3.5 Execute process commands- like PS, wait sleep exit kill

Process Commands

Process commands are used to manage and control processes within an operating system. They allow users to view, manipulate, and control processes that are currently running. Here's a brief overview of some commonly used process commands:

Process Commands Overview

Command	Description	Usage	Example
`ps`	Displays information about active processes.	Lists current processes running on the system.	`ps aux`
`wait`	Waits for a process to complete.	Used in scripts to pause execution until a process finishes.	`wait $pid`
`sleep`	Pauses execution for a specified time.	Delays process execution for a defined period.	`sleep 5` (waits 5 seconds)
`exit`	Terminates the current shell or process.	Ends the execution of a shell or script.	`exit`
`kill`	Sends a signal to a process, usually to terminate it.	Used to stop or send signals to processes.	`kill -9 1234` (forcefully kills process with PID 1234)

Detailed Explanation

1. `ps` (Process Status)

- **Function**: Displays a snapshot of current processes.
- **Usage**:
  - `ps` shows the processes running in the current shell.

- `ps aux` shows detailed information about all processes running on the system.
- **Example**: `ps aux` will list all processes with details such as PID, user, CPU usage, memory usage, and command.

2. `wait`

- **Function**: Waits for a specified process to complete.
- **Usage**:
  - Typically used in shell scripts to wait for background jobs to finish before continuing.
  - Waits for the process with the specified PID or the last background process if no PID is given.
- **Example**: `wait $pid` will pause script execution until the process with ID `$pid` completes.

3. `sleep`

- **Function**: Pauses execution for a set amount of time.
- **Usage**:
  - Used to delay execution in scripts or commands.
  - The time argument can be in seconds (default), or other units like minutes or hours if specified (e.g., `sleep 2m` for 2 minutes).
- **Example**: `sleep 10` pauses the script for 10 seconds.

4. `exit`

- **Function**: Terminates the current shell or process.
- **Usage**:
  - Ends the execution of a shell session or script.
  - Can return an exit status to indicate success or failure (e.g., `exit 0` for success).
- **Example**: `exit 1` exits the current shell with an error status.

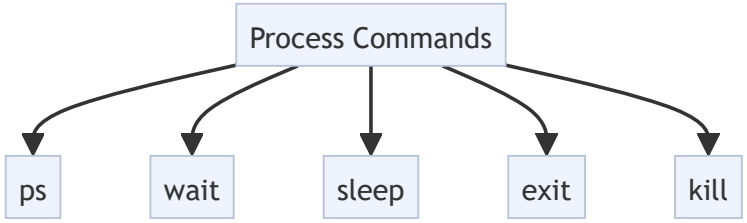
5. `kill`

- **Function**: Sends signals to processes, usually to terminate them.
- **Usage**:
  - `kill` sends a default `SIGTERM` signal which requests the process to terminate.
  - `kill -9` sends a `SIGKILL` signal to forcefully stop the process.

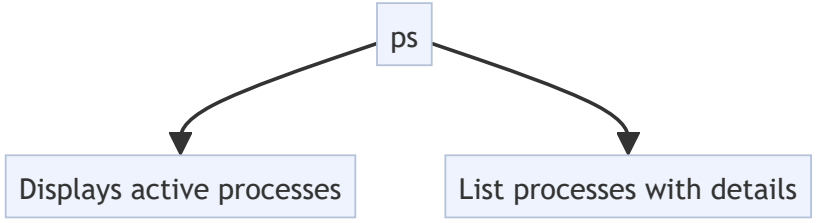
- **Example**: `kill -9 1234` sends a forceful termination signal to the process with PID 1234.

Diagram: Process Commands Overview

1. Process Commands Overview

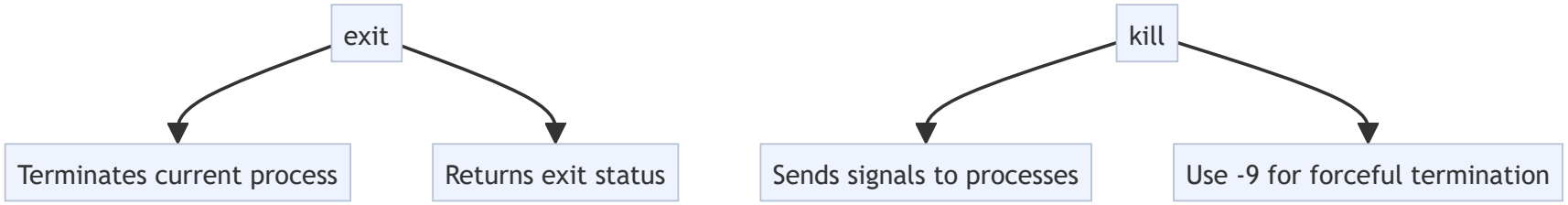


2. Process Command Details: ps



3. Process Command Details: wait and sleep

4. Process Command Details: exit and kill



Summary Table: Process Commands

Command	Function	Usage	Example
`ps`	Displays current processes.	`ps aux` for detailed process list.	`ps aux`
`wait`	Waits for process completion.	`wait $pid` to wait for a PID.	`wait 1234`
`sleep`	Delays execution.	`sleep 5` to wait 5 seconds.	`sleep 10`
`exit`	Terminates the shell/process.	`exit` to end a shell session.	`exit 0`
`kill`	Sends signals to processes.	`kill -9 $pid` to forcefully terminate.	`kill -9 1234`