**SSM Manager – Lab Instructions**

1. **Setup** Goal: set up your lab environment

Find the sample code repo in Github ([here](https://github.com/rjgleave/aws-ssm-immersion-day)) which contains the code needed for this lab. It may be convenient to download the repo to your local machine. It has a folder called *setup-lab* with all the Cloudformation templates needed to build your environment.

* 1. The table below shows the IAM resources you will need for this lab. These roles and profiles will be used by all lab participants. Use the IAM console to verify that they exist.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Role or Profile** | **ARN** | **Use this for…** |
| 1 | ManagedInstanceProfile | arn:aws:iam::<your-account>:instance-profile/ManagedInstanceProfile | Launching instances. Attach to each instance. |
| 2 | AutomationServiceRole | arn:aws:iam:: <your-account>::role/AutomationServiceRole | Running automations |

If they do not yet exist, follow the process below to create them.

* + 1. Find the ***create-ssm-roles.yaml*** template in the repo ([here](https://github.com/rjgleave/aws-ssm-immersion-day/blob/master/setup-lab/create-server-fleet.yaml)) and build the Cloudformation stack.
    2. When the process is complete, you will find the role names under the *Physical ID* column of the *Resources* drop-down section of the Cloudformation page.
    3. Take a few minutes to review the policies attached to the roles in the IAM console and read what they are used for below.
    4. Copy the ARNs of the resources above for the exercises below.
  1. Launch your sample fleet. It will create a number of foundational components, including VPC, subnet, NAT gateway, security group, etc. It will also launch a small fleet of 4 instances (Amazon Linux, Ubuntu, RHEL, Windows).
     1. Find the ***create-server-fleet.yaml*** cloudformation template in the immersion day repo ([here](https://github.com/rjgleave/aws-ssm-immersion-day/blob/master/setup-lab/create-server-fleet.yaml)) and launch it to create the stack.

|  |
| --- |
| Launch your sample fleet |
| 1. Open the AWS Cloudformation console ([here](https://github.com/rjgleave/aws-ssm-immersion-day/blob/master/setup-lab/create-server-fleet.yaml)). 2. Press the *Create Stack* button 3. Select *upload template to Amazon S3 button* and choose the *create-server-fleet.yaml* template you downloaded in step 1.3.1 above. 4. Press the *Next* button. 5. Enter a name for your stack. It needs to be unique from other lab participants, so one approach is to append your initials like the example below.      1. Enter a name for your fleet, such as: *prod*, *dev*, *webservers*, etc. Don’t just accept the default name (*webservers*) because it will create an S3 bucket using this name. It needs to be unique, to help differentiate your resources from the other participants (for example, use your initials again). 2. At the *Create Sample Fleet*? prompt, answer *yes* (see above) 3. Press the *Next* button. 4. On the next two screens, you can leave the default values in place and complete the process to launch your stack. It should take about 5 minutes to build your environment. 5. Once the stack is built, validate that you have 4 running EC2 instances, as well as an S3 bucket named *<your fleet>-ssm-lab*. Notice that each of your instances has a *Fleet* tag with the value you entered. |

* 1. Set up your S3 bucket to hold inventory data and logs
     1. Go to the S3 console and find your bucket. It should be named: *<your fleet name>-ssm-lab*.
     2. Locate and review the bucket policy document, which was created by Cloudformation. You will find it under the *Permissions* tab. This policy is critical for allowing SSM inventory to sync to S3.

1. **SSM Inventory** Goal: set up your inventory system
   1. Select *Systems Manager* from the list of *Services* on the console.
   2. Select ***Inventory*** from the left navigation bar.
   3. Roll down to the bottom of the inventory page. You should see a list showing your 4 instances. Select one of your instances, then click the *Inventory* tab, and notice that there is no inventory displayed for that instance yet.
   4. Follow the steps below to create a global association. This will capture inventory data from all instances.

|  |
| --- |
| Set up SSM Inventory |
| To configure inventory collection   1. In the left navigation bar, choose *Inventory*. 2. Choose *Setup Inventory*. 3. Enter a unique name for your inventory, such as: *Inventory-Association-<your fleet name>* 4. In the Targets section, identify the instances where you want to collect inventory. For this exercise we will choose the tag option.    * Click on the *Specifying a tag* option. This option lets you specify a single tag to identify instances in your account from which you want to collect inventory. If you use a tag, any instances created in the future with that same tag will be added automatically to your inventory.    * Go to the EC2 Instances tab and look at the tags on one of your running instances. You should see a tag *Key* called *Fleet*, with the name of your fleet as the *Value*.    * Enter those values in the *Targets* section. 5. In the Schedule section, choose how often you want the system to collect inventory metadata from your instances. 6. In the Parameters section, select the buttons to enable or disable different types of inventory collection. 7. In the *Advanced* section:    * Click the check box for *Sync inventory execution logs to an S3 bucket*    * Enter the name of your S3 inventory bucket from 1.4.1 above.    * In the S3 bucket prefix – *optional* enter the name of the S3 *inventory-logs* folder which you defined in step 1.4.2.1 above. 8. Choose *Setup Inventory*. Systems Manager creates a State Manager association and immediately runs Inventory on the instances. It should run quickly. You will see a green ‘Success’ message at the top of the panel when it is done. You can click on the *View Details* button to see information about it. 9. In the navigation pane, choose *State Manager*.    * Verify that a new association was created using the *AWS-GatherSoftwareInventory* document.    * Verify that the Status field shows Success. 10. If you want to view inventory data for a specific instance:     * Select *Managed Instances* in the navigation pane.     * Choose an instance, and then choose *View details*.     * On the instance details page, choose *Inventory*. Use the Inventory type lists to practice filtering your inventory list. |

* 1. Now archive your inventory history, by following the process below. It will create a resource sync to an S3 folder.

|  |
| --- |
| Set up Inventory Resource Sync |
| 1. In the left navigation bar, choose *Inventory*. 2. Press the *Resource Data Syncs* button (top right of page) 3. Press the *Create resource data* sync button. 4. Enter a name for your sync, such as: *<your fleet name> - resource-sync* 5. Enter the name of the S3 bucket created for your fleet in step 1. 6. Select the region where your bucket is located. 7. Refresh the screen. Once the sync is complete, check the *inventory-data* folder in S3. You should see inventory information in your S3 bucket. |

1. **SSM Automation** Goal: use Automation to create an AMI, patch it, and launch instances from it
   1. Select *Automation* from the left navigation bar.
   2. Launch an automation to stop your Red Hat instance (below).

|  |
| --- |
| Using automation to stop an instance |
| 1. In the navigation pane, choose *Automation*. 2. Choose *Execute automation*. 3. In the Automation document list, find and select *AWS-StopEC2Instance.* 4. In the Document details section, set the Document version to *Default version at runtime*. 5. In the Execution mode section, choose *Execute the entire automation at once*. 6. Leave the Targets and Rate Control option disabled. 7. In the Input parameters section, click the button to *Show interactive instance picker*. Select your RedHat instance (if you are using a shared account, make sure it is from your fleet!) 8. Choose Execute automation. The console displays the status of the Automation execution. |

* 1. Use the ***AWS-StartEC2Instance*** automation document to restart the instance you stopped above. However, this time try to execute it one step at a time.
     1. Tip #1: you must change the dropdown from *Show Managed Instances* to *Show ALL Instances* to see the stopped instance in your list of targets.
     2. Tip #2: many automations require elevated privileges. Use the automation service role created in 1.2.4 as needed.
  2. Create an AMI from one of the instances in your fleet.
     1. Use the ***AWS-CreateImage*** automation to do this.
     2. Notice the parameters for the document. Some are required and others are not.
     3. Go to the AMI page of the EC2 console ([here](https://us-west-1.console.aws.amazon.com/ec2/v2/home?region=us-west-1#Images:sort=name)) to find your new image. Copy the image name to help you complete the steps below.
  3. Now create a process to automatically patch the AMI you created in step 3.4 above. For Linux images, follow the steps defined below (or see full instructions [here](https://docs.aws.amazon.com/systems-manager/latest/userguide/automation-consolewalk.html)). If you created a Windows AMI, follow the patching steps indicated [here](https://docs.aws.amazon.com/systems-manager/latest/userguide/automation-createdoc.html).

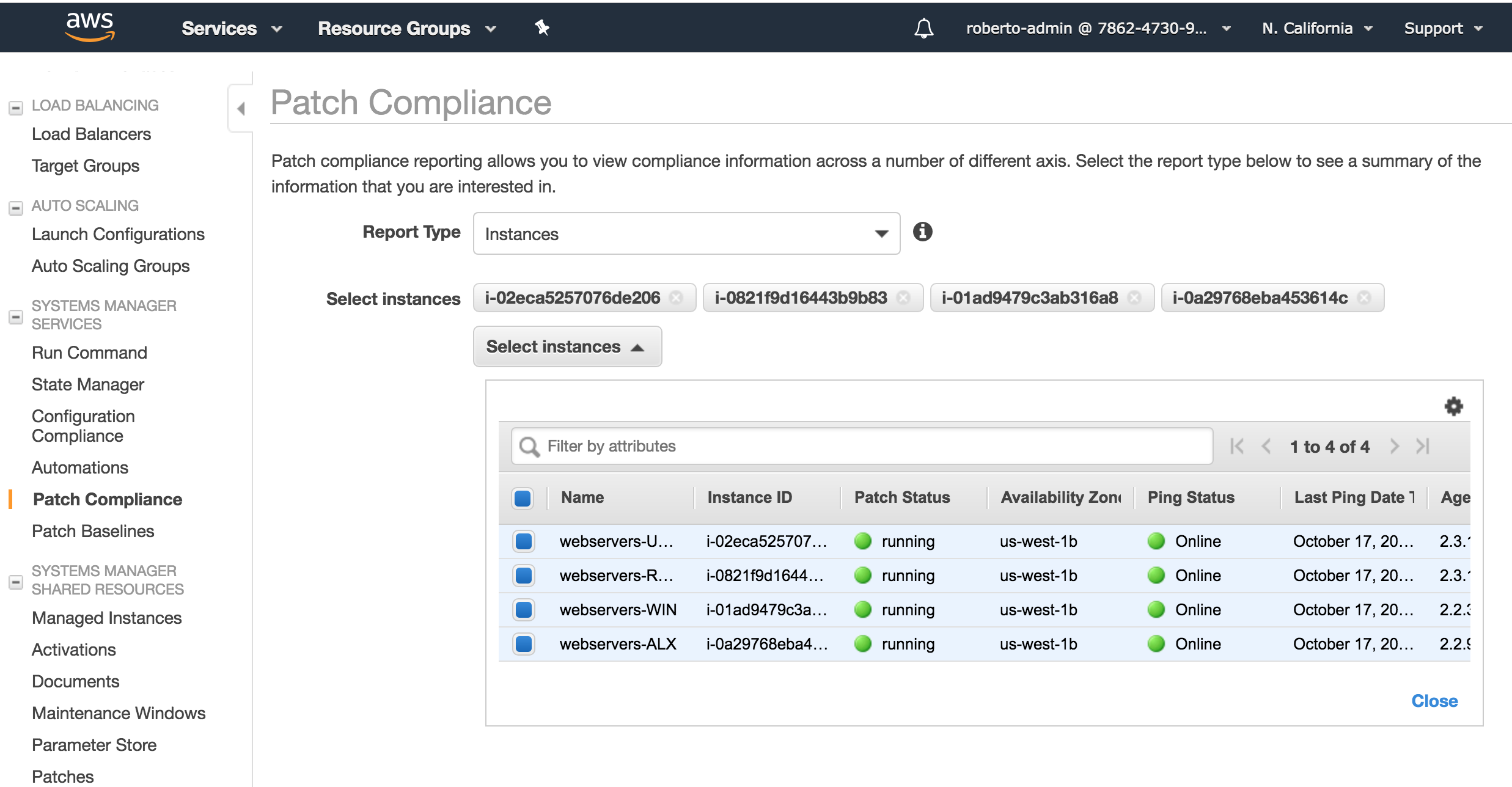
|  |
| --- |
| Patch a Linux AMI |
| This process will walk you through how to use the AWS-UpdateLinuxAmi document to automatically patch a Linux AMI with the latest versions of packages that you specify.   |  | | --- | | Note: when you run the *AWS-UpdateLinuxAmi* document, SSM Automation performs the following tasks**.**   * Launches a temporary Amazon EC2 instance from a Linux AMI. The instance is configured with a User Data script that installs SSM Agent. SSM Agent runs scripts sent remotely from Systems Manager Run Command. * Updates the Instance by performing the following actions:   1. Invokes a user-provided pre-update script on the instance (optional).   2. Updates AWS tools on the instance, if any tools are present.   3. Updates distribution packages on the instance by using the native package manager.   4. Invokes a user-provided post-update script on the instance (optional). * Stops the temporary instance. * Creates a new AMI from the stopped instance. * Terminates the instance. |   After Automation successfully completes this workflow, the new AMI is available on your AMIs page.  Steps to follow:   1. Open the AWS Systems Manager console at <https://console.aws.amazon.com/systems-manager/>. 2. In the navigation pane, choose *Automation*. 3. Choose *Execute automation*. 4. In the Automation document list, find and choose *AWS-UpdateLinuxAmi*. 5. In the Document details section, select *Default version at runtime*. 6. In the Execution mode section, choose *Execute the entire automation at once*. 7. Leave the Targets and Rate Control option disabled. 8. In the Input parameters section please note the required parameters that you MUST enter. Also, pay special attention to the roles indicated below, which must have the proper rights (info [here](https://docs.aws.amazon.com/systems-manager/latest/userguide/automation-setup-user.html))      1. Choose Execute automation. The console displays the status of the Automation execution. 2. When the automation finishes, select AMIs from the EC2 menu to see your updated AMI. |

* 1. Use the AWS-CreateManagedLinuxInstance command to create an instance from your AMI.
     1. Use SSM Automation to launch the new instance
     2. Once the instance is running, check your inventory to see that the instance has been added to it

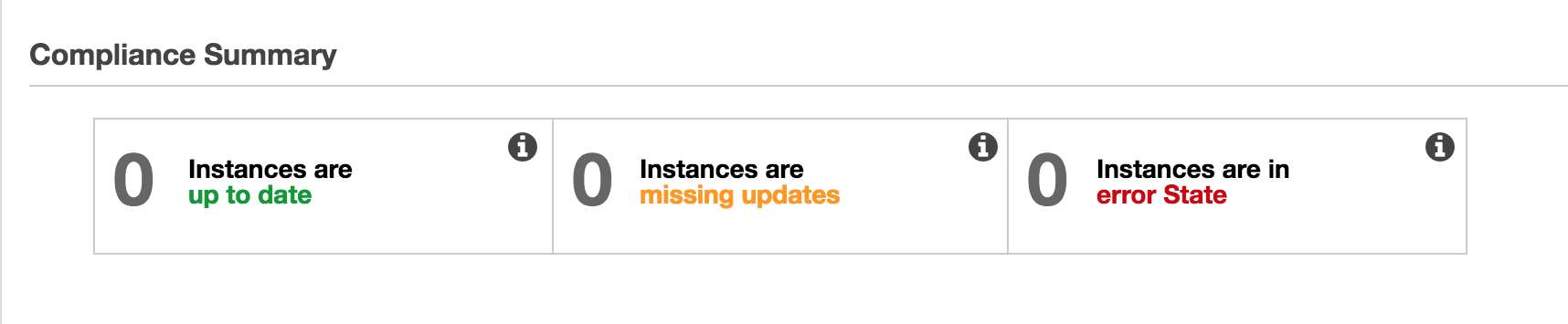
1. **SSM Run Command** Goal: practice using the *RUN* command
   1. Select ***Run Command*** from the left navigation bar.
   2. Press the *Run Command* button.
   3. Use the search functionality to find the *AWS-RunShellScript* command and select it.
   4. Select the target instances to run the command (for example, select all). (QUESTION: why do only 3 instances show up on the target list?)
   5. In the ‘command’ section, enter the linux command ‘ls’, to list files on the instance.
   6. Refresh the screen until you see ‘success’.
   7. See the results of your run command:
      1. On the *Run Command* menu, select *Command History*.
      2. Click on the Command ID to see the results of the command
      3. Click on one of the instance IDs to see results.
      4. View the Output of the command. You should see a list of files from each instance.

|  |
| --- |
| **Tip:** You can also configure run commands to log their results to an S3 bucket |

1. **SSM Session Manager** Goal: learn how to log into an instance without using a bastion host
   1. Select ***Session Manager*** from the left navigation bar.
   2. Define Preferences (click *EDIT* button to start)
      1. Configure logging of session activity to the S3 folder you created in 1.4.2.3 above.
      2. Enter a log group name to configure Cloudwatch logging
   3. Click the *Start Session* button to start a session
   4. Select one of the Linux instances from the target list.
   5. Enter some sample commands on the remote instance command line
   6. End the session. There are two ways to do this (try both):
      1. Type ‘exit’ on the command line of the instance
      2. Click the *Terminate* button)
   7. History – review your activity on the *Session History* tab.
   8. Now select the Windows instance.
   9. Try to start a session on the instance. (Note: It may give an error and say that Systems Manager is not supported by the version of the agent on that instance. If so, press the button to update the version of the agent and try again once the update completes)
2. **SSM Patch Manager** Goal: learn how to configure patching in multiple ways and view compliance.
   1. View Patch Compliance – find the Patch Compliance Dashboard.
      1. Go to the EC2 Menu and find Patch Compliance on the left navigation bar.
      2. Select *Instances* from the dropdown and check mark all the instances



* + 1. You should see a compliance dashboard that looks like this:

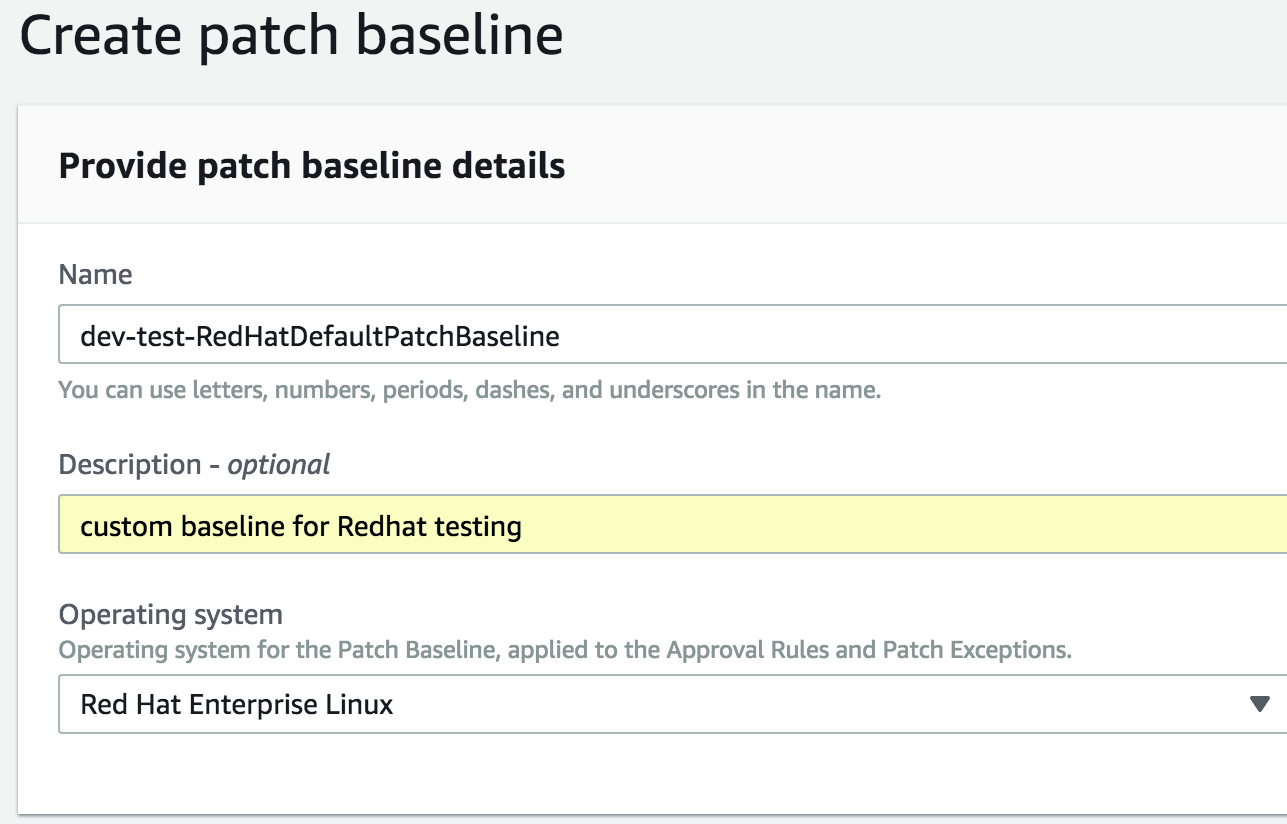


|  |
| --- |
| **Tip: Nothing registers on this dashboard until you associate instances with baselines (below)** |

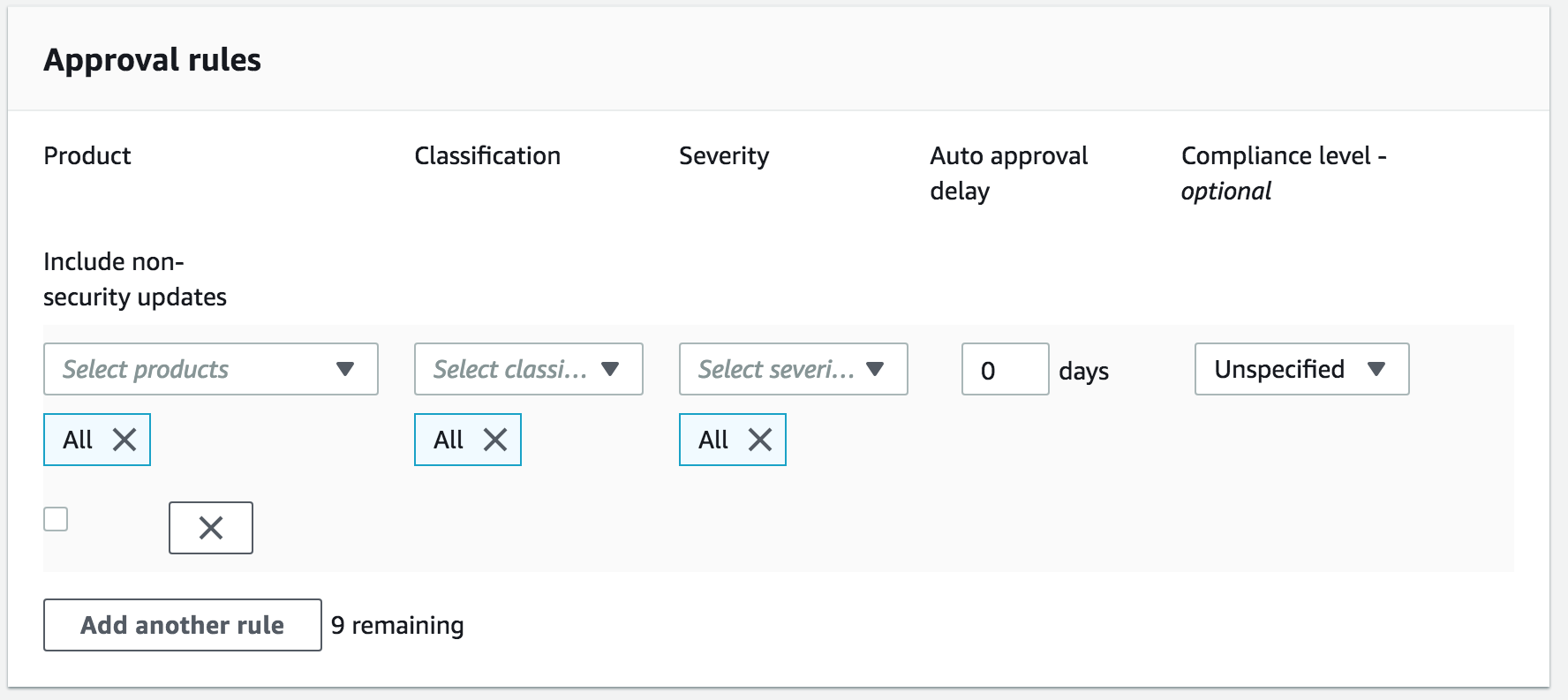
* 1. Work with your patch baselines. Select ***Patch Manager*** from the left navigation bar.
     1. Look for the *default patch baselines* link on the Patch Manager Screen (also on the main EC2 console)



* + 1. Select one of the baselines and review each of the tabs: *Description*, *Approval Rules* and *Patch Exceptions* to familiarize yourself with what baselines contain.
    2. Create a custom baseline
       1. On the *Patch Baselines* screen, press the *Create patch baseline* button
       2. Enter a name, description and choose the operating system, as shown below.



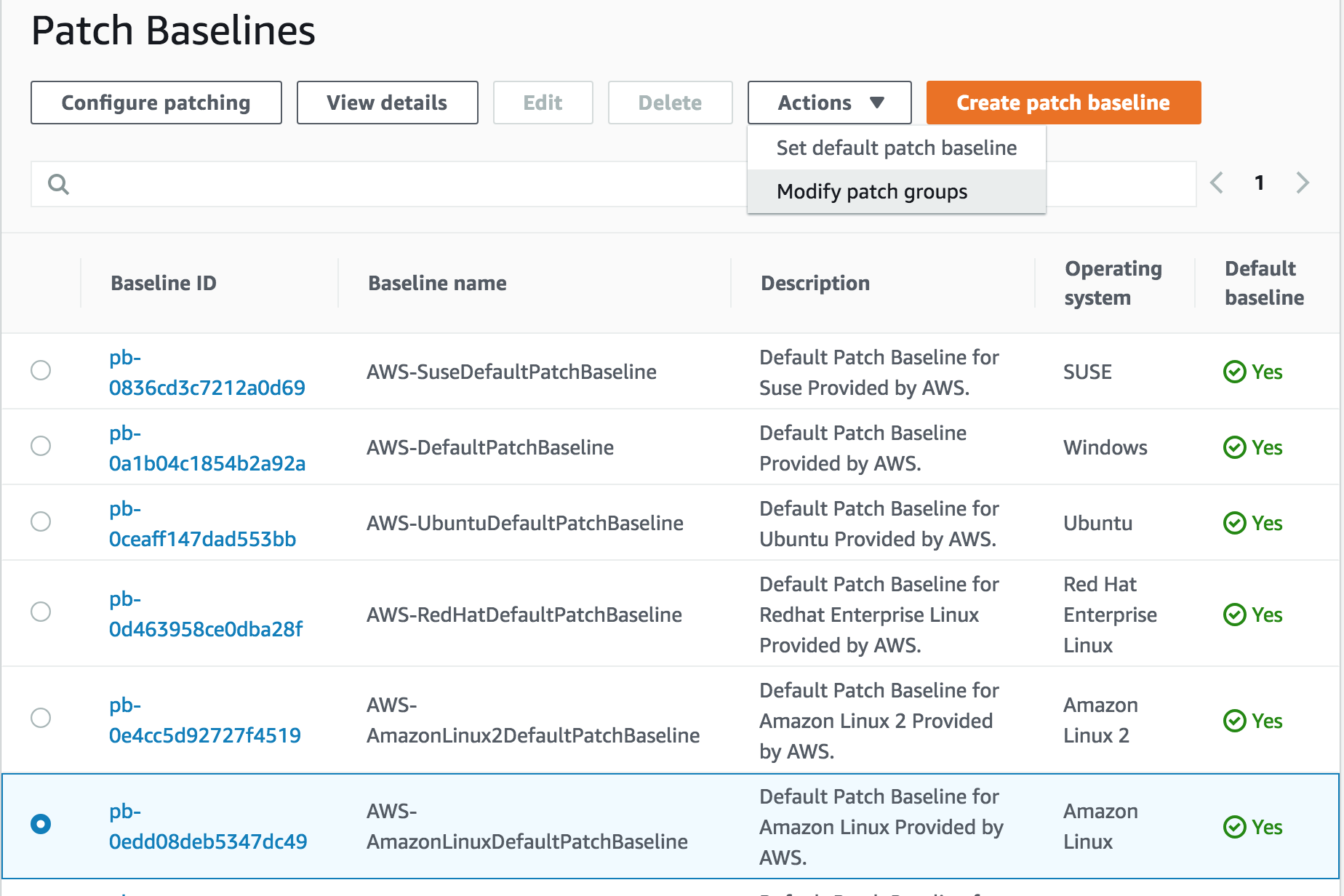
* + - 1. Enter rules for your custom baseline. Leave them all blank, with zero days delay so that all patches apply immediately to the dev and test instances.



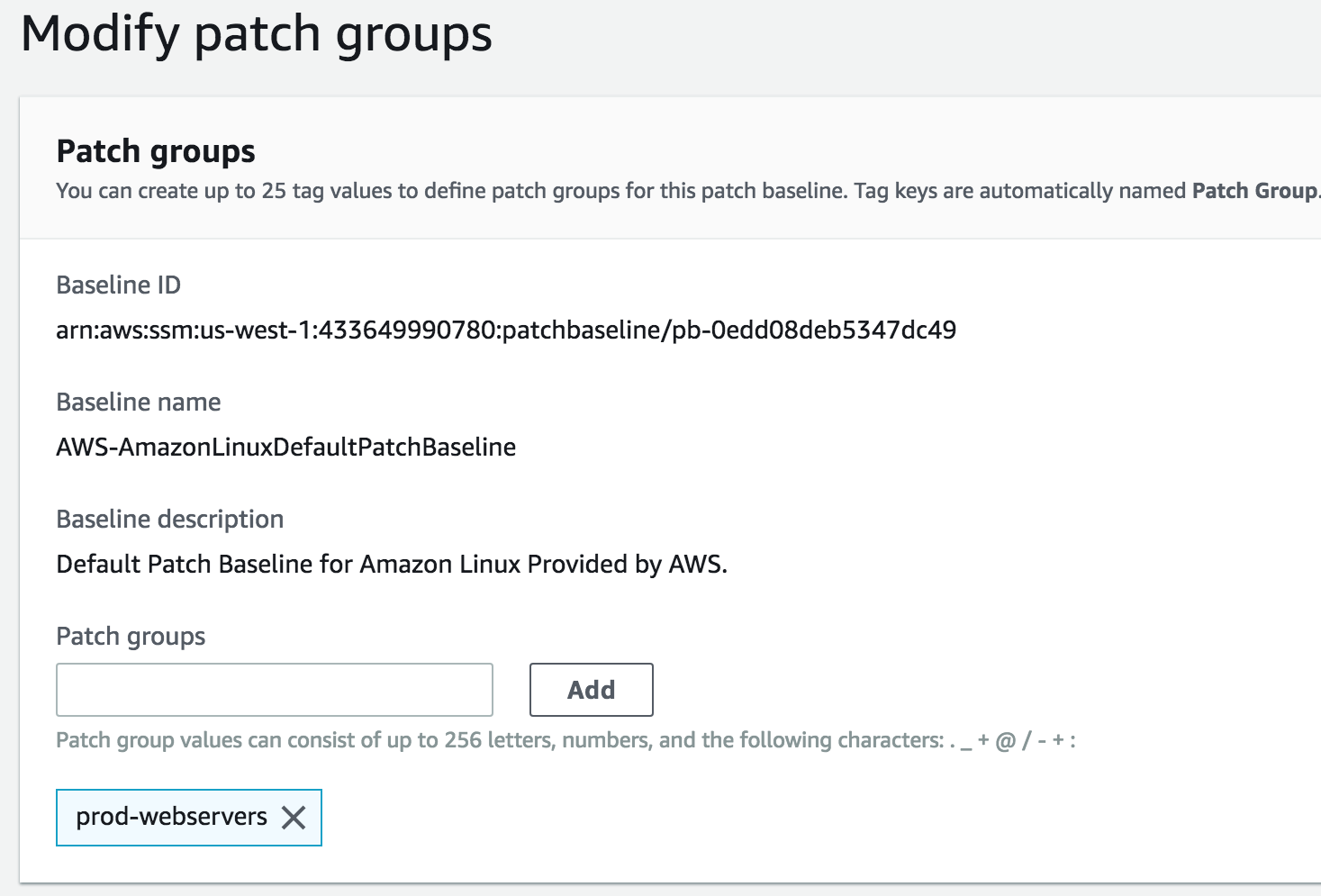
* + - 1. Press the *Create patch baseline* button
      2. You now have created a custom baseline!

|  |
| --- |
| What is a Patch Group? |
| A Patch Group is an optional means of organizing 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.  IMPORTANT: Patch Groups are very useful because they are the ONLY way to associate non-default baselines to instances.   |  | | --- | | The mechanism for creating patch groups is to tag instances with special EC2 tags. Patch groups require use of the special tag key name ***Patch Group***. You can specify any tag value you prefer, but the tag key must be **Patch Group**. Note that the key is case sensitive. |   When Patch Manager runs the AWS-RunPatchBaseline task on that instance, the service checks to see which patch baseline is registered with a given patch group. If a patch baseline is found, the system uses that baseline. If no patch baseline is registered, the system uses the default patch baseline.  For more information about patch groups, see [Organize Instances into Patch Groups](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html). |

* + 1. Define a patch group and attach it to one of your default and custom baselines
       1. On the *Patch Baselines* screen, select the Amazon Linux baseline
       2. On the *Actions* dropdown, select *Modify Patch Groups* as shown below



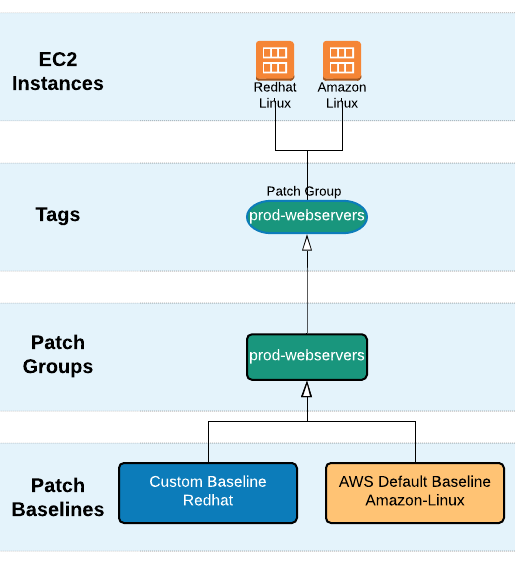
* + - 1. Enter the name ***prod-webservers*** for this patch group.
      2. Press the *Add* button to save it.



* + - 1. On the *Patch Baselines* screen, select the CUSTOM Redhat Linux baseline you created in step 6.2.3.
      2. Repeat steps 6.2.4.2 through 6.2.4.4. to add the same *prod-webservers* patch group to this baseline.
      3. You have now created a patch group and associated it with two of your baselines.
    1. Now that you have created a patch group, you will link instances to it, by adding ***Patch Group*** tags to the Amazon Linux and Redhat instances.
       1. Open the [Amazon EC2 console](https://console.aws.amazon.com/ec2/), and choose *Instances* in the navigation pane.
       2. Choose the Amazon Linux instance.
       3. From the *Actions* menu, choose *Instance Settings*, *Add/Edit Tags*.
       4. Choose *Create Tag*.
       5. In the Key field, type *Patch Group* (NOTE: this must be typed exactly as shown).
       6. In the Value field, type *prod-webservers* and press *Save*.
       7. Repeat this procedure to add the Redhat instance to this patch group.

|  |
| --- |
| NOTE: An instance can only be assigned to ONE patch group! |

Congratulations! You have completed the definition of the patch group below.

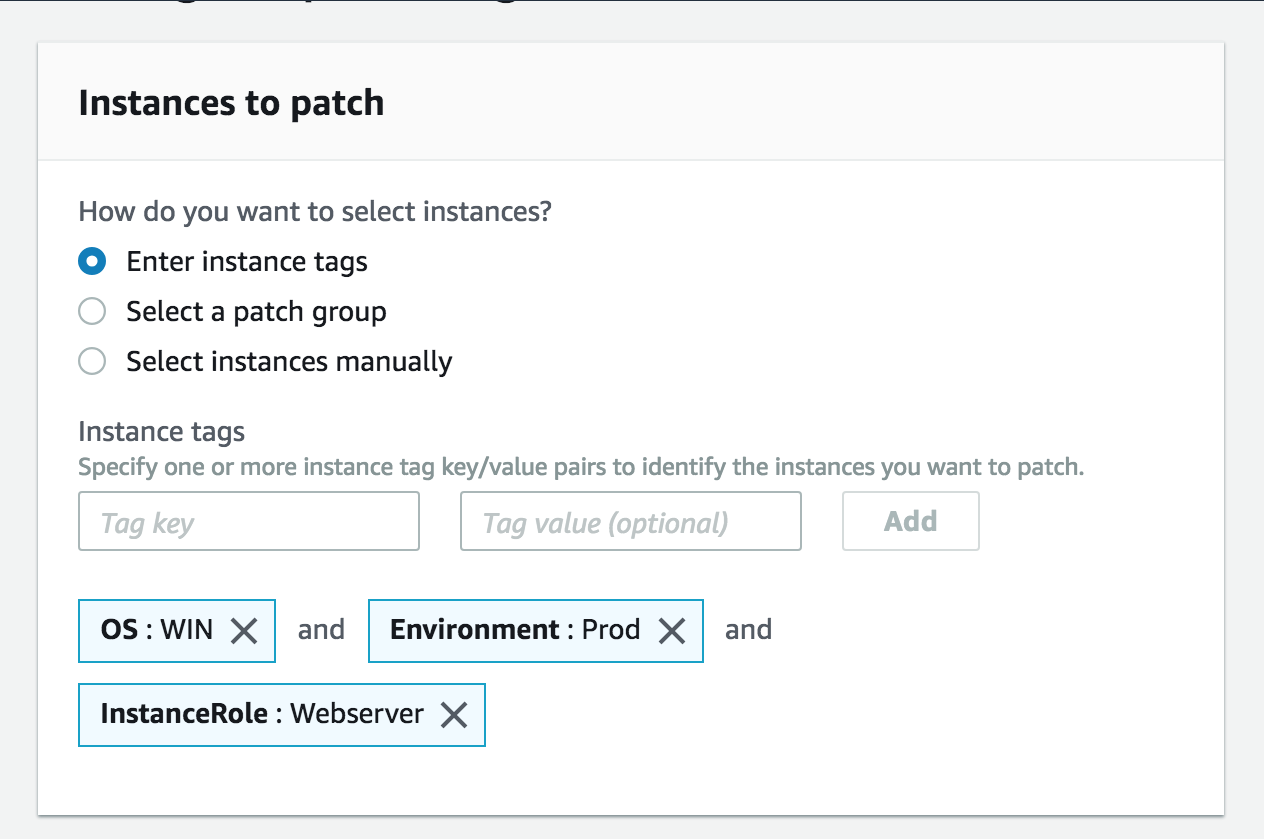
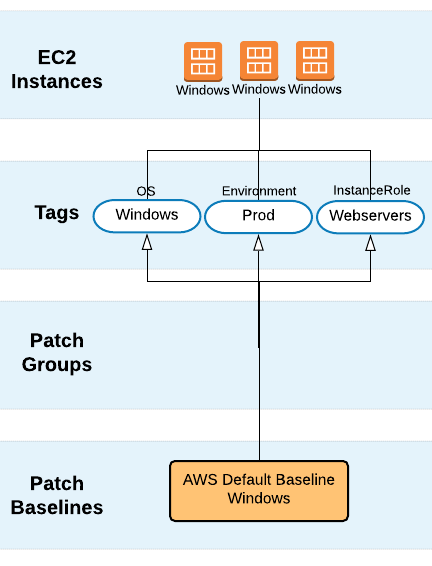


* 1. Configuring Patching – there are three ways to configure patching

* + 1.  Option 1: Configure patching manually for individual instances
       1. Select ***Patch Manager*** from the left navigation bar.
       2. Press the *Configure Patching* button
       3. For instances to patch, choose *Select instances manually*
       4. On the instances list, select your UBUNTU webserver
       5. Select *Skip scheduling and patch instances now*
       6. For patching operation, select *Scan only*
       7. Press the *Configure Patching* button. This will execute a run command immediately to scan the instance and compare against the baseline.
       8. Return to the Patch Compliance Dashboard (6.1 above) and you should see that this instance is now registered for patching and being tracked.

|  |
| --- |
| NOTE: You can only associate DEFAULT baselines when you select individual instances! |

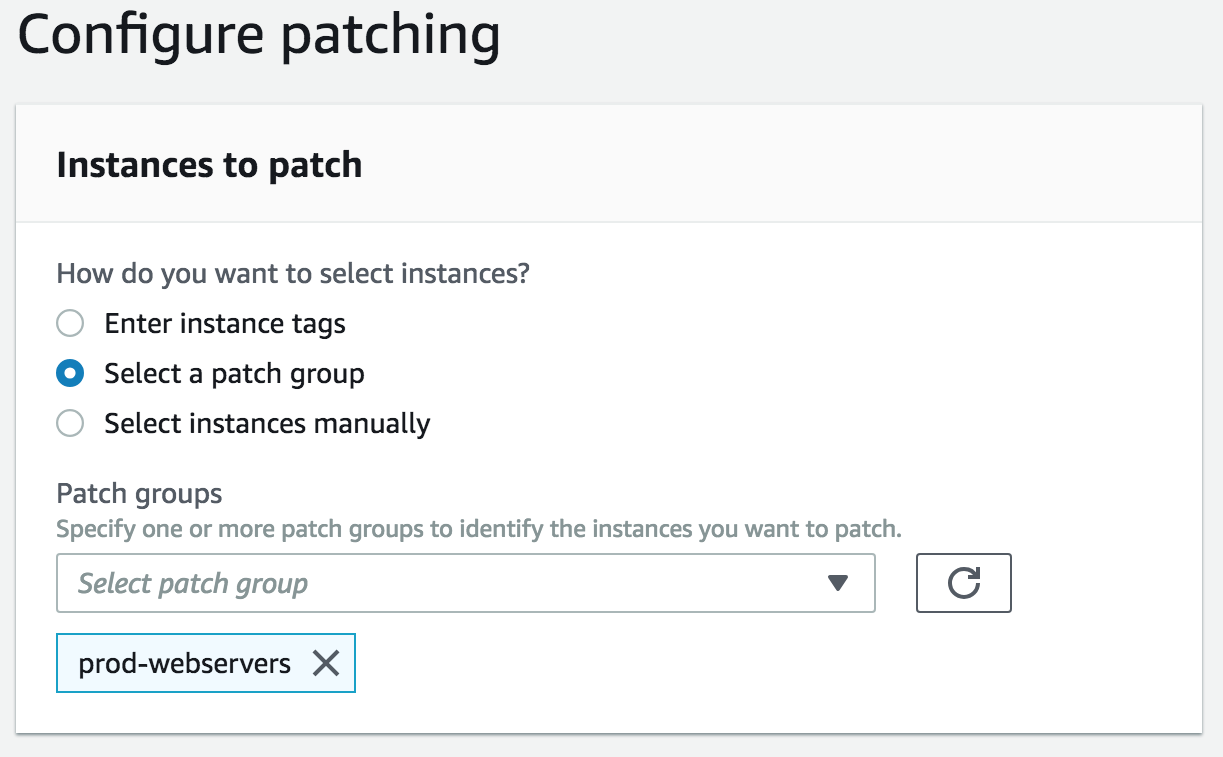
* + 1. Option 2: Configure patching using Instance Tags. For example, we will patch all Windows instances using the OS = WIN, Environment=Prod and InstanceRole=Webserver tag combinations.
       1. Select ***Patch Manager*** from the left navigation bar.
       2. Press the *Configure Patching* button
       3. Choose *Enter instance tags*
       4. For *Instance Tags* specify key/value pairs for OS, Environment and InstanceRole tags, as shown below. (NOTE: the Cloudformation template in section 1 automatically created these tags for all instances)

* + - 1. Select *Skip scheduling and patch instances now*
      2. This time, install patches on the instance. Select *Scan and install.*
      3. Press the *Configure Patching* button
      4. This will patch the instance immediately, if it is out of compliance. You can monitor the status of the run command on the *Command history* tab of the *Run Command* menu.

|  |
| --- |
| NOTE: You can only associate DEFAULT baselines using regular tags! |

* + 1. Option 3: Configure Patching using pre-existing Patch Groups
       1. Select ***Patch Manager*** from the left navigation bar.
       2. Press the *Configure Patching* button
       3. Choose *Select a Patch Group*
       4. Select the *prod-webservers* patch group you created in section 6.2.4



* + - 1. For patching schedule, select *Schedule in a new maintenance window*
      2. Select the *CRON schedule builder* with a run frequency of 30 minutes, so that it runs in a short period of time
      3. Leave the *Maintenance window duration* at 1 hour
      4. Enter a *Maintenance window name* such as *webservers-patch-windows*
      5. For *patching operation*, select *Scan and install*
      6. Press the *Configure Patching* button
      7. Check the Patch Compliance dashboard later to see patching results

1. **SSM Maintenance Windows** Goal: define a maintenance window to perform automated tasks
   1. Select ***Maintenance Windows*** from the left navigation bar.
   2. Follow the steps below (for a deeper explanation, [watch this video](https://www.youtube.com/watch?v=G5X-FjFQTos))

|  |
| --- |
| Schedule Patch Updates Using a Maintenance Window |
| To create a Maintenance Window, you must do the following:   * Create the window and define its schedule and duration. * Assign targets for the window. * Assign tasks to run during the window.   To create a Maintenance Window for patching (AWS Systems Manager)   1. Open the AWS Systems Manager console at <https://console.aws.amazon.com/systems-manager/>. 2. In the navigation pane, choose *Maintenance Windows*. 3. Choose *Create maintenance window*. 4. In the *Name* field, type a name that describes the type of patching, such as ‘critical-patches’. 5. In the top of the Schedule section, choose the schedule options you want. 6. In the *Duration* field, type the number of hours you want the Maintenance Window to be active. 7. In the *Stop initiating tasks* field, type the number of hours before the Maintenance Window duration ends that you want the system to stop initiating new tasks. 8. Choose *Create maintenance window*. 9. In the *Maintenance Windows* list, choose the Maintenance Window you just created, and then choose *Actions*, *Register targets*. 10. (Optional) In the *Maintenance window target details* section, provide a name, a description, and owner information (your name or alias) for this target. 11. In the *Targets* section, choose Specifying tags. 12. For *Tag*, enter a tag key and a tag value to identify the instances to register with the Maintenance Window. 13. Choose *Register target*. The system creates a Maintenance Window target. 14. In the details page of the Maintenance Window you created, choose *Actions*, *Register run command task*. 15. (Optional) In the *Maintenance window task details* section, provide a name and description for this task. 16. In the *Command document list*, choose AWS-RunPatchBaseline. 17. In the *Task priority* list, choose a priority (one is the highest priority). 18. In the *Targets* section, under *Target by*, choose the Maintenance Window target you created earlier in this procedure. 19. Rate control (optional):     * In *Concurrency*, specify either a number or a percentage of instances on which to run the command at the same time.  |  | | --- | | **Note:** If you selected targets by choosing Amazon EC2 tags, and you are not certain how many instances use the selected tags, then limit the number of instances that can run the document at the same time by specifying a percentage. |  * + In *Error threshold*, specify when to stop running the command on other instances after it fails on either a number or a percentage of instances. For example, if you specify 3 errors, then Systems Manager stops sending the command when the 4th error is received. Instances still processing the command might also send errors.  1. In the *Role* section, enter the ARN of a IAM role to which the AmazonSSMMaintenanceWindowRole is attached. For more information, see [Controlling Access to Maintenance Windows](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-maintenance-permissions.html). 2. In the SNS Notifications section, if you want notifications sent about the status of the command execution, select the Enable SNS notifications check box.  |  | | --- | | For more information about configuring Amazon SNS notifications for Run Command, see [Configuring Amazon SNS Notifications for Run Command](https://docs.aws.amazon.com/systems-manager/latest/userguide/rc-sns-notifications.html). |  1. In the Parameters section:    * In the *Operation list*, choose*Scan* to scan for missing patches, or choose Install to scan for and install missing patches.    * You don't need to specify anything in the Snapshot Id field. This system automatically generates and provides this parameter.    * (Optional) In the *Comment* box, enter a tracking note or reminder about this command.    * In the *Timeout* *(seconds)* box, enter the number of seconds the system should wait for the operation to finish before it is considered unsuccessful. 2. Choose *Register run command task*. |

1. **SSM Documents** Goal: learn how to create SSM documents
   1. Select ***Documents*** from the left navigation bar.
   2. Press the *Create Document* button.
      1. Give the new document a name. Call it *<your initials>-get-host-name*.
      2. Leave the document type as *Command document*.
      3. Go to the Github repo ([here](https://github.com/rjgleave/aws-ssm-immersion-day)) and find the *get-host-name.json* document. Copy it.
      4. Paste the document into the content field in the console.
      5. Click the *Create Document* button.
2. **SSM State Manager** Goal: create a custom configuration state, then execute it
   1. Select ***State Manager*** from the left navigation bar.
   2. Now create an association (configuration state) for the document created in section 8 above.
      1. Follow the steps below (or see full instructions [here](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-state-assoc.html) )

|  |
| --- |
| Create an Association |
| This section describes how to create a State Manager association by using the Amazon EC2 console. The example in this section shows you how to create an association based on a custom SSM document.  To create a State Manager association   1. Open the AWS Systems Manager console at <https://console.aws.amazon.com/systems-manager/>. 2. In the navigation pane, choose *Documents*. 3. Choose *Create document*. 4. In the *Name* field, type a descriptive name that identifies this document as a test document for State Manager. 5. In the Document type list, choose *Command document*. 6. In the Content area:    * Select the button next to JSON.    * Delete the pre-populated brackets {} in the Content, and then copy and paste the following sample document into the Content field.  |  | | --- | | {  "schemaVersion":"2.2",  "description":"Sample document",  "mainSteps":[  {  "action":"aws:runShellScript",  "name":"runShellScript",  "inputs":{  "runCommand":[  "hostname"  ]  }  }  ]  } |   The document above includes one step that invokes the *aws:runShellScript* plugin to return the instance host name. This document can be run on Linux instances.   1. In the navigation pane, choose *State Manager*. 2. Choose *Create association*. 3. In the *Name* box, type a descriptive name such as: associationTestHostnameAssociation. 4. In the *Command document list*, choose the document you just created. 5. In the *Document version list*, leave the default value. 6. Disregard the *Parameters* section, as the test document does not take parameters. 7. In the *Targets* section, identify the instances where you want to run this operation by specifying tags or selecting instances manually.  |  | | --- | | **Note**  If you choose to select instances manually, and an instance you expect to see is not included in the list, see [Where Are My Instances?](https://docs.aws.amazon.com/systems-manager/latest/userguide/troubleshooting-remote-commands.html#where-are-instances) for troubleshooting tips. |  1. Rate control (optional):    * In *Concurrency*, specify either a number or a percentage of instances on which to run the command at the same time.  |  | | --- | | **Note**  If you selected targets by choosing Amazon EC2 tags, and you are not certain how many instances use the selected tags, then limit the number of instances that can run the document at the same time by specifying a percentage. |  * + In *Error threshold*, specify when to stop running the command on other instances after it fails on either a number or a percentage of instances. For example, if you specify 3 errors, then Systems Manager stops sending the command when the 4th error is received. Instances still processing the command might also send errors.  1. Disregard the *Output options* section. Enabling the storage of command output in an S3 bucket is described in the next procedure, [Edit and Create a New Version of an Association (Console)](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-state-assoc-version.html). 2. Choose *Create association*. The system attempts to create the association on the instance and immediately apply the state. In this case, after creating the association, the system attempts to return the host name. The association status shows *Pending*. 3. Choose your browser's Refresh button. The status changes to *Success*. |

* + 1. Run the association immediately.
    2. Now, create a new version of the association. Schedule it to run every hour and save the output to S3 (the folder you created in 2.xxx above. See full instructions here>>> [**Edit and Create a New Version of an Association (Console)**](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-state-assoc-version.html)
    3. View the association history. See full instructions here >>>[**Viewing Association Histories**](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-state-assoc-history.html)

1. **Parameter Store** Goal: learn how to set up parameters
   1. Select ***Parameter Store*** from the left navigation bar.
   2. Execute all the walkthroughs ([here](https://docs.aws.amazon.com/systems-manager/latest/userguide/param-create-console.html))
2. **SSM Agent** Goal: install the SSM agent on an existing instance
3. **Inventory Reporting** Goal: learn how to report on inventory data using Athena and Quicksight
   1. Follow the abbreviated steps below to export a small subset of your inventory data to Athena (complete inventory export process, plus Quicksight described ([here](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-inventory-resource-data-sync.html))