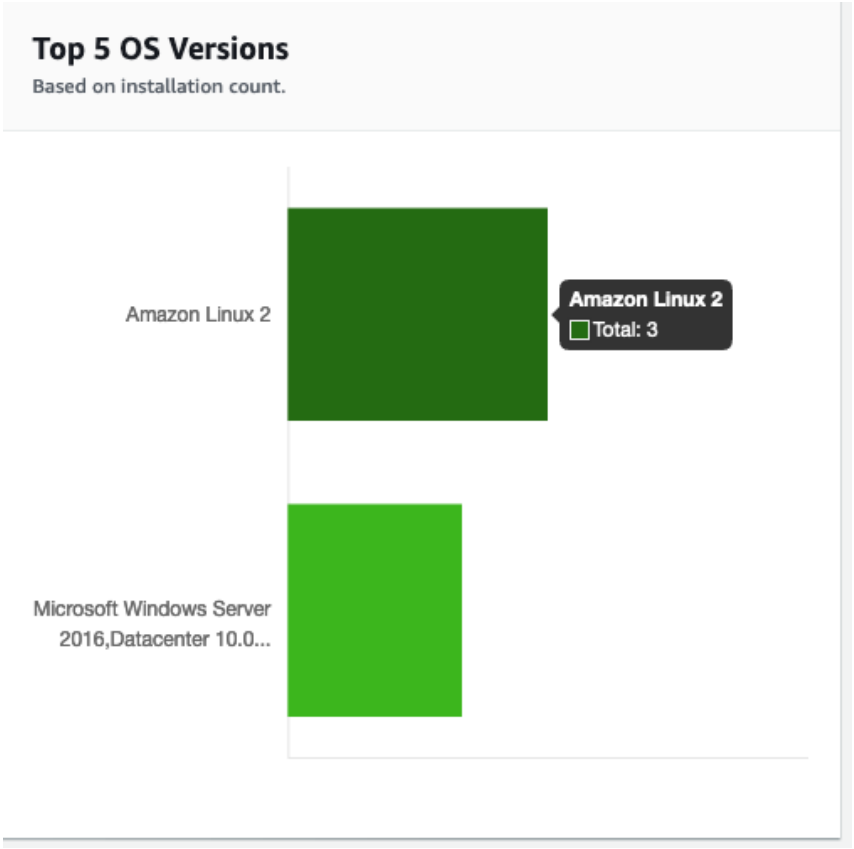
2. Ensure Inventory is enabled

Hover your mouse over the donut graph to confirm there are five instances. You may see that not all five of the instances have inventory enabled i.e. instead of a fully green donut graph, you see part or fully red donut.

If this is the case, enable the inventory and wait for this process to complete. The page will update after a few minutes to show progress, you can also manually refresh the page.



Scroll down the inventory page to see summaries of the operating system versions and other software running on the instances.



Hovering your mouse over the bars on the charts will display the total counts.

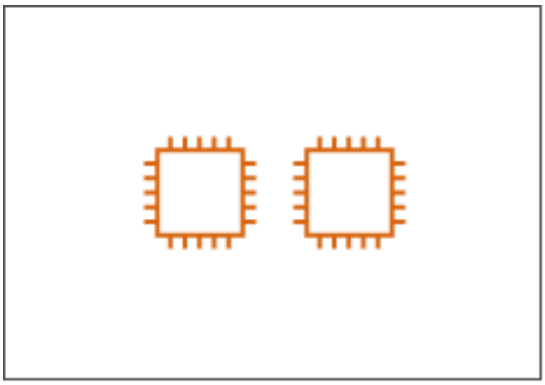
You can see it's quite easy to get a high level of visibility into your environment. The inventory lists the operating systems, services and applications running on your instances.

Set patch baselines

Now that we have enabled inventory and gained insight into the environment, let's look at Patch Manager.

To understand how Patch Manager works it important to understand two key terms; patch groups and patch baselines.

- **Patch baseline** - a set of patches and software versions which together represent the patch standard or baseline. You need to update this baseline as new patches are released.
- **Patch group** - a set of instances (virtual servers) that are to be patched to the same standard. These only changes when new instances are introduced or removed from the group.



Patch group - the group of instances that should all be patched to the same standard.

Patch baseline - the group of updates and software versions that together form a patching level or standard.

Patch Baseline

Patch Manager has predefined patch baselines for each operating system it supports, or if you have specific requirements, you can create your own. In this activity we will use the predefined patch baselines.

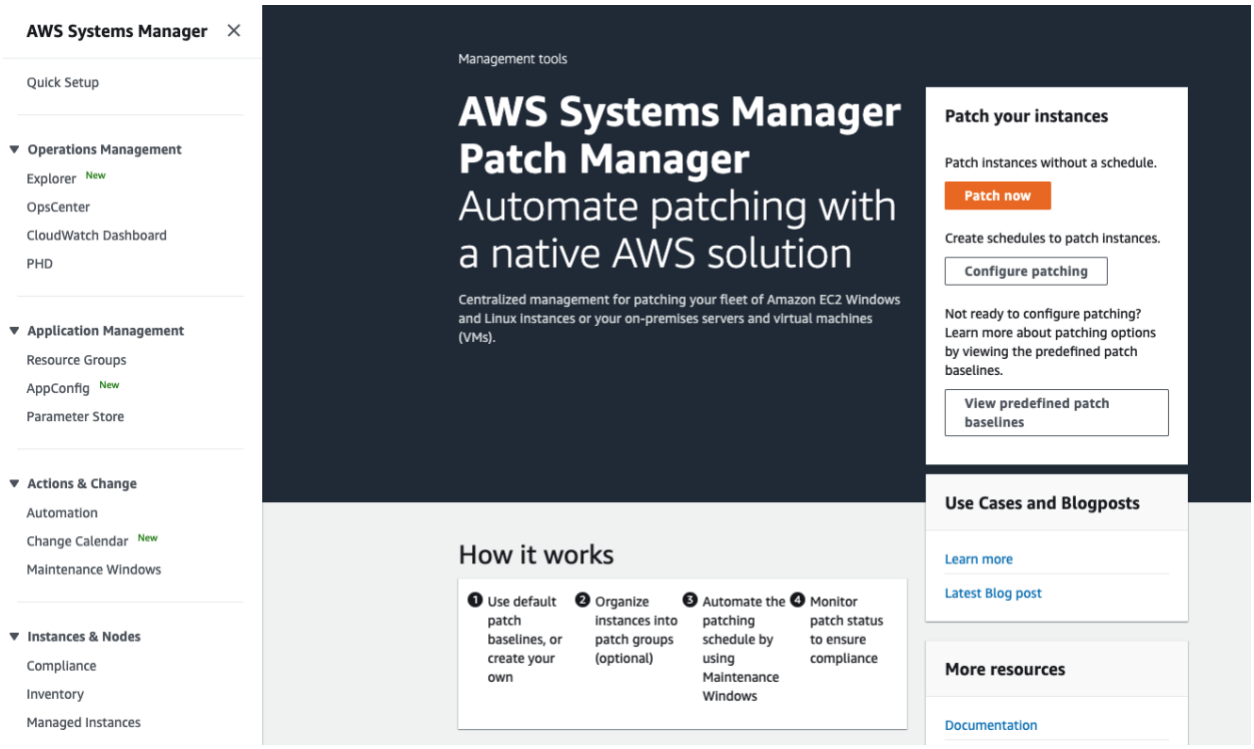
Patch Group

Patch groups are defined by a specific Tag associated with each instance in the Patch Group. Tags are just labels and values associated with AWS resources. to learn more about tags see Tagging AWS Resources . The EC2 instances provided for this lab have already been grouped into two Patch Groups: **linux2-app-patch-group** and **windows-app-patch-group**.

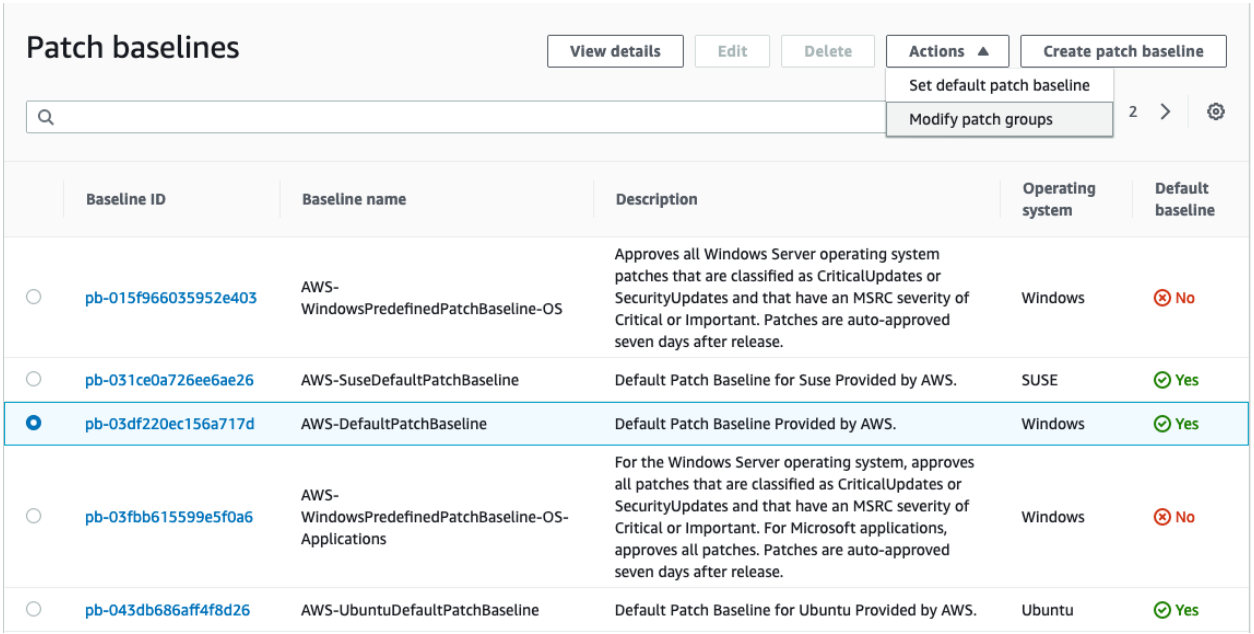
1. Open Patch Manager - Patch Baselines

Your first task is to tell Patch Manager which Patch Baselines to use for each of the two Patch Groups. To set a patch baseline open the AWS Systems Manager console and in the navigation pane, choose **Patch Manager** under **Node Management**

You should be taken directly to this **Patch baselines** screen, however if you see the AWS Patch Manager landing page, click **View predefined patch baselines** on top right.

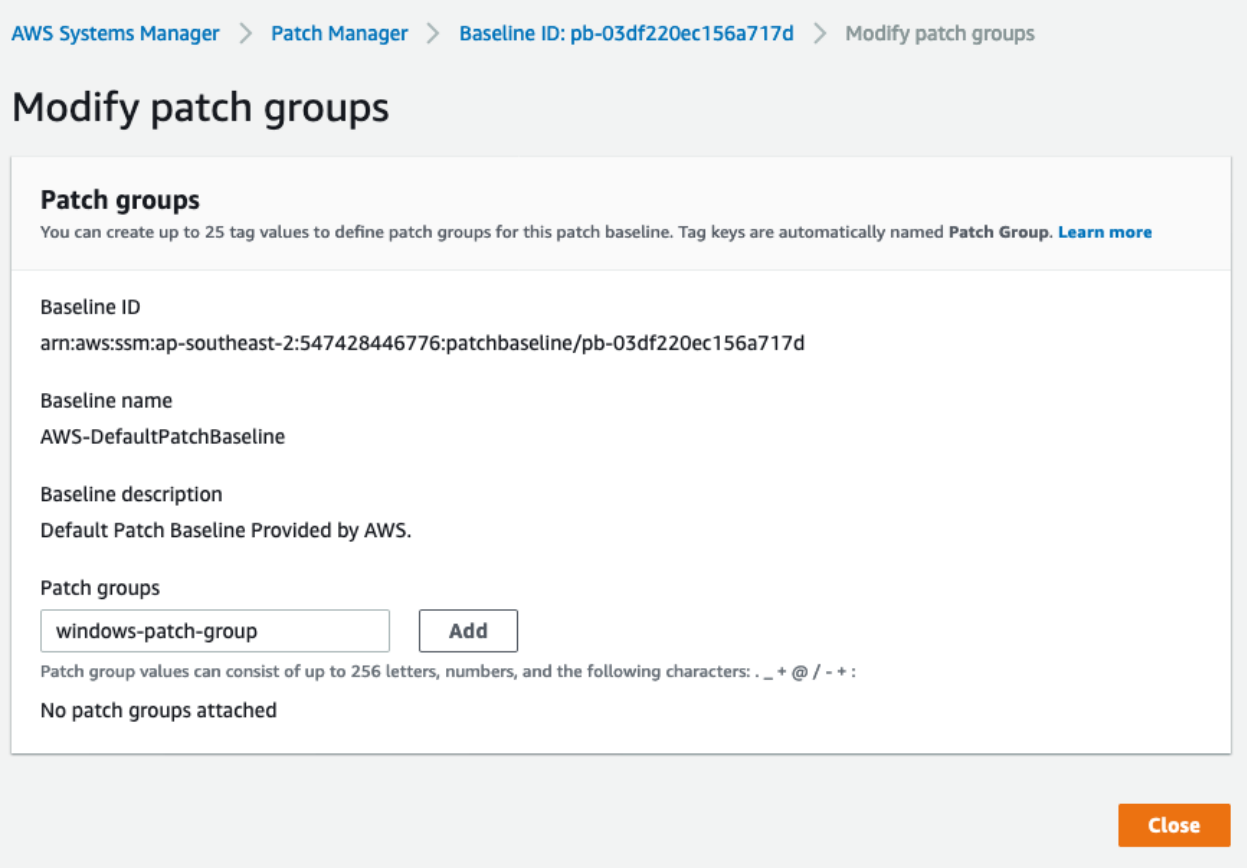


On the patch baselines page, you will see the standard patch baseline for each of the supported operating systems. For some operating systems you may have more than one entry - but only one marked "Yes" in the Default baseline column.



Select the default patch baseline for Windows (AWS-DefaultPatchBaseline) - make sure it is the Default, and then under **Actions** select **Modify patch groups**

From here add the Patch group by typing **windows-patch-group** as shown below. Make sure you click **Add** and then **Close**.



Repeat this process for Amazon Linux 2 instances (AWS-AmazonLinux2DefaultPatchBaseline)

Select the Amazon Linux 2 patch baseline and set the patch group to linux2-patch-group.

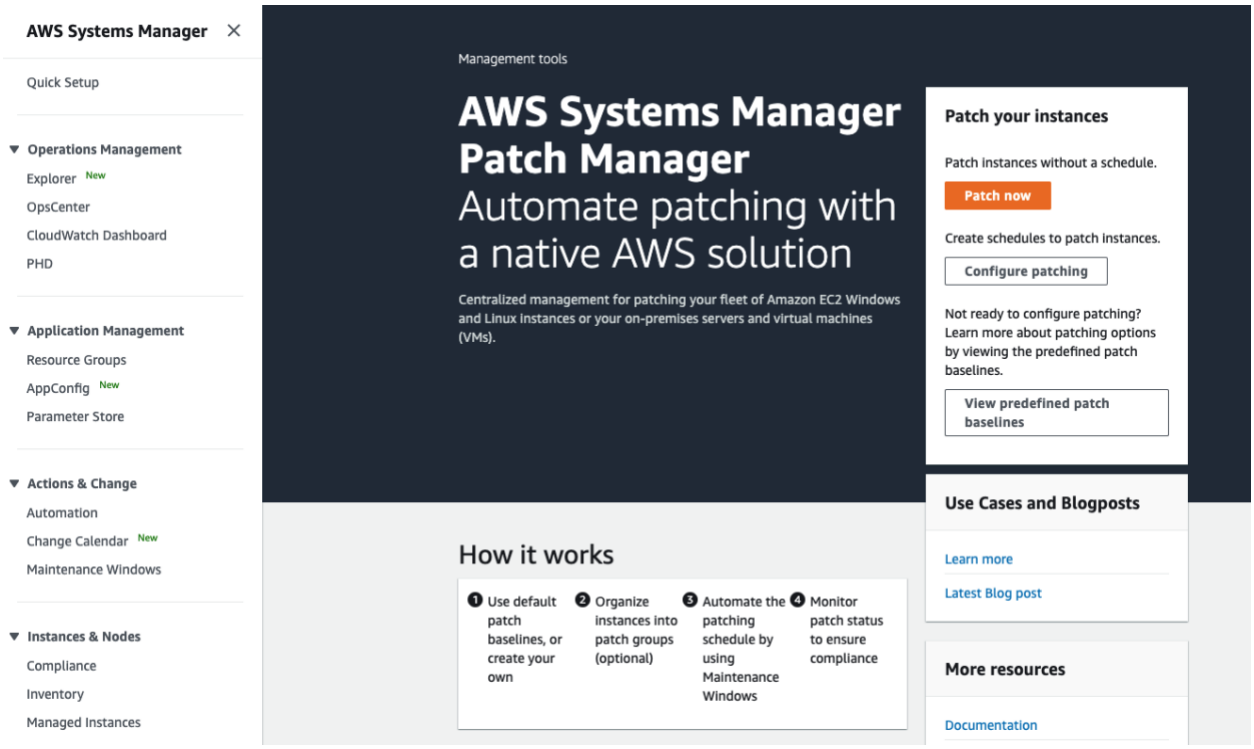
'Amazon Linux 2' not 'Amazon Linux'. Be aware we are working with "Amazon Linux 2", there is also "Amazon Linux".

Review compliance

Now that you have set the Patch Baseline for your two Patch Groups (windows-patch-group and linux2-patch-group) you can use Patch Manager to compare the instances in the Patch Groups against the standards defined. Where the patch levels on the instances in the Patch Groups do not meet the Patch Baseline they will be marked as not compliant.

1. Run Patch Manager Scan

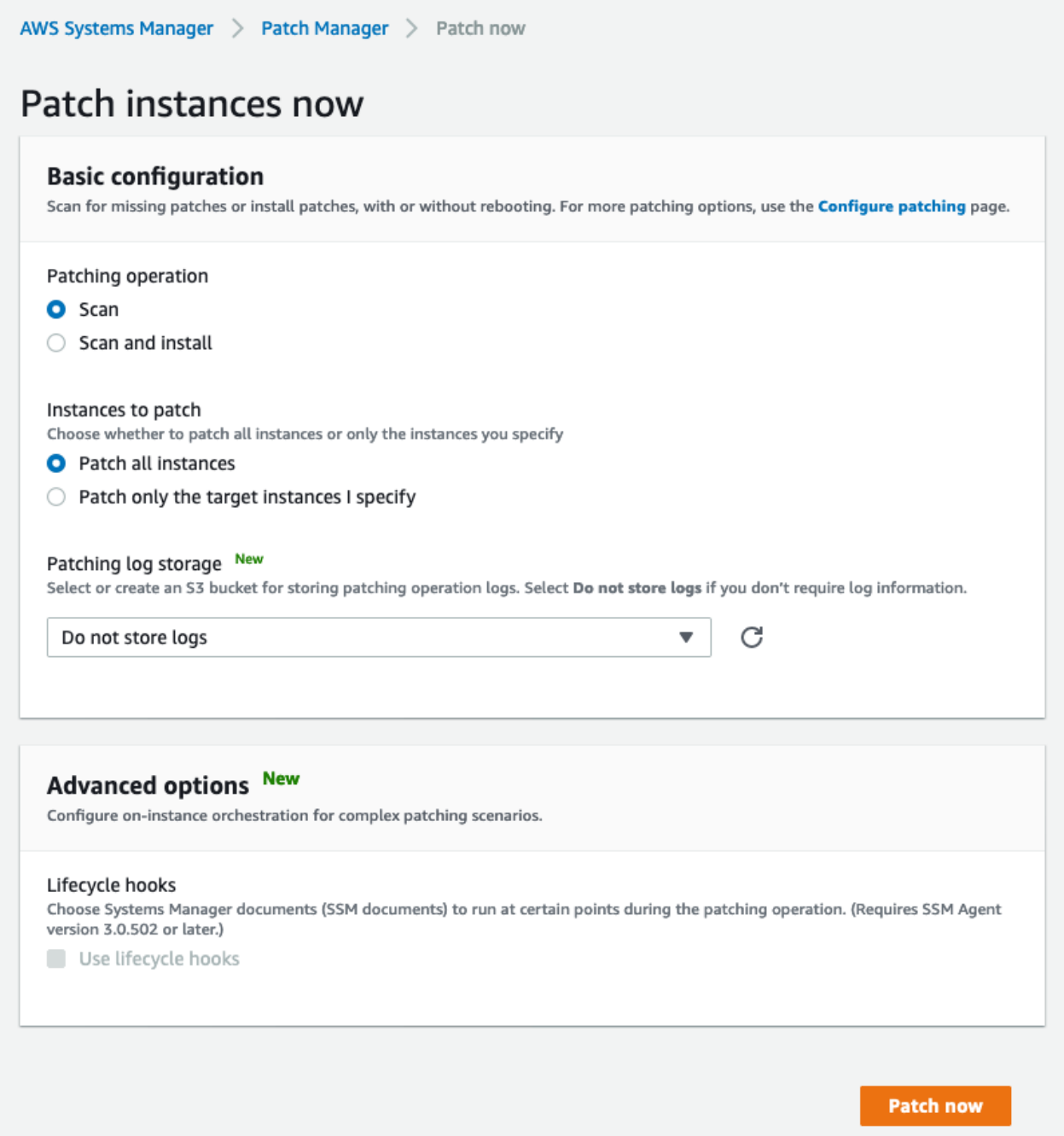
From Patch Manager select **Patch now**



2. Scan Instances

At this stage we just want to check which instances require a patch so run Patch Manager to **Scan** only.

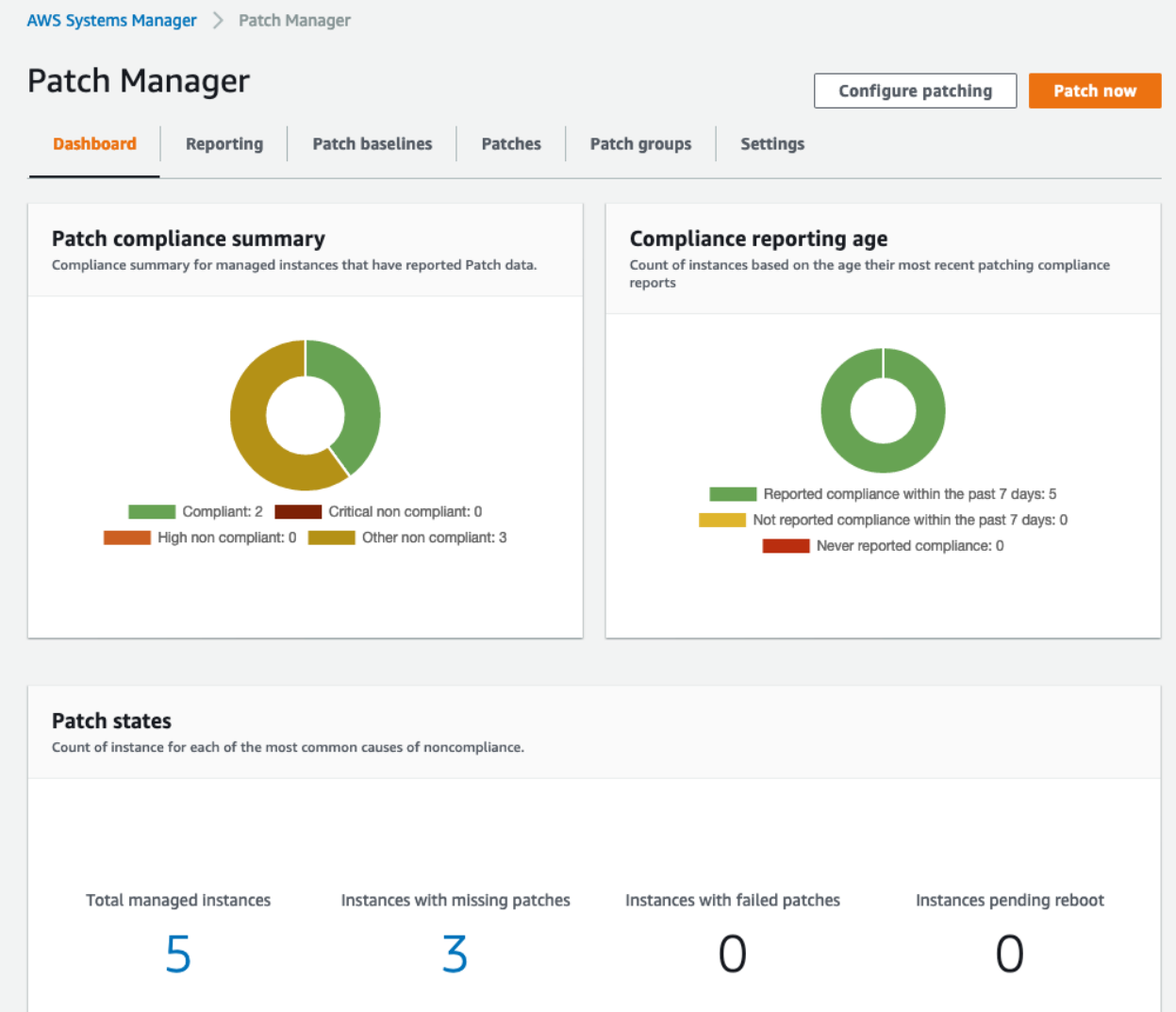
Patch Manager can also write detailed logs of the patching of each instance, but for this activity set **Patching log storage** to **Do not store logs**.



Click **Patch now**
You will see the progress tracked on screen, wait for a couple of minutes for the scan to complete.

3. Review Compliance

Head back to the Patch Manager home page by selecting **Patch Manager** under **Node Management** in the navigation pane.



You will now see a summary of the compliance status of the instances in the account and a history of the patching activities.

Patching compliance
As the instances for this workshop were created using the latest AMI you may see that they are fully up to date and do not require any patches. AWS maintains these default AMIs and continuously updates them to the latest security and operating system patches. You can also create your own patch baselines which will be up to you to maintain.

Create a maintenance window

AWS Systems Manager includes sophisticated tools which can be used together to build change management workflows which will check for conflicts and enforce approval gates. You can also build change calendars with restrict changes to specific periods, or change freeze periods.

For this activity we'll keep it simple and use Maintenance Windows to define the schedule to perform potentially disruptive such as patching, updating drivers, or installing software which may require Systems Manager to perform a restart.

Create a single maintenance window which you will use to patch both the Windows and Amazon Linux 2 instances. The details of the maintenance window such as time, duration, frequency is all up to you.

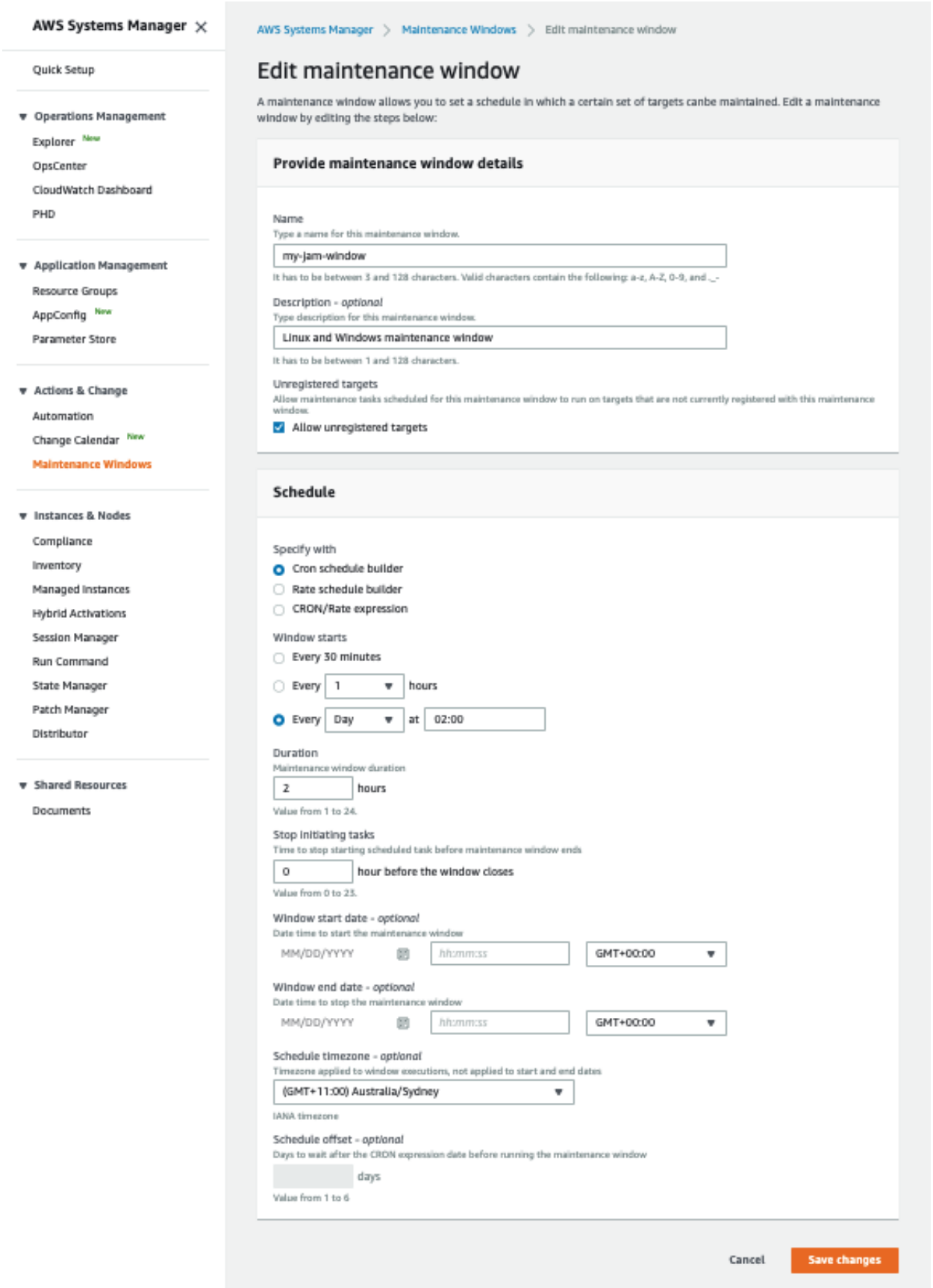
1. Create a Maintenance Window

To create a maintenance window go to **Maintenance Windows** under **Change Management** on the Systems Manager navigation pane.

Select **Create Maintenance Window**

You can then set the **Name** and **Description** of your maintenance window to whatever you like, noting the valid characters.

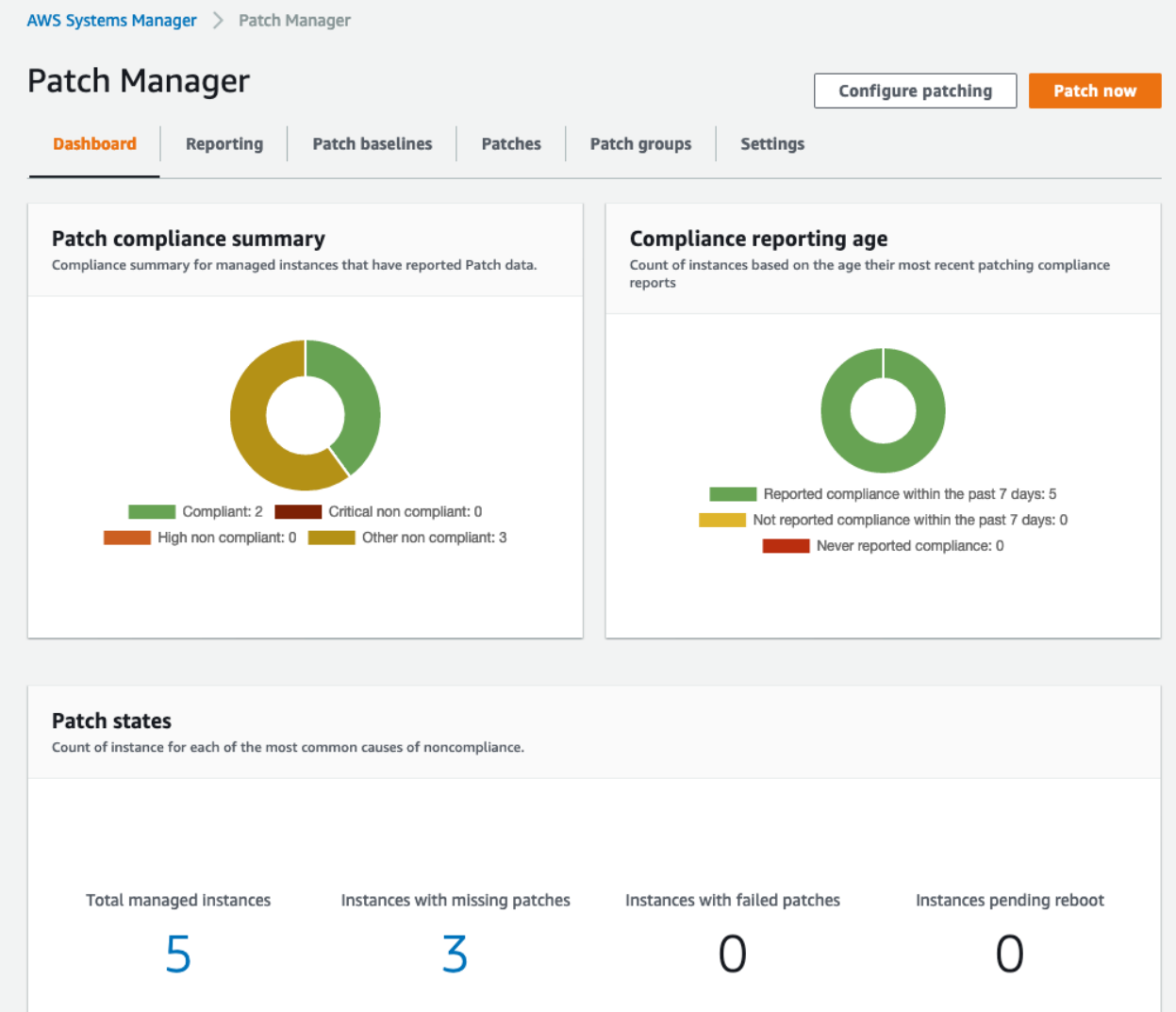
To set the Schedule use the **Cron schedule builder**, but you can set schedule details as you wish.


AWS Systems Manager
Quick Setup
Operations Management
Explorer New
OpsCenter
CloudWatch Dashboard
PHD
Application Management
Resource Groups
AppConfig New
Parameter Store
Actions & Change
Automation
Change Calendar New
Maintenance Windows
Instances & Nodes
Compliance
Inventory
Managed Instances
Hybrid Activations
Session Manager
Run Command
State Manager
Patch Manager
Distributor
Shared Resources
Documents
AWS Systems Manager > Maintenance Windows > Edit maintenance window
Edit maintenance window
A maintenance window allows you to set a schedule in which a certain set of targets canbe maintained. Edit a maintenance window by editing the steps below:
Provide maintenance window details
Name
Type a name for this maintenance window.
my-jam-window
It has to be between 3 and 128 characters. Valid characters contain the following: a-z, A-Z, 0-9, and ._-
Description - optional
Type description for this maintenance window.
Linux and Windows maintenance window
It has to be between 1 and 128 characters.
Unregistered targets
Allow maintenance tasks scheduled for this maintenance window to run on targets that are not currently registered with this maintenance window.
Allow unregistered targets
Schedule
Specify with
Cron schedule builder
Rate schedule builder
CRON/Rate expression
Window starts
Every 30 minutes
Every 1 hours
Every Day at 02:00
Duration
Maintenance window duration
2 hours
Value from 1 to 24.
Stop initiating tasks
Time to stop starting scheduled task before maintenance window ends
0 hour before the window closes
Value from 0 to 23.
Window start date - optional
Date time to start the maintenance window
MM/DD/YYYY hh:mm:ss GMT+00:00
Window end date - optional
Date time to stop the maintenance window
MM/DD/YYYY hh:mm:ss GMT+00:00
Schedule timezone - optional
Timezone applied to window executions, not applied to start and end dates
(GMT+11:00) Australia/Sydney
IANA timezone
Schedule offset - optional
Days to wait after the CRON expression date before running the maintenance window
days
Value from 1 to 6
Cancel
Save changes

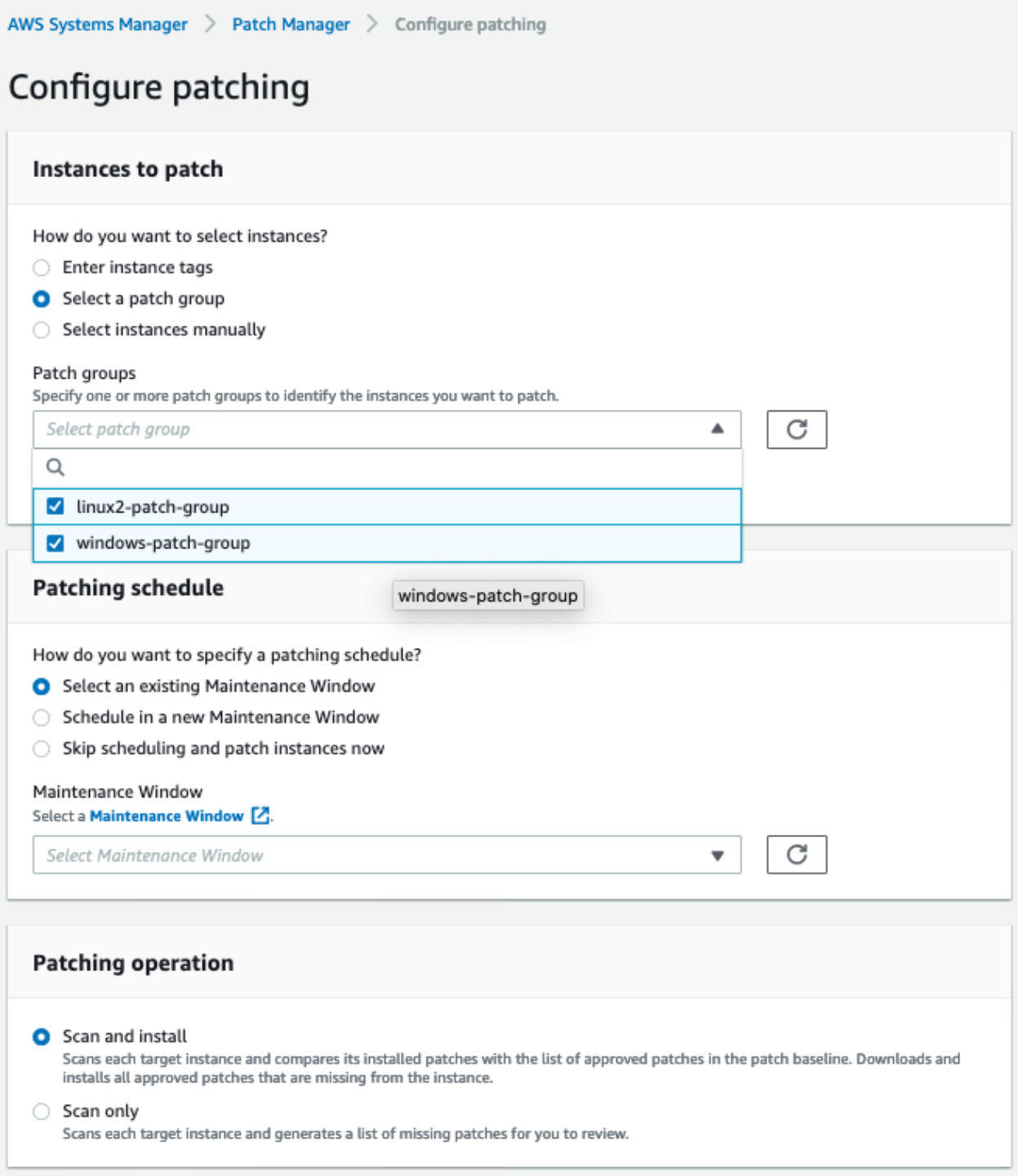
Once you satisfied with your maintenance window click **Save changes**

Create Patch Manager configuration

Now that you have setup the patch baselines and the maintenance window there is one final task. We need to configure Patch Manager to automatically apply patches as per your schedule. From the Patch Manager console select **Configure patching**



Complete the Patch manager configuration by selecting the two patch groups, the maintenance schedule you created earlier, and selecting **Scan and install**, then click **Configure patching**.



Select **Configure patching**

That's it, you've now set up automated patching! Patches will now be applied to the instances in the Linux 2 and Windows patch groups during the maintenance window.

Well done! You've completed this activity! The objective has been to give you the chance to further explore automated controls assurance and the continuous assurance that AWS tools

can provide. Keep in mind the patching approach in this activity should only be used for long running instances. A more cloud-oriented approach is to use short lived immutable instances that are regularly replaced. These short-lived instances are built from a fully patched image you can maintain, called and Amazon Machine Image (AMI).