Corporate Training Programmes from kaiwanTECH :: updated Mar 2017

Course Code	Course Name	Level	Brief Synopsis	Prerequisite	Typical Target Audience	Typical Duration [1]
L0	Linux Fundamentals	Basic	Linux Intro. Linux CLI, tools & utilities. Permissions model. Filters (grep/sort) + regular expressions. Scripting with awk, bash and sed.	Any programming language	Freshers as well as experienced people new to the Linux OS, QA, developers and architects who requires real-world bash/awk/sed scripting	3 days
L1	Linux Systems Programming	Intermediate	Linux basic architecture / system calls. Process management (exec/fork/wait/zombies/etc). Signalling. Using GDB. Memory. Scheduler access. Multithreaded programming Pthreads, thread-safety, etc. IPC Mechanisms- Pipes/SysV IPC/Sockets.	Course L0 + 'C' programming skills	Developers, architects, QA, support staff, working on or intending to work on application and/or middleware layers on Linux	5 days
L2	Linux Kernel Internals	Advanced+Hot	OS Architecture. Kernel source tree. Writing kernel modules. Process descriptor. Process/Thread creation code implementation. Linux Memory Management deep dive: hardware paging to arch-independent paging, VM arch, mm org, algos, APIs, kernel segment layout, OOM, COW, THP, fault handling, VMAs, etc. Scheduler internals. Cgroups. Ftrace. Syscall internals. Container technology- intro, LXC, namespaces. Virtualization technology -intro, kvm hypervisor, kvm internals.	Course L1	Developers, architects, QA, support staff, working on or intending to work on OS-level layers on Linux. Includes core OS, platform teams, device driver, embedded Linux teams. As deep knowledge of the OS internals is essential to success.	5 days
L3	Linux Device Drivers	Advanced	OS Architecture recap. Concurrency: locking, deadlock avoidance, mutex & spinlocks, specialized locking, etc. VFS data structures. Character device driver framework, first sample driver. User-kernel interfaces- procfs, sysfs, debugfs, ioctl. IO memory. Blocking IO, reentrant-safety. Hardware Interrupt handling. Timers, kernel threads, workqueues. Driver model. Platform devices and drivers. Device tree. Block Drivers. Linux network stack, NIC drivers deep-dive (with DMA, netfilter, driver code).	Course L2	Developers, architects, QA, support staff, working on or intending to work on Linux device drivers. Includes core OS, platform teams, device driver, embedded / server-side Linux teams, building and/or supporting products.	5 days
L4	Embedded Linux	Advanced+Hot	OS Architecture recap. Workspace setup (toolchain etc). Linux kernel configuration, build and deploy. Building a custom skeleton Linux OS & using via Qemu (ARM-32). Intro to using Buildroot. Intro to Yocto: Building minimal Linux for Aarch64 (ARM-64) with Yocto & test. U-Boot: usage, build, customizing. Boot code walkthru- from U-boot to Linux OS init. Root filesystem- flash types, fs choice. Linux driver model, platform drivers. Device tree.	Course L2 (and preferably L3)	Developers, architects, QA, support staff, working on or intending to work on an embedded Linux project(s). Includes core OS, platform teams, device driver, embedded / server-side Linux teams, building and/or supporting products.	3 days
L5	Debugging Techniques	Advanced	OS Architecture recap. Debugging. Usermode- code browsing, tracing tools, memory debug tools. Using GDB - deep-dive: core dump, dbg without symbolic info, stack analysis (x86/ARM), disassembly. Bug prevention. System monitoring tools. Performance monitoring/analysis. Kernel debug- code-based (printk, ratelimiting, ftrace, procfs, debugfs). Oops generation, analysis. System hangs, panic handling via notification chains. Tools- KGDB, kdb, objdump, Kprobes, Jprobes. Kdump intro and crash tool for analysis.	Courses L2 or L3	Developers, architects, QA, support staff, working on or intending to work on any and all Linux systems project (s)! Includes application layer teams, core OS, platform teams, device driver, embedded / server-side Linux teams, building and/or supporting products. Bugs in software is an unfortunate reality - learning theory, skills and tools to prevent, tackle and debug them is imperative for all involved in systems software.	5 days
L6	Linux for Technical Managers	Intermediate	OS Architecture. <Custom content>.	Course L0 + basic 'C' programming skills	This course has been designed for managers of software projects who would like to gain technical insight into the Linux OS. Will definitely aid in evaluation, project management etc when a good technical background is understood.	1 or 2 days

L7 [2]	Linux OS Security & Hardening [2]	Advanced+Hot	OS Architecture recap. x86 arch, prg model, ARM arch, prg model. Vulnerabilities - focus on BOF, mention of others. Prevention- lang, libraries, compiler prots, DEP/NX, [K]ASLR, tools, testing tech, etc. Case studies. OS Security- traditional UNIX, POSIX Capabilities, Intro to LSMs, LSM stacking. Containers (lxc). Misc- seccomp, OpenWall/KSPP, kernel hardening - settings, etc. <More to come>.	Course L2	This course is targeted mainly at developers and architects of modern products based on the Linux OS. Includes application layer teams, core OS, platform teams, device driver, embedded / server-side Linux teams, building and/or supporting products. The phenomenal rise in technology, and especially, software-driven products (domains like networking, telecom, automotive, infotainment, and now IoT) begs for better security on end-products. Hackers currently have a field day! Learn where vulnerabilities exist, while programming and after; OS hardening techniques; use tools and methodologies to help prevent and mitigate security issues.	2 - 3 days

[1] Can Vary based on customer requirement

[2] Under Construction

Note- can build customized sessions too.