# OS Patching for a UAT/Test Server.

**Note: In this document you will see both theoretical knowledge and hands on also. So stay tuned.**

Theory: The criticality of OS patching  in IT security management cannot be overstated, especially now when the increase in BYOD(Bring your own device) has seen more devices - operating systems connecting to company networks, increasing your attack surface.

## =================================

## Why is OS patching important?

Done well, OS patching can be the difference between a well-supported environment and one that is susceptible to unplanned downtime and performance issues. Here are some of the critical benefits of a robust approach to OS patching:

* **Compliance**: Many organizations now have regulatory requirements or insurance directives mandating a regular patching regime. Non-compliance can lead to severe penalties.
* **Availability**: The sad truth is that as an IT professional, you are only as good as your last issue. Keeping your systems' patches will prevent extended downtime due to security threats and remedial maintenance/emergency patch activity.
* **Performance**: Devices can crash due to software defects, so keeping your services patched means they are updated with the latest bug fixes and are more secure.
* **Security**: A common cause of network security breaches is missing patches in operating systems. Having a regular patch schedule means installing updates promptly, reducing the opportunity for data loss and damage to your infrastructure.
* **New features**: Patches are not always about protection from malware or fixing bugs. Sometimes patches can include new features that can give users greater functionality.

## =================================

## Dos and Don’ts of OS patching

* **Don't use unsupported** or EOL (end-of-life software)..
* **Don’t install patches from** ad content.
* Do **communicate patch windows** beforehand and agree to any potential downtime with the rest of the business.
* Do **scan your environment** post patching.
* Do **understand each vendor's release schedule** for patches and updates so that you can plan and schedule maintenance work accordingly.

## =================================

Lab:

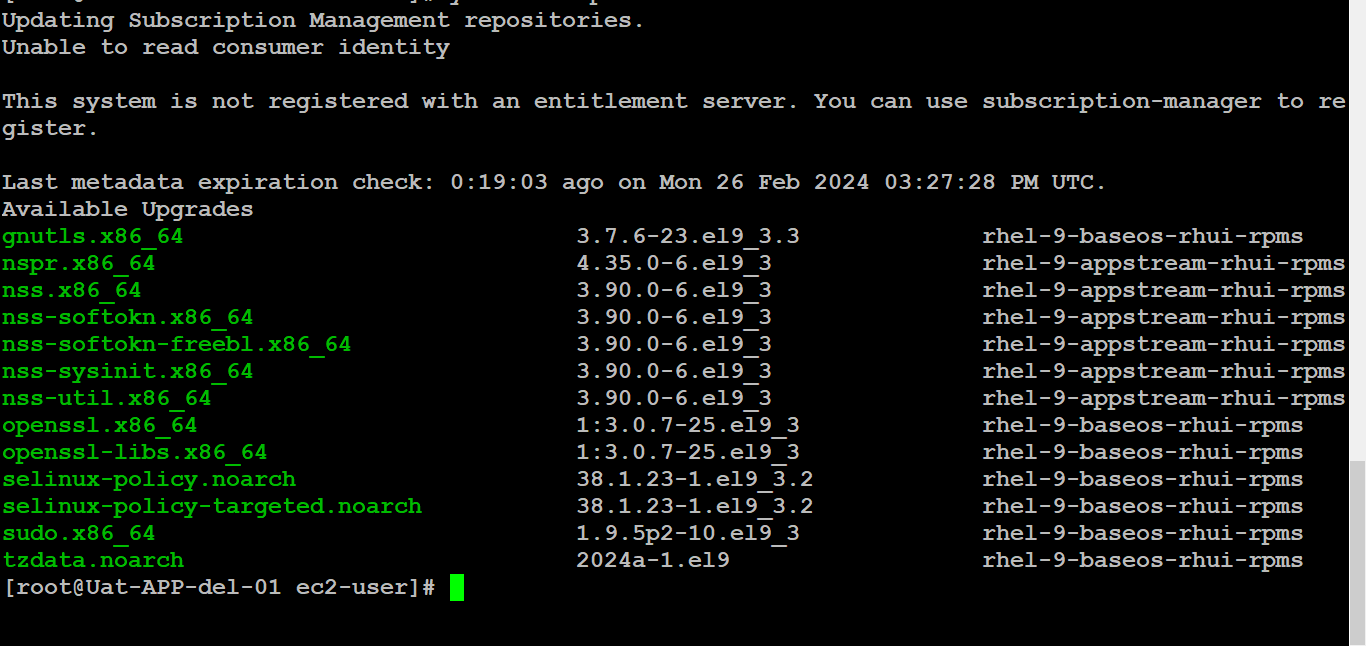
1. Step 1: login to your instance.



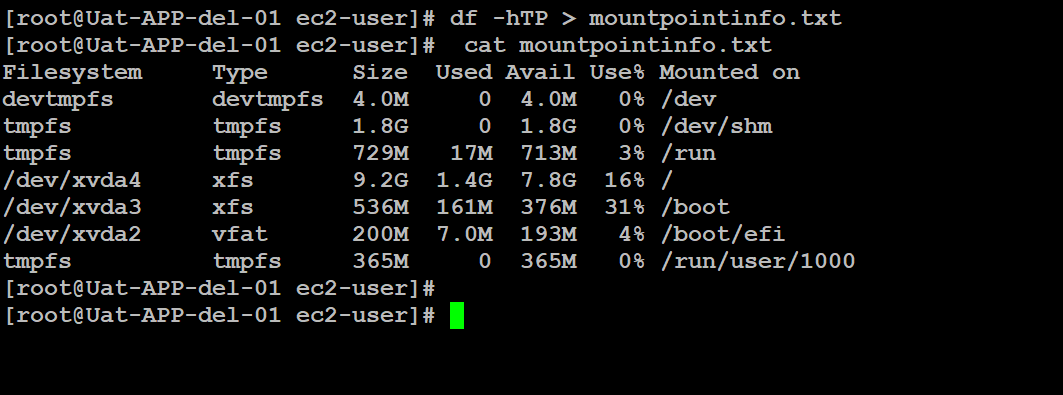
1. Step 2: Check the POA, Pre patch and Post patch**. POA (Plan of action document which is a team/collaborative effort.)** which has information like approval from all stake holders, Ask infra team to take a backup, take pre patch reports, take required config/ property files on local machines. Do it in mentioned downtime. **Pre patch report**: it is generated before we start the patching process and it has the list of all the patch versions which will be upgraded post activity. Post patch
2. Step 3 : Generate Pre patch report:

[ec2-user@Uat-APP-del-01 ~]$ sudo su

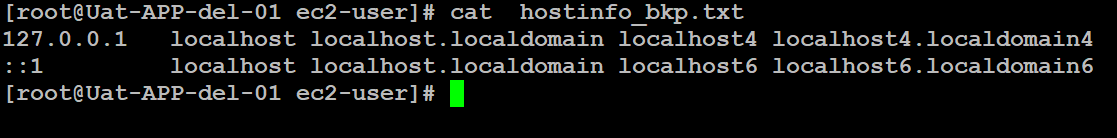
[root@Uat-APP-del-01 ec2-user]# yum list updates > prepatch.txt



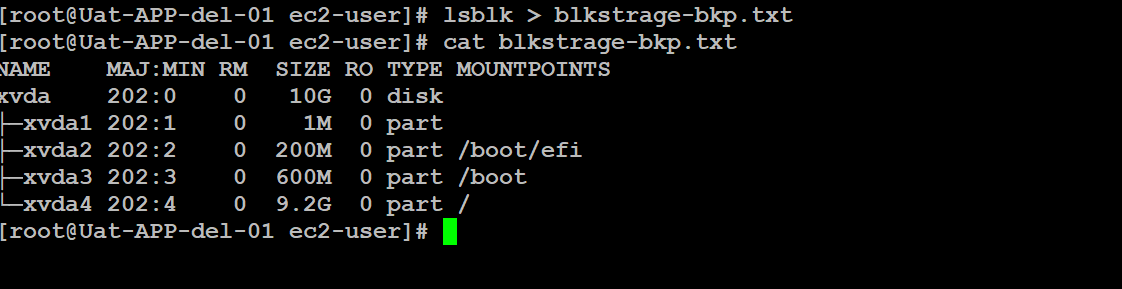
1. Take a backup of previous/ current/ pre ptach kernel version copy, luckily our kernel version is latest but we have to upgrade these packages: **uname -r > oldkernelversion.txt**
2. Before we initiate patching check the mount point by: **[root@Uat-APP-del-01 ec2-user]# mount –a > mountpointinfo.txt** and take a backup of all the mount points pre activity: **[root@Uat-APP-del-01 ec2-user]# df -hTP > mountpointinfo.txt.:**

****

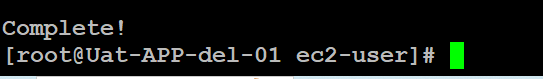
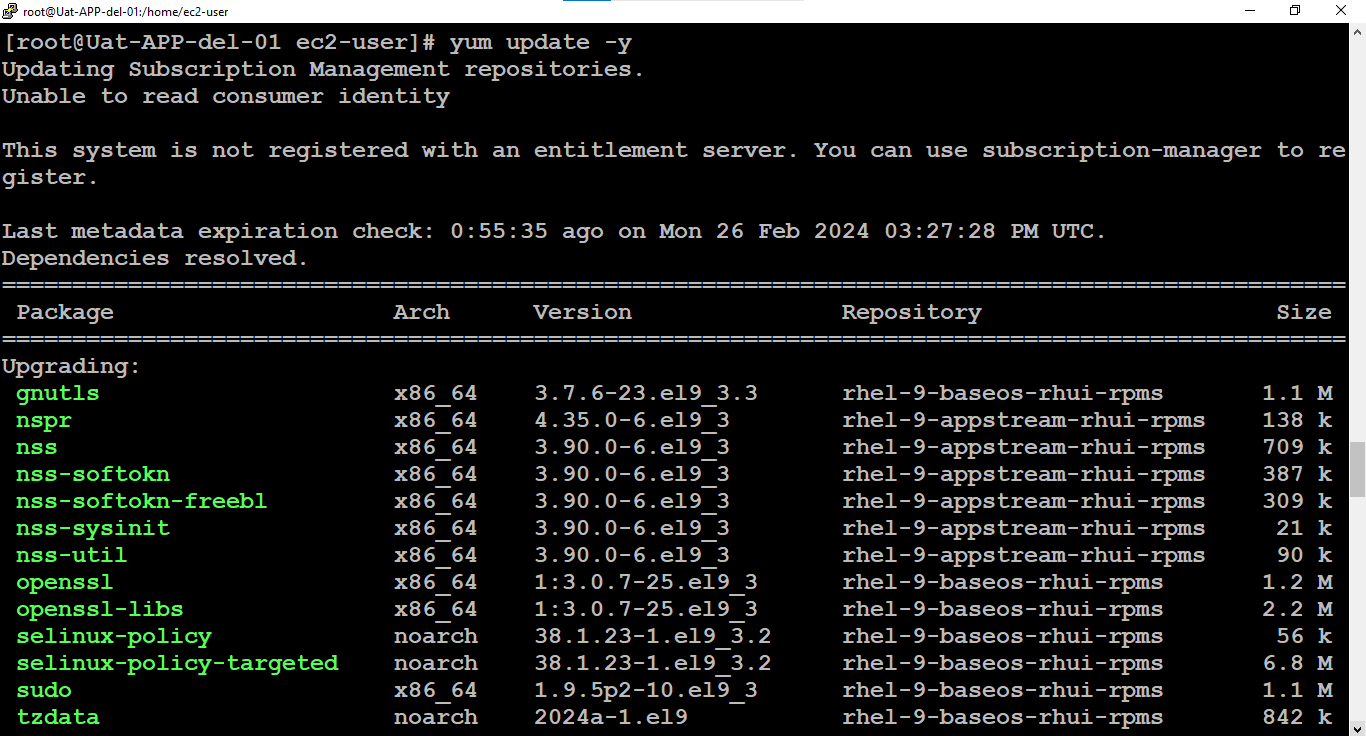
1. Take a backup of host entries: **[root@Uat-APP-del-01 ec2-user]# cat /etc/hosts > hostinfo\_bkp.txt**

****

1. Block storage information: **[root@Uat-APP-del-01 ec2-user]# lsblk > blkstrage-bkp.txt**



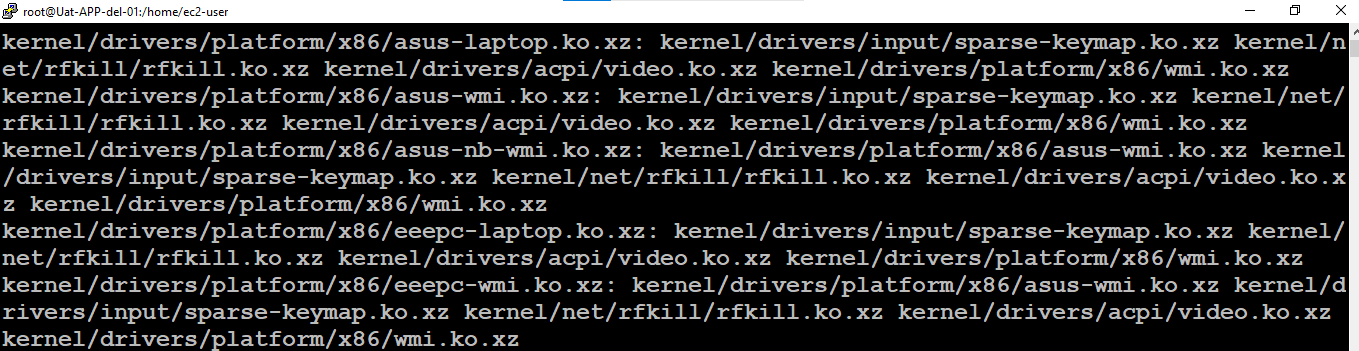
1. Initiate manual update if you don’t have automation ease which can be achieved by Ansible playbook or some crons configured earlier. BY: **[root@Uat-APP-del-01 ec2-user]# yum update –y**

****

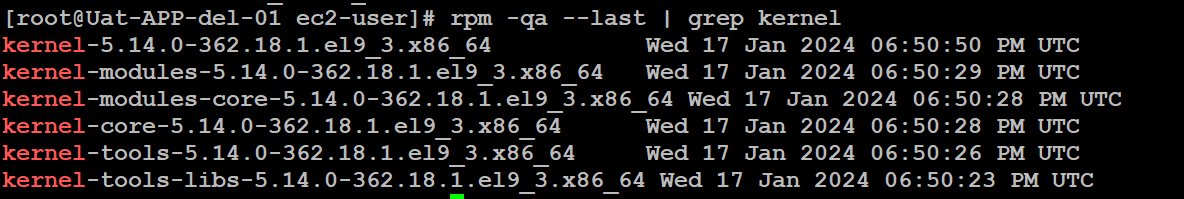
1. Check module.dep file whether it is generated pre reboot or not to check whether there is kernel panic or not.

Note: The *modules.dep* as generated by **module-init-tools**  **depmod**, lists the dependencies for every module in the directories under */lib/modules/version*, where *modules.dep* is. It is quite a big file.

**[root@Uat-APP-del-01 ec2-user]# cat /lib/modules/5.14.0-362.18.1.el9\_3.x86\_64/modules.dep > module.dep\_bkp.txt**

****

1. Run: uname –r // check kernel version before the reboot in your case it might upgrade.
2. Check which kernel version will be upgraded post reboot: [**root@Uat-APP-del-01 ec2-user]# rpm -qa --last | grep kernel**

****

**12 Take a reboot and match all the files. With same but \_new.txt**

# The End