DEPARTMENT OF CYBERSECURITY

Linux Assignment-10

Name: Mohd Sahil Ansari Class:3A1

USN: ENG24CY0134 Roll No: 22

---

chown COMMAND

1. WHAT IS chown?

The "chown" command stands for "change owner." It's a fundamental Linux utility that allows you to change the ownership of files and directories. Think of it as transferring the keys of a house (file) from one person (user) to another. In Linux, every file and directory has an owner (a user) and sometimes a group associated with it. The chown command is your tool to reassign who owns what.

2. **BASIC SYNTAX & UNDERSTANDING:**

chown [OPTIONS] USER[:GROUP] FILE

- USER: The new owner's username
- GROUP: The new group (optional, use colon to specify)
- FILE: The file or directory to change ownership of
- OPTIONS: Additional flags like -R for recursive changes

3. **PRACTICAL EXAMPLES**:

EXAMPLE 1: Simple ownership transfer :

$ chown alice file.txt

This transfers the ownership of file.txt to user alice.

EXAMPLE 2: Change both owner and group

$ chown alice:developers file.txt

This makes alice the owner and assigns the group developers to the file.

EXAMPLE 3: Recursive ownership change (for directories)

$ chown -R alice:developers /home/alice/project/ directory

This changes ownership of the entire /project/ directory and all files inside it to alice and group developers.

4. **HISTORY & BACKGROUND** : The chown command has been part of Unix systems since the early 1970s. It evolved as a necessity for multi-user systems where security and file ownership needed to be carefully managed. In modern Linux systems, it remains essential for system administration, permission management, and maintaining security boundaries between different users and applications.

5. **REAL-WORLD USE CASES**:

System Administration: When a system administrator needs to transfer file ownership to a service account •

Development Teams: Changing project directory ownership when a developer leaves and someone new takes over

• Server Deployment: Ensuring web server files are owned by the correct web server user (like www-data) •

Backup & Recovery: After restoring files from backup, you might need to fix ownership

6**. IMPORTANT CONSIDERATIONS** :

You typically need sudo/root privileges to use chown •

Changing ownership of system files carelessly can break applications

Using -R on large directories takes time and should be done carefully

Always verify file ownership with 'ls -l' before and after making changes

---

unzip -l COMMAND

1. WHAT IS unzip -l?

   The "unzip -l" command is used to LIST THE CONTENTS OF A ZIP ARCHIVE WITHOUT EXTRACTING IT. Imagine you have a suitcase (ZIP file) and you want to see what's inside before unpacking everything. The -l flag tells unzip to just show you a list of items in the archive, leaving the file untouched. This is incredibly useful when you want to preview archive contents or check file sizes before deciding whether to extract.

2**. BASIC SYNTAX & UNDERSTANDING** :

unzip -l archive.zip –

-l: Lists archive contents (doesn't extract)

- archive.zip: The ZIP file to examine

- Other useful flags: -L (lowercase filenames), -v (verbose/detailed info)

3. **PRACTICAL EXAMPLES**:

EXAMPLE 1: Simple list of contents

$ unzip -l documents.zip

Output shows all files and folders inside documents.zip with their sizes.

EXAMPLE 2: Verbose listing with more details:

$ unzip -lv project.zip

Shows detailed information including compression ratio, dates, and permissions.

EXAMPLE 3: List with search/filter:

$ unzip -l backup.zip | grep ".pdf"

Shows only PDF files inside the archive.

EXAMPLE 4: Check archive before extracting:

$ unzip -l large_backup.zip | head -20

Shows the first 20 items, useful for large archives.

4. **HISTORY & BACKGROUND**: ZIP is one of the oldest and most portable archive

formats, created in 1989. The unzip utility became the standard Linux tool for handling ZIP files. The -l option was designed to provide a quick preview capability, making it essential for system administrators and users who need to inspect archives without committing disk space or time to extraction. It remains relevant today for quick file verification and archive inspection.

5. **REAL-WORLD USE CASES:**

- Software Distribution: Checking what's inside a software package before installation

• File Sharing: Previewing ZIP files received from colleagues or downloads

• Backup Verification: Confirming that important files are actually in the backup archive

• Storage Optimization: Checking file sizes before extracting to ensure adequate disk space

• Quality Control: Verifying that the right files were included when someone created a ZIP archive

6. **IMPORTANT CONSIDERATIONS:**

• No disk space required: -l only reads the archive, doesn't extract

• Fast operation: Listing is quick even for very large archives

• Non-destructive: The ZIP file remains completely unchanged

• Cross-platform: Works on Windows (with unzip installed), Mac, and Linux

• Large file sizes: While listing is fast, very large archives (100GB+) might take a moment to scan.

SUMMARY & COMPARISON TABLE

ASPECT	chown	unzip -l
Purpose	Changes file/directory ownership	List ZIP archive contents
Require permissions	Yes (usually sudo/root)	No (read-only operation)
Common Use	Permission Management	Archive preview

Risk Level	High (can break permissions)	Very-low (read-only)

CONCLUSION:

This assignment has provided a comprehensive overview of two essential Linux commands used in cybersecurity and system administration. Understanding these commands is crucial for managing file permissions, securing systems, and efficiently handling compressed archives in professional environments.

The chown command is vital for system administrators who need to manage access controls and ensure proper file ownership across multi-user systems. Meanwhile, unzip -l serves as a safe preview tool that allows professionals to inspect archive contents before committing resources to extraction.

Both commands are essential tools in a Linux user's toolkit—chown for managing permissions and access control, and unzip -l for safely inspecting archives before extraction.