Linux Introduction

Module-2
OSSD
B.Tech.(CSE-6th Sem)

JIIT, Noida

History

- Unix-like operating system, based on a kernel written by Linus Benedict Torvald
  - Inspired by the Minix operating system
  - The GNU "guh-NEW“ project started in 1983 by Richard Stallman at the Massachusetts Institute of Technology
  - A UNIX-compatible software system developed by the Free Software Foundation (FSF)

History

- Philosophy of GNU
  - To produce software that is non-proprietary
  - Anyone can download, modify and redistribute GNU software
  - The only restriction is that they cannot limit further redistribution
  - Given away for free.
- Symbol of Linux
  - Linux doesn't have a formidable serious looking symbol
  - Rather Tux, as the penguin is lovingly called
  - Symbolizes the care-free attitude of the total movement

History

- Linux is a modern, 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 operating system kernel is entirely original, but it can run much existing free

Exciting Facts: Usage Statistics

- Normal Desktops
  - In January 2021, 1.93% of all desktop operating systems worldwide ran on Linux.
  - In February 2021, Linux held a market share of 0.81 of the global desktop/tablet/console market.
  - In January 2021, the net market share of Linux was 2.35%.

- Specialized Computers
  - In 2021, 100% of the world’s top 500 supercomputers run on Linux.
  - Out of the top 25 websites in the world, only 2 aren’t using Linux.
  - 96.3% of the world’s top 1 million servers run on Linux.
  - 90% of all cloud infrastructure operates on Linux and practically all the best cloud hosts use it.

Linux Philosophy

- Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new features.
- Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

Linux Distributions

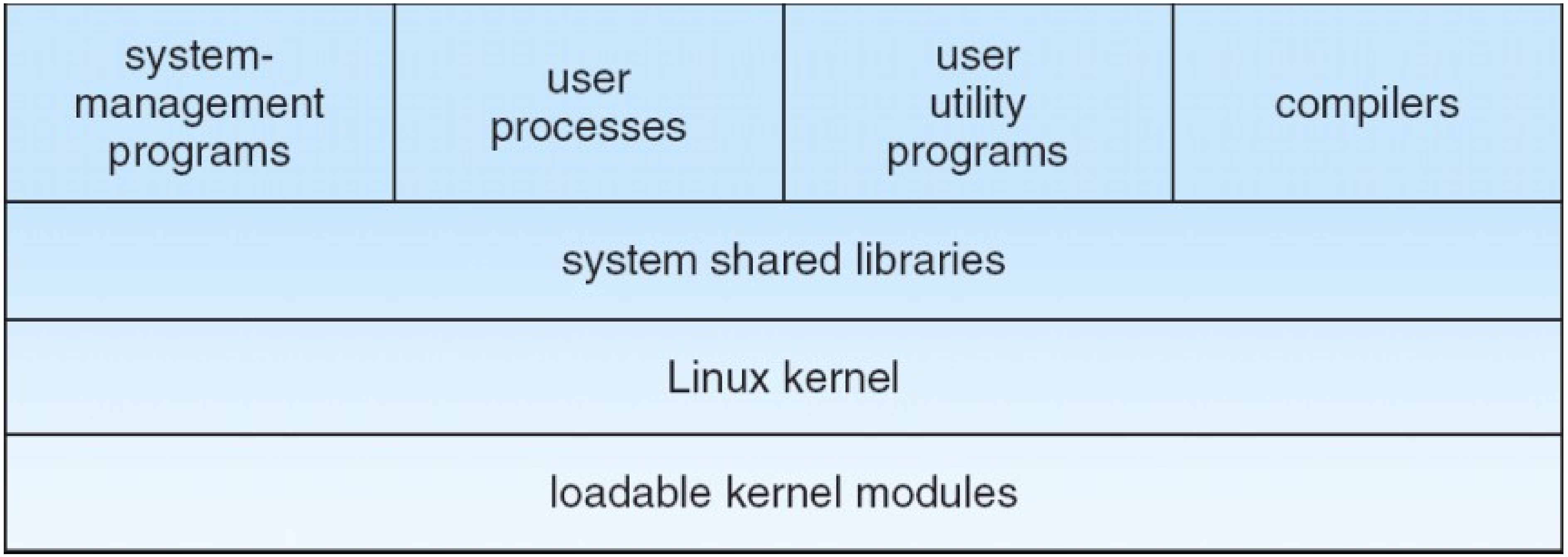
Linux Licensing

- Linux kernel is distributed under GNU General Public License (GPL)
  - GPL is defined by the Free Software Foundation
- GPL implications:
  - anyone using Linux, or creating their own derivative of Linux, may not make the derived (public) product proprietary
  - software released under GPL may not be redistributed as binary-only
- LGPL: Lesser GPL
  - allow non-(L)GPL software to link to LGPL licensed software

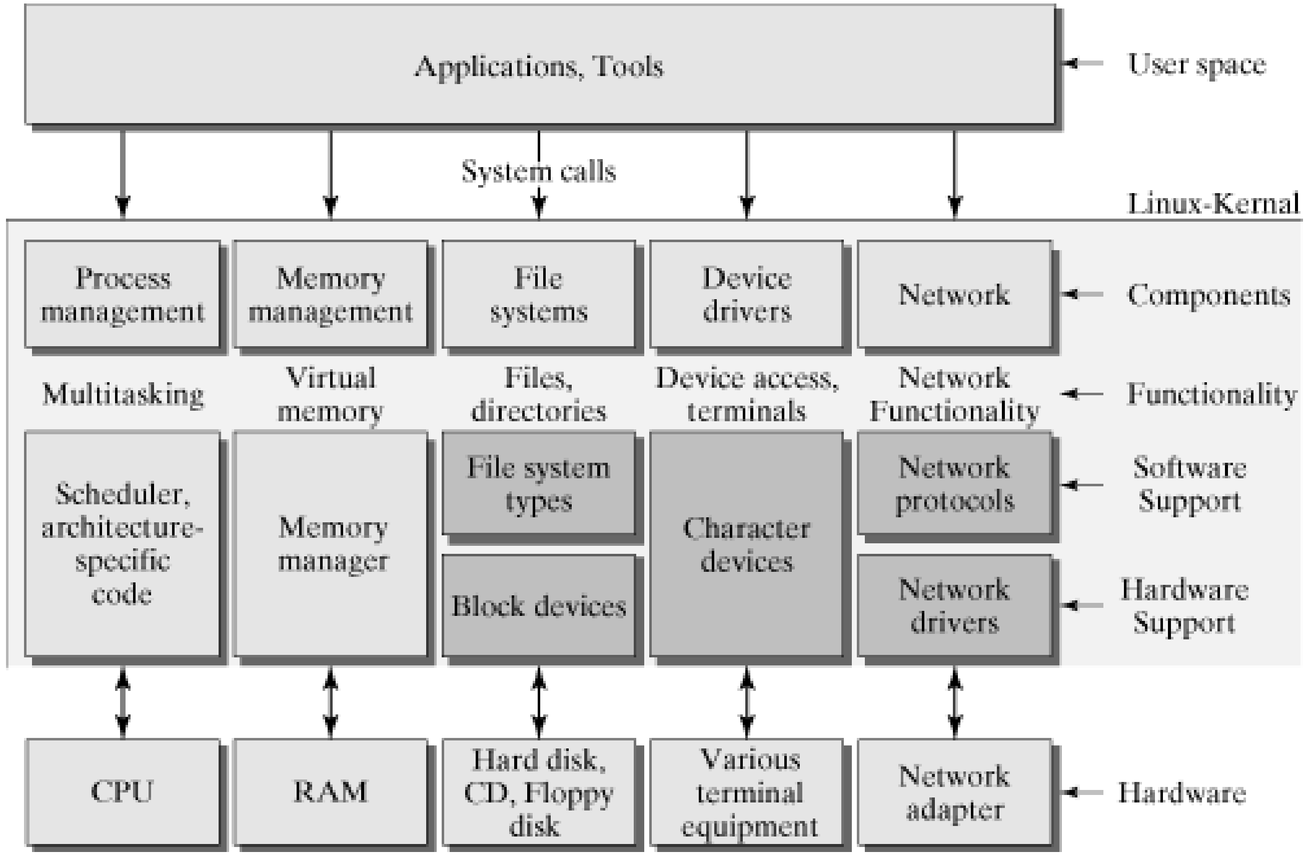
Design Principles

- Linux is a multiuser, multitasking system
- Linux is UNIX compatible
  - its file system adheres to traditional UNIX semantics
  - it fully implements the standard UNIX networking model
  - its API adheres to the SVR4 UNIX semantics
  - it is POSIX-compliant
- Linux supports a wide variety of architectures
- Main design goals are speed, efficiency, and standardization

Components of a Linux System



Linux Architecture



What is Linux Kernel?

- AKA: executive, system monitor.
- Controls and mediates access to hardware.
- Implements and supports fundamental abstractions:
  - Processes, files, devices etc.
- Schedules / allocates system resources:
  - Memory, CPU, disk, descriptors, etc.
- Enforces security and protection.
- Responds to user requests for service (system calls).
- Etc…etc…

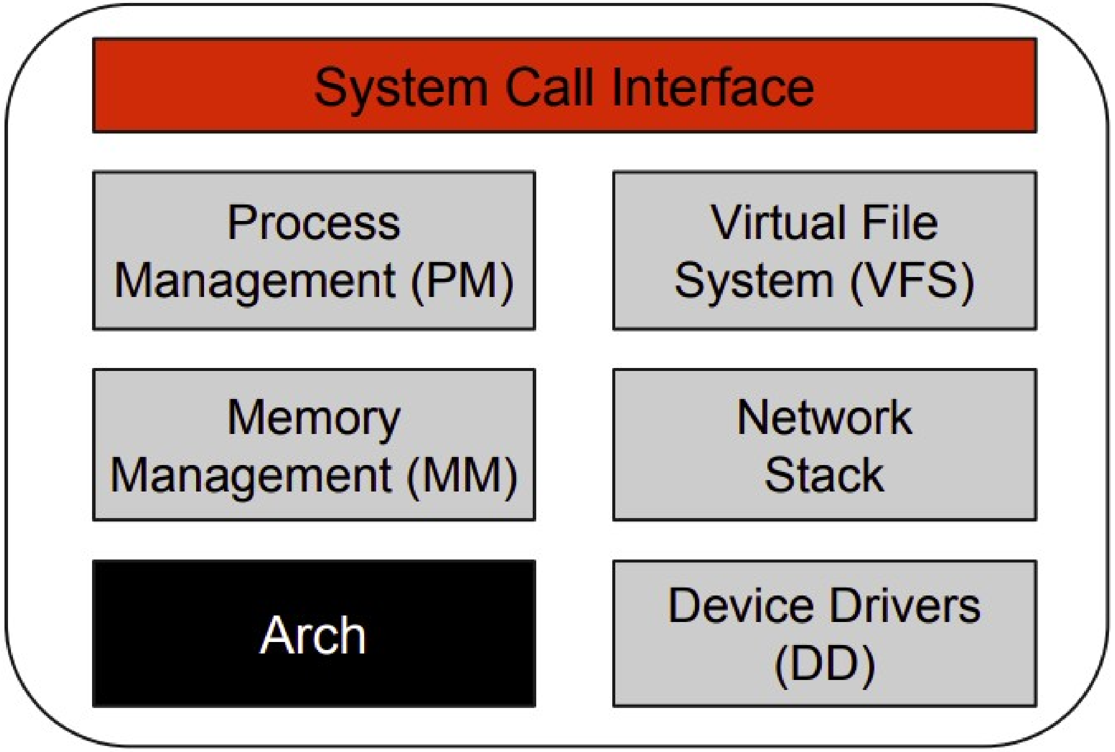
Kernel Design Goals

- Performance: efficiency, speed.
  - Utilize resources to capacity with low overhead.
- Stability: robustness, resilience.
  - Uptime, graceful degradation.
- Capability: features, flexibility, compatibility.
- Security, protection.
  - Protect users from each other & system from bad users.
- Portability.
- Extensibility.

Kernel Modules

- Kernel code that can be compiled, loaded, and unloaded independently
  - to implement device drivers, file systems, or networking protocols
  - it allows a Linux system to be set up with standard minimal kernel
    - other components loaded as modules
- Three components to Linux module support:
  - module management
    - load/unload the module
    - resolve symbols (similar to a linker)
  - driver registration
    - kernel define an interface, module implement the interface
    - module registers to the kernel, kernel maintain a list of loaded modules
  - conflict resolution
    - resource conflicts
- Tools to support kernel modules: lsmod, rmmod, modprobe

Kernel – Main Components

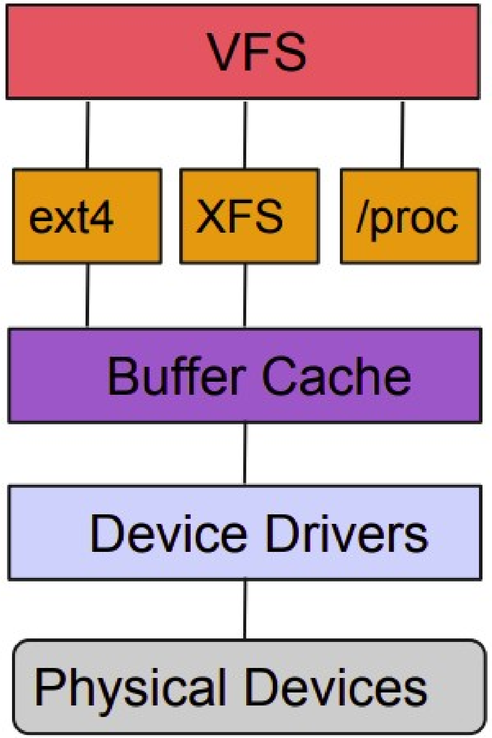


Kernel – Main Components

- 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

Kernel – Main Components

- 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



Kernel Components

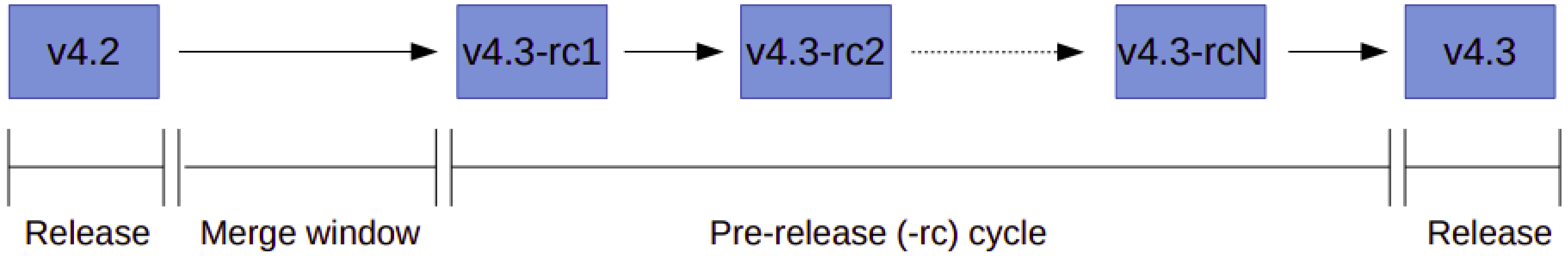
- Network Stack
  - Implement the network protocols
  - Deliver packets across programs and network interfaces
- Device Drivers (DD)
  - Interact with the hardware
  - Extract an abstraction of the device functionalities
- Arch
  - Architecture dependent code

Kernel Contribution Stats

- Nowadays: changing at an unprecedented speed
- 2017 Linux Kernel Development Report by The Linux Foundation:
  - 24,7M lines of code
  - 15600 developers from >1500 companies have contributed since 2005
  - 4319 developers from nearly 519 companies in the past 15 months
  - 8.5 patches merged per hour
  - nearly 15 files and 7500 lines of code added every day
  - a new kernel every 9-10 weeks

How to Contribute to Linux Kernel

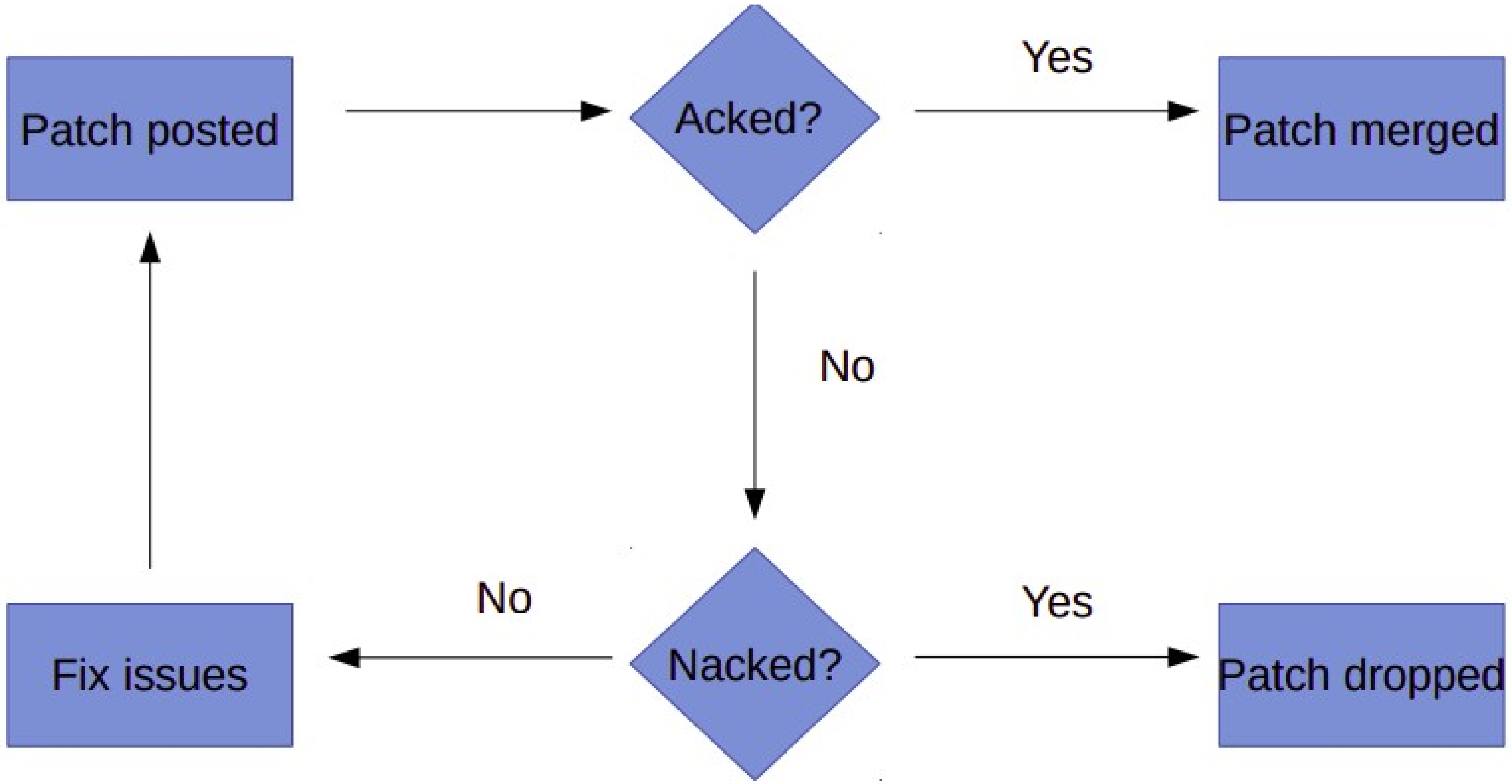
- Linux kernel release cycle (for eg. Kernel v4.2)



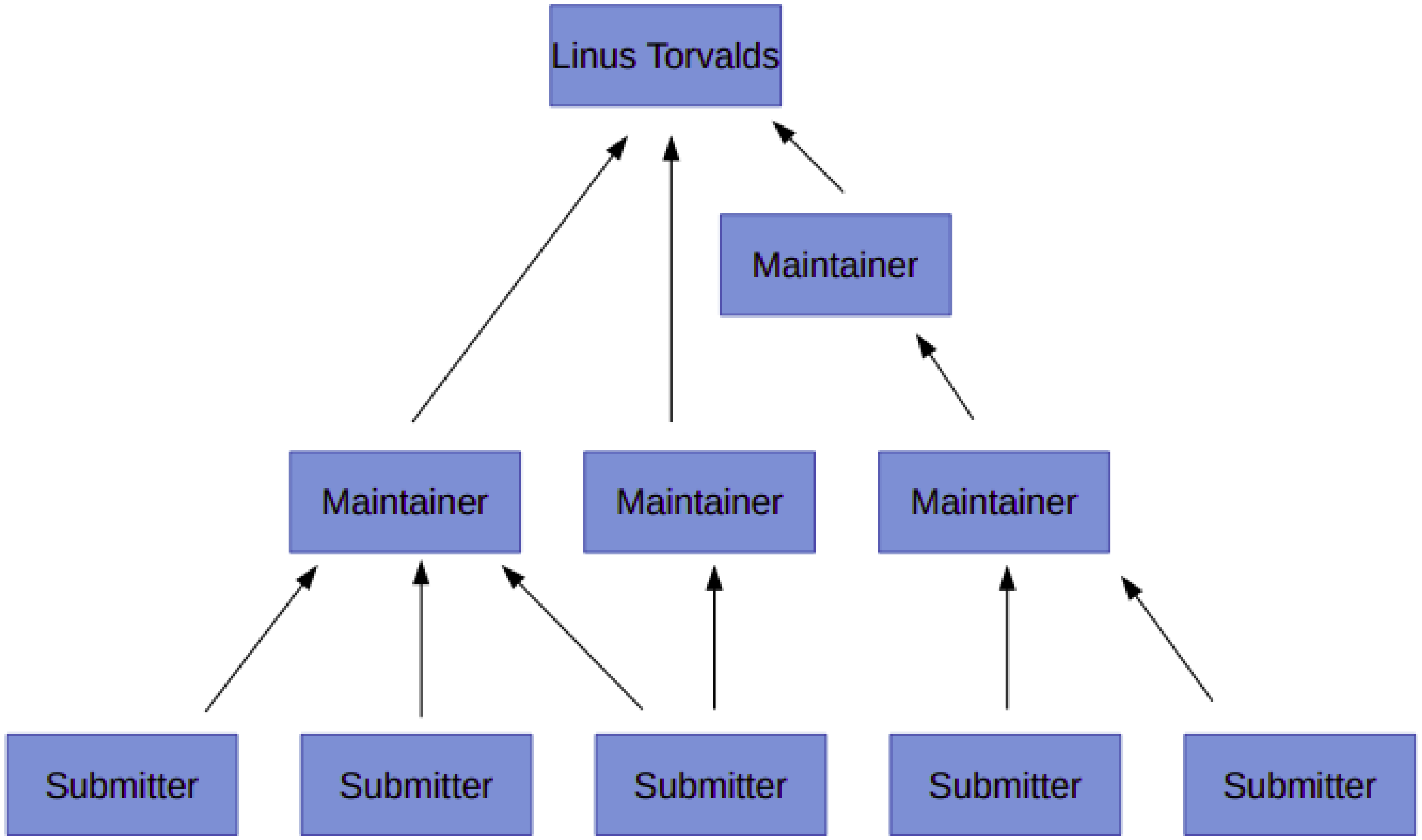
How to Contribute to Linux Kernel

- Linux Kernel Trees
  - linux.git: Linus Torvalds' tree
    - git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
  - linux-stable.git: contains previous versions on which fixes are backported
    - git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git
  - subsystem trees: each maintainer has a tree used for development
  - linux-next.git: integrates all the subsystem maintainer trees for testing
    - git://git.kernel.org/pub/scm/linux/kernel/git/next/linux-next.git

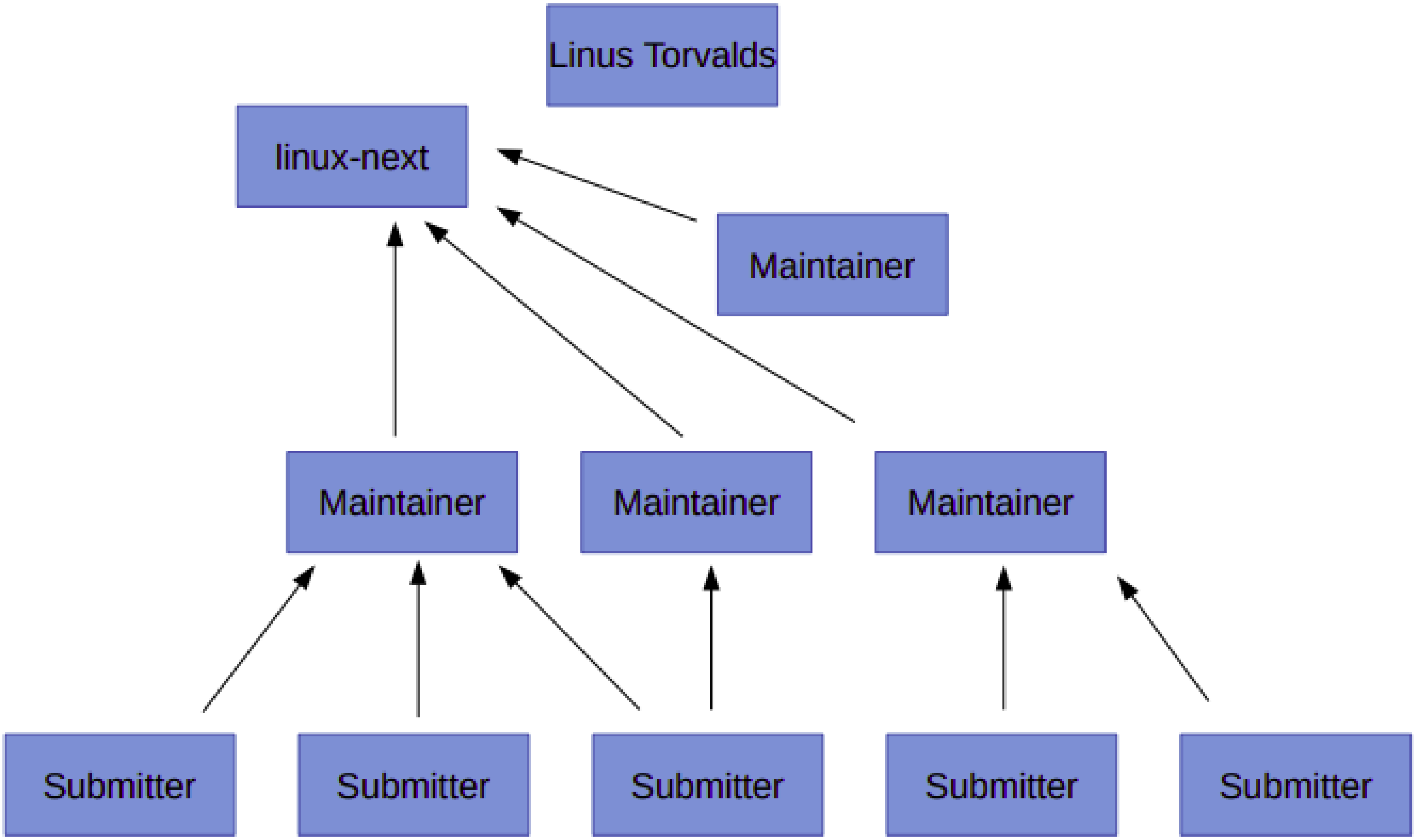
Patch Flow for Mainline



Patch Flow for Mainline

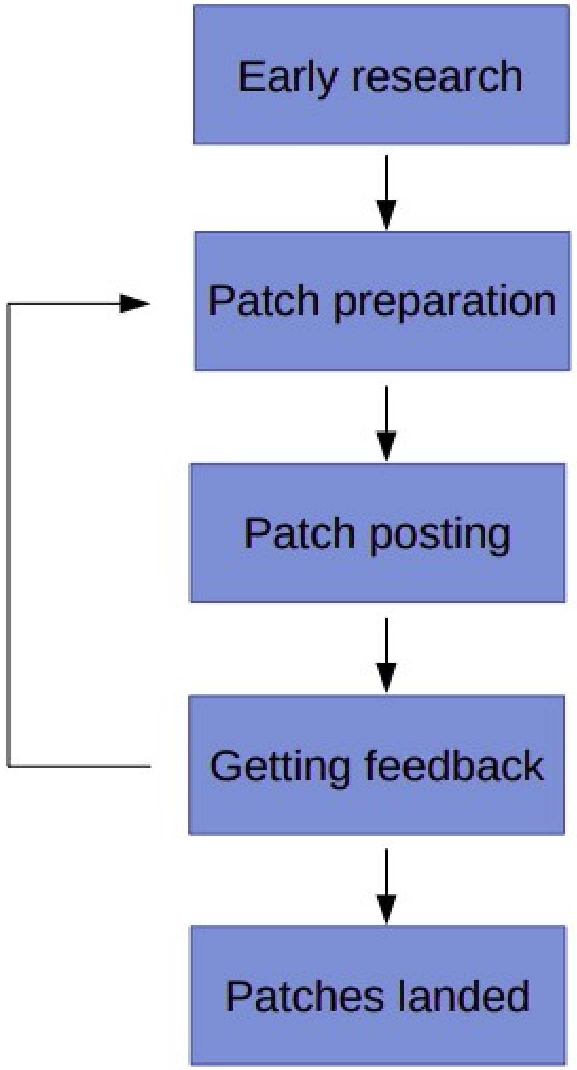


Patch Flow for Mainline



Contribution Steps

- Early Research
- Patch Preparation
- Patch Posting
- Getting Feedback
- Patches Landed



Simply: Download, Configure, Build

- The official versions of the Linux kernel are available at http://www.kernel.org
  - Work-in-progress (mainline), stable and long-term kernels
  - May not contain latest development for specific subsystems (not ready for inclusion yet)
- Kernel sub-communities maintain their own kernel repos (like architecture-specific, drivers, kernel infrastructure etc)
  - Only development trees available

Simply: Download, configure, build

- The kernel sources from http://kernel.org are available as full tarballs (complete kernel sources) and patches (differences between two kernel versions).
- However, usually people use the git version control system. Must have for kernel development!
- Fetch the entire kernel sources and history
  - git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
- Create a branch that starts at a specific kernel version
  - git checkout -b v4.16
- Web interface available at
  - http://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/

Simply: Download, configure, build

- The kernel configuration and build system is based on multiple Makefiles
- Configure kernel using make
  - cd linux-4.16/
  - make menuconfig
  - make defconfig
- Build kernel
  - make
- Produces
  - vmlinux, the raw uncompressed kernel image in the ELF format
  - arch/<arch>/boot/*Image, the final, usually compressed, kernel image
  - all kernel modules, spread over the kernel source tree as .ko files.

Linux File System

- File System is responsible for storing information on disk and retrieving and updating this information.
  - Examples: FAT16, FAT32, NTFS, ext2, ext3 etc.

- In Linux everything is a file.

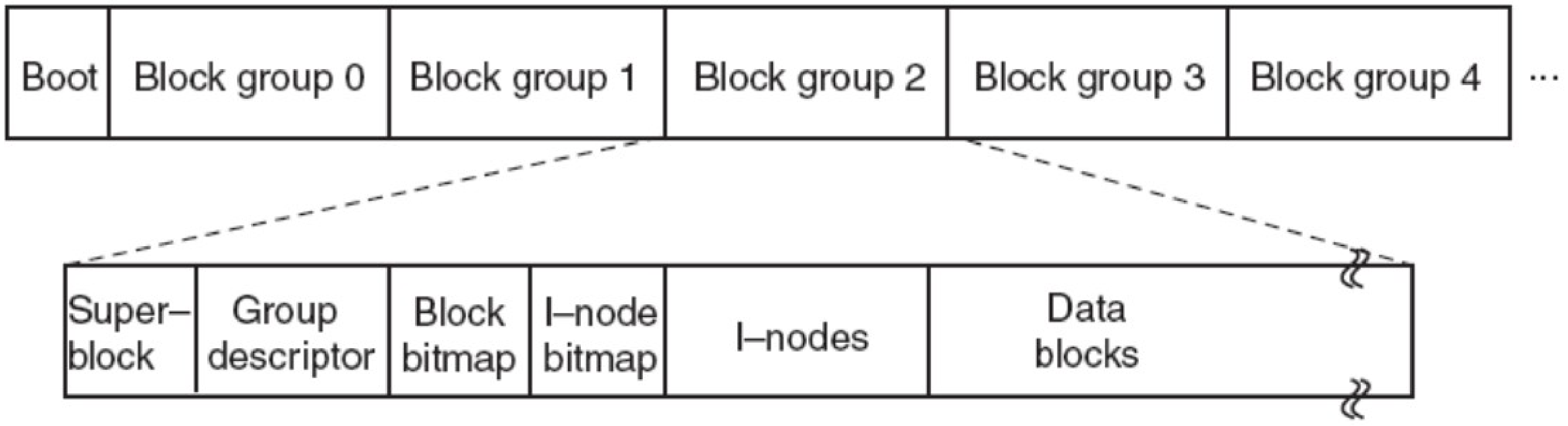
- Standard file system for Linux is Ext2 (Second Extended File System)

Ext 2

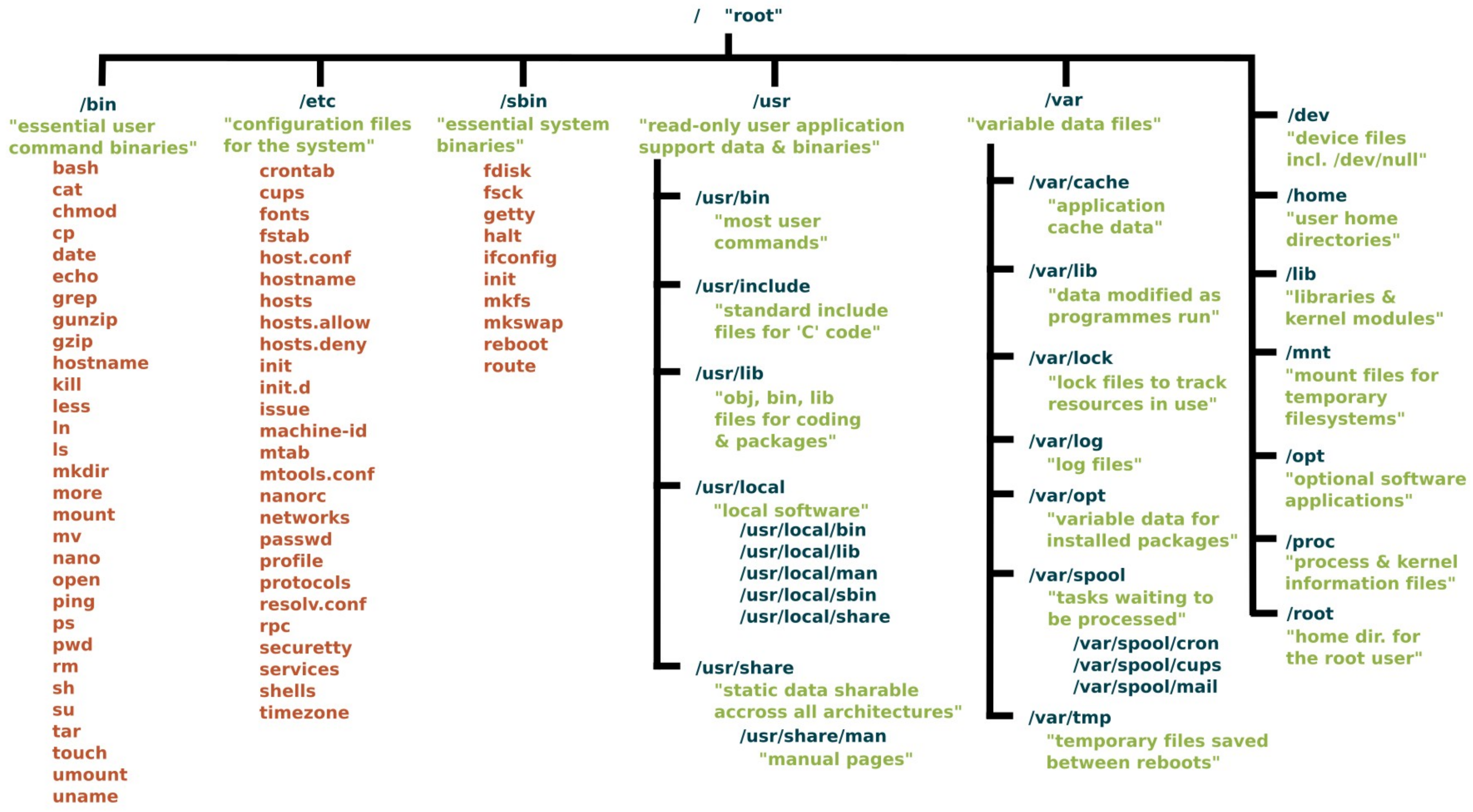
- “Standard” Linux File System
  - Was the most commonly used before ext3 came out
- Uses FFS-like layout
  - Each FS is composed of identical block groups
  - Allocation is designed to improve locality
- inodes contain pointers (32 bits) to blocks
  - Direct, Indirect, Double Indirect, Triple Indirect
  - Maximum file size: 4.1TB (4K Blocks)
  - Maximum file system size: 16TB (4K Blocks)

Ext2 Disk Layout

- Files in the same directory are stored in the same block group
- Files in different directories are spread among the block groups



Linux File Structure



Some Important Directories

- The / (root) directory is the most important directory in Linux file system structure. The / is the parent of Linux file system structure.
- The /home directory keeps all Linux user account's home directory. Make sure you provide big enough hard disk for /home directory. It's a good idea to set quota for each user account in /home directory.
- The /etc directory keeps all servers and application system's configuration files. This directory perhaps the most visited directory if you are working in Linux command line terminal.

Directories, Files and Inodes

- Every directory and file is listed in its parent directory.
- In the case of the root directory, that parent is itself.
- A directory is a file that contains a table listing the files contained within it, giving file names to the inode numbers in the list.
- The information about all the files and directories is maintained in INODE TABLE
- An Inode (Index Nodes) is an entry in the table containing information about a file (metadata) including file permissions, UID, GID, size, time stamp, pointers to files data blocks on the disk etc.

Users, Groups & Access Permissions

- In UNIX/LINUX, there is a concept of user and an associated group
- The system determines whether or not a user or group can access a file or program based on the permissions assigned to them.
- Apart from all the users, there is a special user called Super User or the root which has permission to access any file and directory

Access Permissions

- There are three permissions for any file, directory or application program.
- The following lists the symbols used to denote each, along with a brief description:
  - r — Indicates that a given category of user can read a file.
  - w — Indicates that a given category of user can write to a file.
  - x — Indicates that a given category of user can execute the file.

Access Permissions

- Each of the three permissions are assigned to three defined categories of users.
- The categories are:
  - owner — The owner of the file or application.
  - group — The group that owns the file or application.
  - others — All users with access to the system

Access Permissions

- One can easily view the permissions for a file by invoking a long format listing using the command ls -l.
- For instance, if the user <striker> creates an executable file named test, the output of the command ls -l test would look like this:
- **drwxrwxrwx 2 striker striker 4096 Feb 10 2020 aodv**

Access Permissions

- The permissions for this file are listed are listed at the start of the line, starting with rwx.
- This first set of symbols define owner access.
- The next set of rwx symbols define group access
- The last set of symbols defining access permitted for all other users.

Access Permissions

- This listing indicates that the file is readable, writable, and executable by the user who owns the file (user striker) as well as the group owning the file (which is a group named striker).

- The file is also world-readable, world-executable, and world-writable.

What’s Next?

- Important Commands and Editors
- Shell, AWK, SED
- Some Linux Utilities