**Report: File Creation and Permission**

**Name:** Fariha Shariff  
**Date:** 1/08/25  
**Topic:** File Permissions Using Numeric Notation (751)

**Methodology**

1. Opened terminal in a Linux environment.
2. Created a new file using touch.
3. Set file permissions using chmod 751.
4. Verified permissions with ls -l.
5. Explained permission breakdown and security implications.

**Commands Used**

Step 1: Create the file

touch secure\_script.sh

Step 2: Apply permissions

chmod 751 secure\_script.sh

Step 3: Verify permissions

ls -l secure\_script.sh

**Output**

fariha@linux:~$ touch secure\_script.sh

fariha@linux:~$ chmod 751 secure\_script.sh

fariha@linux:~$ ls -l secure\_script.sh

-rwxr-x--x 1 fariha users 0 Jul 31 12:30 secure\_script.sh

**Findings**

* **Permission 751** breakdown:
  + **Owner (7)** = rwx → Read, Write, Execute
  + **Group (5)** = r-x → Read, Execute
  + **Others (1)** = --x → Execute only
* Only the file **owner** can modify it.
* **Group members** can run/view the file but not edit.
* **Others** can only **execute** — they cannot read or write the contents.

**Security Analysis**

This setup is secure for script files where:

* Only the owner should edit.
* Group users can run the script.
* Others can run it without accessing or modifying it.  
  Useful for executable scripts or tools in shared environments.

**Conclusion**

* Learned how to set **granular file permissions** using numeric mode.
* chmod with octal values offers **precise control** over access.
* Reinforced importance of least privilege principle to prevent accidental or malicious access.