Introduction to the Linux Kernel

Praktikum Kernel Programming

University of Hamburg

Scientific Computing

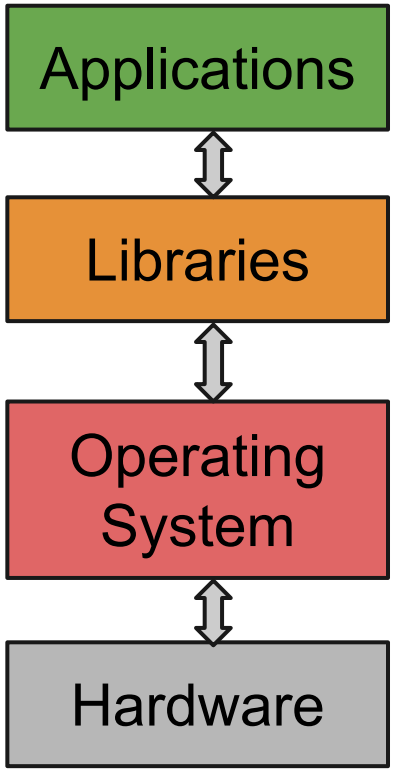
Winter semester 2014/2015

Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
- Summary

What is an OS

- Hard to define
- Abstracts a set of hardware resources
  - High level interface instead of machine code
    - e.g File storage from block devices
- Resource management
  - Multiplexing (sharing) resources
    - e.g Assign CPU time to applications

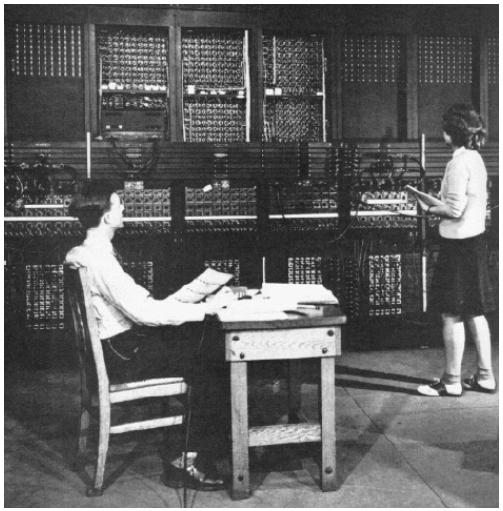


Outline

- What is an Operating System
- ➢ History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
- Summary

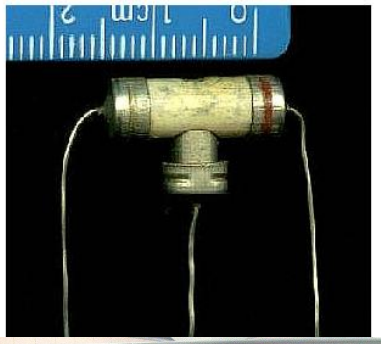
1st Generation

- Vacuum Tubes (1945-55)
  - ~20.000 vacuum tubes where used
  - Programming was done in absolute machine code
  - Assembly language was unknown
  - Each program used the machine exclusively
  - Most famous ENIAC
    - Announced in 1946
    - Solve large class numerical problems



2nd Generation

- Transistors and batch systems (1955-65)
  - Designers / Builders / Operators / Programmers / Mainterers
  - Programmers first wrote the program in paper, then punch it on cards
  - Card readers to read the program source
  - Output stored on tapes and also printed
  - 1st use of Compilers (FORTRAN)



3rd Generation

- ICs and Multiprogramming (1965-1980)
  - IBM 360 Mainframe
    - Multiprogramming
      - Several programs in memory at onces with separate memory Overlap I/O with Computation
  - Timesharing
    - Each user has an online terminal
    - CTSS (Compatible Time Sharing System)
    - MULTICS (MULTiplex Information and Computing System)
    - UNIX, a stripped-down version of MULTICS
    - BSC (Berkeley Software Distribution)

4th Generation

- Personal Computers (1980-today)
  - SYSTEM V, 1st commercial UNIX operating System (1983)
  - LSI (Large Scale Integration)
  - IBM PC (early 1980)
    - Intel 80286 CPU
    - DOS (Disk Operating System)
    - MS-DOS (Microsoft DOS)
  - LISA
    - First Computer with GUI
    - Protected memory, preemptive multitasking,

Modern Operating Systems

Outline

- What is an Operating System
- History of Operating Systems
- ➢ Types of Operating Systems
- The Linux Kernel
- Summary

Types of OS's

- Multi-user
  - Multiple users access the computer simultaneously
- Single-tasking
  - Only one running program
- Multi-tasking
  - Allows more than one program to run parallel
  - Two types:
    - Pre-emptive, the OS slices the CPU time and dedicates one slot to each of the programs
    - Co-operative, each process give time to the others
- Real-time
  - Aims at executing real-time applications

Types of OS's

- Distributed
  - Manages a group of independent computers and makes them appear to be a single computer
- Templated
  - A single virtual machine image as a guest operating system, then saving it as a tool for multiple running virtual machines
- Embedded
  - Designed to be used in embedded computer systems

Monolithic kernel

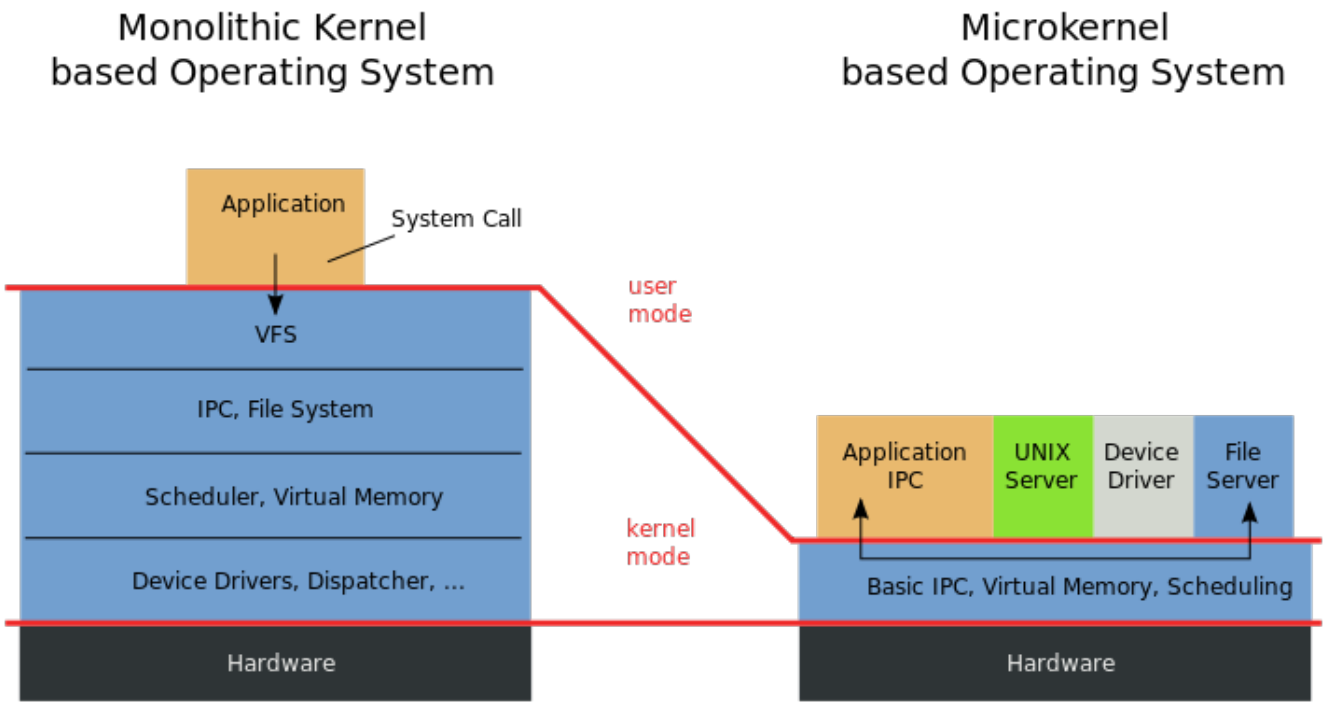
- Single image that runs in a single address space
  - A set of primitives operations are implemented in operating system level
    - Process management
    - Memory management
    - Device Drivers
  - Trivial (IPC) Inter Process Communication
  - Easy to design
  - Difficult to maintain and extend
  - Examples:
    - MULTICS, SunOS, Linux, BSD

Micro-kernel

- The minimum amount of software that provide the mechanisms needed to implement an OS
  - Also known as μ-kernel
  - Provides
    - Build in IPC
    - Low level address space management
    - Thread management
  - Easy to extend
  - Performance penalties (requires IPC calls)
  - Examples
    - Symbian, Mac OS, WinNT

Monolithic VS. μ-kernel

Everything that runs in kernel mode defines the OS



Source: http://en.wikipedia.org/wiki/Microkernel#mediaviewer/File:OS-structure.svg

Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- ➢ The Linux Kernel
  - Introduction (story, licence, versioning)
  - Main parts
  - Loadable Kernel Modules
  - System Calls
  - Security
- Summary

Introduction

- Developed by Linus Torvalds (1991)
  - Just for Fun: The Story of an Accidental Revolutionary by Linus Torvalds
- Based on Unix
- 1st version supported Intel 80386
- Currently various platforms are supported
- Implemented in GNU C
- Several Distributions (distro)
  - RedHat, CentOS, Ubuntu, SUSE, Debian, Arch
  - Different package system, configuration etc.
  - Apply different patches

Introduction (cont.)

- X-Server is not implemented within the Kernel
- Everything run in “Kernel mode”
  - Privileged access to hardware
- Monolithic but boasts modular design
  - Kernel preemption (under certain conditions)
    - The scheduler is permitted to forcibly perform a context switch
  - Supports kernel threads
  - Dynamic load and unload binaries (kernel modules)
  - Reentrant, several processes can be in kernel mode simultaneously

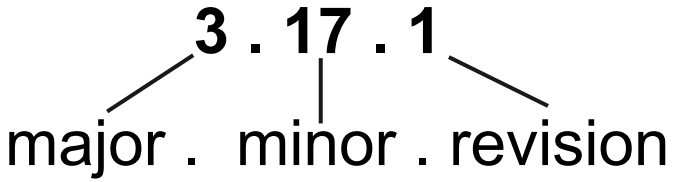
Introduction (cont.)

- License Terms
  - is licensed under the Version 2 of the GNU General Public License (GPL)
  - Allows anybody to redistribute and even sell a product covered by GPL as long as the recipient has access to the source and is able to exercise the same rights
  - Any software derived by a product covered by GPL must be released under the GPL
- Democratize, everyone can contribute
  - If you want your code to go into the mainline or you have modified the kernel then you have to use GPL-compatible license

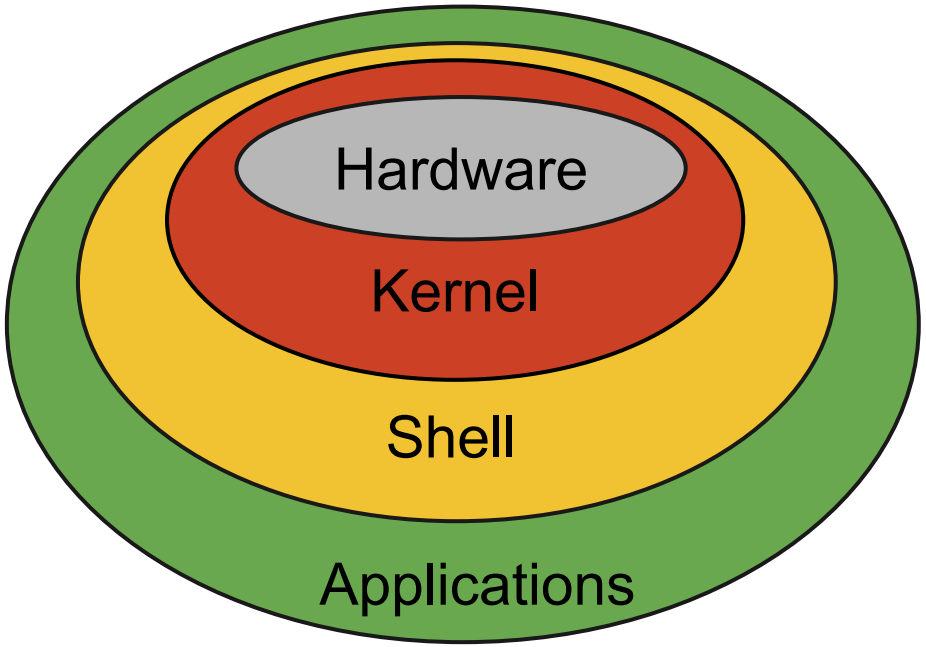
Introduction (cont.)

- Use of binary Blobs (Modules, firmware)
  - The source is not given
  - May contain part of the driver from another file system
  - If the code has been ported from another operating system is legal
  - If a company wants to keep the source private
  - Using such software is discourage
- Versioning

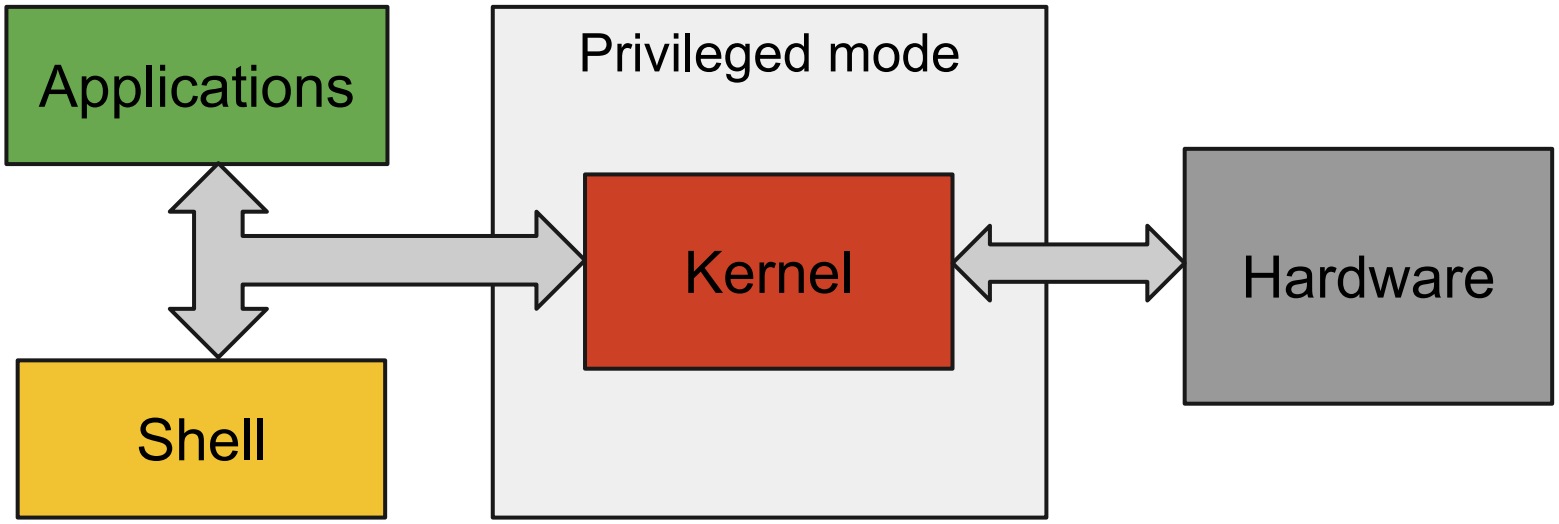
  $ uname -a



Linux system overview



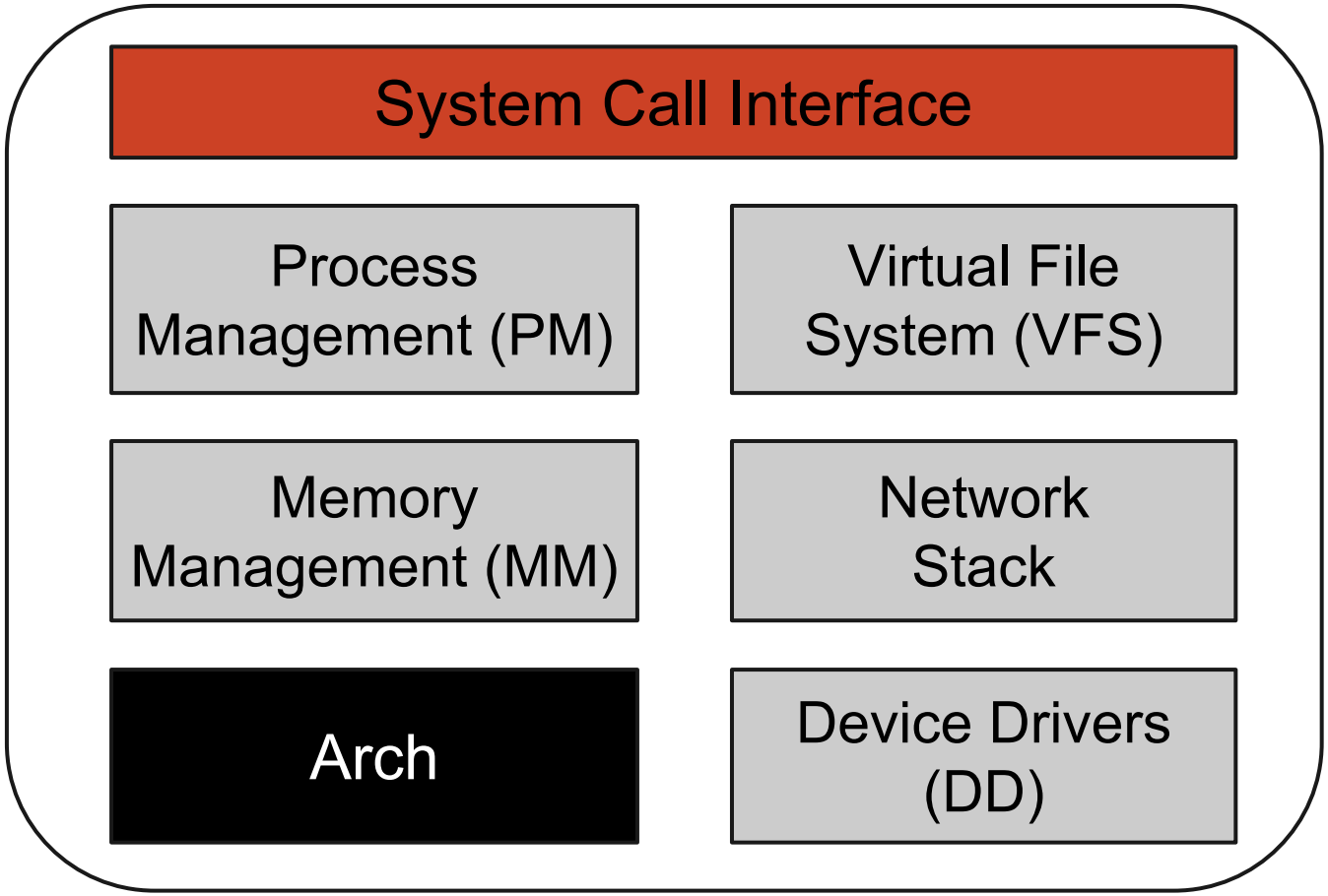
Request flow



Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
  - Introduction (story, licence, versioning)
  - ➢ Main parts
  - Loadable Kernel Modules
  - System Calls
  - Security
- Summary

Main parts

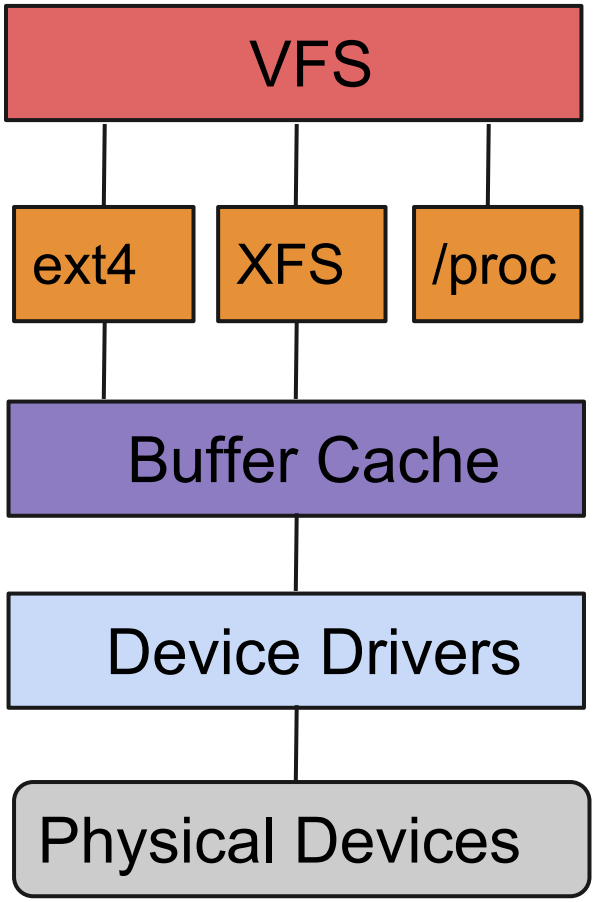

System Call Interface
Process Management (PM)
Virtual File System (VFS)
Memory Management (MM)
Network Stack
Arch
Device Drivers (DD)

Main parts (cont.)

- System call interface (SCI)
  - A thin layer that provides a method to interact from user space to kernel space
- Process Management (PM)
  - Create, destroy processes
  - Communication between different processes (kernel threads)
  - CPU scheduling
- Memory Management (MM)
  - Physical to virtual memory management
  - Memory allocation
  - Swapping, from memory to hard disk

Main parts -- I/O Path

- Virtual File System (VFS)
  - Eports the common file interface
  - Abstract file system functionality from implementation
- File Systems
  - Implementation of FS functionality
- Buffer Cache
  - A set of functions to manipulate main memory designed for FS
- Device Driver
- Physical Device
  - Where data live



Main parts (cont.)

- Network Stack
  - Implement the network protocols
  - Deliver packets across programs and network interfaces
- Device Drivers (DD)
  - Interact with the hardware
  - Extract an abstraction of the device functionalities
- Arch
  - Architecture dependent code

Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
  - Introduction (story, licence, versioning)
  - Main parts
  - ➢ Loadable Kernel Modules
  - System Calls
  - Security
- Summary

LKMs

- LKMs (Loadable Kernel Modules)
- Pre-compiled binary pieces
- Each piece is called “module”
- Can be loaded at runtime
- Extend the functionality of the system
- Enforce modularity
  - Easy to develop, debug and maintain
  - No need to rebuild the kernel
- Can save memory (load only the necessary)

What are LKMs used for

- Everything that is not required in the core
- 6 main categories
  - Device drivers
  - File system drivers
    - Implementation of a specific file system
  - System calls
  - Network stack
    - Interprets a network protocol
  - TTY line disciplines
  - Executable interpreters for the supported formats

Character Device Driver

- Read or Write a byte at a time
- Accessed by a stream of bytes
- Usually permit only sequential access
- Implement: open, close, read, write
- Similar to regular files
- Examples:
  - /dev/console
  - /dev/ttyS0

Block Device Driver

- Read or Write block-size multiples
- Permit random access
- Accessed in the /dev/
- File systems can be mount on top
- Handle I/O operations
- Differ with the char module in the way the manage data inside the kernel
- Different interface to the kernel than char modules

Network Drivers

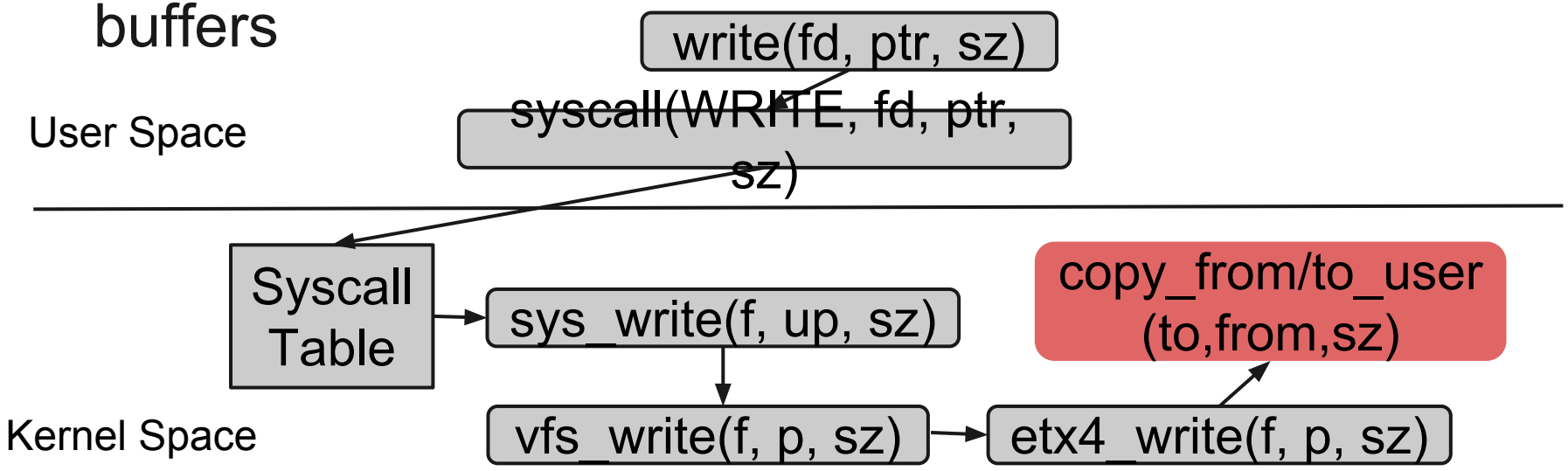
- Handle any network transaction made
- Transfer packets of data
- Independent of a specific protocol
- Reception and Transmission instead of Read/Write
- Usually the interface is a hardware device but it can also be software like the loopback
  - loopback is used to communicate with the servers that run in the same node, debugging etc.
- They are not mapped to the file system; they are identified by a name

Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
  - Introduction (story, licence, versioning)
  - Main parts
  - Loadable Kernel Modules
  - ➢ System Calls
  - Security
- Summary

System calls

- A syscall causes a programmed exception (trap) on the CPU
  - syscall(number, arguments)
- Within the kernel you cannot access user space buffers



Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
  - Introduction (story, licence, versioning)
  - Main parts
  - Loadable Kernel Modules
  - System Calls
  - ➢ Security
- Summary

Security considerations

- Security check is enforced by the kernel
- If the Kernel has „holes“ ⇨ System has holes
- Avoid introducing typical programming bugs
  - Module parameters
  - Buffer overrun
  - Memory corruption
- Zero or initialize memory given to user
- Run precompiled kernels found in your distro
- In official distros only the superuser can load and unload modules

Outline

- What is an Operating System
- History of Operating Systems
- Types of Operating Systems
- The Linux Kernel
- ➢ Summary

Summary

- Definition of the Operating system
  - Exports hardware functionality
  - Resource manager
- Main types of OS’s
  - Multi-user
  - Multi-tasking
  - Single-tasking
  - Real-time
  - Embeded
  - Micro-kernel
  - Macro-kernel

Summary

- Linux
  - Follows Unix principles
  - Monolithic with Loadable modules
  - Main parts:
    - System Call Interface
    - Process Management (PM)
    - Virtual File System (VFS)
    - Memory Management (MM)
    - Network Stack
    - Device Drivers
    - Arch

Kernel programming is vital for as long as new hardware is being designed and produced or old-obsolete hardware is maintained.