Mastering `grep`, `sort`, `sed`, and `awk` in Linux

These four commands are **super powerful for file manipulation, searching, and text processing in Linux**, especially for **ethical hacking and automation**.

☑ 1. `grep` (Global Regular Expression Print)

- `grep` is used to **search for text or patterns inside files.**
- It works with **regular expressions**.

📌 Basic Syntax:

```
bash
CopyEdit
grep "pattern" filename
```

🎯 Example:

```
bash
CopyEdit
grep "password" /etc/passwd
```

- This will **search for the word "password" inside `/etc/passwd` file.**

⚡ Advanced `grep` Tricks:

Option	Description	Example
`-i`	Ignore case	`grep -i "root" file.txt`
`-r`	Search recursively in all directories	`grep -r "error" /var/log`
`-n`	Show line numbers	`grep -n "admin" file.txt`
`-v`	Show lines that **do NOT match**	`grep -v "error" file.txt`
`-E`	Use advanced regex	`grep -E "[0-9]{3}-[0-9]{2}-[0-9]{4}" file.txt`

☑ 2. `sort` (Sorting Command)

- Sorts the **content of a file or output.**
- By default, it sorts **alphabetically**.

📌 Basic Syntax:

```
sort filename
```

🎯 Example:

```
sort names.txt
```

⚡ Advanced `sort` Options:

Option	Description	Example
`-r`	Sort in reverse	`sort -r names.txt`
`-n`	Sort numerically	`sort -n numbers.txt`
`-u`	Remove duplicates	`sort -u names.txt`
`-k`	Sort by a specific column	`sort -k 2 file.txt`

☑ 5. `sed` (Stream Editor)

- Used for **editing, finding, and replacing text in a file**.

📌 Basic Syntax:

```
bash
CopyEdit
sed 's/oldtext/newtext/' filename
```

🎯 Example:

```
bash
CopyEdit
sed 's/hello/world/' file.txt
```

- Replaces the **first occurrence** of "hello" with "world".

⚡ Advanced `sed` Tricks:

Task	Command
Replace all occurrences	`sed 's/hello/world/g' file.txt`
Delete a line	`sed '5d' file.txt`
Remove empty lines	`sed '/^$/d' file.txt`
Remove specific words	`sed 's/password//g' file.txt`

☑ 4️⃣ `awk` (Text Processing King)

- Used for **manipulating columns and fields in a file.**
- Perfect for **parsing logs and CSV files**.

📌 Basic Syntax:

```
awk '{print $1}' filename
```

- This prints **the first column** of a file.

🎯 Example:

```
awk '{print $1, $3}' /etc/passwd
```

- This prints the **username and user ID** from `/etc/passwd`.

⚡ Advanced `awk` Tricks:

Task	Command
Print specific columns	`awk '{print $2, $3}' file.txt`
Filter data based on condition	`awk '$3 > 50' data.txt`
Count number of lines	`awk 'END {print NR}' file.txt`
Sum a column of numbers	`awk '{sum += $2} END {print sum}' data.txt`

🎯 Practical Ethical Hacking Example:

```
cat /etc/passwd | grep "user" | awk '{print $1}' | sort -u
```

- Searches for **users with "user" in their username**,
- Extracts **only the username**,
- Sorts and removes duplicates.

📂 Summary of Use Cases:

Command	Use Case
`grep`	Searching for text or patterns
`sort`	Sorting and removing duplicates
`sed`	Find and replace text
`awk`	Extracting and manipulating columns