EMBISYSLABS • TRAINING PROGRAM

Linux Systems Programming Course Bengaluru

Embisyslabs Embedded Systems Training Institutes in bangalore courses offered on Linux system programming and Linux Internals Weekend and Weekdays. Linux is a modern Unix-like system, written from scratch by Linus Torvalds and a loose-knit community of programmers around the globe. Although Linux shares the goals and philosophy of Unix, Linux is not Unix. Instead, Linux follows its own course, diverging where desired and converging only where practical. The core of Linux system programming is the same as on any other Unix system. Beyond the basics, however, Linux differentiates itself—in comparison with traditional Unix systems, Linux supports additional system calls, behaves distinctly, and offers new features.

MODULE 1: UNIX and LINUX SYSTEM PROGRAMING

CH1. INTRODUCTION TO UNIX/LINUX

- Histoty of Unix/Linux
- Linux Layered Architecture
- Type of Kernels
- Micro and Monolithic kernel
- Different types of kernel structure
- Linux Bootup Sequence

CH2. FILE SYSTEM MANAGEMENTS

- File Systems – VFS
- File Systems Layouts
- Super Block & Inode Block
- Inode block Structure
- Device Special Files
- Types of File
- File descriptor table
- System calls Sequence
- System Vs Function Calls
- File related System Calls
- open(),read(),write(),close()
- stat(),lstat(),dup() etc.

CH3. FILE LOCKING PROGRAMMING

- File Control Operations
- Types of File Locking
- Advisory and Mandatory File locking
- fcntl() and flock()calls

CH4. PROCESS MANAGEMENTS

- Program and Process
- Process Control Block (PCB)
- States Of Process
- Mode of Execution User mode and Kernel mode
- Context Switching

- Scheduling & Priority

CH5. PROCESS RELATED PROGRAMMING

- Process Creation by fork() amd vfork()
- Why fork() not vfork()
- Creation and Destroying Zombie Process
- Creation of Orphan Process
- wait() and waitpid() calls
- exit() and exec() ,sleep() calls
- Creating , synchronizing and performing multiprocessing concepts
- Setting and changing nice value and Prority no.

CH6:MEMORY MANAGEMENTS AND MMU

- Memory Policy and Hirarchy
- Memory allocation Technique
- Physical memory &Virtual Memory
- Paging & Demand paging
- Memory Mapping using TLB
- Swap in & Swap out
- Internal & External Fragmentation

MODULE 2: LINUX INTERNALS AND IPCs

CH1. THREADS AND MULTI-THREAD CONCEPTS

- Threads on different O.S
- Why Threads in Linux
- Threads Vs Process
- Thread APIs
- Creation of Multithreading
- Performig Multiple operation using multi-threading

CH2. SIGNALS VS. INTERRUPTS

- Sources of Signals
- Diffrents type of Signals
- Actions of Signals
- Receiving a Signal
- Handling a Signal
- Signal System Calls

CH3. USER AND DAEMON PROCESS

- Creating a Daemon Process
- Characteristics of a Daemon
- Writing and Running Daemon

CH4 . PRIMITIVE INTERPROCESS COMM (IPCS)

- PIPES
- Creation of Half and Full-duplex
- Half and Full-duplex communication

- FIFO

CH5 . SYSTEMS V IPCs

- Shared Memory
- Message Queues
- Semaphores

CH6 . NETWORK AND SOCKET PROGRAMMING

- Description of ISO/OSI Model
- Types of IP Classes (A,B,C,D and E)
- Configuring IP address on Systems
- Network addresses and Host addresses
- Types of Socket:
- UDP Connectionless Oriented Socket
- TCP/IP Connection Oriented Socket
- Iterative Server-Client Programming
- Concurrent Server- Client Programming
- One Server and Many client Programming
- Weekend and Weekdays Training Courses on Linux Systems and Internals programming
- What would you learn about Linux system programming in Embisyslabs?
- A Linux kernel, C library, and C compiler overview
- Basic I/O operations, such as reading from and writing to files
- Advanced I/O interfaces, memory mappings, and optimization techniques
- The family of system calls for basic process management
- Advanced process management, including real-time processes
- Thread concepts, multithreaded programming, and Pthreads
- File and directory management
- Basic and advanced signal interfaces, and their role on the system