

The Linux System

- **History**
- **Design Principles**
- **Kernel Modules**
- **Process Management**
- **Scheduling**
- **Memory Management**
- **File Systems**
- **Input and Output**
- **Interprocess Communication**
- **Network Structure**
- **Security**

History

- **Linux is a modem,**
 - ☞ free operating system
 - ☞ based on UNIX standards.
- **First developed as a small**
 - ☞ but **self-contained kernel in 1991** by Linus Torvalds,
 - ☞ with the major design goal of UNIX compatibility.
- **Its history has been one of collaboration**
 - ☞ by many users from all around the world,
 - ☞ corresponding almost exclusively over the Internet.
- **It has been designed to run efficiently and reliably**
 - ☞ on common PC hardware,
 - ☞ but also runs on a variety of other platforms.
- **The core Linux kernel is entirely original,**
 - ☞ but it can run much existing free UNIX software,
 - ☞ resulting in an entire UNIX-compatible operating system
 - ☞ free from proprietary code.

Linux Kernels

■ **Version 0.01 (May 1991)**

- ☞ had no networking,
- ☞ ran only on 80386-compatible Intel processors and on PC hardware,
- ☞ had extremely limited device-driver support, and
- ☞ supported only the Minix file system.

■ **Linux 1.0 (March 1994) included these new features:**

- ☞ Support for UNIX's standard TCP/IP networking protocols
- ☞ BSD-compatible socket interface for networking programming
- ☞ Device-driver support for running IP over an Ethernet
- ☞ Enhanced file system
- ☞ Support for a range of SCSI controllers
 - 📄 for high-performance disk access
- ☞ Extra hardware support

■ **Version 1.2 (March 1995) was the final PC-only Linux kernel.**

Linux 2.0

- Released in *June 1996*, 2.0 added two major new capabilities:
 - ☞ Support for multiple architectures,
 - 📄 including a fully 64-bit native Alpha port.
 - ☞ Support for **multiprocessor architectures**
- Other new features included:
 - ☞ Improved memory-management code
 - ☞ Improved TCP/IP performance
 - ☞ Support for internal kernel threads,
 - ☞ for handling dependencies between loadable modules,
 - 📄 and for automatic loading of modules on demand.
 - ☞ Standardized configuration interface
- Available for next CPU-systems
 - ☞ Motorola 68000-series processors,
 - ☞ Sun Sparc systems,
 - ☞ PC (Intel)
 - ☞ PowerMac systems.

The Linux System

- **Linux uses many tools developed as part of :**
 - ☞ **Berkeley's BSD** operating system,
 - ☞ **MIT's X Window System**, and the
 - ☞ **Free Software Foundation's GNU project**.
- **The min system libraries**
 - ☞ **were started by the GNU project**,
 - ☞ **with improvements provided by the Linux community**.
- **Linux networking-administration tools**
 - ☞ **were derived from 4.3BSD code**;
 - ☞ **recent BSD derivatives**
 - ☞ **such as Free BSD have borrowed code from Linux in return**.
- **The Linux system is maintained by**
 - ☞ **a loose network of developers collaborating over the Internet**,
 - ☞ **with a small number of public ftp sites**
 - ☞ **acting as de facto standard repositories**.

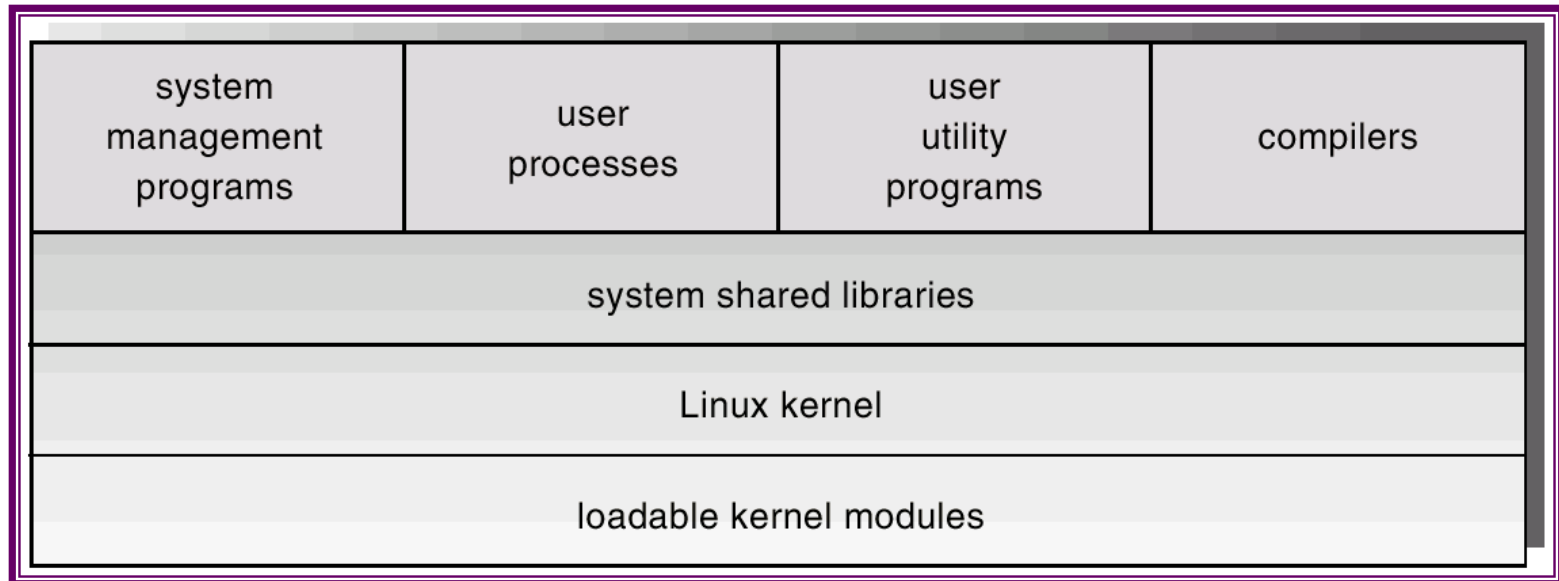
Linux Distributions

- **Distributions = Standard, precompiled sets of package**
- **Distributions include**
 - ☞ basic Linux system
 - ☞ system installation and management utilities
 - ☞ ready-to-install packages of common UNIX tools.
- **First distributions**
 - ☞ managed these packages
 - ☞ by simply providing a means of unpacking all the files
 - ☞ into the appropriate places
- **Modern distributions include advanced package management.**
- **Early distributions included SLS and Slackware.**
- **Popular distributions from commercial and noncommercial sources:**
 - ☞ Red Hat and Debian
 - ☞ Caldera, Craftworks, WorkGroup Solutions
 - ☞ SuSE, Unifix
- **The RPM Package file format**
 - ☞ permits compatibility
 - ☞ among the various Linux distributions.

Linux Licensing

- The **Linux kernel** is distributed
 - ☞ under the **GNU General Public License** (GPL),
 - ☞ the terms of which are set out
 - ☞ by the **Free Software Foundation**.
- **Linux is free software**
- **No product proprietary**
- **Source code must be available**
- **Anyone using Linux,**
- **or creating their own derivative of Linux,**
- **may not make the derived product proprietary;**
- **software released under the GPL**
 - ☞ may not be redistributed
 - ☞ as a binary-only product.

Components of a Linux System



Process Scheduling

- Linux uses two process-scheduling algorithms:
 - ☞ 1. **time-sharing algorithm** for fair preemptive scheduling between multiple processes
 - ☞ 2. **real-time algorithm** for tasks where **absolute priorities** are more important than fairness
- A process's scheduling class defines which algorithm to apply.
- For **time-sharing processes**, Linux uses a prioritized, credit based algorithm.
 - ☞ The crediting rule

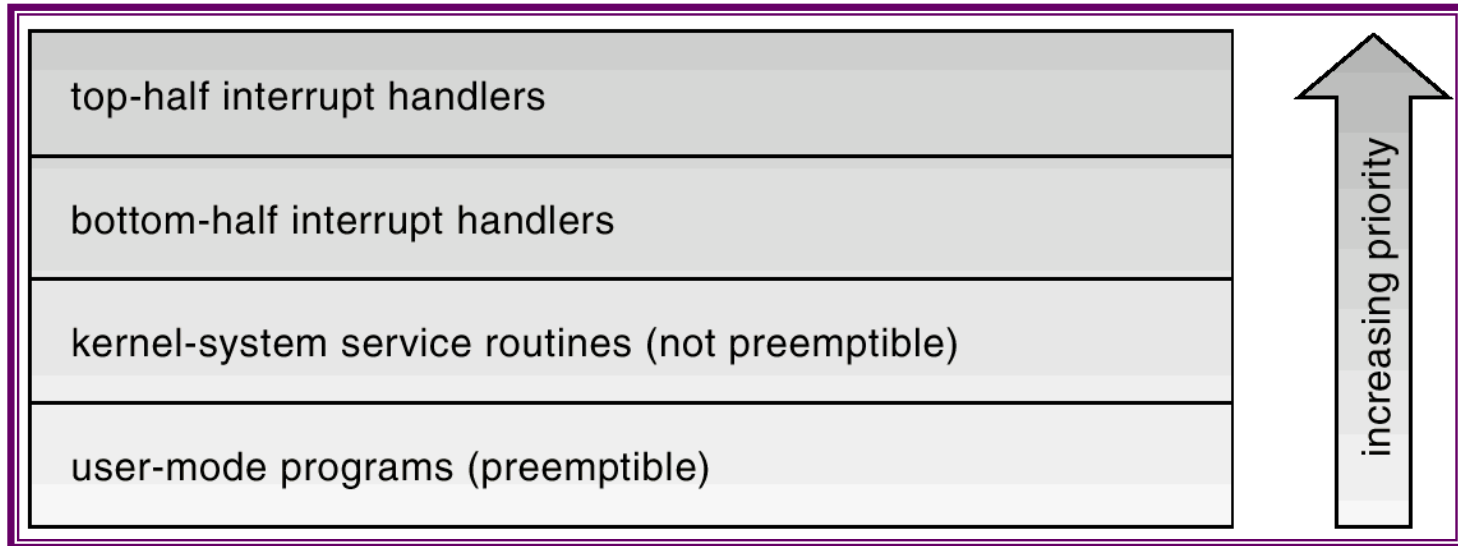
$$\text{credits} := \frac{\text{credits}}{2} + \text{priority}$$

- ☞ factors in both the process's history and its priority.
- ☞ This crediting system automatically prioritizes  interactive or I/O-bound processes.

Kernel Synchronization (Cont.)

- Linux uses two techniques to **protect critical sections**:
 1. **Normal kernel code is non-preemptible**
when a time interrupt is received
while a process is executing a kernel system service routine,
the kernel's `need_resched` flag is set
so that the scheduler will run
once the system call has completed and
control is about to be returned to user mode.
 2. The **second technique applies** to **critical sections**
that occur in an interrupt service routines.
By using the processor's interrupt control hardware
to disable interrupts during a critical section,
the kernel guarantees
that it can proceed
without the risk of concurrent access of shared data structures.

Interrupt Protection Levels



- **Each level may be interrupted by code running at a higher level, but will never be interrupted by code running at the same or a lower level.**
- **User processes can always be preempted by another process when a time-sharing scheduling interrupt occurs.**

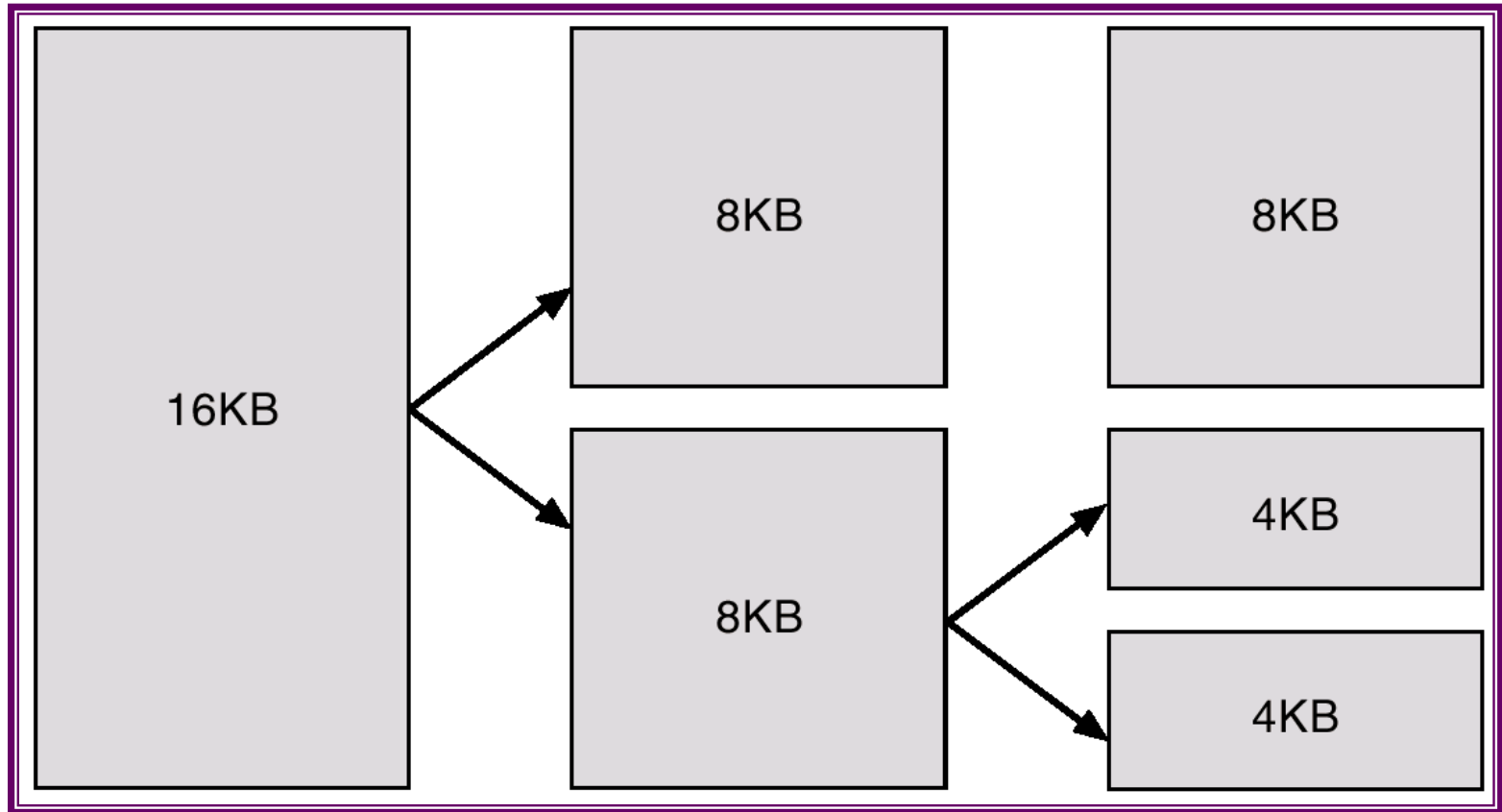
Symmetric Multiprocessing

- **Linux 2.0 was the first Linux kernel**
 - ☞ to support **SMP hardware**;
 - ☞ **separate processes or threads**
 - ☞ can **execute in parallel**
 - ☞ **on separate processors.**
- **To preserve the kernel's**
- **non-preemptible synchronization requirements,**
 - ☞ **SMP imposes the restriction,**
 - ☞ **via a single kernel spinlock,**
 - ☞ **that only one processor at a time**
 - ☞ **may execute kernel-mode code.**

Memory Management

- **Linux's physical memory-management system**
- **deals with**
- **allocating and freeing**
 - ☞ **pages** (normal extent)
 - ☞ **groups of pages** (large extent)
 - ☞ **small blocks of memory** (small extent)
- **It has additional mechanisms for handling**
 - ☞ **virtual memory,**
 - ☞ **memory mapped into the address space of running processes.**

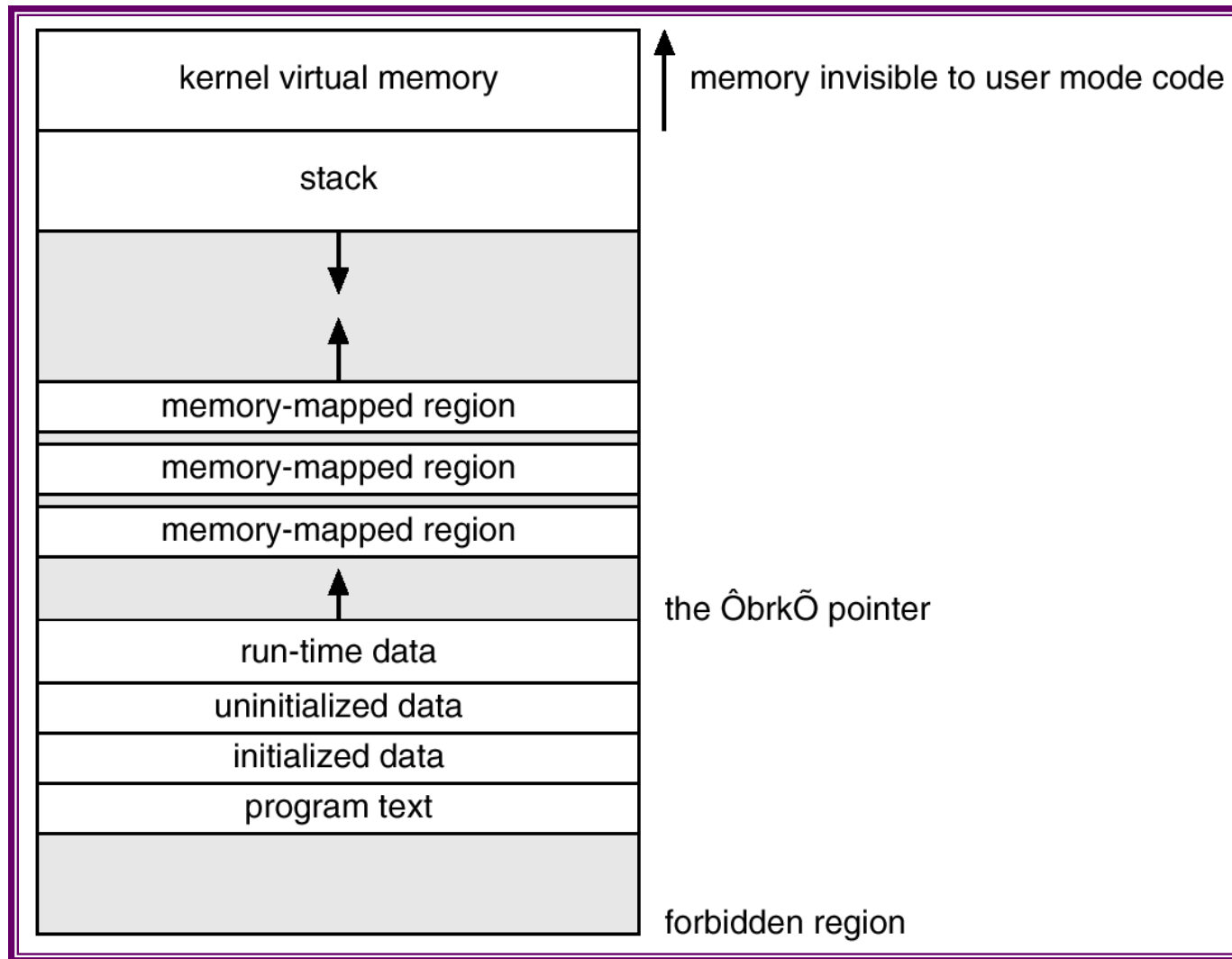
Splitting of Memory in a Buddy Heap



Virtual Memory (Cont.)

- The VM paging system
 - ☞ relocates pages of memory
 - ☞ from physical memory out to disk
 - ☞ when the memory is needed for something else.
- The VM paging system can be divided into two sections:
 - ☞ 1. page-out-policy algorithm
 - ☞ decides
 - ☞ which pages to write out to disk,
 - ☞ and
 - ☞ when. (LFU)
 - ☞ 2. paging mechanism
 - ☞ actually carries out the transfer, and
 - ☞ pages data back, into physical memory as needed.

Memory Layout for ELF Programs

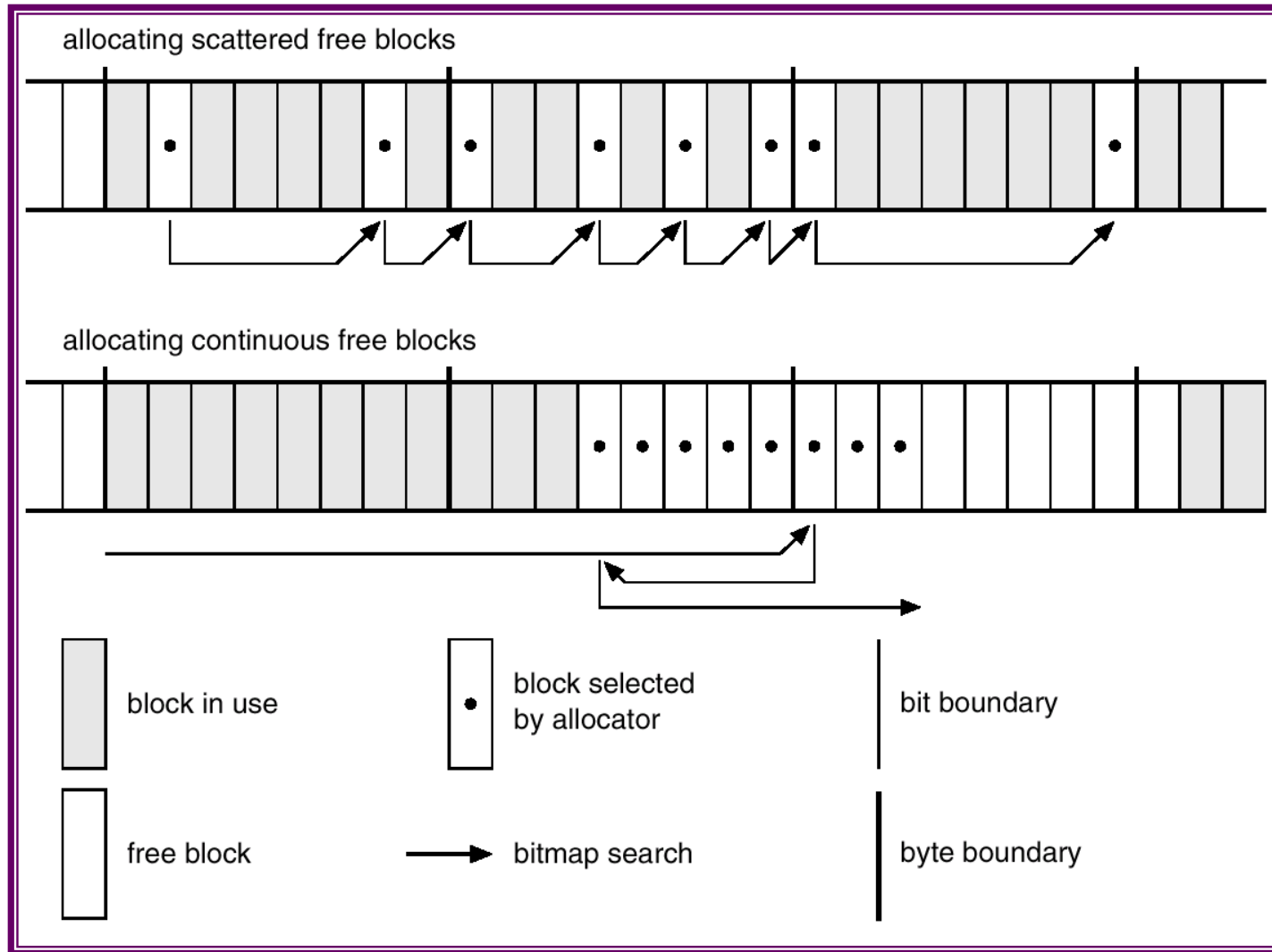


The Linux Ext2fs File System

- **Ext2fs uses allocation policies**
 - ☞ **designed to place**
 - ☞ **logically adjacent blocks of a file**
 - ☞ **into physically adjacent blocks on disk,**
 - ☞ **so that it can submit**
 - ☞ **an I/O request for several disk blocks**
 - ☞ **as a single operation.**

- **Block group**
- **similar as**
- **BDS's cylinder group**

Ext2fs Block-Allocation Policies



The Linux Proc File System

■ Proc FS contain

☞ **Process information:**

- ☞ each directory symbolizes an process
- ☞ Dir name = PID

☞ **Extra directory**

- ☞ Various statistics about kernel or loaded drivers

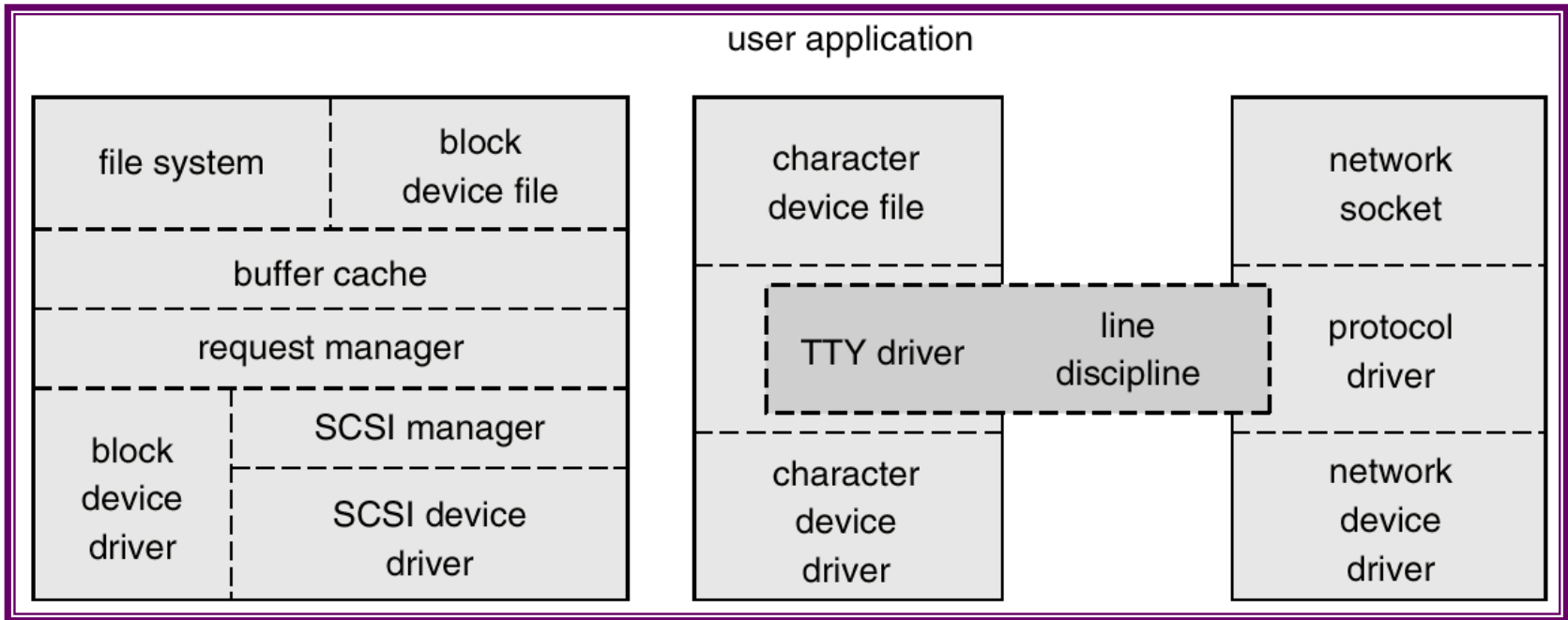
☞ **Kernel variables and kernel tuning**

- ☞ `/proc/sys`

■ The proc file system

- ☞ does not store data, rather,
- ☞ its contents are computed on demand
- ☞ according to user file I/O requests.

Input output



Interprocess Communication

- **IPC =**
 - ☞ Letting another process that some event has occurred
 - ☞ Transferring data from one process to another
- Like **UNIX**, Linux informs processes
 - ☞ that an event has occurred via signals.
 - ☞ Signals can be sent from any process to any other process
- There is a limited number of signals,
 - ☞ and they cannot carry information:
 - ☞ Only the fact that a signal occurred
 - ☞ is available to a process.
- The **Linux kernel does not use signals**
 - ☞ to communicate with processes
 - ☞ with are running in kernel mode,
 - ☞ rather, communication within the kernel
 - ☞ is accomplished
 - ☞ via scheduling states and **wait.queue structures**.

Passing Data Between Processes

■ The **pipe mechanism**

- ☞ allows a child process
- ☞ to inherit a communication channel to its parent,
- ☞ data written to one end of the pipe
- ☞ can be read at the other.

■ **Shared memory**

- ☞ offers an extremely fast way of communicating;
- ☞ any data written by one process
- ☞ to a shared memory region
- ☞ can be read immediately by any other process
- ☞ that has mapped that region into its address space.

■ To obtain synchronization, however,

- ☞ shared memory must be used in conjunction
- ☞ with
- ☞ another Interprocess-communication mechanism.

Network Structure

- **Networking** is a **key area** of functionality for Linux.
- It supports the standard Internet protocols for **UNIX to UNIX** communications.
- It also implements protocols native to non-UNIX operating systems,
- in particular, protocols used on PC networks, such as
 - ☞ **AppleTalk**
 - ☞ **IPX**
 - ☞ **SMB**
- Internally,
- **networking** in the Linux kernel is implemented
- **by 3 layers of software:**
 - ☞ **1. socket interface**
 - ☞ **2. protocol drivers**
 - ☞ **3. network device drivers**

Network Structure (Cont.)

- The most important set of protocols
- in the Linux networking system is the IP
 - ☞ It implements routing
 - 📄 between different hosts anywhere on the network.
 - ☞ On top of the routing protocol
 - 📄 are built the **UDP**, **TCP** and **ICMP** protocols.
- Linux as
 - ☞ Router
 - ☞ Firewall
 - ☞ Web server
 - ☞ Mail server