SNA Labs (Finals) - Jaish Khan

Table of Contents

1. File Systems and Architecture

1.1. File System Architecture

File System --> The method operating systems use to organize and store files on storage devices. It handles data storage, retrieval, and organization of files and directories. Choosing the correct file system is very important for system efficiency and reliability since they have different levels of performance, data integrity, and security.

1.2. Types of File System Architecture

1. **FAT32** (File Allocation Table 32) --> introduced by Microsoft in the mid-1990s, is simple and highly compatible across operating systems, making it ideal for USB drives and external storage. However, it has significant limitations, such as a maximum file size of 4GB and partition size of 2TB, which makes it unsuitable for modern needs.
2. **NTFS** (New Technology File System), also developed by Microsoft, is the default file system for Windows. It supports larger file and partition sizes, offers better security with file permissions and encryption, and provides advanced features like journaling and compression. It is less compatible with non-Windows systems without additional software.
3. **ext4** (Fourth Extended Filesystem) is the standard file system for Linux systems. It balances performance and reliability with features like journaling, support for large volumes, and reduced fragmentation. Although it works well in Linux, its compatibility with Windows and macOS is limited without third-party tools. Each file system serves a

specific purpose, with FAT32 focusing on portability, NTFS on advanced Windows integration, and ext4 on Linux efficiency and scalability.

Feature	NTFS	EXT4
Made By	Microsoft	Linux community (open source)
Platform	Windows	Linux
Max File Size	16 exabytes (practical lower)	16 terabytes
Max Volume Size	256 terabytes	1 exabyte
Journaling	Yes, transaction logs	Yes, write-ahead logging
File Permissions	Advanced (ACLs)	Standard POSIX (user/group/world)
File System Type	MFT-based	Inode-based, uses block groups
Compression	Supported	Not natively supported
Encryption	Supported (EFS)	Not natively supported
Performance	Good, large files	Excellent, particularly for Linux workloads
Use Cases	Windows, large file storage	Linux, general-purpose computing
Hard Link Support	Yes	Yes
Symbolic Link Support	Yes	Yes

1.3. Disk Partitioning and Mounting

Disk Partitioning --> Dividing a physical disk into separate, isolated sections (partitions), each functioning as a logical disk.

- **Why Partition?**
  - Organization --> Separate data into sections (OS, user data, backups).
  - Dual Booting --> Install multiple OSes on the same disk.
  - Data Management --> Improve performance by isolating filesystems, simplify backup/restore.
  - Security --> Limit access to specific disk sections.
  - Tools --> Disk Management tool, Diskpart command line.

Disk Mounting --> Making a partition's filesystem accessible to the OS and user. The OS can then read/write to the disk. Linux mounts to directories like /mnt or /media. Windows uses drive letters (D:, E:).

- **Why Mount?** --> Allows the OS to interact with new disks/partitions by assigning a logical location (directory) for file access.

2. Linux Installation and Configuration

Linux --> An open-source OS known for stability, security, and customization, popular among developers and system administrators.

- **WSL** (Windows Subsystem for Linux) --> Enables running a full Linux environment directly on Windows 10, providing access to Linux tools, applications, and a terminal.

2.1. Installation on Windows

1. **Check Windows Version**
   WSL requires Windows 10 version 1607 or later. Check by pressing `Win+R`, typing `winver`, and pressing Enter.
2. **Enable WSL and Virtual Machine Platform**
   Open PowerShell as Administrator and run:

   ```
   dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
   dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
   ```

   Restart your computer.
3. **Install a Linux Distribution**
   Open the Microsoft Store, search for "Linux" or "WSL," select a distribution (e.g., Ubuntu), install, and launch it.
4. **Set Up Linux**
   On first launch, set up a UNIX username and password.
5. **(Optional) Set WSL 2 as Default**
   Run in PowerShell as Administrator:

   ```
   wsl --set-default-version 2
   wsl --set-version <DistroName> 2
   ```

6. **Access Files**
   Open File Explorer: `\\wsl$`
7. **Install Software**
   Use your distribution's package manager:

   ```
   sudo apt update
   sudo apt install <package-name>
   ```

2.2. Linux File System Structure

Linux uses a hierarchical structure with `/` as the root directory.

Common Directories:

- `/home/` : User home directories
- `/etc/` : Configuration files
- `/bin/` : Essential binaries
- `/usr/` : Programs, libraries, docs
- `/var/` : Logs, spools
- `/tmp/` : Temporary files

Commands:

- Create a directory: `mkdir <directory>`
- Create a file: `touch <file>` or `echo 'text' > <file>`
- Verify files: `ls` or `stat <file>`
- Add a user: `useradd -m -s /bin/bash <username>`
- View users: `cat /etc/passwd`

2.3. File and Directory Permissions in Linux

Permissions control access to files and directories (read: `r`, write: `w`, execute: `x`).

View Permissions:

```
ls -l <filename>
```

Modify Permissions:

```
chmod u+x <file>   # Add execute for owner
chmod g-w <file>   # Remove write for group
chmod o=r <file>   # Read-only for others
chmod 755 <file>   # Owner rwx, group/others rx
chmod 644 <file>   # Owner rw, group/others r
```

---

3. System Security and Hardening Best Practices

1. **Least Privilege:** Only give users and services the access they truly need. Avoid giving admin or root privileges unnecessarily, and check access regularly.
2. **Regular Updates:** Keep your system and apps updated, especially for security fixes. Use automated tools to manage updates.
3. **Strong Authentication:** Use multi-factor authentication (MFA) and strong passwords. Disable any default or unused accounts.
4. **Encryption:** Protect sensitive data by encrypting it during transfer and storage. Use secure protocols like HTTPS and SSH.
5. **Backups:** Automatically back up important data and system settings, store backups securely off-site, and test that you can restore them.
6. **Firewalls & Network Segmentation:** Set up firewalls, separate networks by purpose, and use tools to detect and prevent threats.
7. **Logs and Audits:** Turn on detailed logging, check logs regularly, and use a centralized system to manage them.
8. **Remote Access:** Use VPNs for secure remote connections, limit access to trusted IPs, and enable MFA. Consider using bastion hosts for extra security.
9. **System Hardening:** Strengthen systems by removing unnecessary apps and services, and follow best-practice guidelines like CIS Benchmarks.
10. **User Awareness:** Train users on safe practices, like avoiding phishing scams and using strong passwords. Test their knowledge with simulated attacks.
11. **Access Control:** Use role-based access controls (RBAC), check user access logs, and remove unneeded permissions.
12. **Traffic Monitoring:** Monitor network traffic to spot unusual patterns and potential threats.
13. **Separation of Duties:** Split important security tasks among multiple people to reduce the risk of insider threats.