CSE321 – Theory Assignment 01

Title: Research and Exploration of Modern Bootloader Systems

1. Boot Sequence Workflow and its Components

1.1 Bootloaders based on BIOS

The BIOS (Basic Input/Output System) is older firmware interface that is only found on older systems, and boot workflow consists of the following:

1. Power-On Self-Test: when the computer powers-on, BIOS will confirm the hardware components are operational (RAM, CPU, keyboard, storage).

2.Firmware Initialization: BIOS will initialize needed system hardware; then, it will boot from the selected device in the boot sequence

3.Boot Record Execution:

- The BIOS reads the first sector (512 bytes) of a boot device, called the Master Boot Record (MBR).

- The MBR contains the Partition Table and a small Stage 1 boot code program.

4.Stage 2 Bootloader: The Stage 1 bootloader loads an advanced Stage 2 bootloader such as GRUB or LILO.

5.OS Kernel Loading: The bootloader identifies an operating system kernel (Linux kernel, Windows NT Loader) and loads it.

6.OS Handoff: The control is handed over to the operating system to finish the startup.

📌 **Important Roles:**

- Firmware (BIOS): Initializes hardware, finds boot device.
- Boot Record (MBR): Contains partition information and bootloader code.
- Boot manager (GRUB): Presents menu and loads OS kernel.
- Boot Record which in this case is the MBR takes care of the storage of important pieces of information such as the partition details and also holds the first boot loader code.

1.2 Bootloaders Based on UEFI

The UEFI (Unified Extensible Firmware Interface) is the modern replacement for BIOS, supporting larger storage and additional features with its own workflows which are:

1..POST (UEFI) - Hardware initialization and diagnostic

2.Firmware initialization - UEFI will read the boot parameters from NVRAM.
UEFI Boot Manager:

3.UEFI Boot Manager:

- A built-in boot manager that finds Operating System boot loaders located in the EFI System Partition (ESP).
- Example loaders - bootmgfw.efi (Windows), grubx64.efi (Linux).

4.Bootloader Execution - UEFI directly executes .efi programs (bootloaders) while not being restricted by a 512-byte MBR.

5.Loading The Operating System Kernel: The operating system bootloader loads the kernel into memory.

6.Handing Control To The Operating System: The operating system is handed control and gets to fully execute.

📌 **Important Roles:**

- Firmware layer (UEFI): initializes any hardware, manages boot order, and loads .efi files to launch an operating system or specific executable.
- Boot records (ESP): holds multiple bootloader executables.
- Boot Manager (individual UEFI software bits + OS-specific bootloader): presents menus, how to access advanced options, verifies secure boot option.

2. Architectural Comparison

Feature	BIOS	UEFI
Boot Mode	16-bit Real Mode, MBR	32/64-bit Protected Mode, EFI executables
Boot Loader Code Limit	446 bytes in MBR	No strict size limit
Partition Table	MBR (4 primary partitions max)	GPT (up to 128 partitions)
Disk Size Support	Up to 2 TB	Up to 9 ZB

Storage of Settings	Stored in ROM/Flash	Stored in NVRAM
Security Features	None	Secure Boot, signed executables
Hardware Compatibility	Legacy systems, older hardware	Modern hardware, advanced features
File System Support	Limited, usually FAT16	FAT32, NTFS, ext4, etc.

3. Multi-Boot Systems

3.1 Multi-Boot Loaders and Their Functions

Multi-purposed boot loaders allow individuals' installation and use of multiple operating systems on one machine, which presents the user with a boot menu at startup with their options.

Examples:

- GRUB (Linux): Text or graphical boot menu.
- rEFInd (UEFI): Graphical boot manager with OS icons.
- Windows Boot Manager: Primarily for Windows, but can chainload Linux.

3.2 Menu Implementation Techniques

1.Text-Interface Menu:

- GRUB shows a list of its operating systems in a terminal-like menu.

2.Graphical Interface Menu:

- rEFInd and modern GRUB versions can create GUI menus that utilize all the EFI icons and logos.

3.3 Difficulties with Multi-Booting Systems Engagement

1.OS Detection Issues:

- The operating system one is installing may not be able to detect the operating system already on your computer.
- **Solution:** Use detection applications (os-prober) or manually add entries.

2.Chainloading Issues:

- Installing Grub allows you to chainload into the next loader (Grub →Windows Boot Manager).
- Solution: Set up the required chainloader directives correctly.

3.File System & Partition Issues:

- Windows uses NTFS file system and Linux uses ext4 files system.
- Solution: Use a shared EFI System Partition to store bootloaders.

4.Secure Boot Problems:

- Secure Boot may render your unsigned bootloaders useless.
- Solution: Use a signed bootloader or disabled Secure Boot.

4. Conclusion

The transition from BIOS to UEFI has allowed modern systems to avoid the legacy limitations of disk size, partitioning, and hardware support. BIOS utilizes the MBR which imposes restrictions, whereas UEFI provides a dedicated EFI System Partition so that firmware has more flexibility.

Multiple boot loaders, like GRUB, rEFInd, and Windows Boot Manager, allow users to run multiple operating systems; however, there will be challenges associated with OS detection (i.e., identifying operating systems), chainloading, and Secure Boot, there are safe and dependable options that accommodate any user of multi-boot systems.

Last but not least, UEFI with GPT and newer boot managers provide the fastest, most flexible and safest support for today's multi-boot operating systems.