

LINUX DRIVER WRITING

Current Version: oi-drivwrit-1.0

Course Length: 3 days

Course Description: This course provides a general introduction to Linux device driver development. Students gain a clear practical understanding of the way drivers are designed, interfaced with the kernel, implemented, and tested.

Audience: Programmers and software designers who plan to integrate hardware devices to the Linux kernel.

Prerequisites: Linux Internals is recommended. Strong C programming skills and intermediate knowledge of UNIX/Linux shell commands are required. Experience with the data structures and basic functions used in the Linux kernel is necessary. Proficiency at configuring and installing a new Linux kernel on a system is assumed.

Course Contents

Introduction to Linux Driver Development

- Introduction and environmental setup
- Kernel versions and compatibility
- Components of the Kernel
- Aims of driver development
- Steps associated with the development of a driver
- How device drivers work
- Stability and security issues

Device Drivers

- Elements of a driver
- Benefits and drawbacks
- Classes of drivers

Linux Kernel Facilities

- System calls
- Data structures
- Functions

Modules

- Benefits of using modules
- Module-related tools

- Compiling, loading, and unloading
- Module implementation
- Automatic module loading

Character Devices

- Accessing the device
- File and inode structure
- File operations
- Reading and writing
- IOCTLs
- Example of a character device

Hardware Aspects

- Accessing memory
- Direct Memory Access
- I/O Management
- PCI and ISA

Block Drivers

- Registration

Networks

- Layer model
- Network communications
- Implementation of the TCP/IP stack
- Data structures
- Socket
- sk_buff
- Inet socket
- proto
- ARP and IP Protocols
- IP Filters
- UDP and TCP

Network Devices

- Integration in the kernel
- Ethernet Devices
- SLIP and PPP

- Loopback
- Dummy devices
- Loading network drivers
- Transmitting and receiving packets
- Device configuration
- Statistics

SCSI Subsystem

- Architecture overview
- Names and conventions
- Upper level
- Block devices (hard disks, CD-ROM)
- Character devices (Tape)
- Generic drivers
- Mid level (boot parameters, proc interface)
- Lower (hardware) level and pseudo drivers

Device Drivers Debugging

- Printing with printk
- Queries
- /proc entries
- Tracing and debuggers
- The blk.h header
- Requests and Mounting