trainsec
where pros train pros.

Syllabus: Linux Internals for Windows Professionals



> *Learn how Linux works from the inside, building on what you already know about Windows.*

Instructor	Pavel Yosifovich
Duration	32 hours: 8 live sessions of 4 hours
Format	Live online course. Ask questions and interact with the trainer during the sessions.
Dates	Dec 14, 17, 21, 28, 2026 and Jan 4, 7, 11, 14, 2027, 11:00-15:00 ET (16:00-20:00 UTC)
Recordings	Lifetime access to the recordings
Certificate	Credly digital certificate to prove your attendance
Price	$1,500, or 10 monthly payments of $157

Course Description

A hands-on course that explains how Linux works from the inside, building on what you already know about Windows. It covers the shell, system calls, processes, memory, threads, and inter-process communication, and builds up to loading and debugging a kernel module. Programming is kept to the essentials: you will mostly read, run, trace, and modify small C programs. Observation and debugging tools are used throughout, with GDB as the main one, first for user-space programs and then for the kernel.

This is a live course. You learn directly from the trainer, ask questions as they come up, and interact with Pavel throughout the sessions. Every attendee gets lifetime access to the recordings, and receives a Credly digital certificate as proof of attendance.

About the Author

Pavel Yosifovich is a software developer, trainer, consultant, author and speaker with over 25 years of experience in the industry. He is the co-author of *Windows Internals, 7th Edition, Part 1* and the author of *Windows Kernel Programming*, *Windows 10 System Programming* and *Windows Native API Programming*. He is one of the world's foremost experts on Windows internals and low-level system architecture, and is known for making complex technical topics practical and accessible.

Pavel has spent his career explaining how an operating system works from the inside. This course brings the same approach to Linux: every Linux concept is introduced through the Windows concept you already know, so you learn Linux from the person who wrote the book on Windows internals.

The contents of this document and the associated course materials are protected by copyright law and may not be reproduced or distributed without explicit permission from TrainSec.net.

Target Audience

Windows system administrators, security and forensics analysts, reverse engineers, driver and platform developers, and support or performance engineers who understand how Windows works and want the same depth on Linux.

Course Goals

By the end of this course, students will be able to:

1. Work confidently at a Linux bash shell: navigation, pipelines, redirection, permissions, basic scripting
2. Explain the user/kernel boundary and how a system call reaches the kernel
3. Read and interpret ELF binaries
4. Describe how Linux processes, file descriptors, signals, threads, and memory work, and where they differ from Windows
5. Investigate a running Linux system using standard tools and GDB
6. Build, load, and debug a basic Linux kernel module and character device

Prerequisites

Required: - Working knowledge of Windows internals concepts: processes and threads, handles, virtual memory, user mode versus kernel mode, and what a device driver is - Ability to read C (structs, pointers, function calls) - Comfort with a command-line environment

Helpful but not required: - Experience with WinDbg or another debugger - Experience with Sysinternals tools - Basic x86-64 assembly familiarity - Prior exposure to Linux or WSL

No prior Linux experience is needed.

Course Modules

Module 1: Bash Shell Fundamentals

Topics covered: - Filesystem navigation and common commands - Pipes and redirection - Shell variables, the environment, wildcards, and quoting - Permissions, ownership, and users - Job control - Getting help - Minimal scripting

Module 2: The Syscall Interface and ELF

Topics covered: - System calls and how they reach the kernel - The C library, static and dynamic linking - ELF file structure - The dynamic linker, PIE executables, and ASLR - Tracing system calls

The contents of this document and the associated course materials are protected by copyright law and may not be reproduced or distributed without explicit permission from TrainSec.net.

Module 3: Processes

Topics covered: - Creating and running programs: `fork`, `exec`, `wait` - Exit status, zombies, and orphans - Process image layout - Process information and the `/proc` filesystem - The process tree and services

Module 4: File Descriptors and I/O

Topics covered: - File descriptors and the descriptor table - Reading, writing, and positioning - Buffered and unbuffered I/O - Duplicating descriptors and redirection - Pipes

Module 5: Signals

Topics covered: - What signals are and how they are delivered - Handling signals safely - Blocking and masking signals - Process groups and job control

Module 6: GDB for Windows Debugger Users

Topics covered: - Breakpoints, watchpoints, and stepping - Inspecting memory, registers, and structures - Core dump analysis - Multi-process and multi-threaded debugging - Customizing and scripting GDB

Module 7: Inter-Process Communication

Topics covered: - Pipes and FIFOs - Shared memory - Message queues - Unix domain sockets - Observing IPC objects

Module 8: Threads and Synchronization

Topics covered: - Threads: what they share and what they do not - Mutexes, condition variables, and read-write locks - The futex - Debugging concurrency bugs in GDB

Module 9: Memory Management

Topics covered: - The process address space - Memory mapping, demand paging, and page faults - Inspecting memory use of a process - Memory overcommit and the OOM killer

Module 10: Kernel Modules

Topics covered: - Kernel module architecture - Building, loading, and unloading modules - The kernel log - Kernel-to-user-space interfaces through `/proc`

Module 11: Character Devices

Topics covered: - Device files and device numbers - Registering a character device - Implementing read, write, and ioctl - Basic kernel synchronization

Module 12: Kernel Debugging with GDB and QEMU (Optional, if time permits)

Topics covered: - Booting a debug kernel under QEMU and attaching GDB - Setting breakpoints in a live kernel module - Reading kernel crash reports

The contents of this document and the associated course materials are protected by copyright law and may not be reproduced or distributed without explicit permission from TrainSec.net.

About TrainSec Academy

TrainSec is an online learning hub for current and aspiring cybersecurity pros who understand the need for excellence. Level up or change career using skill-expanding modular learning way. We turn beginners into experts and experts into masters. TrainSec is designed for those who are serious about being at the vanguard of the latest knowledge, skills and trends. We're already working with the best.

Contact & support

Student support	info@trainsec.net
Discord community	discord.com/invite/qugcNyWdaU
LinkedIn	linkedin.com/company/trainsec
X (Twitter)	x.com/TrainSec
Website	trainsec.net

The contents of this document and the associated course materials are protected by copyright law and may not be reproduced or distributed without explicit permission from TrainSec.net.