Essential Commands

Log in to Local and Remote Graphical and Text Mode

Lecture:Consoles

Both Ubuntu/Debian and CentOS/Redhat

VNC Viewer is a tool used to connect to a VNC Server which allows for remote operation of a Linux distribution in graphics mode. It is similar to other technologies such as Remote Desktop for Windows. VNC ordinarily runs over tcp/ip ports starting at 5800, but can be configured.

Secure Shell or ssh is used to connect to remote servers at a command line,ordinarily over port 22. The connection is encrypted, and while it usually relies on username/password authentication, it can be configured instead to make use of user-generated keys for authentication purposes.

Lecture:Search for Files

Linux is case sensitive! This means that "TEST", "test", "Test", and "TesT" all register as completely different.

find –help for all of the **find** command options

Remember it also has a man entry:
man find

Examples:

Find all files in the entire system that are named test.txt:
find / -name "test.txt"

Find all files in the /etc directory not named test.txt:
find /etc/ -not -name "test.txt"

Find all character devices on the system:
find / -type c

Find all directories called log:
find / -type d -name "log"

Find all files in the /usr/bin directory or subdirecties that are larger than 27,000 bytes:
find /usr/bin -size +27000c or find /usr/bin -size +27k

Find all files in the /usr/bin directory or subdirectories that are larger than 1 megabyte:
find /usr/bin/ -size 1M

Find all files created more than a day ago:
find / -mtime 1

Find all files create less than a day ago:
find / -mtime -1

Find all files owned by the user named "chad":
find / -user chad

Find all files in the /etc directory owned by the user root, and paginate the results:
find /etc/ -user root | more

Find all files in the /usr/bin directory with user permissions of 755:
find /usr/bin -perm 755

Find all files called test.txt and set their permissions to 700:
find / -name "test.txt" -exec chmod 700 {} ;

The **more** command will cause the output to pause after it fills a page and wait for you to hit the spacebar before continuing.
There is also a **less** command that will allow you to scroll back and forth through the output using arrow keys.

The **chmod** command changes a file's permissions

The **which** command allows you to find the location of a command that will be **executed** if invoked.

This is useful for locating the **executable** file, especially if the behavior is not what you expected.

which

For instance, to find out which python executable would be invoked:

which python

The **locate** command can also help you find files but relies on a database rather than looking directly at the file system. The two can easily be out of sync. To synchronize locate's database, run the command (as root) **updatedb**.

locate

Lecture:Compare and Manipulate File Content and Use Input-Output Redirection

The **cat** tool will read file contents to standard out, which is usually the screen.

The **more** command pauses output.

The **pipe** character takes the standard output from one command and passes it to another command. For example, to send the output from cat shoppinglist through more:

cat shoppinglist | more
or
more shoppinglist

The **less** command will paginate, similar to more, but it will also allow you to use your up and down arrow keys to scroll backward and forward through the file.

cat shoppinglist | less
or
less shoppinglist

The **sort** command sorts content passed to it on standard in and places the output on standard out.

sort shoppinglist

To sort in reverse order:

sort -r shoppinglist

The **cat(categories)** tool's name is short for concatenate. You can combine the output of multiple files like this:

cat file1 file2

You can achieve complex results by combining commands:

cat file1 file2 | sort > file1and2

The **>** and **<** symbols allow you to send the standard output to a particular file on disk. The above example would have sorted the two files called file1 and file2 together and created a file in the current working directory called file1and2.

The **fmt** (format) command will clean up and format text.

The **nl** command counts the number of lines in a file, and also will number each line in a file for reference. (satır numaralandırma)

The **cut** command will remove a specified delimiter and write out a part to standard output.

One;Two

Three]Four

If we execute the following on the above file called numbers,
cat numbers | grep ";" | cut -d ";" -f1

The output would be One. (dosyadan ; 1 numara dısında herseyi keser, gösterir)

If we execute the following on the above file
cat numbers | grep ";" | cut -d ";" -f2,

then the output would be Two. (dosyadan ;2 numara dısında herseyi keser,gösterir)

Lecture:Analyze text using basic regular expressions

Some basic regular expression help:

Token	**Meaning**
^	Start of line
$	End of line
\<	Start of word
\>	End of word
\d	Digit
\D	Not a digit
[Aa1]	Either match on A,a or 1
[A-Z] or [:upper:]	Matches on any capital letter
a-z] or [:lower:]	Matches on any lower case letter
[^lmno]	Matches on any charactr but l, m, n. or o.
\$	Matches a literal dollar sign

^A : line daki A ile başlayan satırlar
Z$: line daki Z ile biten satırlar

grep '^The' alice.txt : alice.txt deki THE ile başlayan satırları gösterir.

grep '^T[a-z][aeiou]' alice.txt : alice.txt deki T ile başlayan,2.harfi a-z arasında olan ve 3.harfi seçilen(aeiou) harflerden olan satırları gösterir.

grep '^T[a-z][^e]' alice.txt : alice.txt deki line'larda T ile başlayıp ikinci harfi bilmiyorsak ve 3.harfi "e" ile bitmeyen (This, Those vs.) satırları göster.

grep '\<[tT]he\>' alice.txt : alice.txt deki küçük ya da büyük t ile başlayan ve he ile devam eden kelimeleri gösterir.

Lecture:Archive, Backup, Compress, Unpack, and Uncompress Files

mkdir: make a new directory (mkdir data_backup)
gzip: dosya sıkıstırma komutu(gzip data_backup.tar)
rm : dosya silme (remove) komutu (rm data_backup.tar.gz)
**clear (Control+l): terminali temizler

The cp command allows you to directly copy files from one location to another.

cp (copy)

Such as:
cp /home/user/test.txt /opt/test.txt

The **scp** command allows you to copy files securely over the network to another host running an ssh server.

scp

Such as sending a file to another host:
scp /home/user/test.txt user@otherhost:/home/user/test.txt
Or retrieving a file from another host:
scp user@otherhost:/home/user/test2.txt /home/user/test2.txt

The **rm** command immediately deletes files.
rm unneededfile.txt

The **tar** (**t**ape_**ar**chieve) command is the de facto standard for creating backups.

To create an uncompressed backup:
tar cvf backupfile.tar /path/to/backup
To create a compressed backup:
tar cfzv backupfile.tar**.gz** /path/to/backup

Mnemonics for commonly used tar flags:

Flags	Means
c	Create
v	Verbous
f	File
z	Zip
x	Extract
t	List

The **grep** tool allows you to search for a **particular** output. It accepts input on standard in and puts only lines containing the pattern you are looking for back out.

tar tvf databackup.tar | grep searchingfor.txt

The **gzip** tool is a compression utility that will compress and uncompress files. Text files compress very efficiently, so zipping them up will result in a far smaller file, with the disadvantage that it will have to be uncompressed before it can be read. By default, it will compress files "in place," meaning it will take a file, compress it into the same name with a ".gz" appended to the end, and remove the original file.

gzip data_backup.tar

Creates in a file in the directory called somefile.tar.gz, but the original somefile.tar is gone.

To create a compressed backup file, use the compression options in tar:
tar cvfz databackup.tar.gz /data

To extract the contents of a compressed tar file to the current directory:

tar xvfz databackup.tar.gz

Lecture:Create, Delete, Copy and Move Files and Directories

The **touch** command quickly creates a zero-byte file. (0 byte lık bir dosya yaratır)
touch testfile/data

The **cp** command allows you to copy files from one location to another.
cp testfile testfile2

The **nano** command is a simple text editor. On some Ubuntu systems, there is an alias to this for users of the older, deprecated pico editor.
Pico testfile2

The **mkdir** command allows you to create directories.(Klasör yaratır)
mkdir secondfiles

You can create several at once using the -p flag: (alt klasör olusturma)

mkdir -p secondfiles/textfiles/otherfiles/resources

The **mv** command is used for moving files from one directory to another. It's also used for renaming files. Think of renaming as just moving a file from one name to another:
mv testfile secondfiles/

The **rm** command allows you to delete or remove files (and sometimes directories with the right flags and wildcards).
rm testfile

or, beware, a command like this will remove all files and directories in /home/user that start with "unneeded" and, if a subdirectory, remove it completely:
rm -rf /home/user/unneeded* (klasör ya da altkalsör silme yaparken en sondan başlarız)

The **rmdir** command is for specifically removing empty directories. It will fail if there is anything in the directory. !!!! For your own safety, get in the habit of using rmdir to remove directories, and use those error messages to your advantage so you don't accidentally wipe out half (or more) of your important files.

**cd .. : Bir önceki dizini okuma komutu **

Lecture:Create and Manage Hard and Soft Links

The directory structure on your hard drive is basically a list of named pointers (directory file names) to locations on the disk. A **hard link** is a direct pointer to that location, and a soft link is a link to another file name in the directory structure. A file on disk can have multiple hard links pointing to it so that you can edit the file from multiple directory locations.

** pico file1 yarattık
** pico file2 yarattık ... Bunlar block device da hard link denir.

** mkdir dir1
** mkdir dir2 klasörleri olusturduk.

**mv file1 dir1/ (file1 dosyasını dir1 klasörüne gönderdik)
**mv file2 dir2/ (file2 dosyasını dir2 klasörüne gönderdik)

The **ln(link)** tool allows you to create both hard and soft links to files.

To create a hard link:

**dir1 klasöründen file2 dosyasının içindeki dataya erişim için hard-link olusturuldu.(hard-line-to-file2 isminde)

ln ../dir2/file2 hard-line-to-file2

To create a soft link: -s (symbolic)

ln -s ../dir1/file1 file1

**dosyadaki datalar herhangi bir klasörde editlense bile değiştirilen datalar diğer klasörden görülebilir.(hard-link)

**aynı klasörde farklı sistemlerde bir klasörü spesifik olarak tanımlayıp erişme (soft-link..symbolic link..sim links..)

Lecture:List, Set, and Change Standard File Permissions

*ls -la = ll >> listeleme komutu
*chmod >> change mode (permission değiştirmede kullanılır)

The **chmod** tool allows you to set permissions on files.

Directories listed with the -la flag show privileges in three sets of characters like this:

drwxrwxrwx 24 owner group size date filename

The first character tells you whether the entry represents a directory or not:

d— — —

The next three characters tell you which of the read, write, and execute rights that the file's owner has:

-rwx— —

The next three characters tell which of the read, write, and execute rights that the file's group has:

—-rwx—

The next three characters tell which of the read, write, and execute rights that everyone else has:

— —-rwx

To change file permissions, you can use chmod with the character method like below.

1) To add group read and write access to a file:
chmod g+rw file

(Not: eğer everyone(other) için yapsaydık; chmod o+rwx filename)
(Not: eğer herkes (user,group,everyone) için yapsaydık ; chmod a+rwx filename

2)To summarize the character method, a truncated usage statement is presented below:
chmod [ugoa]*([-+=][rwx]*[ugo]

4: read
2:write
1: execute
0: not permission
7: all(a) (read+write+execute)
6: read+write
5: read+execute
3: write+execute

So if you want to give the owner full rights, the group read and write rights, and everyone else only read rights to a file, you'd use:

	owner	group	everyone	
chmod	7	6	4	filename

Justification:

Math Characters Meaning

7 = 4 + 2+ 1	rwx	Full rights.
6 = 4 + 2	rw-	Read/Write
4 = 4	r–	Read only

Lecture:Manage Access to the root Account

Protect your server. Don't log in as root and don't allow anyone else to, either. Don't share a single password. Use the **sudo** command to escalate your privileges. Change to the root user (**su or sudo su**) only when necessary, and log out as soon as you've completed your tasks.

The **sudo** command allows you to "do" things as the "super user" that you ordinarily wouldn't have appropriate permissions to do.

sudo
sudo !!(önceki komutu çalıştırır.)

You'll be asked for YOUR password to ensure that it's you.

Use the **su** command to become another user, usually the root user.

su root (#root is implied if you only type "su") : super user

You will be prompted for the password belonging to the user whose identity you are attempting to assume.

The **whoami** command will return the name of the Linux user you are currently logged in as.

root@somehost$ whoami

root

To provide a user with access to use the sudo command, use **visudo** to edit the sudoers file:

visudo

Add the user like others in the file.

Alternatively, you can set up a group that has sudo privileges in the sudoers file, and you can then assign users to the group to dynamically alter privileges

usermod -a -G wheel chad

su chad

Operation of Running Systems

Lecture:Boot, Reboot, and Shut Down a System Safely

The **shutdown** command allows you to safely shut down the system while notifying users of what's going on.

To power off a system now with a message of "System Maintenance – New Hard drive! Back online in half an hour or so!":

shutdown -h now "New Hard drive! Back online in half an hour or so!"

Common (but not complete) usage (Always see man pages for complete documentation)

shutdown -[h - halt | r - reboot] <when?>

Shortcut commands reboot and poweroff do exactly what you'd expect them to do.

Note that changing the running status of the system requires root privileges.

NOTE:

1.Verify completion of scheduled jobs: cd /var/log -> cat syslog | grep CRON
2. reboot or shutdown a system : shutdown -r now,, shutdown -h now
3.cmd to see how long a computer has been running : uptime

Lecture:Install, Configure, and Troubleshoot Bootloaders

Grub 2 is the de facto entrenched bootloader in use by most Linux distributions as of the writing of this guide.

/boot/grub/grub.cfg is grub's main configuration file. It is autogenerated and should not be manually edited. It is autogenerated by the **update-grub tool.**

/etc/grub.d is a configuration directory used by **the update-grub tool** to generate the grub.conf file and also specified menu entries.

The sample grub code I created in the video is:

```
#!/bin/sh -e
echo "displayed when update-grub is run"
cat << EOF
menuentry "Other Linux Partition" {
set root=(hd0,3)
linux /boot/vmlinuz
initrd /boot/initrd.img
}
```

grub-install /dev/hda
grub-install --root-directory=/mnt /dev/sda

Lecture:Diagnose and Manage Processes

The **top** command will show you a real-time, live updated console containing processes, their CPU utilization, and lots of other great information.

The **htop** command offers similar functionality, with a few more interactive options and major cosmetic differences.

The **ps** command will allow you to find processes running on the current machine to identify their owner or pid (process id).

Common ps invocation:

ps aux

The **kill** command is used to send termination signals to running applications.

Here is a brief list of kill signals commonly used (not complete). (kill -l)

NOTE: List of all running process: ps aux
NOTE: Linux system priorities : 0 to 139. 0 to 99 for real time. 100 to 139 for users
NOTE: start process with lowest priority level: nice -n 20 <programme>
NOTE: how to see hierarchy of process : ps acjf
NOTE: anothr way to find specific process running using ps: ps aux | grep "whatever"

Lecture:Locate and Analyze System Log Files

Usually in /var/log/messages for CentOs and /var/log/syslog for Ubuntu, most relevant logs are located there. For application files, consult their documentation.
(messages log file herseyi gösterir)

specific olarak Error loglarını görmek için;
grep "Error" messages

Shift + G : son satıra getirir

Cat, less, more, grep, and input-output redirection covered in a previous lesson are your best friends with log file analysis.

Lecture:Schedule Tasks to Run at a Set Date and Time

A user's **crontab** contains a list of commands a user wishes run at very particular times. These commands are run by a daemon called cron. To view a user's crontab, use this command (if you only want to see your own, you need not include the username):

crontab -u user -l

To delete a user's entire crontab (all commands GONE!):

crontab -u saduser -r (remove)

To edit a user's crontab:

crontab -u user -e (edit)

Here is an example of running a script called "/opt/do-the-things.sh" at 8:00 AM every weekday in March:

0 8 * 3 1,2,3,4,5 /opt/do-the-things.sh

m h dom mon dow command

m: minute
h : hour
dom: day of month
mon: month
dow: day of week

Position	Meaning
1	Minutes after the hour the scheduled command is to run.
2	Hour of the day
3	Day of the month (1 - 31)
4	Month (1 - 12)
5	Day of the week (0-6) (Sunday = 0)
6+	The command

Lecture:Update and Manage Software to Provide Required Functionality and Security

Part 1: Ubuntu/Debian

On Debian/Ubuntu systems, the package management system is Aptitude and the two common tools used with it are **apt** and **apt-get**. Apt-get tends to receive preferential treatment because it doesn't do fancy stuff to the screen, thus making it ideal for inclusion in scripts.

Often, you'll need to update the repository cache by running an update:

apt **update**
or
apt-get update (update all of list or all of packages)

And to update all out-of-date packages:

apt **upgrade**
or
apt-get upgrade

On most systems, you will need escalated privileges, so remember to sudo where necessary.

To install a new package, use:

apt install or apt-get install

To remove a package, you will usually want to remove everything associated with it, like so:

apt purge
or
apt-get purge nmap = apt-get remove - - purge nmap

The **dpkg** tool is used when you already have a Debian package file (*.deb) that you want to query or install.

To install a package file, use the command:

dpkg -i somepackage.deb **-i (install)

Part 2: CentOS/Redhat

On Redhat and CentOS, the Redhat Package Management system is in place. The packages are **.rpm** files and the package management command, the equivalent to Debian's dpkg command is also called **rpm.** There is an apt type of system called **yum**.

To update the repository cache and packages on the system:
yum update

This lets us know if a package is installed:
yum list

To find out if a packagename exists (wildcards are okay): ***nmap: namepackages
yum search nmap

To install a package and any necessary dependencies:
yum install

The **-y** switch will answer yes to all questions so that we're not prompted during an install:
yum install -y

The yum-utils package includes some excellent additional tools, such as yumdownloader which fetches the package but does not install it.

The **yum-list** command will list all packages available in the database (grep is helpful here).

Display enabled yum repositories:
yum repolist

Display ALL yum repositories:
yum repolist all

The rpm tool is useful when you have a .rpm file that you'd like to install or query. To install a package from an rpm:
rpm -i somepackage.rpm

List all files installed by a particular package:
rpm -ql

Examine docs inside a package:
rpm -qdf

Flashcards

"yum update" to update packages
"yum search (package name)" for information on package
"yum install (name of package)" to install package
"yumdownloader (name of package)" to download the file/package without installing
"yum list" shows all package names
"yum search (name of package)" to find a specific package
"yum list installed" tells you the packages installed on the system
"yum list installed | grep cron" will show you all the cron packages in the system
"yum grouplist" shows groups available as some applications are grouped together for a node
"yum groupinstall 'name of group'" for installing groups
"yum repolist" shows all repos
"yum repolist all" shows all the repos whether enabled or disabled
"yum --enablerepo=extras-source/7 pkgname"
"yum clean all" will clean or remove disabled or repo packages
"yum history" to show the history of what's been done recently
yum install xterm looks for all dependencies and packages in order to install onto the system
In order to remove packages and the dependences, use **"yum autoremove"** to remove dependencies

"rpm -qpR (package)" to query dependencies in a specific package (xterm for example)
"rpm -q (package name)" to determine if the package is installed
"rpm -ql (package)" to show all the files that were installed along with the specific package
"rpm -qa --last" shows all the recently altered packages
"rpm -qdf /user/bin/vmstat" shows everywhere vmstat is shown in the documentation dependency
rpm -qa gpg-pubkey* tells you the public keys that have been installed in order to access different repositories.
rpm --rebuilddb to rebuild the database
rpm -i <pkg_name> *** install***
rpm -r <pkg_name> *** remove ***
rpm -q <pkg_name> *** query ***

Lecture:Change kernel Runtime Parameters, Persistent and Non-Persistent

The **sysctl** tool is made for examining and changing kernel parameters at runtime. It examines the virtual process file system in **/proc.**

To change a system parameter until the next boot:
sysctl parameter.name=value

To change a system parameter permanently, you'll need to change the appropriate parameter by editing the appropriate file in /etc/sysctl.

Flascards

- **sudo sysctl -a** (lists all of the different kernel runtime parameters)
- **sudo sysctl -a | wc -l**
- **/proc/sys** (location of kernel runtime parameters)

Check value of kernel parameter:
- **cat /proc/sys/net/ipv4/ip_forward**
OR
- **sysctl net.ipv4.ip_forward**

Persisting changes:
- **cd /etc/sysctl.d**

Modifying .conf files (add comment(s) describing changes made and then add the changed parameter - Ex: **net.ipv4.ip_forward=1)**

sysctl -w <param>=number -- will change for session
sysctl -p will make persistant.
---
example

sysctl -w vm.swappiness=10 will set swappiness to 10 for session.

can also echo number > /proc/sys/vm .....

sysctl -p will make persistant although may not survive reboot.

changes should be made to /etc/sysctl.conf files etc. !!!!!!

Lecture:Use Scripting to Automate System Maintenance Tasks

1. Always start with a **shebang**. That tells the system how to interpret the script.

#!/bin/bash
And point to an appropriate shell or interpreter.

2. Scripts are executed sequentially. The order of the commands matters.

3. Avoid relative paths like **../../home/**. Instead, use absolute paths like /home. This way, if you ever decide to move your script, you don't have to go back in and fix it immediately just because it lives in a new directory.

4. Log it! Get in the habit of "echoing" important information to the screen or writing results to either your own log file or appending it to the system log file. This can be very useful for debugging larger, more complex scripts later on.

Add to PATH : export PATH=$PATH:/home/chad/script

Lecture:Manage the Startup Process and Services (In Services Configuration)

In SystemD systems such as Ubuntu 14+, Centos, and RedHat systems, starting a service at boot is simply a matter of using the **systemctl** tool. Starting one of these services at boot is referred to as "enabling" it:

systemctl start/stop cron

To gain information about a particular service on a systemd system, you'd use:

systemctl status servicename

In older systems using **upstart** for services management, you'd create an appropriate configuration script in /etc/init.

To gain status information on upstart systems, you'd type:

status cron*

Flascards

In Older System

cd /etc/init (contains configuration files for telling the OS how to get up and running)

- status cron
- stop cron
- start cron
-restart cron (((her işlemde yeni pid<process id> verir)))

- **echo manual | sudo tee /etc/init/cron.override** (to manually start cron instead of it starting on system boot)
- rm cron.override (to reenable automatic startup of cron upon system boot)

In Newer System

systemctl start <cron>
systemctl status
systemctl stop <cron> (needs sudo)
systemctl enable/disable <cron> (needs sudo) - **is persistent.**

enable/disable basically creates/removes softlinks to the daemon files within /etc/systemd to service file.

crond for example. symlink /etc/systemd/system/multi-user.target.wants/crond.service --> /usr/lib/systemd/systemd/crond.service

Lecture:List and Identify SELinux/AppArmor File and Process Contexts

SELinux (Redhat/Centos) and **AppArmor (Ubuntu/Debian)** prevent applications on their respective systems from overstepping due to defects or malicious behavior.

SELinux

SELinux can be enabled/enforced, logged, or disabled.

To view all contexts on your system:
`semanage fcontext -l`

To find a specific context:
`semanage fcontext -l | grep httpd`

To view security context on a file:
`ls -Z filename`
`ps auxZ`

Used without a file name the command would show the security context of every file in a directory.

To show security context on processes:
`ps auxZ`

AppArmor:

The **aa-status** tool will show you similar security contexts on AppArmor systems.

Profiles are stored in the /etc/apparmor.d directory and you can view these profiles and see how they are constructed.

Regular expression experience is helpful, so you might want to review the basic regular expression lesson.

ls -Z is usually available on Ubuntu systems, but does not work for AppArmor profiles.

ps auxZ, however, does work

Lecture:Identify the Component of a Linux Distribution That a File

Belongs To

Specifically, you can find out which package provides a particular file on Redhat and Centos systems:

rpm -qf
or
yum whatprovides

**rpm -qf */bin/znew*
***yum whatprovides /bin/znew*

On Debian/Ubuntu machines, we'll use **dpkg**:

dpkg -S

*** dpkg -S /usr/bin/zdump*
***dpkg -L libc-bin*

User and Group Management

Lecture:Create, Delete, and Modify Local User Accounts

The tools for user creation are pretty straightforward: **useradd** and **adduser**. If you've got a lot of manual scripts you need to run or other manual work, then:

1. You should put that on your list of things to automate.

2. You should use the **useradd** command, which on many systems will not create the home directory automatically. The **adduser** command will ask you a set of questions about the user to get the new person all set up.

useradd

adduser

**adduser + <directory> -d + /home/testuser1 + <username> testuser1

Flascards

1. create username and path (old version) : useradd -d /home/testuser testuser
2. add a new user (new version) : adduser <user>
3. Create,delete and Modify Local user account: - Created user user1
   useradd -d user1 or adduser user1
   passwd user1

**chown => update user rights

4. Create directory for new user in home and change ownership
   chown user1:user1 user1

5. Remove user:
userdel user1 or deluser (userdel not delete home dir)

6. expire an account = passwd -e <username> or change -d 0 <username>

(d here stands for number of days remaining)

Lecture:Create, Delete, and Modify Local Groups and Group

Memberships

To add groups to a system, you can use either the **groupadd** or **addgroup** command, or you can edit the /etc/group file to create your group and designate its membership.

groupadd newgroup

or

addgroup newgroup

To add a user to a group, either manually edit the /etc/group file or use the usermod tool, **carefully.**

**usermod -a newgroup olduser

Note that if you hastily read the usage, you'll see that -G GROUPS is supposed to add users to groups, so if you do a:

**usermod -G newgroup olduser

Flascards:

1. CentOS
To add new group:

sudo groupadd <group name>
sudo vim /etc/group then add user to end of group

sudo gpasswd <group name> -- change passwd on group

to change group - newgrp <groupname> then enter passwd.

to delete the group - sudo groupdel <groupname>

2. change default boot target to graphical: systemctl set-default graphical

3. look at diff group: cat /etc/group
add a new group: addgroup test1 or groupadd test1
add user to group: nano /etc/group

Lecture:Manage System-Wide Environment Profiles

Environment variables are easy to use and easier to set. The **export** command is a way to explicitly define an environment variable in your current shell.

export USER_VAR="This is a test"

To unset a variable, either:
unset VARIABLE

Set persistent environment variables in .bashrc for a particular user, or system-wide in a script file located in the /etc/profiles.d directory.

Flascards:

1. Manage System-Wide Environmet Profiles:
- To set a variable you export it
export USER_VAR= "This is test"
- To unset it you do:
unset USER_VAR

- .bashrc or .profile is the file to setup env

- Where are system-wide env variables stored?
/etc/environment
/etc/profile
/etc/profile.d (this is directory)

- Remove all env variables in bash:
env -i bash

2. show environment: env
set var for just this session: export USER_VAR= "This is test"
clear out env: unset USER_VAR
edit to set global profile : nano /etc/enviroment

Lecture:Manage Template User Environment

The **/etc/skel** directory is designed to be a template of a standard set up for your users. You can place files like a standard .bashrc script (see the previous section for an application of this) or standard directory tree, or whatever you like. It beats having to do it manually and is less work than writing a script.

Lecture:Configure User Resource Limits

The **/etc/security/limits.conf** file is used to set limits on user resource utilization.

Order	Value Expected
1	Domain: Like the username, group, or wildcard that is being limited
2	Type: Either "hard" or "soft". Either the limit is absolute, or the user may change the value (up to a hard limit, if one exists)
3	What is being limited, such as the core file size, maximum data, maximum memory locked, etc. *See the man page for limits.conf for a complete list
4	Value – The value of the limit being set, in appropriate units

Flascards:

1. man page to about limits
   man limits.conf

2. edit limits.conf file to set the limits
   sudo nano /etc/security/limits.conf or /etc/security/limits.d

Lecture:Manage User Privileges

The **/etc/access.conf** file allows or disallows users from logging into a machine and starting up a shell. You can thereby create a user that cannot log in remotely, or can only log in from specific hosts or networks.

The user, group, and "everyone" execute "bit" (permissions) on executable files and scripts determine who can execute any given command.

From our previous experience with file permissions, that's these permissions:
—x–x–x

Flascards:

1. permission for group to run shell script
   sudo chgrp adm chadcmd.sh

2.change user permissions for shell script
   sudo chmod 754(rwx,r-x,r—) chadcmd.sh

3. change the privileges in the access.conf
   nano /etc/security/access.confg

Lecture:Configure PAM

PAM stands for Pluggable Authentication Modules. They allow the admin to change the way applications authenticate users.

PAM is configured EITHER by the file **/etc/pam.conf** or by all the files in a directory called **/etc/pam.d**

If this directory exists, then the /etc/pam.conf file is completely ignored.

The single configuration file method, while still valid, is rarely used. It's much easier to organize services by storing them in separate files within /etc/pam.d/.

PAM Deals with 4 management tasks:

- AuthenticationManagement
- Account Management
- Session Management
- Password Management

Networking

Lecture:Configure Networking and Hostname Resolution Statically or Dynamically

The **ifconfig** command will give you information about your networking interfaces.

1.Debian/Ubuntu Systems

On older Debian/Ubuntu systems check /etc/network/interfaces file or the /etc/network/interfaces.d for a group of the configurations

Here is an example of a simple DHCP client configuration for an ethernet device:(nano eth0.cfg)

```
bash
auto eth0
iface eth0 inet dhcp
```

Here's an example for a simple static host for an ethernet device:(nano eth0.cfg)

```
bash
auto eth0
iface eth0 inet static
address 10.9.8.7
netmask 255.255.255.0
gateway 10.9.8.1
dns-search mydomain.com
dns-nameservers 8.8.8.8 8.8.4.4
```

To restart your network interface from the command line.

```
sudo ifdown eth0 && sudo ifup eth0
```

For the new Ubuntu(16) : nano 50-cloud-init.cfg

2.Redhat/CentOS Systems

On Redhat and CentOS systems, the network is configured by scripts in a directory called **/sysconfig/network-scripts.**

Ordinarily, each interface has its own configuration file along the lines of **ifcfg-ethX.cfg** which is very similar in many respects to the **/etc/network/interfaces.d** files on an Ubuntu or Debian system.

Here is an example of a DHCP configured client:(nano ifcfg-eth0)

```
bash
BOOTPROTO=dhcp
DEVICE=eth0
HWADDR=0a:67:42:8d:24:9e
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
```

Here is an example of a static client:(nano ifcfg-eth0)

```
bash
BOOTPROTO=none
DEVICE=eht0
HWADDR=0a:67:42:8d:24:9e
ONBOOT=yes
TYPE=Ethernet
IPADDR=10.9.8.7
PREFIX=24
GATEWAY=10.9.8.1
DNS1=10.9.8.53
DNS2=8.8.8.8
DNS3=8.8.4.4
```

Restart the network with this:

```
systemctl restart network
```

NOTE: *A* **telnet** server was the progenitor of modern SSH. It allowed shell access across plain text connections, making it inherently extremely insecure, so its use is actively discouraged today. But it's an easy way to test network connectivity. If you choose to use this tool, you should forcibly disable it when you're not actively using it. And you should not use it at all on a machine connected directly to the internet, unless you like being a spam relay.

Flascards:

(Ubuntu)

1. - What run on system

sudo systemctrl

2.- Install telnet and xinetd if not listed

sudo apt install <telnet|xinetd>

3.- Check xinetd status

sudo /etc/init.d/xinetd status

4.- Check which one have in service

less /etc/services

5. - Turn on xinetd

sudo systemctl enable xinetd

6. check whether telenet is running

chkconfig telnet on/off

(CentOS)

1. Install telnet

sudo yum -y install telnet-service

2. check whether inetd

chkconfig xinetd

3. check whether telenet is running

chkconfig telnet on/off

4. Turn on xinetd

sudo service xinetd start

Lecture:Implement Packet Filtering

The **iptables** tool puts you in control of which packets should be "let in," which should be "turned away," and which ones should just be ignored.

The **ping** tool allows you to send special packets to another host to see if that host can respond.

The **-c** flag is used to limit the number of requests that the tool sends. You may use a host IP address or host name.

Send three ping requests to host 10.9.8.7:

```
ping -c  3  10.9.8.7
```

Continue sending ping requests to host www.linuxacademy.com until manually canceled:

```
ping www.linuxacademy.com
```

List of current policies:

```
iptables -L
```

Remove current rule set:

```
iptables –flush ( iptables -F)
```

Change the current "forward" policy to "Accept" everything:

```
iptables -P FORWARD ACCEPT
```

Reject all ping requests sent to device eth0 with an error message:

```
iptables -A INPUT - - protocol icmp - -in-interface eth0 -j REJECT
```

To drop all ping requests (stealth mode):

```
iptables -A INPUT - - protocol icmp - -in-interface eth0 -j DROP
```

Lecture:Start, Stop, and Check the Status of Network Services

The **netstat** tool is a favorite of Linux administrators as it provides the process and port-level details about what is listening and communicating on your system, using what protocols.

Running netstat as root will give you additional information than it would if you're just a user.

netstat -a: Shows the current status of everything port on your system
netstat -at: Shows the current status of only TCP ports and connections
netstat -au: Does the same for UDP
netstat -l: Lists TCP listeners
netstat -lu: Lists UDP listeners
netstat -s: Will show you lots of cool statistics
netstat -tp: Will include PID numbers with active Internet connections

Show the kernel routing table:
netstat -r: (similar to ip route list)
netstat -ie: Will list all the network interfaces, similar to ipconfig
netstat -c: Will continuously monitor and update every few moments

Lecture:Statically Route IP Traffic

To view the kernel route table, you can make an **ip route list (route -n)**, or you can use the older method of **netstat -r**. This will show you which networks will be sent which traffic based on the destination host's IP address. A route table might look like this:

```
bash
10.9.185.0/24 dev eth0 proto kernel scope link src 10.9.185.148
10.9.185.1 dev eth0 proto dhcp scope link src 10.9.185.148 metric
100
```

1.LİNE: This basically says if a packet is on the network, 10.9.185.0/24 (that slash 24 at the end means it's got a netmask of 255.255.255.0) then that's the local network, and it doesn't need a gateway – everything is right here.
2.LİNE: The second line says "Okay, for everything else, send it to 10.9.185.1, because it needs routing. I could set up another network inside my home network at 10.9.185.0/24 and provide manual, static routing to it.

IP addresses on my host by typing:

```
sudo ip route add 8.8.8.0/16 proto static metric 10 *via inet 10.9.185.143 dev eth0
sudo ip route del 8.8.8.0/16 proto static metric 10 *via inet 10.9.185.143 dev eth0
```

Lecture:Synchronize Time Using Other Network Peers

NTP, or **network time protocol,** is a way for you to synchronize the time on your servers with the time on other servers. Setups like that are usually synchronized to the atomic clock or some other super-accurate clock.

Most of what you need to do to get your time synchronized is:

1. Install ntpd

2. Configure ntpd

Install ntpd by using the package manager on your distribution on the package "ntp" – it's probably already installed.

sudo apt install ntp (Ubuntu)
sudo yum -y install ntp (CentOS)

Edit /etc/ntp.conf to contain pointers to servers of your choosing by adding pool lines to the configuration file.

sudo nano /etc/ntp.conf

Flascards:

**For more chech date/time:
www.pool.ntp.org (select your region on local machine)

** Restarting NTP Time on linux servers:
sudo /etc/init.d/ntp restart

** check the date/time:
sudo date

**On CentOS

sudo yum -y install ntp

systemctl enable & start ntpd

sudo /etc/ntp.conf

systemctl restart ntpd
date

Service Configuration

Lecture:Configure a Caching DNS Server

In this lesson, we install and configure BIND9 on CentOS 7.

bash
yum -y install bind bind-utils
nano /etc/named.conf

Alter the file such that the options section with the line allow-query looks like this:
allow-query { localhost; any };

And add this line immediately following the line above:
allow-query-cache { localhost; any; }

Check the permissions on the file **/etc/named.conf**. It should be owned by **root** and be assigned to the **named** group. The permissions on it should be 640. In short, an **ls -la /etc/named.conf** should return something that looks mostly like this;

-rw-r—–. 1root named 1757 Apr 1 00:48 named.conf

ls -lZ should return:

-rw-r—–. root named system_u:object_r:named_conf_t:s0 named.conf

If not, then we need to correct its SELinux context by:

semanage fcontext -a -t named_conf_t:s0 /etc/named.conf

semanage fcontext -a -t named_conf_t:s0 /etc/named.rfc1912.zones

Continue the process:

bash
named-checkconf /etc/named.conf : to check syntax is correct run
systemctl restart named
systemctl enable named
systemctl status named

**If we have a problem to connect by firewall , we may need to open port 53 - usually not necessary

Lecture: Maintain a DNS Zone

Sample zone file and cheatsheet:

#add zone (nano /etc/named/named.conf)

```
zone 'la.local' in {
type master;
file "la.local.zone" ;
} ;
```

#Create zone file (nano la.local.zone)

$INCLUDE (if we want to or need to other files)

```
$ORIGIN=la.local.
$TTL=600  (#10minutes)

   @    IN       SOA       dns1. la.local          mail.la.local (
        1                  # Serial number - - update this every time
        21600;                    #Time to live in seconds - Refresh after 6 hours
        3600;              #Retry after one hour
        604800;             # Expires after a week
        86400              #Minimum time to live of a day
) ;

host-records  IN       A        1.1.1.1
are-all-of    IN       A        1.1.1.2
type-a        IN       A        1.1.1.3


host10                 IN       A       1.1.1.10
host11                 IN       A       1.1.1.11

web-server    IN       A        1.1.1.80
mail          IN       A        1.1.1.99
dns1          IN       A        1.1.1.53
dns2          IN       A        1.1.1.54

cname-records IN     CNAME        type-a
point-to      IN      CNAME       are-all-of
other-records IN     CNAME        host-records

www           IN     CNAME        web-server
              IN MX    10          mail.la.local
              IN NS   dns1. la.local
              IN NS   dns2.la.local
```

Lecture: Connect to Network Shares

On the server:

yum install nfs-utils (to install nfs-utils packages)
mkdir /share (to create share directory)
chmod -R 755 /share (to be sure that permission to directory)
chown nfsnobody:nfsgroup /share (to be sure that user and nobody name)

systemctl **enable** rpcbind
systemctl **enable** nfs-server
systemctl **enable** nfs-idmap (ENABLE and START some other services)

systemctl **start** rpcbind
systemctl **start** nfs-server
systemctl **start** nfs-idmap

On the server, configure the share and the client (you will need IP addresses for all clients you wish to grant access to):

Edit the **/etc/exports** file to look like this: (nano /etc/exports)

/share 172.31.96.178 (rw,sync,no_root_squash,no_all_squash) ***rw: read-write

Restart the server to pick up the changes to the configuration files.
systemctl restart nfs-server

****To connect Client-server: ssh user@ 172.31.96.178

On the client:
sudo su
yum install nfs-utils (to create nfs-utils)
mkdir -p /mnt/remote (to create mount directory that is name remote)
mount -t nfs 172.31.124.130:/share /mnt/remote (do a mount,give a type)*** df -h to check mount active

Test and verify, on client:
echo "Test File" > /mnt/remote/testfile.txt
cat /mnt/remote/testfile.txt

Test and verify, on server:
cat /test.txt

**NFS: Network File System

Lecture:Configure Email Aliases

Using Postfix as the email server, we create a new file in **/etc/postfix** called **aliases.**

** cd /etc/postfix
**ls -la

**sudo nano aliases

Send webmaster's email to the user named chad:

webmaster: chad

Send terry's email to both terry and boss:

terry: terry, boss

And so the whole file would read, simply:

webmaster: chad
terry: terry, boss

You must tell Postfix about the aliases changes:

sudo postalias /etc/postfix/aliases

Lecture:Configure SSH Servers and Clients

apt-get install openssh-server (To create install open-ssh server)

Make desired configuration changes to the config file /etc/ssh/sshd_config.

To generate a key-pair:

ssh-keygen

This will create two files. A private key file which should be kept to yourself as you would a password. This file is called, by default, **/.ssh/id_rsa.** There is also a matching public file, which is similar to a username in that servers you will be logging in to will all have a copy, and it will be associated with your account.
This file is called, by default **/.ssh/id_rsa.pub.**

Copy your public key file to the server you wish to log in to without typing a password:

ssh-copy-id user@chadrm6.mylabserver.com

**cd authorized_keys (that is public key means everyone can access the key)

(to edit your key by manually) : Copy the key from the clipboard by using
cat id_rsa.pub command...

And after use the
cat ~/.ssh/id_rsa.pub | ssh user@chadrm6.mylabserver.com "mkdir -p ~/.ssh&&cat >> ~/.ssh/authorized_keys"

Lecture:Restrict Access to HTTP Proxy Servers

Using Squid as the proxy server, we edit the configuration file **/etc/squid/squid.conf**.

**cd /etc/squid
**ls -la
**sudo nano squid.conf

For a single host, add a line like this to the file:
acl hosttodeny(nochad) src 10.9.8.200

For a whole network, add a line like this to the file:
acl nettodeny(nochad) src 10.10.10.0/24

Then alter the line that says:
http_access allow localnet

To deny the appropriate acls:
http_access allow localnet !hosttodeny(nochad)
http_access allow localnet !nettodeny(nochad)

Lecture:Configure an IMAP and IMAPS Service (Including POP3 and POP3S)

We're going to use Dovecot as our IMAP, IMAPS, POP3 and POP3s servers.

sudo apt install dovecot-core (to create Dovecot on Ubuntu)

cd /etc/dovecot

cd/private (is a private directory which is a place we could put our pkI certificate)

cat 10-mail.conf | grep mail_location # make a note of this value for later.

cat 10-mail.conf | grep mail_privileged_group # make a note of this value for later.

Verify that the location of the **mail_location** exists.

Edit the file **/etc/dovecot/dovecot.conf.**

Find the line protocols = and add the protocol names so it looks like:

protocols = imap pop3 imaps pop3s

Verify that certificates have been created and placed in **/etc/dovecot/conf.d** directory or in the **/etc/ pki/dovecot** directory, depending on your distribution.

Verify that the location of the certificates is identical to the location cited in **/etc/dovecot/10-ssl.conf.**

Sudo systemctl restart dovecot

Verify:

sudo ps aux | grep dove

sudo netstat -nptlu | grep dove

We should be listening on ports 110, 143, 993, and 995.

Lecture:Configure an HTTP Server (RHEL/CentOS)

As root: (su - root)

yum install httpd (to install the httpd on CentOS)

systemctl enable httpd (to enable httpd)

systemctl start httpd (to start httpd)

(((*** **lynx localhost:** We check the WebServer is running or not))

curl http://localhost # should get the Apache2 test page

To enable Ubuntu/Debian-like virtual hosts in separate files, which is extremely convenient, edit the /etc/httpd/httpd.conf **((cd /etc/httpd>> cd httpd.conf>>nano httpd.conf))** file. At the very bottom of the file ((dosyanın en altına)), add this line:

IncludeOptional vhost.d/*.conf

Create the new directory:

mkdir -p /etc/httpd/vhost.d (httpd dosyasının içine yeni bir vhost klasörü ekledik)

Restart the service:

systemctl restart httpd

Check teh service:

systemctl status httpd (check the status for httpd)

Lecture:Configure an HTTP Server (Ubuntu/Debian)

As root: (su - root)

apt-get install apache2 (to install apache2(like httpd on centhos))

lynx localhost (to test the webServer is running or not)

Configuration files are located in **/etc/apache2** instead of where they are on CentOS/Redhat machines. The main configuration file is**/etc/apache2/apache2.conf**, though it's been split up into multiple files for ease of management.

bash It is split into several files forming the configuration hierarchy outlined below, all located in the /etc/apache2/ directory:

```
/etc/apache2/
        -- apache2.conf
                \-- ports.conf
        -- mods-enabled
                |-- *.load
                \-- *.conf
        -- conf-enabled
                \-- *.conf
\       -- sites-enabled
                \-- *.conf
```

!!!1NOTE !!!: **RHEL/CentOS => httpd**

Debian/Ubuntu => apache2

Lecture:Configure HTTP Server Log Files

Log format types are defined in **httpd.conf** on CentOS/RedHat systems and

apache2.conf on Debian/ Ubuntu systems in the **ifmodule log_config_module** section.

For any vhost, there are two primary log files for Apache: **error** and **access**. Both are in /var/log by default, but this can be customized.(/var/logs/apache2/access.log)

The error log's format is static and cannot be customized.

Access logs can be customized almost however you'd like. Two are already defined for us: **customlog** is **logs/access_log** combined meaning it's in this format:

	Meaning
h	hostname/ip address of remote requester
l	remote login name
u	remote user
t	date/time
r	first line of request to server
s	final status
b	size of response
	Legacy
	(browser)

Examine the logs as they are:
cat /etc/httpd/logs/access_log

---

cd httpd
cd conf **TO CREATE NEW LOG**
sudo nano httpd.conf

Creating a simplified log format called **mycustom**:
"host: %h - Date and Time: %t - Requested: %r" mycustom

Change the access_log definition to use the new **mycustom** format and save the file.

Then restart Apache:
systemctl restart httpd

If you made any mistakes in the configuration file, it will be revealed at this point. Generate some data with your new log:

lynx http://localhost
Verify the changes:
cat /etc/httpd/logs/access_log

Lecture:Restrict Access to a Web Page

In either the **apache2.conf** or **httpd.conf**, depending on which distribution you're running, find the section that contains <Directory entries.
<Directory> directive in /etc/apache2/apache2.conf

""sudo nano apache2.conf"" (Ubuntu)
sudo nano httpd.conf(CentOS)
Create one for a site you wish to hide, except from certain networks or IP addresses:

```
<Directory /var/www/html/test/>
        Order allow,deny
        Allow from 52.206.180.246 (public ip adrress the particular machine)
        Allow from 172.31.34.52    (private ip adrress the particular machine)
        Allow from 127             ( localhost)
</Directory>
```

Save the file and restart the web server:

systemctl restart httpd (RH/CentOS)
or
systemctl restart apache2 (Debian/Ubuntu)

In this example, the site at /var/www/html/test will be accessible from only the local machine, and from the IP addresses 52.206.180.246 and 172.31.34.52.

****2.sayfaya erişmek için;
create directive to deny access outside of local

```
Allow from 127
Allow from ::1
</Directory>
```

Lecture:Configure a Database Server

To install MariaDB, a MySQL drop-in replacement:

sudo apt-get install mariadb-server mariadb-client

mysql_secure_installation # CHANGE THE ROOT PASSWORD!

To execute SQL statements interactively:

mysql -u root -p

Some simple commands for reference:

- show databases;
- create database anythingyoucreate;
- show databases;
- use anythingyoucreate;

show tables;

systemctl status mariadb;

Lecture:Manage and Configure Containers

A great place to find docker images is the Docker Hub. To install Docker, install

Docker.io using either apt or yum.

List of running containers:
docker ps

Create a containerized web server that will serve content out of /home/user/webstuff:

********docker run -dit --name la-test-web -p 80:80 -v /home/user/ webstuff/:/usr/local/apache2/htdocs/ httpd:2.4****** (creat webserver via Docker)

Pull up the host's ip address in a web browser, and you should see your content.
Add additional content. The directory is shared between the host and the container.
Stop and start the container (Verify in a browser between commands).

docker stop la-test-web

docker start la-test-web

List local docker images:

docker image ls

Remove an image:

docker image rm httpd:2.4

Lecture:Manage and Configure Virtual Machines

On a Centos 7 machine:

yum install qemu-kvm libvirt libvirt-client virt-install virtviewer #to install key move

**this packages are already installed by defalut.but we must to make sure using by command.

cat /proc/cpuinfo |grep vmx # grep for svm if you're on an AMD system

virt-install --name=tinyalpine --vcpus=1 --memory=1024 --cdrom=/tmp/ image.iso --disk size=5

virsh list --all # will list all running virtual machines on cli

virsh edit tinyalpine # will open the XML defining the running VM to alter in vi

virsh autostart tinyalpine #will autostart this vm on boot

virsh autostart --disable virtly # will disable the above option

vmmanager #will launch the graphic version of Virtual machine Manager

virt-clone --original=tinyalpine --name=tiny2 –file=/path/to/ mynewdisk/tinyalpine2.qcow2

***tinyalpine is just the name that we created new virtual machine.

Storage Management

Lecture:List, Create, Delete, and Modify Physical Storage Partitions

The **lsblk** tool lists all block devices and partitions on the system, along with their size, current mount status, and mount point.

The **fdisk** tool allows you to format block storage devices.
fdisk /dev/sdx # or whatever block device you want to partition.

Fdisk Menu Options:

Command	Action	Notes
a	Toggle bootable flag	Tells older non-UEFI systems that the partition contains a bootable operating system
b	Edit bsd label	
c	Toggle the dos compatibility flag	
d	Delete a partition	**Danger!** If you accidentally delete a partition, hit q and start over. It doesn't get written to disk immediately, so if you get out now, then you avoid nuking your system.
g	Create a new empty GPT partition table	
G	Create an IRIX (SGI) partition table	
l	List known partition types	
m	Print this menu	
n	Add a new partition	
o	Create a new empty DOS partition table	
p	Print the partition table	
q	Quit without saving changes	

s	Create a new empty Sun disklabel	
t	Change a partition's system id	
u	Change display/entry units	
v	Verify the partition table	
w	Write table and exit	
x	Extra functionality (experts only)	

Flascards:

1.) Which disk that wouldnt be displayed when using "lsblk"command? --RAM disks
2.) What number of partition you can assign to when you create a new partition in GPT disk label type ?? --- 128
3.) -o ///Create a new empty DOS partition table ; creat a new partittion: g,n

Lecture:Manage and Configure LVM Storage

The lvm2 package allows you to take several physical block devices and bind them together and present them to the system as a single device mountable at a single

mount point while using all the drive space.
**sudo yum install lvm2
**sudo apt install lvm2

When you use **fdisk** to partition your devices, make sure you change the partition type to 8e, which is the Linux LVM file type.

COMMANDS LINES:
(!!)
**lsblk
**fdisk /dev/xvdf
**n (to create new partition)
**p(will be a primary one)
** Press Enter (starting default number 1 for partition number)
** Press Enter (to starting default first sector 2048)
** +300M (the size of last sector)
**p (print the table)
** l (for list and find Linux LVM (8e))
** t (to change to file type)
**8e
**p (print the table)
**w (write table and exit)

**pwcreate /dev/xvdf1 /dev/xvdf2 (Create physical Volumes)
**vgcreate tinydata[#name of group] /dev/xvdf1 /dev/xvdf2 (Create volume group)
** lvcreate --name logical-tiny --size 590M tinydata
**lvdisplay (show us detail of logical volume)

**mkfs -t ext4 /dev/tinydata/logical-tiny (make a new file system)

**cd /mnt
**ls -la
**mkdir teeny (create new mount point – mnt -)
**mount /dev/tinydata/logical-tiny /mnt/teeny
**cd teeny
*ls -la

*df -h (to detail file system)

Now we create a file then add to this file system.
** nano testfile.txt (teeny director içindeyiz)

we create !! commands lines again.
Then we reboot system.

The hierarchy of pieces you'll be using to assemble something you can mount is:
Partition > Physical Volume > Logical Volume > Volume Group > Filesystem

Object	Create Tool	List Tool	Detail Tool
Partition	fdisk	fdisk	fdisk
Physical Volume	pvcreate	pvs	pvdisplay
Logical Volume	lvcreate	lvs	lvdisplay
Volume Group	vgcreate	vgs	vgdisplay
Filesystem	mkfs	lsblk	df

**pvs: physical volum list
**vgs: volume group list
**lvs: logical volume list

To add a device to an existing LVM set:

1. **Back up your existing volume group and unmount.**

2. Partition your new device. Make sure the partition type is 8e.

3. Create a new physical volume in the new partition with pvcreate (/dev/xvdf3)

4. Extend the volume group with the new partition using vgextend (tinydata /dev/xvdf3)

5. Extend the logical volume with the new with lvextend (-l [FREE SIZE] +105 */dev/tinydata/logical-tiny*)

6. Check the existing file system with e2fsck (-f /dev/tinydata/logical-tiny)

7. Resize the file system using resize2fs (/dev/tinydata/logical-tiny)

8. Remount the volume group.

Lecture:Create and Configure Encrypted Storage

To install and set up your first encrypted partition:

grep -i config_dm_crypt /boot/config-$(uname -r)

lsmod | grep dm_crypt #We need to be sure our system supports encrypted file systems!

---

yum install cryptsetup

cryptsetup -y luksFormat /dev/xvdf1
enter a passphrase and verify

cryptsetup luksOpen /dev/xvdf1 mySecret

mkfs -t ext4 /dev/mapper/mySecret

cd /mnt
mkdir encrypted

mount /dev/mapper/mySecret /mnt/encrypted/

Next, write a file to test that everything works:

echo "encryption of this file" > /mnt/encrypted/encryptedfile
df -h
Lock it back up:
umount /mnt/encrypted
cryptsetup luksClose mySecret

To lock up the partition when not in use:
cryptsetup luksClose mySecret
This protects it from being mounted without the passphrase;
To unlock it:
cryptsetup luksOpen /dev/xvdf1 mySecret
mount /dev/mapper/mySecret /mnt/encrypted/
Lock it back up: (we will test mounting the drive without unlocking so we locked up files again.)
umount /mnt/encrypted
cryptsetup luksClose mySecret

Test mounting the drive without unlocking first:
mount /dev/xvdf1 /mnt/encrypted

We can't mount it unless we use the encryption system. Our passphrase protects it.

Lecture:Configure Systems to Mount File Systems at or During Boot

This is all controlled from the **/etc/fstab** file. Like many other Linux configuration files, this one is arranged such that each line in the file refers to a single device.

(fstab: file system table)
device /mnt/point/ fs-type options 0 2

The last number is a 0, 1, or 2 depending on what you want the **fsck** at boot behavior to be: 0 is don't do it, 1 is this is the root drive, so this gets the primary check, and 2 checks it after the primary drives.

To create a new entry in this file, you'll need its **UUID.**
You can get this with the
**blkid

Copy the uuids of ext4 and btrfs.
**Sudo nano /etc/fstab

UUID=xxxxxxx /mnt/ext4 ext4 defaults 0 2
UUID=xxxxxxx /mnt/btrfs btrfs defaults 0 2

fstab option	**Meaning**
default	Use rw, suid, dev, exec, auto, nouser, and async as the options
relatime	Access times are only updated if they are earlier than the modification time
noatime	Turns off tracking of access times
user	Permit users to mount the file system (imples noexec, nosuid, nodev unless you override these)
noauto	Do not mount when mount -a is given
exec / noexec	Permit or block the execution of binaries from the file system
suid / nosuid	Permit of blocking the operation of suid and sgid bits
nobootwait	Mounts the device, but the rest of the boot process does not wait for it to complete
nofail	Do not report errors for this device if it doesn't exist. Particularly useful for usb drives which may or may not be plugged in.
credentials=/path/to/file	Used to pass a username and password to cifs network partitions
usrquota	Enable quota on this device (quota package must be installed and configured)

Lecture:Configure and Manage Swap Space

Swap is used when a system needs more RAM but is using all its RAM. It swaps out pages to swap space – either a partition or a file. Partitions should be type 82, Linux

swap.

Turn all your swap options off with swapoff -a and back on again with swapon -a.

Swap is defined for the system in the fstab like this:
/dev/sd3 swap swap sw 0 0

You can also create a 512M swap file using the dd tool like this:
dd if=/dev/zero of=/path/to/extraswap bs=1024 count=523288

The new file should be owned by root and have permissions of 600. Start using it by using the swapon command.

swapon /path/to/extraswap

You can add this file to be mounted as boot time, as well, just by specifying the file as the device.
/path/to/extraswap swap swap sw 0 0

Flascards:

**swap space is used when RAM is full. Move RAM to disk. Swap space is used when the kernel needs more RAM, but there is no more RAM. It "swaps" pages of memory out to disk in either a swap file or swap partition.

**swap is located : nano /etc/fstab

Lecture: Create and Manage RAID Devices

*** we created new 450 M 2 partitions, then we change the type that is fd.

The **mdadm** tool, or **Multi Disk Admin** tool, allows you to manage software RAID on Linux.

Partitions that will be used as part of a RAID array should have a type of fd.

RAID Type	Meaning
0	Many disks, no parity. For assembling disks into a single unit, though LVM might be a better choice if this is your goal
1	Also called Mirroring. All files on at least two drives
5	Drives with parity, for file and backup servers with high availability requirements
6	Drives with double parity
Others	There are more RAID types than we discussed in the video. See the man pages for details!

To create a RAID0 array from two partitions:

mdadm --create --verbose /dev/md0 --level=stripe --raid-devices=2/dev/xvdf1 /dev/xvdf2

cat /proc/mdstat (#we check md status)

mdadm --detail /dev/md0 (# we can also get more information about detail)

mkfs -t ext4 /dev/md0 (# we creat a new file system and assign md0)

mount /dev/md0 *mnt* (# we mounted that to /mnt)

mdadm --detail –scan (# it gives us info the array we created)

[# we copied that array line and add it to nano /etc/mdadm.conf)
#(add this line to /etc/mdadm/mdadm.conf on Ubuntu/Debian
or /etc/mdadm.conf on CentOS/RedHat)]

mdadm --assemble –scan

update-rc.d mdadm defaults

#On Ubuntu:
nano /etc/default/mdadm (# we also wanted automatically start)
set AUTOSTART=true

On CentOS/Redhat:
systemctl enable mdmonitor

systemctl start mdmonitor

To check on your RAID array:

mdadm –detail /dev/md0

If a device fails, then add a new device to have the raid array go into self-repair mode (if you've chosen other than RAID0):

mdadm /dev/md0 --add /dev/NEWDEVX

Configure Systems to Mount File Systems on Demand

One video outlines connecting to a samba share.

Install utilities:

yum install samba-client samba-common cifs-utils
on Debian/Ubuntu systems, use apt-get
get available shares (omit -I and the IP address if you're on a
private network)

smbclient -I x.x.x.x[Static IPs] -U <username> -L share

mkdir samba
mkdir -p /mnt/sambacredentials
create a credentials file with YOUR Samba credentials

echo "username=mustafa" > /mnt/.smbcredentials

echo "password=sbtwef24" >> /mnt/.smbcredentials

chmod 600 /mnt/.smbcredentials

Edit the **nano /etc/fstab** file – use the IP address, **not the hostname!**
//x.x.x.x[StaticIPs]/la-share /mnt/samba cifs credentials=/mnt/. smbcredentials,defaults 0 0

Then, to mount the share
mount -a [/mnt directory içinde]

or

mount -t cifs -o credentials=/mnt/.smbcredentials,defaults //x.x.x.x/share /mnt/newmountpoint

Setup User and Group Disk Quotas for Filesystems

Disk quotas are managed with the **quota** tool. Install the quota package with your system's package manager. (sudo apt install quota)

Edit the **nano/etc/fstab** to enable quotas. Add the keyword usrquota to the options section.

to enable quotas, change the line above to the line below
/dev/sda1 /opt ext4 defaults,usrquota 0 2

Once that's done, you'll need to remount the device (or reboot the system):
mount -o remount /dev/sda1

Create the database (note this will remove any old databases along with any information they contained):

quotacheck -cu /opt # Note the mount point is the same one for the one we're using Optional **m** and **g** flags also

Check the database with the a (all) and v (verbose) and u (user) options:
quotacheck -avu

Edit quotas with the command:
edquota <username or group>

Quotas are in blocks, not bytes. You can get the block size of your partition's filesystem with the command:
blockdev –getbsz /dev/sda1

So, back to editing the file. You'll see something like this:

Disk quotas for user chad (uid 401):

Filesystem	blocks	soft	hard	inodes	soft	hard
/dev/sda1	24	20000000	25000000	0	8	0

Save the file and then run **quota** to verify that things worked out like you'd hoped.

You can run a quota report to see how things are going with:
repquota -a

Create and Configure File Systems

The **mkfs** command will create filesystems. In other operating systems, this is referred to as formatting.

To make a filesystem, you'll usually need a partition on a block device, so use **fdisk** to set that up. Some newer file systems don't strictly need a partition table, so consult your file system of choice's man pages to find out precisely what you need.

To use it:

```
mkfs -t ext4 /dev/xvdf1
```

Once you've created the file system, you can mount it and add it to your system's **/etc/fstab** so it gets mounted at boot time.

Flascards:

- Create File System

```
mkfs -t ext4 /dev/<partition>
mkfs -t btrfs/dev/<partition2>
```

- Configure FS: Go to /mnt to mount File System

```
cd /mnt
sudo mkdir ext4
sudo mkdir btrfs
sudo  mount /dev/<partition1> /mnt/ext4
sudo  mount /dev/<partition2> /mnt/btrfs
```