**File Permissions Cheat Sheet**

**A Beginner's Guide to Ownership, Modes, and Security**

Linux is a multi-user system, which means controlling who can access files and directories is essential. Fedora, like other Linux distributions, uses a permissions-based model to manage file security.

In this guide, we'll explore:

* What file permissions are
* How to read file permission output
* Types of users and their roles
* The chmod, chown, and chgrp commands
* Numeric and symbolic permission modes
* Practical examples to reinforce learning

📂 What Are File Permissions?

Every file and directory in Fedora has **permissions** associated with it. These permissions determine:

* Who can read it
* Who can write (edit) it
* Who can execute (run) it

👥 Types of Users in File Permissions

Each file belongs to two key identities:

User The owner of the file

Group The group assigned to the file

Others Everyone else on the system

🔎 Reading File Permissions with ls -l

To view file permissions:

1. ls -l

Example output:

1. -rwxr-xr-- 1 alice developers 1024 May 20 12:00 script.sh

Breakdown of the first 10 characters:

1. -rwxr-xr--
2. | | | |
3. | | | └── "Others" permissions (r--)
4. | | └──── "Group" permissions (r-x)
5. | └────── "User" permissions (rwx)
6. └───────── File type (- means file, d means directory)

🔤 Understanding Permission Characters

Symbol Meaning

r          Read permission

w          Write permission

x          Execute permission

-          No permission

🔢 Numeric (Octal) Representation

Each permission is also represented by a number:

PermissionValue

r (read) = 4

w (write) = 2

x (exec) = 1

Add the values together to form the numeric mode:

* rwx = 4+2+1 = **7**
* rw- = 4+2+0 = **6**
* r-- = 4+0+0 = **4**

So -rwxr-xr-- becomes: 754

✍️ Changing Permissions with chmod

Use chmod to **change file or directory permissions**.

1. Symbolic Mode

1. chmod u+x script.sh # Give execute to user
2. chmod g-w file.txt # Remove write from group
3. chmod o+r notes.txt # Add read for others

2. Numeric Mode

1. chmod 755 script.sh # rwxr-xr-x
2. chmod 644 document.txt # rw-r--r--
3. chmod 700 secrets.txt # rwx------

👑 Changing Ownership with chown

Only root or sudo users can change file ownership.

1. sudo chown dan notes.txt # Change owner to 'dan'
2. sudo chown dan:staff notes.txt # Change owner and group

👥 Changing Group with chgrp

1. sudo chgrp developers project.txt

📁 Directories vs Files

Permissions apply slightly differently to directories:

Permission On a File                 On a Directory

r                 Read file content List contents of the directory

w                 Modify file                 Add/remove files in the directory

x                 Execute file                 Enter the directory (cd)

🧪 Try It Yourself – Practice Examples

1. Create a new file and give it rw-r--r-- permissions:
2. touch testfile.txt
3. chmod 644 testfile.txt
4. ls -l testfile.txt
5. Make a directory private:
6. mkdir secrets
7. chmod 700 secrets
8. Change ownership of a file:
9. sudo chown yourusername testfile.txt

🛡️ Why File Permissions Matter

File permissions are your first line of defense against:

* Accidental deletions or overwrites
* Unauthorized access
* Security breaches

Incorrect permissions are one of the most common issues in Linux server misconfigurations. Learning to manage them gives you control and confidence.

Course content

AI Assistant