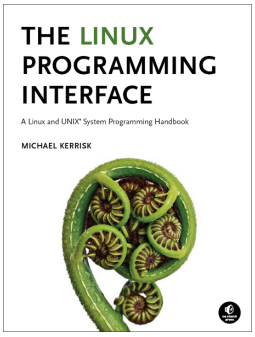
Linux Security and Isolation APIs

Course code: M7D-SECISOL02

This course provides a deep understanding of the low-level Linux features–cgroups (control groups), seccomp, set-UID/set-GID programs, capabilities, and namespaces–used to implement privileged applications and build container, virtualization, and sandboxing technologies. Detailed presentations coupled with carefully designed practical exercises provide participants with the knowledge needed to understand, design, develop, and administer such applications. (The course does *not* cover administering container systems such as Docker and LXC, but by completion of the course you will have a good understanding of various aspects of the underlying implementation and operation of such systems.)



Audience and prerequisites

The primary audience comprises designers and programmers building privileged applications, container applications, and sandboxing applications. Systems administrators who are managing such applications are also likely to find the course of benefit.

Participants should have working knowledge of the fundamental system programming topics covered in the *Linux System Programming Fundamentals* course (M7D-SPINTRO01). This includes file descriptors, file I/O using system calls, signals, and the system calls that define the lifecycle of a process (*fork()*, *exec()*, *wait()*, *exit()*).

Participants should have a good reading knowledge of the C programming language and some programming experience in a language suitable for completing the course exercises (e.g., C, C++, Go, Rust).

Course materials

- A course book (written by the trainer) that includes all course slides and exercises
- A source code tarball containing all of the (many) example programs written by the trainer to accompany the presentation

Course duration and format

Four days, with around 40% of the course time devoted to practical sessions.

Course inquiries and bookings

For inquiries about courses and consulting, you can contact us in the following ways:

- Email: training@man7.org
- Phone: +49 (89) 2155 2990 (German landline)

Prices, dates, and further details

For course prices, upcoming course dates, and further information about the course, please visit the course web page, *http://man7.org/training/secisol/*.

About the trainer

Michael Kerrisk has a unique set of qualifications and experience that ensure that course participants receive training of a very high standard:

- He has been programming on UNIX systems since 1987 and began teaching UNIX system programming courses in 1989.
- He is the author of *The Linux Programming Interface*, a 1550-page book widely acclaimed as the definitive work on Linux system programming.
- He is actively involved in Linux development, working with kernel developers on testing, review, and design of new Linux kernel–user-space APIs.
- Since 2004, he has been the maintainer of the Linux *man-pages* project, which provides the manual pages documenting the Linux kernel–user-space and GNU C library APIs.

Linux Security and Isolation APIs: course contents in detail

Topics marked with an asterisk (*) may be covered, if time permits.

1. **Course Introduction**
2. **Security and Isolation APIs Overview (*)**
   - Sandboxing
   - Containers
3. **Privileged Programs**
   - Process credentials
   - Set-user-ID and set-group-ID programs
   - Changing process credentials
   - A few guidelines for writing privileged programs
4. **Capabilities**
   - Process and file capabilities
   - Setting and viewing file capabilities
   - Text form capabilities
   - Capabilities and *execve()*; further capability sets
   - Ambient capabilities
5. **Capabilities: Further Topics (*)**
   - Root, UID transitions, and capabilities
   - Making a capabilities-only environment: securebits
   - Programming with capabilities
6. **Namespaces**
   - Namespace types
   - Mount namespaces
   - UTS, IPC, cgroup, and network namespaces
   - PID namespaces
7. **Namespaces APIs**
   - API Overview
   - Creating a child process in a new namespace: *clone()*
   - `/proc/PID/ns`
   - Entering a namespace: *setns()*
   - Creating a namespace: *unshare()*
   - PID namespaces idiosyncrasies
   - *ioctl()* operations
   - Namespace lifetime
8. **User Namespaces**
   - Overview of user namespaces
   - Creating and joining a user NS
   - User namespaces: UID and GID mappings
   - User namespaces, *execve()*, and user ID 0
   - Security issues
   - Use cases
   - Combining user namespaces with other namespaces
9. **User Namespaces and Capabilities**
   - User namespaces and capabilities
   - What does it mean to be superuser in a namespace?
   - User namespace "set-UID-root" programs (*)
   - Namespaced file capabilities (*)
10. **Mount Namespaces and Shared Subtrees (*)**
    - Mount namespaces
    - Shared subtrees
    - Bind mounts
    - Peer groups
    - Private mounts
    - Slave mounts
    - Unbindable mounts
11. **Network Namespaces (*)**
    - Introduction
    - Creating and deleting network namespaces
    - Executing commands inside a network namespace
    - Virtual networking devices
    - Connecting namespaces with a veth pair
    - Physical networking devices
    - Using a bridge or switch to connect namespaces
    - Connecting a network namespace to the Internet
    - Use cases for network namespaces
12. **Seccomp**
    - Introduction and history
    - Seccomp filtering and BPF
    - The BPF virtual machine and BPF instructions
    - Checking the architecture
    - BPF filter return values
    - BPF programs
    - Discovering the system calls made by a program
    - Audit logging of filter actions
    - Caveats
    - Productivity aids (*libseccomp* and other tools)
13. **Cgroups (Control Groups)**
    - Introduction to cgroups v1 and v2
    - Cgroups v1: hierarchies and controllers
    - Cgroups v1: populating a cgroup
    - Cgroups v1: release notification
    - Cgroups v1: a survey of the controllers
    - Cgroups `/proc` files
14. **Cgroups Version 2**
    - Problems with cgroups v1; rationale for v2
    - Cgroups v2 controllers
    - Enabling and disabling controllers
    - Organizing cgroups and processes
    - Release notification (`cgroup.events` file)
    - Delegation
15. **Cgroups v2 Thread Mode (*)**
    - Overview of thread mode
    - Creating and using a threaded subtree