AWS Patching with System Manager Services (SSM)

Created by Nikhil Nishant
Last updated Apr 19, 2018

AWS Systems Manager automates the process of patching managed instances with security-related updates. For Linux-based instances, also install patches for non-security updates. SSM is used for both Linux and Windows instances.

Patch Manager automates the process of patching your managed instances. This feature enables you to scan instances for missing patches and apply missing patches individually or too large groups of instances by using Amazon EC2 instance tags./ select instances.

The equivalent yum command for this workflow is:
#sudo yum update-minimal --security --bugfix

How to patch instances in ACCA AWS Account.

We have 2 AWS account DEV-TEST and OAT-PROD AWS account and every application having 5 environments (INT, SYS, UAT, OAT & PROD). We should always follow patching sequence from INT → SYS → UAT → OAT → PROD.

We use SSM agent for patching the same set of packages version in the same type of instances. i.e int-rp-app-back1 sys-rp-app-back1 uat-rp-app-back1. Before patching always ensure that all application instance should have the same set of packages/packages version before and after patching.

Below is the sequence we will be following to patch instances.

Note: For maintaining the same level we will be using the snapshot-id feature of SSM. this snapshot-id is not that we take the snapshot of EBS(Volumes). When we patch our first instances we will generate snapshot-id in patching log.

Step 1: Take a snapshot of the all attached disk on which patching needs to done by SSM AWS service. This will be useful in case of we need to revert our changes..

Step 2: Install SSM Agent on the on all instances with below link.

http://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-install-ssm-agent.html#agent-install-rhel

For rhel 7 64 bit please use below command to install ssm agent on instance and start the service:-

#sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm && sudo systemctl status amazon-ssm-agent && sudo systemctl enable amazon-ssm-agent && sudo systemctl start amazon-ssm-agent && sudo systemctl status amazon-ssm-agent

Step 3: For allowing access to SSM to perform patching on instances need to add below policy with Role.

Below is the example of reporting engine instances patching that we have successfully patched/tested.

Add the Policy to allow SSM actions. Policy needs to be added in IAM.

```
{
```

```
    "Version":"2012-10-17",
    "Statement":[
       {
          "Action":"ssm:*",
          "Effect":"Allow",
          "Resource":"*"
       },
       {
          "Action":"ec2:DescribeInstanceStatus",
          "Effect":"Allow",
          "Resource":"*"
       },
       {
          "Action":"ec2messages:*",
          "Effect":"Allow",
          "Resource":"*"
       }
    ]
}
```

Step 4: Attach the above policy to IAM WebServer IAM role(int-**rp-web-front1**).

Step 5: Attach the above policy to JbossServer IAM role(int-**rp-app-back1**).

Step 6: Trust relationships for AnsbileAdmin,WebServer and JbossServer IAM role.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ssm.amazonaws.com",
               "ec2.amazonaws.com"
            ]
```

```
            },
            "Action":"sts:AssumeRole"
        }
    ]
}
```

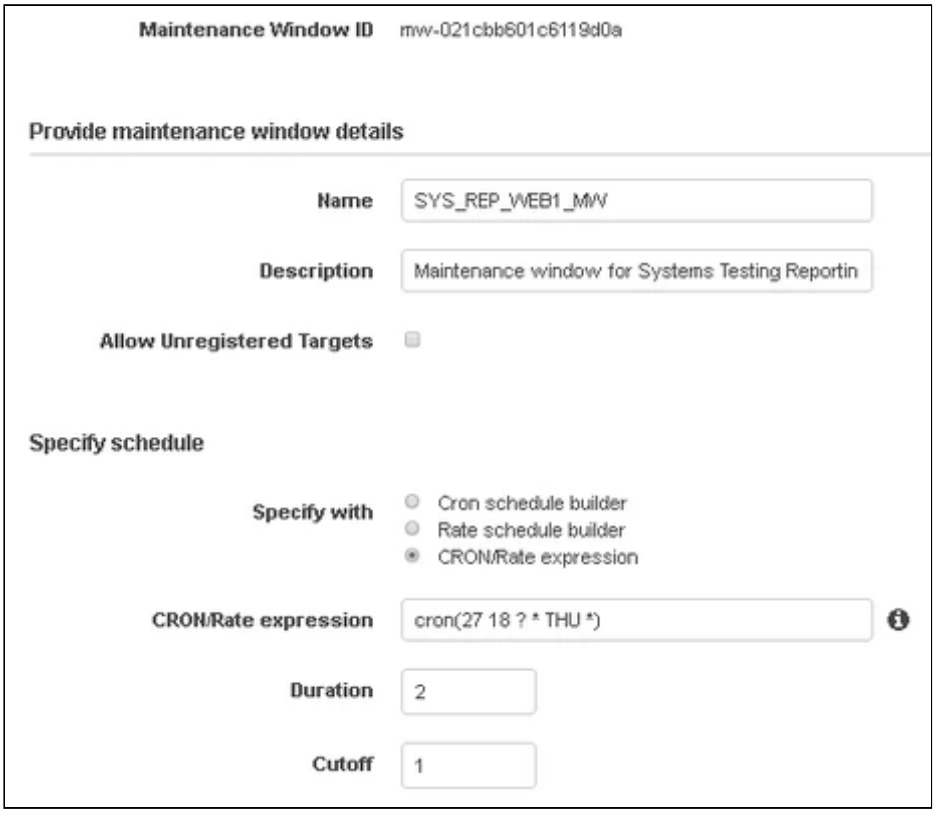
AWS Systems Manager Maintenance Windows(Maintenance window for Instance):-

AWS Systems Manager Maintenance Windows let you define a schedule for when to perform potentially disruptive actions on your instances such as patching an operating system (OS), updating drivers, or installing software. Each Maintenance Window has a schedule, a duration, a set of registered targets, and a set of registered tasks. With Maintenance Windows, you can perform tasks like the following:

- **Create a maintenance window in AWS Console for sys-rp-web-front1.**

Step 1:EC2->Systems Manager Shared Resources->Maintenance Windows->Create Maintenance window.

> **Name**: SYS_REP_WEB1_MW
> **Description**: Maintenance window for Integration re[prting Web Server.
> **Specify schedule**: cron(0 15 ? * TUE *)
> **Duration**: 2 hours
> **Stop initiating tasks**: 1 hour



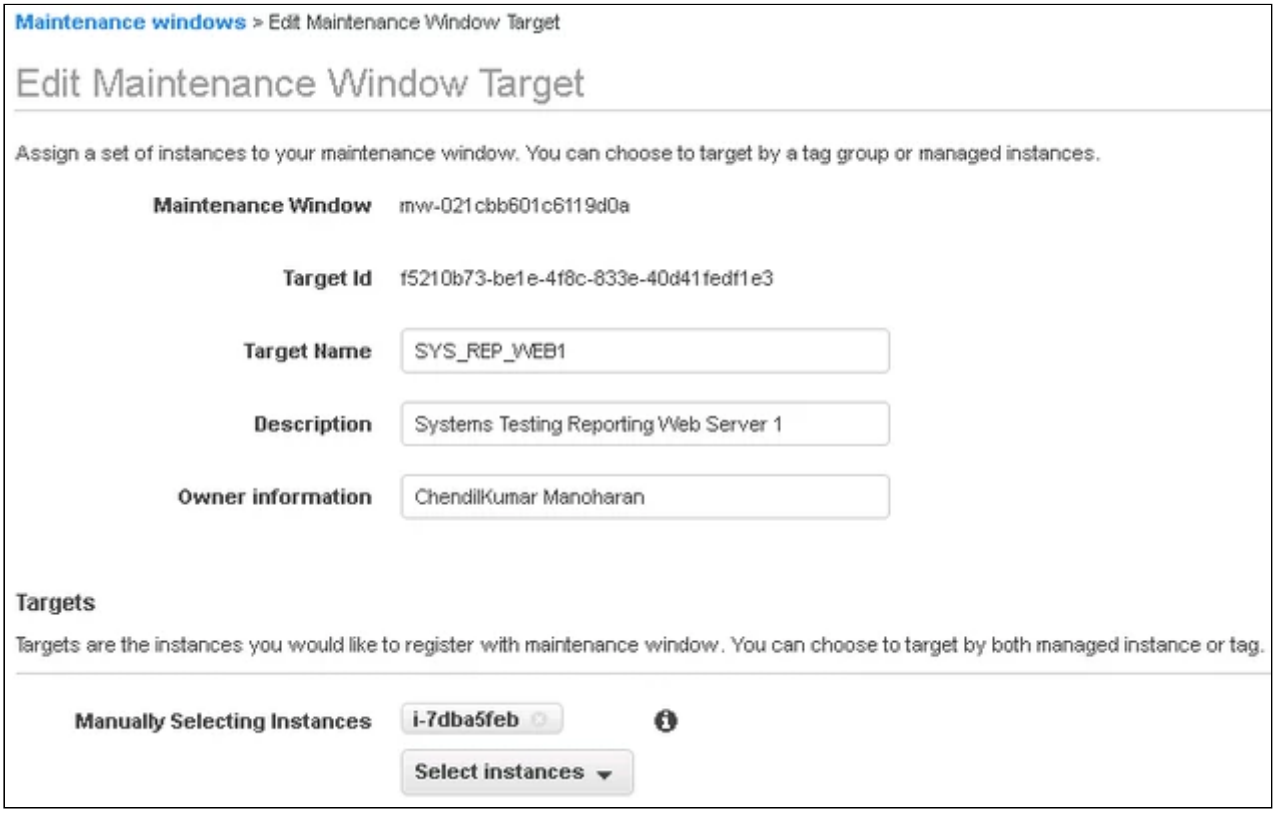
Step 2: Select the create maintenance window. (SYS_REP_WEB1_MW)

- Select the maintenance window, click on the Targets Tab and click Register new target.

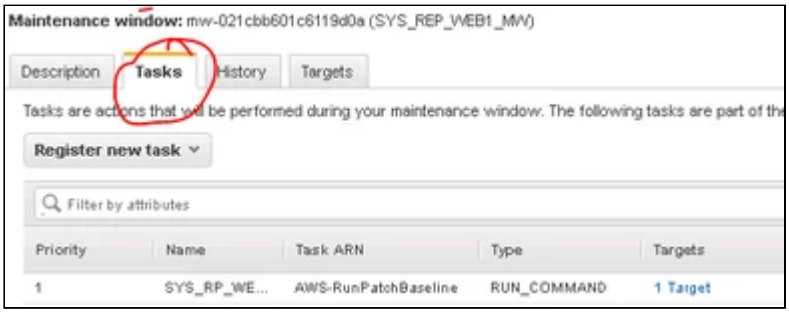
- Fill the following information according to application.

```
Target name: SYS_REP_WEB1
Description: Systems Testing Reporting Web Server 1
Owner Information: Nikhil.

Select Targets->Manually Selecting Instances->Select the int-web-front1 server
```



Step 4: Cick Register targets



- Click on the Tasks Tab and click Register new task->register run command task

> Name: INT_WEB_INSTALL_OS_Patch.
>
> Description: Apply latest OS patches to Intergration Testing Reporting Web server.
>
> Document: AWS-RunPatchBaseline.
>
> Task Priority: 1
>
> Targets->Strict targets: Select SYS_REP_WEB1

Role: arn:aws:iam::223715066669:role/AnsibleAdmin

Executes on: 100 Percent

Stops after: 1 errors

Operation: Install

Snapshot Id : (will leave this filled balnk for patching)

Comment: Task to install the latest OS patches on Systems Testing Reporting Web Server.

Execution Timeout: 3600 seconds

Timeout: 600

Advanced Options: Check Write to S3

S3 bucket: acca-aws-infra-engineering

S3 key prefix: systems-manager-2018

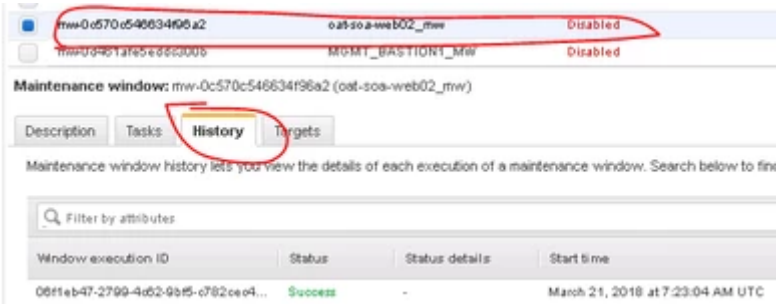
- Click Register task..

Step 4: Select the maintenance window->Actions→Edit Maintenance window.

Change cron time when you want patching to start. ex. cron(0 15 ? * TUE *) → It will run at 3PM on Tuesday.

When patching will be started we can check status on aws console as well as in server.

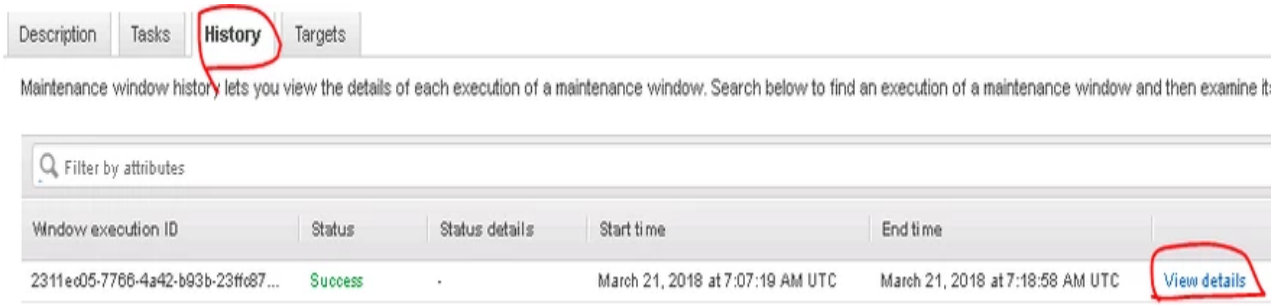
To check on aws console patching status:- Select maintenance window > click on the history tab.



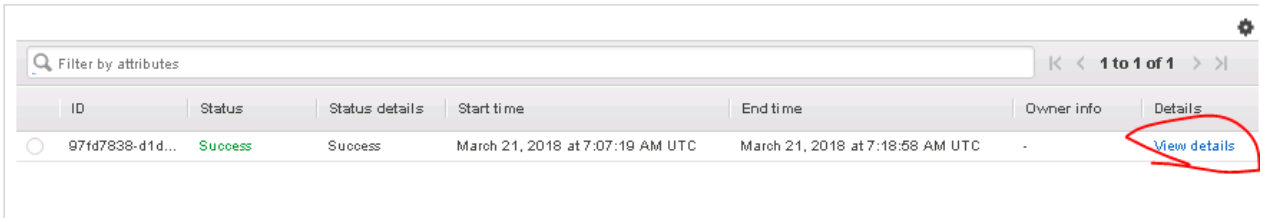
- To check on instances: login to server. i.e: ssh #sys-rp-web01 > #sudo cd /var/log > #sudo tail -f yum.log messages.

When patching get completed we need to take snapshot-id from first patching.

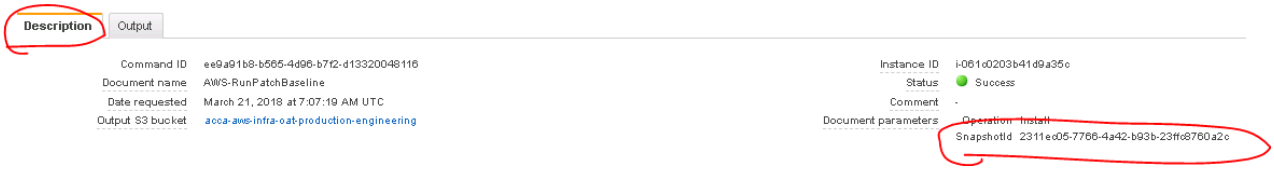
- Select the maintenance window → Click on tab History → Click on : view details → new window open: View execution history → click on : view details. → new windows open: run command → Click on tab :Description → copy snapshot-id.
- Below is the screenshot how to get snapshot-id:



- View execution history



- new windows open: run command

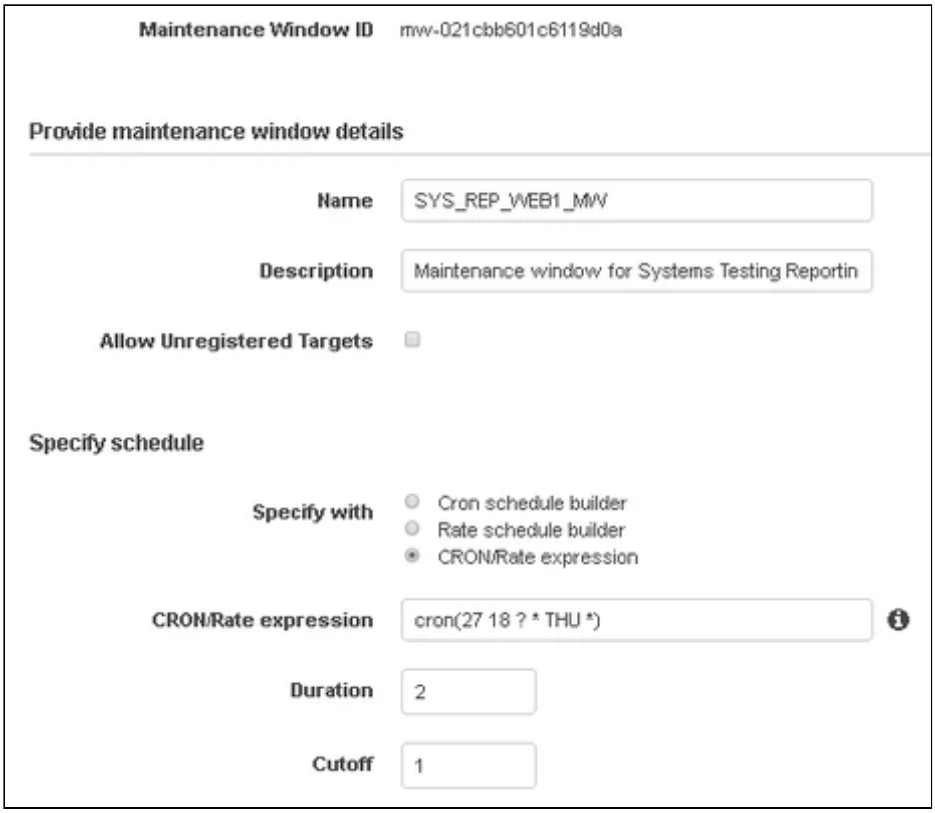


Note: snapshot-id will only use in the case when we need to patch same environment & application. if we need to patch int-soa-web01, int-soa-web02 , dev-soa-web01, dev-soa-web02, sys-soa-web01, sys-soa-web02... so in that case we will patch int-soa-web01 with Document: AWS-RunPatchBaseline (select task tab) with our snapshot..after that we will wait till patching get completed. Then we will copy snapshot-id(Step 4) and use snapshot-id for patching reset instances.

Patching Instance with Snapshotid:-

Step 1:EC2->Systems Manager Shared Resources->Maintenance Windows->Create Maintenance window.

> **Name**: SYS_REP_WEB1_MW
> **Description**: Maintenance window for Integration re[prting Web Server.
> **Specify schedule**: cron(0 15 ? * TUE *)
> **Duration**: 2 hours
> **Stop initiating tasks**: 1 hour



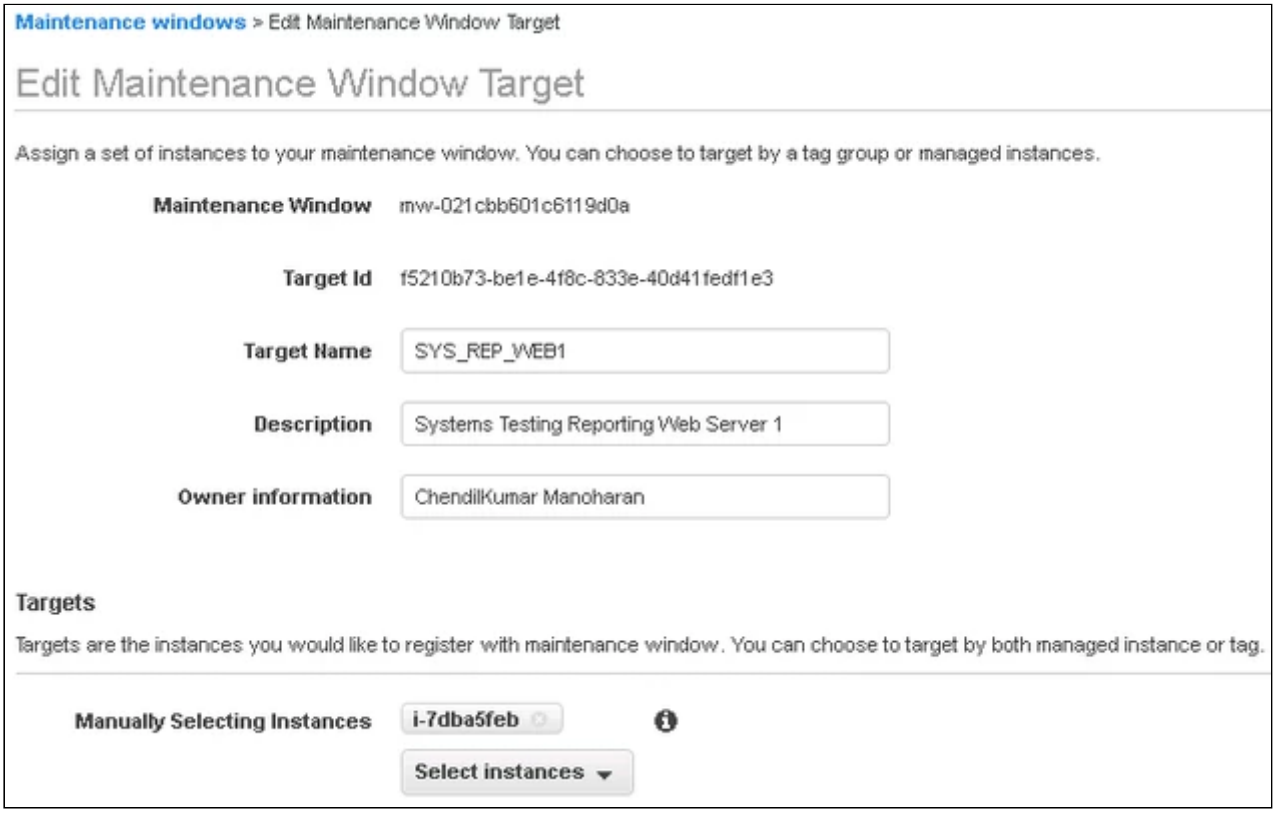
Step 2: Select the create maintenance window. (SYS_REP_WEB1_MW)

- Select the maintenance window, click on the Targets Tab and click Register new target.

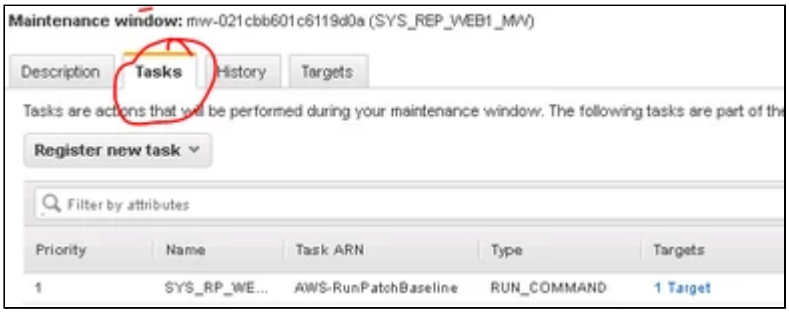
- Fill the following information according to application.

```
Target name: SYS_REP_WEB1
Description: Systems Testing Reporting Web Server 1
Owner Information: Nikhil.

Select Targets->Manually Selecting Instances->Select the int-web-front1 server
```



Step 4: Cick Register targets



- Click on the Tasks Tab and click Register new task->register run command task

> Name: INT_WEB_INSTALL_OS_Patch.
>
> Description: Apply latest OS patches to Intergration Testing Reporting Web server.
>
> Document: AWS-RunPatchBaseline.
>
> Task Priority: 1
>
> Targets->Strict targets: Select SYS_REP_WEB1

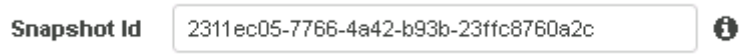
Role: arn:aws:iam::223715066669:role/AnsibleAdmin

Executes on: 100 Percent

Stops after: 1 errors

Operation: Install

Snapshot ID: (Will use snapshot of 1st patching)



Comment: Task to install the latest OS patches on Systems Testing Reporting Web Server.

Execution Timeout: 3600 seconds

Timeout: 600

Advanced Options: Check Write to S3

S3 bucket: acca-aws-infra-engineering

S3 key prefix: systems-manager-2018

- Click Register task..

Step 4:Select the maintenance window->Actions→Edit Maintenance window.

Change cron time when you want patching to be start. ex. cron(0 15 ? * TUE *) → It will run at 3PM on Tuesday.

To Check/list packages version between instances:

To list packages version and packages between the same type of instance for this we will use ansible playbook for it. For example

1. List out packages list of instances.

ansible-playbook list_packages.yml -e "host=rp_web" -u ec2-user > /tmp/rp_web-output-1.log

With the help of above command we are trying consolidate package list of below servers:

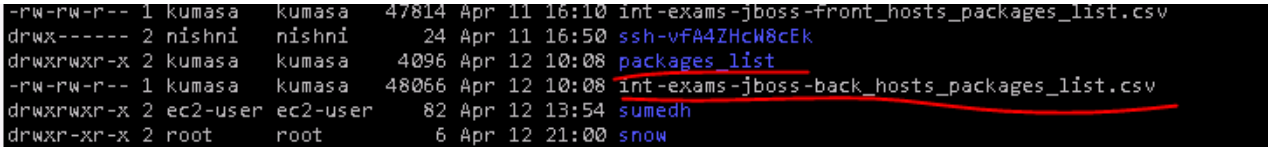
```
nishni 16:51:36 dev ansible01 develop:$ansible rp_web -a "hostname"
172.20.16.166 | SUCCESS | rc=0 >>
sys-rp-web-front1.dev.eu-west-1.acca-awscloud.net

172.20.7.147 | SUCCESS | rc=0 >>
int-rp-web-front1.dev.eu-west-1.acca-awscloud.net

172.20.9.204 | SUCCESS | rc=0 >>
uat-rp-web-front1.dev.eu-west-1.acca-awscloud.net
```

2. After running successfully above playbook we will get output in /tmp directory.
- we have used int exams jboss back server to list packages of int-exams-jboss-dc-back01,int-exams-jboss-eap-back01& int-exams-jboss-eap-back02 for example. So we have .csv file in /tmp

example: int-exams-jboss-back_hosts_packages_list.csv



2. After getting above .csv file, then we will compare package difference between them. For comparing packages we will be using python script for it.

Below is the python script:

```
#!/usr/bin/env python
import sys, csv
FILE = sys.argv[1]
print FILE

web1 = []
web2 = []

with open(FILE,'rb') as file:
for line in file:
row = line.rstrip('\r\n').split(',')
web1.append(row[1])
web2.append(row[2])


#print web1
#print web2

#print set(web1) - set(web2)

s = set(web2)
difference = [x for x in web1 if x not in s]

print difference
```

Like Be the first to like this

No labels