CMPSCI 577 Operating Systems Design and Implementation **Spring 2020**

Lecture 17: 03/31

Lecturer: ***Prashant Shenoy*** *Scribe:* ***Shashwat***

17.1 File Systems Overview

This lecture discusses about File system which maintains in-memory data structures that allows access to on-disk files and implements all kinds of interfaces that both user and processes use to get access to their files.

17.2 File System abstraction

Device interface: It provides access to the hard disk through a device driver.

Device independent interface: It provides a high level abstraction to users. In this case abstraction is one of the files. This system is open to application processes.

17.3 User-Space I/O Software

All of the device drivers and most of the kernel tasks are implemented as user space processes. Device driver sets up the device registers and waits for the interrupt to come.

17.4 User requirements on data

Persistence:- Data should stay across reboots. Data on RAM is volatile and is non-volatile in case of hard disks. Speed: Data structures used should allow to access data quickly and efficiently Size: Ability to store lots of data on disk. File system should have no restriction on file sizes except the size of disk. Sharing: Users can share files in a fine grain manner Ease of use: Can get to data in later time.

17.5 Hardware/OS Features

Hardware: It provides persistence, speed through random access and size.

OS provides: Persistence, sharing/protection through access control list, ease of use in terms of human readable file names in which OS maps to internal user id and also provides ease of search on the basis of contents of file data.

17.6 Files

These are named collection of related information and recorded on secondary storage. It can contain program or data and can be structured or unstructured. File attributes include name, type, location,size, protection and creation time.

17.7 User interface to the file system

It includes common file operations which includes data operations like create(), open(), read() etc. and also includes naming operations such as SetAttribute() and GetAttribute()

17.8 OS File Data Structures

Open file table- Tracks files which are open and are currently being accessed by multiple processes in the OS.

Per process file table- Has pointers to file buffers.

17.9 File Operations

Create: System call to create a new file. Operations include – allocate disk space, create file descriptor, add file descriptor to the directory containing the file and add a file type which is optional. UNIX doesn't have a file type associated with it.

Delete: Finds the directory containing the file, frees the disk blocks used by the file and removes the file descriptor from the directory.

Open: Check if the file is already open by another process, if not, find the file, copy the file descriptor, check protection-if not okay then abort, increment the open count and finally create an entry in the process's file table.

Close: Remove the entry for the file in the file table, decrement the open count

Read(fileID,from,size, bufAddress): It is a random access which reads "size" bytes from file position "from" into "bufAddress" Read(fileID,size, bufAddress): It is a sequential access

Write: Similar to read, but copies from the buffer to the file.

Seek: Can read from any location by changing the file offset

Memory Mapping: It takes a file and maps it to a address space that extends the RAM and access to the file takes place as if it's part of RAM.

17.10 File access methods

Programmers perspective

Sequential: Data is processed in order.
Keyed: A block is accessed based on a key value

OS perspective

Sequential: Keep a pointer to the next byte in the file. Update on each read/write

Random: Address any block in the file given its offset within the file.

17.11 Naming and Directories

OS uses numbers for each files.

Single directory: Early personal computers used this strategy since their disks were very small. Two level directory: Each user has a separate directory though all of each users file must have unique names. Multiple directories: Can have arbitrary nesting.

17.12 Referential naming

Multiple names are given to a file using soft and hard links

Soft Links: A soft link only makes a symbolic pointer from one file to another.
Hard Links: Creates two entries that point to the same file.

For individual files both hard and soft links can be created. Soft links but not hardlinks can be used for directories

17.13 Directory operations

Search, Create, Delete, List, Rename and traverse.

17.14 Protection

Grant or deny access to file operations depending on protection information. Windows NT maintains a list for each file with user name and type of access whereas UNIX maintains access control bits where there are three categories of users(Ownerm, Group and World) and three types of access privileges(R,W,X).

17.15 Disks

The disk surface is circular and coated with a magnetic material. Tracks are concentric rings on disks and each track is split into sectors or blocks. Disks are organized in disk pack consisting of a stack of platters and use both sides of the platters.

17.16 Disk Overheads

Overhead:- Time taken by CPU to start a disk operation

Latency: The time to initiate a disk transfer of 1 byte to memory

Bandwidth: Rate of I/O transfer

17.17 File Organization on disk

We can use Logical Block Addressing for disk rather than physical addresses as well. File table tracks the blocks of file.

On disk data structures: The structure used to describe where the file is on the disk and attributes of the file is the file descriptor. It has to be stored on disks just like files.

17.18 Contiguous Allocation

OS maintains an ordered list of free disk blocks and allocates a contiguous chunk of free blocks when it creates a file. Only size and start location needs to be stored in the file descriptor.

17.19 Linked Files

Keeps a list of all the free sectors/blocks, a pointer to the first sector/block in the file descriptor. In each sector a pointer to the next sector is maintained. Example includes MS-DOS

17.20 Indexed files

OS keeps an array of block pointers for each file and the OS must declare the maximum length of the file when it is created. An array is maintained to hold pointers to all blocks but blocks are allocated on demand.

17.21 Multilevel indexed files

Each file descriptor contains 14 block pointers where first 12 pointers point to data blocks.

17.22 Free-Space management

If there is a need to find and release free space quickly then a bitmap is used, in which bit-1 denotes that the block is free and 0 denotes that its allocated. An alternative implementation to bitmaps is to link together free blocks wherein each block contains a pointer to the next free block.

17.23 Super blocks

Block 0 is typically called super block. It keeps metadata for the file system.They store inodes sequentially. A real file system has more than 1 super block because you precreate more number of inodes.