MINISTRY FOR DEVELOPMENT OF INFORMATION TECHNOLOGIES AND COMMUNICATIONS THE REPUBLIC OF UZBEKISTAN
TASHKENT UNIVERSITY OF INFORMATION TECHNOLOGIES

To protection
Chair manager
Irgasheva D.Ya _______
«___»_________2015y

FINAL BACHELOR WORK

Thesis: Control access methods to the generated file objects in the information and communication systems

The graduate	______________ (signature)	Makhmudov F.F. (n.l.s)
Supervisor	______________ (signature)	Karimov M.M. (n.l.s)
Reviewer	______________ (signature)	Sharipov Z.Z. (n.l.s)
Advisor for SLA	______________ (signature)	Kodirov F.M. (n.l.s)

Tashkent- 2015

CONTENT

Introduction

Every organization uses information, most are dependent on it. Information is an asset that, like other important business assets, is essential to your business and consequently needs to be suitably protected. This is especially important in the increasingly interconnected business environment, where information is now exposed to a growing number and a wider variety of threats and vulnerabilities. Causes of damage such as malicious code, computer hacking, and denial of service attacks have become more common, more ambitious, and increasingly sophisticated.

The underlying legislation in relation to protection of information in the Republic of Uzbekistan is the Law “On principles and guarantees of freedom of information”, which defines the basic concepts such as “information”, “information sphere”, “confidential information”, “information security”, “protection of information” and others. The adoption of the major legislation acts in the Republic of Uzbekistan, including the laws “On electronic digital signature”, “On electronic document flow”, “On electronic commerce” and the Resolution of the Cabinet of Ministers of the Republic of Uzbekistan №215 dated September 26, 2005 “On improving the regulatory framework in the use of electronic digital signatures” allowed to create the necessary legal conditions for the use of electronic digital signatures and development of electronic digital signatures public key infrastructure in the country.

With a large workforce, it can be difficult to control who is going in and out of the building. Even small offices need to focus on security, as they are particularly vulnerable when under threat of crime because they often cannot afford the high costs involved with unwanted visitors. An access control system will help you to manage all of this and to ensure that your premises, staff, products and data are secure.

In fact, security is the primary reason for having an access control system, although it can also be used for monitoring staff comings and goings. Many types of business simply cannot afford for unwanted and unknown visitors to enter their premises, through fear of theft or that confidential data might be compromised or abused. Of course, access control isn't just about stopping outsiders from coming in, but also about stopping workers from going where they shouldn't or don't need to be. The right kind of system can provide different levels of access to different people if you need it that way.

Every business is different, and even within a single business there can be many different requirements as regards access control. The best kind of system will be one that is flexible and that can work with you to give you everything you need. Whether you have an apartment building, run a school, hospital, bank or office, you should choose the system that matches your needs ideally.

Controlling access into and around your building is not only good for security, but also for health and safety reasons, with users being granted or denied access to certain zones at different times.

Access Control is vitally important in business, as there is usually a lot of foot traffic. With this in mind, if anything were to be stolen then it would be difficult to pinpoint who was there, perhaps when they were not supposed to be. Most businesses like to have a way for new visitors or clients to sign in and out of their premises, not only for security reasons, but also for peace of mind.

Above everything, access control is very important for theft and safety reasons.

Purpose of this dissertation work is researching control access methods to the generated file objects in the information systems. For this, following tasks are taken:

- ✓ Researching file and file access methods;
- ✓ Analyzing of basics of Access Control, identification and authentication methods, attacks in access control;
- ✓ Researching access control models and techniques;
- ✓ Design program for file access control.

This final diploma work consist of introduction, three part, conclusion and references.

Importance of access control is described in *introduction* part.

In the theoretical part, file systems, file attributes, file access control, file rights are researched.

In the main part, basics of access control, identification and authentication methods, file and data ownership, related methods of attacks, access control models and techniques are analyzing and “Control access of generated file objects” program is designed.

1. The theoretical part. File Access Methods

1.1. File System

In computing, a file system (or file system) is used to control how data is stored and retrieved. Without a file system, information placed in a storage area would be one large body of data with no way to tell where one piece of information stops and the next begins. By separating the data into individual pieces, and giving each piece a name, the information is easily separated and identified. Taking its name from the way paper-based information systems are named, each group of data is called a “file”. The structure and logic rules used to manage the groups of information and their names is called a “file system” [6].

There are many different kinds of file systems. Each one has different structure and logic, properties of speed, flexibility, security, size and more. Some file systems have been designed to be used for specific applications. For example, the ISO 9660 file system is designed specifically for optical discs.

File systems can be used on many different kinds of storage devices. Each storage device uses a different kind of media. The most common storage device in use today is a hard drive whose media is a disc that has been coated with a magnetic film. The film has ones and zeros 'written' on it sending electrical pulses to a magnetic "read-write" head. Other media that are used are magnetic tape, optical disc, and flash memory. In some cases, the computer's main memory (RAM) is used to create a temporary file system for short term use.

Some file systems are used on local data storage devices; others provide file access via a network protocol (for example, NFS, SMB, or 9P clients). Some file systems are "virtual", in that the "files" supplied are computed on request (e.g. procfs) or are merely a mapping into a different file system used as a backing store. The file system manages access to both the content of files and the metadata about those files. It is responsible for arranging storage space; reliability, efficiency, and tuning with regard to the physical storage medium are important design considerations.

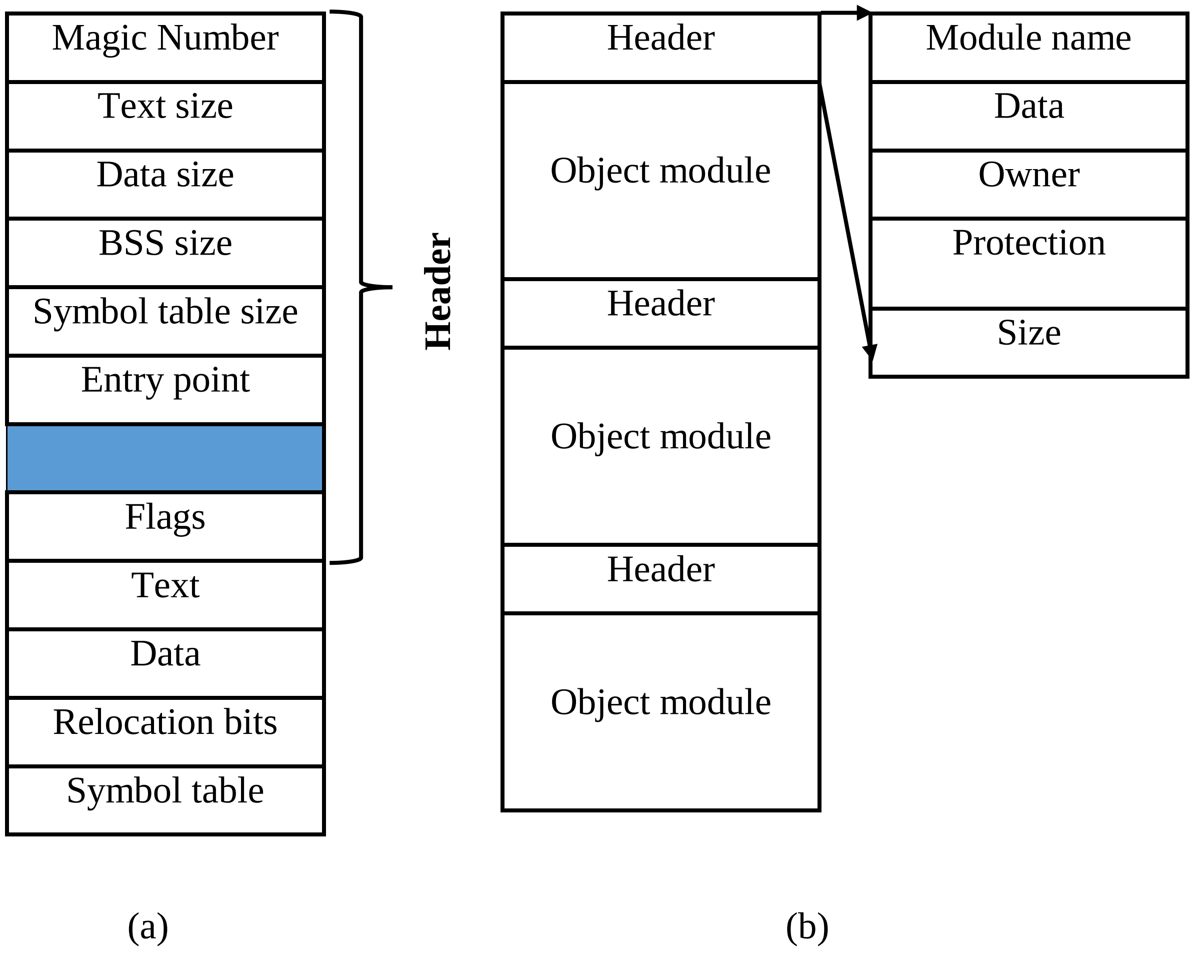


Figure 1.1. File types: (a) an executable file, (b) an archive file

Space management. File systems allocate space in a granular manner, usually multiple physical units on the device. The file system is responsible for organizing files and directories, and keeping track of which areas of the media belong to which file and which are not being used. For example, in Apple DOS of the early 1980s, 256-byte sectors on 140-kilobyte floppy disk used a track/sector map.

This results in unused space when a file is not an exact multiple of the allocation unit, sometimes referred to as slack space. For a 512-byte allocation, the average unused space is 256 bytes. For 64 KB clusters, the average unused space is 32 KB. The size of the allocation unit is chosen when the file system is created.

Choosing the allocation size based on the average size of the files expected to be in the file system can minimize the amount of unusable space. Frequently the default allocation may provide reasonable usage. Choosing an allocation size that is too small results in excessive overhead if the file system will contain mostly very large files [6].

File system fragmentation occurs when unused space or single files are not contiguous. As a file system is used, files are created, modified and deleted. When a file is created the file, system allocates space for the data. Some file systems permit or require specifying an initial space allocation and subsequent incremental allocations as the file grows. As files are deleted, the space they were allocated eventually is considered available for use by other files. This creates alternating used and unused areas of various sizes. This is free space fragmentation. When a file is created and there is not an area of contiguous space available for its initial allocation the space must be assigned in fragments. When a file is modified such that it becomes larger, it may exceed the space initially allocated to it, another allocation must be assigned elsewhere and the file becomes fragmented.

Filenames. A filename (or file name) is used to identify a storage location in the file system. Most file systems have restrictions on the length of filenames. In some file systems, filenames are not case sensitive (i.e., filenames such as FOO and foo refer to the same file); in others, filenames are case sensitive (i.e., the names FOO, Foo and foo refer to three separate files).

Most modern file systems allow filenames to contain a wide range of characters from the Unicode character set. However, they may have restrictions on the use of certain special characters, disallowing them within filenames; those characters might be used to indicate a device, device type, and directory prefix, file path separator, or file type.

Directories. File systems typically have directories (also called folders) which allow the user to group files into separate collections. This may be implemented by associating the file name with an index in a table of contents or an inode in a Unix-like file system. Directory structures may be flat (i.e. linear), or allow hierarchies where directories may contain subdirectories. The first file system to support arbitrary hierarchies of directories was used in the Multics operating system. The native file systems of Unix-like systems also support arbitrary directory hierarchies, as do, for example, Apple's Hierarchical File System, and its successor HFS+ in classic Mac OS (HFS+ is still used in Mac OS X), the FAT file system in MS-DOS 2.0 and later and Microsoft Windows, the NTFS file system in the Windows NT family of operating systems, and the ODS-2 (On-Disk Structure-2) and higher levels of the Files-11 file system in OpenVMS.

Metadata. Other bookkeeping information is typically associated with each file within a file system. The length of the data contained in a file may be stored as the number of blocks allocated for the file or as a byte count. The time that the file was last modified may be stored as the file's timestamp. File systems might store the file creation time, the time it was last accessed, the time the file's metadata was changed, or the time the file was last backed up. Other information can include the file's device type (e.g. block, character, socket, subdirectory, etc.), its owner user ID and group ID, its access permissions and other file attributes (e.g., whether the file is read-only, executable, etc.).

A file system stores all the metadata associated with the file—including the file name, the length of the contents of a file, and the location of the file in the folder hierarchy—separate from the contents of the file.

Most file systems store the names of all the files in one directory in one place—the directory table for that directory—which is often stored like any other file. Many file systems put only some of the metadata for a file in the directory table, and the rest of the metadata for that file in a completely separate structure, such as the inode.

Restricting and permitting access. There are several mechanisms used by file systems to control access to data. Usually the intent is to prevent reading or modifying files by a user or group of users. Another reason is to ensure data is modified in a controlled way so access may be restricted to a specific program. Examples include passwords stored in the metadata of the file or elsewhere and file permissions in the form of permission bits, access control lists, or capabilities. The need for file system utilities to be able to access the data at the media level to reorganize the structures and provide efficient backup usually means that these are only effective for polite users but are not effective against intruders.

Methods for encrypting file data are sometimes included in the file system. This is very effective since there is no need for file system utilities to know the encryption seed to effectively manage the data. The risks of relying on encryption include the fact that an attacker can copy the data and use brute force to decrypt the data. Losing the seed means losing the data.

Maintaining integrity. One significant responsibility of a file system is to ensure that, regardless of the actions by programs accessing the data, the structure remains consistent. This includes actions taken if a program modifying data terminates abnormally or neglects to inform the file system that it has completed its activities. This may include updating the metadata, the directory entry and handling any data that was buffered but not updated on the physical storage media.

Other failures, which the file system must deal with, include media failures or loss of connection to remote systems.

In the event of an operating system failure or "soft" power failure, special routines in the file system must be invoked similar to when an individual program fails.

The file system must also be able to correct damaged structures. These may occur as a result of an operating system failure for which the OS was unable to notify the file system, power failure or reset.

The file system must also record events to allow analysis of systemic issues as well as problems with specific files or directories.

User data. The most important purpose of a file system is to manage user data. This includes storing, retrieving and updating data.

Some file systems accept data for storage as a stream of bytes, which are collected and stored in a manner efficient for the media. When a program retrieves the data, it specifies the size of a memory buffer and the file system transfers data from the media to the buffer. Sometimes a runtime library routine may allow the user program to define a record based on a library call specifying a length. When the user program reads the data, the library retrieves data via the file system and returns a record.

Some file systems allow the specification of a fixed record length, which is used for all write and reads. This facilitates updating records.

An identification for each record, also known as a key, makes for a more sophisticated file system. The user program can read, write and update records without regard to their location. This requires complicated management of blocks of media usually separating key blocks and data blocks. Very efficient algorithms can be developed with pyramid structure for locating records.

Types of file systems. File system types can be classified into disk file systems, network file systems and special-purpose file systems [6].

Disk file systems. A disk file system takes advantages of the ability of disk storage media to randomly address data in a short amount of time. Additional considerations include the speed of accessing data following that initially requested and the anticipation that the following data may also be requested. This permits multiple users (or processes) access to various data on the disk without regard to the sequential location of the data. Examples include FAT (FAT12, FAT16, FAT32), exFAT, NTFS, HFS and HFS+, HPFS, UFS, ext2, ext3, ext4, XFS, btrfs, ISO 9660, Files-11, Veritas File System, VMFS, ZFS, ReiserFS and UDF. Some disk file systems are journaling file systems or versioning file systems.

Network file systems. A network file system is a file system that acts as a client for a remote file access protocol, providing access to files on a server. Examples of network file systems include clients for the NFS, AFS, SMB protocols, and file-system-like clients for FTP and WebDAV.

A shared disk file system is one in which a number of machines (usually servers) all have access to the same external disk subsystem (usually a SAN). The file system arbitrates access to that subsystem, preventing write collisions. Examples include GFS2 from Red Hat, GPFS from IBM, SFS from DataPlow, CXFS from SGI and StorNext from Quantum Corporation.

Special file systems. A special file system presents non-file elements of an operating system as files so they can be acted on using file system APIs. This is most commonly done in Unix-like operating systems, but devices are given file names in some non-Unix-like operating systems as well.

A device file system represents I/O devices and pseudo-devices as files, called device files. Examples in Unix-like systems include devfs and, in Linux 2.6 systems, udev. In non-Unix-like systems, such as TOPS-10 and other operating systems influenced by it, where the full filename or pathname of a file can include a device prefix, devices other than those containing file systems are referred to by a device prefix specifying the device, without anything following it.

File operations. The OS provides systems calls to create, write, read, reset, and delete files. The following discusses the specific duties an OS must do for each of the five basic file operations.

Creating a file. First, a space in the file system must be found for the file. Second, an entry for the new file must be made in the directory. The directory entry records the name of the file and the location in the file system.

Writing a file. To write a file, a system call is made specifying both the name and the file and the information to be written to the file. Given the name of the file, the system searches the directory to find the location of the file. The directory entry will need to store a pointer to the current block of the file (usually the beginning of the file). Using this pointer, the address of the next block can be computed where the information will be written. The write pointer must be updated - in this way, successive writes can be used to write a sequence of blocks to the file.

Reading a file. To read a file, a system call is made that specifies that specifies the name of the file and where (in memory) the next block of the file should be put. Again, the directory is searched for the associated directory entry, and the directory will need a pointer to the next block to be read. Once the block is read, the pointer is updated.

Resetting a file. The directory is searched for the appropriate entry, and the current file position is reset to the beginning of the file.

Deleting a file. To delete a file, the directory is searched for the named file. Having found the associated directory entry, the space allocated to the file is released (so it can be reused by other files) and invalidates the directory entry.

The five operations described comprise only the minimal set of required file operations. More commonly, we shall also want to edit the file and modify its contents. A special case of editing a file is appending new information at the end of the file. Copies of the file can also be created, and since files are named object, renaming an existing file may also be needed. If the file is a binary object format, we may also want to execute it. Also of use are facilities to lock sections of an open file for multiprocess access, to share sections, and even to map sections into memory or virtual-memory systems. This last function allows a part of the virtual address to be logically associated with section of a file. Reads and writes to that memory region are then treated as reads and writes to the file. To that memory region are then treated as reads and writes to the file, greatly simplifying file use.

File allocation methods. One main problem in file management is how to allocate space for files so that disk space is utilized effectively and files can be accessed quickly. Three major methods of allocating disk space are contiguous, linked, and indexed. Each method has its advantages and disadvantages. Accordingly, some systems support all three (e.g. Data General's RDOS). More commonly, a system will use one particular method for all files [6].

Contiguous Allocation. The contiguous allocation method requires each file to occupy a set of contiguous address on the disk. Disk addresses define a linear ordering on the disk. Notice that, with this ordering, 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), it is only one track. Thus, the number of disk seeks required for accessing contiguous allocated files in minimal, as is seek time when a seek is finally needed. Contiguous allocation of a file is defined by the disk address and the length of the first block. 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.

The difficulty with contiguous allocation is finding space for a new file. If the file to be created is n blocks long, then the OS must search for n free contiguous blocks. First-fit, best-fit, and worst-fit strategies (as discussed in Chapter 4 on multiple partition allocation) are the most common strategies used to select a free hole from the set of available holes. Simulations have shown that both first-fit and best-fit are better than worst-fit in terms of both time storage utilization. Neither first-fit nor best-fit is clearly best in terms of storage utilization, but first-fit is generally faster.

These algorithms also suffer from external fragmentation. As files are allocated and deleted, the free disk space is broken into little pieces. External fragmentation exists when enough total disk space exists to satisfy a request, but this space not contiguous; storage is fragmented into a large number of small holes.

Another problem with contiguous allocation is determining how much disk space is needed for a file. When the file is created, the total amount of space it will need must be known and allocated. How does the creator (program or person) know the size of the file to be created. In some cases, this determination may be fairly simple (e.g. copying an existing file), but in general the size of an output file may be difficult to estimate.

Linked Allocation. The problems in contiguous allocation can be traced directly to the requirement that the spaces be allocated contiguously and that the files that need these spaces are of different sizes. These requirements can be avoided by using linked allocation.

In linked allocation, each file is a linked list of disk blocks. The directory contains a pointer to the first and (optionally the last) block of the file. For example, a file of 5 blocks which starts at block 4, might continue at block 7, then block 16, block 10, and finally block 27. Each block contains a pointer to the next block and the last block contains a NIL pointer. The value -1 may be used for NIL to differentiate it from block 0.

With linked allocation, each directory entry has a pointer to the first disk block of the file. This pointer is initialized to nil (the end-of-list pointer value) to signify an empty file. A write to a file removes the first free block and writes to that block. This new block is then linked to the end of the file. To read a file, the pointers are just followed from block to block.

There is no external fragmentation with linked allocation. Any free block can be used to satisfy a request. Notice also that there is no need to declare the size of a file when that file is created. A file can continue to grow as long as there are free blocks.

Linked allocation, does have disadvantages, however. The major problem is that it is inefficient to support direct-access; it is effective only for sequential-access files. To find the i^{th} block of a file, it must start at the beginning of that file and follow the pointers until the i^{th} block is reached. Note that each access to a pointer requires a disk read.

Another severe problem is reliability. A bug in OS or disk hardware failure might result in pointers being lost and damaged. The effect of which could be picking up a wrong pointer and linking it to a free block or into another file.

Indexed Allocation. The indexed allocation method is the solution to the problem of both contiguous and linked allocation. This is done by bringing all the pointers together into one location called the index block. Of course, the index block will occupy some space and thus could be considered as an overhead of the method.

In indexed allocation, each file has its own index block, which is an array of disk sector of addresses. The i^{th} entry in the index block points to the i^{th} sector of the file. The directory contains the address of the index block of a file. To read the i^{th} sector of the file, the pointer in the i^{th} index block entry is read to find the desired sector.

Indexed allocation supports direct access, without suffering from external fragmentation. Any free block anywhere on the disk may satisfy a request for more space.

Directory structures. The structure of the directories and the relationship among them are the main areas where file systems tend to differ, and it is the area that has the most significant effect on the user interface provided by the file system. The most common directory structures used by multi-user systems are [6]:

- ✓ single-level directory;
- ✓ two-level directory;
- ✓ tree-structured directory;
- ✓ acyclic directory.

Single-Level Directory. In a single-level directory system, all the files are placed in one directory. This is very common on single-user OS's.

A single-level directory has significant limitations, however, when the number of files increases or when there is more than one user. Since all files are in the same directory, they must have unique names. If there are two users who call their data file "test", then the unique-name rule is violated. Although file names are generally selected to reflect the content of the file, they are often quite limited in length.

Even with a single-user, as the number of files increases, it becomes difficult to remember the names of all the files in order to create only files with unique names.

Two-Level Directory. In the two-level directory system, the system maintains a master block that has one entry for each user. This master block contains the addresses of the directory of the users.

There are still problems with two-level directory structure. This structure effectively isolates one user from another. This is an advantage when the users are completely independent, but a disadvantage when the users want to cooperate on some task and access files of other users. Some systems simply do not allow local files to be accessed by other users.

Tree-Structured Directories. In the tree-structured directory, the directory themselves are files. This leads to the possibility of having sub-directories that can contain files and sub-subdirectories.

An interesting policy decision in a tree-structured directory structure is how to handle the deletion of a directory. If a directory is empty, its entry in its containing directory can simply be deleted. However, suppose the directory to be deleted id not empty, but contains several files, or possibly sub-directories. Some systems will not delete a directory unless it is empty. Thus, to delete a directory, someone must first delete all the files in that directory. If these are any sub-directories, this procedure must be applied recursively to them, so that they can be deleted also. This approach may result in a insubstantial amount of work. An alternative approach is just to assume that, when a request is made to delete a directory, all of that directory's files and sub-directories are also to be deleted.

Acyclic-Graph Directories. The acyclic directory structure is an extension of the tree-structured directory structure. In the tree-structured directory, files and directories starting from some fixed directory are owned by one particular user. In the acyclic structure, this prohibition is taken out and thus a directory or file under directory can be owned by several users.

File system protection. When designing file systems, one of the main considerations is how to protect sharable files. Sometimes this is avoided by not permitting sharing of files and hence, protection is quite easy to implement, or make all files public and provide complete sharing. However, neither of this is acceptable to users. There are files that users want to share and there are some which only a group of users should be allowed to access. The solution to this is to allow limited sharing, i.e., users can share but to a limited extent only.

Protection is achieved by limiting the type of file access, which can be made. Access is permitted or denied depending upon several factors, one of which is the type of access requested. Several operations on files can be controlled. Some of these are:

- ✓ read - read a file;
- ✓ write - write a file;
- ✓ execute - load and execute a file;
- ✓ append - append information at the end of a file;
- ✓ Delete - free the space allocated to a file.

The most common implementation of the file systems allow the owners of the file to do operations 1-5, whereas other users can only invoke those operations that do not modify the file, e.g., file read. However, in some systems, e.g., UNIX, the user can change the access control of a file such that he can let anybody access (modification allowed) the file or he can completely deny any user (including himself) access to a file.

The three most popular implementations of file protection are the following:

File naming - this depends upon the inability of a user to access a file he cannot name. This can be implemented by allowing only users to see the files they have created. However, since most file systems allow only a limited number of characters for filenames, there is no guarantee that two users will not use the same filenames.

Passwords - this scheme associates a password to each file. If a user does not know the password associated to a file then he cannot access it. This is a very effective way of protecting files but for a user who owns many files, and constantly changes the password to make sure that nobody accesses these files will require that users have photographic memories.

Access control - an access list is associated to each file or directory. The access list contains information on the type of users and accesses that they can do on a directory or file. An example is the following access list associated to a UNIX file or directory:

drwxrwxrwx

The d indicates that this is an access list for a directory, the first rwx indicates that it can be read, written, and executed by the owner of the file, the second rwx is an access information for users belonging to the same group as the owner (somewhere on the system is a list of users belonging to same group as the owner), and the last rwx for all other users. The rwx can be changed to just r-- indicating that it can only be read, or -w- for write-only, --x for execute only.

File Attributes. File attributes are metadata associated with computer files that define file system behavior. Each attribute can have one of two states: set and cleared. Attributes are considered distinct from other metadata, such as dates and times, filename extensions or file system permissions. In addition to files, folders, volumes and other file system objects may have attributes [17].

Traditionally, in MS-DOS and Microsoft Windows, there were four attributes: archive, hidden, read-only and system. Windows has added new ones. Systems derived from 4.4BSD-Lite, such as FreeBSD, NetBSD, OpenBSD, DragonFly BSD, and OS X, have sets of “system” and “user” attributes; newer versions of the Linux kernel also support a set of file attributes.

Traditionally, in DOS and Microsoft Windows, files and folders accepted four attributes:

Archive: When set, it indicates that the hosting file has changed since the last backup operation. Windows' file system sets this attribute on any file that has changed. Backup software then has the duty of clearing it upon a successful backup.

Hidden: When set, indicates that the hosting file is hidden. MS-DOS commands like *dir* and Windows apps like File Explorer do not show hidden files by default, unless asked to do so.

System: When set, indicates that the hosting file is a critical system file that is necessary for the computer to operate properly. MS-DOS and Microsoft Windows use it to mark important system files. MS-DOS commands like dir and Windows apps like File Explorer do not show system files by default even when hidden files are shown, unless asked to do so.

Read-only: When set, indicates that a file should not be altered. Upon opening the file, file system API usually does not grant write permission to the requesting application, unless the application explicitly requests it. Read-only attributes on folders are usually ignored.

As new versions of Windows came out, Microsoft has added to the inventory of available attributes on the NTFS file system, including but not limited to:

Compressed: When set, Windows compresses the hosting file upon storage.

Encrypted: When set, Windows encrypts the hosting file upon storage to prevent unauthorized access.

Indexed: When set, Indexing Service or Windows Search do not include the hosting file in their indexing operation.

Information is kept in files. Files reside on secondary storage. When this information is to be used, it has to be accessed and brought into primary main memory. Information in files could be accessed in many ways. It is usually dependent on an application. There are three file access methods.

- ✓ Sequential Access;
- ✓ Direct Access;
- ✓ Indexed Sequential Access.

1.2. Sequential Access

Information in the file is processed in order, one record after the other. This is by far the most common mode of access of files. For example, computer editors usually access files in this fashion [15].

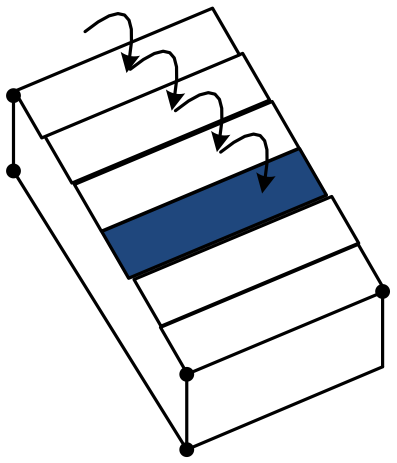


Figure 1.2. Sequential access

A read operation reads the next portion of the file and automatically advances the file pointer. Similarly, a write appends to the end of the end of the file and advances to the end of the newly written material (the new end of file). Such a file can be reset to the beginning, and, on some systems, a program may be able to skip forward or backward n records, for some integer n. This scheme is known as sequential access to a file. Sequential access is based on a tape model of a file [4].

Problems accessing ordered Sequential files. Records in an ordered Sequential file are arranged, in order, on some key field or fields. When we want to insert, delete or amend a record we must preserve the ordering. The only way to do this is to create a new file. In the case of an insertion or update, the new file will contain the inserted or updated record. In the case of a deletion, the deleted record will be missing from the new file.

The main drawback to inserting, deleting or amending records in an ordered Sequential file is that the entire file must be read and then the records written to a new file. Since disk access is one of the slowest things, we can do in computing this is very wasteful of computer time when only a few records are involved.

For instance, if 10 records are to be inserted into a 10,000 record file, then 10,000 records will have to be read from the old file and 10,010 written to the new file. The average time to insert a new record will thus be very great.

Contrast with direct access files. While Sequential files have a number of advantages over other types of file organization (and these are discussed fully in the final section) the fact that a new file must be created when we delete, update or insert a records causes problems.

These problems are addressed by direct access files. Direct access files allow us to read, update, delete and insert individual records, in situ, using a key.

Inserting records in an ordered Sequential file. To insert a record in an ordered Sequential file:

1. All the records with a key value less than the record to be inserted must be read and then written to the new file.
2. Then the record to be inserted must be written to the new file.
3. Finally, the remaining records must be written to the new file.

Deleting records from an ordered Sequential file. To delete a record in an ordered Sequential file:

1. All the records with a key value less than the record to be deleted must be written to the new file.
2. When the record to be deleted is encountered, it is not written to the new file.
3. Finally, all the remaining records must be written to the new file.

To amend a record in an ordered Sequential file:

1. All the records with a key value less than the record to be amended must be read and then written to the new file.
2. Then the record to be amended must be read the amendments applied to it and the amended record must then be written to the new file.
3. Finally, all the remaining records must be written to the new file.

The problem with Sequential files is that unless the file is ordered very few (only read and add) operations can be applied to it.

Even when a Sequential file is ordered; delete, insert and amend operations are prohibitively expensive (in processing terms) when only a few records in the file are affected (i.e. when the "hit rate" is low).

Following, code of “creating a sequential-access file in C#” is given [18]:

```
using System;
   using System.Data;
   using System.IO;
   using System.Runtime.Serialization.Formatters.Binary;
   using System.Runtime.Serialization;
   public class CreateFile {
      static void Main() {
         BinaryFormatter formatter = new BinaryFormatter();
         FileStream     output     =     new     FileStream(
"test.dat",FileMode.OpenOrCreate, FileAccess.Write );
         Record record = new Record();
         record.Account = 1234;
         record.FirstName = "FirstName";
         record.LastName = "LastName";
         record.Balance = 1234.345;
         formatter.Serialize( output, record );
         output.Close();
      }
   }

   [Serializable]
   public class Record{
      public int Account;
      public String FirstName;
      public String LastName;
      public double Balance;
   }
```

1.3. Direct Access

In computer science, random access (more precisely and more generally called direct access) is the ability to access an item of data at any given coordinates in a population of addressable elements. As a rule the assumption is that each element can be accessed roughly as easily and efficiently as any other, no matter how many elements may be in the set, nor how many coordinates may be available for addressing the data. For example, data might be stored notionally in a single sequence like a row, in two dimensions like rows and columns on a surface, or in multiple dimensions. However, given all the coordinates, a program can access each record about as quickly and easily as any other, and in particular, access it in time to be of value to the user. In this sense the choice of data item is arbitrary in the sense that no matter which item is sought, all that is needed to find it, is its address, that is to say, the coordinates at which it is located, such as its row and column (or its track and record number on a magnetic drum). At first the term "random access" was used because the process had to be capable of finding records no matter in which sequence they were required. However, soon the term "direct access" gained favour because one could directly retrieve a record, no matter what its position might be. The operative attribute however is that the device can access any required record immediately on demand. The opposite is sequential access, where a remote element takes longer time to access [16].

Direct access is based on a disk model of a file. For direct access, the file is viewed as a numbered sequence of block or records. A direct-access file allows arbitrary blocks to be read or written. Thus, after block 18 has been read, block 57 could be next, and then block three. There are no restrictions on the order of reading and writing for a direct access file. Direct access files are of great use for intermediate access to large amounts of information.

The file operations must be modified to include the block number as a parameter. Thus, we have “read n”, where n is the block number, rather than “read next”, and “write n”, rather than “write next”. An alternative approach is to retain “read next” and “write next” and to add an operation; “position file to n” where n is the block number. Then, to effect a “read n”, we would issue the commands “position to n” and then “read next”.

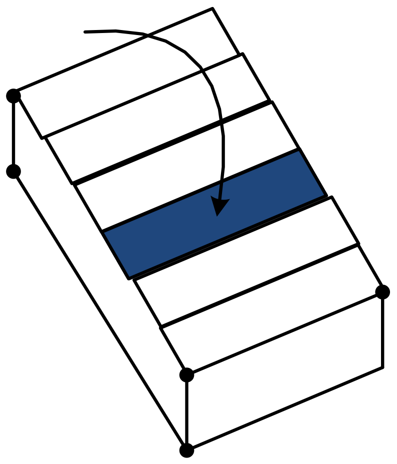


Figure 1.3. Direct access

Not all OS support both sequential and direct access for files. Some systems allow only sequential file access; others allow only direct access. Some systems require that a file be defined as sequential or direct when it is created; such a file can be accessed only in a manner consistent with its declaration.

Following, code of “reading Data from Random Access Files in C#” is given [19]:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
namespace ReadFile
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] byData = new byte[200];
            char[] charData = new Char[200];

            try
            {
                FileStream        aFile        =        new
FileStream(“sometext.txt”, FileMode.Open);
                aFile.Seek(113, SeekOrigin.Begin);
                aFile.Read(byData, 0, 200);
            }
            catch (IOException e)
            {
                Console.WriteLine(“An IO exception has been
thrown!”);
```

```
                        Console.WriteLine(e.ToString());
                        Console.ReadKey();
                        return;
                    }

                    Decoder d = Encoding.UTF8.GetDecoder();
                    d.GetChars(byData,  0,  byData.Length,  charData,
0);

                    Console.WriteLine(charData);
                    Console.ReadKey();
                }
            }
        }
```

1.4. Indexed Sequential Access

Indexed Sequential Access Method (ISAM) is a record-oriented database architecture that permits extremely fast access to data. ISAM data is organized into records that are stored in data files. Separate index files are used to store key values that identify each record, along with pointers that locate the corresponding record in the data file. The index makes it possible to retrieve individual records quickly without having to search an entire data set. Programmers often use a C ISAM API to define the record and index structures for their application [7].

For example, an accounts payable system using an ISAM database structure may define a vendor data file indexed by vendor name and number, and an invoice data file indexed by vendor number and invoice number. In this case, each data file has two indices. Although vendor number indexes each data file, there is a separate vendor number index for each data file.

The record-oriented ISAM database architecture has been around for many years, but FairCom has continuously evolved and improved the fundamental concepts. The c-treeACE ISAM database engine provides an advanced set of functions for manipulating data and index files. Each function performs multiple file and index access operations resulting in fast performance as compared to other database engines.

A significant way in which ISAM (and other non-sequential file organization methods) differs from sequential organization is that the record keys in an indexed file must be unique; this is a system requirement, not just a programming practice. Consequently, an indexed file is typically a master file. Also, there is a clear difference between updating a sequential file and updating an indexed file. When you update a sequential file, you rewrite the entire file; this practice leaves the original file as a convenient backup in case the job must be rerun. When you update an indexed file, the system rewrites records in the file directly in place, thereby providing no automatic backup file. To create a backup, you periodically copy the file onto another device.

The flexibility of indexed sequential access method is realized at some cost in both storage space and accessing time. First, the system requires various levels of indexes to help locate records in the file. Second, the system stores new added records in special reserved overflow areas. Check that your system supports ISAM before attempting to use it [7].

Characteristics of indexed sequential files. ISAM initially stores records sequentially and permits both sequential and random processing. The features that provide this flexibility are indexes to locate a correct cylinder and track and keys to locate a record on a track.

Keys. A key is a record control field such as customer number or stock number. Records in an indexed file are in sequence by key to permit sequential processing and to aid in locating records randomly, and blocks are formatted with keys. That is, ISAM writes each block immediately preceded by the highest key within the block, namely, the key of the last or only record in the block. The key is usually also embedded within each data record, as normal.

Unblocked Records. This is the layout of keys for unblocked records:

Key 201		Record 201	Key 205		Record 205	Key 206		Record 206

Figure 1.4. Unblocked Records

The records could represent, for example, customer numbers, and the keys could be for customer numbers 201, 205, and 206. In this example, the key is 3 characters long and the data record is the conventional size. Under unblocked format, a key precedes each block containing one record.

Blocked Records. This is the layout of keys for blocked records based on the preceding unblocked example:

Key 206		Record 201	Record 205	Record 206

Figure 1.5. Blocked Records

Under blocked format, the key for the last record in the block, 206, precedes the block.

ISAM automatically handles this use of keys, and when you perform a read operation, the system delivers the block, not the separate key, to main storage.

Indexes. To facilitate locating records randomly, ISAM maintains three levels of indexes on disk: track index, cylinder index, and an optional master index.

Track index. When ISAM creates a file, it stores a track index in track 0 of each cylinder that the file uses. The track index contains the highest key number for each track on the cylinder. For example, if track 4 on cylinder 12 contains records with keys 201,205,206, and 208, the track index contains an entry for key 208 and a reference to cylinder 12, track four.

If a disk device has ten tracks per cylinder, there are ten key entries for each track index, in ascending sequence.

Cylinder index. When ISAM creates a file, it stores a cylinder index on a separate cylinder containing the highest key for each cylinder. For example, if the file is stored on six cylinders, the cylinder index contains six entries.

Master index. An optional master index facilitates locating an appropriate cylinder index. This index is recommended if the entries in the cylinder index exceed four cylinders, which is a very large file.

2. The main part. Access Control Methods

2.1. Overview of Access Control

2.1.1. Basics of Access Control

Access control is concerned with determining the allowed activities of legitimate users, mediating every attempt by a user to access a resource in the system. A given information technology (IT) infrastructure can implement access control systems in many places and at different levels. Operating systems use access control to protect files and directories. Database management systems DBMS apply access control to regulate access to tables and views. Most commercially available application systems implement access control, often independent of the operating systems and/or DBMSs on which they are installed.

The objectives of an access control system are often described in terms of protecting system resources against inappropriate or undesired user access. From a business perspective, this objective could just as well be described in terms of the optimal sharing of information. After all, the main objective of IT is to make information available to users and applications. A greater degree of sharing may get in the way of resource protection; in reality, a well-managed and effective access control system actually facilitates sharing. A sufficiently fine-grained access control mechanism can enable selective sharing of information where in its absence, sharing may be considered too risky altogether.

Access control is all about, well, controlling access. First, let’s define a few terms [5].

Object: An entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are records, fields (in a database record), blocks, pages, segments, files, directories, directory trees, process, and programs, as well as processors, video displays, keyboards, clocks, printers, and network nodes. Devices such as electrical switches, disc drives, relays, and mechanical components connected to a computer system may also be included in the category of objects.

Subject: An active entity, generally in the form of a person, process, or device that causes information to flow among objects (see below) or changes the system state.

Operation: An active process invoked by a subject; for example, when an automatic teller machine (ATM) user enters a card and correct personal identification number (PIN), the control program operation on the user's behalf is a process, but the subject can initiate more than one operation-deposit, withdrawal, balance inquiry, etc.

Permission (privilege): An authorization to perform some action on the system. In most computer security literature, the term permission refers to some combination of object and operation. A particular operation used on two different objects represents two distinct permissions, and similarly, two different operations applied to a single object represent two distinct permissions. For example, a bank teller may have permissions to execute debit and credit operations on customer records through transactions, while an accountant may execute debit and credit operations on the general ledger, which consolidates the bank's accounting data.

Access Control List (ACL): A list associated with an object that specifies all the subjects that can access the object, along with their rights to the object. Each entry in the list is a pair (subject, set of rights). An ACL corresponds to a column of the access control matrix (described next). ACLs are frequently implemented directly or as an approximation in modern operating systems.

Access Control Matrix: A table in which each row represents a subject, each column represents an object, and each entry is the set of access rights for that subject to that object. In general, the access control matrix is sparse: most subjects do not have access rights to most objects. Therefore, different representations have been proposed. The access control matrix can be represented as a list of triples, having the form <subject, rights, object>. Searching a large number of these triples is inefficient enough that this implementation is seldom used. Rather, the matrix is typically subdivided into columns (ACLs) or rows (capabilities).

Table 2.1

An example of access matrix

	File 1	**File 2**	**File 3**	**Program 1**
Ann	Own, read, write	Read, write		Execute
Bob	Read		Read, write	
Karl		Read		Execute, read

Separation of Duty (SOD): The principle that no user should be given enough privileges to misuse the system. For example, the person authorizing a paycheck should not also be the one who can prepare it. Separation of duties can be enforced either statically by defining conflicting roles (i.e., roles which cannot be executed by the same user) or dynamically by enforcing the control at access time. An example of dynamic separation of duty is the two-person rule. The first user to execute a two-person operation can be any authorized user, whereas the second user can be any authorized user different from the first. There are various types of SOD; an important one is a history-based SOD that regulates, for example, that the same subject (role) cannot access the same object a certain number of times.

Safety: Measures that the access control configuration (e.g., access control mechanism or model) will not result in the leakage of permissions to an unauthorized principal. Thus, a configuration is said to be safe if no permission can be leaked to an unauthorized or unintended principal.

Domain and Type Enforcement: The grouping of processes into domains, and objects into types, such that access operations (such as read, write, execute, and create) are restricted from domains to types and between domains. A process belongs to one domain at any given time and transits to other domains by sending signals or executing a file in a new domain.

Access Control Administration. Once an organization chooses an access control design, the next step is to decide on the method of access control administration. Access control administration can be implemented in both centralized and decentralized modes. It is not uncommon to find hybrid environments where both approaches are used. The best choice of administration style depends on the needs of the organization and the sensitivity of information stored on the affected computer systems.

Centralized access control administration requires that all access requests go through a central authority that grants or denies the request. This approach simplifies administration because objects must be maintained only in a single location. One drawback is that the central access control unit is a single point of failure. If the centralized access control unit fails, no access can be granted to objects, so all objects are effectively unavailable. In addition, the central point of access control can have a negative effect on performance if the system is unable to keep up with all access requests. You can choose from several common packages to implement centralized access control administration [20].

Remote Authentication Dial-In User Service (RADIUS) provides centralized access control for dial-in users. Users are validated against the user list on the RADIUS server. You can configure the server to hang up and then call the valid user back at a predefined telephone number.Another example of centralized access control for dial-in users is *Challenge Handshake Authentication Protocol (CHAP)*. CHAP presents a challenge when a user requests access. If the user responds to the challenge properly, the access request is granted. CHAP enhances overall security by using encryption during the message exchanges.

Centralized access control for networked applications can use *Terminal Access Controller Access Control System (TACACS)*. TACACS provides general centralized authentication and authorization services. Extended TACACS

(XTACACS) extends TACACS by separating the authentication, authorization, and accounting processes, and TACACS+ adds two-factor authentication.

Decentralized access control places the responsibility of access control administration closer to the object in question. This approach requires more administration than centralized access control because an object may need to be secured at several locations. It can, however, be more stable because no single point of failure or single point of access exists. Decentralized access control is most often implemented through the use of security domains. A security domain is a sphere of trust, or a collection of subjects and objects, with defined access rules or permissions. A subject must be included in the domain to be trusted. This approach makes it fairly easy to exclude an untrusted subject, but makes general administration more difficult due to the fine granularity of security rules.

2.1.2. Identification and Authentication Methods

The first user interface element most subjects encounter when accessing an information system is the *identification* and *authentication* challenge. The identification phase allows a subject to claim to be a specific entity by presenting identifying credentials. These credentials could be as simple as a user ID or personal identification number (PIN), or more complex, such as a physical attribute. Once a subject has claimed an identity, the system validates that the user exists in the user database, and then authenticates that the subject really is who she claims to be. The authentication phase asks the subject to present additional information that matches stored information for that subject. These two phases, often called *two-factor authentication*, provide reasonable protection from unauthorized subjects accessing a system. After a subject has been authenticated, the access control system then evaluates the specific rights or permissions for the subject to grant or deny object access requests. This phase is called the authorization phase [20].

There are three general categories, or types, of authentication information. Best security practices generally dictate that the identification and authentication phases require input from at least two different types. Table 2.2 lists and describes the three common types of authentication data.

Table 2.2

Authentication Types

Authentication Type	**Description**	**Examples**
Type 1	What you know	Password, passphrase, PIN, lock combination
Type 2	What you have	Smart card, token device
Type 3	What you are	Biometrics—fingerprint, palm print, retina/iris pattern, voice pattern

The most common and easiest type of authentication to implement is Type 1 authentication. All you have to do is ask the subject to make up a password, passphrase, or PIN. The alternative is to provide one for the user. The difficulty with Type 1 authentication is that you must encourage subjects to create challenge phrases that are very difficult for others to guess, but not so complex that they cannot be easily remembered. If your requirements are so stringent that passwords (or passphrases or PINs) cannot easily be remembered, you will start to see notes stuck to monitors and keyboards with passwords written on them. That negates any value of the password. The same result can occur when administrators require that passwords be changed so often users do not have time to memorize the new ones. Keep passwords safe and secret. The following rules are a good starting point for creating secure passwords:

- ✓ Passwords should be at least six characters in length.
- ✓ Passwords should contain at least one number or punctuation character.
- ✓ Do not use dictionary words or combinations of dictionary words.

- ✓ Do not use common personal data, such as birth date, social security number, family member or pet name, or favorite song or hobby.
- ✓ Never write down your password.
- ✓ Try to make your password easy to remember but hard to guess.

Type 2 authentication data solutions are more complex to administer because subjects are required to carry a device of some sort. The device generally is electronic in nature and either generates a time-sensitive value or generates a value in response to input data. Although Type 2 authentication is more complex, it is almost always more secure than Type 1 authentication.

The most sophisticated authentication type is Type 3, or *biometrics*. Biometrics describes the detection and classification of physical attributes. There are many different biometric techniques, including:

- ✓ Fingerprint/palm scan
- ✓ Hand geometry
- ✓ Retina/iris scan
- ✓ Voice print
- ✓ Signature/keyboard dynamics

Due to the complexity of biometrics, it is the most expensive authentication type to implement. It is also more difficult to maintain due to the imperfect nature of biometrics analysis. You should be aware of several important issues regarding biometrics errors. First, a biometrics system could reject an authorized subject. The rate at which this failure occurs is called the false rejection rate (FRR). On the other hand, the biometrics system could accept an invalid subject. The rate at which this failure occurs is called the false acceptance rate (FAR). The problem is that when you adjust the sensitivity of the biometrics system to reduce the FRR, the FAR increases. The inverse is also true. So, what is the best setting? The best balance between the FRR and FAR occurs when the rates are equal. This occurs at the crossover error rate (CER). Figure 2.1 shows the CER in relation to the FRR and FAR of a general biometrics device.

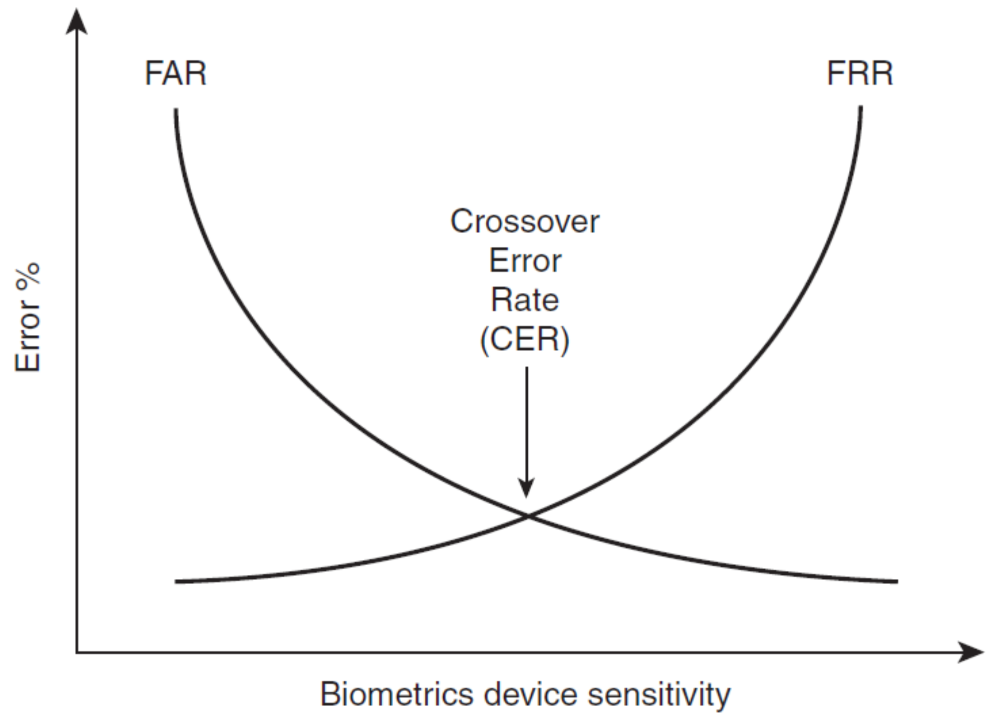


Figure 2.1. Biometrics errors

Single Sign-On. The more pieces of information, or factors, you request from a subject, the more assured you could be that the subject is who she claims to be. Thus, two-factor authentication is more secure than single-factor. The problem is that if a subject needs to access several resources on different systems, she may be required to provide identification and authentication information at each different system. This quickly becomes tedious. Single sign-on (SSO) systems avoid multiple logins by positively identifying a subject and allowing the authentication information to be used within a trusted system or group of systems. Users love SSO, but administrators have a lot of additional work to do. You must take extreme care to ensure the authentication credentials are not compromised or intercepted as they pass across the network [13].

Several good SSO systems are in use today. It is not important to understand the details of each one. The important concepts and difficulties are common to all SSO products. We look at one product, Kerberos, to examine how these systems work.

Kerberos. The Kerberos system came from the Massachusetts Institute of Technology's (MIT's) project Athena. It is named after the three-headed dog from Greek mythology that guards the gates of the underworld. Kerberos provides both authentication and message protection. It uses symmetric key cryptography (both sides have the same key) to encrypt messages. The encryption feature provides end-to-end security, meaning the intermediate machines between the source and target machines cannot see the contents of messages. Kerberos is growing in popularity for use in distributed systems. Although it works well in distributed environments, Kerberos itself uses a centralized server to store the cryptographic keys [13].

Kerberos includes a data repository and authentication process. The Key Distribution Center (KDC) is at the heart of Kerberos. The KDC stores all of the cryptographic keys for subjects and objects. The KDC is responsible for maintaining and distributing these keys, as well as for providing authentication services.When the KDC receives a request for access to an object, it calls the Authentication Service (AS) to authenticate the subject and its request. If the subject's request is authenticated, the AS creates an access ticket that contains keys for the subject and the object. It then distributes the keys to both the subject and the object. Here are the basic steps in a Kerberos access request cycle:

1. The subject requests access to an object. The subject's Kerberos software prompts for a user ID, and sends the user ID along with the request to the KDC.
2. The KDC calls the AS to authenticate the subject and the object.
3. If authenticated, the KDC sends an encrypted session key to the subject and the object's machine.
4. The subject's Kerberos client software prompts the subject for a password and uses it, along with the subject's secret key, to decrypt the session key.
5. The subject then sends the access request with the session key to the object.

6. The object decrypts the session key it received from the KDC and compares it to the session key it received with the access request.
7. If the two session keys match, access is granted.

The centralized nature of the KDC exposes one of Kerberos' main weaknesses: The KDC is a single point of failure. KDC failure means object access failure. The KDC can also cause a performance bottleneck on heavily utilized machines. In addition, there is a small window of time when the session key lives on the client machines. It is possible for an intruder to capture this key and gain unauthorized access to a resource. In spite of several weaknesses, Kerberos is a good example of SSO systems and has enjoyed widespread acceptance.

2.1.3. File and Data Ownership

Files and data may contain important and valuable information. This important information should be the focus of your security efforts. But who is responsible for ensuring the security of your organization's information? This question is answered by assigning different layers of responsibility to each piece of important information. Each file, or data element, should have at least three different responsible parties assigned. The three layers of responsibility represent different requirements and actions for each group. The most common layers are data owner, data custodian, and data user. Each layer has specific expectations to support the organization's security policy [20].

Data Owner. The data owner accepts the ultimate responsibility for the protection of the data. The data owner is generally a member of upper management and acts as the representative of the organization in this duty. It is the owner who sets the classification level of the data and delegates the day-to-day responsibility of maintenance to the data custodian. If a security violation occurs, it is the data owner who bears the brunt of any negligence issues.

Data Custodian. The data owner assigns the data custodian to enforce security policies according to the data classification set by the data owner. The custodian is often a member of the IT department and follows specific procedures to secure and protect assigned data. This includes implementing and maintaining appropriate controls, taking backups, and validating the integrity of the data.

Data User. Finally, the users of data are the ones who access the data on a day-to-day basis. They are charged with the responsibility of following the security policy as they access data. You would expect to see more formal procedures that address important data, and users are held accountable for their use of data and adherence to these procedures. In addition to a commitment to follow security procedures, users must be aware of how important security procedures are to the health of their organization. All too often, users use shortcuts to bypass weak security controls because they lack an understanding of the importance of the controls. An organization’s security staff must continually keep data users aware of the need for security, as well as the specific security policy and procedures.

2.1.4. Related Methods of Attacks

The main purpose for implementing access controls is to block unauthorized access to sensitive objects. The primary purpose of attackers is to access these same objects. Several attack types are related to access controls. Most attacks directed toward access controls are designed to thwart, or bypass, the controls and allow access to unauthorized subjects. One of the best ways to decide which controls to put into place is to understand the nature of the attack you are trying to stop. Let’s take a look at three of the most common access control attacks [20].

Brute Force Attack. Brute force attacks are unsophisticated attacks that can be effective. The purpose of such an attack is to attempt every possible combination of characters to satisfy Type 1 authentication. Often called password guessing, a program submits many login attempts, each with a slightly different password. The hope is that the program will hit on the correct password before anyone notices an attack is underway. One variation of the brute force attack is war dialing, in which a program dials a large group of telephone numbers and listens for a modem to answer. When the war dialing program finds a modem, the number is logged for later probing and attacks. These attacks are called brute force attacks because they attempt a very large number of possibilities to find the password or access number.

The best defense is a good offense. A great way to protect your system from a brute force attack is to run one yourself. It is a good idea to run a password cracking or war dialing program against your system periodically. Make sure you have written permission to execute the attack first. You could find that you are violating your security policy as you try to protect it. Once is not enough. Any time a user gets tired of a password or finds that getting access to the Internet is too hard, you will start seeing easily cracked passwords and unauthorized modems showing up. Run your brute force attacks periodically to find users who are taking shortcuts.

In addition to running your own attacks, set your monitoring system clipping levels to warn you when unusual activity occurs. It is also a good idea to set aggressive lockout levels so accounts are locked after a certain number of login failures. As frustrating as this is to honest users who have forgotten passwords, it provides a great defense against brute force attacks.

Dictionary Attack. A dictionary attack is actually a subset of a brute force attack. Instead of trying all password combinations, a dictionary attack attempts to satisfy a password prompt by trying commonly used passwords from a list, or dictionary. Many lists of commonly used user IDs and passwords exist and are easy to find. Although they make great input sources for dictionary attacks, they also provide examples of user IDs and passwords to avoid. In fact, one of the best deterrents to a dictionary attack is a strong password policy. A password policy tells users how to construct passwords and what types of passwords to avoid. You can avoid having nearly all of your passwords appear in a password dictionary by creating and enforcing a strong password policy. Once passwords are in place, run dictionary attacks periodically. These attacks are not as intensive as brute force attacks and give you a good idea who is abiding by your password policy. You can also avoid password disclosure by never sending passwords as clear text. Avoid using HTTP or Telnet for that reason. When you need to send a password to a Web application, use another protocol, such as HTTP-S.

Spoofing Attack. Another interesting type of access control attack is login spoofing. An attacker can place a fake login program that prompts a user for a user ID and password. It probably looks just like the normal login screen, so the user likely provides the requested information. Instead of logging the user into the requested system, the bogus program stores or forwards the stolen credentials, then returns a notice that the login has failed. The user is then directed to the real login screen. The beauty of the approach is that few of us would ever think of a spoofing attack if we were presented with a failed login screen. Most of us would chalk it up to a typo.

The best defense against this type of attack is to create trusted paths between users and servers when at all possible. Attempt to minimize the opportunities for attackers to step in between users and servers. In environments where security is extremely important, users should carefully examine all failed login attempts and ensure the failure is properly recorded and reported. If, after being alerted your login has failed, you find that the system thinks the last login failure happened last week, you may have been spoofed. Security awareness goes a long way in preventing and detecting these types of attacks.

2.2. Access Control Models

Access control models are very useful when deciding what controls are necessary to support your security policy. An access control model provides a conceptual view of your security policy. It allows you to map goals and directives of your security policy to specific system events. This mapping process allows for the formal definition and specification of required security controls. In short, access control models make it possible to decompose complex policies into a series of manageable steps. Many different models have been developed over the years. We look at some of the more important models and discuss some of their unique characteristics in the following sections. Most sound security policy implementations employ a combination of the following access control models.

State Machine Model. *A state machine model* is a collection of defined instances, called states, and specific transitions that permit a modification to occur that changes an object from one state to another. State machines are often used to model real-life entities when specific states and the transitions from one state to another exist and are understood. Think of a state as being objects at a certain point in time.When a subject requests to read an object, there must be a defined transition that allows an object to change from a closed, unread object to an open object. Figure 2.2 shows a diagram of a simple state machine. States are represented with circles, and transitions are represented with arrows [20].

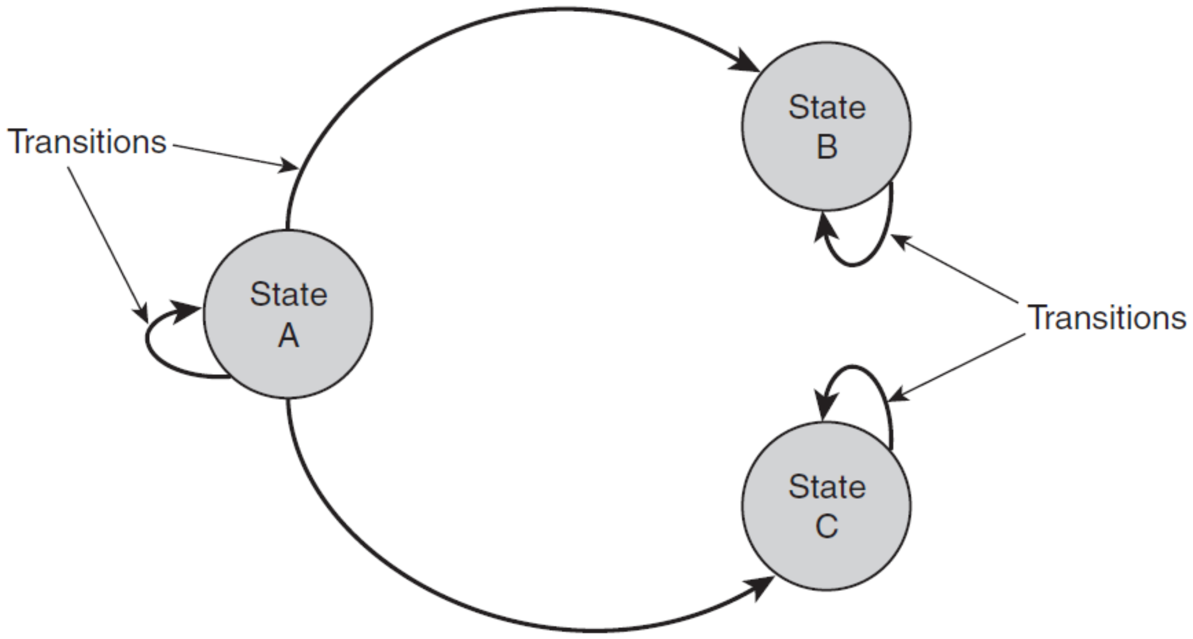


Figure 2.2. Simple state machine

The following sections cover four important models: Bell-LaPadula, Biba, Clark-Wilson, and noninterference.

Bell-LaPadula Model. The Bell-LaPadula model was developed in the 1970s to help better understand and implement data confidentiality controls. The

U.S. military was very interested in protecting classified data while allowing an increasing number of users access to the machines that stored the confidential data. Because the military is most interested in data confidentiality, this model works well in organizations that focus mainly on the confidentiality controls. The Bell-LaPadula model is a state machine model that employs access control lists and security labels to implement object security [20].

The model uses two basic properties to evaluate access requests. Table 2.3 shows the basic properties and their common names.

Table 2.3

Bell-LaPadula Properties

Property	**Common Name**	**Description**
Simple security rule	No read up	A subject of a given security clearance cannot read data from a higher security level.
*-property (star property)	No write down	A subject of a given security clearance cannot write to an object at a lower security level.

The properties may seem confusing at first, but think about what each property states. Remember that confidentiality is the focus. The simple security rule protects information from being disclosed to an unauthorized subject. The *-property protects sensitive or secret data from being inserted into an object of a lower security level. If this were allowed, you could paste a paragraph from a top-secret document into a document that is classified as public. Such a write would disclose the top-secret information to anyone cleared to see public documents. This would clearly violate the confidentiality of the information that was pasted into the public document.

Biba Model. The Biba model was developed after the Bell-LaPadula model to address the issue of data integrity. The Biba model is also built on the state machine model and defines states and transitions that focus on the integrity of the data instead of the confidentiality. The Biba model quickly became popular with businesses because its main focus is to ensure that unauthorized subjects cannot change objects [20].

Similar to the Bell-LaPadula model, the Biba model uses two basic properties to evaluate access requests. Table 2.4 shows the basic Biba properties and their common names.

Table 2.4

Biba Properties

Property	Common Name	Description
Simple integrity property	No read up	A subject cannot read an object of a lower integrity level.
*-property (star property)	No write up	A subject cannot write to an object of a higher integrity level.

Clark-Wilson Model. The Clark-Wilson model was developed after the Biba model. Unlike the Bell-LaPadula and Biba models, the Clark-Wilsonmodel is not based on the state machine model; it takes a different approach to ensure data integrity. Instead of granting access of a subject to an object, the Clark-Wilson model restricts all accesses to a small number of tightly controlled access programs. The model uses security labels to grant access to objects through the access programs. This approach works well in commercial applications where data integrity is often more important than overall data confidentiality.

The Clark-Wilson model defines several terms that are necessary to understand in order to follow the model's access path:

- ✓ **Constrained data item(CDI):** Any data item protected by the model.
- ✓ **Unconstrained data item (UDI):** Data not protected by the model (for example, data input or output).
- ✓ **Integrity verification procedure (IVP):** Procedure that verifies the integrity of a data item.

- ✓ **Transformation procedure (TP):** Any procedure that makes authorized changes to a data item.

The Clark-Wilson model ensures all unconstrained data is validated by the IVP, and then submitted to the system by the TP. All subsequent modifications are first validated by the IVP, and then the modification takes place by the TP. Of course, the IVP and TP are not called until the subject has been properly authenticated and cleared to access the object in question.

Noninterference Model. The last access controlmodel is often an addition to othermodels. The noninterference model ensures that changes at one security level do not "bleed over" into another security level and affect an object in another context. For example, what would happen if you saved a secret document that was embedded in a public document? The dangers in this case are obvious: You risk disclosing secret data when the information is copied to the public document. The basic premise of the noninterference model is that each security level is distinct and changes will not interfere across levels. This assurance reduces the scope of any change and reduces the possibility that a change will have unintended side effects. By isolating modifications to a specific security level, this model can maintain both data integrity and confidentiality.

2.3. Access Control Techniques

Access Control Designs. An access control design defines rules for users accessing files or devices.We refer to a user, or any entity, that requests access as a subject. Each subject requests access to an entity called an object. An object can be any entity that contains data or resources a subject requests to complete a task. Objects can be files, printers, or other hardware or software entities. The access control type in use for a particular request has the responsibility of evaluating a subject's request to access a particular object and returning a meaningful response. Let's look at three common access control designs [5].

Mandatory Access Control (MAC). Mandatory access control assigns a security label to each subject and object. A security label is an assigned level of sensitivity. Some examples of sensitivity levels are public, sensitive, and secret. Tables 2.6 and 2.5 list common security labels for military and commercial uses. A subject's security label defines the security clearance, or category of object, that the subject is permitted to access. For example, a subject with a clearance of "secret" can only access objects with a security label of "secret." Some access control methods allow the same subject to access objects of a lower classification, whereas others do not. Mandatory Access Control is usually associated with the 1973 Bell-LaPadula Model of multi-level security [8,10,12].

Table 2.5

Military Data Classifications, from Lowest Sensitivity to Highest

Classification	Description
Unclassified	Data that is not sensitive or classified
Sensitive but unclassified (SBU)	Data that could cause harm if disclosed
Confidential	Data for internal use that is exempt from the Freedom of Information Act
Secret	Data that could cause serious damage to national security
Top secret	Data that could cause grave damage to national security

Table 2.6

Commercial Data Classifications

Classification	Description
Public	Data not covered elsewhere
Sensitive	Information that could affect business and public confidence if improperly disclosed
Private	Personal information that could negatively affect personnel, if disclosed
Confidential	Corporate information that could negatively affect the organization, if disclosed

One common implementation of mandatory access control is *rule-based access control*. In a rule-based access control system, all access rights are granted by referencing the security clearance of the subject and the security label of the object. Then, a rule set determines whether an access request should be granted or denied. The rules in place depend on the organization's needs. In addition to matching subject security clearance to an object's security label, many systems require a subject to possess a need to know. The need to know property indicates that a subject requires access to an object to complete a task. Thus, access is granted based on both security labels and specific task requirements [12].

Benefits. Through its implementation of Bell-LaPadula in Multi-Layer Secure (MLS) systems, MAC is the main access control model used by the military and intelligence agencies to maintain classification policy access restrictions. The combination of Bell-LaPadula and trusted component assurance also has the nice benefit of making MLS systems immune to Trojan Horse attacks. In perfect implementations, MLS systems implementing Bell-LaPadula MAC are not susceptible Trojan Horse forced security violations because users do not have the ability to declassify information. Additionally, MAC is relatively straightforward and is considered a good model for commercial systems that operate in hostile environments (web servers and financial institutions) where the risk of attack is very high, confidentiality is a primary access control concern, or the objects being protected are valuable.

Problems. MAC, however, is not without serious limitations. The assignment and enforcement of security levels by the system under the MAC model places restrictions on user actions that, while adhering to security policies, prevents dynamic alteration of the underlying policies, and requires large parts of the operating system and associated utilities to be "trusted" and placed outside of the access control framework.

Trusted components are processes and libraries, such as declassifying cryptographic processes, that need to violate MAC principles and thus must sit outside of the MAC model. In order to maintain the security policies and prevent unauthorized or inappropriate access, the code behind these components is assumed (hopefully with verification) to be correct and conforming to the underlying security policies of the system. Additional access control methods must be used to restrict access to these trusted components. Unfortunately, in practice it has been shown that it is virtually impossible to implement MLS using MAC without moving essentially the entire operating system and many associated utilities outside the MAC model and into the realm of trusted components.

Additionally, MAC can unnecessarily over-classify data through the high-water mark principle and hurt productivity by limiting the ability to transfer labeled information between systems and restricting user control over data. MAC also does not address fine-grained least privilege, dynamic separation of duty or security or validation of trusted components.

MAC systems are difficult and expensive to implement due to the reliance on trusted components and the necessity for applications to be rewritten to adhere to MAC labels and properties. It is rumored that application developer reluctance to take these steps is the reason behind Microsofts abandonment of MAC related trusted computing in their long awaited new operating system.

Discretionary Access Control (DAC). MAC, while immensely important to military applications, is not the most widely used method of access control. Discretionary access control uses the identity of the subject to decide whether to grant or reject an access request. The object’s owner defines which subjects can access the object, so all access to the object is at the discretion of the object owner. This access control design is generally less secure than mandatory access control, but is the most common design in commercial operating systems. Although it tends to be less secure, it is easier to implement and more flexible for environments that do not require stringent object security. Most objects have permissions, or rights, that specify which users and groups can access the object. This method of granting rights is an example of discretionary access control [12].

Discretionary access control implementations include identity-based access control and access control lists. Identity-based access control makes object access decisions based on a user ID or a user’s group membership. An object owner specifies what users or user groups can access each object. When a subject requests access to the object, the subject’s credentials are presented and evaluated to grant or deny the request. Most operating systems allow the owners of files and other resources to specify the read, write, and execute permissions based on users and groups. To make the administration a little easier, access control lists (ACLs) allow groups of objects, or groups of subjects, to be controlled together. An access control list can grant a subject access to a group of objects or grant a group of subjects access to a specific object.

Benefits. A primary benefit associated with the use of DAC is enabling fine-grained control over system objects. Through the use of fine-grained controls, DAC can easily be used to implement least-privilege access. Individual objects can have access control restrictions to limit individual subject access to the minimum rights needed. DAC is also intuitive in implementation and is mostly invisible to users so it is regarded as the most cost-effective for home and small-business users.

Problems. DAC, however, is not without issues. Allowing users to control object access permissions has a side-effect of opening the system up to Trojan horse susceptibility. Additionally maintenance of the system and verification of security principles is extremely difficult for DAC systems because users control access rights to owned objects. The so-called “Safety Problem, the lack of constraints on copy privileges, is another liability inherent to DAC. The lack of constraints on copying info from one file to another makes it difficult to maintain safety policies and verify that safety policies have are not compromised while opening potential exploits for Trojan horses.

Nondiscretionary Access Control (or Role Based Access Control (RBAC)). The third common access control design is nondiscretionary access control. This design most commonly uses a subject’s role, or a task assigned to the subject, to grant or deny object access. Because nondiscretionary access control is generally based on roles or tasks, it is also called role-based access control or task-based access control. This type of access control works well in cases with high turnover or reassignments.When security is associated with a role or task, replacing the person who carries out the task makes security administration easier. At first glance, a role may look like a group, but there are several differences. Although users generally can be associated with multiple groups, users normally are assigned only to a single role. Groups can also represent several types of user associations, but a role represents general tasks a user must perform [12].

Lattice-based access control is a variation of the nondiscretionary access control design. Instead of associating access rules with specific roles or tasks, each relationship between a subject and an object has a set of access boundaries. These access boundaries define the rules and conditions that allow object access. In most cases, the access boundaries define upper and lower limits that correspond to security classifications and labels.

Differences from MAC/DAC. First proposed by Ferraiolo and Kuhn in 1992,[7] RBAC addresses most of the failings of DAC while maintaining DACs focus on non-military systems. RBAC approaches commercial and civilian governmental security needs from an integrity first, confidentiality second position based on Clark and Wilsons research into commercial security policies.

Security policies are maintained in RBAC through the granting of rights to roles rather than individuals. Right granting and policy enforcement is consolidated in the hands of a security administrator and users are prevented from transferring permissions assigned to a role they are allowed to perform to other users. This rule removes the ownership rights granted in DAC and thus behaves like a finer-grained version of the MAC model. In fact, this is precisely how Ferraiolo and Kuhn see RBAC.

RBAC also stands apart from the more traditional MAC and DAC by granted rights on transactions, not on underlying subjects. These rights are granted to roles, which at first glance appear to be a synonym for DAC groups. The difference lies in that groups consist of a collection of users while roles are a bridge between a collection of users and a collection of the Clark-Wilson model of transaction rights.

Benefits. Transaction based rights help ensure system integrity and availability by explicitly controlling not only which resources can be accessed but also how access can occur. In large organizations, the consolidation of access control for many users into a single role entry allows for much easier management of the overall system and much more effective verification of security policies.

Another benefit of RBAC is integrated support for principle of least-privilege, separation of duties, and central administration of role memberships and access controls. Separation of duties and least-privilege are not a part of MAC while central administration is loosely supported in MAC with trusted components and impossible in DAC due to the violation of the safety principle.

Problems. While RBAC marks a great advance in access control, the administrative issues of large systems still exist, albeit in a markedly more manageable form. In large systems, memberships, role inheritance, and the need for finer-grained customized privileges make administration potentially unwieldy.

Additionally, while RBAC supports data abstraction through transactions, it cannot be used to ensure permissions on sequences of operations need to be controlled. To do this, a less general and more sophisticated access control model must be used.

2.4. Design of “Control access of generated file objects” software

In this bachelor final work, C# programming languages used to create “Control access of generated file objects”. For C#, *FileSystemRights* enumeration used to get and set file rights. Following, quick information about this enumeration is given [21].

FileSystemRights Enumeration. Defines the access rights to use when creating access and audit rules. This enumeration has a *FlagsAttribute* attribute that allows a bitwise combination of its member values.

The *FileSystemRights* enumeration specifies which file system actions are allowed for a particular user account and which file system actions are audited for a particular user account.

Use the *FileSystemRights* enumeration when creating an access rule with the FileSystemAccessRule class or when creating an audit rule with the FileSystemAuditRule class.

This enumeration contains several granular system rights values and several values that are a combination of those granular values. It is easier to use the combination values such as FullControl, Read, and Write, rather than specifying each component value separately (Table 2.7) [21].

Table 2.7

FlagsAttribute

Member name	**Description**
AppendData	Specifies the right to append data to the end of a file.
ChangePermissions	Specifies the right to change the security and audit rules associated with a file or folder.
CreateDirectories	Specifies the right to create a folder. This right requires the Synchronize value. Note that if you do not explicitly set the Synchronize value when creating a file or folder, the Synchronize value will be set automatically for you.
CreateFiles	Specifies the right to create a file. This right requires the Synchronize value. Note that if you do not explicitly set the Synchronize value when creating a file or folder, the Synchronize value will be set automatically for you.
Delete	Specifies the right to delete a folder or file.
DeleteSubdirectories AndFiles	Specifies the right to delete a folder and any files contained within that folder.

ExecuteFile	Specifies the right to run an application file.
FullControl	Specifies the right to exert full control over a folder or file, and to modify access control and audit rules. This value represents the right to do anything with a file and is the combination of all rights in this enumeration.
ListDirectory	Specifies the right to read the contents of a directory.
Modify	Specifies the right to read, write, list folder contents, delete folders and files, and run application files. This right includes the ReadAndExecute right, the Write right, and the Delete right.
Read	Specifies the right to open and copy folders or files as read-only. This right includes the ReadData right, ReadExtendedAttributes right, ReadAttributes right, and ReadPermissions right.
ReadAndExecute	Specifies the right to open and copy folders or files as read-only, and to run application files. This right includes the Read right and the ExecuteFile right.
ReadAttributes	Specifies the right to open and copy file system attributes from a folder or file. For example, this value specifies the right to view the file creation or modified date. This does not include the right to read data, extended file system attributes, or access and audit rules.
ReadData	Specifies the right to open and copy a file or folder. This does not include the right to read file system attributes, extended file system attributes, or access and audit rules.
ReadExtendedAttributes	Specifies the right to open and copy extended file system attributes from a folder or file. For example, this value specifies the right to view author and content information. This does not include the right to read data, file system attributes, or access and audit rules.
ReadPermissions	Specifies the right to open and copy access and audit rules from a folder or file. This does not include the right to read data, file system attributes, and extended file system attributes.
Synchronize	Specifies whether the application can wait for a file handle to synchronize with the completion of an I/O operation.

	The Synchronize value is automatically set when allowing access, and automatically excluded when denying access. The right to create a file or folder requires this value. Note that if you do not explicitly set this value when creating a file, the value will be set automatically for you.
TakeOwnership	Specifies the right to change the owner of a folder or file. Note that owners of a resource have full access to that resource.
Traverse	Specifies the right to list the contents of a folder and to run applications contained within that folder.
Write	Specifies the right to create folders and files, and to add or remove data from files. This right includes the WriteData right, AppendData right, WriteExtendedAttributes right, and WriteAttributes right.
WriteAttributes	Specifies the right to open and write file system attributes to a folder or file. This does not include the ability to write data, extended attributes, or access and audit rules.
WriteData	Specifies the right to open and write to a file or folder. This does not include the right to open and write file system attributes, extended file system attributes, or access and audit rules.
WriteExtendedAttributes	Specifies the right to open and write extended file system attributes to a folder or file. This does not include the ability to write data, attributes, or access and audit rules.

Setting any, rights to file or directory are done by “File.SetAccessControl Method” and “Directory.SetAccessControl Method”.

File.SetAccessControl Method. Applies access control list (ACL) entries described by a FileSecurity object to the specified file.

public static void SetAccessControl(string path, FileSecurity fileSecurity) {}

Parameters:

path

Type: System.String

A file to add or remove access control list (ACL) entries from.

fileSecurity

Type: System.Security.AccessControl.FileSecurity

A FileSecurity object that describes an ACL entry to apply to the file described by the path parameter.

Directory.SetAccessControl Method. Applies access control list (ACL) entries described by a DirectorySecurity object to the specified directory.

public static void SetAccessControl(string path, DirectorySecurity directorySecurity)

Parameters:

path

Type: System.String

A directory to add or remove access control list (ACL) entries from.

directorySecurity

Type: System.Security.AccessControl.DirectorySecurity

A DirectorySecurity object that describes an ACL entry to apply to the directory described by the path parameter.

Add the FileSystemAccessRule to the security settings:

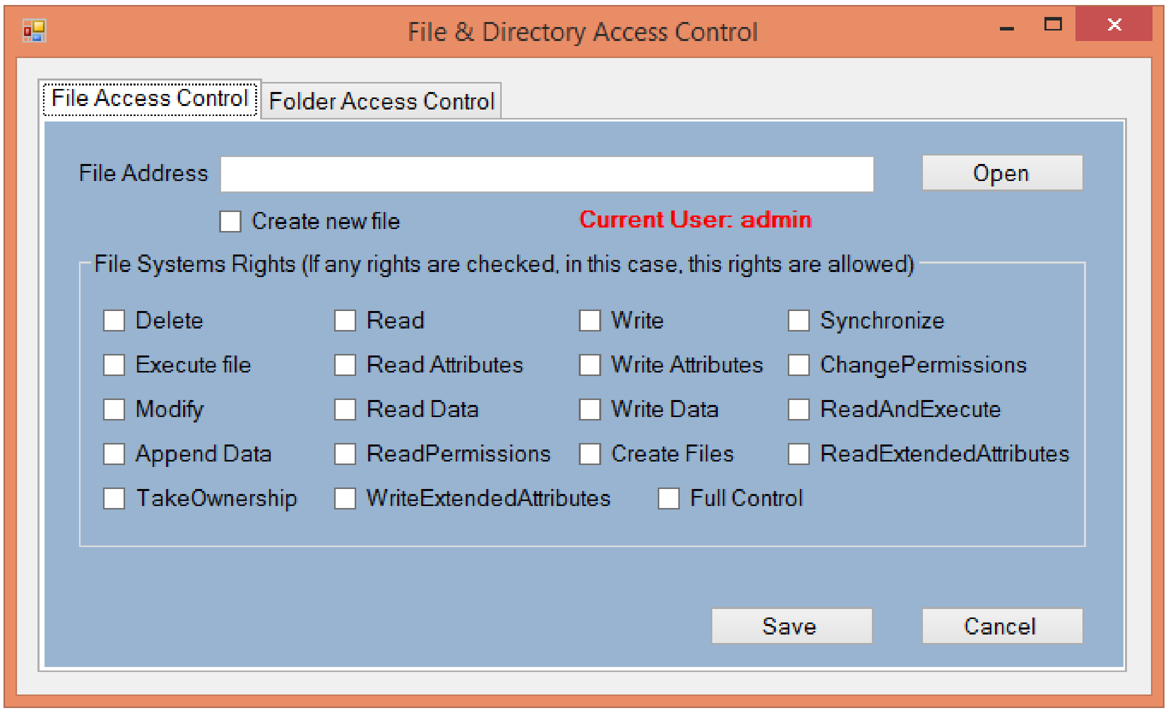
DirectorySecurity.AddAccessRule (new FileSystemAccessRule(Account, Rights, ControlType));

Remove the FileSystemAccessRule from the security settings:

DirectorySecurity.RemoveAccessRule (new FileSystemAccessRule (Account, Rights, ControlType));

To design “File Access Control” program, C# programming language is used. Program consist two part: file access control and directory access control [14].

Overall form of program is followed:


File & Directory Access Control

File Access Control | Folder Access Control

File Address [ ] Open

☐ Create new file — **Current User: admin**

File Systems Rights (If any rights are checked, in this case, this rights are allowed)

☐ Delete	☐ Read	☐ Write	☐ Synchronize
☐ Execute file	☐ Read Attributes	☐ Write Attributes	☐ ChangePermissions
☐ Modify	☐ Read Data	☐ Write Data	☐ ReadAndExecute
☐ Append Data	☐ ReadPermissions	☐ Create Files	☐ ReadExtendedAttributes
☐ TakeOwnership	☐ WriteExtendedAttributes	☐ Full Control	

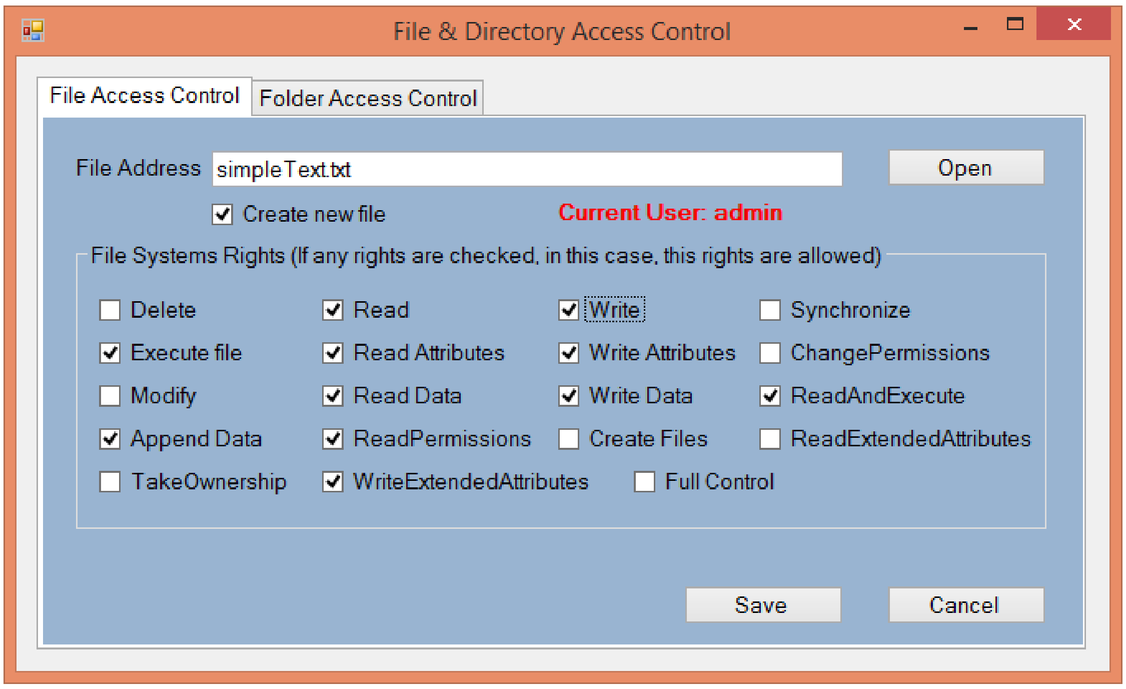
Save | Cancel

Figure 2.3. Main window of program

First part consists of File Access Control and File Systems Rights given Table 2.7. By this program, change current file rights or create new file with defined right.

To create new file, first check “create new file icon”, just enter new file name and check file right. Note that, these operations do for windows current windows (Figure 2.4, 2.5).

Second advantage of program is change current file rights. For this, first select file and program read file rights. After that, users can change file rights (Figure 2.6).


File & Directory Access Control

File Access Control | Folder Access Control

File Address: simpleText.txt

☑ Create new file

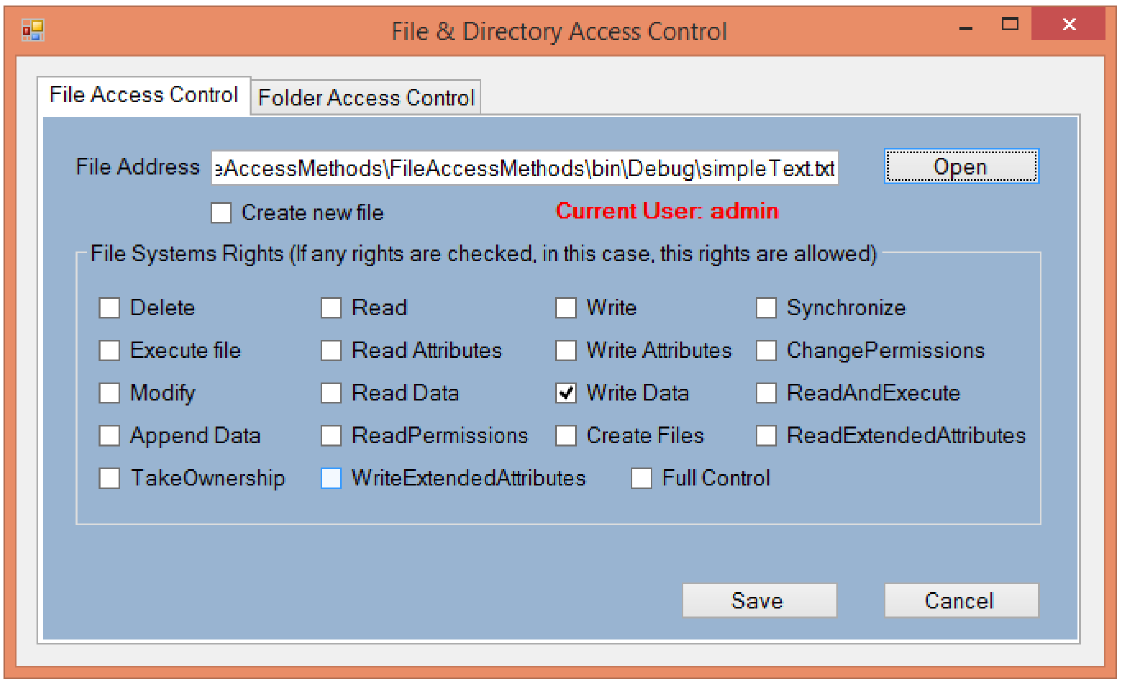
Current User: admin

File Systems Rights (If any rights are checked, in this case, this rights are allowed)

☐ Delete	☑ Read	☑ Write	☐ Synchronize
☑ Execute file	☑ Read Attributes	☑ Write Attributes	☐ ChangePermissions
☐ Modify	☑ Read Data	☑ Write Data	☑ ReadAndExecute
☑ Append Data	☑ ReadPermissions	☐ Create Files	☐ ReadExtendedAttributes
☐ TakeOwnership	☑ WriteExtendedAttributes	☐ Full Control	

Figure 2.4. Create new file with rights

For example, in Figure 2.4, creating new file with “read” and “write” rights is given. Generated new file is given as following.


File & Directory Access Control

File Access Control | Folder Access Control

File Address: ≥AccessMethods\FileAccessMethods\bin\Debug\simpleText.txt

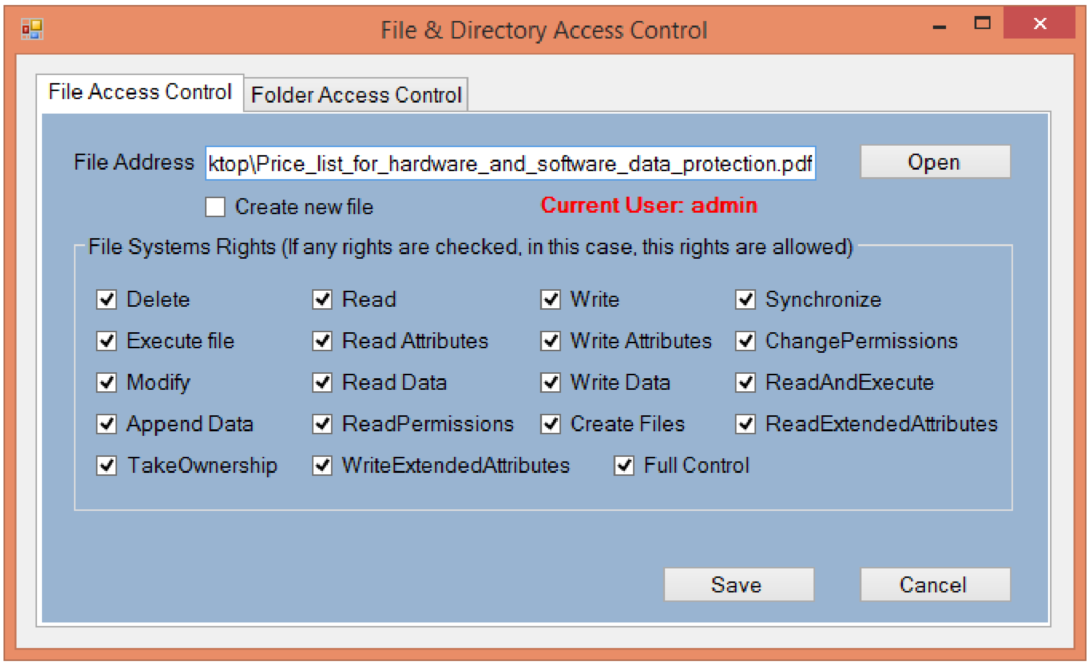
☐ Create new file

Current User: admin

File Systems Rights (If any rights are checked, in this case, this rights are allowed)

☐ Delete	☐ Read	☐ Write	☐ Synchronize
☐ Execute file	☐ Read Attributes	☐ Write Attributes	☐ ChangePermissions
☐ Modify	☐ Read Data	☑ Write Data	☐ ReadAndExecute
☐ Append Data	☐ ReadPermissions	☐ Create Files	☐ ReadExtendedAttributes
☐ TakeOwnership	☐ WriteExtendedAttributes	☐ Full Control	

Figure 2.5. Generated new file



File & Directory Access Control

File Access Control | Folder Access Control

File Address: ktop\Price_list_for_hardware_and_software_data_protection.pdf [Open]

☐ Create new file   **Current User: admin**

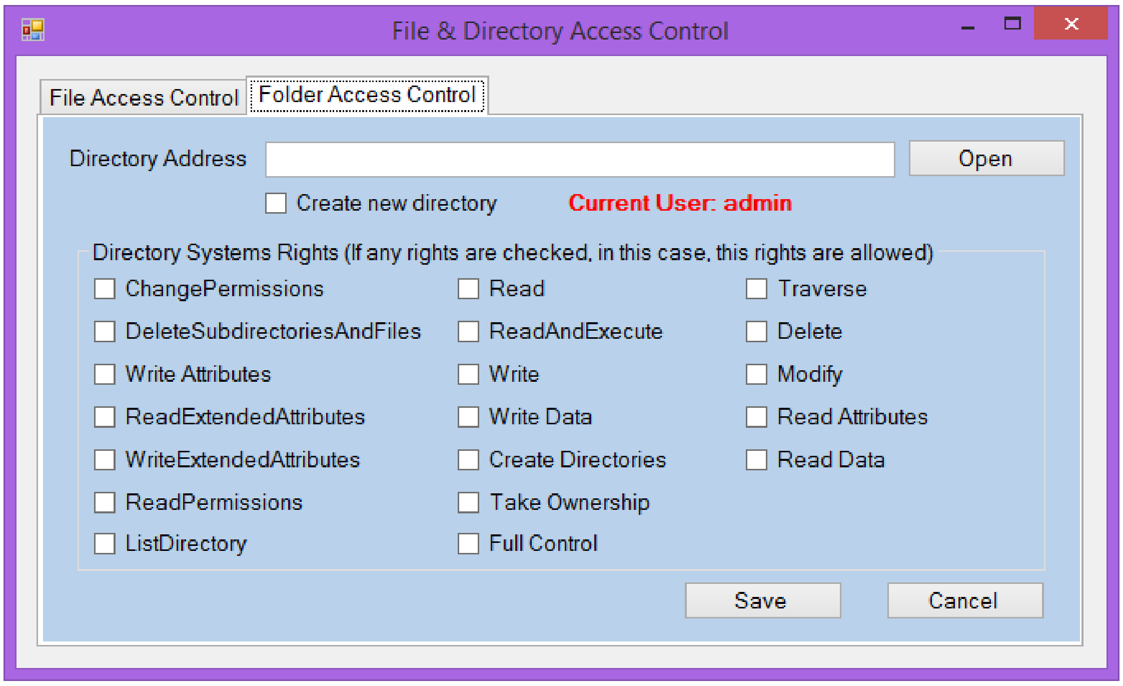
File Systems Rights (If any rights are checked, in this case, this rights are allowed)

☑ Delete	☑ Read	☑ Write	☑ Synchronize
☑ Execute file	☑ Read Attributes	☑ Write Attributes	☑ ChangePermissions
☑ Modify	☑ Read Data	☑ Write Data	☑ ReadAndExecute
☑ Append Data	☑ ReadPermissions	☑ Create Files	☑ ReadExtendedAttributes
☑ TakeOwnership	☑ WriteExtendedAttributes	☑ Full Control	

[Save] [Cancel]

Figure 2.6. Read exsistent file rights

Second part of program can do creating new folder with any rights and change existent folder rights. Second part of program illustrated as following:



File & Directory Access Control

File Access Control | Folder Access Control

Directory Address: [Open]

☐ Create new directory   **Current User: admin**

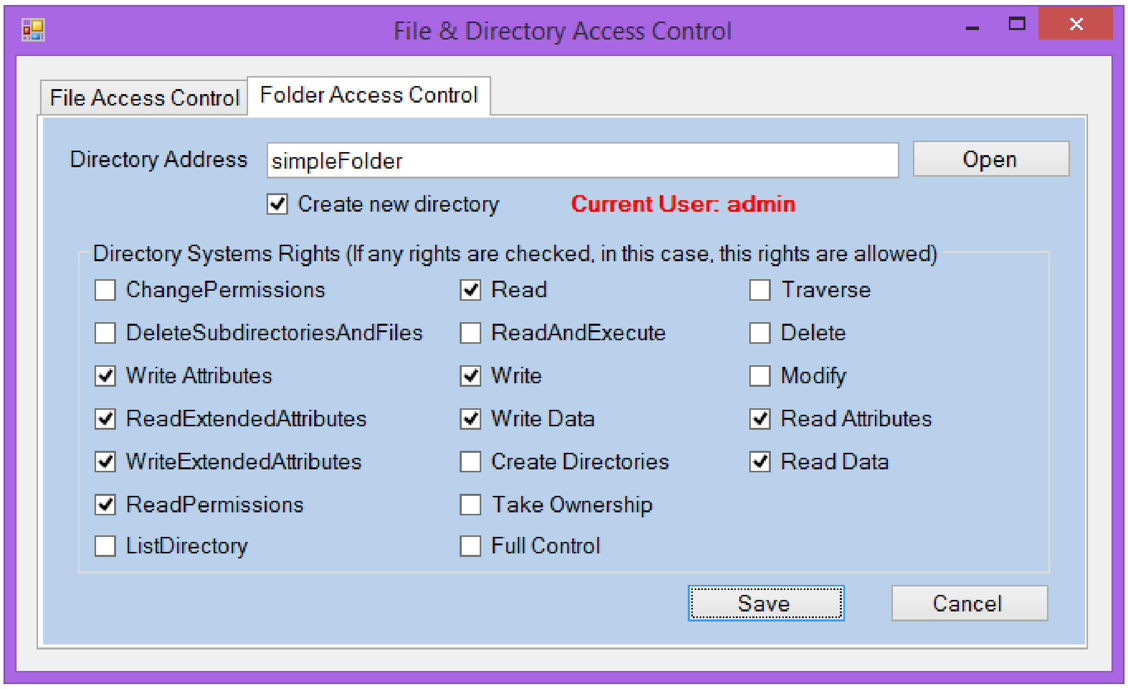
Directory Systems Rights (If any rights are checked, in this case, this rights are allowed)

☐ ChangePermissions	☐ Read	☐ Traverse
☐ DeleteSubdirectoriesAndFiles	☐ ReadAndExecute	☐ Delete
☐ Write Attributes	☐ Write	☐ Modify
☐ ReadExtendedAttributes	☐ Write Data	☐ Read Attributes
☐ WriteExtendedAttributes	☐ Create Directories	☐ Read Data
☐ ReadPermissions	☐ Take Ownership	
☐ ListDirectory	☐ Full Control	

[Save] [Cancel]

Figure 2.7. Directory Access Control rights

Directory rights is given in Table 2.7. In figure 2.8, creating new folder and changing existent folder rights process are given.



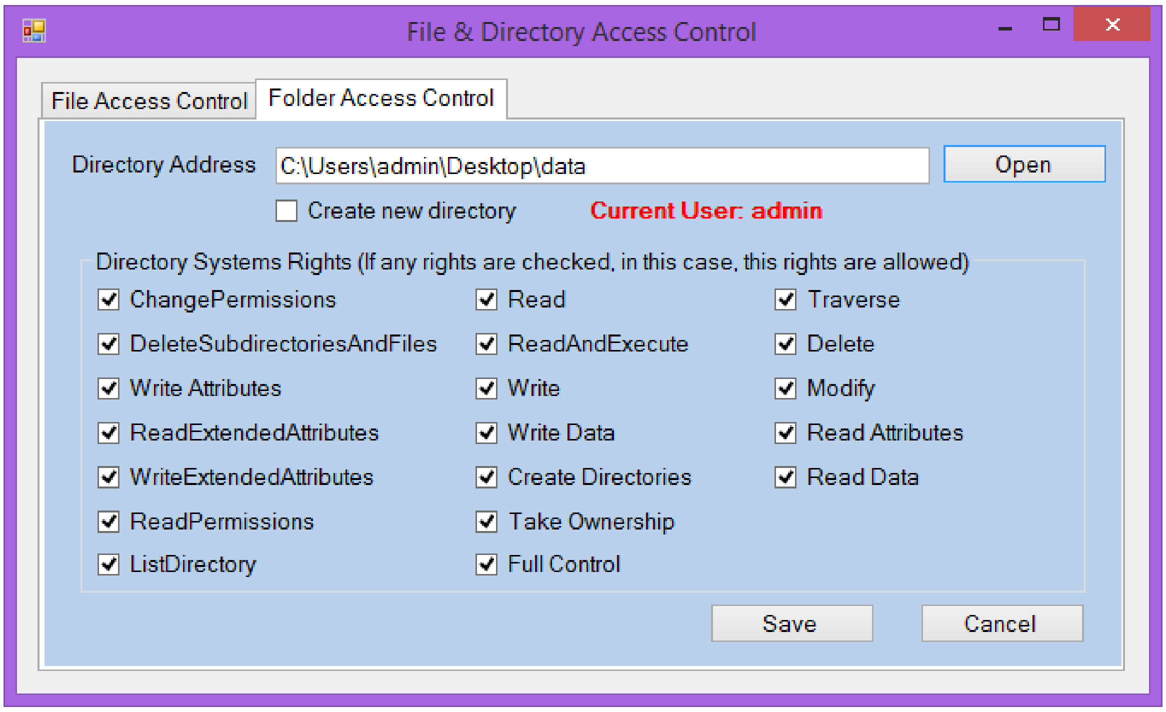
Directory Address: simpleFolder

[✓] Create new directory   **Current User: admin**

Directory Systems Rights (If any rights are checked, in this case, this rights are allowed)

[ ] ChangePermissions   [✓] Read   [ ] Traverse
[ ] DeleteSubdirectoriesAndFiles   [ ] ReadAndExecute   [ ] Delete
[✓] Write Attributes   [✓] Write   [ ] Modify
[✓] ReadExtendedAttributes   [✓] Write Data   [✓] Read Attributes
[✓] WriteExtendedAttributes   [ ] Create Directories   [✓] Read Data
[✓] ReadPermissions   [ ] Take Ownership
[ ] ListDirectory   [ ] Full Control

Figure 2.8. Create new folder



Directory Address: C:\Users\admin\Desktop\data

[ ] Create new directory   **Current User: admin**

Directory Systems Rights (If any rights are checked, in this case, this rights are allowed)

[✓] ChangePermissions   [✓] Read   [✓] Traverse
[✓] DeleteSubdirectoriesAndFiles   [✓] ReadAndExecute   [✓] Delete
[✓] Write Attributes   [✓] Write   [✓] Modify
[✓] ReadExtendedAttributes   [✓] Write Data   [✓] Read Attributes
[✓] WriteExtendedAttributes   [✓] Create Directories   [✓] Read Data
[✓] ReadPermissions   [✓] Take Ownership
[✓] ListDirectory   [✓] Full Control

Figure 2.9. Read existent folder rights

If program execute as system administrator level, in this case, you can create new files /folders or change rights existent files/ folders for all systems user.

3. LIFE SAFETY AND ECOLOGIYA

3.1. Ergonomics

Ergonomics (from Greek ergon - work and nomos - law) - a scientific discipline that studies human beings in its activities related to the use of machines. The goal of ergonomics - the optimization of working conditions in the "man-machine" (SChM). Ergonomics pertains to technology and human conditions of its operation. Ergonomic engineering is the most general indicator of the properties and other indicators of technology.

Ergonomics - the science of how people with their different physical characteristics and ways of functioning interact with the operating environment (equipment and machines they use). The goal of ergonomics is to provide comfort, efficiency and safety in the use of computers at the stage of development of keyboards, computer boards, furniture and others working to eliminate physical discomfort and health problems in the workplace. Due to the fact that more and more people are spending a lot of time in front of computer screens, scientists from many fields including anatomy, psychology and the environment, are involved in the study right from the point of view of ergonomics, work environment.

Most furniture manufacturers do not take into account the individual characteristics of the human body in the design of computer workstations. Construction ergonomically equipped places may require additional expenses. If your budget allows, buy ergonomically designed furniture such as chairs, shelves and tables that can be tailored to your individual physical data.

The objectives of ergonomics as an applied discipline are:

- ✓ Design the system "man-machine" that is, the distribution of functions between man and machine;
- ✓ design of the workspace so that the physical environment consistent with the characteristics of a person;
- ✓ designing environment in accordance with the requirements of the operator;

- ✓ Design work situations (working hours, rest breaks, etc.).

Psychology, as it follows from the above, is almost an integral part of ergonomics, the crucial problem of the organization of the system "man-machine" by:

- ✓ the distribution of functions between man and machine;
- ✓ analysis of the functions performed by a person in the "man-machine";
- ✓ Information system design, selection of a sensitive channel;
- ✓ design of controls;
- ✓ the design of workplaces;
- ✓ providing facilities maintenance vehicles;
- ✓ recruitment and training.

Accounting, ergonomic requirements must be implemented at all stages of the project, and includes:

- ✓ Development professiogram defining the goals and objectives of employment, its physiological characteristics, demands on man and technology.
- ✓ Analysis and specification of purpose, principles of operation and design technology, its characteristics for the purposes of employment.
- ✓ The distribution of functions between man and technology on the basis of quality assessment tasks man and machine and the overall efficiency of the system.
- ✓ Establishment of a sequence of operations performed by a man and a determination of the amount and form of presentation of information.
- ✓ Orientation assessment, time and accuracy requirements for human activities.

Based on the cited papers is determined by: the composition of experts, their function and organization of work, the composition of displays, controls, jobs and controls; arrangement of displays and controls, posting jobs in production areas.

Relationship with the environment and the working environment.

Workplace - this is the area in which work activity is performed performer or group of performers. Jobs can be individual or collective, universal, specialized and specific.

General requirements to be met in the design of workplaces, the following:

- ✓ Adequate working space for human rights;
- ✓ optimal working posture;
- ✓ adequate physical, visual and auditory communication between man and machine;
- ✓ Optimal placement of the workplace in the room;
- ✓ acceptable level of the factors of production conditions, the optimal placement of information and motor field;
- ✓ a means of protection against occupational hazards.

The construction zone should provide easy access and optimal motor field of the workplace and the optimal coverage of the information field workplace. Viewing angle to the horizontal should be 30-40 °.

Selecting the operating position must take into account the efforts expended by a person, the magnitude of movements, the need for movement, the tempo of operations. Selection of working postures should take into account human physiology, and workplace settings determined by the choice of body position at work (sitting, standing, alternately).

Jobs for work "sitting" are organized under light to moderate work, and with a heavy - working posture - "standing."

The design of the equipment and the work area must be such as to permit the regulation of individual elements to ensure optimum working position.

Equipment design should ensure that it complies anthropometric and biomechanical characteristics of a person on the basis of consideration of the dynamics change the size of heat when it is moved, the range of motion in the joints.

To take into account in the design of equipment anthropometric data should:

- ✓ determine which people, which is intended for equipment;
- ✓ select a group of anthropometric characteristics;
- ✓ establish the percentage of employees who must meet the equipment;
- ✓ Determine the boundaries of the interval size (forces) to be implemented in hardware.

In the design used anthropometric dimensions of the body, and take into account differences in body size between men and women, national, age, professional. To determine the boundaries of intervals, which take into account the percentage of the population uses a system pertseteley. Construction equipment must allow for at least 90% of consumers.

For operation in the "sitting" used various working seat. Distinguish workers seat. Long and short-term use. General requirements for the seats durable following: sitting posture should provide that reduces the statistical work of muscles, to create the conditions for the possibility of changing the working posture, no hindrances in the systems of the body, to ensure the free movement relative to the work surface, have adjustable parameters, have a semi-soft upholstery. For short-term use is recommended hard chairs and stools of various types.

With the increasing mechanization and automation of production processes are particularly important means of displaying information about the object. The widespread use of received information model, that is organized according to specific rules about the state of the control object. For information models must meet the following requirements:

- ✓ The content of the information model should be adequate to display the object management;
- ✓ Information Model should provide an optimal balance of information;
- ✓ Shape and composition information model must be consistent with the work process and the possibilities of man to receive the information.

Practice allows you to map out the sequence of the development of the information model: the definition of the tasks of the system, the sequence of their solutions and sources of information, an inventory control objects and their attributes, the distribution of objects in order of importance, the distribution of functions between automatic and man, the choice of a coding system objects and the drawing up of the overall composition of the model; definition of executive actions of man.

In the process of designing the information model defined location of the media in the workplace, selected label dimensions and layout. Display means are placed in the field of view with the optimum angles and observation areas. Dimensions are determined by observation of signs with the maximum accuracy and speed of perception and brightness character, the values of contrast, the use of color. Considered optimum brightness values, which provide maximum contrast sensitivity. Its value will be greater, the smaller the size of the object of discrimination. The optimum region the contrast value is equal to 60-90%.

In my eyes there is a certain inertia, which requires taking into account the exposure time and the visual signal timing for a sense of separateness signals one after the other. In most cases, the exposure time signal must be at least 50 ms. Each species has its own area of indicators used: Backlit indicators are used to display high-quality information that requires immediate attention of the operator, dial indicators are used to read measured values, integral indicators for combining information about several parameters.

A chip usually represents the structure and dynamics of the managed object. In some cases, a display for displaying information and the perception of its team of operators.

In the design of the workplace should be considered rules of economics movement: the work of the two arms of the movement must be simultaneous and symmetrical, the movement should be smooth and rounded, rhythmic and familiar to the employee. Equipment design should take into account the rules on speed and accuracy of movement of workers. For example, the most rapid movement to itself, in the horizontal speed of the hands more than the vertical, the accuracy of motion in the sitting position rather than standing up etc. The controls used in the workplace, must comply with the general requirements ergonogetiki: the direction of motion controls should correspond to the movement of the associated indicator, matching the location of the control sequence of the operator's ease of use, the creation of government agencies in the mechanical resistance, etc. Besides, for each type of pressure corresponds to a different area of use and the special requirements for size, shape, force, etc.

On the workstation operator-communicator (the operator in the control room) are generally used:

- ✓ Display means for individual use (imaging units, signaling devices, etc.);
- ✓ Controls and input (remote display, keyboard control, individual controls, and so on);
- ✓ Device information and communication (modem, telegraph and telephones)
- ✓ Documenting and storage device information (printing apparatus, recording, etc.);
- ✓ Auxiliary equipment (office equipment, storage for media, local lighting devices).

At the workstation must be provided information and structural compatibility of the technical means of anthropometric and psycho-physiological characteristics of the person.

If your work area should be taken into account not only the factors that reflect the experience, level of training, individual and personal property of the operators, signalers, but the factors that characterize the compliance forms, methods of presentation and data entry capabilities psychophysiological person.

When optimizing the procedures of interaction operators, signalers with the technical means in terms of automation ergonomic factors are the principal causing probability characteristics and hard work. These factors are sensitive to variations in the properties of the individual personality of the operator.

Workshop furniture should be comfortable to perform the planned work operations. The construction work of furniture: table, chair is essential for a healthy environment and a high-efficiency work. Workshop furniture is constructed taking into account the anthropometric data of human, technical, aesthetic, and economic factors.

Included working furniture has important industrial design chair as it determines the position worker and therefore energy consumption and the degree of fatigue. Operating the seat should have the required dimensions corresponding anthropometric rights and be mobile. The most comfortable seats and chairs with adjustable back tilt and height of the seat. By varying the height of the seat from the floor and back angle, you can find a position that most closely matches the labor process and the individual characteristics of the employee.

As a rule, all the surfaces of written and desktops should be at elbow height at working man's position. When you select a table height should be considered a man sitting during work or cost.

Inconvenient table height reduces efficiency and causes rapid fatigue. The lack of sufficient space for knees and feet causes constant irritation of the employee. Minimum working height of the table should be at least 725 mm. Practice shows that the average height for the working height of the desktop received 800 mm. For another employee growth can change the height of a chair or working status of its steps so that the distance from the object to the processing of eye working height was equal to about 450 mm.

Accommodation facilities and the operator's seat in the work area to provide convenient access to the major functional units and units of equipment for technical diagnostics, maintenance inspection and repair, and the ability to quickly take up and leave the work area, with the exception of accidental actuation of the controls and data entry; convenient posture and pose recreation. In addition, the layout must meet the requirements of integrity, and compactness of the technical and aesthetic expression of the working posture.

The display should be placed on a desk or table so that the viewing distance to the screen does not exceed 700 mm (optimal length 450 - 500 mm). Display adjustment should be located so that the angle between the center line of the screen and the horizontal view was 200. The horizontal viewing angle of the screen should not exceed 600. Remote display must be placed on a table or stand, so that the keypad height relative to the floor was 650 - 720 mm. When placing the console on a standard table height of 750 mm is necessary to use a chair with adjustable seat height (450 - 380 mm) and a footrest.

Document (blank) for operator input of data is recommended to have a distance 450 - 500 mm from the eyes of the operator, especially the left, the angle between the display screen and the document in the horizontal plane 40 should be 30 °. Angle of the keyboard should be equal to 15 °.

Screen display and keyboard instruments display panel should be arranged to drop the brightness of surfaces, independent of their location relative to the light source does not exceed 1:10 (recommended value 1: 3). At nominal values of the brightness of the screen image 50 - 100 cd/m2 luminance of the document should be 300 - 500 lux.

The workplace should be equipped in such a way that the movement of the worker would be the most rational, less tedious.

Device documentation and other infrequently used technical tools is recommended to have the right of an operator in the zone of maximum reach and means of communication to the left to release the right hand to take notes.

3.2. Psychophysiological load per person

In the section of psychophysiological stress, the most important is stress and fatigue.

Under stress is understood the emotional state that arises in response to all sorts of extreme exposure.

When stress ordinary emotions are replaced by anxiety, causing disturbances in physiological and psychological terms. This concept was introduced by Hans Selye to refer to non-specific response of the body to any adverse effects. His research showed that the various adverse factors - fatigue, fear, hurt, cold, pain, humiliation, and more in the body cause the same kind of comprehensive response regardless of what kind of stimulus acts on it at the moment. Moreover, these stimuli need not exist in reality. A man reacts not only to the actual danger and the threat or reminder of her.

Human behavior in situations of stress is different from the affective behavior. Under stress a person can usually control their emotions, to analyze the situation, make appropriate decisions.

Currently, depending on the stress factor identify different types of stress, including the pronounced physiological and psychological. Psychological stress, in turn, can be divided into information and emotional. If a person is unable to cope with the problem, do not have time to make the right decisions at the required rate with a high degree of responsibility, ie, when there is information overload may develop informational stress. Emotional stress arises in situations of danger, resentment, etc. Hans Selye identified in the development of stress three phases. The first stage - the alarm reaction - the mobilization phase defenses, which increases the stability with respect to a particular traumatic stress. In this case, there is a redistribution of body reserves: our primary objective is due to minor problems. The second step - the stabilization of parameters derived from the balance in the first phase, fixed at a new level. Externally, the behavior is not very different from the norm, as if everything is adjusted, but internally is overrun adaptive reserves. If the stressful situation persists, there comes the third stage - exhaustion, which can lead to a significant deterioration of health, various diseases, and in some cases death.

Stages of development of the state of stress in humans:

- ✓ build-up of tension;
- ✓ proper stress;
- ✓ Reduction of internal tension.

In its first phase duration is strictly individual. Some people "plant" for 2-3 minutes, and another increase in stress can take place over several days or even weeks. But in any case, the state and behavior of the person who is in stress, change pas' opposite sign. "

So, quiet reserved person becomes fussy and irritable, he may even become aggressive and violent. And the person in real life lively and agile, it becomes dark and taciturn.

In the first stage of stress weakens a person self-control: it gradually loses the ability to knowingly and intelligently regulate their own behavior.

The second stage of the stress state is manifested in the fact that man is a loss of effective self-conscious (full or partial). "The Wave" destructive stress damaging to the human psyche. He cannot remember what he said and did, or be aware of their actions, rather vague and incomplete. Many then noted that under stress they have done that in a tranquil setting would not have done. Usually all later regret it very much.

Also, like the first, the second phase in duration strictly individual - from several minutes or hours - several days or weeks. Having exhausted its energy resources (achieving higher voltage observed when a person feels the devastation, fatigue and

Stress conditions significantly affect the activities of man. People with different features of the nervous system to react differently to the same psychological burden. In some people there is increased activity, mobilization of forces, improve business performance. On the other hand, the stress can cause disruption of the sharp reduction of its effectiveness, and total inhibition of inactivity.

Human behavior in a stressful situation depends on many factors, but primarily on the psychological preparation of a person, which includes the ability to quickly assess the situation, the instantaneous orientation skills in unexpected circumstances, a strong-willed discipline and determination, experience, behavior in similar situations.

Methods of dealing with stress

Stress - the feeling that one experiences when she believes that it cannot effectively cope with the situation.

If the situation is causing stress depends on us, a more rational to focus on how to change it. If the situation is not up to us to accept and change your perception, your attitude to this situation.

In most situations, the stress goes through several stages.

1. Phase anxiety. This mobilization of energy resources of the body. Moderate stress useful in this step, it leads to higher efficiency.

2. Phase resistance. This is a balanced spending reserve. Outwardly, everything looks normal, people effectively solves the problems faced by them, but if this step takes too long and is not accompanied by relaxation, then, the body works hard.

3. Phase depletion (distress). Man feels weakness and fatigue, reduced performance, dramatically increases the risk of disease. Short time this can still fight at will, but then the only way to restore power - it is a solid rest.

One of the most common causes of stress - the contradiction between reality and perceptions of man.

Stress response is equally easy to run as real events, and existing only in our imagination. In psychology, this is called "the law of the emotional reality of the imagination." As psychologists have calculated, about 70% of our experiences come about events that do not exist in reality, but only in the imagination.

By the development of stress can lead not only negative but also positive life events. When something changes dramatically for the better, the body also reacts to this stress.

Usually, the fatigue is understood the reduction in the workability caused by previous work, which has a temporary character. If it occurs during mental activity, talk about mental fatigue. State of fatigue is manifested in changes physiological processes, reducing productivity and techno-economic indicators, change in mental status.

Psychologists say that the development of fatigue, the person has a special psyche, which is called the fatigue - a subjective reflection arising processes in the body, leading to fatigue. It appears long before the loss of productivity lies in the fact that there is a special experience painful stress and uncertainty. Manage feels that he could not continue to work properly. Thus there is a disorder of attention - in the development of fatigue, people are easily distracted, becomes sluggish, inactive, or, on the contrary, it appears chaotic mobility instability. There are disturbances in the sensory area - for fatigue changes work receptor, for example, there is a visual fatigue - decreased ability to process information coming through the visual analyzer, with the duration ion manual work is reduced tactile and kinesthetic sensitivity. Lead to abnormalities in the motor area: a slowing of movements, movements appear haste, rhythm disorder, weakening the accuracy and co-ordination of movements, de-automatization of movements. There are defects in memory and thinking, weakened the will, determination, endurance, self-control. With strong fatigue, somnolence.

Intensity of change depends on the depth of fatigue. For example, significant changes in mental status almost there and with fatigue all these changes is extremely pronounced.

Due to changes in the mental state of a number of physiologists has isolated 3 stages of fatigue. Stage 1: When her with the feeling of fatigue significantly, labor productivity is not reduced. Stage two - characterized by a significant reductiontion of labor and severe mental changes. The third stage, which some scholars regard as acute fatigue, accompanied by the expression experience fatigue.

Utation can be physical (musclemen) or neuropsychiatric (central). Both forms of fatigue combined with hard work, and they cannot be strictly separated from one another. Heavy physical work leads primarily to muscle fatigue, and enhanced mental functions or monotonous work is tiring ion of central origin. It should be a clear distinction between exhaustion and fatigue, caused need for sleep.

In addition, determine the primary Utition, which is developing quite rapidly at the beginning of the work shift and is a recognized cominsufficient consolidation of skills, it can be overcome in the process, resulting in an "second wind" - a significant increase workable STI. Secondary, slowly progressive fatigue actually tiring ion, which occurs after about 2.5-3 hours from the beginning of the work shift, and to remove it needs rest.

Fatigue or chronic fatigue - another type of fatigue. It is due to the lack of proper rest between each working day, is regarded as a pathological condition. Manifests the general decline in productivity, increased incidence, the slowdown in the cultural and technical level and skills of running, decreased creativity and mental capacity, changes in the cardiovascular system.

According to K.K Platonov are four degrees of fatigue restarting, lung, and severe, each of which requires appropriate methods of struggle. Therefore, to relieve fatigue suf beginning precisely regulate the regime of work and rest. Mild fatigue optionally sary to wait for release and use it effectively. In marked overworked SRI urgent needs rest, better organized. In severe pereutomtion to treatment.

Conclusion

Access control systems provide the essential services of authorization, identification and authentication (I&A), access approval, and accountability where:

- ✓ authorization specifies what a subject can do;
- ✓ identification and authentication ensure that only legitimate subjects can log on to a system;
- ✓ access approval grants access during operations, by association of users with the resources that they are allowed to access, based on the authorization policy;
- ✓ Accountability identifies what a subject (or all subjects associated with a user) did.

As result of this bachelor diploma, followings are taken:

- ✓ Researching file systems, file attributes, file systems are done;
- ✓ File access methods, namely, indexed sequential, sequential and random methods are analyzing;
- ✓ Access control systems, consistent of access control, identification and authentication systems in access control, related methods of attack in access control are researched;
- ✓ Access control models are analyzed;
- ✓ Access control techniques, mandatory access control, discrete access control and no discrete access control are analyzed;
- ✓ File/ Folder access control methods in C# programming languages are researched and created “Control access of generated file objects” software.

References

1. “Ахборотлаштириш тўғрисида”ги Ўзбекистон Республикаси қонуни, 11-декабр 2003-йил.
2. “Ўзбекистон Республикасида ахборотни криптографик муҳофаза қилишни ташкил этиш чора-тадбирлари” тўғрисида Ўзбекистон Республикаси президентининг қарори.
3. “Миллий Ахборот-коммуникация тизимларининг компютер хавфсизлигини таъминлаш борасидаги қўшимча чора-тадбирлар тўғрисида”ги Ўзбекистон Республикаси Президентининг қарори, 5 - сентабр 2005 – йил.
4. File Organization. Binnur Kurt. Istanbul Technical University. Computer Engineering Department.
5. Assessment of Access Control Systems. National Institute of Standards and Technology. Interagency Report 7316. Vincent C.Hu, David F.Ferraiolo, D.Rick Kuhn.
6. http://en.wikipedia.org/wiki/File_system
7. Indexed Sequential Access Method (ISAM). Peter Abel.
8. An Improved Administration Method on Role-Based Access Control in the Enterprise Environment. SEJONG OH AND SEOG PARK. Department of Computer Science Sogang University Seoul 121-742, Korea. JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 17, 921-944 (2001).
9. File-System Interface. Silberschatz, Operating System Concepts – 8th Edition Galvin and Gagne ©2009.
10. Mandatory Access Control. Fred B. Schneider. Draft of March 31, 2014 Copyright 2014.
11. Extending Role Based Access Control. A SANS Whitepaper. Written by J. Michael Butler, Information Security Consultant – April 2011.
12. Methods for Access Control: Advances and Limitations. Ryan AusankaCrues, Harvey Mudd College. 301 Platt Blvd Claremont, California.

13. Access Control: Policies, Models, and Mechanisms. Pierangela Samarati and Sabrina De Capitani di Vimercati. Dipartimento di Tecnologie dell’Informazione – Universit`a di Milano. Via Bramante 65 – 26013 - Crema (CR) Italy.
14. http://www.itsec.ru/articles2/Inf_security/novyy-podhod-k-realizatsii-kontrolya-i-razgranicheniya-prav-dostupa-k-dannym-v-informatsionnyh-sistemah/
15. http://en.wikipedia.org/wiki/Sequential_access
16. https://en.wikipedia.org/wiki/Random_access
17. https://en.wikipedia.org/wiki/File_attribute
18. http://www.worldbestlearningcenter.com/index_files/csharp-file-stream-sequential.htm
19. http://www.worldbestlearningcenter.com/index_files/csharp-file-stream-random.htm
20. Access Control Methodologies
21. https://msdn.microsoft.com/en-us/library/system.security.accesscontrol.filesystemrights(v=vs.110).aspx