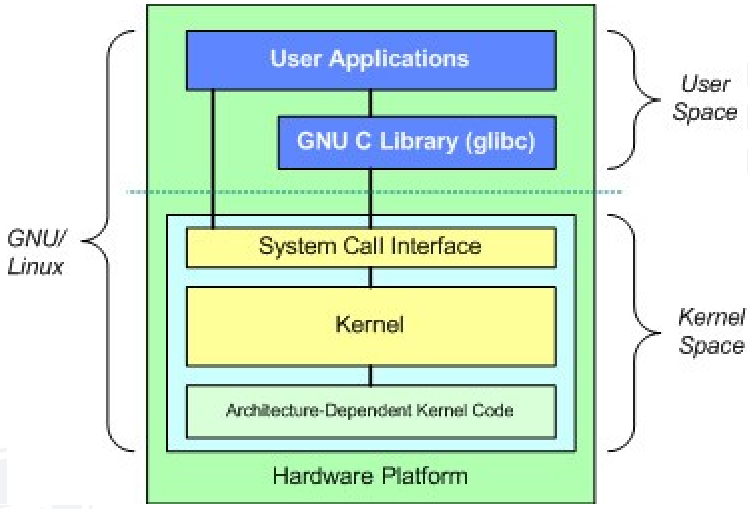
The LINUX Operating System

A Very Brief Introduction to it's Architecture

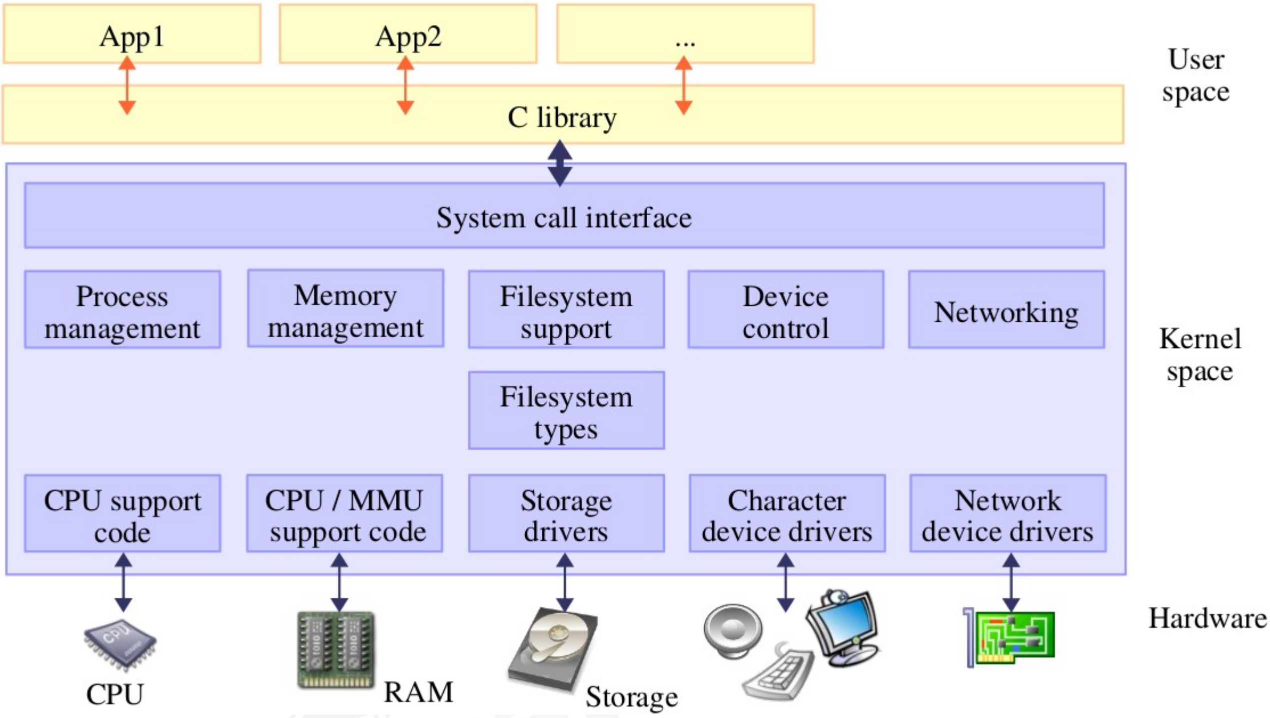
Important Notice

Linux / Unix Architecture

Simplified



More Detailed



<< Above Pic: © Copyright 2006 2004, Michael Opdenacker , © Copyright 2004 2008 Codefidence Ltd. >>

Discussion

- The SCI – System Call Interface – Layer
- CPU Privilege Levels
  - User Mode
  - Kernel (Supervisor) Mode
- Flow of a Process – Birth to Death – between privilege levels

Source

Architecture

Various layers within Linux, also showing separation between the userland and kernel space

User mode	**User applications**	e.g. bash, LibreOffice, Blender, 0 A.D.				
	Low-level system components:	**System daemons**: *systemd*, *logind*, *networkd*, *soundd*, ...	**Windowing system**: ***display server***	**Other libraries:** *GLib*, *GTK+*, *Qt*, *EFL*, *SDL*, *SFML*,*FLTK*, *GNUstep*, etc.		**Graphics**: *Mesa 3D*,*AMD Catalyst*, ...
	C standard library	`open`, `exec`, `sbrk`, `socket`, `fopen`, `calloc`, ... (up to 2000 subroutines) *glibc* aims to be POSIX/SUS-compatible, *uClibc* targets embedded systems, *bionic* written forAndroid, etc.				
		`stat`, `splice`, `dup`, `read`, `open`, `ioctl`, `write`, `mmap`, `close`, `exit`, etc. (about 380 system calls) The Linux kernel System Call Interface (SCI, aims to be POSIX/SUS-compatible)				
Kernel mode	**Linux kernel**	Process scheduling subsystem	IPC subsystem	Memory management subsystem	Virtual files subsystem	Network subsystem
		Articles: ALSA, DRI, evdev, LVM, device mapper, Linux Network Scheduler, Netfilter Linux Security Modules: *SELinux*, *TOMOYO*, *AppArmor*, *Smack*				
Hardware (CPU, main memory, data storage devices, etc.)						

Linux is a monolithic kernel. Device drivers and kernel extensions run in kernel space (ring 0 in many CPU architectures), with full access to the hardware, although some exceptions run in user space, for example filesystems based on FUSE. The graphics system most people use with Linux doesn't run in the kernel, in contrast to that found in Microsoft Windows. Unlike standard monolithic kernels, device drivers are easily configured as modules, and loaded or unloaded while running the system. Also unlike standard monolithic kernels, device drivers can be pre-empted under certain conditions. This latter feature was added to handle hardware interrupts correctly, and to improve support for symmetric multiprocessing.[*citation needed*] By choice, the Linux kernel has no Binary Kernel Interface.[45]

The hardware is also incorporated into the file hierarchy. Device drivers interface to user applications via an entry in the `/dev`[46] and/or `/sys` directories. Process information as well is mapped to the file system through the `/proc` directory.[46]

Linux supports true preemptive multitasking (both in user mode and kernel mode), virtual memory, shared libraries, demand loading, shared copy-on-write executables (via KSM), memory management, the Internet protocol suite, and threading.

...

Some "Golden Rules" *<< To be read later >>*
• Modern OS's are Virtual Memory (VM / MMU) – based. Thus: ◦ All addresses referred to in user-space are user-mode virtual addresses ◦ All addresses referred to in kernel-space are kernel-mode virtual addresses ◦ (Very) few exception cases do occur: typically, user-mode device drivers may specify a physical address, kernel-mode memory management is aware of physical addresses and their mapping, some kernel-mode drivers specify a physical (or bus) address. ◦ A process can only refer to (lookup) it's own legally-mapped virtual address space: any attempt to look "outside" the process – more correctly – any attempt to lookup an unmapped region result in it being killed by the OS. ◦ *Exceptions* to the above: ▪ memory-mapping physical memory via */dev/mem* (need to be root; also see kernel directive CONFIG_STRICT_DEVMEM) ▪ using the process_vm_[read|write]v(2) system calls (kernel ver 3.2 onward [requires kernel support via CONFIG_CROSS_MEMORY_ATTACH], glibc 2.15 onward); (need to be root or local and remote processes should have common credentials (IOW, owner)).

Source

Monolithic Kernel

A monolithic kernel is an operating system architecture where the entire operating system is working in kernel space and is alone in supervisor mode. The monolithic model differs from other operating system architectures (such as the microkernel architecture)[1][2] in that it alone defines a high-level virtual interface over computer hardware. A set of primitives or system calls implement all operating system services such as process management, concurrency, and memory management. Device drivers can be added to the kernel as modules.

Monolithic architecture examples

Unix kernels
 BSD
 FreeBSD
 NetBSD
 OpenBSD
 Solaris 1 / SunOS 1.x-4.x

UNIX System V
 AIX
 HP-UX

Unix-like kernels
 Linux

DOS
 DR-DOS
 MS-DOS
 Microsoft Windows 9x series (95, 98, Windows 98SE, Me)

OpenVMS
XTS-400

Microkernel

In computer science, a **microkernel** (also known as µ-kernel) is the near-minimum amount of software that can provide the mechanisms needed to implement an operating system (OS). These mechanisms include low-level address space management, thread management, and inter-process communication (IPC). If the hardware provides multiple rings or CPU modes, the microkernel is the only software executing at the most privileged level (generally referred to as supervisor or kernel mode).

[*citation needed*] Traditional operating system functions, such as device drivers, protocol stacks and file systems, are removed from the microkernel to run in user space.[*citation needed*] In source code size, microkernels tend to be under 10,000 lines of code, as a general rule. MINIX's kernel, for

example has fewer than 6,000 lines of code.[1]
Example : Minix, QNX, VxWorks.

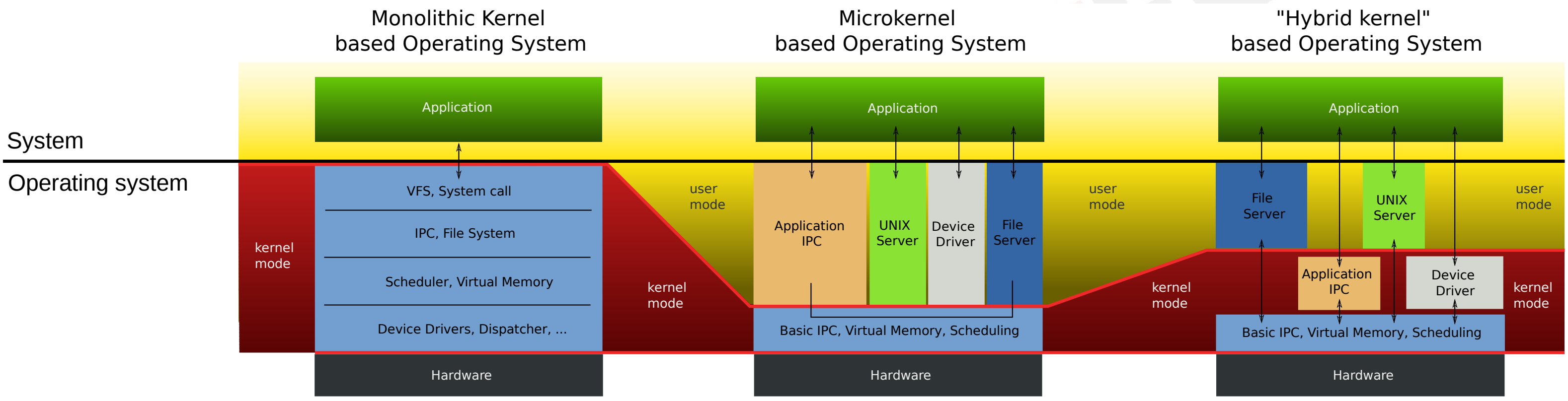
Hybrid

A **hybrid kernel** is a kernel architecture based on combining aspects of microkernel and monolithic kernel architectures used in computer operating systems. The traditional kernel categories are monolithic kernelsand microkernels (with nanokernels and exokernels seen as more extreme versions of microkernels). The category is controversial due to the similarity to monolithic kernel; the term has been dismissed by Linus Torvalds as simple marketing.[1]

The idea behind this category is to have a kernel structure similar to a microkernel, but implemented in terms of a monolithic kernel. In contrast to a microkernel, all (or nearly all) operating system services are inkernel space. While there is no performance overhead for message passing and context switching between kernel and user mode, as in monolithic kernels, there are no reliability benefits of having services in user space, as in microkernels.

Implementations

BeOS kernel
Haiku kernel
Syllable
BSD-based
DragonFly BSD (first non-Mach BSD OS to use a hybrid kernel)
XNU kernel (core of Darwin, used in Mac OS X and iOS)
NetWare kernel[7]
Inferno kernel
NT kernel (used in Windows NT 3.1, Windows NT 3.5, Windows NT 4.0, Windows 2000, Windows Server 2003, Windows XP, Windows Vista, Windows Server 2008, Windows 7, Windows Server 2008 R2,Windows 8, and Windows Server 2012)
ReactOS kernel



Ref: ***Why is Linux called a monolithic kernel?***

[P.T.O. -->]

[OPTIONAL / FYI]

An FAQ regarding keeping track of Linux kernel Changes

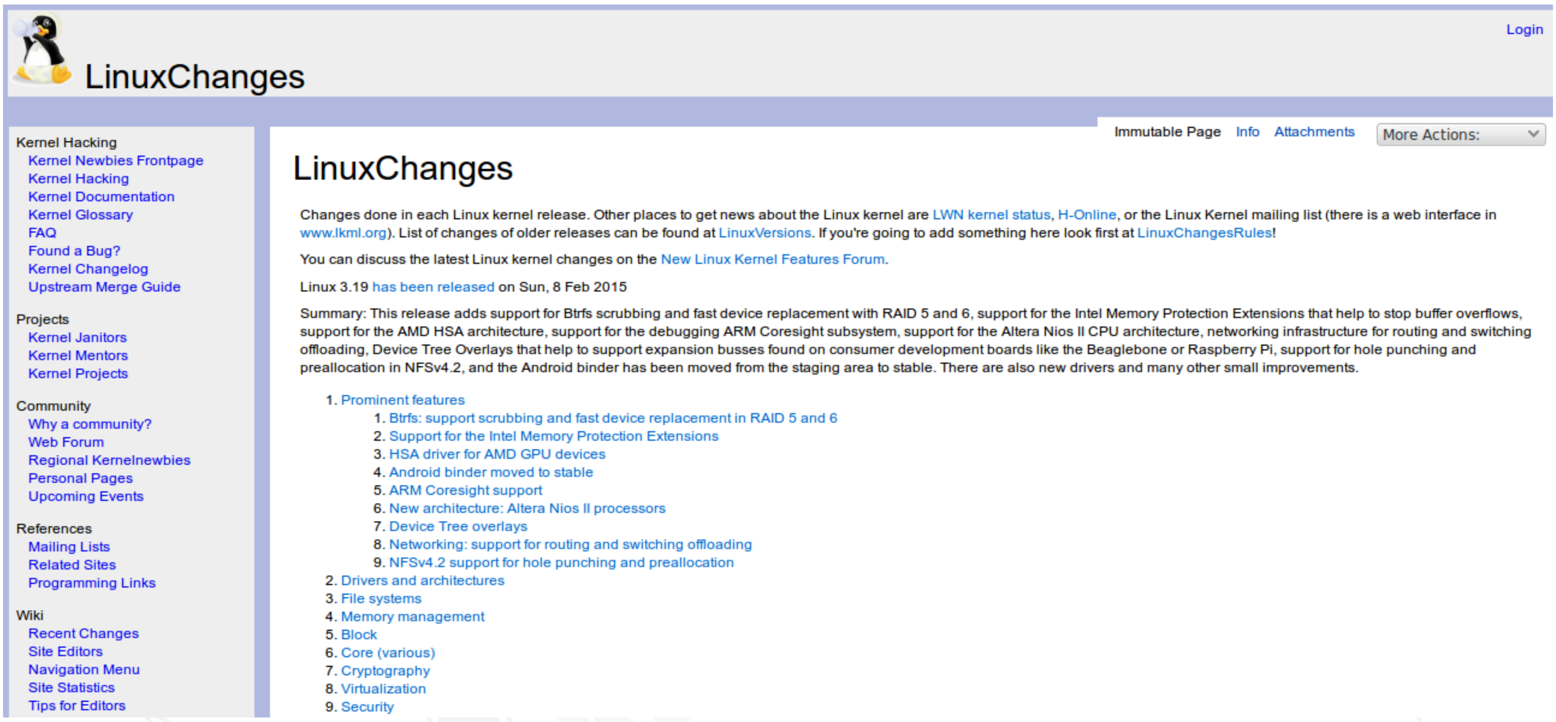
The Linux kernel is a very fast moving target: things change, quite rapidly at times, new enhancements and features get merged, kernel internal APIs / ABIs get deprecated, etc etc. How can one sanely keep track of all these changes?

The answer: follow the LKML (Linux Kernel Mailing List).
But "sanely"?
:-)

Read the kernelnewbies "Linux Changes" website !

1. The page
http://kernelnewbies.org/LinuxChanges

will have the *latest mainline kernel* changes information:
<< 3.19 at the time of this insertion (March 2015) >>



LinuxChanges

Login

Kernel Hacking
- Kernel Newbies Frontpage
- Kernel Hacking
- Kernel Documentation
- Kernel Glossary
- FAQ
- Found a Bug?
- Kernel Changelog
- Upstream Merge Guide

Projects
- Kernel Janitors
- Kernel Mentors
- Kernel Projects

Community
- Why a community?
- Web Forum
- Regional Kernelnewbies
- Personal Pages
- Upcoming Events

References
- Mailing Lists
- Related Sites
- Programming Links

Wiki
- Recent Changes
- Site Editors
- Navigation Menu
- Site Statistics
- Tips for Editors

Immutable Page Info Attachments More Actions:

LinuxChanges

Changes done in each Linux kernel release. Other places to get news about the Linux kernel are LWN kernel status, H-Online, or the Linux Kernel mailing list (there is a web interface in www.lkml.org). List of changes of older releases can be found at LinuxVersions. If you're going to add something here look first at LinuxChangesRules!

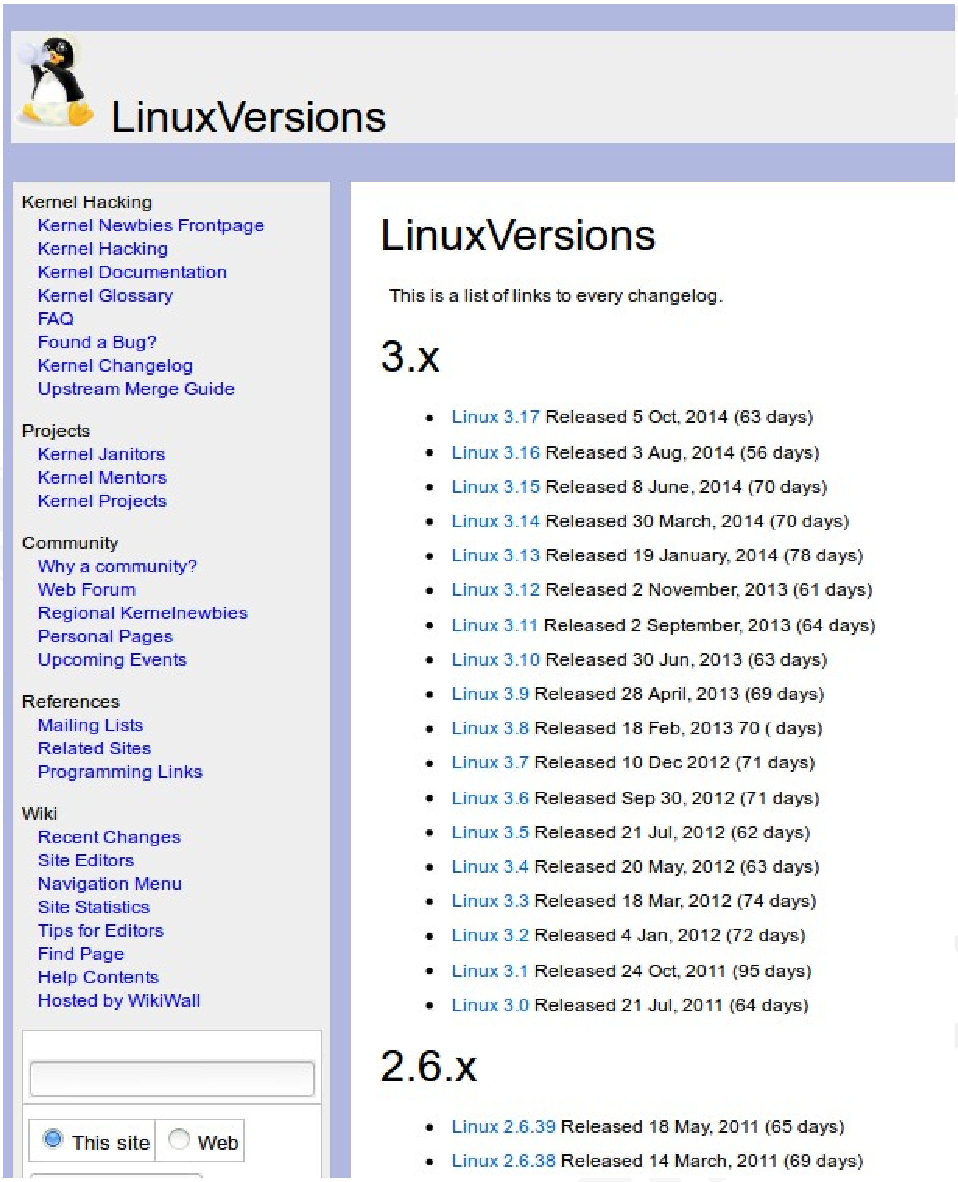
You can discuss the latest Linux kernel changes on the New Linux Kernel Features Forum.

Linux 3.19 has been released on Sun, 8 Feb 2015

Summary: This release adds support for Btrfs scrubbing and fast device replacement with RAID 5 and 6, support for the Intel Memory Protection Extensions that help to stop buffer overflows, support for the AMD HSA architecture, support for the debugging ARM Coresight subsystem, support for the Altera Nios II CPU architecture, networking infrastructure for routing and switching offloading, Device Tree Overlays that help to support expansion busses found on consumer development boards like the Beaglebone or Raspberry Pi, support for hole punching and preallocation in NFSv4.2, and the Android binder has been moved from the staging area to stable. There are also new drivers and many other small improvements.

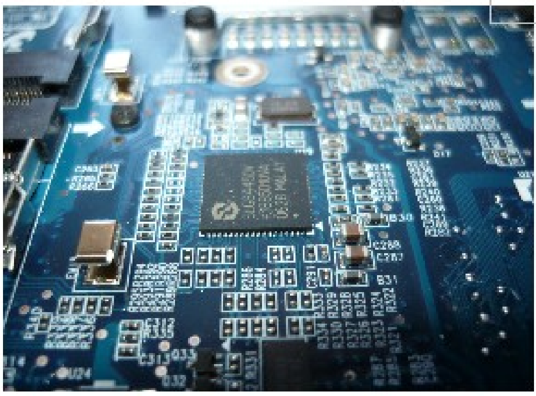
1. Prominent features
   1. Btrfs: support scrubbing and fast device replacement in RAID 5 and 6
   2. Support for the Intel Memory Protection Extensions
   3. HSA driver for AMD GPU devices
   4. Android binder moved to stable
   5. ARM Coresight support
   6. New architecture: Altera Nios II processors
   7. Device Tree overlays
   8. Networking: support for routing and switching offloading
   9. NFSv4.2 support for hole punching and preallocation
2. Drivers and architectures
3. File systems
4. Memory management
5. Block
6. Core (various)
7. Cryptography
8. Virtualization
9. Security

2. To see links to all kernel versions, goto
http://kernelnewbies.org/LinuxVersions



Linux Operating System Specialized

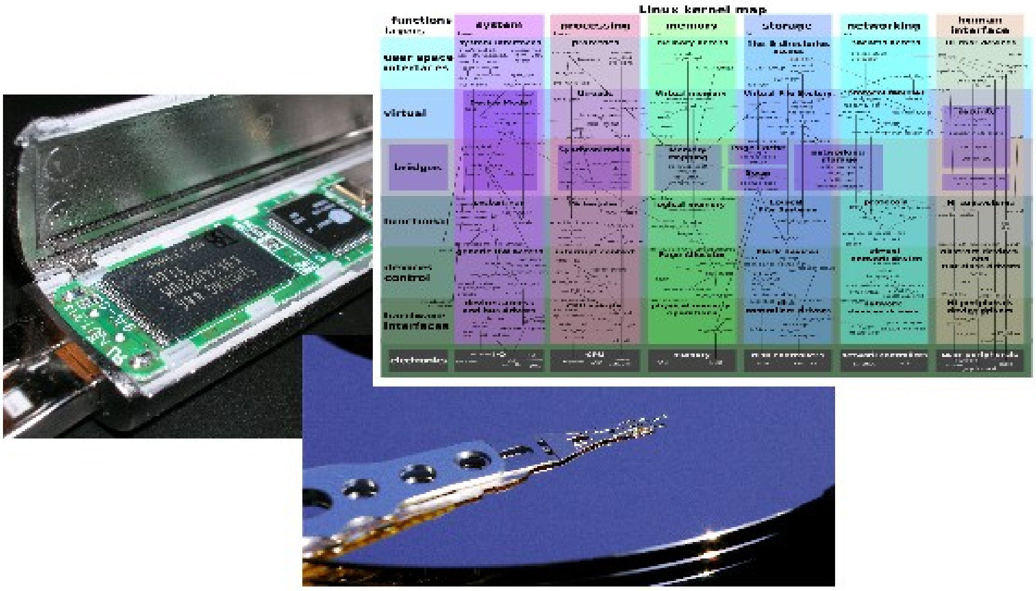
kaiwanTECH



The highest quality Training on:

Linux Fundamentals, CLI and Scripting
Linux Systems Programming
Linux Kernel Internals
Linux Device Drivers
Embedded Linux
Linux Debugging Techniques
New! *Linux OS for Technical Managers*

Please do visit our website for details:
http://kaiwantech.in



http://kaiwantech.in