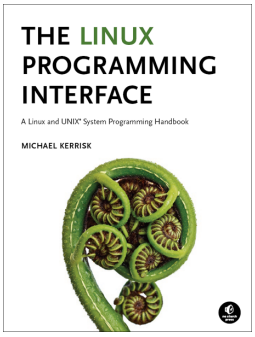
man7.org
Training and Consulting

Linux/UNIX Network Programming

Course code: M7D-NWP01

This course covers network programming using the sockets API on Linux and UNIX systems. Topics covered include: the sockets API; TCP/IP fundamentals; sockets programming in the UNIX and Internet domains; client-server design models; TCP in detail; troubleshooting and monitoring; and alternative I/O models (*poll()*, *epoll*, *select()*). Detailed presentations coupled with many carefully designed practical exercises provide participants with the knowledge needed to write complex network applications. The course touches on some topics specific to Linux, but most of the material is applicable on all UNIX implementations, so that it is also valuable to developers working on other systems.



Audience and prerequisites

The primary audience for this course is programmers developing network applications for Linux and UNIX systems, or programmers porting such applications from other operating systems (e.g., Windows) to Linux or UNIX. By the completion of the course, participants will have the understanding needed to write advanced network applications on a Linux or UNIX system.

To get the most out of the course, participants should have:

- Good reading knowledge of the C programming language
- Solid programming experience in a language suitable for completing the course exercises (e.g., C, C++, D, Go, Rust, or Python)
- Knowledge of basic UNIX/Linux shell commands

Previous network programming experience is *not* required.

Course duration and format

Three days, with up to 50% devoted to practical sessions.

Course materials

- A course book (written by the trainer) that includes all slides and exercises presented in the course
- A copy of the trainer's book, *The Linux Programming Interface*
- A source code tarball containing a large set of example programs written by the trainer

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 and further details

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

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/UNIX Network Programming: course contents in detail

Topics marked with an asterisk (*) are optional, and will be covered as time permits

1. **Background pieces of the UNIX API**
   - Error handling
   - File descriptors; duplicating file descriptors
   - Process lifecycle (*fork()*, *exit()*, *waitpid()*)
   - Executing programs (*execve()*)
   - Signals
2. **Introduction to sockets**
   - Sockets system calls (*socket()*, *bind()*, *listen()*, *accept()*, *connect()*)
   - Stream sockets
   - Stream socket I/O
   - Datagram sockets
   - Datagram socket I/O (*sendto()*, *recvfrom()*)
3. **UNIX Domain sockets**
   - UNIX domain socket addresses
   - Socket permissions
   - Programming with stream and datagram sockets in the UNIX domain
   - Creating a socket pair: *socketpair()*
4. **TCP/IP fundamentals**
   - Protocols and layers: IP, UDP, and TCP
   - IP addresses
   - Port numbers
5. **Internet domain sockets**
   - Internet domain socket addresses
   - Data representation issues (byte order, data marshalling)
   - Domain Name System
   - `/etc/services` file
   - Programming with stream and datagram sockets in the Internet domain
   - Converting host and service names between symbolic and binary form: *getaddrinfo()* and *getnameinfo()*
6. **Client-server design**
   - Concurrent versus iterative server model
   - Multiprocess versus multithreaded model
   - Other server design models
7. **Advanced features of the sockets API**
   - *recv()* and *send()* system calls
   - *shutdown()*
   - *sendfile()* system call
   - Socket options
   - Passing file descriptors between processes
   - *getpeername()* and *getsockname()* system calls
8. **TCP in more detail**
   - TCP segments
   - Sequence numbers and acknowledgements
   - TCP socket options
   - TCP state machine
   - TCP connection establishment and termination
   - TCP socket options
   - Choosing between TCP and UDP
9. **Troubleshooting and monitoring**
   - *netstat*
   - *tcpdump* and *wireshark*
10. **Alternative I/O models**
    - Nonblocking I/O
    - Signal-driven I/O
    - *poll()* and *select()*
    - The *epoll* API
11. **Raw sockets (*)**
    - Uses of raw sockets
    - IP and UDP headers
    - Populating protocol headers
    - Walkthrough of IP and UDP examples
12. **Daemons (*)**
    - Steps in creating a daemon
    - Reinitializing a daemon