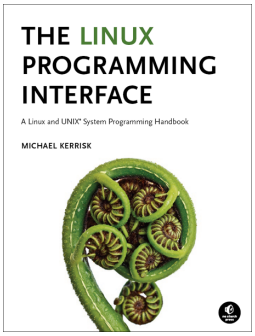
man7.org
Training and Consulting

Linux Capabilities and Namespaces

Course code: M7D-CAPNS01

This course provides an in-depth exploration of Linux namespaces, which are used in a wide array of virtualization and sandboxing technologies such as Docker, LXC, Flatpak, Firejail, Systemd, and various web browsers. In addition, the course covers the Linux capabilities model, since an understanding of that model is essential to understanding the operation of user namespaces, which are a cornerstone of many of the aforementioned applications. Detailed presentations coupled with carefully designed practical exercises provide participants with the knowledge needed to understand, design, develop, and administer such applications.



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

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

Two 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) 2488 6180 (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/capns/`.

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.
- He has more than two decades of experience as a teacher and trainer, and first began teaching UNIX system programming courses in 1989.
- He is the author of *The Linux Programming Interface*, a 1550-page book acclaimed as the definitive work on Linux system programming.
- He has been actively involved in Linux development, working with kernel developers on testing, review, and design of new Linux kernel–user-space APIs.
- Since 2000, he has been the involved in the Linux *man-pages* project, which provides the manual pages documenting Linux system calls and C library APIs, and was the project maintainer from 2004 to 2021.

Linux Capabilities and Namespaces: course contents in detail

Topics marked with an asterisk (*) will be covered as time permits.

1. **Course Introduction**
2. **Classical Privileged Programs**
   - A simple set-user-ID program
   - Saved set-user-ID and and saved set-group-ID
   - Changing process credentials
   - A few guidelines for writing privileged programs
3. **Capabilities**
   - Process and file capabilities
   - Permitted and effective capabilities
   - Setting and viewing file capabilities
   - Capabilities-dumb and capabilities-aware applications
   - Text-form capabilities
4. **Capabilities and *execve()***
   - Capabilities and *execve()*
   - The capability bounding set
   - Inheritable capabilities
   - Summary of process capability sets (so far)
   - Problems with inheritable capabilities
   - Ambient capabilities
   - An alternative summary of process capability sets
   - Summary remarks
5. **Capabilities and UID 0**
   - Capabilities and UID transitions
   - Capabilities, UID 0, and *execve()*
   - Decoupling privilege from UID 0: securebits (*)
6. **Programming with capabilities (*)**
   - Programming with capabilities
7. **Namespaces**
   - An example: UTS namespaces
   - Namespaces commands
   - Namespaces demonstration (UTS namespaces)
   - Namespace types and APIs
   - Namespaces, containers, and virtualization
8. **Mount Namespaces and Shared Subtrees**
   - Mount namespaces
   - Shared subtrees
   - Bind mounts
9. **Mount Namespaces: Further Details (*)**
   - Peer groups
   - Private mounts
   - Slave mounts
   - Unbindable mounts
   - Mounting a container filesystem
10. **PID Namespaces**
    - PID namespaces
11. **Other Namespaces**
    - IPC namespaces
    - Time namespaces
    - Cgroup namespaces
    - Network namespaces
12. **Namespaces APIs**
    - API Overview
    - Creating a child process in new namespaces: *clone()*
    - `/proc/PID/ns`
    - Entering a namespace: *setns()*
    - Creating a namespace: *unshare()*
    - PID namespaces idiosyncrasies
    - Namespace lifetime
13. **User Namespaces**
    - Overview of user namespaces
    - Creating and joining a user namespace
    - UID and GID mappings
    - Rules for defining UID and GID mappings
    - Accessing files (and other objects with UIDs/GIDs)
    - Security issues
    - Combining user namespaces with other namespaces
    - Use cases
14. **User namespaces, *execve()*, and user ID 0**
    - User namespaces, *execve()*, and user ID 0
15. **User Namespaces and Capabilities**
    - User namespaces and capabilities
    - What does it mean to be superuser in a namespace?
    - Discovering namespace relationships
    - File-related capabilities (*)
16. **User Namespaces and Privileged Programs (*)**
    - User namespace "set-UID-root" programs
    - Namespaced file capabilities