Linux Kernel Internal Programming Workshop

This course consists of 3 full days of training on Linux Kernel Internal Programming which covers the 2.4.x and 2.6 kernel programming models. The course starts with a session on the Linux execution environment. This session is a review of the kernel architecture and covers the various features and programming environment. Also covered is a review of processes and threads, exceptions, user versus kernel mode programming aspects and various IPC techniques in Linux Kernel .

Prerequisites : Although not required, the person attending this course should have a basic understanding of the C-programming language and knowledge of Linux or UNIX environment. *This course is prerequisite to the Linux Device Driver Programming workshop.*

Intended Audience : This is an introduction to Linux Kernel Internal, and is intended for system administrators, system and application programmers who desire a comprehensive study of the internals of the Linux operating system.

Introduction to the Linux kernel
History of Linux
Types of kernel
The Linux kernel
Kernel architecture
Configuring and Installing the Kernel
Linux versions
Obtain kernel source
Load the kernel
Configure the kernel
Make commands
Gather dependencies
Build the kernel
Build and install the modules
Install the new kernel
Use the new kernel
Patching the kernel
Lab Exercises

The Boot Process
bios level
boot loader
the start_kernel() function
Virtualization
Concept of function pointers
Concept of objects using c structures
Lab Exercises

Compiling modules and exporting
Module Defined
Static vs dynamic modules
Types of modules in the kernel
Symbols
Header files and compiler options
Compiling drivers
Module documentation
Init_module()
Cleanup_module()
Module_init() and module_exit()
Command line parameters
Kernel symbol table
Lab Exercises

System Calls
System call defined
System calls and API's
System call table, interrupts and handlers
Unistd.h and entry.S files
Implementing a system call
Lab Exercises

The File System
virtual filesystem
VFS data structures super_block,
inode,file,dentry
files associated with process
system calls related to files
Lab Exercises

Process Management
process defined
process descriptor structures in the Kernel
process states
process scheduling
process creation
system calls related to process scheduling
Lab Exercises

Memory Management
Responsibilities of memory management
module
Memory allocation and deallocation system call
Malloc, calloc, free

Demand paging

Process organization in memory
Address translation and page fault handling
in Linux
Memory allocation strategies
Buddy system
Slab Allocator
Swapping memory areas
Memory mapping
Lab Exercises

Programming and debugging tools
Strace: tracing system calls
Tools used to detect memory access error and memory leakage in Linux.

Lab Exercises

Skills Gained
After completing this training, you will be able to:

- Understand Linux Kernel Architecture
- Configuring and Building Kernel Modules
- Compiling and Exporting Modules
- Programming Process and filesystem
- Programming IPC's and Sockets.
- Understanding Memory management and Programming the Memory .

* * * * * * *