Advanced Linux Systems Programming

Course Summary

Description

This course introduces the participants to system level programming in the C language in a Linux environment. The course focuses on Linux system calls and library functions, how to use them, and their underlying mechanisms. The course deals with many facets of the Linux operating system, including: introduction to Linux kernel structure, I/O, Signals, Signal handlers, Timers, Processes, Multi-Tasking, Inter-Process Communication (IPC) Pipes, Shared memory, Message Queues, Semaphores, Networking, Sockets, using TCP/IP and UDP/IP. Throughout the course the information presented is related to the participant through: the execution of common Linux user/administrator commands, and writing, compiling, and executing example C language programs which demonstrate the use of system routines and accessing system data structures on a live Linux system.

This course is the equivalent of Red Hat course 251.

Objectives

After taking this course, students will be able to:

- Explain the programmable mechanisms in a Linux environment
- Write applications using standard Linux system calls and library functions

Topics

- System Programming Environment of Linux
- File Systems
- Process Creation and Control
- Synchronization and System Information
- Interprocess Data Communication Facilities
- Sharing Code between Processes
- Networking

Audience

This course is for those wanting to learn system level programming in the C language in a Linux environment.

Prerequisites

It is assumed that the participant has a solid background in basic Linux utilities and editors (such as vim), and a working knowledge of the C (or C++) programming language(s).

Duration

Five days

Advanced Linux Systems Programming

Course Outline

I. System Programming Environment of Linux
- A. Environment of a C language program
- B. System level programming requirements:
  1. C compiler issues
  2. Header files and libraries
  3. Special data types used
  4. Useful functions
  5. Error handling (basic)
- C. Documentation
- D. Security Issues

II. File Systems
- A. Types of file I/O
- B. File I/O structures
- C. File I/O access types
- D. Dealing with STDIN, STDOUT, STDERR
- E. Creating and using temporary files
- F. Directory file access and manipulation
- G. Permissions

III. Process Creation and Control
- A. Attributes (username, UID, PID, Groups)
- B. Creation methods
- C. Multi-tasking
- D. Shells
- E. Synchronization
- F. An introduction to POSIX threads

IV. Synchronization and System Information
- A. Time issues
  1. How time is maintained
  2. Timers
- B. General synchronization
  1. semaphores
  2. mutexes
  3. spinlocks and barriers
  4. signals (generation and handling)
- C. System information
  1. uname
  2. hostname
  3. load averages

V. Interprocess Data Communication Facilities
- A. Overview of Linux IPC Facilities
- B. Memory Mapped files
- C. Pipes and Named Pipes
- D. Messages Queues
- E. Creating and Using Shared Memory structures

VI. Sharing Code Between Processes
- A. Building shared object (libraries)
- B. Static Linking
- C. Dynamic Linking

VII. Networking
- A. Concepts and basic requirements
- B. Socket creation and usage
- C. TCP/IP level connections
- D. UDP/IP level connections