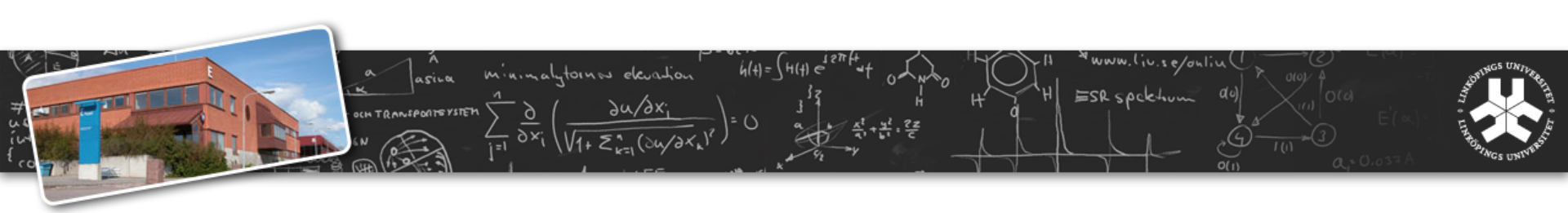


TDDI41 - Systeminstallation

Anders Fröberg

anders.froberg@liu.se

LiU expanding reality

Upplägg

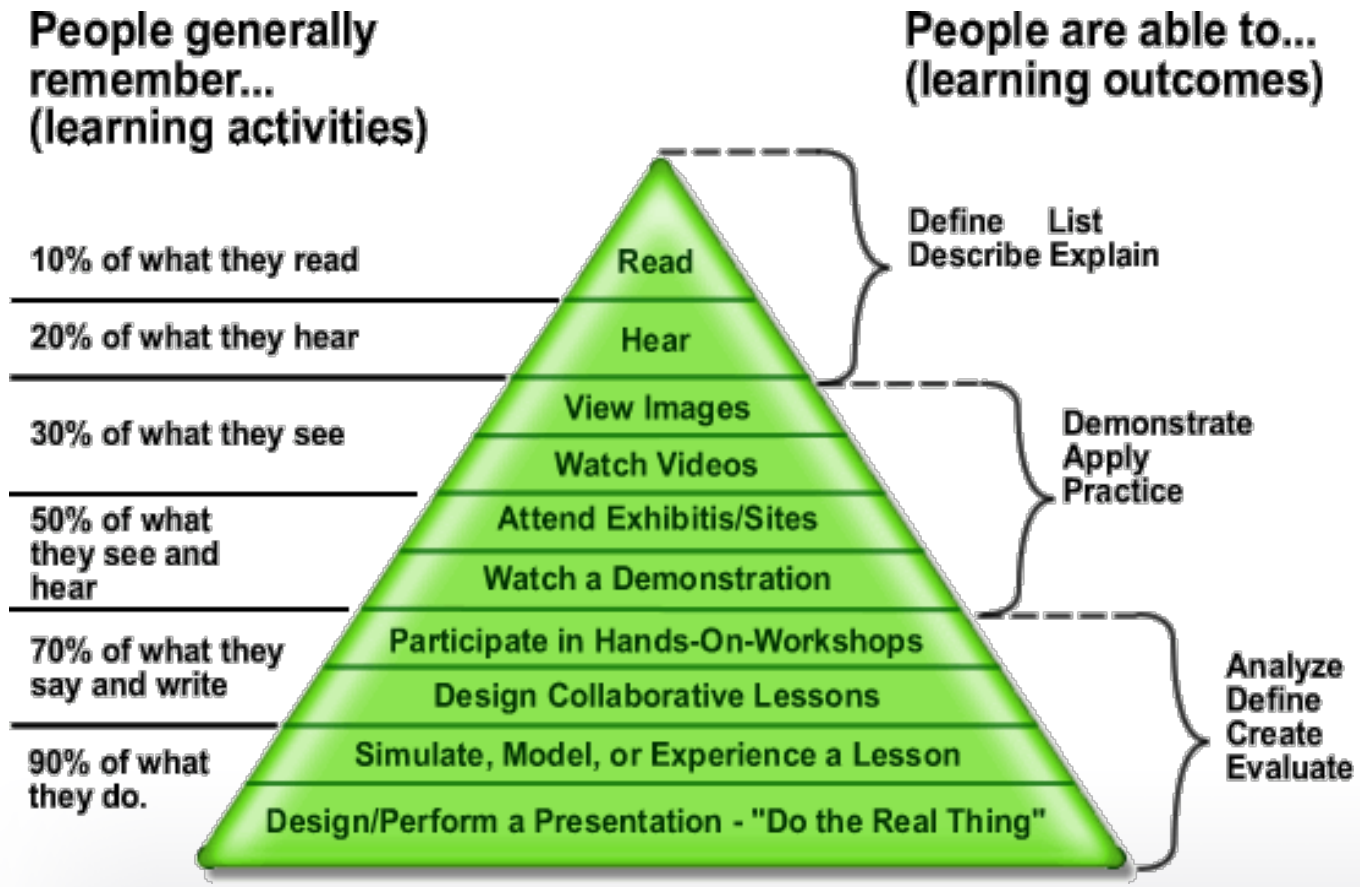
- Vem är jag ?
- Kursen
  - Innehåll
  - Upplägg
- Intrduktion till Linux

Kursinformation

- Kurshemsidan: http://www.ida.liu.se/~TDDI41/
  - Deadlines, labbeskrivningar, labanmälan, errata

- Laborationer
  - Studera – **planera** – implementera – **testa** – dokumentera
  - Krav på ”ingenjörsmässigt” arbetssätt
  - Strikta deadlines

Kursens kunskapsfilosofi



Labmiljö

- Eget (virtuellt) labnät
- Alla på en server
- Virtualiserade system
  - User-mode Linux
  - Debian/Gnu Linux

http://www.ida.liu.se/~TDDI41

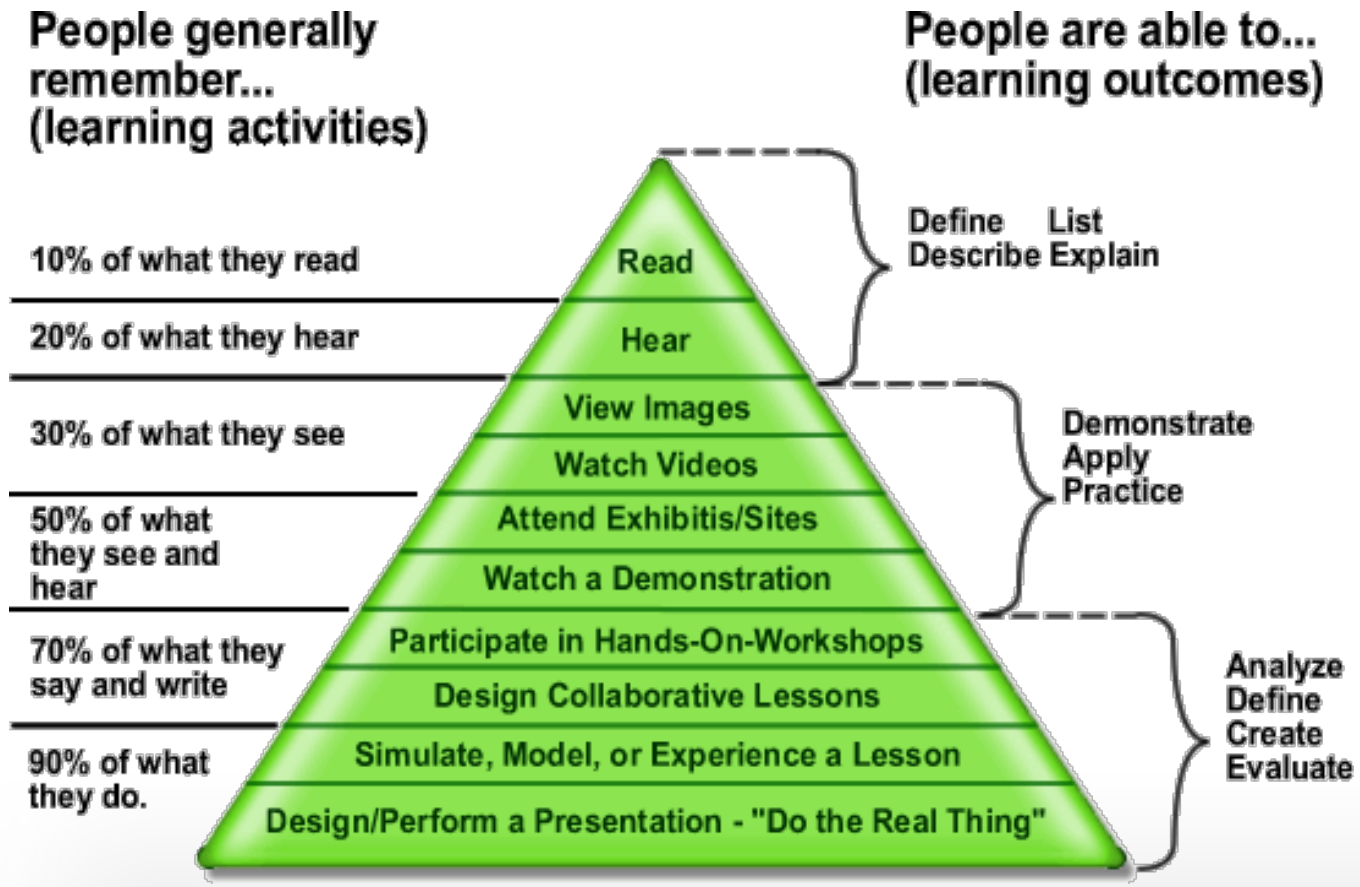
MATURITY

has nothing to do with age. Maturity comes from experiences, mistakes, learning, and understanding.

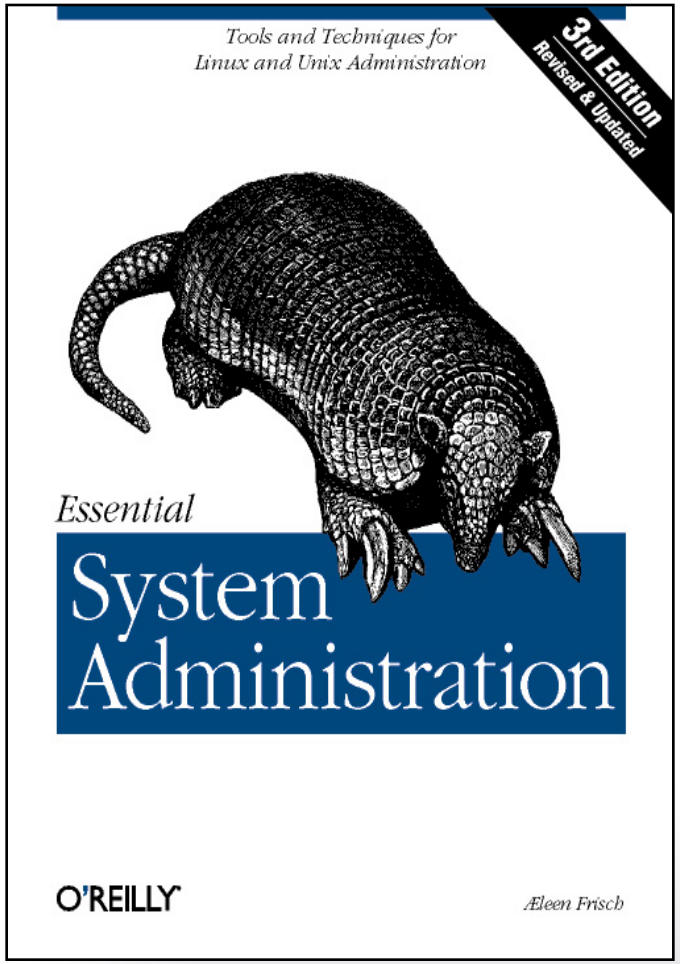
WWW.LIVELIFEHAPPY.COM


T
E
A
M

Kursens kunskapsfilosofi



Rekommenderade Böcker



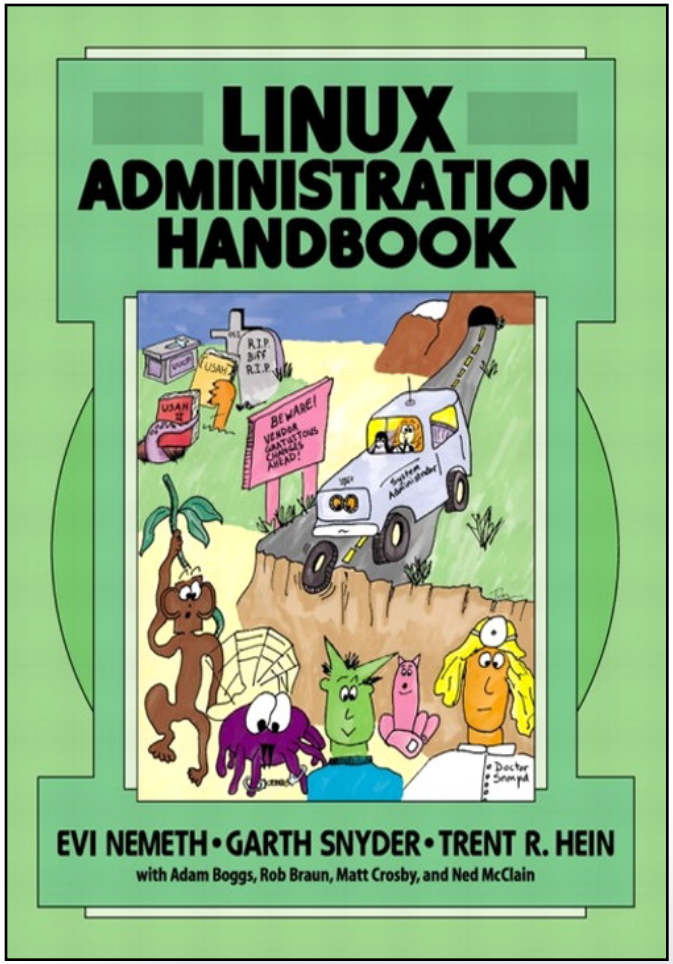
Essential System Administration, 3rd Ed.

Æleen Frisch

Grundlig genomgång av administration av Unix-system. Tar upp alla viktiga delsystem på ett bra sätt. Exempel från Linux, Tru64, BSD, HP-UX, AIX, Solaris och andra system. Fullständigt fokus på tekniken. En hård bok.

Æleen Frisch

Rekommenderade Böcker



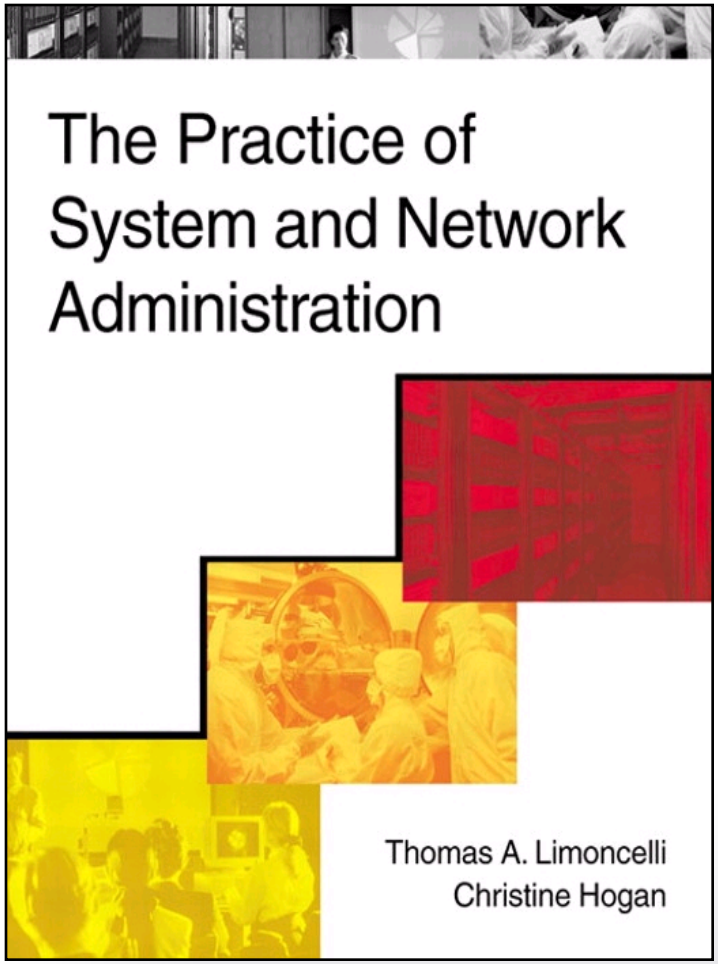
Linux Administration Handbook

Evi Nemeth, et al

Administration av Linux-system. Täcker alla viktiga delsystem både ur Linux-perspektiv och mer generellt perspektiv. Likvärdig Essential System Administration men med ett något annorlunda urval av ämnen. En ganska hård bok.

Evi Nemeth

Rekommenderade Böcker



The Practice of System and Network Administration

Thomas A. Limoncelli

Christine Hogan

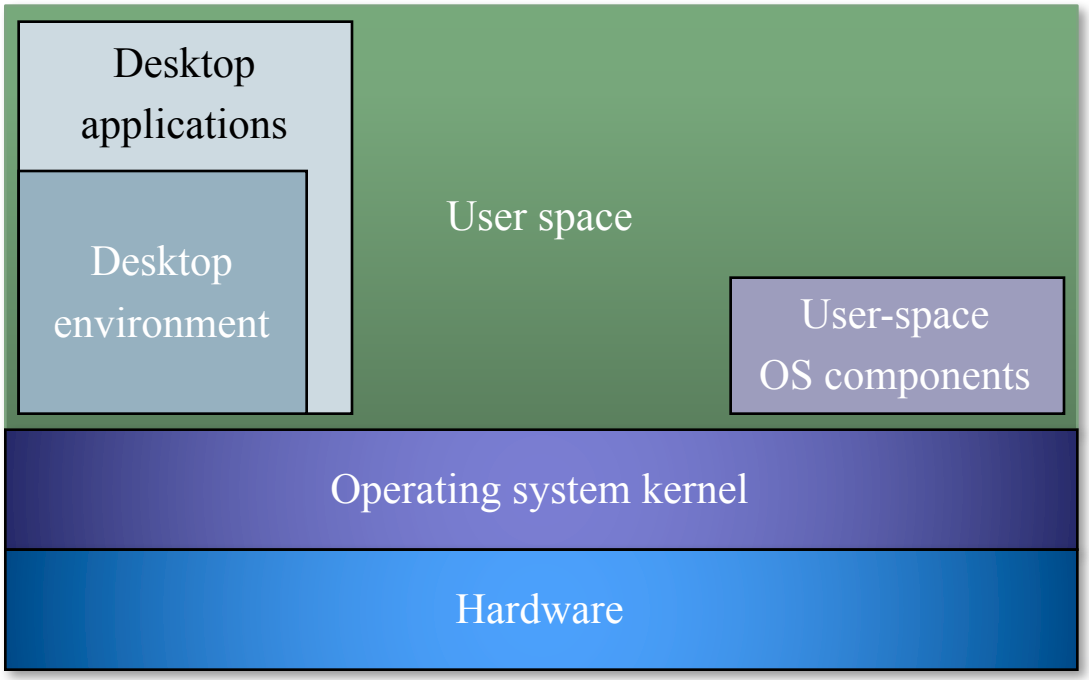
Fokus på systemadministratörens yrke, inte på tekniken. Tar upp alla aspekter, från kundkontakter och gruppdynamik till hur man gör bra kabeldragning. Massivt innehåll. Mycket lite teknik. En mjuk bok.

Tom Limoncelli

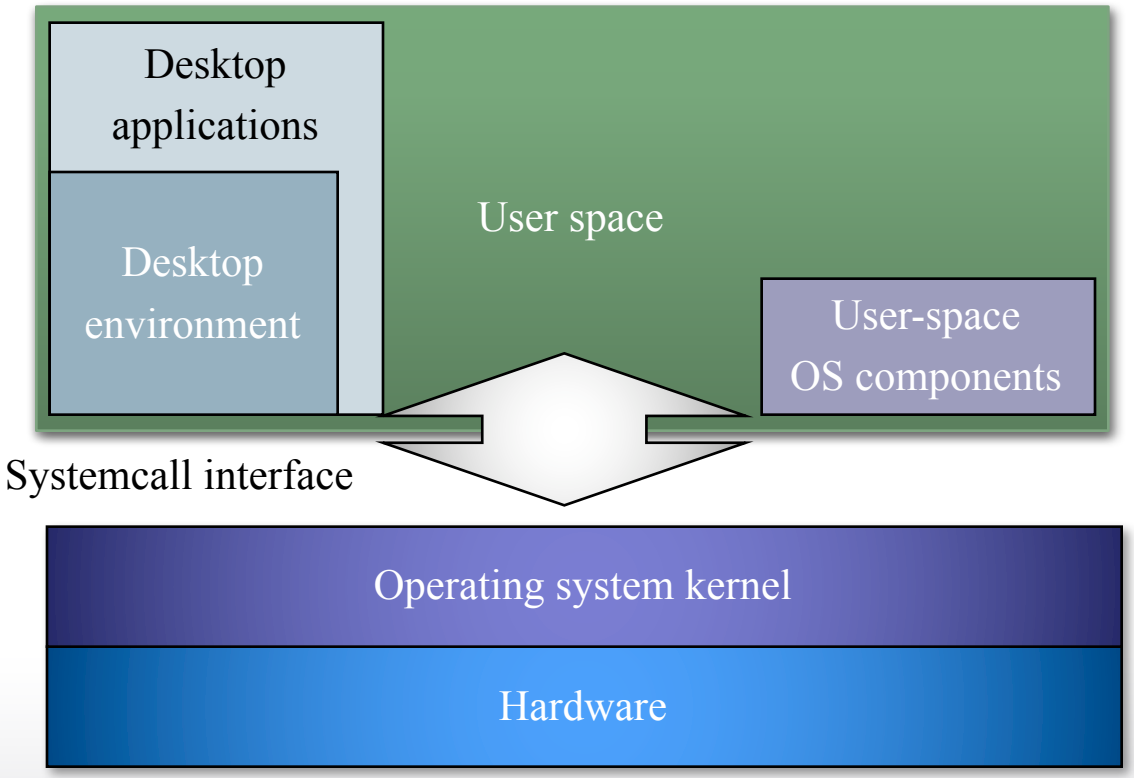
Christine Hogan

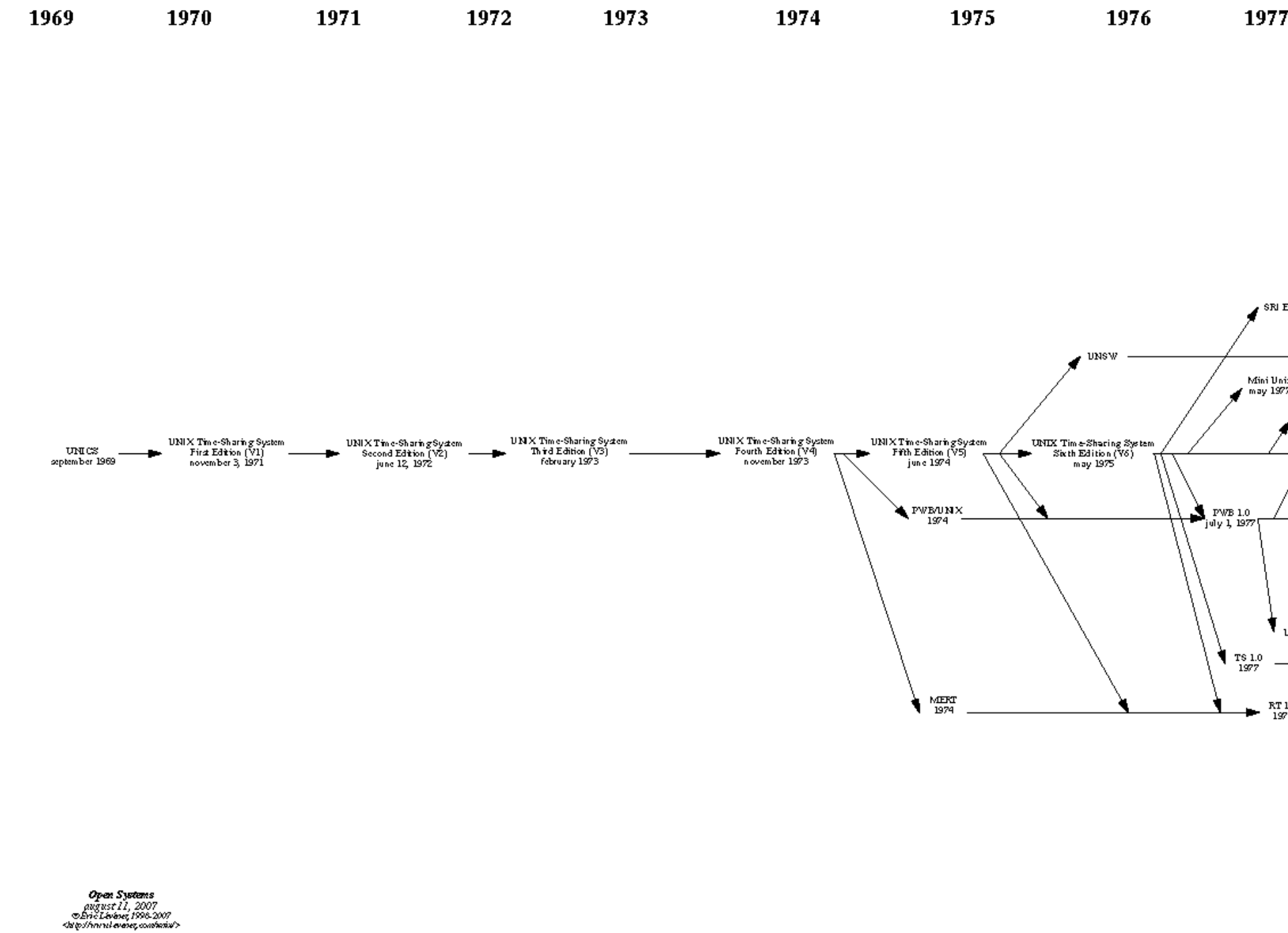
LINUX

What is Linux?



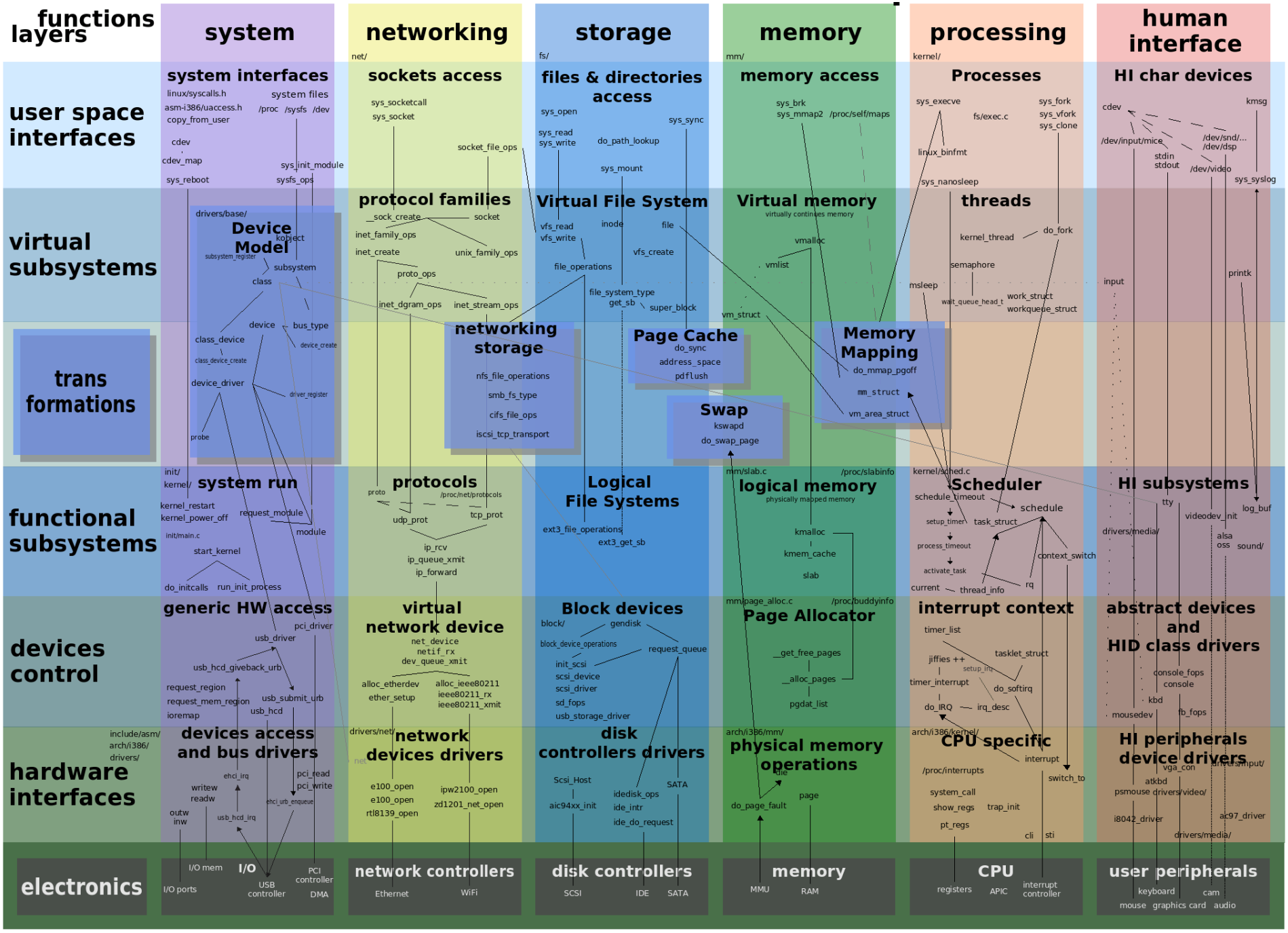
What is Linux?





Open Systems
august 11, 2007

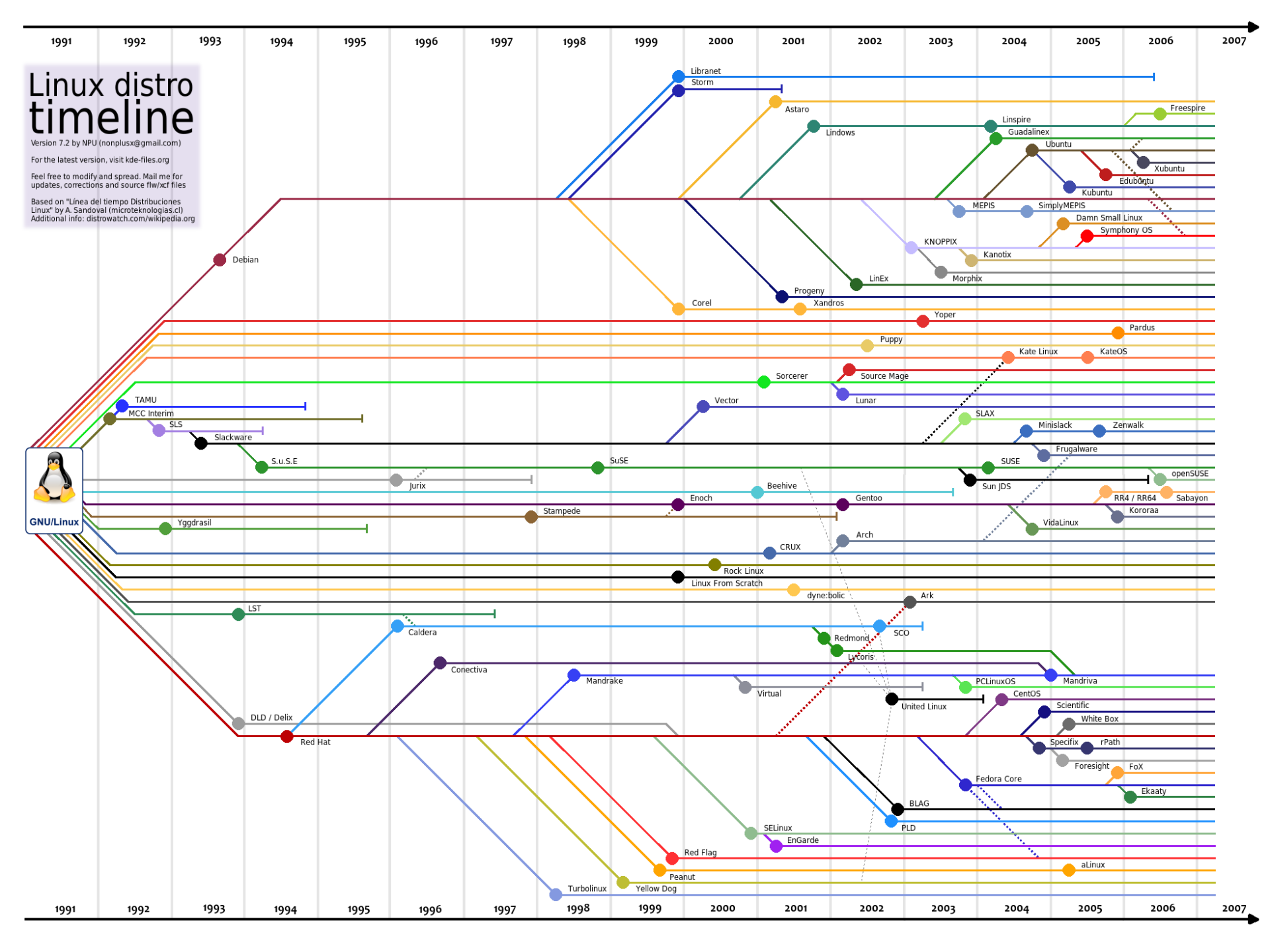
<http://www.levenez.com/unix/>


functions
layers
system
networking
storage
memory
processing
human interface
user space interfaces
virtual subsystems
trans formations
functional subsystems
devices control
hardware interfaces
electronics
system interfaces
sockets access
files & directories access
memory access
Processes
HI char devices
protocol families
Virtual File System
Virtual memory
threads
Device Model
networking storage
Page Cache
Memory Mapping
Swap
system run
protocols
Logical File Systems
logical memory
Scheduler
HI subsystems
generic HW access
virtual network device
Block devices
Page Allocator
interrupt context
abstract devices and HID class drivers
devices access and bus drivers
network devices drivers
disk controllers drivers
physical memory operations
CPU specific
HI peripherals device drivers
network controllers
disk controllers
memory
CPU
user peripherals

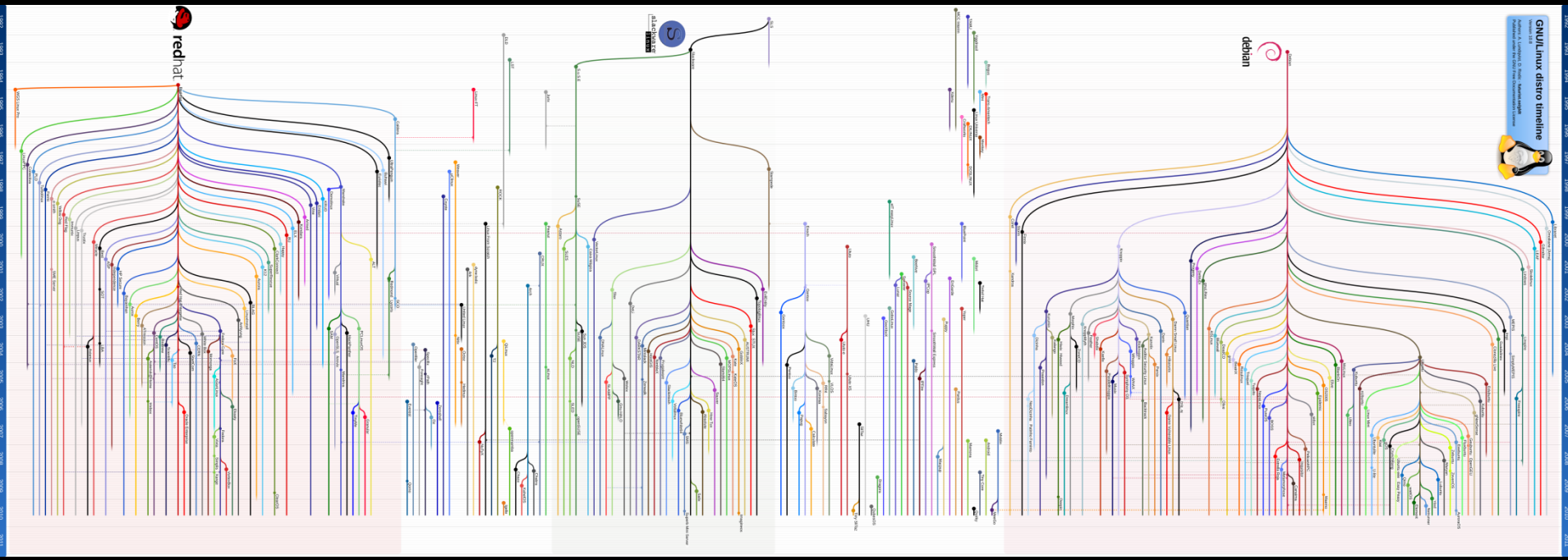
Linux distributions

- Distributions offer a complete operating environment
  - Installation software/procedures
  - Operating system kernel
  - Complete user space

- Different distributions offer different choices
  - Philosophy (e.g. open/closed; stable/cutting-edge; source/binary)
  - Choice of user space software (e.g. desktop environment)
  - Purpose (e.g. desktop/server)
  - Size (e.g. large/small disk; large/small memory)
  - Special-purpose (e.g. PVR, forensics, pen-test)


Linux distro timeline
Version 7.2 by NPU (nonplusx@gmail.com)
For the latest version, visit kde-files.org
Feel free to modify and spread. Mail me for updates, corrections and source flw/xcf files
Based on "Línea del tiempo Distribuciones Linux" by A. Sandoval (microteknologias.cl)
Additional info: distrowatch.com/wikipedia.org
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
GNU/Linux
Debian
TAMU
MCC Interim
SLS
Slackware
S.u.S.E
Jurix
SuSE
Yggdrasil
Stampede
Enoch
Beehive
Gentoo
Sorcerer
Source Mage
Lunar
Vector
Libranet
Storm
Astaro
Lindows
Linspire
Freespire
Guadalinex
Ubuntu
Xubuntu
Edubuntu
Kubuntu
MEPIS
SimplyMEPIS
Damn Small Linux
Symphony OS
KNOPPIX
Kanotix
Morphix
LinEx
Progeny
Xandros
Corel
Yoper
Pardus
Puppy
Kate Linux
KateOS
SLAX
Minislack
Zenwalk
Frugalware
SUSE
Sun JDS
openSUSE
RR4 / RR64
Sabayon
Kororaa
VidaLinux
Arch
CRUX
Rock Linux
Linux From Scratch
dyne:bolic
Ark
LST
Caldera
SCO
Redmond
Lycoris
Conectiva
Mandrake
Virtual
PCLinuxOS
Mandriva
United Linux
CentOS
Scientific
White Box
DLD / Delix
Red Hat
Specifix
rPath
Foresight
FoX
Ekaaty
Fedora Core
BLAG
PLD
SELinux
EnGarde
Red Flag
aLinux
Peanut
Yellow Dog
Turbolinux

1992



2010

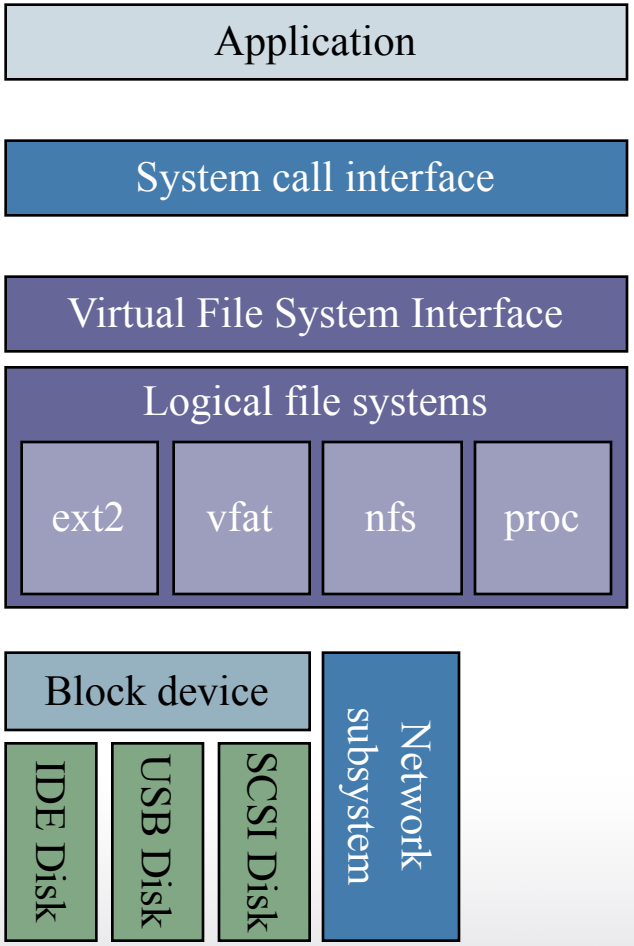
What about Unix?

- What is Unix?
  - Unix was an operating system developed by Bell Labs, then AT&T, then many, many others
  - Unix is a trademark held by the Open Group
  - Unix is an *idea*
- Standard Unix: SUS and POSIX
- Linux *feels* like Unix
  - Same (or similar) user space
  - Based on similar ideas
    - Less now with systemd

A look at Linux: files and file systems

- Applications go through VFS
- VFS passes all requests to appropriate logical file system
- Logical file system gets data from wherever it wants to

- Unified interface to all files
- *Anything* can be a file



What do we mean when we say ”file system”?

- We mean the software that VFS uses to get files and data
  - ”Have you loaded ufs into the kernel?”
- We mean the specification of the data structures that the software uses to store files and data
  - ”The ext2 superblock is replicated for reliability purposes.”
- We mean the *actual* data structures stored on a disk (and the simulated ones provided by non-storage file systems)
  - ”I need to verify the integrity of the file system on /dev/hda1.”

File system organization

- Single directory hierarchy
  - No ”drive letters”
  - Starts at /

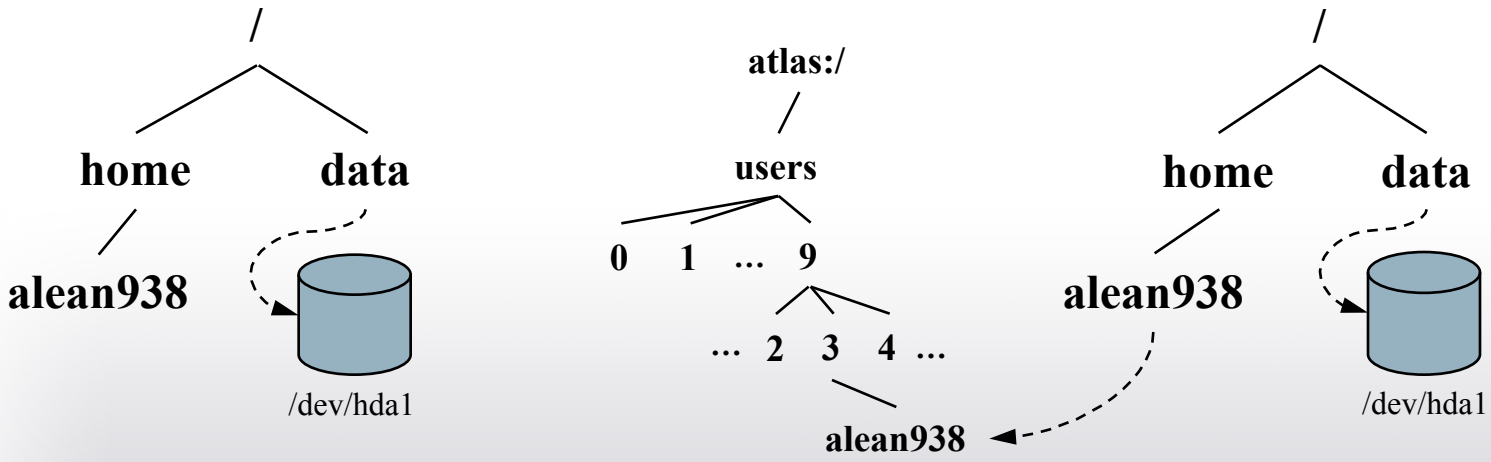
- ”Mounting”
  - Attaches a file system (e.g. ext2 file system on a hard drive, or file system exported using NFS) to a point in the file system hierarchy
  - The ”root file system” is the file system mounted at /, and must be mounted during system startup
  - It is possible to mount the same file system several times in different locations
  - It is possible to specify options (e.g. read only) when mounting
  - Default mounts in /etc/fstab (on Linux)

Example of mounting file systems

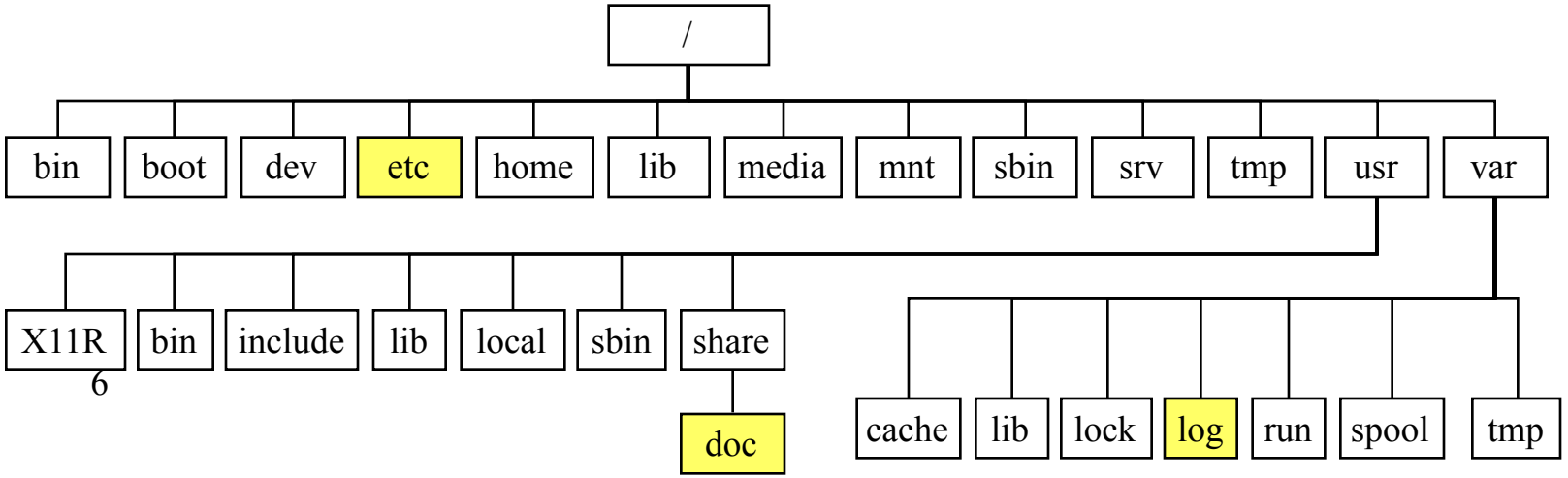
```
mount -t ext2 -o noexec /dev/hda2 /data
mount -t nfs -o rw atlas:/users/9/3/alean938 /home/alean938
```

File system type: ext2
Options: noexec
Source: /dev/hda2 (blockenhet)
Mount point: /data

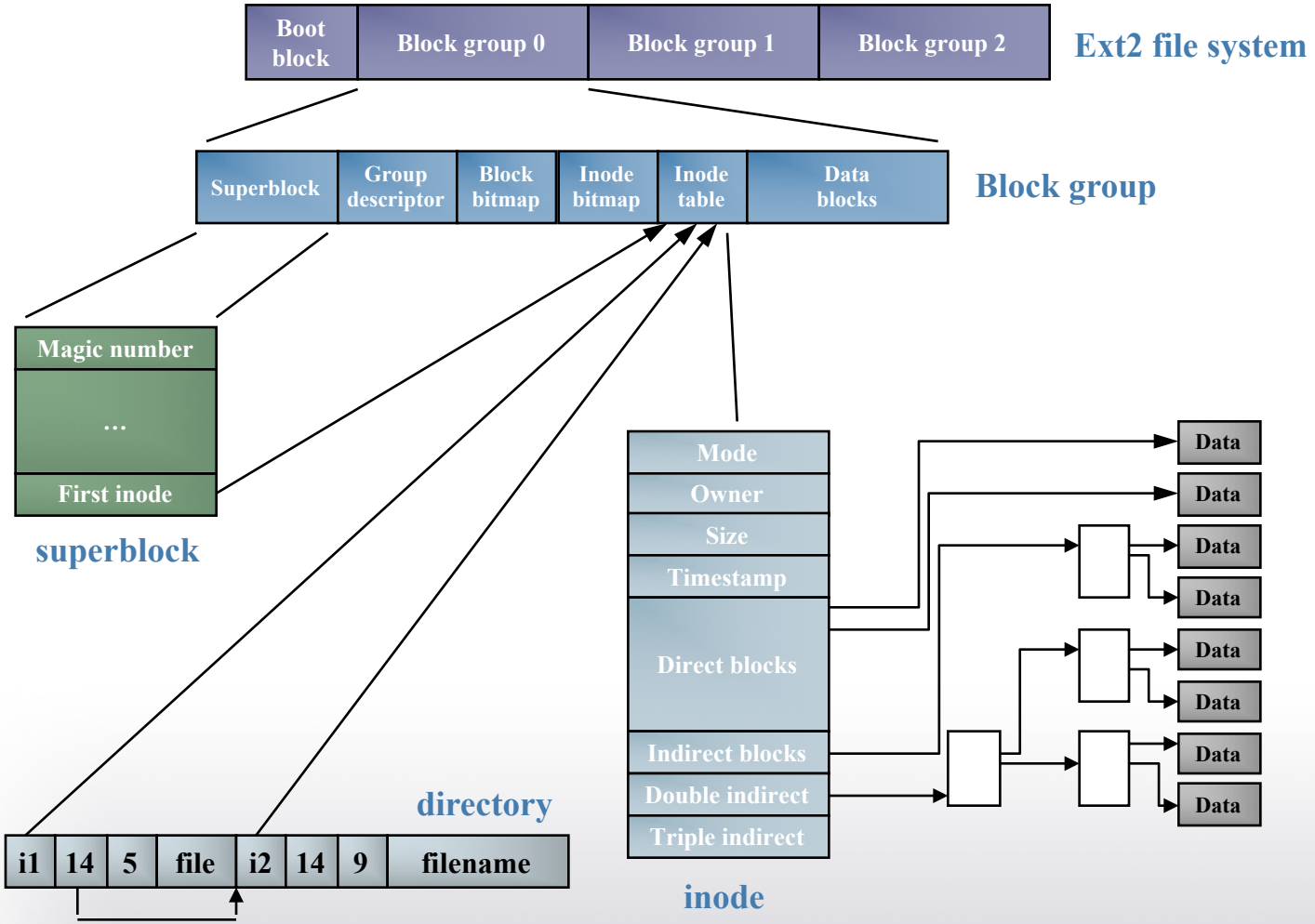
File system type: nfs
Options: rw
Source: atlas:/users/9/3/alean938
Mount point: /home/alean938



Standard hierarchy



Overview of the ext2 filesystem (on disk)

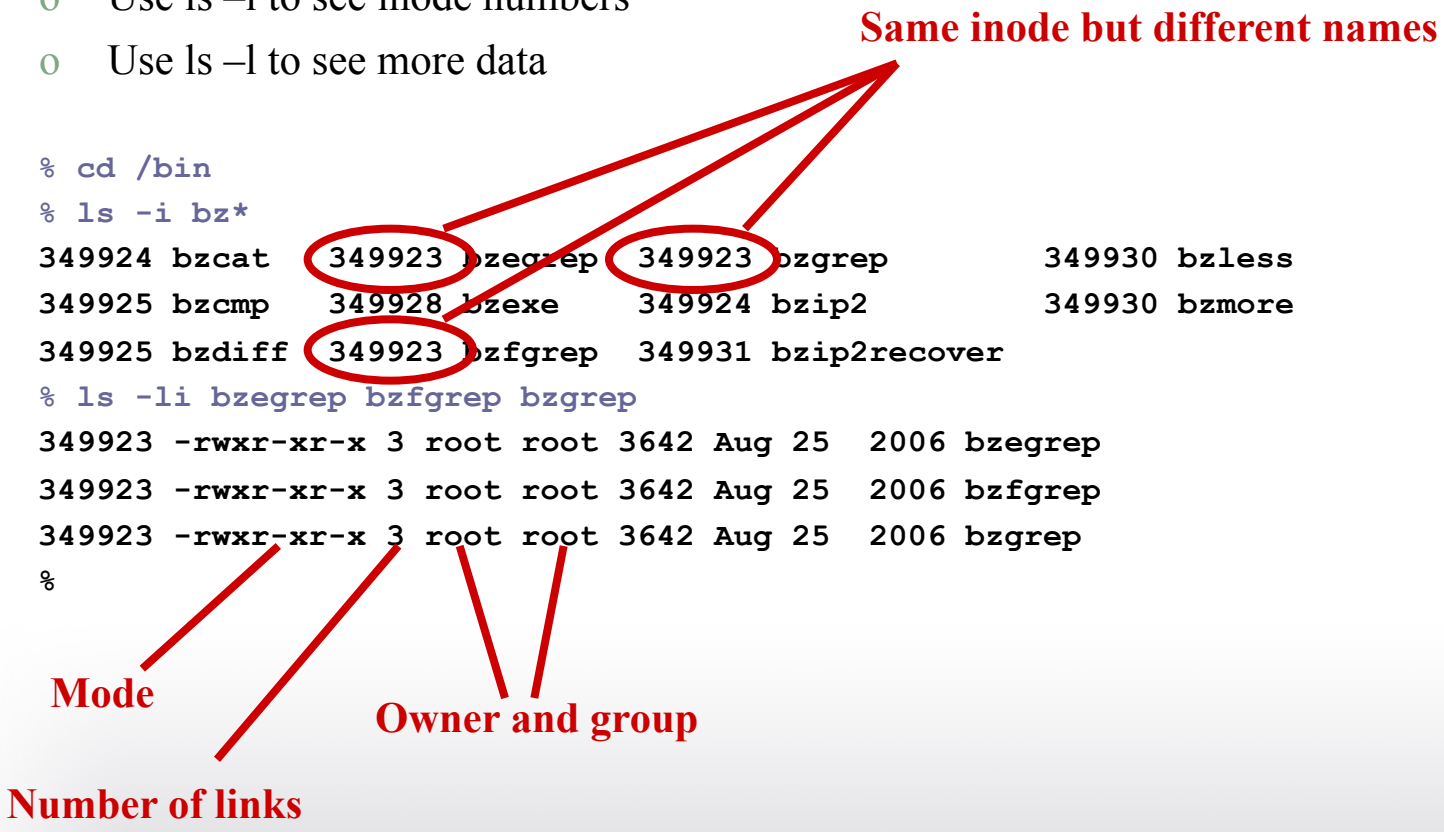


The inode

- Every filesystem object is represented by an inode
  - The VFS does not care about different kinds of objects
  - Higher layers may treat different objects differently
- Some kinds of objects
  - Regular files
  - Directories
  - Device nodes
  - Sockets
  - Pipes
- Some contents of an inode
  - Creation and modification time
  - Mode (permissions)
  - Owner and group
  - Behavior (read, write, etc.)
- The inode does *not* contain the name or location of an object
- Names are *linked* to inodes through directory entries
  - Every inode can have multiple names (known as links)
  - An inode does not need any names at all

An inode example: multiple links

- You can see inodes using the ls command in Linux
  - Use ls –i to see inode numbers
  - Use ls –l to see more data



Another example of multiple links

- Directories have multiple links
  - One for the name
  - One for ”.”
  - One for each ”..” in each subdirectory

```
% ls -di1 /usr /usr/. /usr/bin/.. /usr/local/..
206753 /usr
206753 /usr/.
206753 /usr/bin/..
206753 /usr/local/..
% ls -ldi /usr
206753 drwxr-xr-x 13 root root 4096 Apr 20 12:06 /usr
%
```

Just different names for the same inode

One for /usr

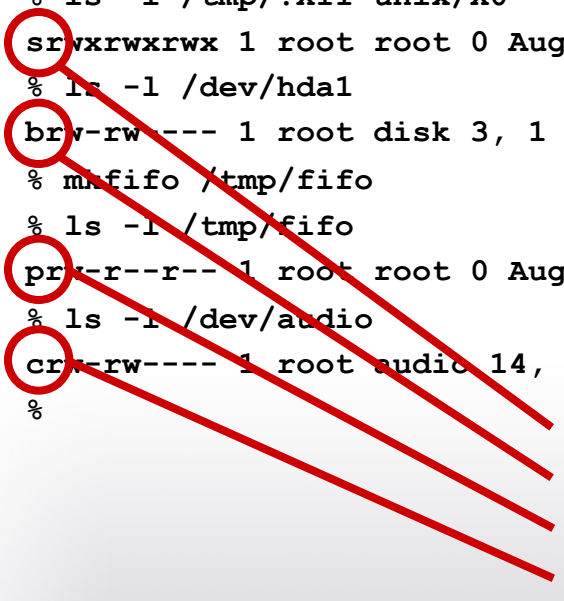
One for /usr/.

Eleven for subdirectories

Another inode example: special files

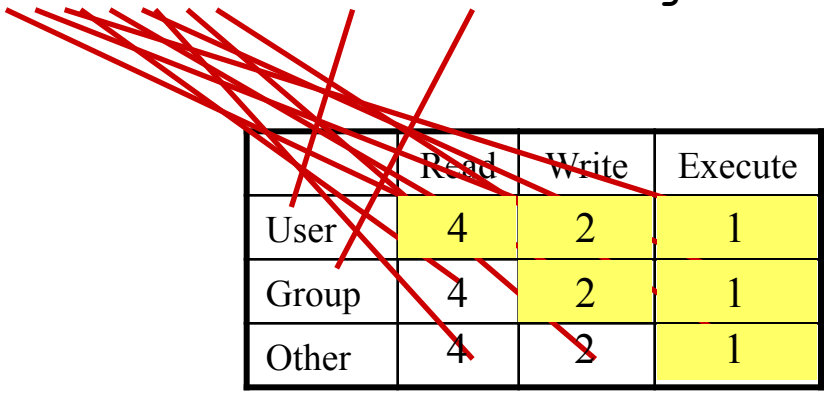
- Non-file and directory objects
  - Block devices (”block special”; e.g. hard disks)
  - Character devices (”character special”; e.g. sound cards)
  - Unix domain sockets
  - Pipes (FIFOs)

```
% ls -l /tmp/.X11-unix/X0
srwxrwxrwx 1 root root 0 Aug 13 13:28 /tmp/.X11-unix/X0
% ls -l /dev/hda1
brw-rw---- 1 root disk 3, 1 Aug 13 13:27 /dev/hda1
% mkfifo /tmp/fifo
% ls -l /tmp/fifo
prw-r--r-- 1 root root 0 Aug 17 11:01 /tmp/fifo
% ls -l /dev/audio
crw-rw---- 1 root audio 14, 4 Aug 13 13:27 /dev/audio
%
```



Permissions

```
% ls -l example
-rwxrwxrwx 1 root root 0 Aug 17 12:27 example
%
```



	Read	Write	Execute
User	4	2	1
Group	4	2	1
Other	4	2	1

Numerical permissions

User:	Read + Write + Execute	=	4 + 2 + 1	= 7
Group:	Read + Execute	=	4 + 2	= 6
Other:	Execute	=	1	= 1
Mode:				**761**

How do ”user” and ”group” work?

- Conceptually: if the user who owns the file attempts to access it, use ”user” permissions; if a user who belongs to the same group as the file attempts to access it, use ”group” permissions
- But how does the operating system know?
- All accesses are performed by processes and…
- …every process has an ”effective user ID” (EUID) and an ”effective group ID” (EGID)
- …every user belongs to a set of groups

Where are users and groups defined

- Depends on what /etc/nsswitch.conf says
  - The ”name service switch” allows users and groups (and other data) to be retrieved from many different sources
  - The default is ”local files” (/etc/passwd and /etc/group)

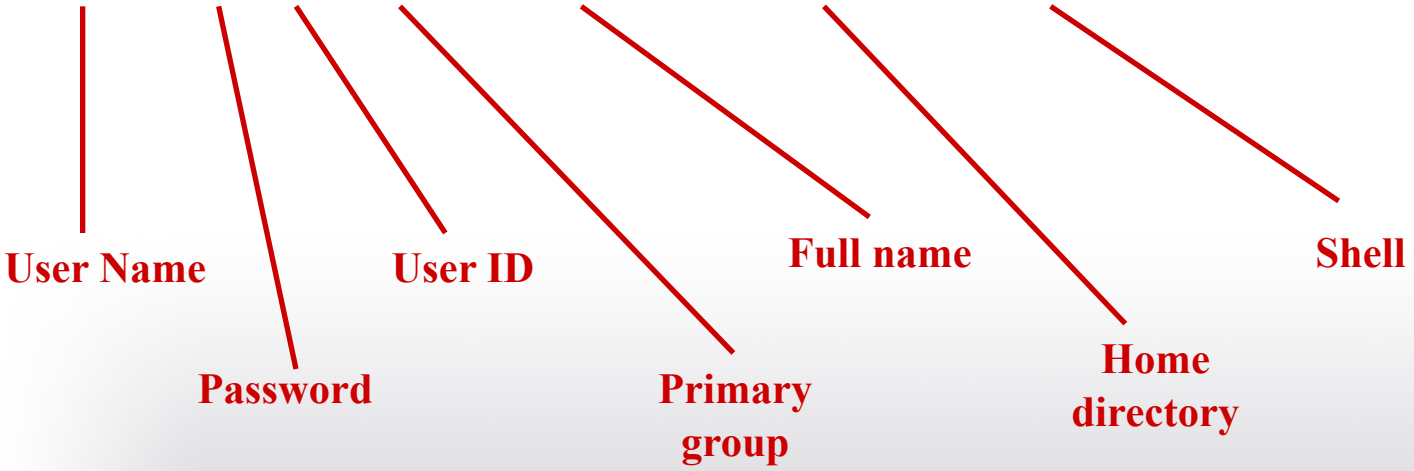
```
% cat /etc/nsswitch.conf
# /etc/nsswitch.conf
passwd:         files nis
group:          files nis
shadow:         files
```

Users are stored in local files and in NIS

Groups are stored in local files and in NIS

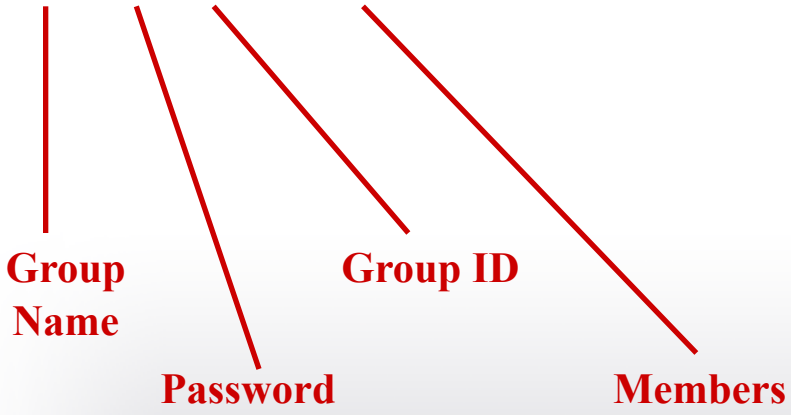
How are users defined?

```
% head -4 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh

% ypcat passwd | head -4
lindu932:*:24083:6000:Linus Dunkers:/home/lindu932:/bin/tcsh
frehi371:*:20883:6000:Fredrik Hillerborg:/home/frehi371:/etc/notsignedsh
thogu383:*:17926:6000:Thomas Gustafsson:/home/thogu383:/etc/closedsh
tanmi350:*:55718:6000:Tanja Mitic:/home/tanmi350:/bin/tcsh
```



How are groups defined?

```
% tail -4 /etc/group
ntp:x:111:
smmta:x:112:
smmsp:x:113:
uml-net:x:114:

% ypcat group | head -4
TDDB34:*:9197:eribe,evalu,johfa,jonlu,mikki
TDDB30:*:9194:diasz,snt,ulfni
TDDA13:*:9161:bouka,patdo
HKGD20:*:9436:bjojo,eriho,jonlu,matar,yuhhu
```



A Look at Linux: Processes

- All execution takes place in processes
  - Each process may consist of several threads
  - Every process has its own (protected) address space
  - Every process has an ID, a parent, and a controlling tty
  - Processes have a state (running, stopped, suspended, etc)

- Processes can communicate
  - Signals are simple asynchronous messages
  - Processes can share memory areas
  - Processes can communicate using pipes
  - Processes can communicate using sockets

Example of processes

```
%  ps -H -eo s,pid,ppid,tty,user,cmd
S    PID   PPID TT       USER     CMD
S      1      0 ?        root     init [2]
S   2188      1 ?        snmp       /usr/sbin/snmpd -Lsd -Lf /dev/null -u snmp
S   2194      1 ?        root       /usr/sbin/sshd
S  24294   2194 ?        root         sshd: davby [priv]
S  24296  24294 ?        davby          sshd: davby@pts/0
S  24297  24296 pts/0    davby            -sh
R  24304  24297 pts/0    davby              ps -H -eo s,pid,ppid,tty,user,cmd
S   2206      1 ?        uml-net    /usr/bin/uml_switch -tap tap0 -unix
S   2273      1 ?        statd      /sbin/rpc.statd
S   2297      1 ?        root       sendmail: MTA: accepting connections
S   2323      1 ?        ntp        /usr/sbin/ntpd -p /var/run/ntpd.pid
S   2333      1 ?        daemon     /usr/sbin/atd
```

Process ID

Parent process ID

Controlling terminal

EUID

Command

Signals

- User point-of-view: suspend, resume, kill processes

```
% ps axu | grep '[e]macs'
davby    24613  0.6  0.2   9604  4596 pts/1    S+   15:47   0:00 emacs -nw
% kill -HUP 24613
% ps axu | grep '[e]macs'
%
```

- Send arbitrary signals using kill command
- If typing directly to process's controlling tty
  - C-c sends INTR
  - C-z sends TSTP
  - C-\ sends QUIT

Privilege elevation

- Users gain extra privileges by changing EUID or starting processes with a different EUID than the current one

```
%  ps -H -eo s,pid,ppid,tty,user,cmd
S    PID   PPID TT       USER     CMD
S      1      0 ?        root     init [2]
S   2194      1 ?        root       /usr/sbin/sshd
S  24294   2194 ?        root         sshd: davby [priv]
S  24296  24294 ?        davby          sshd: davby@pts/0
S  24297  24296 pts/0    davby            -sh
R  24321  24297 pts/0    davby              ps -H -eo s,pid,ppid,tty,user,cmd
S  24312   2194 ?        root         sshd: davby [priv]
S  24314  24312 ?        davby          sshd: davby@pts/1
S  24315  24314 pts/1    davby            -sh
S  24319  24315 pts/1    root               passwd
```

sshd changed EUID from root to davby

passwd being run by davby with EUID root

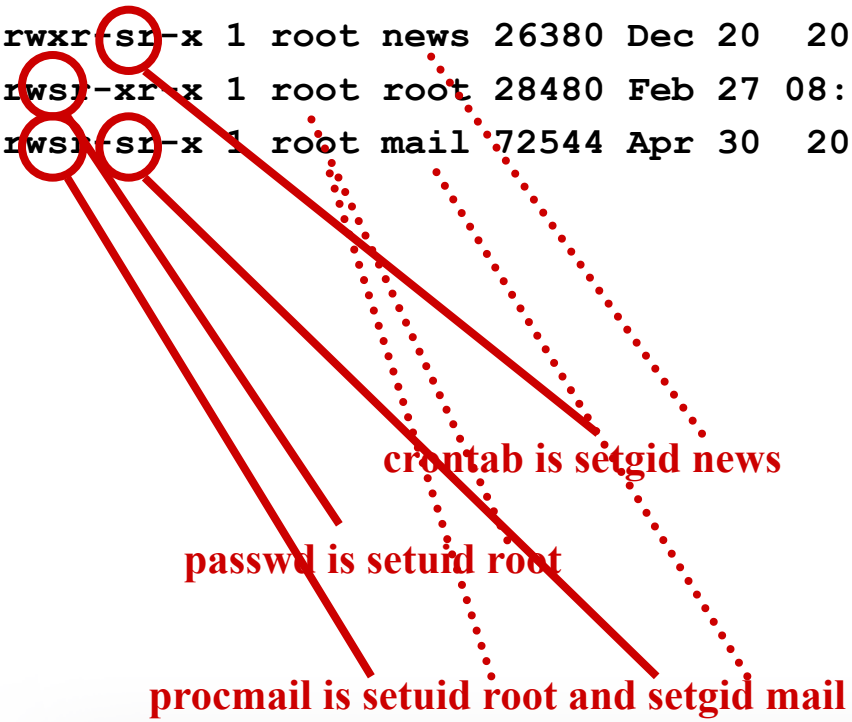
How does privilege elevation work?

- Programs can change their own EUID/EGID
  - The seteuid system call changes the EUID
  - The setegid system call changes the EGID
  - Very strict limitations on who can change to what

- Programs can have the setuid/setgid bits set
  - When setuid program started, process assumes file owner as EUID
  - When setgid program started, process assumes file group as EGID

Example of setuid/setgid programs

```
% ls -l passwd crontab mail
-rwxr-sr-x 1 root news 26380 Dec 20  2006 crontab
-rwsr-xr-x 1 root root 28480 Feb 27 08:53 passwd
-rwsr-sr-x 1 root mail 72544 Apr 30  2006 procmail
%
```



The shell

- When a user logs in, the login program starts a shell
- The shell accepts and interprets commands from the user
  - Handles I/O redirection, environment variables, etc
- Two kinds of commands: built-in and external
  - Built-in: affect the shell itself (e.g. cd) or are run often (e.g. echo)
  - External: most everything else
  - Also: programming structures (e.g. if-then-else)
- External commands are just files with the execute permission set that are in a directory listed in the PATH variable

System startup

- What happens when you start Linux
  1. The computer firmware (BIOS) loads the boot loader
  2. The boot loader loads and executes the operating system
  3. The operating system runs the /sbin/init program
  4. The /sbin/init program does what /etc/inittab says to do

```
% cat /etc/inittab
id:2:initdefault:
si::sysinit:/etc/init.d/rcS
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
1:23:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
%
```

Set default run level to 2

To do when initializing system

To do (once) when entering run level 2

To do when entering run level 2 or 3 (and when process terminates it is restarted

Typical (sysvinit) system startup

- /etc/init.d/rc script is run with run level as argument
  - Runs scripts in /etc/rc*N*.d directory
  - Scripts that start with K are run with argument **stop**
  - Scripts that start with S are run with argument **start**
- Note that /etc/rcS.d scripts are also run during boot

```
% ls /etc/rc2.d
S10sysklogd          S20cupsys            S20snmpd             S21exim
S11klogd             S20dbus              S20ssh               S23ntp
S18portmap           S20devtun-rights     S20sysfsutils        S89atd
S19autofs            S20dirmngr           S20sysinfo           S89cron
%
```

Run with argument start to start ssh service

System logging

- Linux (and unix) systems and software are often very chatty
  - Detailed messages often show up in system and application logs
  - By default, log files are stored in /var/log

```
% ls -F /var/log
account/          cfengine.log.0      fsck              mail.log.0       quagga/
acpid             cfengine.log.1.gz   kern.log          mail.log.1.gz    samba/
acpid.1.gz        daemon.log          kern.log.0        mail.warn        syslog
apache/           daemon.log.0        kern.log.1.gz     mail.warn.0      syslog.0
apache2/          daemon.log.1.gz     ksymoops/         mail.warn.1.gz   syslog.1.gz
aptitude          debug               lastlog           messages         user.log
aptitude.1.gz     debug.0             lpr.log           messages.0       user.log.0
auth.log          debug.1.gz          mail.err          messages.1.gz    user.log.1.gz
auth.log.0        dmesg               mail.err.0        mysql/           uucp.log
auth.log.1.gz     dpkg.log            mail.err.1.gz     nagios/          vtund/
boot              dpkg.log.1          mail.info         nessus/          wtmp
btmp              fai/                mail.info.0       news/            wtmp.1
btmp.1            faillog             mail.info.1.gz    ntpstats/        wtmp.report
cfengine.log      fontconfig.log      mail.log          pycentral.log
%
```

A Look at Linux: System logging

- Controlled by system logging service (syslogd)
  - Configured in /etc/syslog.conf
  - Controls what information goes where
  - Controls what **level** of information is logged

```
% head -2 /var/log/syslog.conf
auth,authpriv.*                 /var/log/auth.log
*.*;auth,authpriv.none          -/var/log/syslog
```

Log everything except auth messages to /var/log/syslog

System logging

```
% /etc/init.d/bind9 reload
% tail -12 /var/log/syslog
Aug 20 08:29:22 sysinst-gw postfix/cleanup[26219]: 3FE26748B9: message-
id=<20070820062922.3FE26748B9@sysinst-gw.sysinst.ida.liu.se>
Aug 20 08:29:22 sysinst-gw postfix/bounce[26258]: F24931F4B7: sender non-
delivery notification: 3FE26748B9
Aug 20 08:29:22 sysinst-gw postfix/local[26259]: 3FE26748B9: to=<nagios@sysinst-
gw.sysinst.ida.liu.se>, relay=local, delay=0.16, delays=0.05/0.04/0/0.07,
dsn=2.0.0, status=sent (delivered to mailbox)
Aug 20 09:00:54 sysinst-gw named[7673]: loading configuration from
'/etc/bind/named.conf'
Aug 20 09:00:54 sysinst-gw named[7673]: zone 189.236.130.in-addr.arpa/IN: loaded
serial 2007081500
Aug 20 09:00:54 sysinst-gw named[7673]: zone 189.236.130.in-addr.arpa/IN:
sending notifies (serial 2007081500)
Aug 20 09:00:54 sysinst-gw named[7673]: zone sysinst.ida.liu.se/IN: loaded
serial 2007081500
Aug 20 09:00:54 sysinst-gw named[7673]: zone sysinst.ida.liu.se/IN: sending
notifies (serial 2007081500)
Aug 20 09:00:54 sysinst-gw named[7673]: client 130.236.177.25#34505: transfer of
'189.236.130.in-addr.arpa/IN': AXFR-style IXFR started
Aug 20 09:00:54 sysinst-gw named[7673]: client 130.236.177.25#34505: transfer of
'189.236.130.in-addr.arpa/IN': AXFR-style IXFR ended
```

Where to next?

- Any basic textbook on Linux will teach you more
- The UML lab will familiarize you with the virtual environment you will be using (and in the virtual environment you root access, so you can do anything you want)
- The LXB lab will go through a large number of basic Linux issues you need when installing and managing systems