AWS

SUMMIT

Amazon EC2 Systems Manager

Hybrid-Cloud Management at Scale

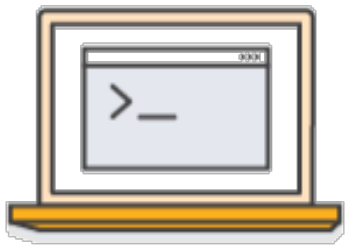
Matt Johnson, Solutions Architect, UK Public Sector

28 June 2017

amazon web services

What to expect from the session

- Overview of Amazon EC2 Systems Manager capabilities
- Setting up EC2 Systems Manager
  - IAM role configuration
  - Agent installation: EC2 and on-premises
- Walkthroughs:
  - Run Command, Associations, Inventory, Documents
  - Advanced usage example
- More use cases



Cloud is the New Normal

Trade capital expense for variable expense

Increase speed and agility

Benefit from massive economies of scale

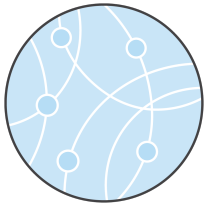
Stop spending money on running/maintaining data centers

Stop guessing capacity

Go global in minutes

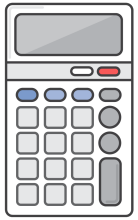
Customer challenges

Traditional IT toolset not built for cloud scale infrastructure



Maintaining enterprise-wide visibility is challenging

Deploying multiple products is a significant overhead



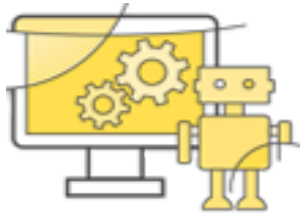
Licensing costs & complexity

Managing cloud and hybrid environments using a traditional toolset is complex and costly

Introducing Amazon EC2 Systems Manager

A set of capabilities that:

- enable automated configuration
- support ongoing management of systems at scale
- work across all of your Windows and Linux workloads
- run in Amazon EC2 or on-premises
- carry no additional charge to use



Amazon EC2 Systems Manager capabilities

Run Command

State Manager

Inventory

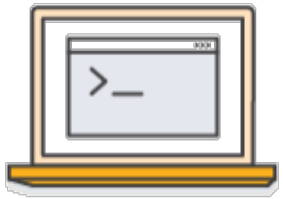
Maintenance Window

Patch Manager

Automation

Parameter Store

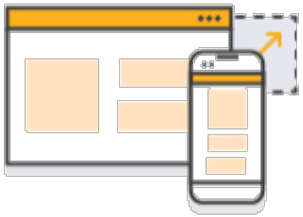
Run Command



Remotely and securely manage servers or virtual machines at scale running in your data center or in AWS

- Automate common administrative tasks
- Execute commands across multiple instances simultaneously
- Support for AWS and on-premises infrastructure
- Granular permissions to control access through AWS IAM
- Logging using AWS CloudTrail

State Manager



Define and maintain consistent configuration of operating systems and applications running in your data center or in AWS

- Control configuration details such as anti-virus settings, iptables, etc.
- Define your own schedules for deployment reviews
- Compare actual deployments against specified configuration policy
- State Manager reapplies policies if state drift is detected
- Query State Manager to view status of deployments

Inventory

Provides visibility into the software catalogue and configuration for your Amazon EC2 instances and on-premises servers

- Gather detail on a variety of attributes, such as:
  - Installed applications & OS details
  - AWS components and agents
  - Network configuration
- Inventory attributes are stored in AWS Config for auditing
- Assess compliance of configurations using AWS Config Rules

Maintenance Window

Define one or more recurring windows of time during which it is acceptable for any disruptive operation to occur

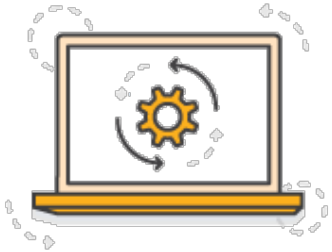
- Associate your instances with defined maintenance windows
- Create different maintenance windows for different groups of servers
- Works with both Amazon EC2 and on-premises infrastructure

Patch Manager

Automated tool that helps you simplify your Windows operating system patching process

- Select the patches you want to deploy
- Control timing for patch roll-outs and instance reboots
- Define auto-approval rules for patches
- Ability to black-list or white-list specific patches
- Schedule the automatic roll out through maintenance windows

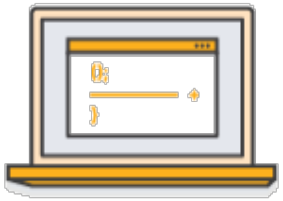
Automation



Simplifies common maintenance and deployment tasks, such as updating Amazon Machine Images (AMIs)

- Patch, update agents, or bake applications into your AMIs
- Build workflows to accomplish complex tasks
- Use pre-defined workflows or build your own

Parameter Store



Centralized store to manage your configuration data, including plain-text data or secrets, encrypted through AWS KMS

- Critical information stored securely within your environment
  - Integrates with AWS IAM, AWS KMS, AWS CloudTrail
- Re-use across your AWS configuration and automation workflows
- Reference parameters from:
  - Other Amazon EC2 Systems Manager capabilities (Run Command, Automation, State Manager, etc.)
  - other AWS services (Amazon ECS, AWS Lambda, etc.)

Pre-requisites

Prerequisites

- User IAM access to Amazon EC2 Systems Manager
- For managed EC2 instances:
  - Amazon EC2 Instance Role
- For managed on-premises instances:
  - AWS IAM Service Role
  - EC2 Systems Manager Activation code
- SSM Agent installed on managed instances
  - Outbound Internet (https) access for the instance

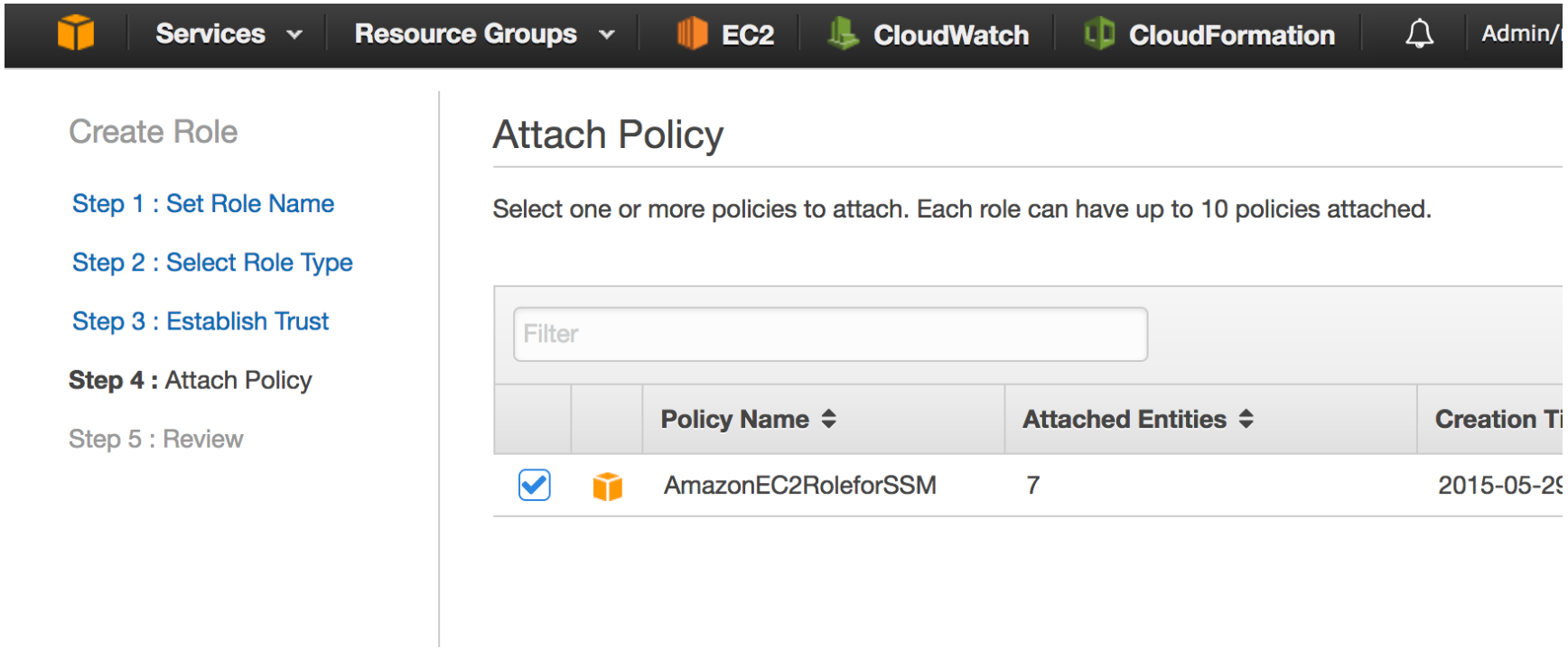
Choose the Amazon EC2 Role for SSM Role

Select Role Type

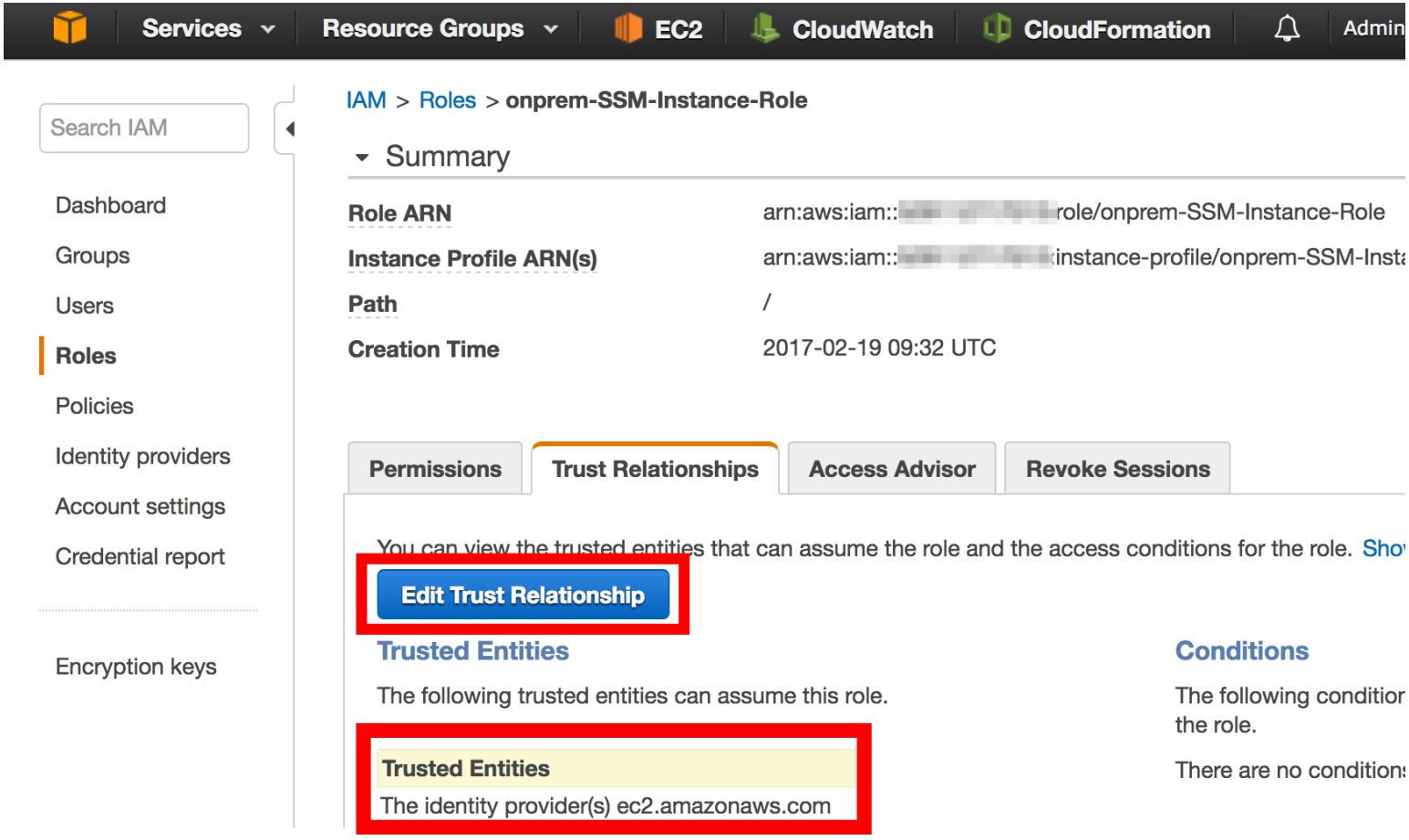
AWS Service Roles

Amazon RDS Allows RDS to call AWS services on your behalf.	Select
Amazon RDS Role for Enhanced Monitoring Allows RDS to manage CloudWatch Logs resources for Enhanced Monitoring on your behalf.	Select
Amazon SNS Allows SNS to call CloudWatchLogs on your behalf	Select
AWS Service Catalog Allows AWS Service Catalog to access AWS resources on your behalf.	Select
Amazon EC2 Role for Simple Systems Manager Provides access to Amazon Simple Systems Manager(SSM), CloudWatch, EC2 and supported plugins in SSM document.	Select

Attach the AWS Managed Policy



Edit the Trust Relationship (on-prem role)



Edit the Trust Relationship (on-prem role)

Edit Trust Relationship

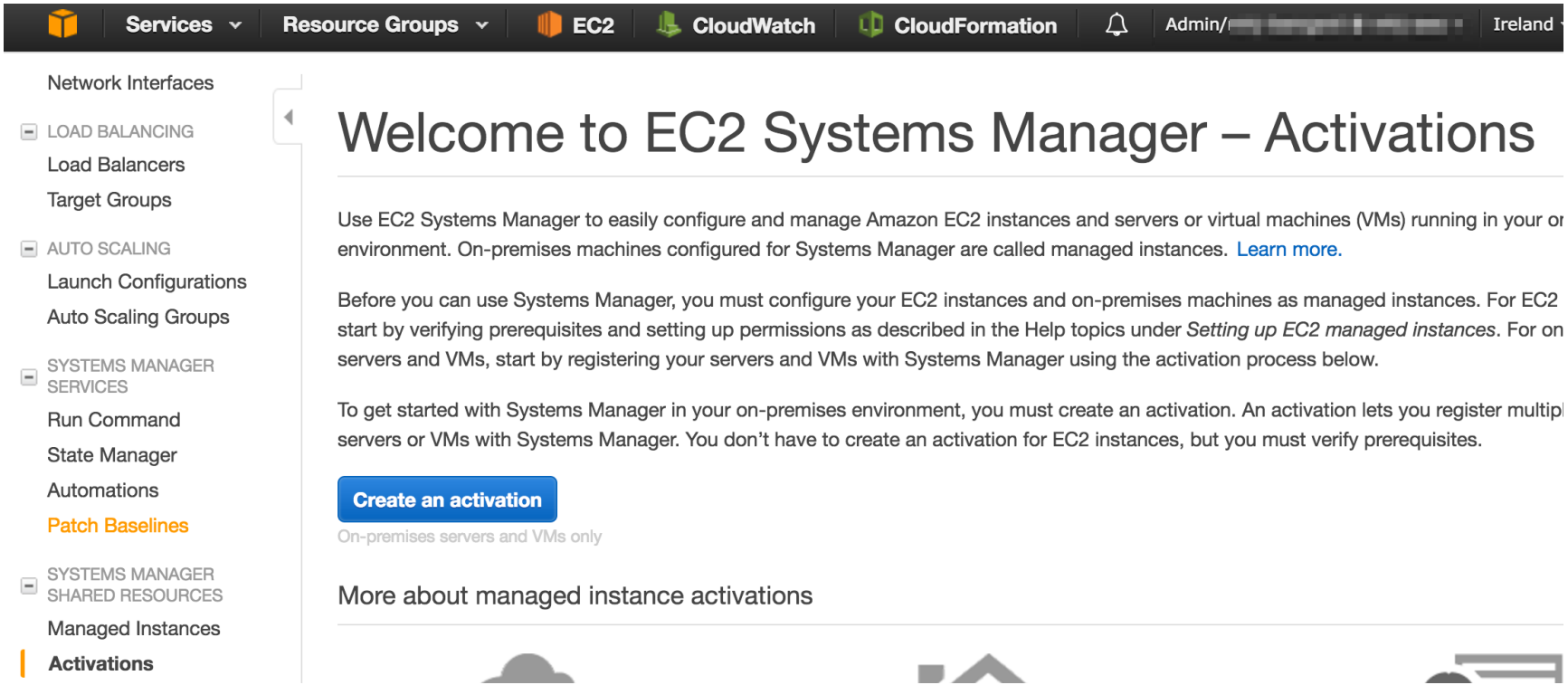
You can customize trust relationships by editing the following access control policy document.

Policy Document

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

http://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-maintenance-permissions.html

Activations for on-premises servers



Create a new Activation using the service role

Creating a new activation allows you to generate a code which can be used to register a run command agent on instances. Specify the details below to cre

Activation description onprem-prod-fleet

Instance limit 50

IAM Role Name* Create a system default command execution role that has the required permissions

Select an existing custom IAM role that has the required permissions

If you select this option, AWS uses an existing role that you specify. The role must have the required permissions or commands fail to execute. Learn more about the minimum required permissions.

onprem-SSM-Instance-Role Add new custom role

Activation expiry date 2017-03-17T00:00+00:00

Default Instance name onprem-prod-fleet

Note down the Activation Code and ID

Activations > Create Activation

Create Activation

Success

You have successfully created a new activation (54ac8bd2-[redacted]).
 Your activation code is listed below. <b>Copy this code and keep it in a safe place as you will not be able to access it again.

Activation Code [redacted]

Activation ID 54ac8bd2-[redacted]

You can now install amazon-ssm-agent and manage your instance using Run Command.Learn more

View result

Amazon SSM Agent Overview

Processes SSM requests and configures instances

Supported Linux operating systems:

- Amazon Linux 2014.03 and later
- Ubuntu 12.04 LTS, 14.04 LTS, 16.04 LTS
- RHEL 6.5+, CentOS 6.3+, SUSE 12+ 

Supported Windows operating systems:

- Windows Server 2003+, including R2 versions

Source code available on GitHub:

- https://github.com/aws/amazon-ssm-agent

Amazon SSM Agent Installation – Linux

Amazon EC2 instances (Amazon Linux, RedHat 6.x, etc.)

```
mkdir /tmp/ssm
REGION=`curl -s http://169.254.169.254/latest/dynamic/instance-identity/document/ | grep "region" |
awk -F\" ' { print $4 }'`

curl https://amazon-ssm-$REGION.s3.amazonaws.com/latest/linux_amd64/amazon-ssm-agent.rpm -o
/tmp/ssm/amazon-ssm-agent.rpm
sudo yum install -y /tmp/ssm/amazon-ssm-agent.rpm
```

On-premises servers:

```
mkdir /tmp/ssm
REGION=‘eu-west-2”  # Specifies the region in which to register the on-premises instances

curl https://amazon-ssm-$REGION.s3.amazonaws.com/latest/linux_amd64/amazon-ssm-agent.rpm -o
/tmp/ssm/amazon-ssm-agent.rpm
sudo yum install -y /tmp/ssm/amazon-ssm-agent.rpm
sudo stop amazon-ssm-agent
sudo amazon-ssm-agent -register -code "code" -id "id" -region "$REGION" sudo start amazon-ssm-agent
```

Amazon SSM Agent Installation – Windows

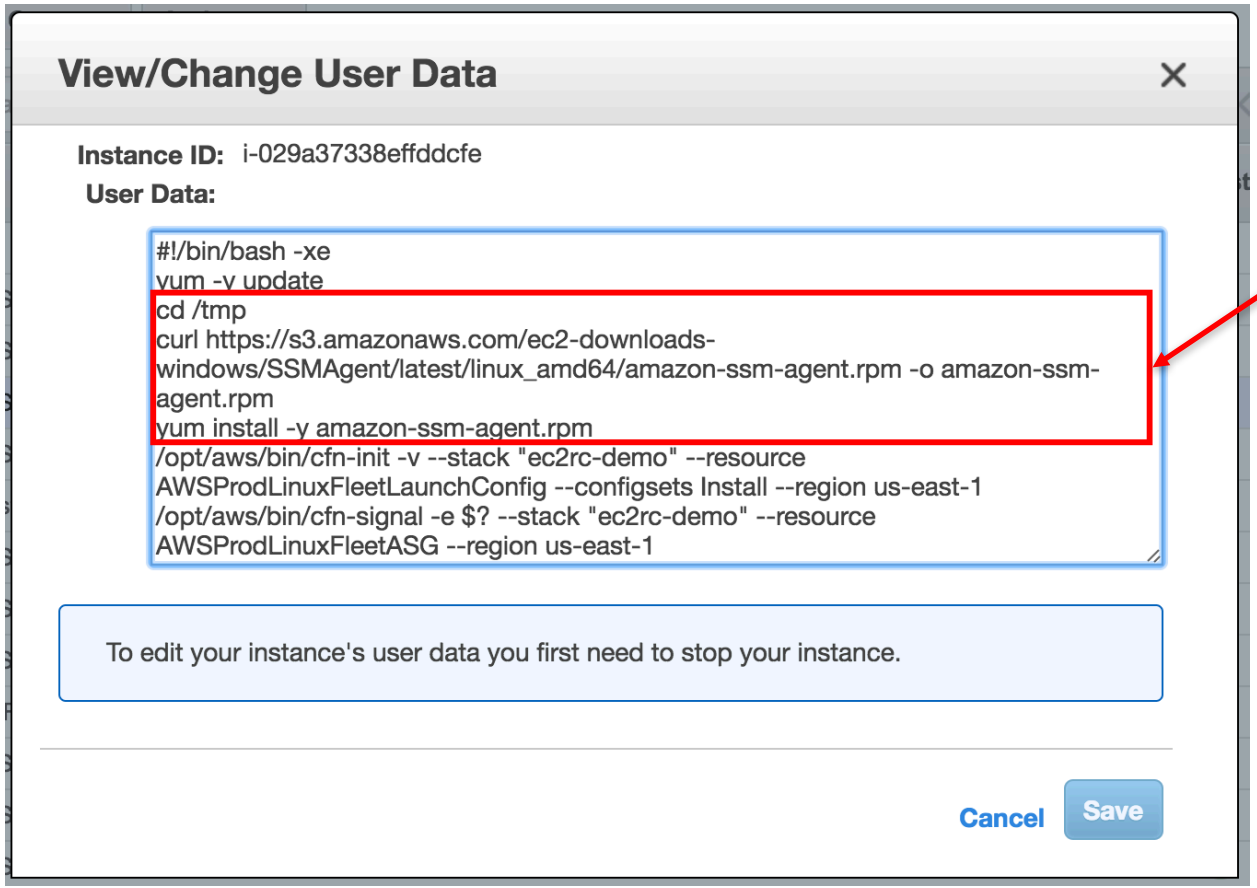
Amazon EC2 instances

```
$ Download: https://amazon-ssm-
region.s3.amazonaws.com/latest/windows_amd64/AmazonSSMAgentSetup.exe
$ Restart-Service AmazonSSMAgent
```

On-premises servers:

```
$dir = $env:TEMP + "\ssm”
New-Item -ItemType directory -Path $dir
cd $dir
(New-Object System.Net.WebClient).DownloadFile("https://amazon-ssm-
region.s3.amazonaws.com/latest/windows_amd64/AmazonSSMAgentSetup.exe", $dir +
"\AmazonSSMAgentSetup.exe")
Start-Process .\AmazonSSMAgentSetup.exe -ArgumentList @("/q", "/log", "install.log",
"CODE=code", "ID=id", "REGION=region") -Wait
Get-Content ($env:ProgramData + "\Amazon\SSM\InstanceData\registration")
Get-Service -Name "AmazonSSMAgent"
```

Boot-strapping installation – EC2 User Data



Installation instructions

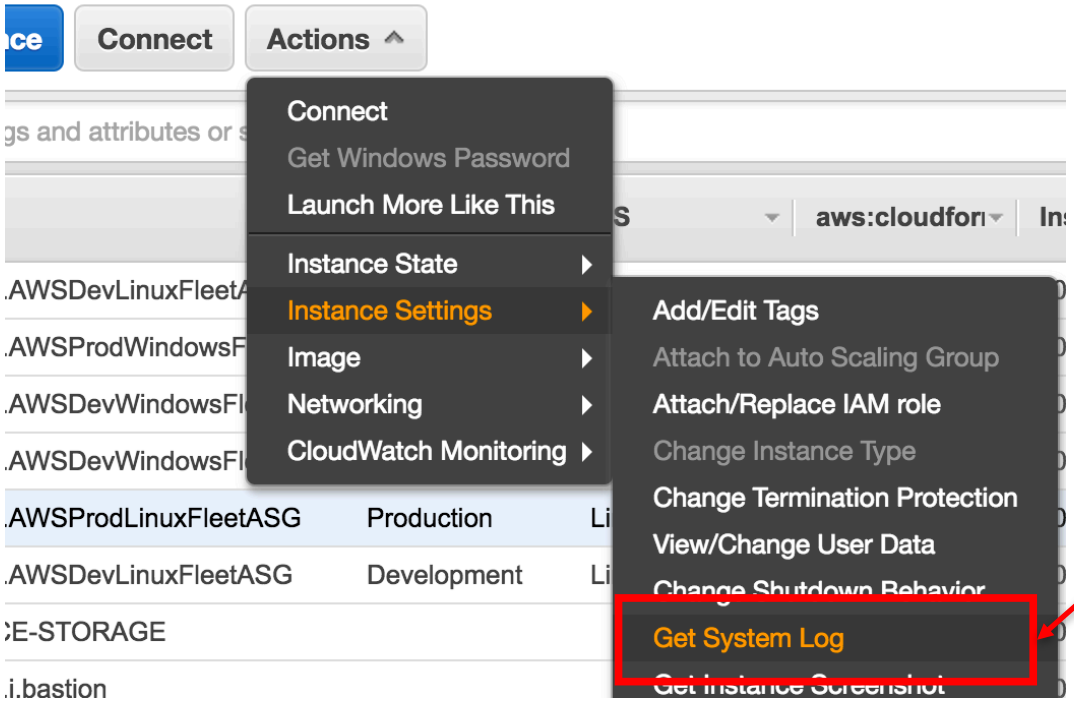
IAM role ec2rc-demo-FleetRole-23MG3TQDGB8F

IAM Role Attachment

Boot-strapping installation – CloudFormation

```
Server:
  Type: AWS::EC2::Instance
  Metadata:
    AWS::CloudFormation::Init:
      configSets:
        AWSTools:
          - "ssmInstall”
      ssmInstall:
        packages:
          rpm:
            amazon-ssm-agent: https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm
        commands:
          01-stopssm:
            command: "stop amazon-ssm-agent”
          02-startssm:
            command: "start amazon-ssm-agent”
  Properties:
    IamInstanceProfile: !Ref EC2SSMProfile
    UserData:
      "Fn::Base64":
        !Sub |
          #!/bin/bash –xe
          yum -y update
          /opt/aws/bin/cfn-init -v --stack "${AWS::StackName}" --resource Server--configsets AWSTools --region ${AWS::Region}
          echo Startup completed.
    ...
```

Checking installation status

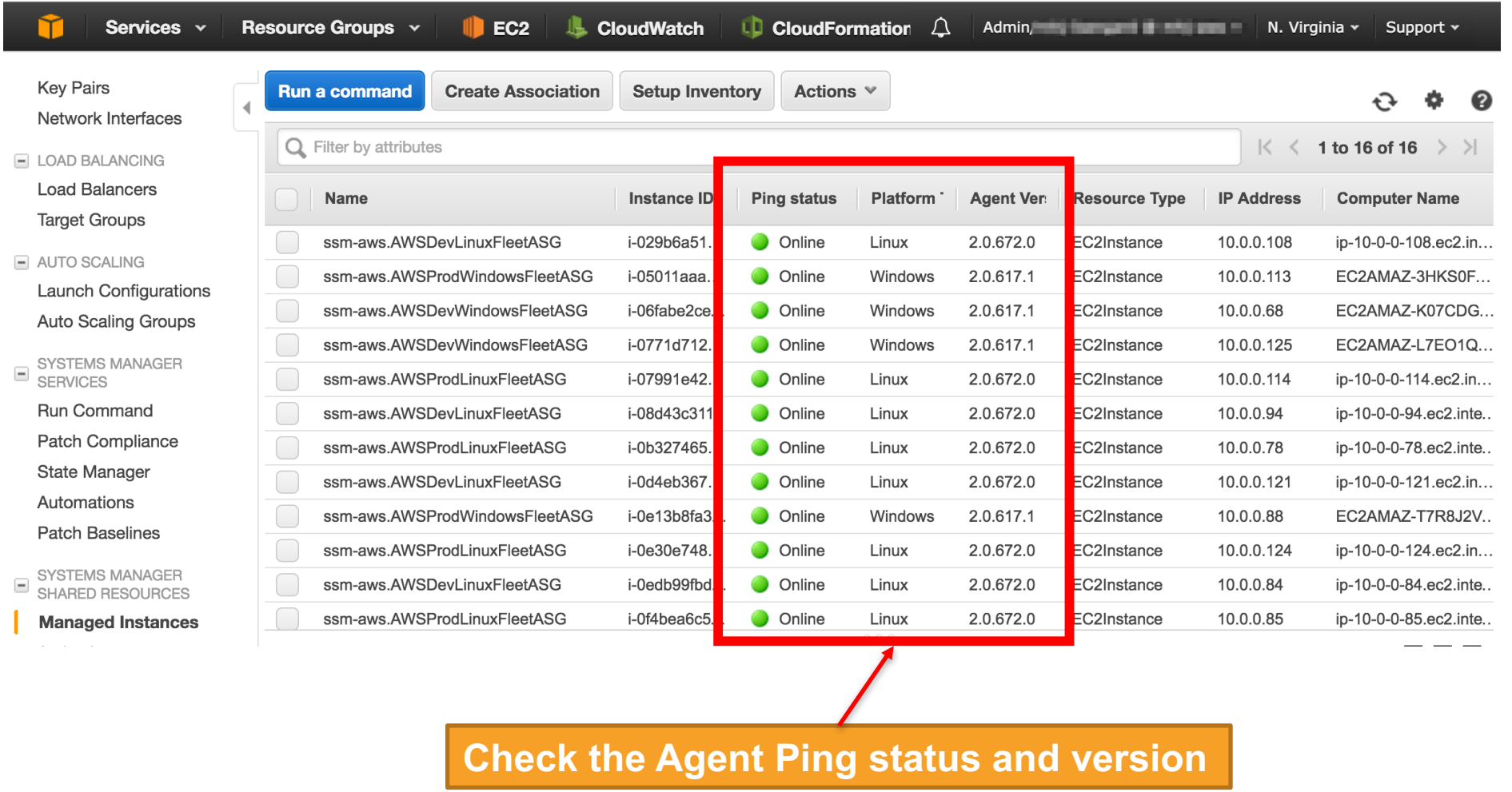


View the System Log of the instance

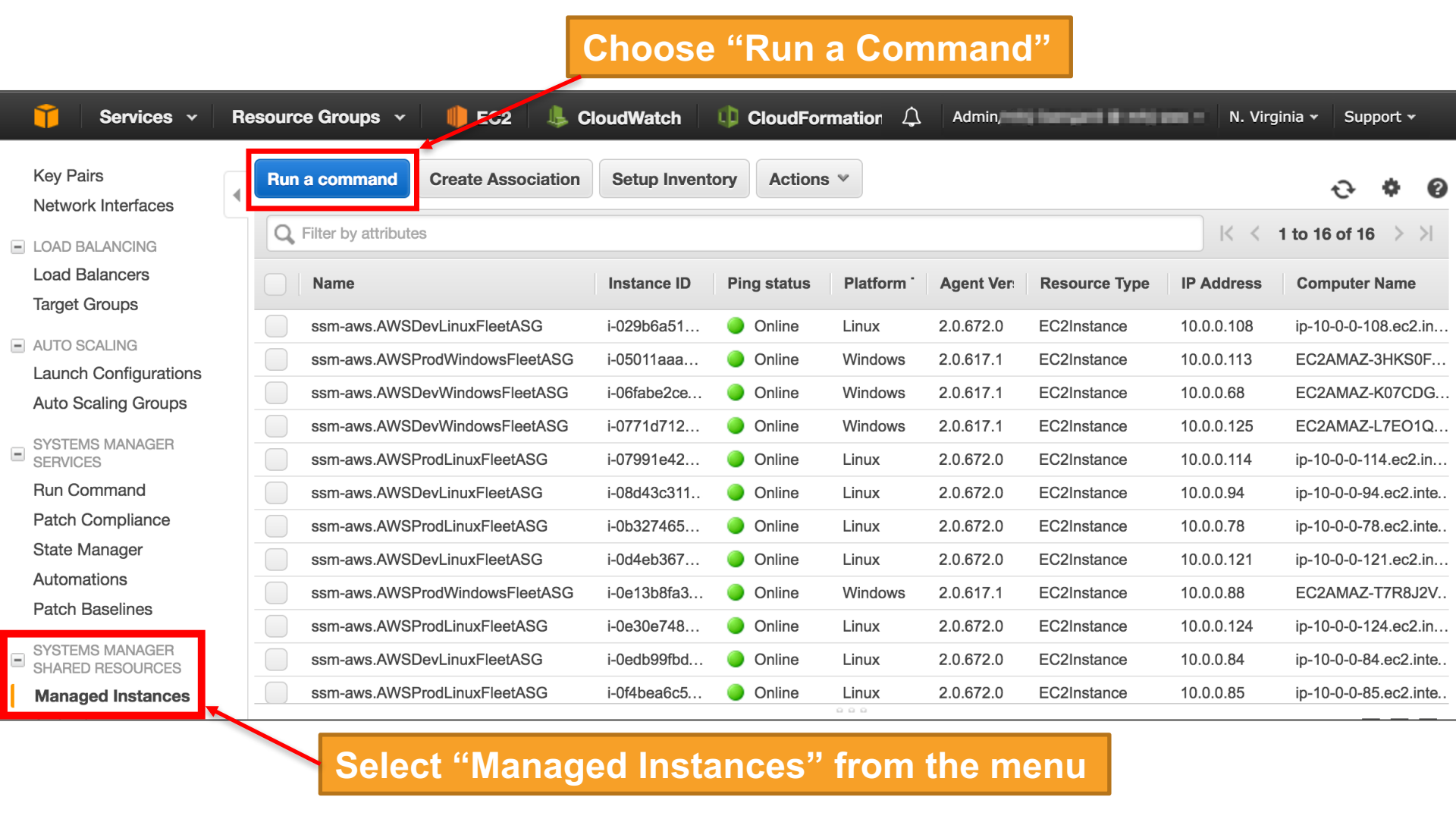
Checking installation status

```
--> Running transaction check
---> Package amazon-ssm-agent.x86_64 0:2.0.672.0-1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package              Arch       Version              Repository           Size
================================================================================
Installing:
 amazon-ssm-agent     x86_64     2.0.672.0-1          /amazon-ssm-agent    16 M

Transaction Summary
================================================================================
Install  1 Package

Total size: 16 M
Installed size: 16 M
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : amazon-ssm-agent-2.0.672.0-1.x86_64                         1/1
amazon-ssm-agent start/running, process 8061
  Verifying  : amazon-ssm-agent-2.0.672.0-1.x86_64                         1/1
2017/02/22 09:01:56Z: OsProductName: Amazon Linux AMI
2017/02/22 09:01:56Z: OsVersion: 2016.09

Installed:
  amazon-ssm-agent.x86_64 0:2.0.672.0-1

Complete!
```

Check that installation succeeded


Services
Resource Groups
EC2
CloudWatch
CloudFormation
Admin
N. Virginia
Support
Key Pairs
Network Interfaces
LOAD BALANCING
Load Balancers
Target Groups
AUTO SCALING
Launch Configurations
Auto Scaling Groups
SYSTEMS MANAGER SERVICES
Run Command
Patch Compliance
State Manager
Automations
Patch Baselines
SYSTEMS MANAGER SHARED RESOURCES
Managed Instances
Run a command
Create Association
Setup Inventory
Actions
Filter by attributes
1 to 16 of 16
Name
Instance ID
Ping status
Platform
Agent Ver
Resource Type
IP Address
Computer Name
ssm-aws.AWSDevLinuxFleetASG i-029b6a51. Online Linux 2.0.672.0 EC2Instance 10.0.0.108 ip-10-0-0-108.ec2.in...
ssm-aws.AWSProdWindowsFleetASG i-05011aaa. Online Windows 2.0.617.1 EC2Instance 10.0.0.113 EC2AMAZ-3HKS0F...
ssm-aws.AWSDevWindowsFleetASG i-06fabe2ce. Online Windows 2.0.617.1 EC2Instance 10.0.0.68 EC2AMAZ-K07CDG...
ssm-aws.AWSDevWindowsFleetASG i-0771d712. Online Windows 2.0.617.1 EC2Instance 10.0.0.125 EC2AMAZ-L7EO1Q...
ssm-aws.AWSProdLinuxFleetASG i-07991e42. Online Linux 2.0.672.0 EC2Instance 10.0.0.114 ip-10-0-0-114.ec2.in...
ssm-aws.AWSDevLinuxFleetASG i-08d43c311 Online Linux 2.0.672.0 EC2Instance 10.0.0.94 ip-10-0-0-94.ec2.inte..
ssm-aws.AWSProdLinuxFleetASG i-0b327465. Online Linux 2.0.672.0 EC2Instance 10.0.0.78 ip-10-0-0-78.ec2.inte..
ssm-aws.AWSDevLinuxFleetASG i-0d4eb367. Online Linux 2.0.672.0 EC2Instance 10.0.0.121 ip-10-0-0-121.ec2.in...
ssm-aws.AWSProdWindowsFleetASG i-0e13b8fa3. Online Windows 2.0.617.1 EC2Instance 10.0.0.88 EC2AMAZ-T7R8J2V..
ssm-aws.AWSProdLinuxFleetASG i-0e30e748. Online Linux 2.0.672.0 EC2Instance 10.0.0.124 ip-10-0-0-124.ec2.in...
ssm-aws.AWSDevLinuxFleetASG i-0edb99fbd. Online Linux 2.0.672.0 EC2Instance 10.0.0.84 ip-10-0-0-84.ec2.inte..
ssm-aws.AWSProdLinuxFleetASG i-0f4bea6c5. Online Linux 2.0.672.0 EC2Instance 10.0.0.85 ip-10-0-0-85.ec2.inte..
Check the Agent Ping status and version

Walkthrough: Run Command

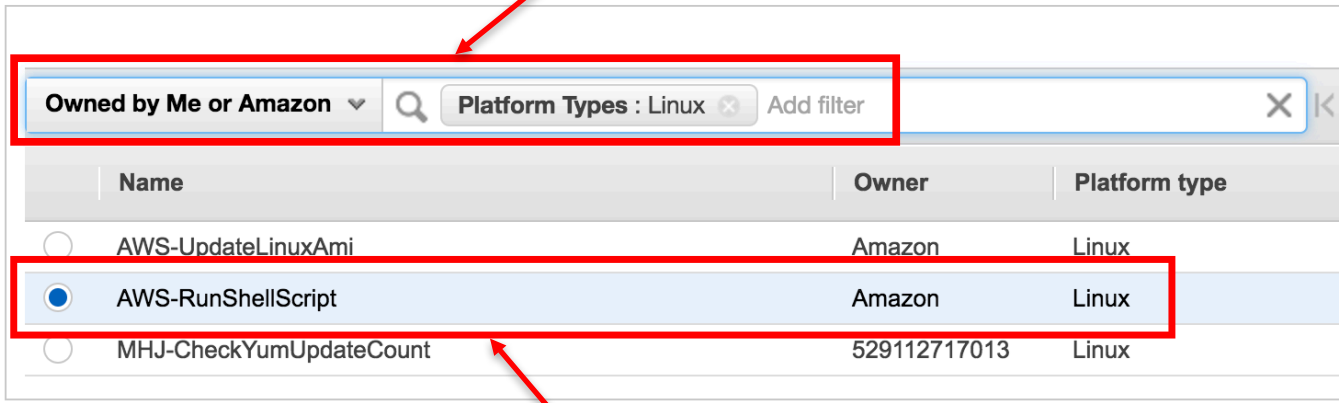

Choose “Run a Command”
Services
Resource Groups
EC2
CloudWatch
CloudFormation
Admin
N. Virginia
Support
Key Pairs
Network Interfaces
LOAD BALANCING
Load Balancers
Target Groups
AUTO SCALING
Launch Configurations
Auto Scaling Groups
SYSTEMS MANAGER SERVICES
Run Command
Patch Compliance
State Manager
Automations
Patch Baselines
SYSTEMS MANAGER SHARED RESOURCES
Managed Instances
Run a command
Create Association
Setup Inventory
Actions
Filter by attributes
1 to 16 of 16
Name
Instance ID
Ping status
Platform
Agent Ver
Resource Type
IP Address
Computer Name
ssm-aws.AWSDevLinuxFleetASG i-029b6a51… Online Linux 2.0.672.0 EC2Instance 10.0.0.108 ip-10-0-0-108.ec2.in…
ssm-aws.AWSProdWindowsFleetASG i-05011aaa… Online Windows 2.0.617.1 EC2Instance 10.0.0.113 EC2AMAZ-3HKS0F…
ssm-aws.AWSDevWindowsFleetASG i-06fabe2ce… Online Windows 2.0.617.1 EC2Instance 10.0.0.68 EC2AMAZ-K07CDG…
ssm-aws.AWSDevWindowsFleetASG i-0771d712… Online Windows 2.0.617.1 EC2Instance 10.0.0.125 EC2AMAZ-L7EO1Q…
ssm-aws.AWSProdLinuxFleetASG i-07991e42… Online Linux 2.0.672.0 EC2Instance 10.0.0.114 ip-10-0-0-114.ec2.in…
ssm-aws.AWSDevLinuxFleetASG i-08d43c311.. Online Linux 2.0.672.0 EC2Instance 10.0.0.94 ip-10-0-0-94.ec2.inte..
ssm-aws.AWSProdLinuxFleetASG i-0b327465… Online Linux 2.0.672.0 EC2Instance 10.0.0.78 ip-10-0-0-78.ec2.inte..
ssm-aws.AWSDevLinuxFleetASG i-0d4eb367… Online Linux 2.0.672.0 EC2Instance 10.0.0.121 ip-10-0-0-121.ec2.in…
ssm-aws.AWSProdWindowsFleetASG i-0e13b8fa3… Online Windows 2.0.617.1 EC2Instance 10.0.0.88 EC2AMAZ-T7R8J2V..
ssm-aws.AWSProdLinuxFleetASG i-0e30e748… Online Linux 2.0.672.0 EC2Instance 10.0.0.124 ip-10-0-0-124.ec2.in…
ssm-aws.AWSDevLinuxFleetASG i-0edb99fbd… Online Linux 2.0.672.0 EC2Instance 10.0.0.84 ip-10-0-0-84.ec2.inte..
ssm-aws.AWSProdLinuxFleetASG i-0f4bea6c5… Online Linux 2.0.672.0 EC2Instance 10.0.0.85 ip-10-0-0-85.ec2.inte..
Select “Managed Instances” from the menu

Commands > Run a command

Run a command

A command document includes the information about the command you want to run. Select a command document from the following list and then specify parar

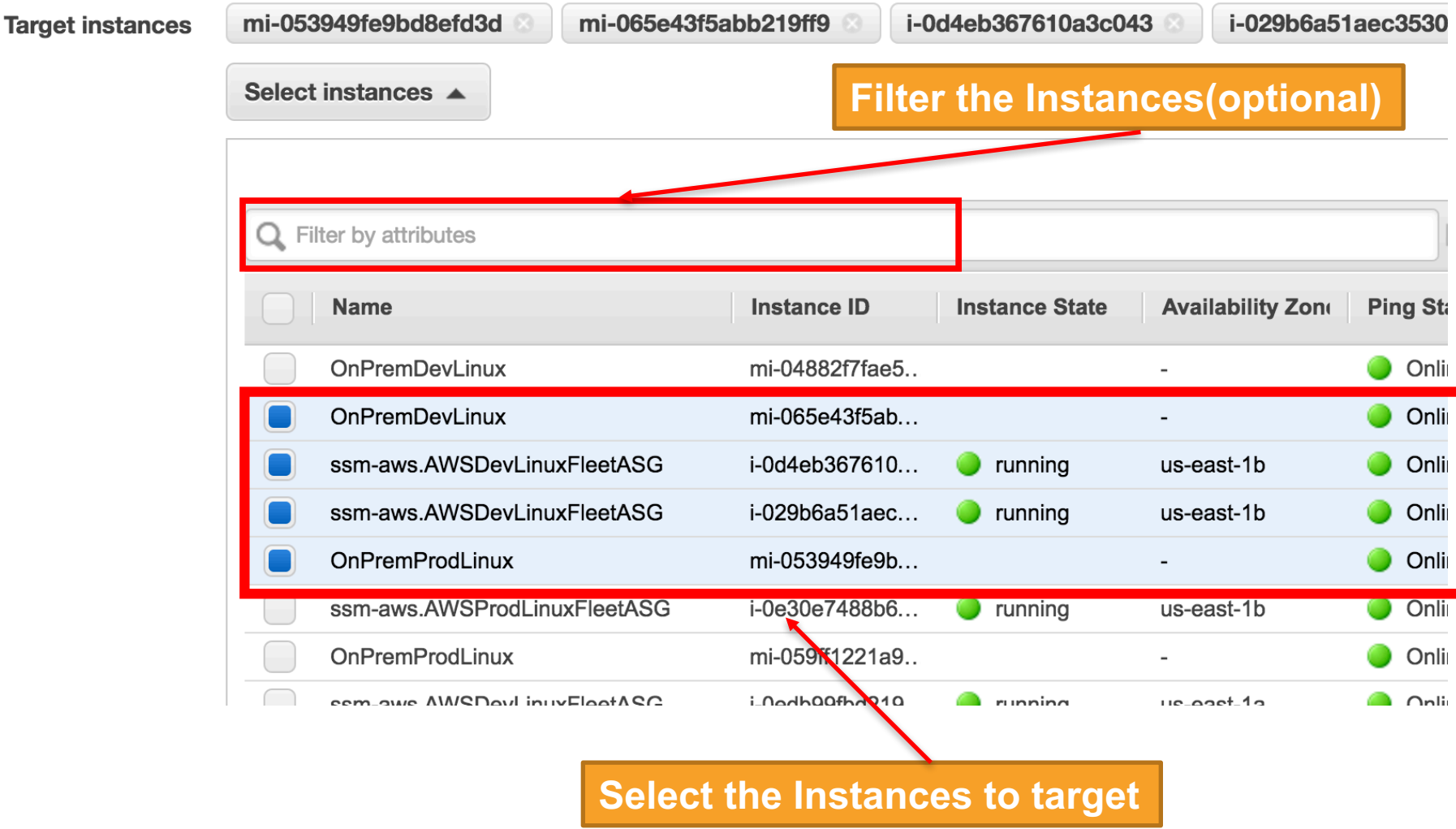
Command document*

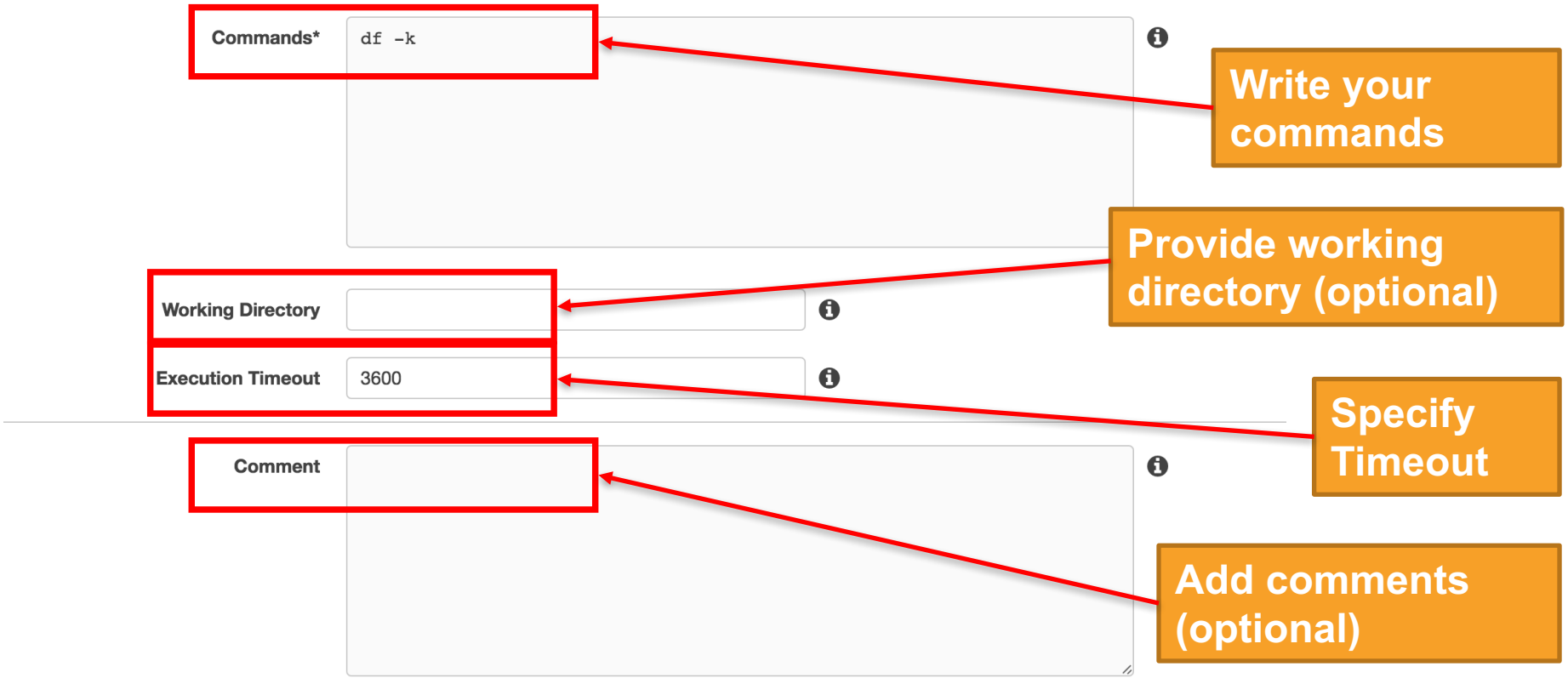


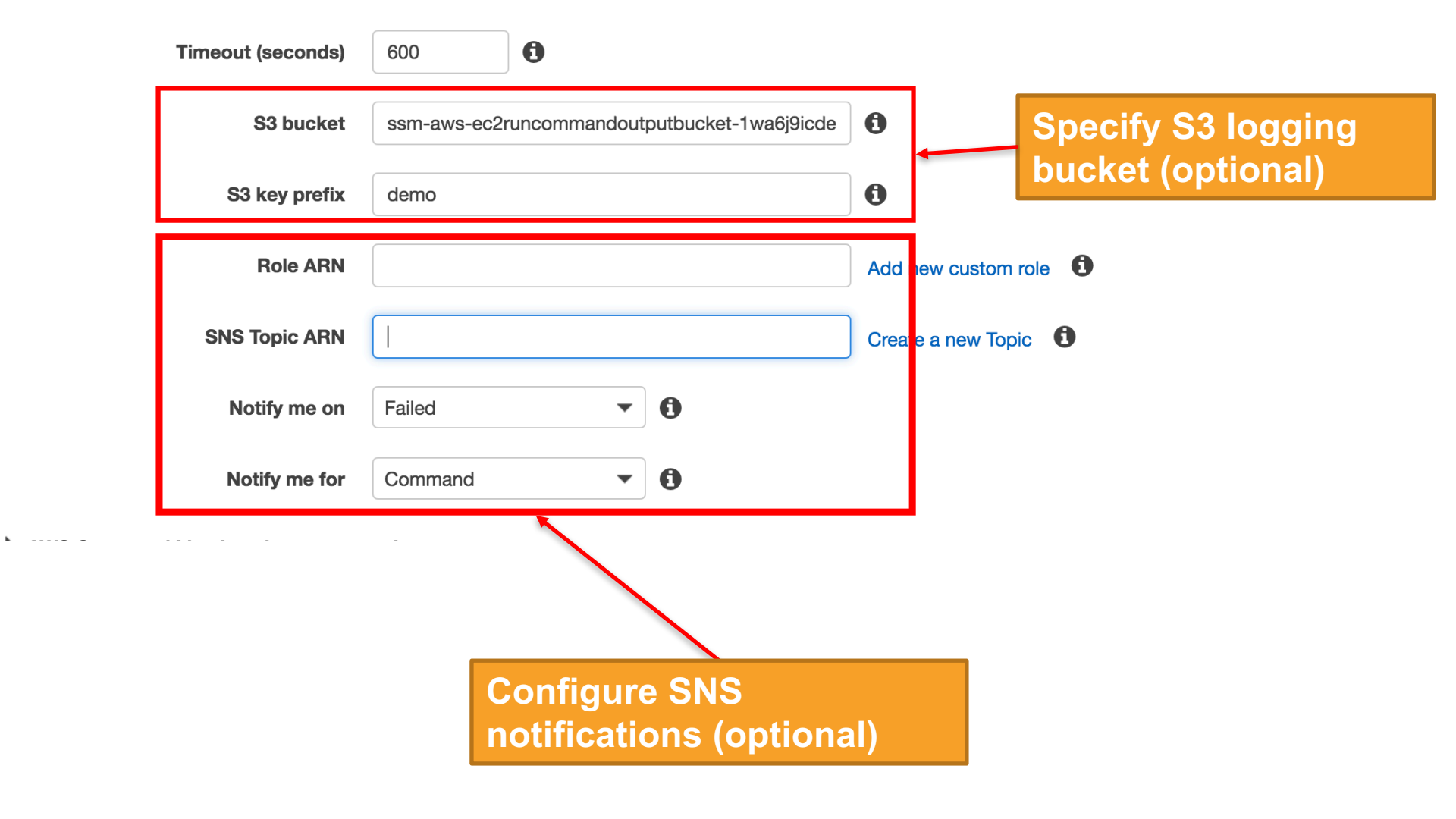
Description Run a shell script or specify the commands to run.

Filter the Command Documents (optional)

Choose the Command Document


Target instances
mi-053949fe9bd8efd3d
mi-065e43f5abb219ff9
i-0d4eb367610a3c043
i-029b6a51aec3530
Select instances
Filter the Instances(optional)
Filter by attributes
Name
Instance ID
Instance State
Availability Zone
Ping Status
OnPremDevLinux
mi-04882f7fae5..
-
OnPremDevLinux
mi-065e43f5ab...
-
ssm-aws.AWSDevLinuxFleetASG
i-0d4eb367610...
running
us-east-1b
ssm-aws.AWSDevLinuxFleetASG
i-029b6a51aec...
running
us-east-1b
OnPremProdLinux
mi-053949fe9b...
-
ssm-aws.AWSProdLinuxFleetASG
i-0e30e7488b6...
running
us-east-1b
OnPremProdLinux
mi-059ff1221a9..
-
Select the Instances to target


Commands*
df -k
Working Directory
Execution Timeout
3600
Comment
Write your commands
Provide working directory (optional)
Specify Timeout
Add comments (optional)


Timeout (seconds)
600
S3 bucket
ssm-aws-ec2runcommandoutputbucket-1wa6j9icde
S3 key prefix
demo
Specify S3 logging bucket (optional)
Role ARN
Add new custom role
SNS Topic ARN
Create a new Topic
Notify me on
Failed
Notify me for
Command
Configure SNS notifications (optional)

Run a command

Success

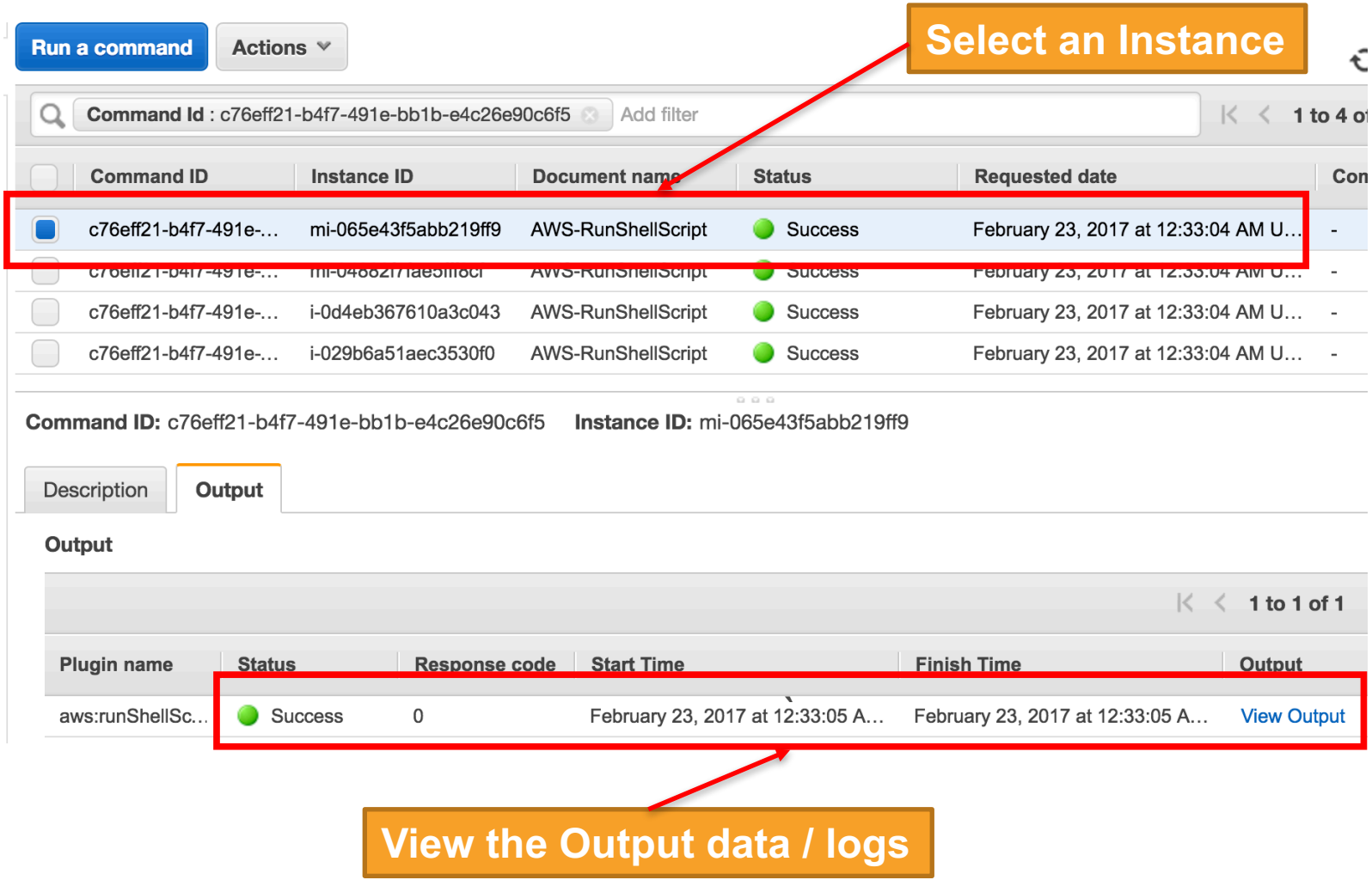
We are running your command against the instances listed below.

Instance IDs i-029b6a51aec3530f0, i-0d4eb367610a3c043, mi-065e43f5abb219ff9, mi-04882f7fae5fff8cf

Command ID 67e40f4c-d84d-469d-8cf4-bad657df8165

View result

View the command invocations


Run a command
Actions
Select an Instance
Command Id : c76eff21-b4f7-491e-bb1b-e4c26e90c6f5
Add filter
1 to 4 of
Command ID
Instance ID
Document name
Status
Requested date
c76eff21-b4f7-491e-...
mi-065e43f5abb219ff9
AWS-RunShellScript
Success
February 23, 2017 at 12:33:04 AM U...
c76eff21-b4f7-491e-...
i-0d4eb367610a3c043
AWS-RunShellScript
Success
February 23, 2017 at 12:33:04 AM U...
c76eff21-b4f7-491e-...
i-029b6a51aec3530f0
AWS-RunShellScript
Success
February 23, 2017 at 12:33:04 AM U...
Command ID: c76eff21-b4f7-491e-bb1b-e4c26e90c6f5
Instance ID: mi-065e43f5abb219ff9
Description
Output
Output
1 to 1 of 1
Plugin name
Status
Response code
Start Time
Finish Time
Output
aws:runShellSc...
Success
0
February 23, 2017 at 12:33:05 A...
February 23, 2017 at 12:33:05 A...
View Output
View the Output data / logs

Output for aws:runShellScript

```
Filesystem     1K-blocks    Used Available Use% Mounted on
devtmpfs          240728      56    240672   1% /dev
tmpfs             251604       0    251604   0% /dev/shm
/dev/xvda1       8123812 1030708   6992856  13% /
```

Concurrency Model

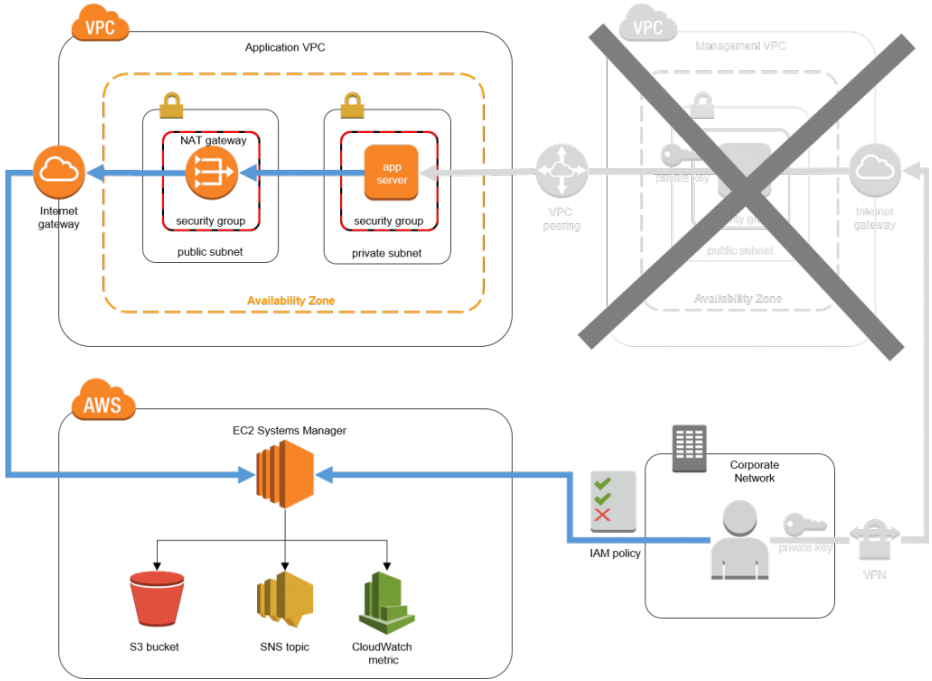
- Concurrency controls allow you to specify max velocity:
  - Execute simultaneously on maximum of N instances, or
  - Execute simultaneously on maximum of N% instances
- Queuing system sends commands exponentially, starting with a single instance, until it reaches max-concurrency

Error Handling

- Error handling allows you to specify an error threshold:
  - Maximum of N error responses, or
  - Maximum of N% of the target set error responses
- Queuing system will stop sending commands once the error threshold has been breached

Example: Replacing Bastion Hosts

- Replace your bastion host by using Amazon EC2 Systems Manager
- Reduce your system's attack surface
- Offers greater visibility into commands issued on your hosts
- Granular IAM controls



https://aws.amazon.com/blogs/mt/replacing-a-bastion-host-with-amazon-ec2-systems-manager/

Walkthrough: State Manager

Associations (via State Manager)

Determine the **actions** to be applied:

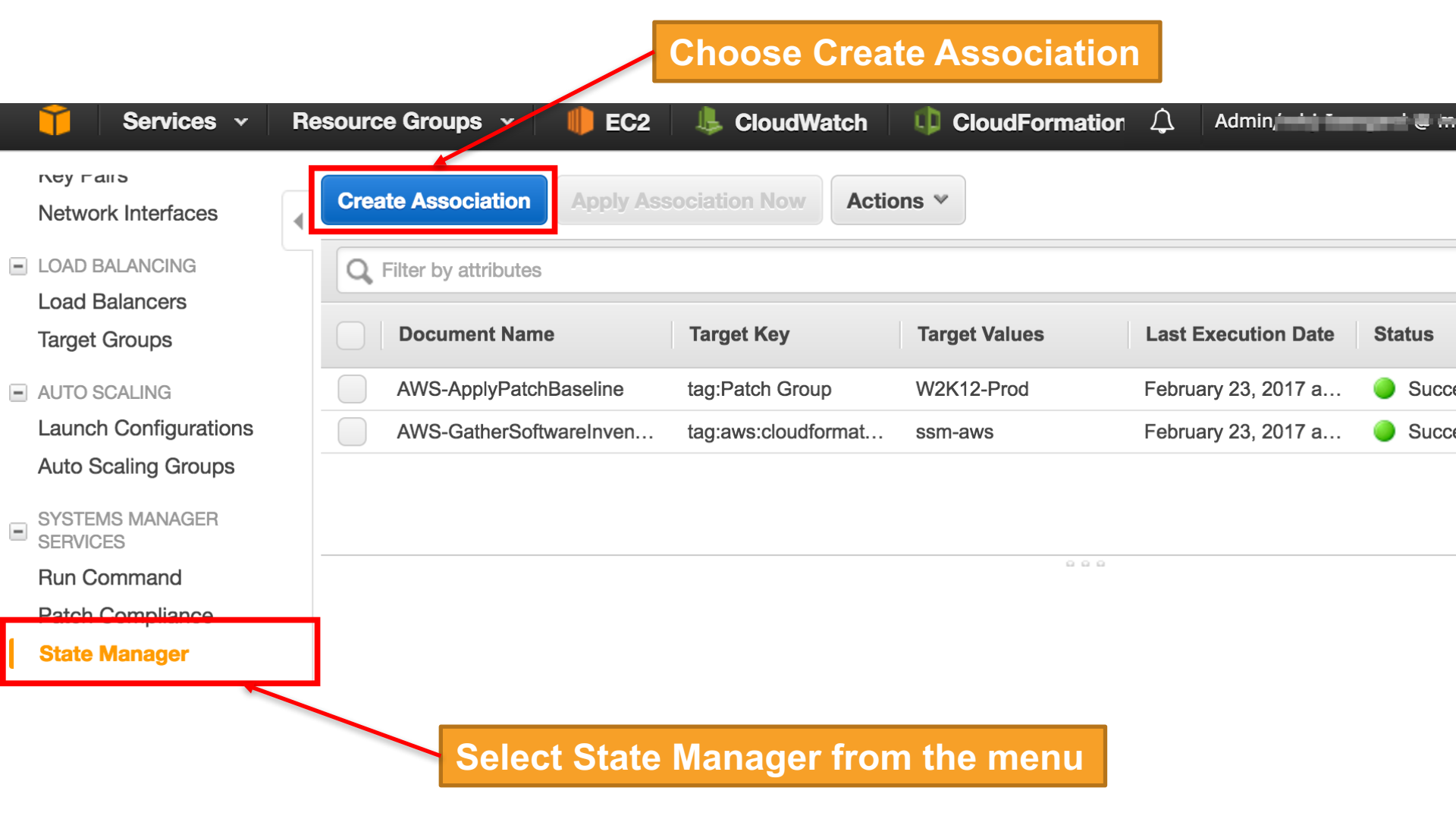
- Defined using a Command or Policy document

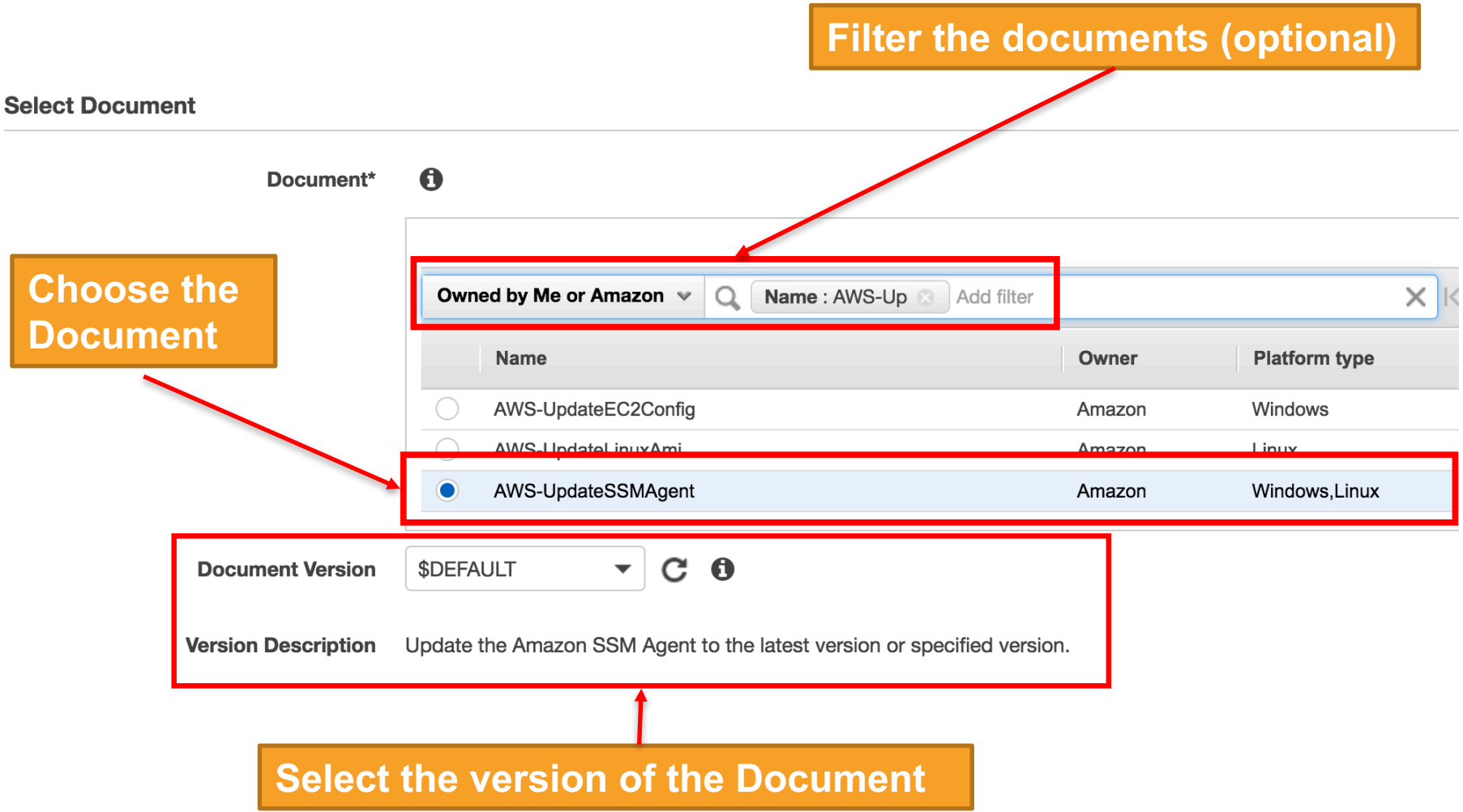
Determine the **managed instances** to be targeted:

- Define a target group based on Tags (Key or Key=Value), or
- Select individual instances manually

Determine the schedule to apply the actions:

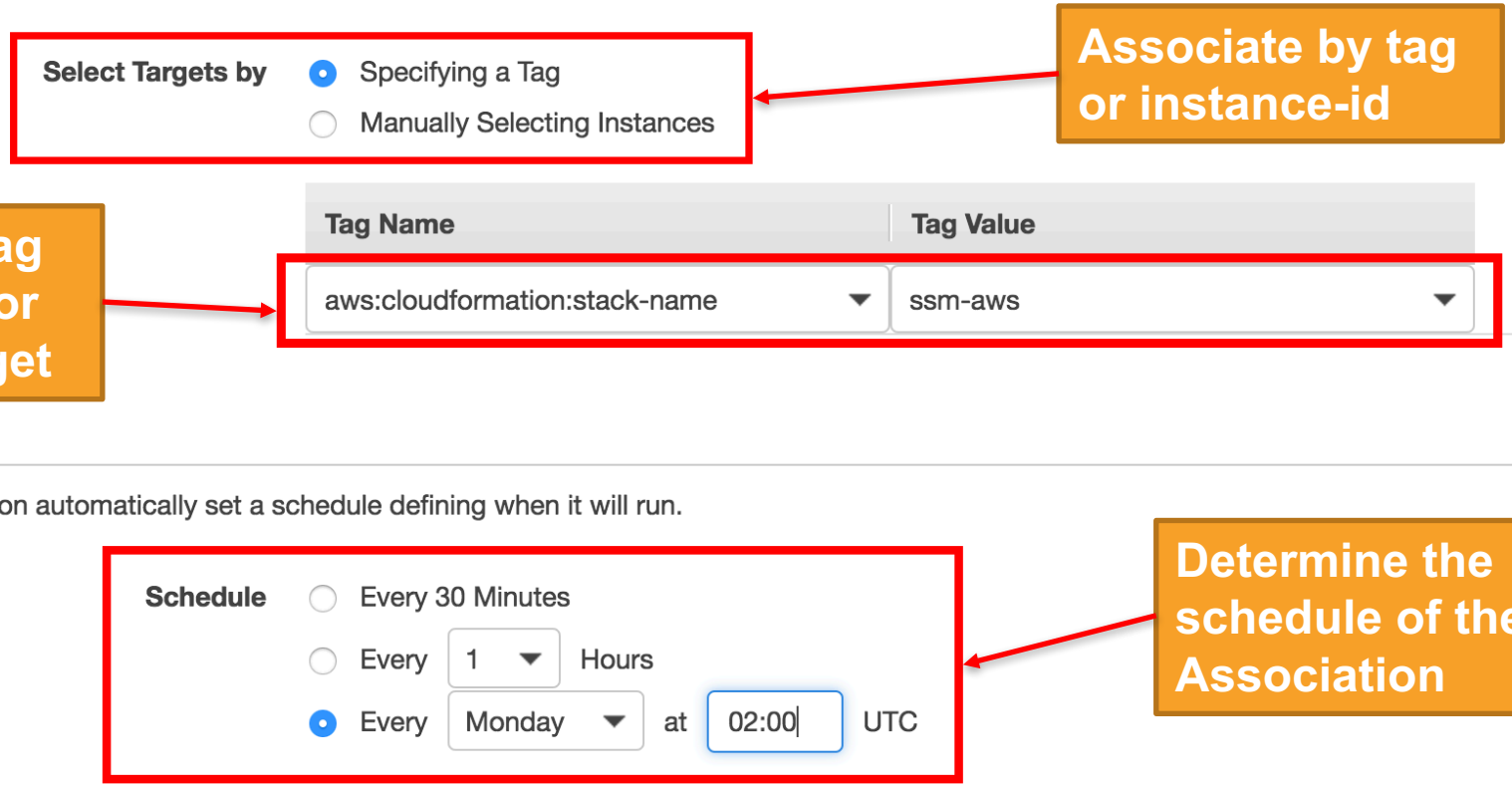
- Every 30 minutes
- Every x hours
- Every week on x day at hh:mm


Choose Create Association
Services
Resource Groups
EC2
CloudWatch
CloudFormation
Admin
Key Pairs
Network Interfaces
LOAD BALANCING
Load Balancers
Target Groups
AUTO SCALING
Launch Configurations
Auto Scaling Groups
SYSTEMS MANAGER SERVICES
Run Command
Patch Compliance
State Manager
Create Association
Apply Association Now
Actions
Filter by attributes
Document Name
Target Key
Target Values
Last Execution Date
Status
AWS-ApplyPatchBaseline
tag:Patch Group
W2K12-Prod
February 23, 2017 a…
AWS-GatherSoftwareInven…
tag:aws:cloudformat…
ssm-aws
February 23, 2017 a…
Select State Manager from the menu


Filter the documents (optional)
Select Document
Document*
Choose the Document
Owned by Me or Amazon
Name : AWS-Up
Add filter
Name
Owner
Platform type
AWS-UpdateEC2Config
Amazon
Windows
AWS-UpdateLinuxAmi
Amazon
Linux
AWS-UpdateSSMAgent
Amazon
Windows,Linux
Document Version
$DEFAULT
Version Description
Update the Amazon SSM Agent to the latest version or specified version.
Select the version of the Document

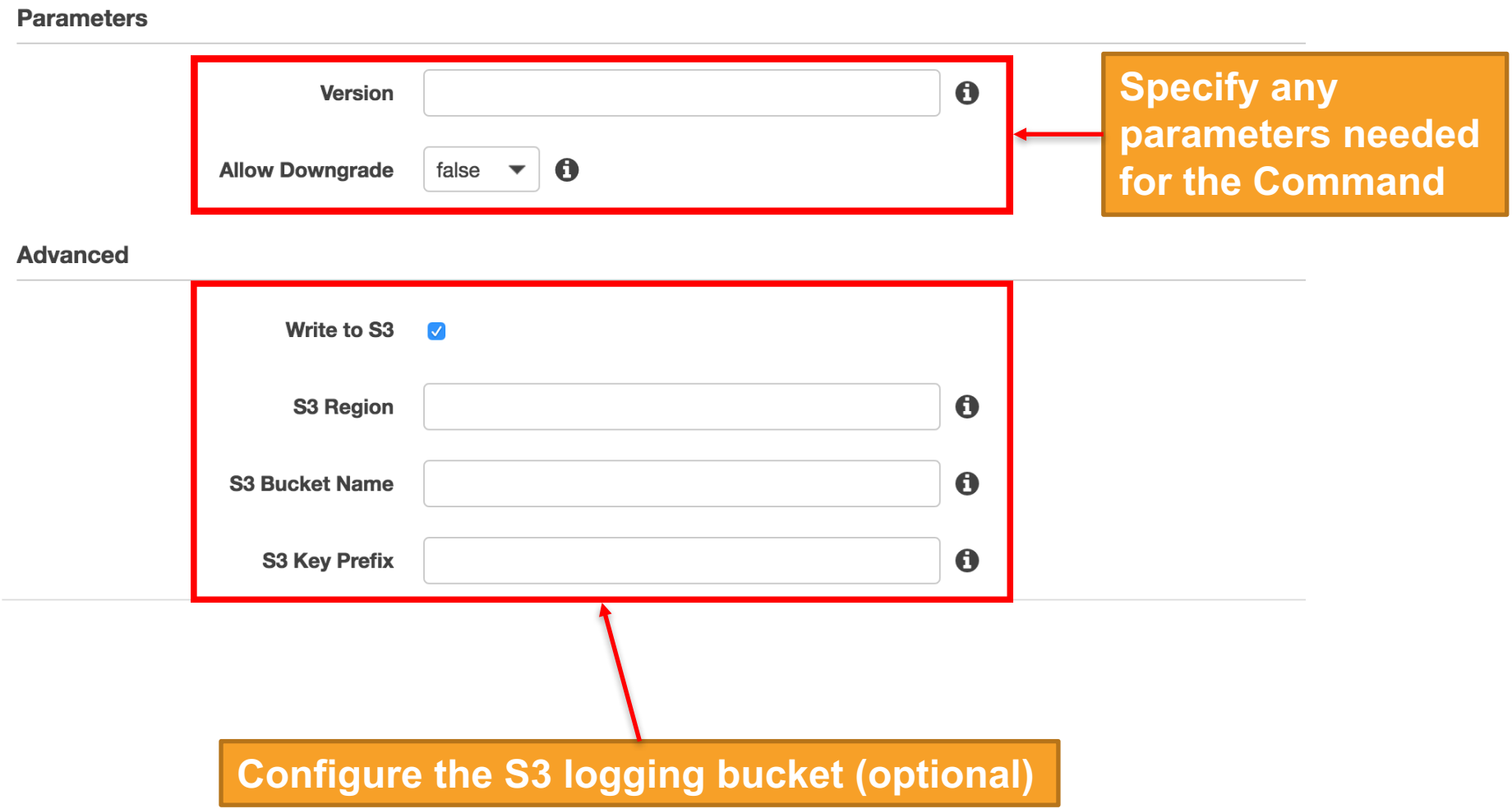
Targets

Targets are the instances you would like to associate with this document. You can choose to target by both managed instance and tag.



Schedule

To run an association automatically set a schedule defining when it will run.


Parameters
Version
Allow Downgrade
false
Specify any parameters needed for the Command
Advanced
Write to S3
S3 Region
S3 Bucket Name
S3 Key Prefix
Configure the S3 logging bucket (optional)

Create Association

Success

We're creating an association for the targets listed below.

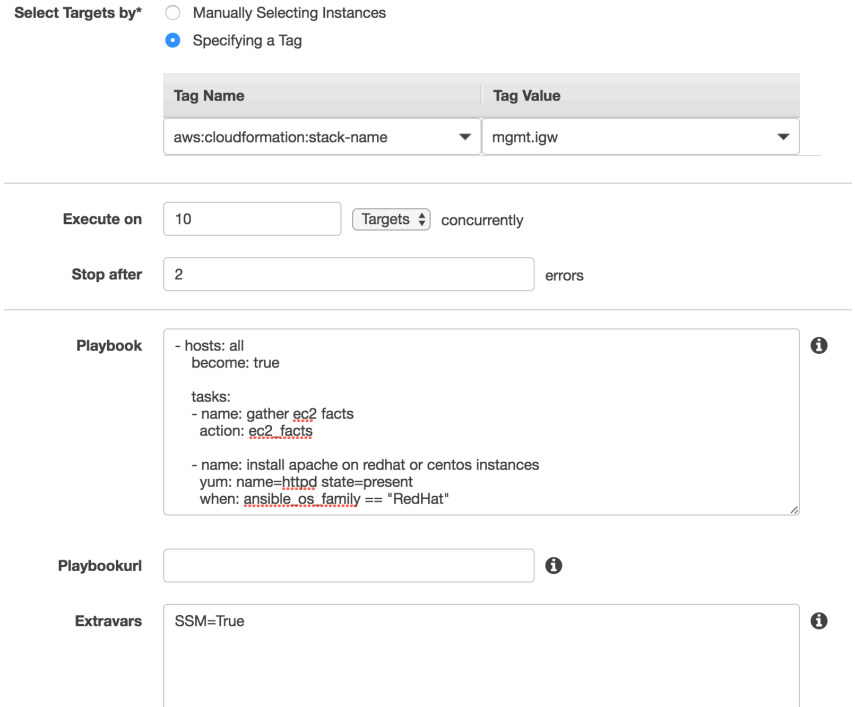
Association ID	fc4fd6e5-8552-4415-b0a9-1bbb7749c55c
Target Key	tag:aws:cloudformation:stack-name
Target Values	ssm-aws

Close

View the new Association

Example: Ansible & State Manager

- Execute configuration management directives using Ansible and EC2 Run Command / State Manager
- Makes use of the new “AWS-RunAnsiblePlaybook” public command document
- Track and audit usage using AWS CloudTrail



https://aws.amazon.com/blogs/mt/running-ansible-playbooks-using-ec2-systems-manager-run-command-and-state-manager/

Walkthrough: Inventory

Inventory (via State Manager)

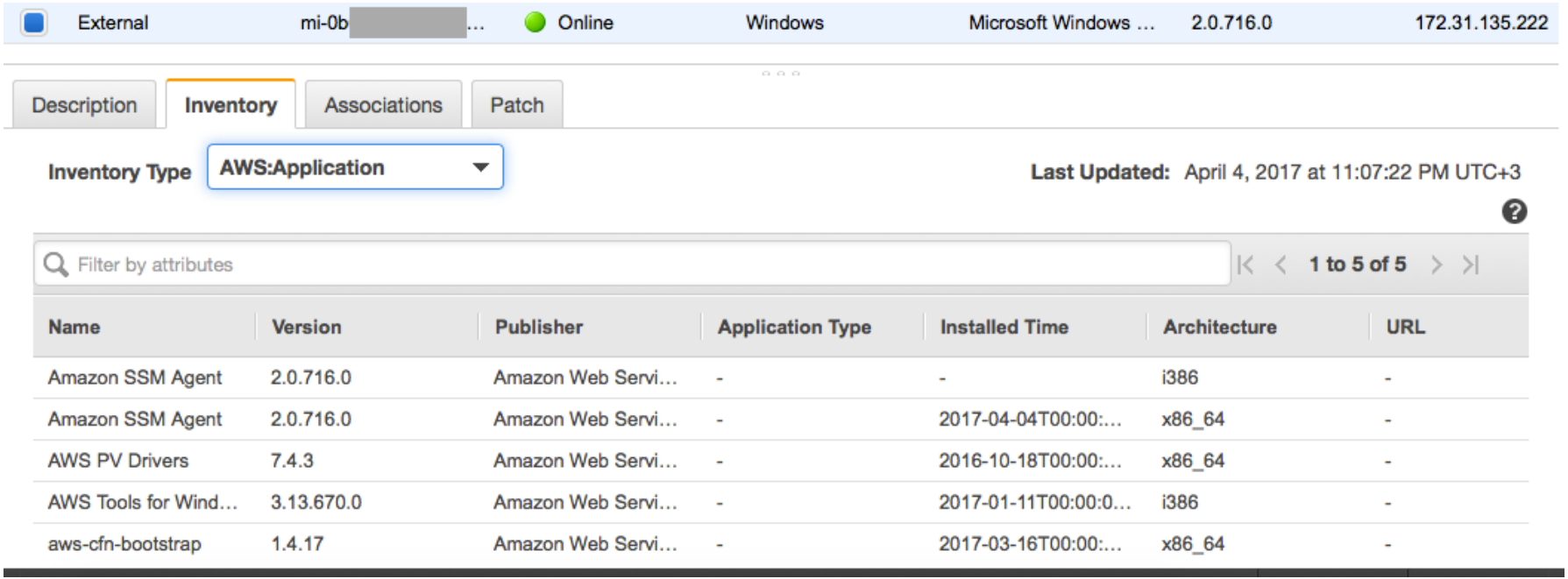
Uses State Manager to schedule inventory collection

- Policy Document: AWS-GatherSoftwareInventory

Determine the **types** of inventory to be collected:

- Installed Applications
- AWS Software Components
- Network Configuration
- Custom Inventory Information
- Windows Updates (Windows instances only)

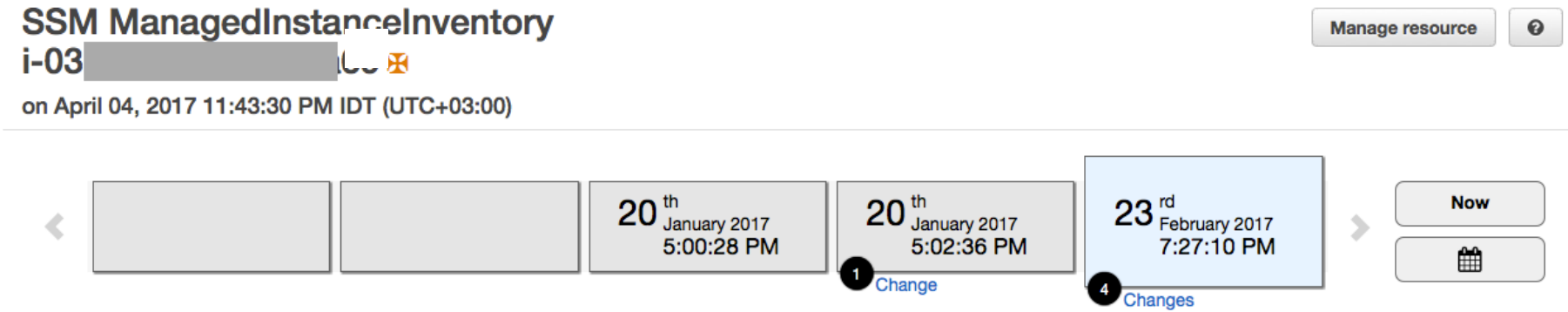
Viewing Inventory details: AWS Console



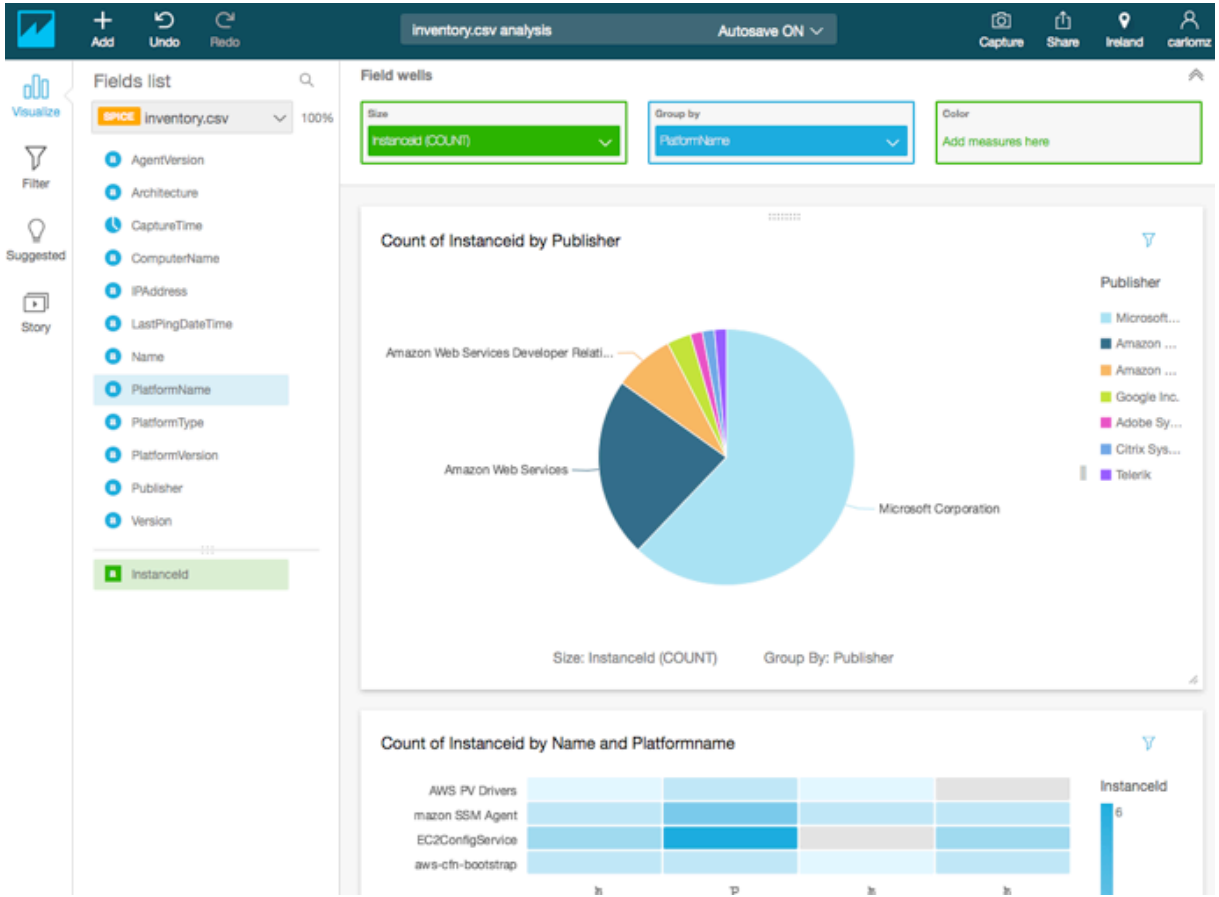
Viewing Inventory details: AWS CLI

```
aws ssm get-inventory –filters
     Key=AWS:InstanceInformation.PlatformType,
          Values=Windows,
          Type=Equal
     Key=AWS:InstanceInformation.ResourceType,
          Values=ManagedInstance,
          Type=Equal
```

Integration with AWS Config



Integration with Amazon QuickSight



Walkthrough: Documents

EC2 Systems Manager Documents

Documents define actions performed on managed instances

- **Command** documents: used to define and execute commands
- **Policy** documents: used to enforce a policy on targets
- **Automation** documents: perform common deployment tasks

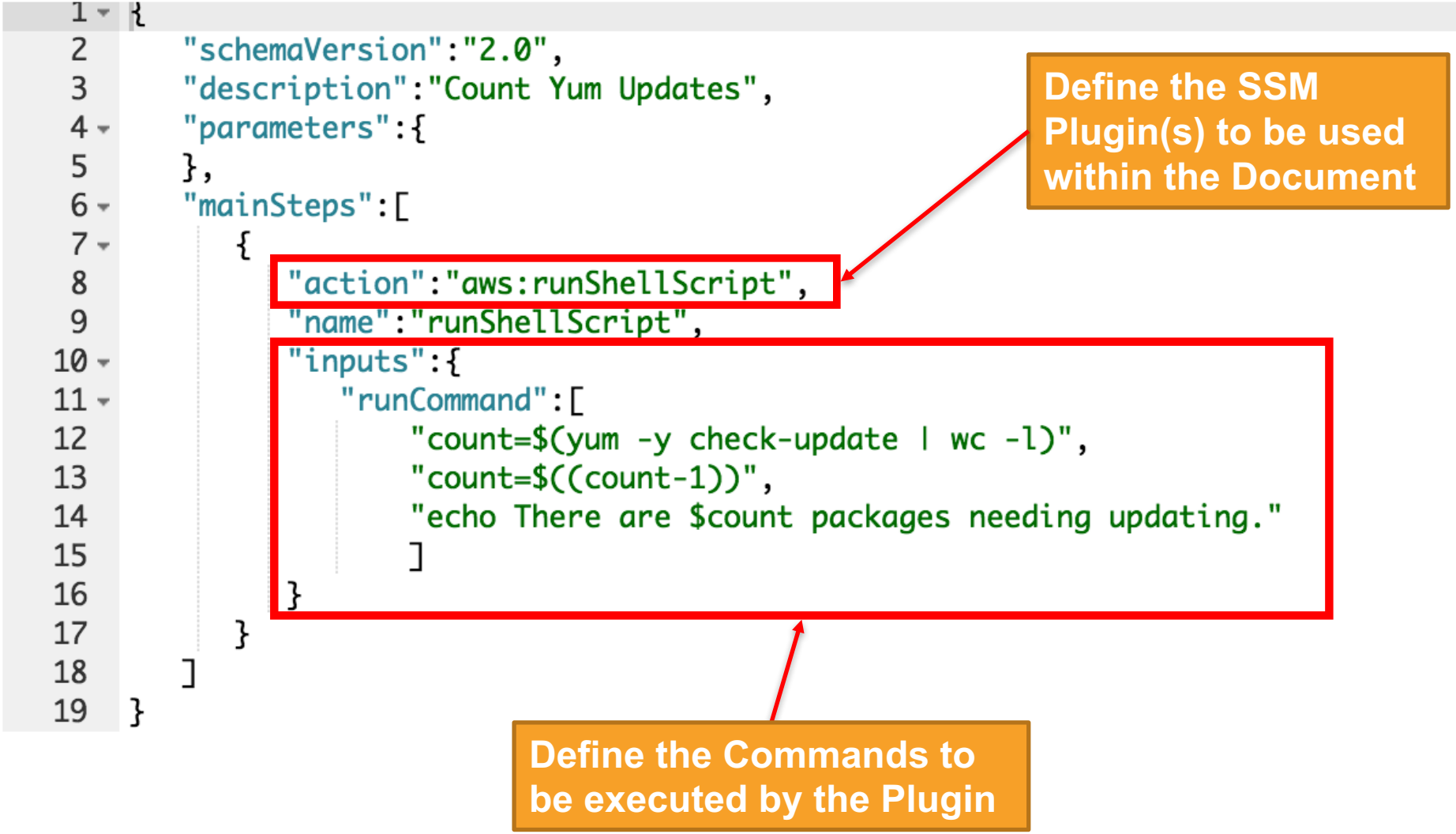
Documents:

- use JSON formatting
- support editing and versioning (using schema v2.0+)
- sequencing of steps
- AWS-managed, user created, or shared from other accounts

Document Structure

Basic structure of a Document:

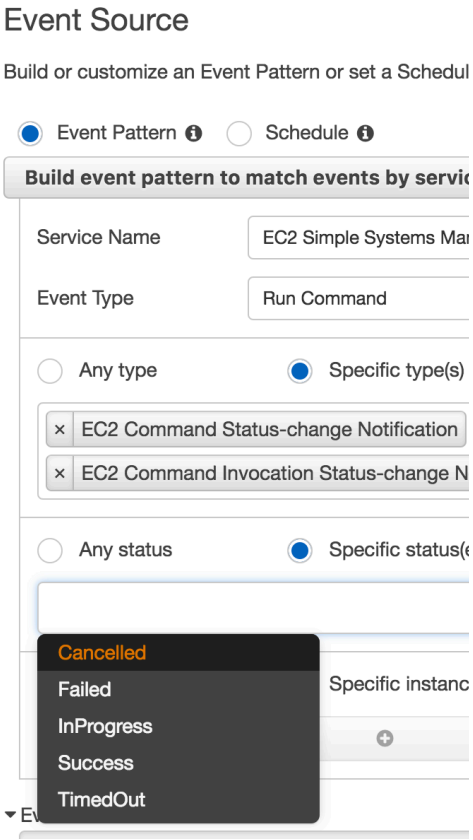
- **schemaVersion**: the schema version to use.
- **Description**: Information you provide to describe the purpose of the document.
- **Parameters**: The parameters the document accepts, for example *command* or *timeout*
- **mainSteps**: An object that can include multiple steps (plugins). Steps include one or more actions, a unique name of the action, and inputs (parameters) for those actions.

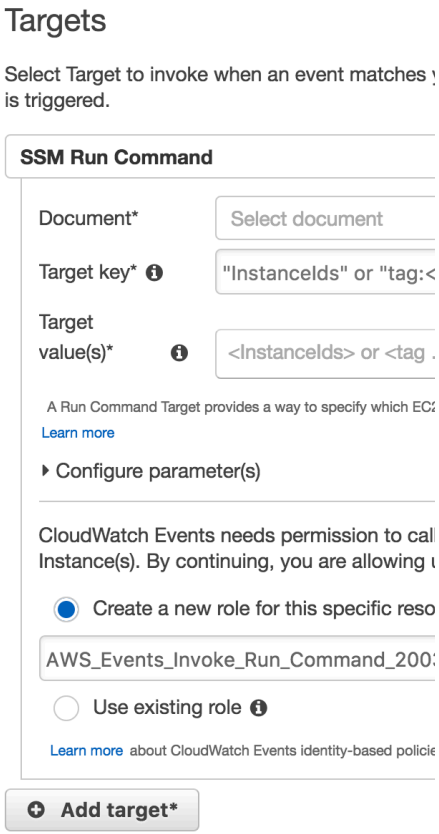

{
"schemaVersion":"2.0",
"description":"Count Yum Updates",
"parameters":{
},
"mainSteps":[
{
"action":"aws:runShellScript",
"name":"runShellScript",
"inputs":{
"runCommand":[
"count=$(yum -y check-update | wc -l)",
"count=$((count-1))",
"echo There are $count packages needing updating."
]
}
}
]
}
Define the SSM Plugin(s) to be used within the Document
Define the Commands to be executed by the Plugin

Walkthrough: Integration with other AWS Services

Integration with CloudWatch Events

- Event **Sources**
  - Event Types
  - Statuses
  - Resources
- Event **Targets**
  - Run Command Documents
  - Target Key / Values
  - Parameters
  - IAM role





Integration with Lambda

```python
from __future__ import print_function

import boto3
import json
import logging
import os

client = boto3.client('ssm')

def lambda_handler(event, context):
    commandId = event['detail']['command-id']
    documentName = event['detail']['document-name']
    pluginName = "aws:runShellScript"

    response = client.list_command_invocations(
        CommandId=commandId,
        Details=True
    )

    print("Executing " + documentName + " on the following instances:")
    invocations = response['CommandInvocations']
    for inv in invocations:
        outputs = inv['CommandPlugins']
        for o in outputs:
            print(inv['InstanceId'] + " " + o['Status'] + ": " + o['Output'])
```

Retrieve information from the CloudWatch Event

Query the Output status of each Invocation

Print the Output status into CloudWatch Logs

Step 1: Create rule

Create rules to invoke Targets based on Events happening in your AWS environment.

Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

Event Pattern Schedule

Build event pattern to match events by service

Service Name: EC2 Simple Systems Manager (SSM)

Event Type: Run Command

Any type Specific type(s)

× EC2 Command Status-change Notification

Any status Specific status(es)

× Success × Failed × TimedOut × Cancelled

Targets

Select Target to invoke when an event matches your Event Pattern or when schedule is triggered

Lambda function

Function*: ssm-EmailOutput

▸ Configure version/alias

▸ Configure input

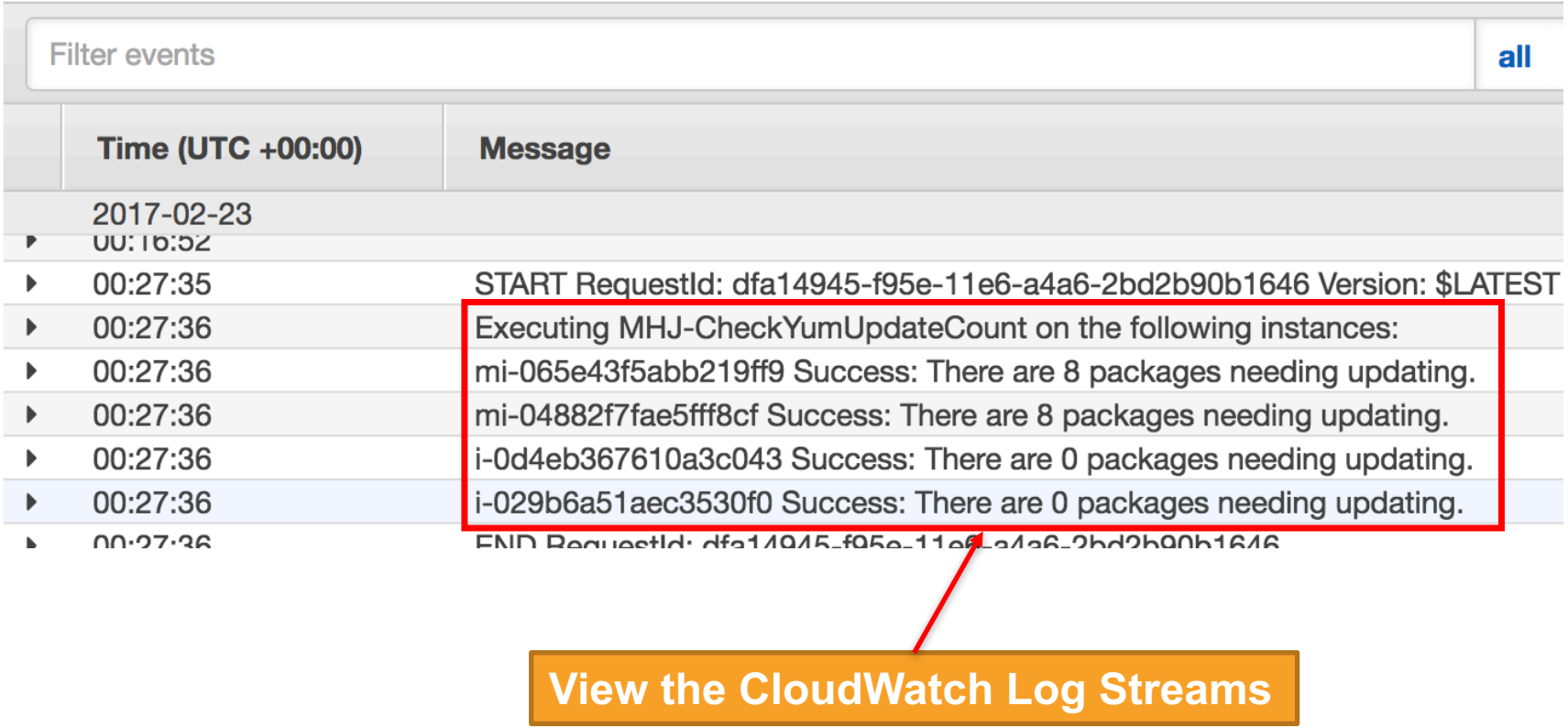
Add target*

Select EC2 SSM as the Event Source

Select the Lambda function as the target of the rule

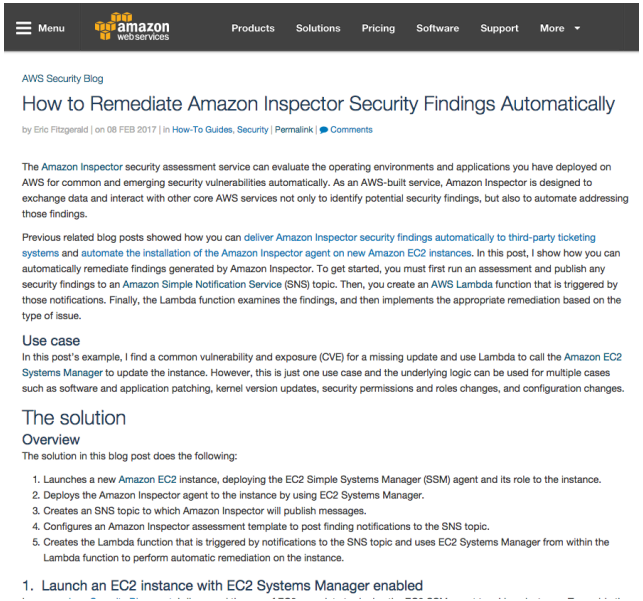
Specify the status(es) that trigger the rule

Viewing the output in CloudWatch Logs



Example: Remediate Amazon Inspector Findings

- Amazon Inspector sends SNS notifications of identified CVEs
- SNS triggers Lambda to call the Amazon EC2 Systems Manager to update the instance
- Broad application to multiple cases such as software and application patching, kernel version updates, security permissions, etc.



Menu | amazon web services | Products | Solutions | Pricing | Software | Support | More

AWS Security Blog

How to Remediate Amazon Inspector Security Findings Automatically

by Eric Fitzgerald | on 08 FEB 2017 | in How-To Guides, Security | Permalink | Comments

The Amazon Inspector security assessment service can evaluate the operating environments and applications you have deployed on AWS for common and emerging security vulnerabilities automatically. As an AWS-built service, Amazon Inspector is designed to exchange data and interact with other core AWS services not only to identify potential security findings, but also to automate addressing those findings.

Previous related blog posts showed how you can deliver Amazon Inspector security findings automatically to third-party ticketing systems and automate the installation of the Amazon Inspector agent on new Amazon EC2 instances. In this post, I show how you can automatically remediate findings generated by Amazon Inspector. To get started, you must first run an assessment and publish any security findings to an Amazon Simple Notification Service (SNS) topic. Then, you create an AWS Lambda function that is triggered by those notifications. Finally, the Lambda function examines the findings, and then implements the appropriate remediation based on the type of issue.

Use case

In this post's example, I find a common vulnerability and exposure (CVE) for a missing update and use Lambda to call the Amazon EC2 Systems Manager to update the instance. However, this is just one use case and the underlying logic can be used for multiple cases such as software and application patching, kernel version updates, security permissions and roles changes, and configuration changes.

The solution

Overview

The solution in this blog post does the following:

1. Launches a new Amazon EC2 instance, deploying the EC2 Simple Systems Manager (SSM) agent and its role to the instance.
2. Deploys the Amazon Inspector agent to the instance by using EC2 Systems Manager.
3. Creates an SNS topic to which Amazon Inspector will publish messages.
4. Configures an Amazon Inspector assessment template to post finding notifications to the SNS topic.
5. Creates the Lambda function that is triggered by notifications to the SNS topic and uses EC2 Systems Manager from within the Lambda function to perform automatic remediation on the instance.

1. Launch an EC2 instance with EC2 Systems Manager enabled

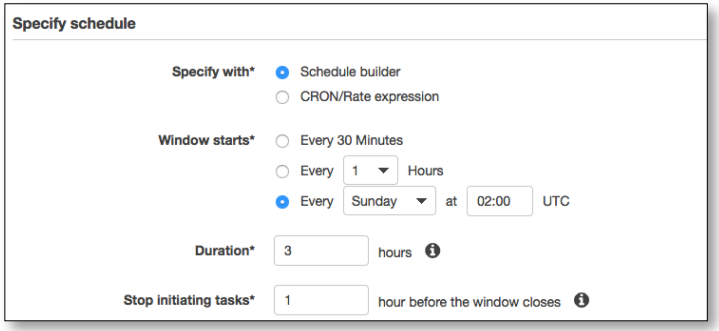
https://aws.amazon.com/blogs/security/how-to-remediate-amazon-inspector-security-findings-automatically/

Only scratched the surface…

Maintenance Window: Use Cases

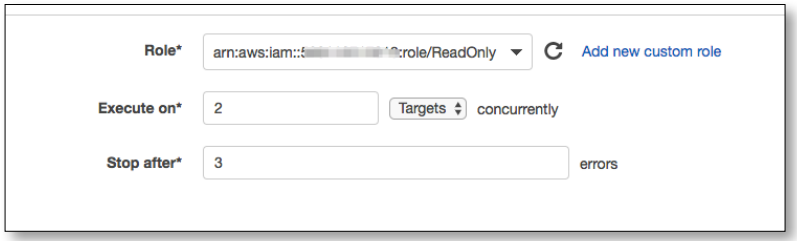
Automatically perform tasks in defined windows of time

- Define a maintenance window using cron or rate expressions
- Ensure maintenance doesn't overlap key business periods



Prioritise tasks and define roll-back and timeout criteria

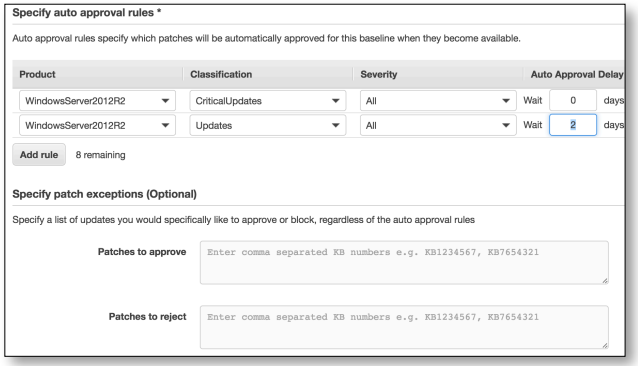
- Ensure key tasks are completed first during maintenance windows
- Execute tasks with specific IAM roles for granular security control



Patch Manager: Use Cases

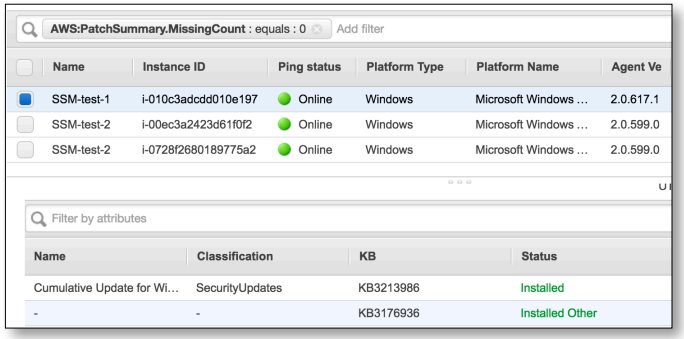
Manage Patch Baselines

- Define patch baselines by products, categories & severities
- Define approval and distribution schedule for specific baselines



Manage Patch Compliance

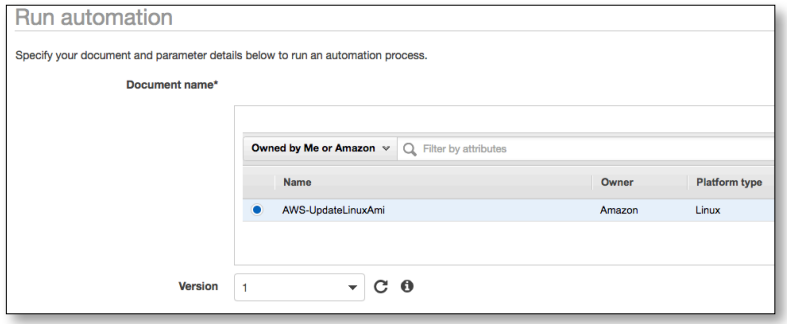
- Scan existing fleet to determine patch levels of the software
- Identify patches currently installed, missing, recently applied, etc.



Automation: Use Cases

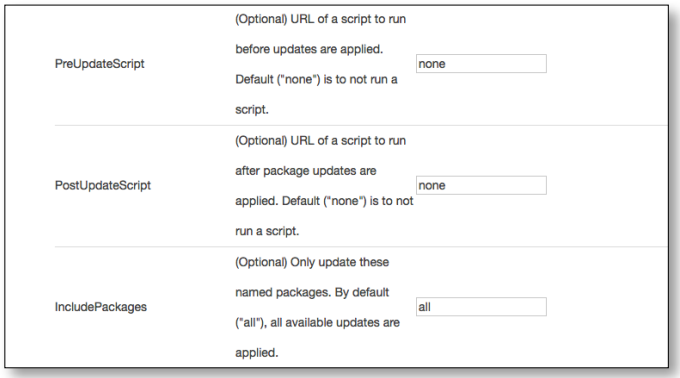
Maintain and Update your AMIs

- Integrates with CloudWatch for proactive notifications
- Use in conjunction with Maintenance Windows



Include Applications in your AMIs

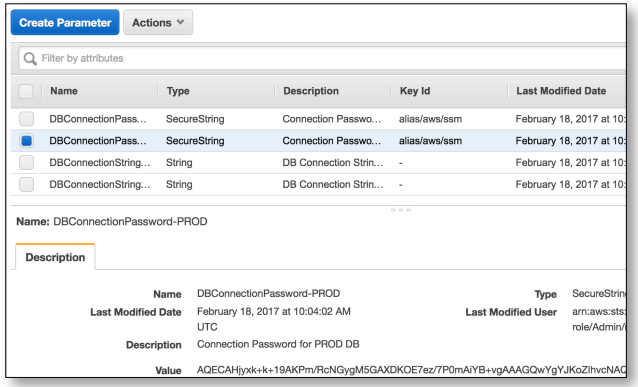
- Bake applications into an image
- Incorporate Automation as part of your change management process



Parameter Store: Use Cases

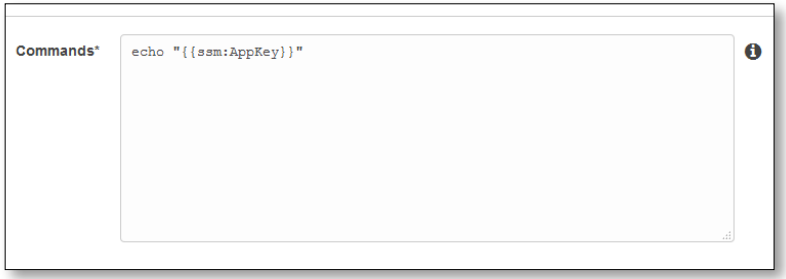
Easy deployment and configuration of applications

- Create env-specific parameters and reference in workflow
- Perform config-management at scale without plain-text passwords



Secure domain join

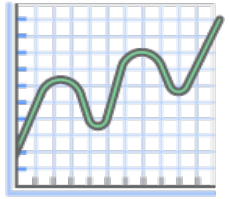
- Create secure string parameter with domain join password
- Control access to specific users and refer using simple syntax



In summary...

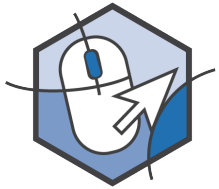
Hybrid

Cross-platform



Scalable

Secure



Easy-to-write automation

Reduced TCO

https://aws.amazon.com/blogs/mt/

AWS

SUMMIT

Thank you!

amazon web services