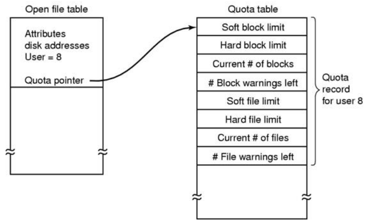
File systems: management

Disk quotas for users



- Quotas for keeping track of each user's disk use
- Soft limit and hard limit

Backup

File System Backup

- Replacing hardware is easy, but not the data
- Backups to handle
  - Recovery from disaster
    - Irrevocably lost file system
  - Recovery from stupidity
    - User accidentally removed a file
- Making a backup takes a long time and a large amount of space
- Whole file system or just part of it?
- Which media to use for backup?

Backup decisions

- Complete vs incremental dump
  - Unchanged files
- Compression of the backup
  - Single bad spot could foil the decompression
- From active or offline system
  - Takes hours in large systems
  - Active changes all the time, how to define snapshot
- Security and protection of the backup (tapes)
  - Storage location

How

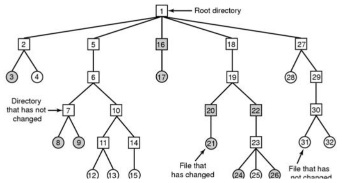
Physical dump

- Dumps blocks of the disk starting from 0
- Simple
- Free blocks? Bad blocks?
- Unable to
  - Skip selected directories
  - Make incremental dump
  - Restore individual file

Logical dump

- Dumps files and directories starting from specified dir changed after a given time
  - Incremental : last backup
  - Complete: installation day
- FS Structure in the dump
  - Restoration of individual files
- Very common

Logical dump algorithm



Dump changed files and full path to them

(a) First scan:

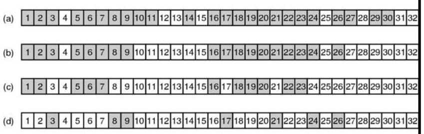
mark changed files and all dirs

(b) Second scan:

unmark dirs that do not have changes in subtree

(c) Dirs to dump

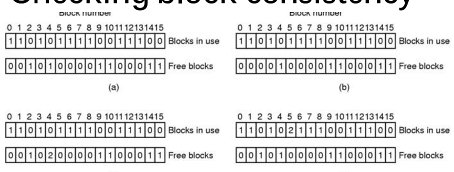
(d) Files to dump



Consistency

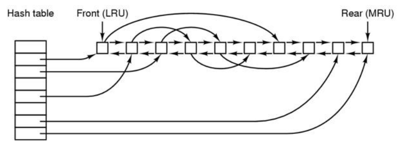
- Blocks are read to memory, modified ther and written back to disk later
- System crash?
  - Some block not yet written to disk
- Check consistency situation
  - Fsck, Scandisk, ...
- Check blocks
  - Check that each block is either free or belongs to a file
- Check directories
  - Each file must be accessed from the given number of dirs

Checking block consistency



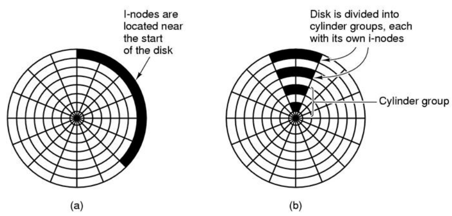
- File system states
  - (a) consistent - everything OK
  - (b) missing block - not used, not free, add to free list
  - (c) duplicate block in free list - rebuild the list
  - (d) duplicate data block - make extra copy to one of the files

Block cache / Buffer cache



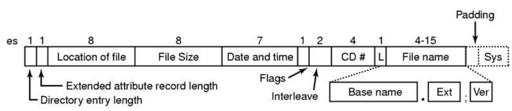
- Access to disk is much slower than access to memory
- Keep recently used blocks in memory for future need
- Fast access using hash with collision chain
- Use write-through caches
  - To maintain content on disk also and
  - keep disk consistency (i-nodes is essential on disk)

Disk layout: reduce arm motion



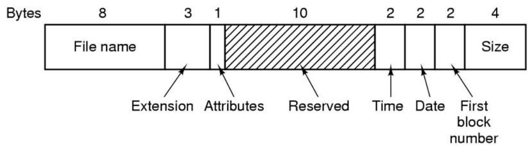
- I-nodes placed at the start of the disk
- Disk divided into cylinder groups
  - each with its own blocks and i-nodes

Example File Systems
CD-ROM File Systems



The ISO 9660 directory entry

The MS-DOS File System (1)



The MS-DOS directory entry

13

The MS-DOS File System (2)

Block size	FAT-12	FAT-16	FAT-32
0.5 KB	2 MB		
1 KB	4 MB		
2 KB	8 MB	128 MB	
4 KB	16 MB	256 MB	1 TB
8 KB		512 MB	2 TB
16 KB		1024 MB	2 TB
32 KB		2048 MB	2 TB

- Maximum partition for different block sizes
- The empty boxes represent forbidden combinations

14

UNIX V7 File System

15

UNIX

- Directory entry has file name and i-node number (information node)
  - Same file can have different names in directories
- File attributes in i-node (64 B)
  - uid, gid
  - Size in bytes
  - File type
  - Bits describing access rights
    - user rwx, group rwx and others rwx
  - Number of links to the file
  - Time stamps: modified, accessed, i-node change

16

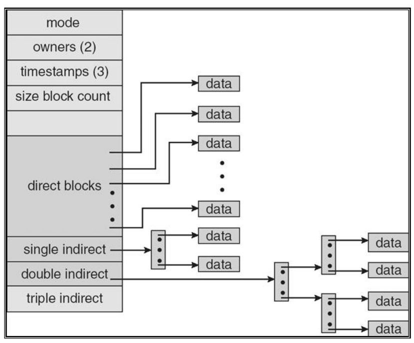
UNIX

- i-node contains 13 elements for block numbers
  - 10 directly to file blocks
  - 1 single indirect to block containing 256 block numers
  - 1 double indirect to block containing 256 block numbers to blocks containing 256 file block numbers
  - 1 triple indirect ...
- Most files short – accessed using the direct blocks
- Largest possible file size > 16 GB
  - BUT i-node only has 32-bit size field => max 4GB

17

UNIX i-node

[SGG07] Fig 11.9



18

UNIX

Tan08 4-35

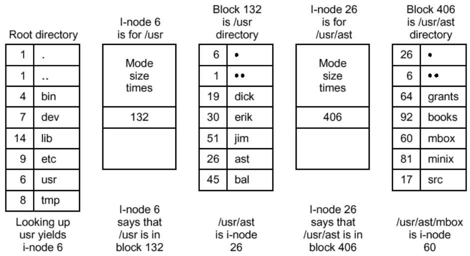


Fig. 6-39. The steps in looking up */usr/ast/mbox*.

19

UNIX

Tan01 10-31

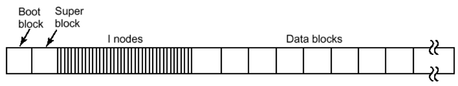
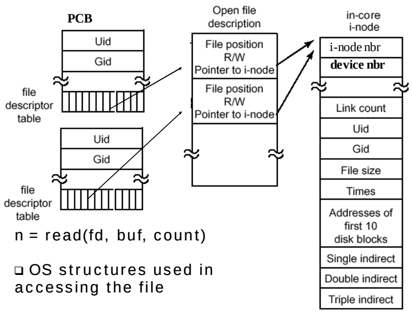


Fig. 10-31. Disk layout in classical UNIX systems.

- Location of boot-block is fixed by manufacturers, not unix specific
- i-nodes collected to one location, because they are small and block size is much larger. I-nodes quite often read to memory and used from there.
- Super block contains at least device#, partition size, start of the free block list, some numbers of free i-nodes

20

UNIX



n = read(fd, buf, count)

- OS structures used in accessing the file

21

NFS
Network File System

Chapter 10.6.4.

22

NFS

- Goal: Join file systems of separate (remote) computers into one logical file system in your own computer
  - Developed by Sun Microsystems
- NFS-protocol
  - Use request-reply -model
  - Just the communication interface definition
    - two roles: NFS-client, NFS-server
- Windows has SMB-protocol for the same purpose
  - Server Message Block

23

NFS architecture

- Server *(Palvelija)*
  - Executes the NFS server program
  - Defines access rights in public directory **`/etc/exports`**
- Client *(Asiakas)*
  - Executes the NFS client program
  - Mounts the remote directory to its local directory hierarchy
  - Mount points defined in file **`/etc/fstab`**
- Virtual file system (VFS) notifies when the file access goes to a mounted remote nfs
  - Deliver the request to the nfs server
  - The server's internal file system structure is not important

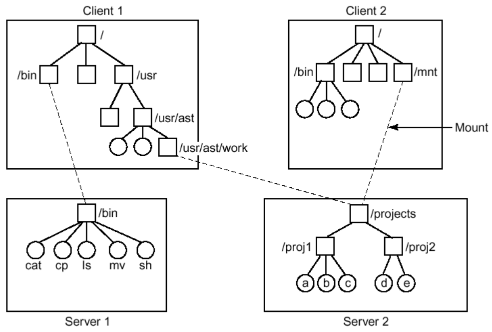
24

More information and examples

- See the content of /etc/fstab on any department Linux
  - cat /etc/fstab
- See the currently mounted file systems
  - cat /etc/mtab
  - df - shows the sizes and free space on those
- man mount gives you detailed information

- NFS protocol v3 – RFC 1813
- NFS protocol v4 – RFC 3530

25



(Fig 10-35, [Tan 08])

26

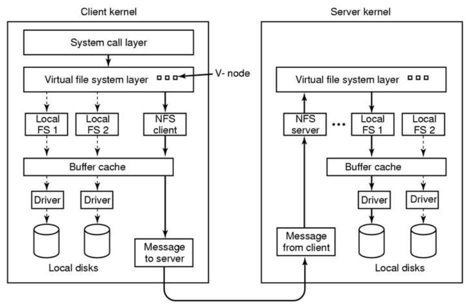
NFS protocol: mount

- Client requests a permission to mount
  - Client sends the path name to server
  - Server return a file handle (*kahva*) to identify the mounted dir
    - file system type, device#, inode#, access rights
    - handle is used in all future requests
- When?
  - During boot: /etc/rc initialzation script makes the mount requests
  - At first reference: automounting
    - Makes its possible to try several servers in parallel (assume identical file systems)

27

NFS layer structure

Fig. 10-36 [Tane 08]



28

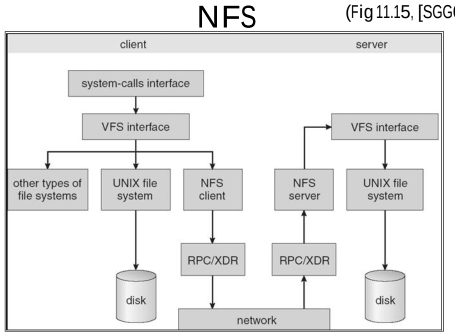
NFS protocol: directory and file access

- Client send requests to manipulate files and directories
  - read(), write(), ...
  - Not open or close!
  - Before accessing a file a lookup operation is needed to get file handle
- Stateless protocol (V3)
  - Each request contains all necessary information
    - file handle, access position, amount, …
  - No synchronization
    - Different users can access (write) the same file simultaneously
    - Client may cache, but server do not know about caches
- Stateless server does not remember anything
  - Restart of a server does not loose any information about file state

29

NFS

(Fig 11.15, [SGG07]



NFS implementation often based on Remote Procedure Call (RPC) that uses eXternal Data Representation (XDR) in its messages

30

NFS v4

- Version 4 server is stateful
  - Server maintains information about each open file
- Support for file locking
- Support for strong security (and its negotiation)
- Compound operations
  - Bundle multiple NFSv3 operations in one network transaction
- Client caching
  - Client must check with each open the cache validity
  - With locks on the server the client caches can remain coherent

31

I/O Hardware

Chapter 5.1

32

Input / Output

- Hardware
  - Block Devices: hard disk, CD-ROM, USB stick, ...
    - Information in fixed-size blocks
  - Character Devices: printer, keyboard, mice, network interface, ...
    - Information is a stream (of characters)

Fig 5-1: Some typical device, network, and bus data rates

Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Scanner	400 KB/sec
Digital camcorder	3.5 MB/sec
802.11g Wireless	6.75 MB/sec
52x CD-ROM	7.8 MB/sec
Fast Ethernet	12.5 MB/sec
Compact flash card	40 MB/sec
FireWire (IEEE 1394)	50 MB/sec
USB 2.0	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
SATA disk drive	300 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec

Figure 5-1. Some typical device, network, and bus data rates.

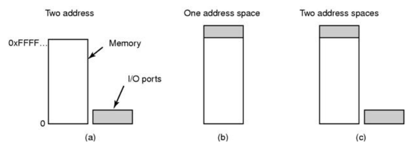
33

Device Controllers

- I/O devices have components:
  - mechanical component
  - electronic component
- The electronic component is the device controller
  - may be able to handle multiple devices
- Controller's tasks
  - convert serial bit stream to block of bytes
  - perform error correction as necessary
  - make available to main memory
- Controller has registers for communication with CPU
  - OS writes commands etc to them
  - OS reads status etc from them

34

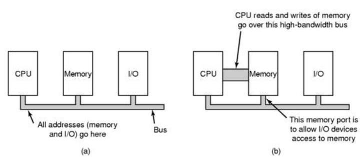
How to access controller's registers?



- Separate I/O and memory space (a)
  - Assign an I/O port number for each control register
- Memory-mapped I/O (b)
  - Assign a unique memory address with no memory for register
- Hybrid (c)
  - I/O port number for registers, Memory-mapped I/O for buffers

35

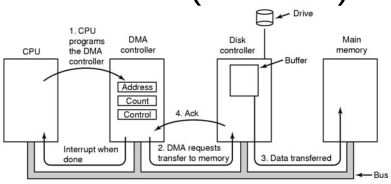
Memory-Mapped I/O



- Using memory-mapped I/O is just a memory reference
- In the architecture
  - Ability to selectively disable caching
    - Device control registers must not be cached
  - The memory-references must reach the correct controller
    - In single-bus easy: controller just picks its own references
    - With multiple buses: snooping device, special chip to detect, …

36

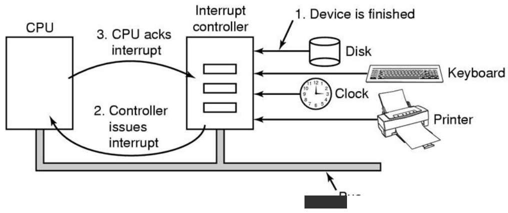
Direct Memory Access (DMA) – controller (hardware)



- The phases of DMA controlled disk read are in the figure
  - In this fly-by mode, the disk controller writes to memory
  - Alternative is to send data first to DMA, which then writes
- Bus reservation
  - Cycle stealing or burst mode

37

Interrupts Revisited



How interrupts happens. Connections between devices and interrupt controller actually use interrupt lines on the bus rather than dedicated wires

38