Topics to be covered:

Week	Broader Topic	Topics	Tools to be covered
5	User Account Management	1. User Account Management: useradd 2. User Account Management: groupadd 3. User Account Management: usermod 4. User Account Management: userdel 5. User Account Management: groupdel 6. User Account Management: Switch users and sudo access (su, sudo) 7. User Account Management: Monitor users (who, last, w, id) 8. System Utility Commands: date, uptime, hostname, uname, which, cal, bc.	Linux Terminal

1. User Account Management: useradd

The `useradd` command in Linux is used for creating new user accounts. It allows you to specify various user properties and settings during the user creation process. Here's an overview of the `useradd` command with suitable examples:

Syntax: `useradd [options] username`

Some commonly used options with the `useradd` command are:

1. `-c` or `--comment`: Specifies a comment or description for the user.
2. `-d` or `--home`: Sets the home directory for the user.
3. `-m` or `--create-home`: Creates the user's home directory if it does not exist.
4. `-g` or `--gid`: Sets the primary group for the user.
5. `-G` or `--groups`: Specifies secondary groups for the user.
6. `-s` or `--shell`: Sets the login shell for the user.
7. `-p` or `--password`: Provides an encrypted password for the user (useful with the `-p` option).
8. `-e` or `--expiredate`: Sets an expiration date for the user account.

Now, let's look at some examples illustrating the usage of the `useradd` command:

1. Create a user with default settings:

```
useradd roy
```

This command creates a user named "john" with default settings. The user's home directory will be `/home/john`, and the login shell will be set according to the default system settings.

2. Create a user with a specific home directory and comment:

```
useradd -d /data/users/jane -c "Jane Doe" jane
```

This command creates a user named "jane" with a home directory set to `/data/users/jane`. The comment field is set as "Jane Doe" to provide additional information about the user.

3. Create a user with a specific primary group and secondary groups:

```
useradd -g staff -G developers,admins john
```

This command creates a user named "john" and assigns them to the primary group "staff". Additionally, the user is added to the "developers" and "admins" secondary groups.

4. Create a user with a predefined password:

```
useradd -p '$6$wesfTqfa$sNMXZCAkD4QoP0...' john
```

This command creates a user named "john" with a predefined encrypted password. The `-p` option is used to specify the hashed password string obtained from tools like `openssl passwd`.

5. Create a user with an expiration date for temporary access:

```
useradd -e 2023-12-31 guest
```

This command creates a user named "guest" with an expiration date set to December 31, 2023. The user account will automatically be disabled after the specified date.

These examples demonstrate various use cases and options that can be utilized with the `useradd` command to create user accounts with specific properties and settings.

2. User Account Management: groupadd

The `groupadd` command in Linux is used for creating a new group. It allows you to add a new group to the system, which can be used for managing file permissions and providing shared access to files and directories. Here's an explanation of the `groupadd` command along with some unique examples:

Syntax:

```
groupadd [options] group_name
```

Options:
- `-g`, `--gid GID`: Specify the numerical group ID (GID) for the new group.
- `-r`, `--system`: Create a system group.
- `-f`, `--force`: Forcefully create the group even if it already exists.
- `-h`, `--help`: Display the help message for the command.
- `-v`, `--version`: Show the version information.

Example 1:
To create a new group called "developers" with the default settings, you can use the following command:

```
groupadd developers
```

This will create a group named "developers" with a unique Group ID (GID) assigned by the system.

Example 2:
To create a group with a specific GID, you can use the `-g` option. For instance, to create a group named "staff" with GID 1001, you can execute the following command:

```
groupadd -g 1001 staff
```

This will create a group named "staff" with the specified GID 1001.

Example 3:
If you want to create a system group, which is primarily used for system processes and services, you can use the `-r` option. For example, to create a system group named "sysadmin," you can run the following command:

```
groupadd -r sysadmin
```

This will create a system group called "sysadmin" with a unique GID.

Example 4:
In some cases, you may need to force the creation of a group, even if it already exists. You can utilize the `-f` option for that. For instance, to create a group named "testgroup" regardless of whether it already exists, you can use the following command:

```
groupadd -f testgroup
```

If the group "testgroup" already exists, it won't be modified, but if it doesn't exist, a new group will be created.

These are just a few examples of how you can use the `groupadd` command to create new groups in Linux. By managing groups, you can control file permissions, assign access rights, and provide collaborative environments for users in your system.

3. User Account Management: usermod

The `usermod` command in Linux is used for modifying user account attributes. It allows you to make changes to an existing user account, such as modifying the user's group, home directory, shell, expiration date, and more. Here's an explanation of the `usermod` command along with some unique examples:

Syntax:

```
usermod [options] username
```

Options:
- `-c`, `--comment COMMENT`: Add or modify the user's comment or description.
- `-d`, `--home HOME_DIR`: Modify the user's home directory.
- `-e`, `--expiredate EXPIRE_DATE`: Set the expiration date for the user account.
- `-g`, `--gid GROUP`: Change the user's primary group.
- `-G`, `--groups GROUPS`: Modify the user's supplementary groups.
- `-s`, `--shell SHELL`: Change the user's login shell.
- `-l`, `--login NEW_LOGIN`: Rename the user account.
- `-L`, `--lock`: Lock the user account.
- `-U`, `--unlock`: Unlock the user account.
- `-h`, `--help`: Display the help message for the command.
- `-v`, `--version`: Show the version information.

Example 1:
To modify the comment or description of a user account, you can use the `-c` option. For example, to add a comment "John Smith" to the user account "john", you can execute the following command:

```
usermod -c "John Smith" john
```

This will modify the comment field of the user account.

Example 2:
If you need to change the home directory of a user, you can use the `-d` option. For instance, to change the home directory of the user "john" to "/home/johnny", you can run the following command:

```
usermod -d /home/johnny john
```

This will update the user's home directory to the specified path.

Example 3:
To set an expiration date for a user account, you can use the `-e` option. For example, to set an expiration date of January 1, 2024, for the user "john", you can execute the following command:

```
usermod -e 2024-01-01 john
```

This will set the expiration date for the user account.

Example 4:
To change the primary group of a user, you can use the `-g` option. For instance, to change the primary group of the user "john" to "staff", you can run the following command:

```
usermod -g staff john
```

This will modify the user's primary group to the specified group.

Example 5:
If you want to add or remove supplementary groups for a user, you can use the `-G` option. For example, to add the user "john" to the groups "developers" and "admins", you can execute the following command:

```
usermod -G developers,admins john
```

This will add the user to the specified supplementary groups.

These are just a few examples of how you can use the `usermod` command to modify user account attributes in Linux. By utilizing `usermod`, you can make necessary adjustments to user accounts to meet specific requirements or manage user access and permissions effectively.

4. User Account Management: userdel

The `userdel` command in Linux is used for deleting user accounts from the system. It allows you to remove a user account, including their home directory and associated files. Here's an explanation of the `userdel` command along with some unique examples:

Syntax:

```
userdel [options] username
```

Options:
- `-r`, `--remove`: Remove the user's home directory and mail spool.
- `-f`, `--force`: Forcefully remove the user account, even if the user is currently logged in or owns files.
- `-h`, `--help`: Display the help message for the command.
- `-v`, `--version`: Show the version information.

Example 1:
To delete a user account named "john" without removing the associated home directory, you can use the following command:

```
userdel john
```

This will remove the user "john" from the system, but the home directory and its contents will remain intact.

Example 2:
If you want to delete a user account and remove their home directory and mail spool, you can use the `-r` option. For instance, to delete the user account "mary" and remove her home directory, you can execute the following command:

```
userdel -r mary
```

This will delete the user "mary" from the system along with her home directory and mail spool.

Example 3:
In some cases, you may need to force the removal of a user account, even if the user is currently logged in or owns files. You can utilize the `-f` option for that. For example, to forcefully delete the user account "testuser," irrespective of active sessions or file ownership, you can use the following command:

```
userdel -f testuser
```

This will remove the user "testuser" forcefully, even if there are active sessions or files owned by the user.

It's important to exercise caution when using the `userdel` command, especially with the `-r` and `-f` options, as it permanently deletes user accounts and associated files. Ensure that you have taken the necessary backups or transferred any required data before executing the command.
These are just a few examples of how you can use the `userdel` command to delete user accounts in Linux. By managing user accounts, you can control access, maintain system security, and organize user-related resources in your system.

5. User Account Management: groupdel

The `groupdel` command in Linux is used for deleting a group from the system. It allows you to remove an existing group, along with its associated files and permissions. Here's an explanation of the `groupdel` command with some unique examples:

Syntax:

```
groupdel [options] group_name
```

Options:
- `-f`, `--force`: Forcefully delete the group, even if it still has users as members.
- `-h`, `--help`: Display the help message for the command.
- `-v`, `--version`: Show the version information.

Example 1:
To delete a group named "developers" from the system, you can use the following command:

```
groupdel developers
```

This will remove the group "developers" from the system, along with any associated files and permissions.

Example 2:
By default, the `groupdel` command does not allow you to delete a group if it still has users as members. However, you can force the deletion using the `-f` option. For instance, to forcefully delete the group "developers" even if it still has users, you can run the following command:

```
groupdel -f developers
```

Be cautious when using the `-f` option as it can result in the removal of group-related files and permissions without proper checks.

Example 3:
To check the status and effect of the `groupdel` command without actually deleting the group, you can use the `-v` option. For example, to see the output of deleting the group "developers" without performing the actual deletion, you can execute the following command:

```
groupdel -v developers
```

This will display the status and messages related to the deletion process without affecting the actual group.

These are a few examples of how you can use the `groupdel` command to delete groups in Linux. It's important to exercise caution when deleting groups, especially when using the `-f` option, as it can have unintended consequences such as the loss of group-related files and permissions. It is recommended to review the group's membership and verify the need for deletion before proceeding.

6. User Account Management: Switch users and sudo access (su, sudo)

In Linux, the commands `su` and `sudo` are used to switch users and execute commands with administrative privileges. These commands play a crucial role in user account management and access control. Here's a detailed explanation of `su` and `sudo`, along with some unique examples:

1. `su` Command (Switch User):
  - **Syntax**: `su [options] [username]`
  - The `su` command allows you to switch to another user account, typically the root user, by providing the username and the associated password.
  - By default, running `su` without specifying a username switch to the root user.

  Examples:
  - To switch to the root user account, simply run:
    ```
    su
    ```

The root password must be entered when requested. You will have access to the root user's privileges after successfully authenticating.

  - To switch to a specific user account, use the following format:

```
su username
```

Replace "username" with the desired username. You'll be asked to enter that user's password.

- To switch to another user account and open a new shell session, use the following command:

```
su – username
```

This command starts a new shell session with the environment variables and settings of the specified user.

2. `sudo` Command (Superuser Do):

- **Syntax**: `sudo [options] command`
- The `sudo` command allows authorized users to execute commands with administrative or superuser privileges.
- Users granted sudo access can perform tasks that require elevated privileges without needing to switch to the root user.

Examples:

- To execute a single command with superuser privileges, use the following format:

```
sudo command
```

Replace "command" with the desired command. You will be prompted to enter your own password (not the root password) to verify your authorization.

- To execute a command as another user with sudo access, use the `-u` option:

```
sudo -u username command
```

This allows you to run the specified command as the specified user, provided you have the necessary permissions.

- To start an interactive shell session with superuser privileges, use the following command:

```
sudo -i
```

This opens a new shell with the environment variables and settings of the root user.

- To edit a file with superuser privileges using a text editor, you can use the following command:

```
sudo nano /path/to/file
```

Replace "/path/to/file" with the path to the file you want to edit. This command launches the Nano text editor with superuser privileges.
These examples demonstrate how `su` and `sudo` commands can be used to switch users and execute privileged commands in Linux. It's important to note that the availability and configuration of `su` and `sudo` can vary across different Linux distributions and system setups.

7. User Account Management: Monitor users (who, last, w, id)

The commands `who`, `last`, `w`, and `id` are used for monitoring user activity and obtaining information about logged-in users in a Linux system. Here's a detailed explanation of each command along with suitable examples:

1. `who` Command:
- The `who` command displays information about users who are currently logged into the system.
- Syntax: `who [options]`
- Commonly used options:
  - `-a` or `--all`: Display information about all users, including system processes.
  - `-H` or `--heading`: Print a header line before the output.
  - `-q` or `--count`: Show the total number of users currently logged in.
  - `-u` or `--users`: Show only the username and login time.

Example:
```
$ who
user1   tty1        2023-06-20 10:30 (:0)
user2   pts/0        2023-06-20 14:45 (:1)
```

2. `last` Command:
- The `last` command displays information about previously logged-in users and their login history.
- Syntax: `last [options] [username]`
- Commonly used options:
  - `-n <number>`: Show the last `<number>` of lines.
  - `-a` or `--all`: Display all available information.
  - `-i` or `--ip`: Display IP addresses along with user login information.

Example:
```
$ last
user1   pts/0192.168.1.10    Mon Jun 19 21:30   still logged in
user2   tty1   Sun Jun 18 14:45   still logged in
```

3. `w` Command:
- The `w` command displays a summary of currently logged-in users and their activities.
- Syntax: `w [options] [username]`
- Commonly used options:
  - `-h` or `--no-header`: Do not print the header line.
  - `-u` or `--from`: Show the duration of the user's idle time.
  - `-s` or `--short`: Display a compact output.

Example:
```
$ w
09:30:40 up 2 days, 3:15, 2 users, load average: 0.12, 0.23, 0.18
USER    TTY   FROM          LOGIN@   IDLE   JCPU   PCPU  WHAT
user1   pts/0 192.168.1.10   21:30    2.5s  0.10s  0.05s  bash
user2   tty1               14:45    1:30m  0.15s  0.12s  vim file.txt
```

4. `id` Command:
- The `id` command displays information about a specific user or the current user.
- Syntax: `id [options] [username]`
- Commonly used options:
  - `-u` or `--user`: Show only the user's numerical ID.
  - `-g` or `--group`: Show only the user's group ID.

Example:
```
$ id user1

uid=1000(user1)gid=1000(user1)groups=1000(user1),4(adm),24(cdrom),27(sudo),30(dip),46(
plugdev),116(lpadmin),126(s
```

8. System Utility Commands: date, uptime, hostname, uname, which, cal, bc.

Sure! Here's detailed information on several system utility commands in Linux: `date`, `uptime`, `hostname`, `uname`, `which`, `cal`, and `bc`.

1. `date`:
- The `date` command is used to display or set the system date and time.

- By default, running `date` without any options will display the current date and time.
- You can also use `date` with options to format the output, set the date and time, or convert timestamps.

2. `uptime`:

- The `uptime` command shows how long the system has been running and the average system load over different time intervals.
- When you type "uptime," it will show you the time right now, how long the system has been running, how many users are signed in, and the system load averages over the previous 1, 5, and 15 minutes.

3. `hostname`:

- The `hostname` command displays or sets the system's hostname.
- Running `hostname` without any options will display the current hostname of the system.
- You can also use `hostname` with options to set a new hostname.

4. `uname`:

- The `uname` command provides information about the system and the kernel.
- Commonly used options for `uname` include:
  - `-a` or `--all`: Displays all available information about the system.
  - `-s` or `--kernel-name`: Prints the kernel name.
  - `-r` or `--kernel-release`: Prints the kernel release.
  - `-v` or `--kernel-version`: Prints the kernel version.
  - `-m` or `--machine`: Prints the machine hardware name.
  - `-p` or `--processor`: Prints the processor type.

5. `which`:

- The executable file linked to a particular command can be found using the "which" command.

- Running `which` followed by the name of a command will display the path to the executable file.
- This command is helpful to determine the location of a command in the system's executable directories.

6. `cal`:

- The `cal` command displays a calendar for a specified month or year.
- Running `cal` without any options will display the current month's calendar.
- You can use `cal` with options to display calendars for specific months or years, highlight the current day, or customize the output format.

7. `bc`:

-The 'bc' command is a language for an arbitrary precision calculator.
- It provides a command-line calculator with support for mathematical expressions, variables, and functions.

- You can use `bc` for basic calculations or write more complex scripts using its scripting capabilities. These system utility commands provide useful information and functionality for managing and interacting with your Linux system.