**Lecture 20**

**Yum-OS-Software-Kernel-Patching-Undo-RollBack**

RPM

How to install “rdesktop” with all dependencies.

* **$ rpm -ivh –ndeps <package\_name>.rpm** 🡪 it will install the package without dependencies.
* **$ rpm -e <package\_name>.rpm** 🡪 to erase the package.
* The purpose of installing the package with “- -nodeps” , if only documentation is the requirement.
* If you are aware that the dependency is not critical.
* The "--nodeps" flag is used in rpm package manager to ignore dependency checks when installing or upgrading a package. This can be useful when installing packages that have unresolved dependencies, or when you want to force an installation or upgrade despite dependency conflicts. However, it should be used with caution as it can potentially result in an unstable system.
* To know where the dependency is ?
* **$ rpm -q –whatprovides <package\_name>.rpm**
* The "whatrequires" flag is a command in RPM package manager that can be used to display a list of packages that require a specific package.
* The syntax of the command is as follows:
* rpm -q --whatrequires <package\_name>
* Replace "package\_name" with the name of the package you want to check for dependencies.
* The command will output a list of all packages that require the specified package as a dependency. This can be useful for troubleshooting dependency issues or determining which packages may be affected by the removal of a certain package.
* here's an example of how to use the "whatrequires" command in RPM package manager.
* Let's say you want to find which packages require the "libcrypto.so.1.1" file. You can use the following command:
* **rpm -q --whatrequires libcrypto.so.1.1**
* The output may look something like this:
* **openssl-libs-1.1.1j-1.fc32.x86\_64**
* This tells you that the "openssl-libs" package requires the "libcrypto.so.1.1" file.
* Alternatively, you can search for which packages require a specific package. For example, let's say you want to find which packages require the "vim-enhanced" package. You can use the following command:
* **rpm -q --whatrequires vim-enhanced**
* The output may look something like this:
* **No package requires vim-enhanced**
* This tells you that no package requires the "vim-enhanced" package. If any package did require it, the command would have listed the packages that depend on it.
* .
* The "whatprovides" flag is a command in RPM package manager that can be used to display a list of packages that provide a certain file or capability.
* The syntax of the command is as follows:
* rpm -q --whatprovides <file\_name or capability\_name>
* Replace "file\_name" with the name of a file, or "capability\_name" with the name of a feature or function that you want to search for.
* The command will output a list of all packages that provide the specified file or capability. This can be useful for troubleshooting dependency issues or finding which package provides a certain file or feature.
* here's an example of how to use the "whatprovides" command in RPM package manager.
* Let's say you want to find which package provides the "libcrypto.so.1.1" file. You can use the following command:

rpm -q --whatprovides libcrypto.so.1.1

* The output may look something like this:

openssl-libs-1.1.1j-1.fc32.x86\_64

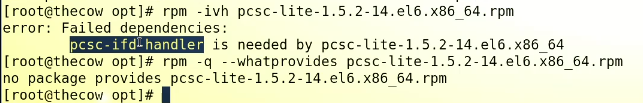
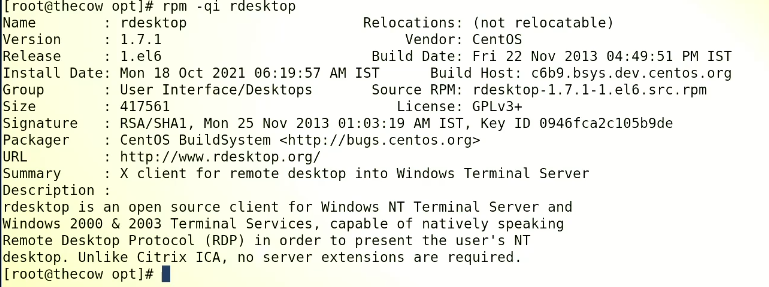
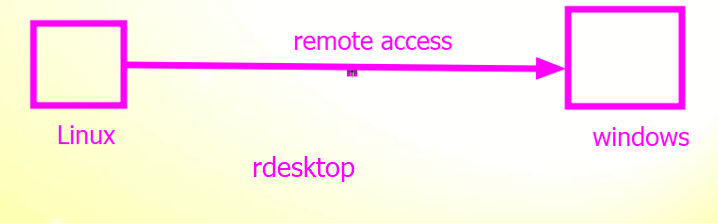
* This tells you that the "openssl-libs" package is the one that provides the "libcrypto.so.1.1" file.
* Alternatively, you can search for a capability provided by a package. For example, let's say you want to find which package provides the "ssh" capability. You can use the following command:

rpm -q --whatprovides ssh

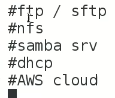
* The output may look something like this:

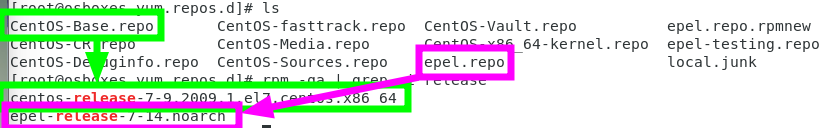
openssh-clients-8.3p1-4.fc33.x86\_64

openssh-server-8.3p1-4.fc33.x86\_64

* This tells you that the "openssh-clients" and "openssh-server" packages both provide the "ssh" capability.
* As described by Sir Kazim
* 
* The ifd-handler is not a SO file which means it is not important (SO files are critical and important to have)
* So the pcs-lite can be installed with “-- nodeps” flag which will further install “rdesktop” package in the system.
* “rdesktop”
* 
* $ rdesktop <IP of windows machine>
* All info of rdesktop
* 
* To know the original package which installed a file or father of that file.
* 
* Working of “rdesktop”
* 
* Linux to linux 🡪 through ssh
* Linux 🡪 Linux GUI 🡪 vncserver & vncviewer

**Yum (Yellowdog updater, modifier)**

* $ cat /etc/redhat-release
* 
* Important interview topics
* 
* To check the origin of these “repo” files
* 



* To check whether a package is installed or not
* $ rpm -qa | grep -I <package\_name>

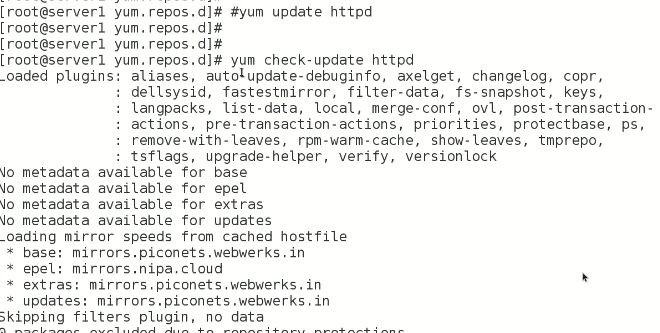


* If a package is installed and needed to be upgraded
  + Vulnerability scanners are run and new package is recommended to be installed
* $ yum update <package\_name>

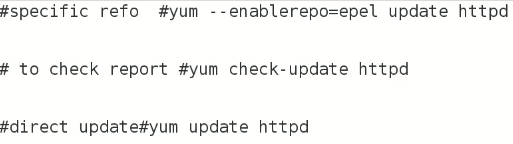


* All the repos will be scanned and whichever repo will have higher version 🡪 it will be downloaded and installed
* To check availability of updates
* $ yum update <package\_name>



* 
* To looking a specific repo for downloading.



* 
* 
* $ yum info <package> 🡪 information related to a package.
* If there is a requirement that an updated version of a package is required to be installed,
* **$ yum check-update <package\_name>**
* If it is not available anywhere within repos



* Search it on internet.



* Google > package\_name repo and **after adding the repo manually** download from the specific webpage.



* If there is no way to find the repo
* 🡪 this topic is yet to come
* It is called 🡪 
* (interview question)
* How to install a specific package
* Version 5.6.6 only
* Answer:-
* I’ll download tar.gz file 🡪 then I’ll install it via compilation from source code,
* Interview question
* How to update “kernel”
* How to update kernel???
  + $ Yum check update 🡪 to check updates availability
  + $ Yum update kernel
  + **Kernal installation process is different**
  + To update whole OS 🡪 **$ yum update**
  + ***But discuss it with IT and developer teams before this action***
* if something unnecessary is how to roll back ?
* **$ yum check-update 🡪 to get report and send this report to developers teams to discuss it 🡪 and get necessary approvals.**
* If there are few packages to be excluded from update.
* 🡪 mysql and php packages will be excluded
* This is called patching in Linux

Patching in Linux refers to the process of applying updates to the Linux operating system to fix bugs, security vulnerabilities, and other issues. Patching is an important part of maintaining a secure and stable Linux system.

There are several ways to apply patches in Linux, including:

1. Using package management tools: Most Linux distributions have a package management system that allows you to easily update your system with patches. Examples of popular package management tools include apt-get, yum, and dnf.
2. Manually applying patches: You can also download patches from the internet and apply them manually. This typically involves downloading a patch file, applying it to the appropriate files on your system, and then recompiling the affected software.
3. Using automatic update tools: Some Linux distributions, such as Ubuntu and Fedora, have automatic update tools that can automatically download and apply patches as they become available.

It's important to keep your Linux system up-to-date with the latest patches to ensure that it remains secure and stable. It's recommended to regularly check for and apply available patches to your system.

* Steps of patching
  + $ apache - - version 🡪 to check current version
  + $ yum check-update 🡪 Check update and if a higher version is foind
  + $ yum update http 🡪 update by this command

The process of patching in an industry typically involves the following steps:

1. Inventory and Risk Assessment: The first step is to identify all the systems, software, and devices in the organization's environment and assess the risk associated with each. This will help prioritize which systems and applications need to be patched first.
2. Patch Testing: Before deploying patches to production systems, it's essential to test them in a controlled environment to ensure that they won't cause any unintended consequences or conflicts with other systems or applications. Testing can be done in a staging or pre-production environment.
3. Deployment: Once patches have been tested and validated, they can be deployed to production systems. The deployment process should be carefully planned and coordinated to avoid any disruptions to business operations.
4. Verification: After patches have been deployed, it's important to verify that they have been applied correctly and that systems are functioning as expected.
5. Monitoring: Ongoing monitoring is necessary to detect any issues or vulnerabilities that may arise after patching. Regular vulnerability assessments and penetration testing can help identify any new vulnerabilities that need to be addressed.
6. Documentation: It's important to maintain a record of all patches applied to systems and software, including when they were deployed, who deployed them, and what changes were made.

It's important to note that the patching process can vary depending on the size and complexity of the organization's environment, as well as the types of systems and applications that need to be patched. Additionally, patching should be part of a broader security strategy that includes other measures, such as access control, security awareness training, and incident response planning.

* + As described by Sir Kazim
  + 
  + If the interviewer asked how to tackle the down time if service is stopped,
    - Answer : - we have another server (2 clusters 🡪 multiple machines doing same job) in the cluster which handles the process while patching the other machine.
    - The service of node 1 is shift (failover) to node so that the clients may connect to node 2 to avoid downtime. And failback to node1 after patching.

Failover and failback are two critical processes that are used in high availability and disaster recovery planning. While failover involves switching from a primary system to a backup system in the event of a failure, failback involves restoring the primary system after it has been repaired or replaced.

The two processes are closely related, and together they form an essential component of high availability and disaster recovery planning. The following is a brief overview of each process:

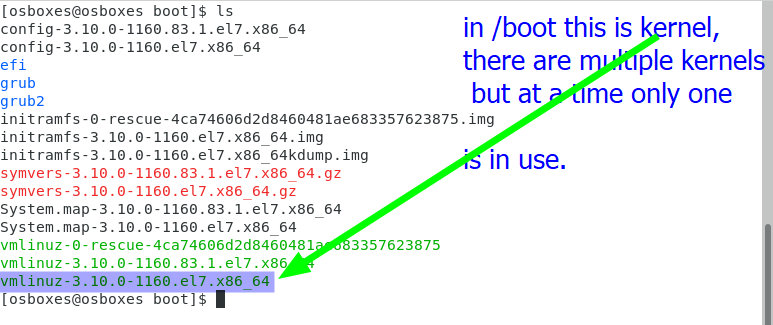
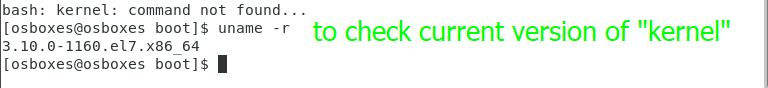
Failover:

1. Detection of failure in the primary system
2. Triggering the failover process
3. Switching to the backup system
4. Re-routing traffic and applications to the backup system
5. Restoring normal operations

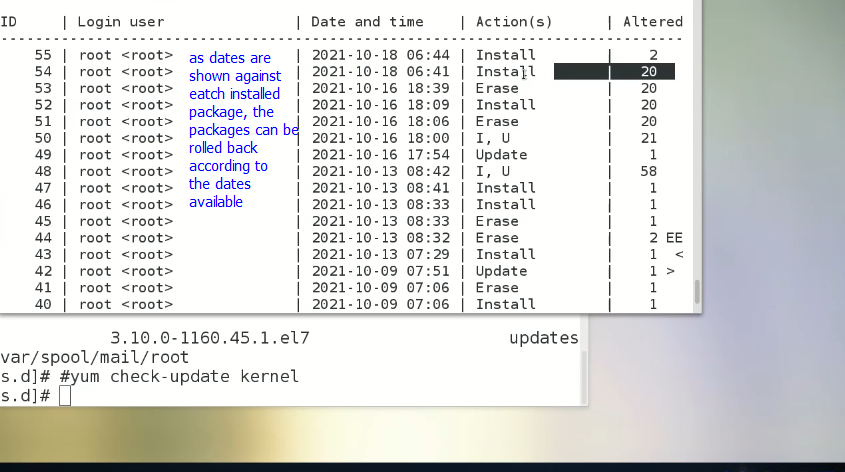
Failback:

1. Repair or replacement of the primary system
2. Preparation of the primary system for failback
3. Switchover from the backup system to the primary system
4. Verification of the primary system's functionality
5. Monitoring the system to ensure stability and prevent further failures

The success of both failover and failback processes depends on careful planning, testing, and execution. It's important to have a comprehensive high availability and disaster recovery strategy that includes failover and failback planning, as well as ongoing monitoring and testing to ensure the system remains stable and available in the event of a failure.

* Kernel patching 🡪 /boot
  + $ yum update kernel 🡪 what is the proper time to update the kernel,
  + If new fiber channel is installed which do not support the kernel and requires a newer version. It requires downtime until failover and failback can be done.
  + 
  + The **kernel update is different,**
    - **If a normal package is updated, it is overwritten by the new version**
    - **But in the case of kernel, the old one remains present and the new one takes the command.**
    - The old kernel remains in use until the system is rebooted, after reboot the new kernel takes the charge.
    - 
    - Two entries are listed in GRUB 🡪 can be selected at the time of boot that which kernel should be used.
  + Interview question,
    - If new kernel is installed and after rebooting server is not coming up ?
    - Answer:- “I’ll boot the machine from old kernel and after booting I’ll diagnose the issue.”

**Rollback**

* ***If patching is to be rollback for some reason,***
* ***Roll back to the system to a specific date.***
* ***How to do yum rollback?***
* **$ yum history** 🡪 it will show what packages are installed.
* $ yum history <ID of specific package>
* $ yum history info 54
* 
* **$ yum history rollback 53 🡪 roll back to date 16-10-2021**
* $ yum history undo <package ID> If only a specific package is to be roll backed
* 