Introduction

- A brief history of Unix
  - AT&T
  - BSD
  - Linux (and why Linux is not Unix)
- Mach
  - Started work in mid 80s at Carnegie Mellon University
  - Supports the Unix programming interface
  - Based on a microkernel approach
  - Provides a small set of essential services and a framework for other OS functions at user level
  - Resulted in commercial systems such as OSF/1 and NextStep (Mach 2.5)
  - Mach 3.0 was the first microkernel implementation
- Standards
  - Deal with interface between programmer and OS and not how the interface is implemented
  - Defines a set of functions and their semantics
  - Functions can be implemented in the kernel or in user-level libraries
  - System V Interface Definition (SVID)
    - AT&T
    - System V programming interface
    - System confirmation to SVID can be verified by System V Verification Suite (SVVS)
  - Portable Operating System Based on Unix (POSIX)
    - IEEE, commissioned 1986, published in 1990
    - Inspired by SVR3 and 4.3BSD
  - X/Open Portability Guide (XPG)
    - X/Open Consortium
    - Based on a draft of POSIX.1, enhances it by adding internationalization, window interfaces, and data management

Recent Improvements/Changes

- Hardware changes
  - Parallelization
    - Traditional Unix kernel is designed for a uniprocessor architecture and cannot protect global data structures from concurrent access by multiple processors
    - Some vendors have added locks to protect these structures, and called the phenomenon *parallelization*
  - Memory vs I/O
    - Earlier Unixen were limited in performance by memory size and processor speed, and hence, kernel used swapping/paging to accommodate a larger number of processes in small memory
    - As memory became cheaper and CPU speed increased, systems became I/O bound, in the sense of spending more time swapping
    - Recent research has concentrated on reducing the disk bottleneck, leading to *Redundant Arrays of Inexpensive Disks* (RAID) and log-structured file systems

- Paradigm shifts
  - 1970s: Centralized computer with *dumb* terminals
  - 1980s: Rise of the workstation
    - High speed bitmapped display
    - Multiple windows
    - Interactive use, one user per workstation even though it could support multiple users simultaneously
    - High speed networks for communications between workstations
  - Client-server computing
    - One or more powerful machines as server
    - Individual workstations as clients
    - Specialized servers as file servers, print servers, mail servers, web servers, and database servers
  - Distributed computing
    - Number of machines collaborate to provide network-based services
    - Avoids network congestion and single point of failure
    - Dot.coms, AOL
- Small is beautiful
  - Unix philosophy was to provide a number of tools such that each tool will do only one job but do it efficiently
  - Complex tasks could be performed by combining a number of these tools
    - Easy to see this by looking at filters and pipelines in Unix
  - The philosophy did not permeate to kernel; kernel is monolithic and not easily extensible
  - A new kernel based on microkernel architecture was created as Mach
    - A small kernel provides the framework for process management
    - User-level server tasks provide other functions
  - Microkernels are not well matched with traditional monolithic kernels in terms of performance optimization
    - Microkernels have the extra overhead of message passing
    - Linus Torvalds has criticized the microkernel architecture in his essay, and argued in favor of monolithic architecture for Linux kernel[1][2]
- Flexibility
  - Earlier versions of Unix kernels were not flexible in terms of file system, process scheduling, and executable file formats
  - Newer systems have had to support local and remote file systems, new executable formats, such as the older a.out (assembler output) and newer elf (executable and linking format)
  - Real-time and multimedia applications require scheduler support for different classes of applications

Pros and cons of Unix

- Pros
  - Truly multitasking and multiuser
  - Robust

[1] Chris DiBona, Sam Ockman, and Mark Stone. *Open Sources: Voices from the Open Source Revolution*, O'Reilly, January 1999.

[2] The article by Torvalds is available online at `http://www.oreilly.com/catalog/opensources/book/linus.html`

-
  -
    - It is rare to see system crashes caused by a user program
    - Memory and file security
  - Excellent programming environment
    - Designed by the hackers, for the hackers
    - Based on “small is beautiful” approach and the KISS principle
      - Files as streams of bytes instead of ISAM (Indexed Sequential Access Method) and HSAM (Hierarchical Sequential Access Method)
  - Simple interface to I/O devices (devices as files)
  - Portability
- Cons
  - Designed by the hackers, for the hackers
  - Cryptic commands
  - Too much material to learn
  - Too many different versions