Training: The Linux Foundation
LFD441 Security and the Linux Kernel

THE LINUX FOUNDATION AUTHORIZED TRAINING PARTNER

TRAINING GOALS:

This instructor-led course provides an understanding of the Linux kernel security model and the mechanisms used to secure the operating system.

What You'll Learn

- The course covers the fundamentals of Linux kernel security, including memory protection, process management, system calls, and filesystem security. Students will learn about various security mechanisms in the Linux kernel, such as Mandatory Access Control (MAC), Linux Security Modules (LSM), and secureboot. Throughout the course, students will gain hands-on experience in securing both userspace and the Linux kernel through various security mechanisms.

What It Prepares You For

- This course prepares students to be able to secure a system using the various mechanisms and systems available as part of the Linux kernel and operating system. These skills can be used to secure anything from embedded systems, to mobile computers, desktop systems, servers or virtual machines.

Who Is It For

- This course is designed for systems level programmers or kernel engineers who want to learn more about the security options provided by the Linux kernel, as well as userspace developers who want to learn more about Linux kernel security mitigations. Learners should know how to build a Linux kernel, write and use Linux kernel modules, as well as have basic Linux command line and system administration skills.

CONSPECT:

- Introduction
  - Objectives

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Who You Are
- The Linux Foundation{
- Copyright and No Confidential Information
- The Linux Foundation{ Training
- Certification Programs and Digital Badging
- Linux Distributions
- Platforms
- Things Change in Linux and Open Source Projects

- Preliminaries
  - Kernel Versions
  - Kernel Sources and Use of git
- Lab environment
  - Virtual Machine
  - Why proxmox?
  - Our Lab Environment
  - Labs
- How to Work in OSS Projects **
  - Overview on How to Contribute Properly
  - Know Where the Code is Coming From: DCO and CLA
  - Stay Close to Mainline for Security and Quality
  - Study and Understand the Project DNA
  - Figure Out What Itch You Want to Scratch
  - Identify Maintainers and Their Work Flows and Methods
  - Get Early Input and Work in the Open
  - Contribute Incremental Bits, Not Large Code Dumps
  - Leave Your Ego at the Door: Don't Be Thin-Skinned
  - Be Patient, Develop Long Term Relationships, Be Helpful
- Reducing Attack Surfaces
  - Why Security?
  - Types of Security
  - Vulnerabilities
  - Layers of Protection
  - Software Exploits
  - Labs
- Kernel Features
  - Components of the Kernel

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- User-Space vs. Kernel-Space
- What are System Calls?
- Available System Calls
- Scheduling Algorithms and Task Structures
- Process Context
- Labs

- Kernel Deprecated Interfaces
  - Why Deprecated
  - __deprecated
  - BUG() and BUG_ON()
  - Computed Sizes for kmalloc()
  - simple_strtol() Family of Routines
  - strcpy(), strncpy(), strlcpy()
  - printk() %p Format Specifier
  - Variable Length Arrays
  - Switch Case Fall-Through
  - Zero-Length and One-Element Arrays in Structs
- Address Space Layout Randomization (ASLR)
  - Why ASLR?
  - How to Use ASLR
  - Disabling ASLR for Specific Programs
  - Kernel Configuration
  - Kernel Address Space Layout Randomization (KASLR)
  - How KASLR Works
  - Enabling KASLR
  - Labs
- Kernel Structure Layout Randomization
  - Benefits
  - How Structure Randomization Works
  - Structure Initialization
  - Opt-in vs Opt-out
  - Partial Randomization
  - Enabling Structure Randomization
  - Building Out-of-tree Modules with Structure Randomization
- Introduction to Linux Kernel Security
  - Linux Kernel Security Basics

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Discretionary Access Control (DAC)
- POSIX ACLs
- POSIX Capabilities
- Namespaces
- Linux Security Modules (LSM)
- Netfilter
- Cryptographic Methods
- The Kernel Self Protection Project

- CGroups
  - Introduction to CGroups
  - Overview
  - Components of CGroup
  - cgroup initialization
  - cgroup Activation
  - cgroups Parameters
  - Testing cgroups
  - systemd and cgroups
  - Labs
- eBPF
  - BPF
  - eBPF
  - Installation
  - bcc Tools
  - bpftrace
  - Labs
- Seccomp
  - What is seccomp
  - The seccomp Interface
  - seccomp Strict Mode
  - seccomp Filter Mode
  - Labs
- Secure Boot
  - Why Secure Boot?
  - Secure Boot x86
  - Embedded Systems Secure Boot
  - Labs

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Module Signing
  - What is Module Signing?
  - Basics of Signatures
  - Module Signing Keys
  - Enabling Module Signature Verification
  - How It Works
  - Signing Modules
  - Labs
- Integrity Measurement Architecture (IMA)
  - Why IMA?
  - Conceptual Operations
  - Modes of Operation
  - Collect Mode textit {(Collect and Store)
  - Logging Mode textit {(Appraise and Audit)
  - Enforcing Mode textit {(Appraise and Protect)
  - Extended Verification Module (EVM)
  - Labs
- DM-Verity
  - What is dm-verity?
  - How dm-verity Works
  - Enabling dm-verity
  - Setting up dm-verity
  - Using dm-verity
  - Signing with dm-verity
  - Booting with dm-verity
  - Labs
- Encrypted Storage
  - Why Encrypted Storage?
  - Data Encryption Solutions
  - Survey of Storage Encryption Options
  - Block Encryption
  - Block Encryption Use
  - Filesystem Encryption
  - Filesystem Encryption Use
  - Layered Filesystem Encryption
  - Layered Filesystem Encryption Use

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Labs
- Linux Security Modules (LSM)
  - What are Linux Security Modules?
  - LSM Basics
  - LSM Choices
  - How LSM Works
  - An LSM Example: Yama
  - Labs
- SELinux
  - SELinux
  - SELinux Overview
  - SELinux Modes
  - SELinux Policies
  - Context Utilities
  - SELinux and Standard Command Line Tools
  - SELinux Context Inheritance and Preservation**
  - restorecon**
  - semanage fcontext**
  - Using SELinux Booleans**
  - getsebool and setsebool**
  - Troubleshooting Tools
  - Labs
- AppArmor
  - What is AppArmor?
  - Checking Status
  - Modes and Profiles
  - Profiles
  - Utilities
- Yama (LSM)
  - Why Yama?
  - Configuring Yama
  - How Yama Works
  - Labs
- LoadPin (LSM)
  - Why LoadPin?
  - Enabling LoadPin

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Using LoadPin
- How LoadPin Works

- Lockdown
  - Why Lockdown?
  - Lockdown Modes
  - What Things are Locked Down?
  - How It Works
  - A Few Notes
  - Labs
- Safesetid
  - Why Safesetid?
  - Configuring Safesetid
  - How Safesetid Works
  - Labs
- Netfilter
  - What is netfilter?
  - Netfilter Hooks
  - Netfilter Implementation
  - Hooking into Netfilter
  - Iptables
  - nftables
  - Labs
- Netlink Sockets**
  - What are netlink Sockets?
  - Opening a netlink Socket
  - netlink Messages
  - Labs
- Closing and Evaluation Survey
  - Evaluation Survey
- Kernel Architecture I
  - UNIX and Linux **
  - Monolithic and Micro Kernels
  - Object-Oriented Methods
  - Main Kernel Components
  - User-Space and Kernel-Space
- Kernel Programming Preview

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Task Structure
- Memory Allocation
- Transferring Data between User and Kernel Spaces
- Object-Oriented Inheritance - Sort Of
- Linked Lists
- Jiffies
- Labs

- Modules
  - What are Modules?
  - A Trivial Example
  - Compiling Modules
  - Modules vs Built-in
  - Module Utilities
  - Automatic Module Loading
  - Module Usage Count
  - Module Licensing
  - Exporting Symbols
  - Resolving Symbols **
  - Labs
- Kernel Architecture II
  - Processes, Threads, and Tasks
  - Kernel Preemption
  - Real Time Preemption Patch
  - Labs
- Kernel Configuration and Compilation
  - Installation and Layout of the Kernel Source
  - Kernel Browsers
  - Kernel Configuration Files
  - Kernel Building and Makefiles
  - initrd and initramfs
  - Labs
- Kernel Style and General Considerations
  - Coding Style
  - Using Generic Kernel Routines and Methods
  - Making a Kernel Patch
  - sparse

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Using likely() and unlikely()
- Writing Portable Code, CPU, 32/64-bit, Endianness
- Writing for SMP
- Writing for High Memory Systems
- Power Management
- Keeping Security in Mind
- Labs

- Race Conditions and Synchronization Methods
  - Concurrency and Synchronization Methods
  - Atomic Operations
  - Bit Operations
  - Spinlocks
  - Seqlocks
  - Disabling Preemption
  - Mutexes
  - Semaphores
  - Completion Functions
  - Read-Copy-Update (RCU)
  - Reference Counts
  - Labs
- Memory Addressing
  - Virtual Memory Management
  - Systems With and Without MMU and the TLB
  - Memory Addresses
  - High and Low Memory
  - Memory Zones
  - Special Device Nodes
  - NUMA
  - Paging
  - Page Tables
  - page structure
  - Labs
- Memory Allocation
  - Requesting and Releasing Pages
  - Buddy System
  - Slabs and Cache Allocations

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Memory Pools
- kmalloc()
- vmalloc()
- Early Allocations and bootmem()
- Memory Defragmentation
- Labs

** These sections may be considered in part or in whole as optional. They contain either background reference material, specialized topics, or advanced subjects. The instructor may choose to cover or not cover them depending on classroom experience and time constraints.

REQUIREMENTS:

To make the most of this course, you should:

- Be proficient in the C programming language.
- Be familiar with basic Linux (UNIX) utilities such as ls, grep and tar.
- Be comfortable using any of the available text editors (e.g. emacs, vi, etc.).
- Experience with any major Linux distribution is helpful but not strictly required.
- Have experience equivalent to having taken LFD420: Linux Kernel Internals and Development.

Difficulty level



CERTIFICATE:

The participants will obtain certificates signed by The Linux Foundation.

TRAINER:

Certified The Linux Foundation Trainer.

Compendium Centrum Edukacyjne Spółka z o.o.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007