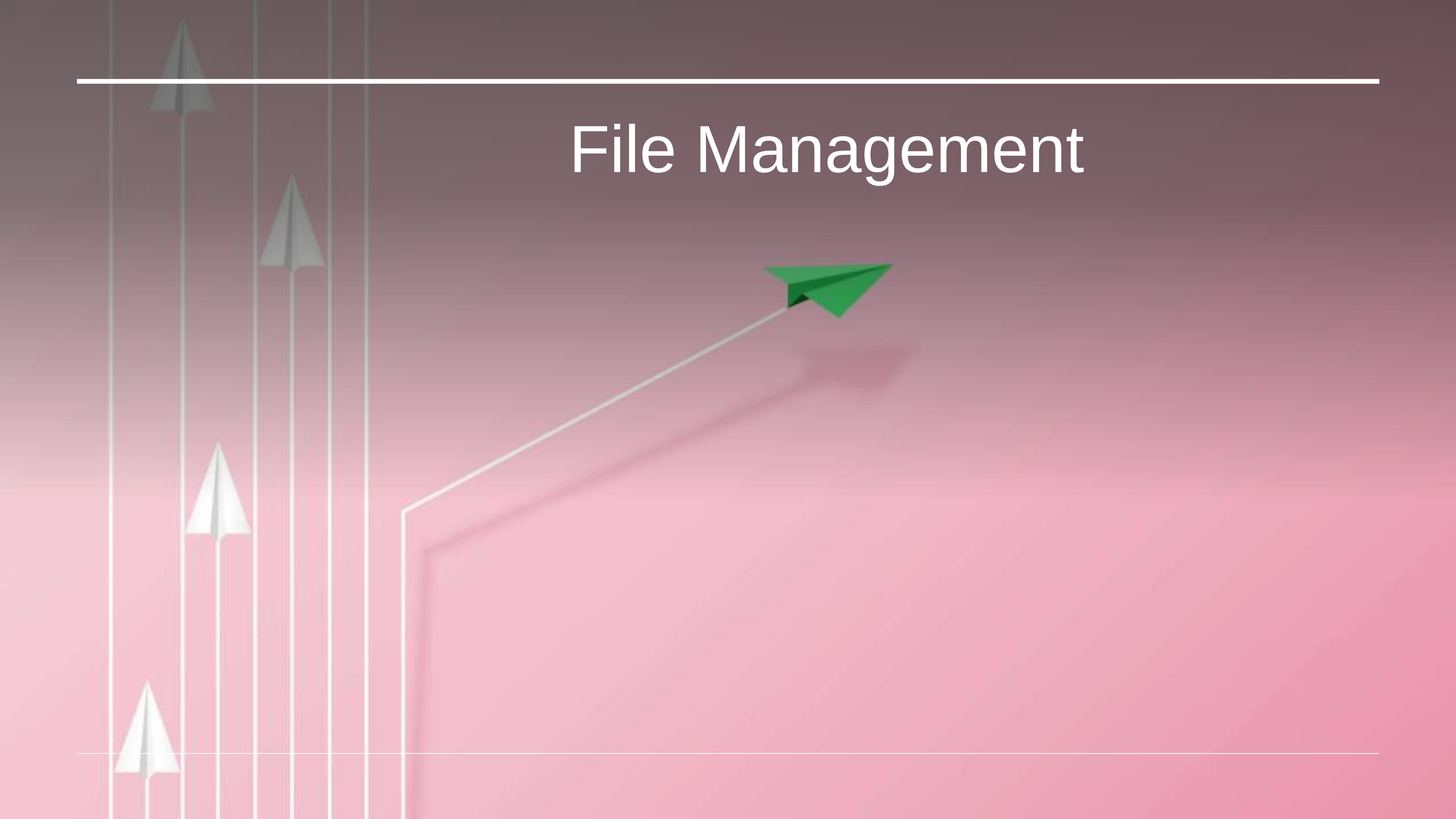
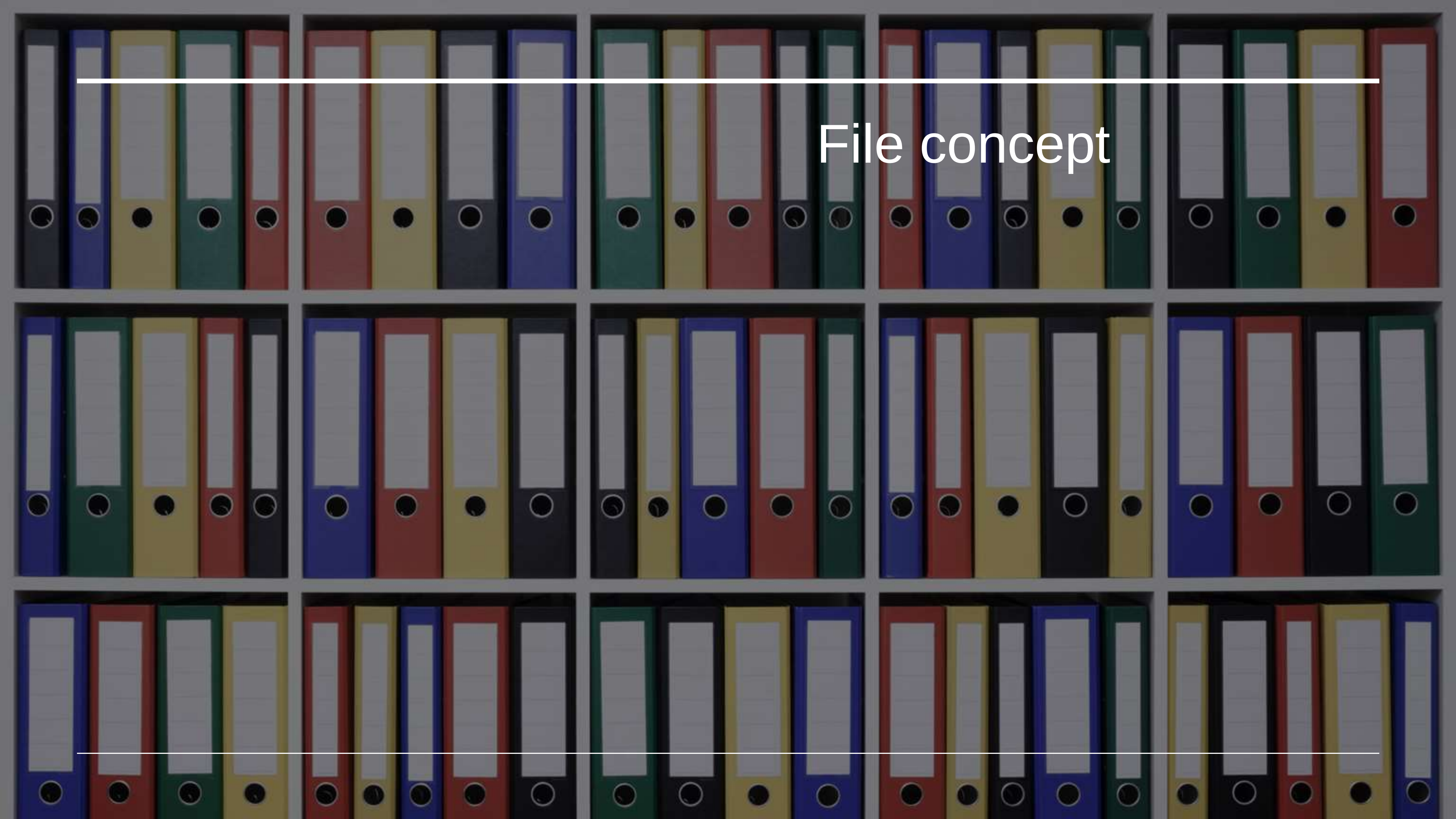
File Management





File concept

File Concept

- File is a named collection of related information that is recorded on secondary storage.
- From a user’s perspective, a file is the smallest allotment of logical secondary storage; that is, data cannot be written to secondary storage unless they are within a file.
- information in a file is defined by its creator. Many different types ofinformation may be stored in a file—source or executable programs, numeric or text data, photos, music, video, and so on.

File Concept-File Atribute

- **Name**. The symbolic file name is the only information kept in humanreadable form.

• **Identifier**. This unique tag, usually a number, identifies the file within the f ile system; it is the non-human-readable name for the file

. • **Type**. This information is needed for systems that support different types of files.

• **Location**. This information is a pointer to a device and to the location of the file on

that device.

Size. The current size of the file (in bytes, words, or blocks) and possibly the maximumallowed size are included in this attribute.

• **Protection**. Access-control information determines who can do reading, writing, executing, and so on.

• **Timestamps and user identification.** This information may be kept for creation, last modification, and last use. These data can be useful for protection, security, and usage monitoring.

File Operations

Creating a file. . Two steps are necessary to create a file. First, space in the f ile system must be found for the file.Second, an entry for the new file must be made in a directory.

Opening a fil e

Writing a file

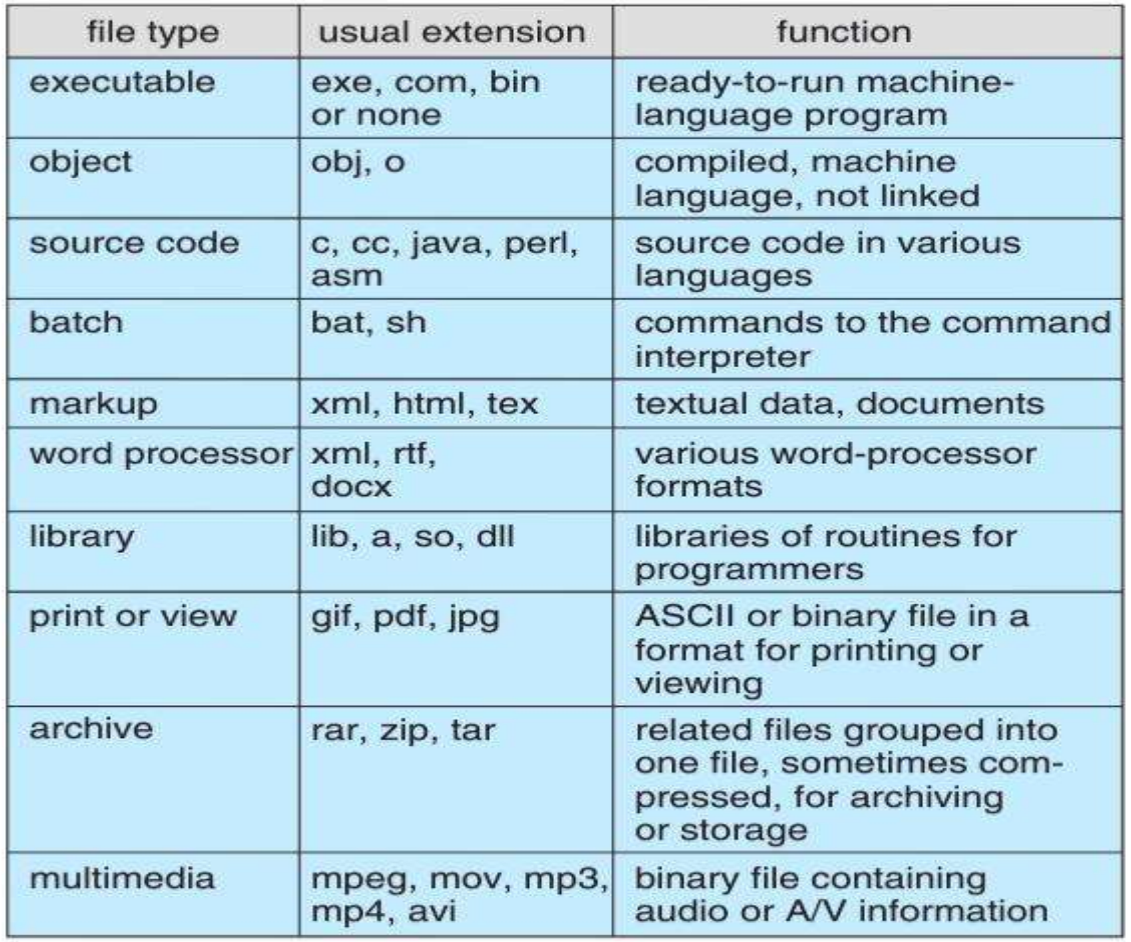
Reading a file

Repositioning within a file

Deleting a file

Truncating a file

File Types



file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, perl, asm	source code in various languages
batch	bat, sh	commands to the command interpreter
markup	xml, html, tex	textual data, documents
word processor	xml, rtf, docx	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	gif, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	rar, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, mp3, mp4, avi	binary file containing audio or A/V information

Figure 13.3 Common file types.

Access Methods

- Files store information. When it is used, this information must be accessed and read into com

puter memory. The information in the file can be accessed in several ways.

1 Sequential Access

The simplest access method is sequential access. Information in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editors and compilers usually access files in this fashion.

read next()—reads the next portion of the file and automatically advances a file pointer, which tracks the I/O location. Similarly, the write operation—write next()—appends to the end of the file and advances to the end of the newly written material

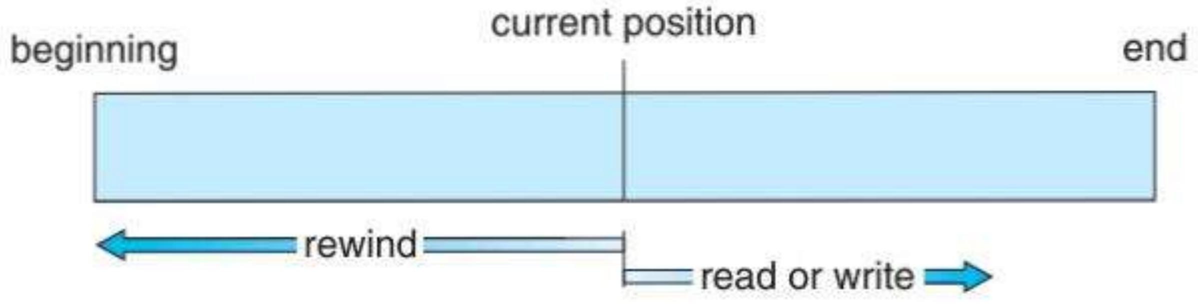


Figure 13.4 Sequential-access file.

Access Methods

- 2 **Direct Access**
- a file is made up of fixed-length logical records that allow programs to read and write records rapidly in no particular order. The direct-access method is based on a disk model of a file, since disks allow random access to any file block.
- wemayreadblock14, thenread block 53, and then write block 7. There are no restrictions on the order of reading or writing for a direct-access file.

Access Methods

- 3 **Other Access Methods**
- These methods generally involve the construction of an index for the file. The index, like an index in the back of a book, contains pointers to the various blocks. To f ind a record in the file, we first search the index and then use the pointer to access the file directly and to find the desired record.

Directory Structure

- The directory can be viewed as a symbol table that translates file names into their directory entries. If we take such a view, we see that the directory itself can be organized in many ways. The organization must allow us to insert entries, to delete entries, to search for a named entry, and to list all the entries in the directory
- 1 **Single-Level Directory**
- The simplest directory structure is the single-level directory. All files are contained in the same directory, which is easy to support and understand.
- A single-level directory has significant limitations, however, when the number of files increases or when the system has more than one user. Since all f iles are in the same directory, they must have unique names.

Directory Structure

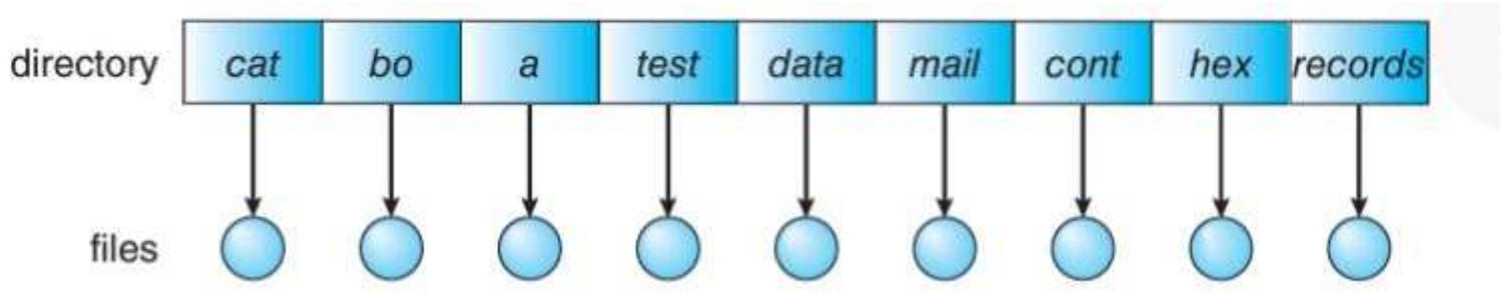


Figure 13.7 Single-level directory.

Directory Structure

2) Two-Level Directory

- a single-level directory often leads to confusion of file names among different users. The standard solution is to create a separate directory for each user.
- In the two-level directory structure, each user has his own user fil directory (UFD). The UFDs have similar structures, but each lists only the files of a single user.
- When a user job starts or a user logs in, the system’s master file directory (MFD)is searched.TheMFD is indexed by user name or account number, and each entry points to the UFD for that user

Directory Structure

- When a user refers to a particular file, only his own UFD is searched. Thus, different users may have files with the same name, as long as all the filenames within each UFD are unique.

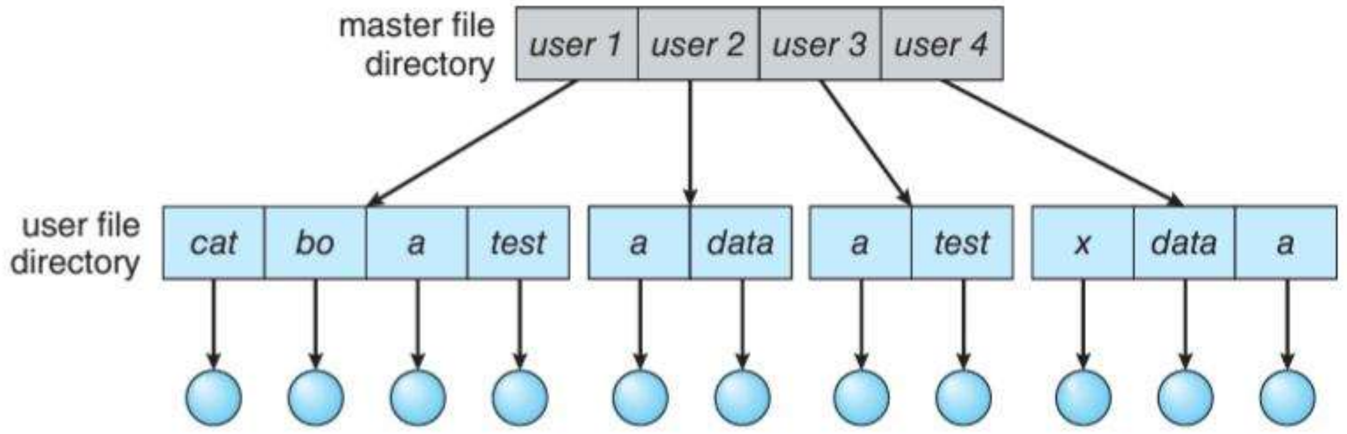


Figure 13.8 Two-level directory structure.

Directory Structure

3 Tree-Structured Directories

A tree is the most common directory structure. The tree has a root directory, and every file in the system has a unique path name.

A directory (or subdirectory) contains a set of files or subdirectories. In many implementations, a directory is simply another file, but it is treated in a special way. All directories have the same internal format.

One bit in each directory entry defines the entry as a file (0) or as a subdirectory (1).

In normal use, each process has a current directory. The current directory should contain most of the files that are of current interesttothe process

Directory Structure

- Path names can be of two types: absolute and relative. In UNIX and Linux, an absolute path name begins at the root (which is designated by an initial “/”) and follows a path down to the specified file, giving the directory names on the path . A relative pathname defines a path from the current directory.

Directory Structure

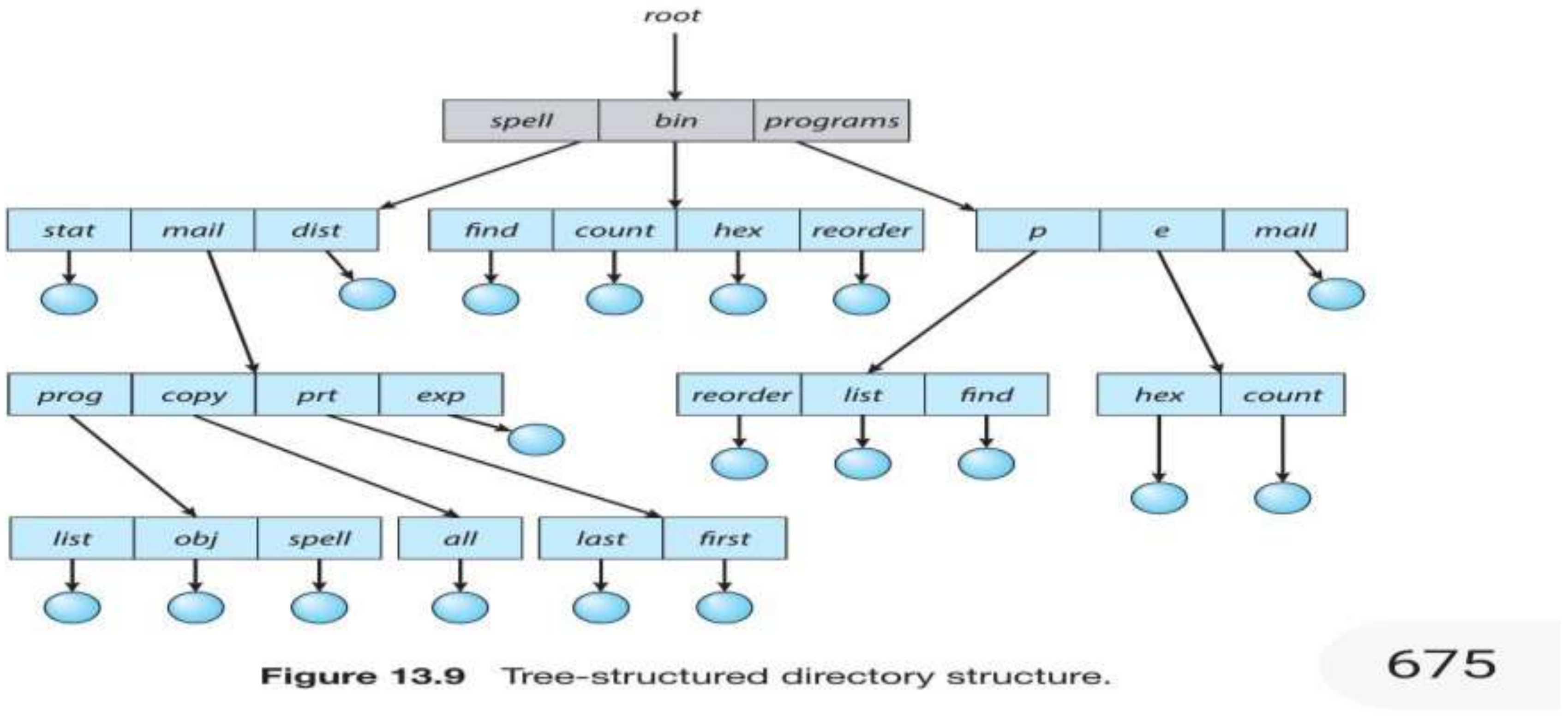


Figure 13.9 Tree-structured directory structure.

Directory Structure

- 4 Acyclic-Graph Directories
- Consider two programmers who are working on a joint project. The files associated with that project can be stored in a subdirectory, separating them from other projects and files of the two programmers.
- But since both programmers are equally responsible for the project, both want the subdirectory to be in their own directories.In this situation,the common subdirectory should be shared.
- A shared directory or file exists in the file system in two (or more) places at once. A tree structure prohibits the sharing of files or directories. An acyclic graph—that is, a graph with no cycles—allows directories to share subdirectories and files.
- The same file or subdirectory may be in two different directories. The acyclic graph is a natural generalization of the tree structured directory scheme. It is important to note that a shared file(or directory)is not the same as two copies of the file

Directory Structure

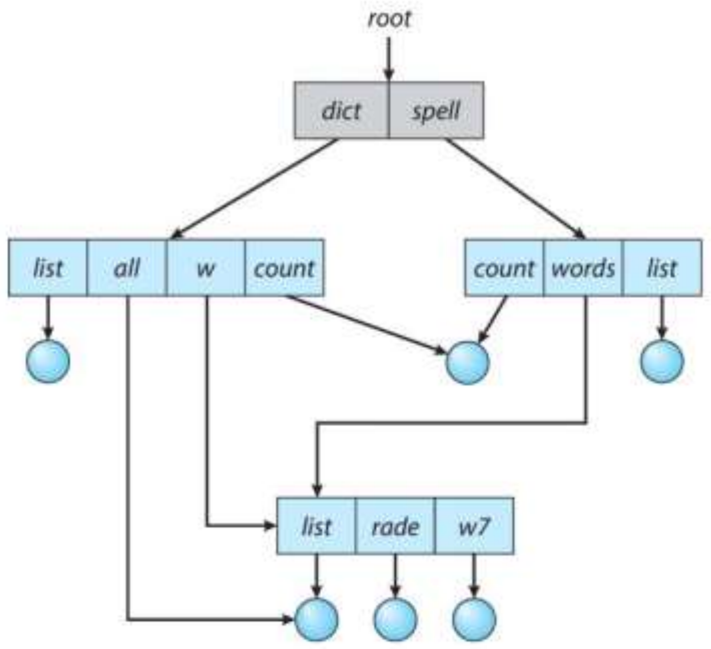


Figure 13.10 Acyclic-graph directory structure.

Directory Structure

- 5 General Graph Directory
- General-graph directory can have cycles
- This type of structure, the users are free to create directories under the root directory along with creating sub-directories under the same structure

Directory Structure

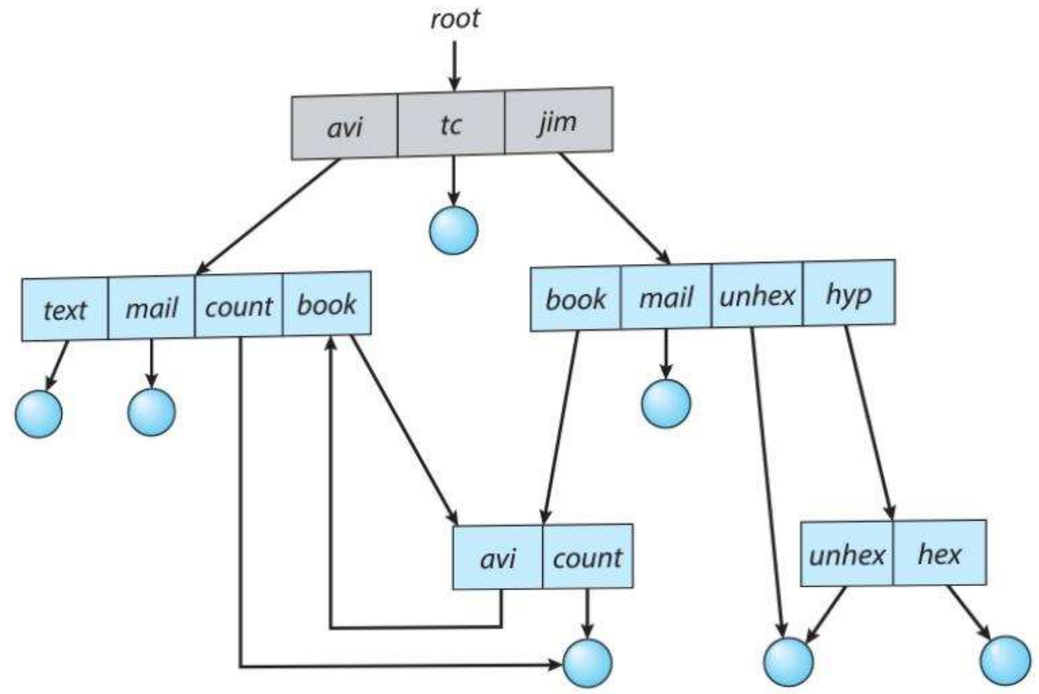


Figure 13.11 General graph directory.

Access Control

- The need to protect files is a direct result of the ability to access files. Systems that do not permit access to the files of other users do not need protection.
- The most common approach to the protection problem is to make access dependent on the identity of the user.
- Different users may need different types of access to a file or directory. The most general scheme to implement identity dependent access is to associate with each file and directory an **access-control list (ACL)** specifying user names and the types of access allowed for each user.
- When a user requests access to a particular file, the operating system checks the access list associated with that file. If that user is listed for the requested access, the access is allowed. Otherwise, a protection violation occurs, and the user job is denied access to the file.

Access Control

- Main problem with access lists is their length. If we want to allow everyone to read a file, we must list all users with read access.
- To condense the length of the access-control list, many systems recognize three classifications of users in connection with each file:
- • Owner. The user who created the file is the owner.
- • Group. A set of users who are sharing the file and need similar access is a group, or work group.
- • Other. All other users in the system.

File-System Structure

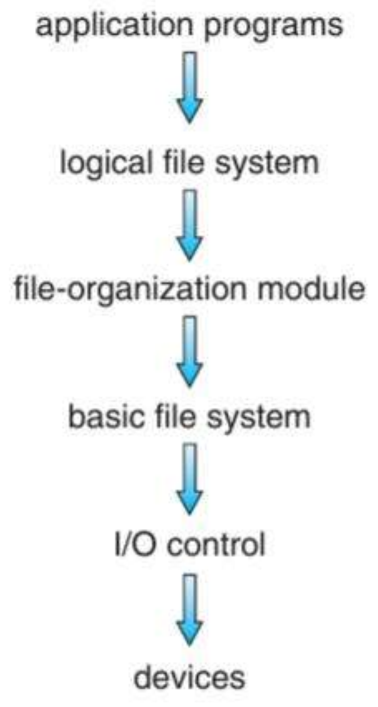


Figure 14.1 Layered file system.

File-System Structure

- Disks provide most of the secondary storage on which file systems are maintained
- To improve I/Oefficiency,I/Otransfers between memory and massstorage are performed in units of blocks. Each block on a hard disk drive has one or more sectors.
- File systems provide efficient and convenient access to the storage device by allowing data to be stored, located, and retrieved easily.

File-System Structure

- The file system itself is generally composed of many different levels. Each level in the design uses the features of lower levels to create new features for use by higher levels.
- The I/O control level consists of device drivers and interrupt handlers to transfer information between the main memory and the disk system.
- The basic file system (called the “block I/O subsystem” in Linux) needs only to issue generic commands to the appropriate device driver to read and write blocks on the storage device. This layer also manages the memory buffers and caches that hold various filesystem, directory, and data blocks.

File-System Structure

- File-organization module knows about files and their logical blocks. Each file’s logical blocks are numbered from 0 (or 1) through N. The file organization module also includes the free-space manager, which tracks unallocated blocks and provides these blocks to the file-organization module when requested.
- logical file system manages metadata information. Metadata includes all of the file-system structure except the actual data
- The logical file system manages the directory structure to provide the file-organization module with the information the latter needs, given a symbolic file name. It maintains file structure via file-control blocks. A file control block (FCB)(an inode in UNIX file systems) contains information about the file ,including ownership, permissions, and location of the file contents.

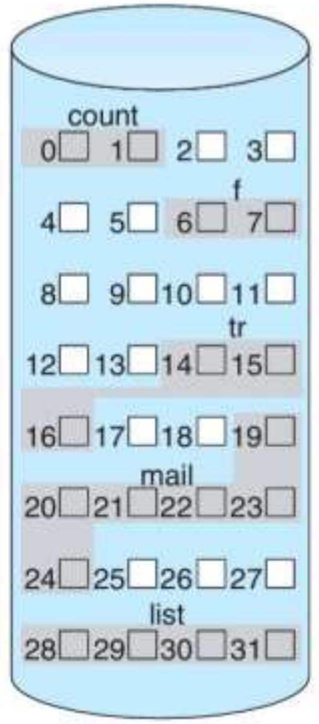
File-System Structure

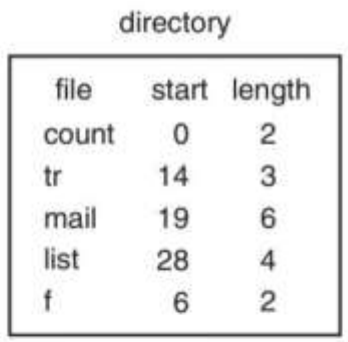
- UNIX uses the UNIX file system(UFS), which is based on the Berkeley Fast File System (FFS).
- Windows supports disk file-system formats of FAT, FAT32,and NTFS(or WindowsNTFileSystem),aswellas CD-ROMandDVD file-system formats

Allocation Methods

- how to allocate space to these files so that storage space is utilized effectively and files can be accessed quickly. Three major methods of allocating secondary storage space are in wide use: contiguous, linked, and indexed.
- 1.Contiguous Allocation
- Contiguous allocation requires that each file occupy a set of contiguous blocks on the device. Device addresses define a linear ordering on the device. With this ordering, assuming that only one job is accessing the device, accessing block b +1 after block b normally requires no head movement. When head movement is needed (from the last sector of one cylinder to the first sector of the next cylinder), the head need only move from one track to the next

Allocation Methods




directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Figure 14.4 Contiguous allocation of disk space.

Allocation Methods

- Contiguous allocation of a file is defined by the address of the first block and length (in block units) of the file. If the file is n blocks long and starts at location b, then it occupies blocks b, b +1,b + 2, ..., b + n −1. The directory entry for each file indicates the address of the starting block and the length of the area allocated for this file.
- Contiguous allocation has some problems, however. One difficulty is finding space for a new file. Next is external fragmentation.

Allocation Methods

- **2 Linked Allocation**
- Linked allocation solves all problems of contiguous allocation. With linked allocation, each file is a linked list of storage blocks; the blocks maybe scattered anywhere on the device.
- The directory contains a pointer to the first and last blocks of the file. For example, a file of five blocks might start at block 9 and continue at block 16, then block 1, then block 10, and finally block 25 .

Allocation Methods

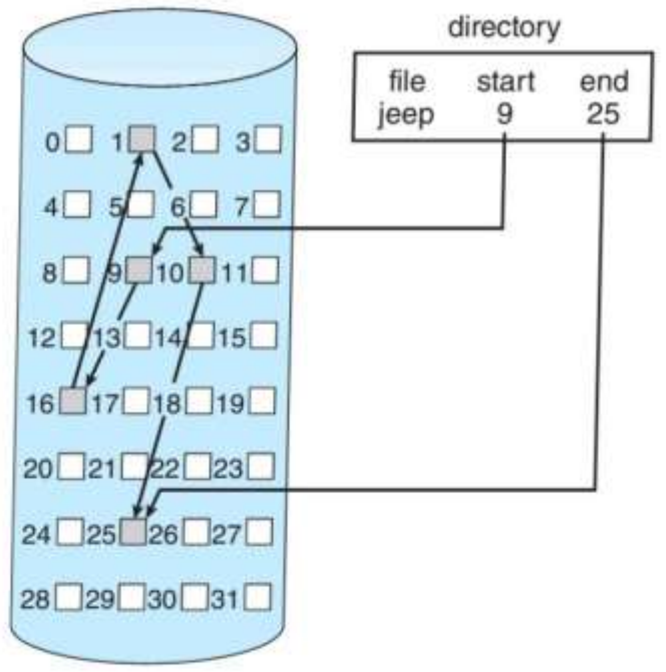


Figure 14.5 Linked allocation of disk space.

Allocation Methods

- The major problem is that it can be used effectively only for sequential-access files. To find the ith block of a file,we must start at the beginning of that file and follow the pointers until we get to the ith block.
- Another disadvantage is the space required for the pointers.
- Theusual solution to this problem is to collect blocks into multiples, called clusters, and toallocateclustersrather thanblocks. Forinstance, thefile system may define a cluster as four blocks and operate on the secondary storage device only in cluster units. Pointers then use a much smaller percentage of the file’s space. T

Allocation Methods

- 3 **Indexed Allocation**
- linked allocation cannot support efficient direct access, since the pointers to the blocks are scattered with the blocks themselves all over the disk and must be retrieved in order. Indexed allocation solves this problem by bringing all the pointers together into one location: the index block.
- Each file has its own index block, which is an array of storage-block addresses. The ith entry in the index block points to the ith block of the file. Thedirectory contains the address of the index block . To find and read the ith block, we use the pointer in the ith index-block entry

Allocation Methods

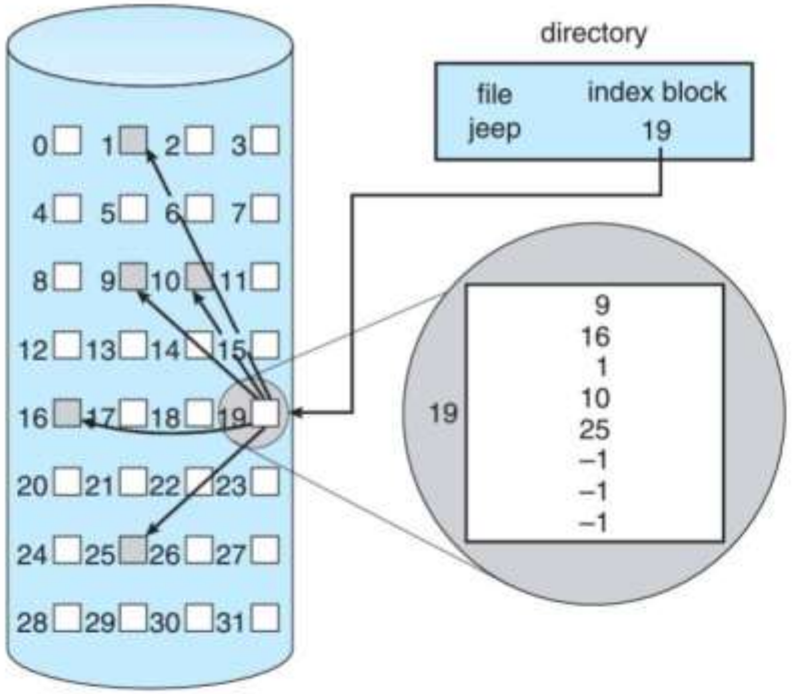


Figure 14.7 Indexed allocation of disk space.

Allocation Methods

- Indexed allocation supports direct access, without suffering from external fragmentation, because any free block on the storage device can satisfy a request for more space. Indexed allocation does suffer from wasted space, however. The pointer overhead of the index block is generally greater than the pointer overhead of linked allocation

Free space Management

- Free space management is a crucial function of operating systems, as it ensures that storage devices are utilized efficiently and effectively.
-
- The system keeps tracks of the free disk blocks for allocating space to files when they are created. Also, to reuse the space released from deleting the files, free space management becomes crucial. The system maintains a free space list which keeps track of the disk blocks that are not allocated to some file or directory.

Free space Management

1) Bitmap or Bit vector

Bit Vector is series or collection of bits where each bit corresponds to a disk block.

The bit can take two values: 0 and 1: 0 indicates that the block is free and 1 indicates an allocated block.

Free space Management

- **1111000111111001**

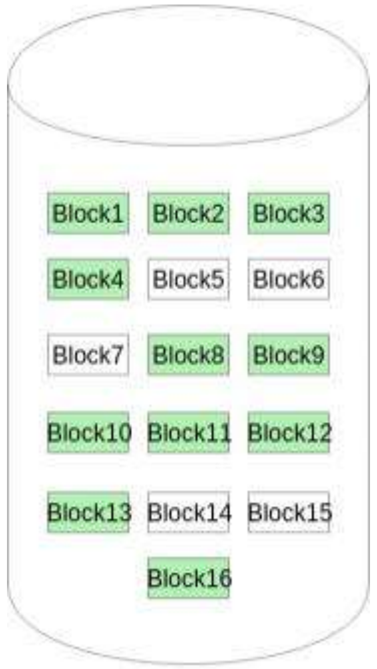


Figure - 1

Free space Management

- Advantages
- Simple to understand.
- Finding the first free block is efficient. It requires scanning the words (a group of 8 bits) in a bitmap for a non-zero word.
- Disadvantages
- For finding a free block, Operating System needs to iterate all the blocks which is time consuming.
- The efficiency of this method reduces as the disk size increases

Free space Management

- **2)Linked List**
- The free disk blocks are linked together i.e. A free block contains a pointer to the next free block. The block number of the very first disk block is stored at a separate location on disk and is also cached in memory.
- **Advantages**:
- The total available space is used efficiently using this method.
- Dynamic allocation in Linked List is easy, thus can add the space as per the requirement dynamically.
- **Disadvantages**
- Overhead of pointers

Free space Management

- 3) Grouping
- Amodification of the free-list approach stores the addresses of n free blocks in the first free block. The first n−1 of these blocks are actually free. The last block contains the addresses of another n free blocks, and so on. The addresses of a large number of free blocks can now be found quickly,

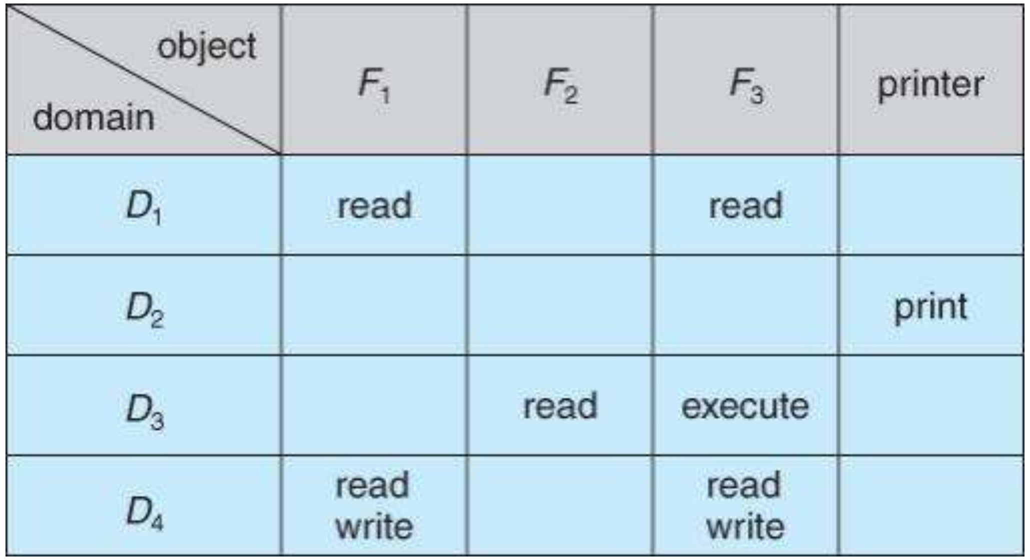
Free space Management

- Counting
- rather than keeping a list of n free block addresses, we can keep the address of the first free block and the number (n)offreecontiguousblocksthat follow the first block.

Access matrix

- Access Matrix is a digital model utilized to control and manage permissions. This model defines the rights each user has for different resources.
- The rows of the matrix represent domains, whereas the columns represent objects. Every matrix cell reflects a set of access rights granted to domain processes, i.e., each entry **(i, j)** describes the set of operations that a domain **Di** process may invoke on object **Oj.**

Access matrix

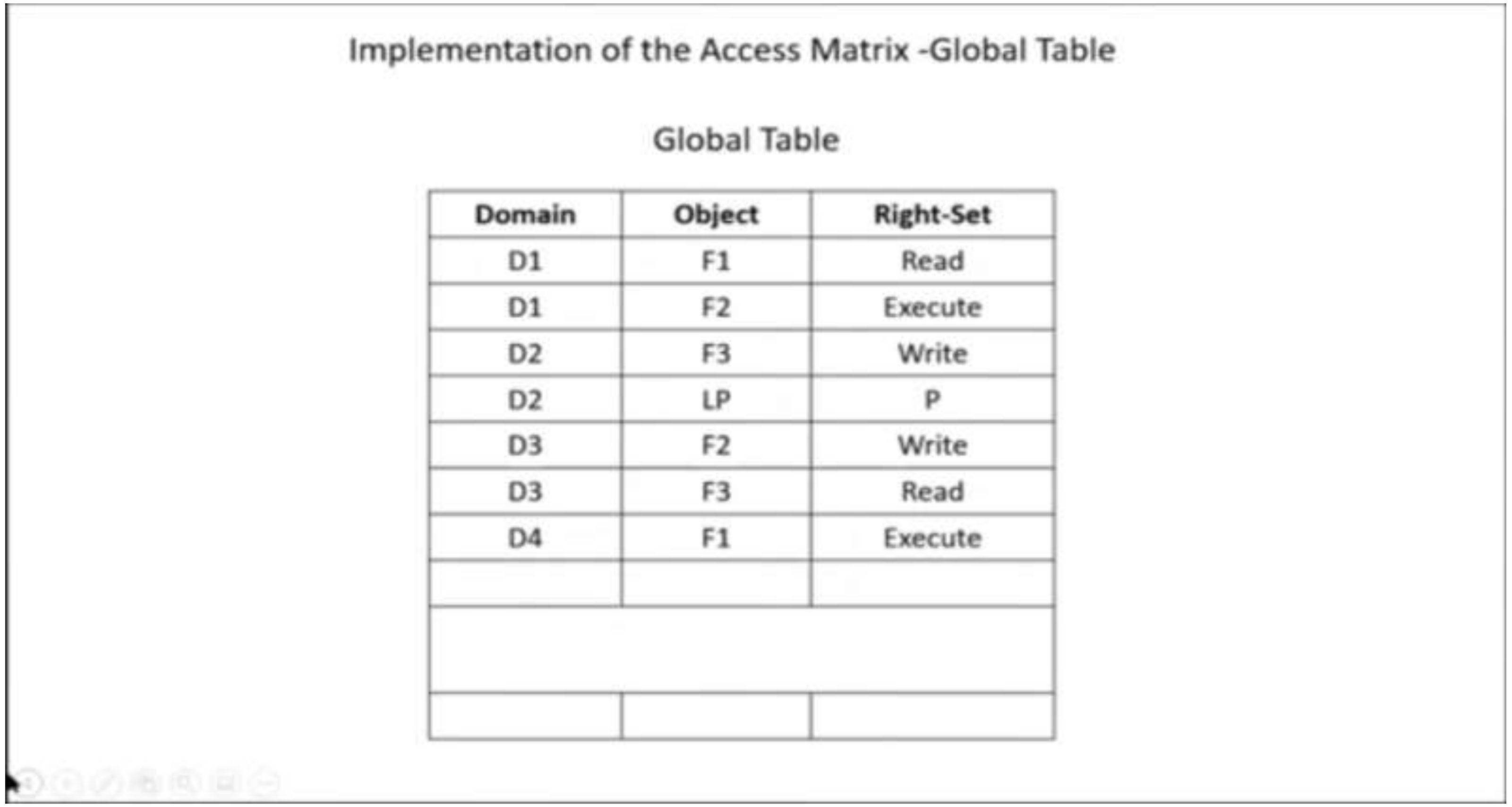


object / domain	F_1	F_2	F_3	printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

Implementation of Access matrix

- There are various methods of implementing the access matrix in the operating system.
- 1) Global Table
- the most basic access matrix implementation. A set of ordered triples **<domain, object, rights-set>** is maintained in a file. When an operation **M** has been performed on an object Oj within domain Di, the table is searched for a triple **<Di, Oj, Rk>.** The operation can proceed if this triple is located; otherwise, an exception (or error) condition has arrived.

Implementation of Access matrix



Implementation of the Access Matrix -Global Table

Global Table

Domain	Object	Right-Set
D1	F1	Read
D1	F2	Execute
D2	F3	Write
D2	LP	P
D3	F2	Write
D3	F3	Read
D4	F1	Execute

Implementation of Access matrix

2) Access Lists for Objects

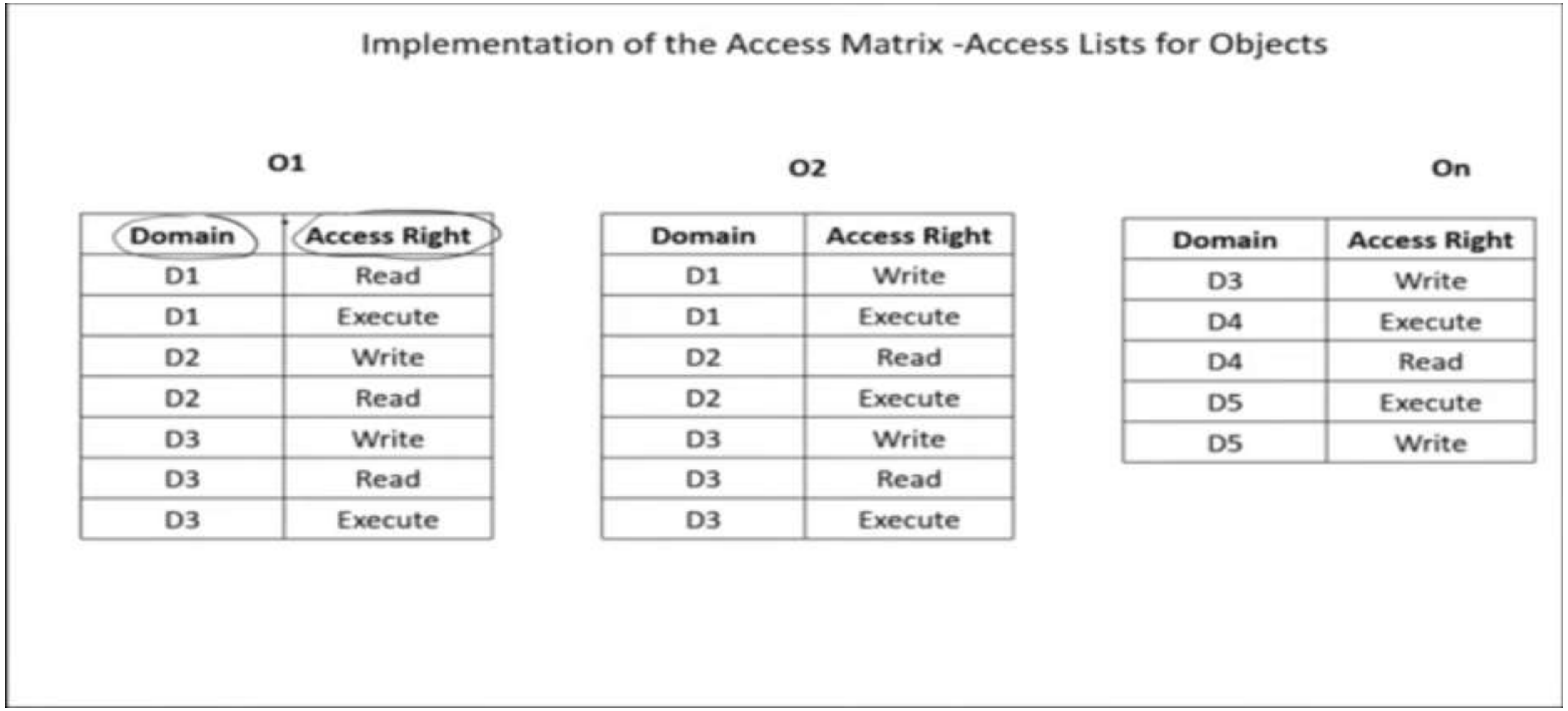
Every access matrix column may be used as a single object’s access list. It is possible to delete the blank entries. For each object, the resulting list contains ordered pairs <domain, rights-set> that define all domains for that object and a nonempty set of access rights.

3) Capability Lists for Domains

Domain’s capability list is a collection of objects and the actions that can be done on them.

If you want to perform operation M on object **Oj,** the process runs operation M, specifying the capability for object **Oj**

Implementation of Access matrix



Implementation of the Access Matrix -Access Lists for Objects

O1

Domain	Access Right
D1	Read
D1	Execute
D2	Write
D2	Read
D3	Write
D3	Read
D3	Execute

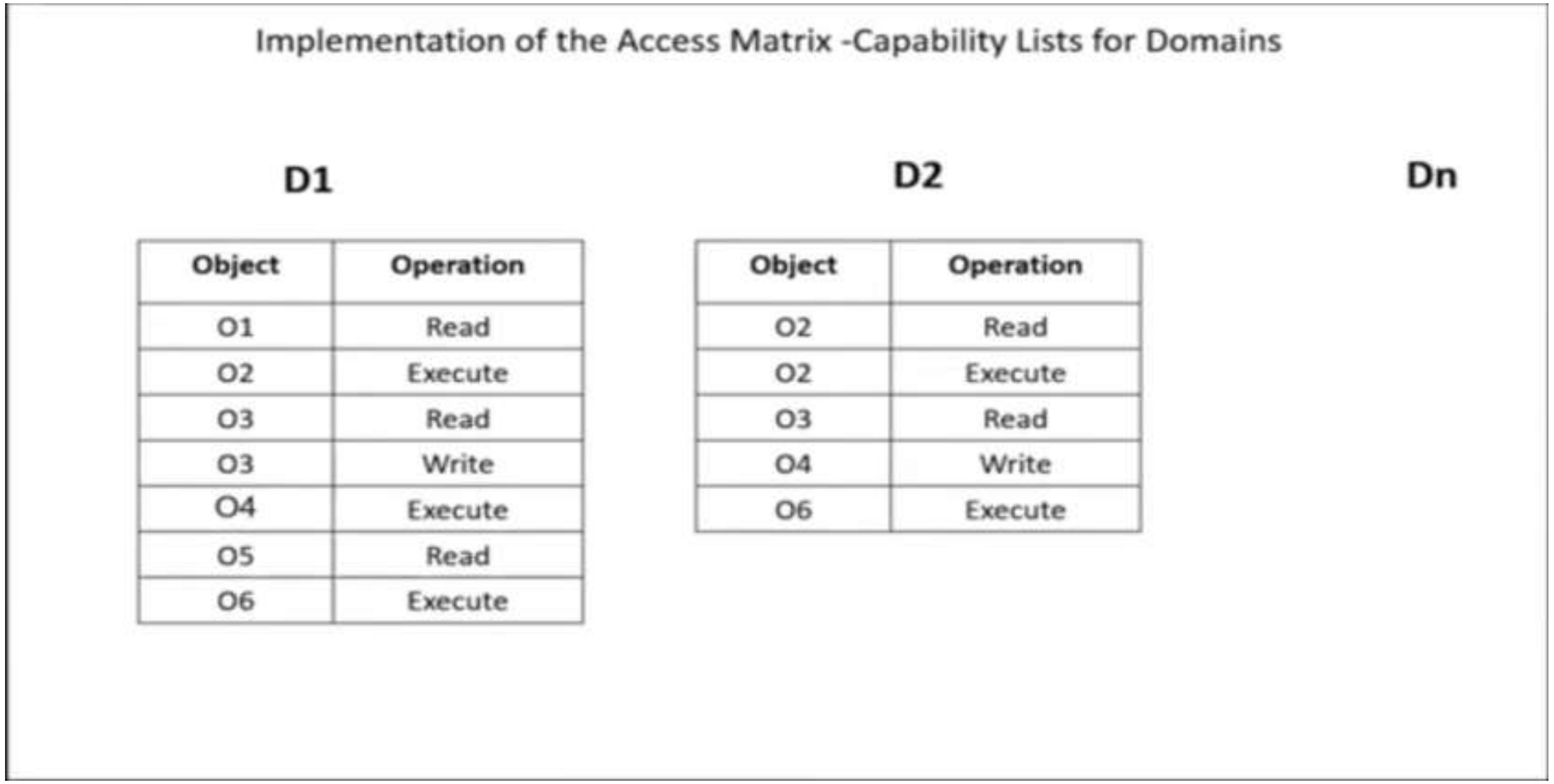
O2

Domain	Access Right
D1	Write
D1	Execute
D2	Read
D2	Execute
D3	Write
D3	Read
D3	Execute

On

Domain	Access Right
D3	Write
D4	Execute
D4	Read
D5	Execute
D5	Write

Implementation of Access matrix



Implementation of the Access Matrix -Capability Lists for Domains

D1

Object	Operation
O1	Read
O2	Execute
O3	Read
O3	Write
O4	Execute
O5	Read
O6	Execute

D2

Object	Operation
O2	Read
O2	Execute
O3	Read
O4	Write
O6	Execute

Dn

Implementation of Access matrix

4) Lock-Key Mechanism

Each domain has a set of keys that are special bit patterns. A domain-based process could only access an object if a domain has a key that satisfies one of the locks on the object. The process is not allowed to modify its keys.