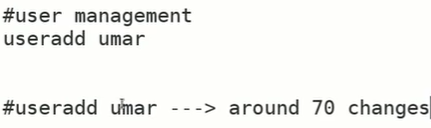
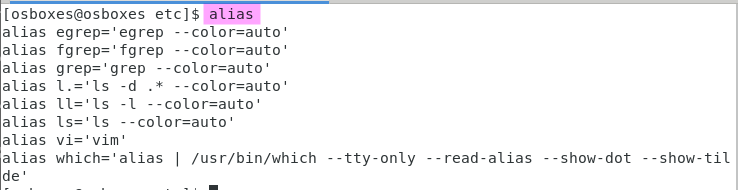
**Lecture 23**

**User-Management**



* Both **useradd** and **adduser** commands are used to create new user accounts on Linux and Unix systems. However, the commands are not entirely interchangeable and have some differences in their behavior and syntax.
* **useradd** is the low-level command to add a new user account to the system, and it requires the user to manually specify all of the options needed to create a new user account, including the user's home directory, shell, and group membership. **useradd** is typically used by system administrators who are familiar with the Linux user and group management system.
* **adduser** is a high-level command that is designed to simplify the process of adding a new user account by prompting the user for all of the necessary information. **adduser** is easier to use for most people because it provides a more user-friendly interface and automatically creates a home directory for the new user.
* In terms of which command is "valid," both commands are valid, and it depends on your system and your preferences which command to use. Some Linux distributions, such as Debian and Ubuntu, have **adduser** installed by default and recommend using **adduser** instead of **useradd**. Other distributions may have **useradd** installed by default or may have both commands available.
* In summary, if you are comfortable with the Linux user and group management system, you can use **useradd**. Otherwise, if you prefer a more user-friendly interface, you can use **adduser**.
* How to check a user 🡪 id <user>
* $ passwd <user> 🡪 to change password.
* As soon as password is set 🡪 the valid user is created
* $ adduser is actual command 🡪 “useradd ” is actually symlink
* 🡪 symlink is being created
* 
* Another this is “aliasing”
* 
* 
* It is temporary and only set in RAM
* $ alias 🡪 check alias list
* 
*  🡪 to remove from alias
* To make it persistent 🡪 
* Sentry can be listed in any boot file

To set an alias persistent, you can add the alias to your shell configuration file. The location and name of the shell configuration file varies depending on the shell you are using. Here are some common locations and file names:

* Bash: ~/.bashrc
* Zsh: ~/.zshrc
* Fish: ~/.config/fish/config.fish
* PowerShell: $PROFILE.CurrentUserAllHosts

To add an alias to your shell configuration file, follow these steps:

1. Open your shell configuration file in a text editor.

For example, to open the Bash shell configuration file in the nano text editor, you would run the following command:

* 
* Add the alias to the file.

For example, to create the alias "ll" for the command "ls -l", you would add the following line to the file:

* 
* Save the file and exit the text editor.
* Reload the shell configuration file to apply the changes.

For example, to reload the Bash shell configuration file, you would run the following command:

* 
* After these steps, your alias will be persistent and available every time you open a new shell session.
* **Tip**

In computing, the tilde character (~) is often used as a shorthand to represent the current user's home directory.

On Unix-like systems (such as Linux, macOS, and other Unix variants), each user has a home directory where their files and personal settings are stored. The home directory is usually located in the /home directory and has the same name as the user's login name. For example, the home directory for a user named "john" would be /home/john.

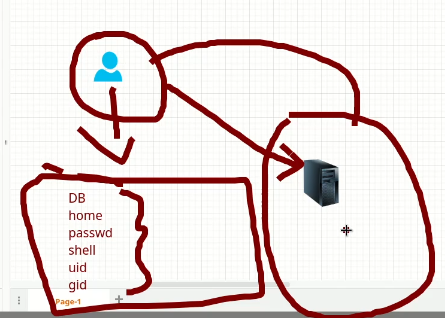
Using the tilde character followed by a username or nothing after it, allows you to reference a specific user's home directory or your own home directory, respectively. For example:

* "~" (tilde without a username): represents the current user's home directory. For example, "cd ~" would change the current directory to the current user's home directory.
* "~username": represents the home directory of the specified user. For example, "cd ~john" would change the current directory to the home directory of the user named "john".

The tilde character is also used in various other contexts in computing, such as in regular expressions and in certain programming languages.

* Bad practice to used alias on a working command 

**User management**

* How “useradd” works
* 
* If a user is to be added it must have a database which covers all essential information as sown in above picture.
* “/etc/login.defs”
* “etc/default/useradd”
  + These two files contain information about database of the user
  + Which home directory is to be assigned to the user
  + Its password
  + Which shell is to be given
  + Uid
  + Gid
* Whenever “useradd” command is executed,
  + It refers to the above mentioned 2 files
  + And it updates 4 files, 🡪 these 4 files are called “user database” of Linux
    - /etc/passwd
    - /etc/shadow
    - /etc/group
    - /etc/gshadow
    - .

The **useradd** command is a standard Linux command used to create a new user account. When you run the **useradd** command, it adds a new user to the system by creating a new user account with a specified username and user ID (UID).

Here is a brief overview of how the **useradd** command works:

1. Syntax: The basic syntax of the **useradd** command is as follows:

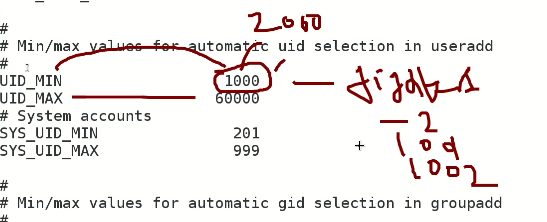
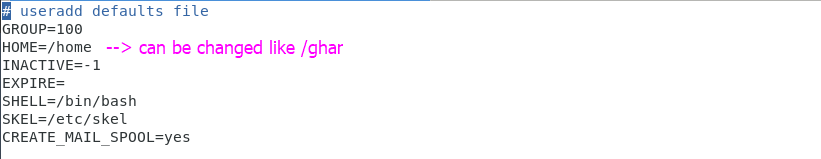
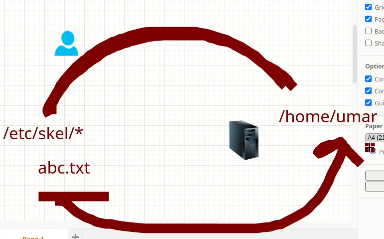
**useradd [options] username**

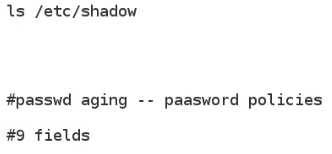
The **options** are optional flags that you can use to customize the user account settings.

1. Creating a new user account: When you run the **useradd** command, it creates a new user account by adding an entry for the user to the system's user database (usually **/etc/passwd** file) and assigning a UID to the user.
2. Home directory creation: By default, the **useradd** command creates a home directory for the user in the **/home** directory, with the same name as the username. The **--home** option can be used to specify a different home directory.
3. Default shell and login group: The **useradd** command sets the default shell (usually **/bin/bash**) for the new user and assigns the user to a login group (usually a group with the same name as the user).
4. Other options: The **useradd** command also provides a number of other options that you can use to customize the user account settings, such as setting the user's password, specifying the user's full name, or setting the expiration date for the account.
5. Additional steps: After creating the user, you may need to perform additional steps to grant the user permissions or access to various resources on the system. For example, you may need to create a password for the user, add the user to additional groups, or configure access control settings.

Overall, the **useradd** command is a powerful and flexible tool for creating new user accounts on a Linux system.

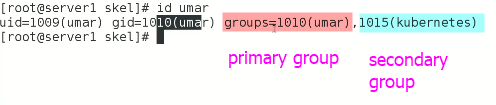
* Top of Form
  + - 
    -  🡪 profile files to home directory
* **Lets look at the 2 files that are referred at the time of creation of a user.**

1. /etc/login.def -🡪 can be customized 🡪 works like a template
   * 
   *  🡪 if yes is changed with no 🡪 user’s home directory will not be created
   *  🡪 the home directory’s permission “700” is because of this value of UMASK
   * 
   * 
2. /etc/default/useradd
   *  or can be changes to /opt
   * 
   * Because all files in /etc/skel are copied to /home directory 🡪 if admin want to give a specific file to a user in /home directory 🡪 that file can be created in “/etc/skel”
   * Use “l. instead of ls” to list all files in this directory.

* **Now look at the other 4 files that are accessed at the time of creation of a user**
* **/etc/passwd -> an important database file of a <user>**
  + It has 7 fields.
  + 
  + 
  + **Id <user\_name> 🡪 to check if the user is created or not. Or refer to “/etc/pwd”**
  + **$ grep <user> /etc/pwd**
  + **$ getent passwd <user>**
  + **$ compgen -u 🡪 displayes first field of /etc/passwd which is users**
  + **$ userdel <user>** 🡪 to delete a user or to undo useradd actions for a specific user 🡪 but **home directory will not be removed**
  + To remove home directory too 🡪 $ userdel -r <user>
* **Tip: how to validate a user** 🡪 **$ su - <user\_name>**
* **/etc/shadow** 
  + 

1. **Username:** This is the name of the user account.
2. **Password:** This field stores the user's encrypted password. The actual password is not stored in this field, but rather a hash value of the password that is generated using a one-way encryption algorithm.
3. **Last password change:** This field stores the date when the user last changed their password. The date is represented in the number of days since January 1, 1970.
4. **Minimum password age:** This field specifies the minimum number of days that must pass before the user is allowed to change their password again.
5. **Maximum password age:** This field specifies the maximum number of days that a password is valid. After this period, the user must change their password.
6. **Password warning period:** This field specifies the number of days before a password expires that the user will be warned to change their password.
7. **Password inactivity period:** This field specifies the number of days that a user is allowed to have an inactive account before it is locked.
8. **Account expiration date:** This field specifies the date when the user account will be automatically locked and inaccessible.
9. **Reserved field:** This field is reserved for future use and is currently not used.
10. **Additional options:** This field stores additional options for the user account, such as account locking or password hashing algorithms.

**Groups**

* + Shortcut to go to files instead of absolute path 
  +  🡪 to add a group
  + for group information 🡪 group information is stored in /etc/group
  + **How to add a user into a group?**
    - * $ usermod-G <group\_name> <user\_to \_be\_added> 🡪
      * **-G means secondary group**
      * **-g means primary group** 
        + $ usermod -G kubernets <user>
      * 
      * the primary group is the main group of a user and is used for administrative purposes, while the secondary group is an additional group that a user can belong to and is used to grant access to files and directories owned by a specific group.
* 
* How to modify a user

In Linux, you can modify a user's account settings, such as their username, password, home directory, shell, and group membership using the **usermod** command. Here's how to modify a user:

* Open a terminal window.
* Type the following command to modify the user's account settings:
* 

Replace **username** with the name of the user you want to modify.

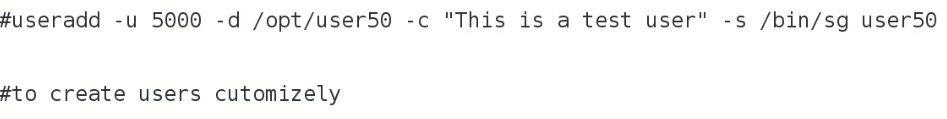
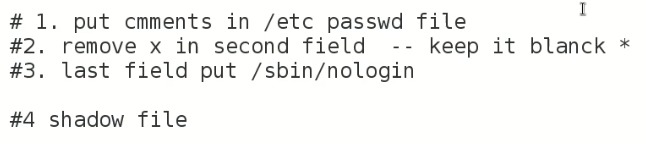
1. Specify the options to modify the user's account settings. Here are some common options:

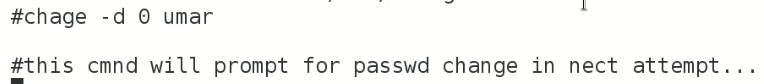
* **-l new\_username**: Change the user's username to **new\_username**.
* **-p new\_password**: Set the user's password to **new\_password**. Note that the password should be encrypted using the **crypt** function.
* **-d new\_home\_dir**: Change the user's home directory to **new\_home\_dir**.
* **-s new\_shell**: Change the user's default shell to **new\_shell**.
* **-g new\_group**: Change the user's primary group to **new\_group**.
* **-aG additional\_group**: Add the user to an additional group.

For example, to change the username of the user "jdoe" to "johndoe", you can use the following command:

* 
* Press Enter to run the command. If the command is successful, you should see no output.

Note that some options, such as **-l** and **-p**, require root privileges, so you may need to use **sudo** to run the command. Also, be careful when modifying user accounts as it can affect the user's ability to access the system or their files.

* 
* 
* 
* 
* **How to lock a user**
* Diagram

  Description automatically generated with medium confidence
* To unlock
* 
* **To disable all users**
*  🡪 that would be an empty file.
* 
* Task
* 
* ***Few interview questions related to groups.***

Sure, here are the answers to the questions related to primary and secondary groups in Linux:

1. What is the difference between a primary and a secondary group?

A primary group is the default group assigned to a user when the user account is created. This group determines the ownership of the files and directories created by the user. A secondary group, on the other hand, is an additional group that a user can be a member of. The secondary group allows a user to access resources owned by that group.

1. How do you change a user's primary group?

The primary group of a user can be changed using the **usermod** command with the **-g** option. For example, to change the primary group of the user "jdoe" to "sales", you can use the following command:

**sudo usermod -g sales jdoe**

1. How do you add a user to a secondary group?

To add a user to a secondary group, you can use the **usermod** command with the **-aG** option. For example, to add the user "jdoe" to the "developers" group, you can use the following command:

**sudo usermod -aG developers jdoe**

1. How do you remove a user from a secondary group?

To remove a user from a secondary group, you can use the **gpasswd** command with the **-d** option. For example, to remove the user "jdoe" from the "developers" group, you can use the following command:

**sudo gpasswd -d jdoe developers**

1. What is the command to create a new group in Linux?

The command to create a new group in Linux is **groupadd**. For example, to create a new group named "sales", you can use the following command:

**sudo groupadd sales**

1. What is the purpose of the group ID (GID) in Linux?

The group ID (GID) is a unique numerical identifier assigned to a group. It is used to identify the group in the system and to determine the access rights of the group to files and directories.

1. What is the difference between the owner and the group permissions of a file or directory?

The owner permissions control the access rights of the user who owns the file or directory. The group permissions control the access rights of the members of the group that owns the file or directory.

1. How do you change the group ownership of a file or directory?

To change the group ownership of a file or directory, you can use the **chgrp** command. For example, to change the group ownership of a file named "file.txt" to the "developers" group, you can use the following command:

**sudo chgrp developers file.txt**

1. What is the difference between the setuid and setgid permissions in Linux?

The setuid permission allows a user to execute a file with the permissions of the owner of the file. The setgid permission allows a user to create a file with the group ownership of the directory in which the file is created.

1. How do you set the default group for new files created by a user?

To set the default group for new files created by a user, you can change the user's primary group to the desired group. Any new files created by the user will then have the group ownership of the primary group. You can use the **usermod** command to change the primary group of the user, as described in question 2 above.

* All commands used in this lecture
* #useradd
* to add user
* #passwd username
* to set passwd for specific user
* #id username
* to check id of specific user
* #adduser
* same as useradd command
* #alias hello=ls
* alias create shortcut name for any command
* eg if you type hello it will act as ls command.
* But the alias command are updated in RAM it will remove when the system is reboot
* in order to make it permanent you should add it in some file which the system read during its booting
* # vi /etc/bashrc
* you should add the alias command in the bashrc command to make it permanent.
* #/etc/login.defs
* #/etc/default/useradd
* These two files are the database of the user.
* useradd command refer the above two files and the information get from these two files and update the below 4 files.
* #etc/passwd
* #etc/shadow
* #etc/group
* #etc/gsshadow
* These files are called linux user database.
* When user login it validates these 4 files.
* #/etc/passwd
* This has 7 fields
* 1  username
* 2  passward (which is in shadow file)
* 3 Uid
* 4 Gid
* 5  gecos file
* 6  homdir
* 7 shell
* #userdel username
* to delete a user
* # userdel -r username
* to delete a user and it data as well
* This command delete user form home directory also
* #cat /etcshadow
* this file has 9 fields
* All the files detail are show when you type
* #man shadow
* This command will show details of all 9 fields.
* shorcuts:
* #vipw
* is shortcut for cat /etc/passwd
* #vigr
* is shortcut for cat /etc/group
* #groupadd  groupname
* To add group in your system
* #usermod -G groupname username
* to add certain user in certain group we run the above command.
* #usermod -g groupname username
* to add certain user in certain group.
* -G is secondary group
* -g is primary group
* The difference between primary and secondary group is when user makes a files it shows primary group in details.
* Q Ask question in interview about primary and secondary group.
* # usermod -u 5000 -d /home/username -c "This is user" -s /bin/sh username
* To create user customizely.
* We also add a user by using useradd in place of usermod in above commands.
* Q How to disable user in linux
* 1 put comments in etc/passwd file
* 2 remove x in second field or keep it blank.
* 3 last field put /sbin/noloing
* #passwd -l username
* To lock user
* #passwd -u username
* to unlock user