**Patching Server security packages Using Ansible**

In your company, there is a need to patch all your servers every month to update the security packages. This process is tedious doing it manually. As the DevOps of your Company, you received a ticket to automate this task. With your knowledge of **Ansible**, you decided to write a **playbook** to do this task. You will use a **Cron job** to schedule the task monthly.

Here we are going to use just 3 servers to demonstrate. One of the servers will be used as the master node and the other two as the nodes.

So let's go ahead and write our playbook.

## **What will our playbook do?**

**Connect to the Remote host and execute the following tasks**

* Upgrade security package on Amazon Linux servers
* Upgrade security package on Ubuntu/Debian servers

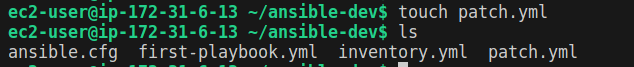


## **Writing the playbook**

Connect remotely to the master

Create a file **patch.yml** inside **ansible-dev** folder and paste the code below inside.

touch patch.yml



| --- - name: Security patches  hosts: databaseservers  become: yes   tasks:  - name: Update package cache and upgrade security packages on Ubuntu  when: "'Ubuntu' in ansible\_facts.os\_family"  apt:  name: '\*'  state: latest  update\_cache: yes  only\_upgrade: yes   - name: Update package cache and upgrade security packages on Amazon Linux  when: "'Amazon' in ansible\_facts.os\_family"  yum:  name: '\*'  state: latest  security: yes  update\_cache: yes   - name: Update package cache and upgrade security packages on Debian  when: "'Debian' in ansible\_facts.os\_family"  apt:  name: '\*'  state: latest  update\_cache: yes  only\_upgrade: yes |
| --- |

save and exit. Then do the syntax check.

Before we run the playbook, let's see in detail the tasks that the playbook will perform.

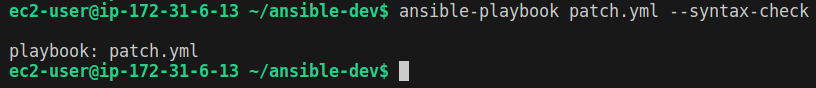
## **Let us now understand all the tasks one by one.**

* Update package cache and upgrade security packages on Ubuntu: This task updates the package cache and security-related packages for Ubuntu systems. The when clause ensures that this task will only run when the ansible\_facts.os\_family variable indicates that the host is running Ubuntu. The state: latest option ensures that all packages are at the latest version. The only\_upgrade: yes clause specifies that only existing packages will upgrade (no new installations).
* Update package cache and upgrade security packages on Amazon Linux: This task updates the package cache and security-related packages for Amazon Linux systems such as CentOS 7. The when clause ensures that this task will only run when the ansible\_facts.os\_family variable indicates that the host is running CentOS 7. The yum: module is used to update the package cache and applies security updates.
* Update package cache and upgrade security packages on Debian: This task upgrades any package cache or security-related packages on Debian the system. The apt module is used to perform the upgrade, and the update\_cache: yes argument refreshes the package cache before upgrading. The name: "\*" argument specifies that all packages should be upgraded, and the state: latest argument ensures that packages are upgraded to the latest available version. As with the previous task, the when clause is used to limit this task to only running on Debian systems.

## **Running the playbook**

let’s validate the syntax of our playbook to avoid errors when running it.

| ansible-playbook patch.yml --syntax-check |
| --- |

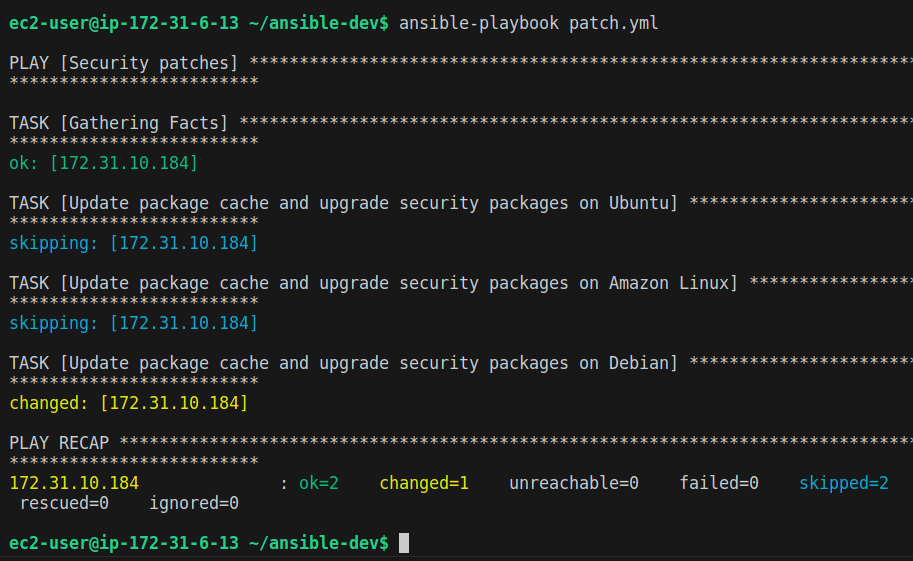


After validating the syntax, let's go ahead and run our playbook.

To run the playbook, run the code below:

| ansible-playbook patch.yml |
| --- |

When the playbook is done running, you will have the following recap.



The next thing we will do is to automate this task to run monthly.

## **Automating The Task to run monthly.**

Running the playbook monthly can be done in different ways. we can use a Cron job or we use Jenkins to do this.

**NOTE:** Using a cron Job is one of the many ways of doing this. You can also schedule this job using Jenkins.

### **Using a cron Job:**

Let's configure a cron job to run our playbook on the first day of every month at midnight.

verify if you have any running jobs for this particular user

| /bin/crontab -l |
| --- |

Then configure our cron job so that our playbook can run on the first of every month at midnight

| /bin/crontab -e |
| --- |

copy the code below and paste it into.

| 0 0 1 \* \* /usr/local/bin/ansible-playbook ~/ansible-dev/patch.yml |
| --- |

## 

## **Conclusion.**

This is just a very simple playbook example showing how you can automate very complex tasks with just a few commands and use a cronjob.

**NB: Another approach will be to schedule a periodic job build in Jenkins and set the job to login to Ansible and run the task.**

You can further practice by writing a playbook to update all your packages and scheduling the task.

Please read more about Ansible in the Ansible documentation.

<https://docs.ansible.com/ansible/latest/cli/ansible-playbook.html>