**Chapter 9: LINUX SYSTEM CALLS**

**Topic – 1: Introduction To System Calls**

**Definition**

* System calls are **API** between the **user** & **kernel**.

**Steps**

* **Step 1:** Put the **system call number** in **RAX** register.
* **Step 2:** Store the argument to system call in the **appropriate register**.
* **Step 3:** Call the **interrupt** (**0x80** or **80h**), result is returned in **RAX** register.

**Topic – 2: Argument Storage**

* Registers storing arguments of system call – **RBX**, **RCX**, **RDX**, **RSI**, **RDI** & **RBP**.
* For **more** than **6 arguments**, the memory location of **first argument** is stored in **EBX**.

**Topic – 3: System Call Codes**

**System Exit**

***xor ebx, ebx***

***mov eax, 1 ; sys\_exit***

**System Write**

***mov ebx, 1 ; file descriptor (stdout)***

***mov eax, 4 ; sys\_write***

**System Read**

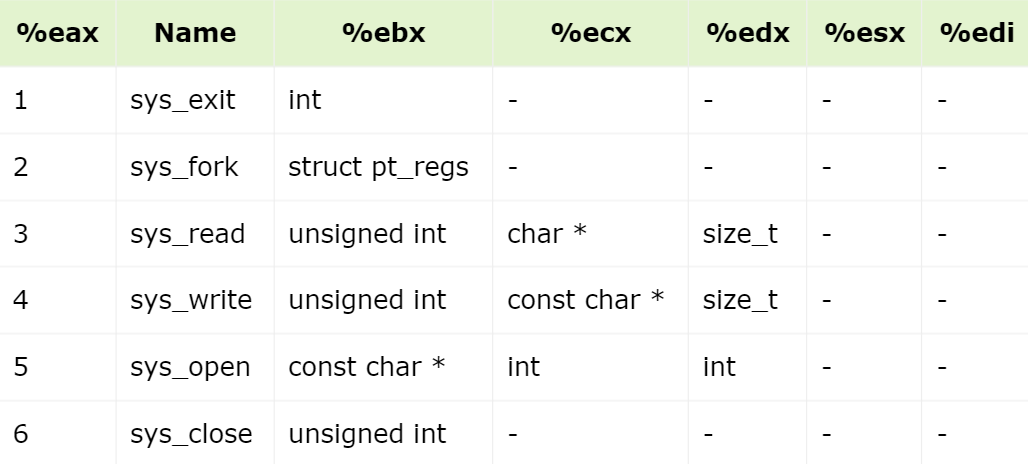
***mov ebx, 2 ; activates error checking (stderr)***

***mov eax, 3 ; sys\_read***

**Note!**

**🡪 All system calls are listed in /usr/include/asm/unistd.h.**

**System Call Table**



**Topic – 4: Input Read & Display Program**

**Note!**

**🡪 It doesn’t matter whether we use single or double inverted commas for character or string.**

**🡪 When we calculate the length, the length isn’t stored anywhere but its memory is referred when it is mentioned.**

**Bss Segment**

***segment .bss***

***num resb 5 ; "resb" means reserve bytes, 1 byte extra for sign***

**Taking Input**

***mov edx, 5***

***mov ecx, num***

***mov ebx, 2***

***mov eax, 3***

***int 0x80***