Protection and Security

- The whole world in interconnected with the advent of internet technology.
- It has taken human life into much higher levels of sophistication and ease. But the computer systems are faced with new threats through internet.
- So Protection and Security of computer systems are becoming important as the world is closely interconnected.
- Operating System plays a major role in providing protection of user data and system resources as it is responsible for serving computer users locally and remotely through the Internet.
- With respect to OS, protection may be viewed as its internal requirement whereas Security deals with protecting a complete system from threats arising from the external environment in which the computer resides.

Protection

- What is Protection?
- The role of protection in a computer system is to provide a mechanism for the enforcement of the policies governing resource use.
- Protection prevents accidental or intentional misuse of a system.
- Primarily the system must have a mechanism to identify users who are eligible to use the computer.
- It must have a means to authenticate eligible users who attempt to use the system.

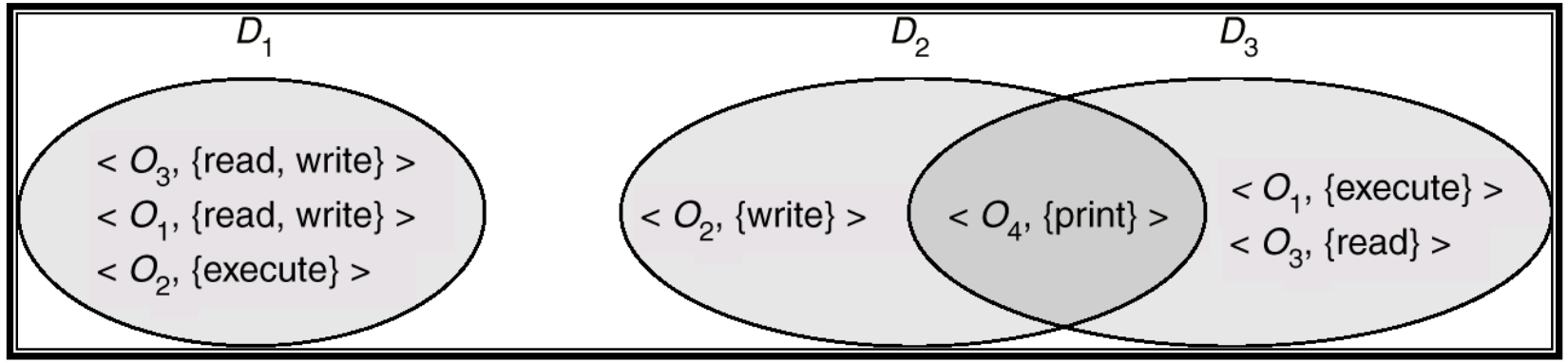
Protection Domain and its structure

- A computer can be viewed as a collection of processes and objects both hardware and software.
- A process can have access to those objects it needs to accomplish its task in the modes for which it needs access and during the time frame when it needs access.

Domain Structure

- Access-right = <*object-name*, *rights-set*>
  where *rights-set* is a subset of all valid operations that can be performed on the object.

- Domain = set of access-rights



The three aspects to a protection mechanism are as follows:

- Authentication: identify a responsible party (principal) behind each action.
- Authorization: determine which principals are allowed to perform which actions.
- Access enforcement: combine authentication and authorization to control access.

Authentication

- A secret piece of information used to establish identity of a user. Typically it is done with passwords and must not be stored in an encrypted form.
- The other form of authentication is using key which should not be forgeable or copyable. Once authentication is complete, the identity of the principal must be protected from tampering, since other parts of the system will rely on it.
- Once the log in process is over, the user id is associated with every process executed under that login: each process inherits the user id from its parent.

Authorization

- Its goal is to determine which principals can perform which operations on which objects.
- Logically, authorization information is represented as an access matrix:
  - One row per principal.
  - One column per object.
- Each entry indicates what that principal can do to that object.

Access Matrix

- View protection as a matrix (*access matrix*)
- Rows represent domains
- Columns represent objects
- *Access(i, j)* is the set of operations that a process executing in Domain_i can invoke on Object_j

Access Matrix

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

Use of Access Matrix

- If a process in Domain D_i tries to do "op" on object O_j, then "op" must be in the access matrix.

- Can be expanded to dynamic protection.
  - Operations to add, delete access rights.
  - Special access rights:
    - *owner of* O_i
    - *copy op from* O_i *to* O_j
    - *control –* D_i *can modify* D_j *access rights*
    - *transfer – switch from domain* D_i *to* D_j

Use of Access Matrix (Cont.)

- Access matrix design separates mechanism from policy.
  - Mechanism
    - Operating system provides access-matrix + rules.
    - If ensures that the matrix is only manipulated by authorized agents and that rules are strictly enforced.
  - Policy
    - User dictates policy.
    - Who can access what object and in what mode.

Implementation of Access Matrix

The access matrix is usually large and sparse. Different ways of implementing access matrix are:

- store the matrix by columns or by rows
- store only the non-empty elements
- Storing the matrix by columns corresponding to access control lists
- Storing the matrix by rows corresponding to capabilities

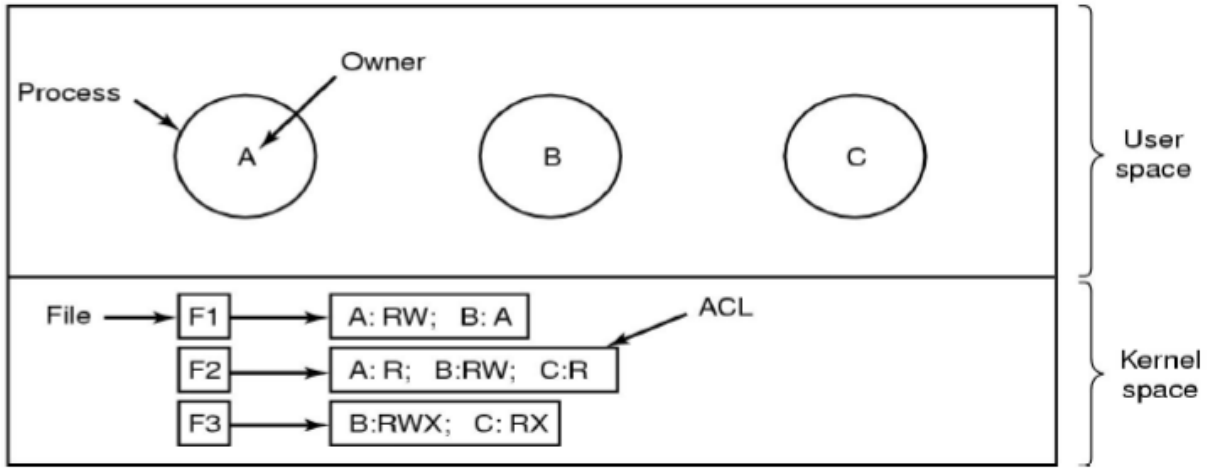
The simplest approach to implement access matrix is using one big global table or using linked lists with entries.

As the size of access matrix increases the global table/ list will be large and so it is stored in the following two compressed forms.

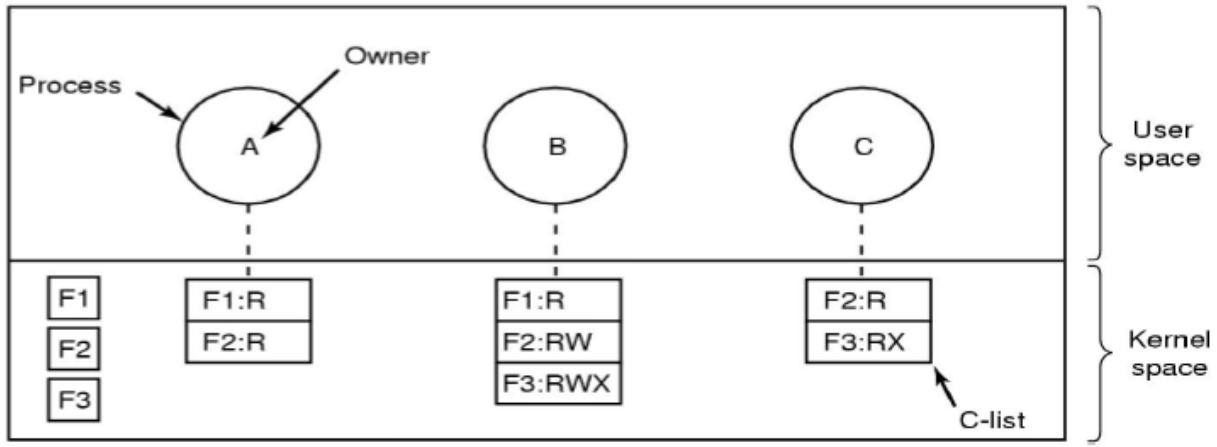
- Access control lists
- Capabilities

Access control lists (ACLs):

- The information stored is about the users who are allowed to perform operations on each object. It is a list of < user, privilege> pairs
- The users may be single or group and the operations may be read, write or execute. ACLs are simple and are used in almost all file systems.



- Capabilities:
- They are organized by rows and have the information about each user and the objects that can have access by them and in what ways.
- Capabilities store a list of pairs with each user
- Capabilities have been used in experimental systems attempting to be very secure.



- Revocation of Access Rights:
- In a dynamic protection system, sometimes there is a need to revoke access rights to objects shared by different users.
- The revocation may be Immediate versus Delayed, Selective versus General, Partial versus Total, and Temporary versus Permanent.
- *Access List* – Delete access rights from access list.
  - Simple
  - Immediate

Schemes that implements revocation for capabilities include the following:

- Reacquisition: Periodically, capabilities are deleted from each domain.
- Back pointers: A list of pointers is maintained with each object, pointing to all capabilities associated with that object.
- Indirection: The capabilities point indirectly, not directly, to the objects.
- Keys: A key is a unique bit pattern that can be associated with a capability.

Security

What is security?

- It is the quality or state of being secure i.e. Protecting the system from its users and preventing the unauthorized disclosure or modification of data.

Need for security

- Security is needed to prevent illegal users from using the system, to make unauthorized actions impossible, to confine errors to the immediate contexts of their occurrences and to detect and correct damage before it spreads to other parts of the system.

Computer security is about making its software behave appropriately.

Three factors that make this task difficult are

- Complexity: Software systems acquire complexity easily as software with more lines may contain more bugs that may lead to undesirable behaviour of the system.
- Extensibility: Most modern computing systems are extendable and evolve over time. Most software accept updates or extensions from remote hosts mainly through mobile codes which act as a carrier for attacks.
- Connectivity: The growing trend of Internet connectivity renders computers vulnerable to remote attacks.

Categories of security

- Based on Usage of software
  - Software security is related to its design construction and testing.
  - Application security is concerned with protecting the software during operation.

- Based on the perspective of the system
  - Internal security
    - Ensuring security within the system
    - Deals with the security issues related to the behavior of the active entities within the system.
  - Boundary Security
    - Ensuring security along the boundary of the system
    - Deals who can enter the system from outside

- Based on Undesirable behaviours
  - Malicious behaviour attempts to read or destroy sensitive data or disrupt the system operation intentionally, usually initiated from outside.
  - Incidental behaviours need not be intentional and usually arise from within the system but the consequences are as serious as of malicious behaviour and may be due to hardware malfunction or undetected errors from software or natural disaster damage etc.

- Based on resources
  - Hardware security – Security attacks having resources such as processors, main memory, secondary storage devices, communication lines and devices.
  - Software security- Data and service (process) are subject to security risk. Confidentiality and integrity of data may be compromised. The service/process may behave in certain way, proliferated or destroyed.

- Based on User’s point of view
  - Unauthorized use- Gaining access to the system or account using another person’s system or pirated version of software
  - Denial of Service- Intentional prevention of authorized users obtaining their services on time.

- Based on source of attack
  - Attacker within system may be an unauthorized user or a malicious process
  - Attacker may be from outside through internet using legitimate channels or through illegitimate channels.

- A secure system should not allow:
  - unauthorized reading of information
  - unauthorized modification of information
  - unauthorized destruction of data
  - unauthorized use of resources
  - denial of service for authorized uses

- The key aspect of security is identifying the user to the system. Authentication is establishing identity and authorization is establishing rights of an identity.
- Authentication is based on one or more of three items:
  - user possession (a key or card),
  - user knowledge (a user identifier and password), and
  - a user attribute (fingerprint, retina pattern, or signature).

Authentication

- User identity most often established through *passwords*, can be considered a special case of either keys or capabilities.

- Passwords must be kept secret.
  - Frequent change of passwords.
  - Use of “non-guessable” passwords.
  - Log all invalid access attempts.

- Passwords may also either be encrypted or allowed to be used only once.

Threats - Program Threats

- In an environment where a program written by one user may be used by another user, there is an opportunity for misuse, which may result in unexpected behavior.

Common methods

Trojan horses

Trap doors

Stack and Buffer Overflow

Trojan Horse:

- Code segment that misuses its environment.
- Exploits mechanisms for allowing programs written by users to be executed by other users.
  The Trojan-horse problem is exacerbated by long search paths. The search path lists the set of directories to search when an ambiguous program name is given. The path is searched for a file of that name and the file is executed. All the directories in the search path must be secure, or a Trojan horse could be slipped into the user's path and executed accidentally.

Trap Door:

- The designer of a program or system might leave a hole in the software that only Operating System is capable of using.
- Specific user identifier or password that circumvents normal security procedures.
- A clever trap door could be included in a compiler. The compiler could generate standard object code as well as a trap door, regardless of the source code being compiled.

Stack and Buffer Overflow

- Exploits a bug in a program (overflow either the stack or memory buffers.)

System Threats

- Worms – use spawn mechanism; standalone program
- Internet worm
  - Exploited UNIX networking features (remote access) and bugs in *finger* and *sendmail* programs.
  - Grappling hook program uploaded main worm program.
- Viruses – fragment of code embedded in a legitimate program.
- Denial of Service
  - Overload the targeted computer preventing it from doing any sueful work.

- Worms
- A worm is a process that uses the spawn mechanism that repeatedly reduce system performance.
- The worm spawns copies of itself, using up system resources and may lock systems resources to all other processes. Since it may reproduce itself among systems and thus shut down the entire network.
- The worm was made up of two programs a grappling hook (also called bootstrap or vector) program and the main program.
- The grappling hook consists of 99 lines of C code compiled and run on each machine it accessed. The grappling hook is connected to the machine where it originates and upload a copy of the main worm onto the "hooked" system.
- The main program proceeded to search for other machines to which the newly infected system could connect easily.

Viruses

- Another form of computer attack is a virus. Like worms, viruses are designed to spread into other programs and can completely destroy a system, i.e. modifying or destroying files, causing system crashes and program malfunctions.
- A virus is a fragment of code embedded in a legitimate program whereas a worm is structured as a complete, standalone program.
- Viruses are a major problem especially for single user systems. They are usually spread by users downloading viral programs from public bulletin boards or using secondary devices such as floppy disks, pen drives, etc. containing an infection.
- The best protection against computer viruses is prevention, or the practice of Safe Computing.

Threat Monitoring

- Check for suspicious patterns of activity – i.e., several incorrect password attempts may signal password guessing.
- Audit log – records the time, user, and type of all accesses to an object; useful for recovery from a violation and developing better security measures.
- Scan the system periodically for security holes; done when the computer is relatively unused.

Threat Monitoring

- Check for:
  - Short or easy-to-guess passwords
  - Unauthorized set-uid programs
  - Unauthorized programs in system directories
  - Unexpected long-running processes
  - Improper directory protections
  - Improper protections on system data files
  - Dangerous entries in the program search path (Trojan horse)
  - Changes to system programs: monitor checksum values

FireWall

- A firewall is placed between trusted and untrusted hosts.
- The firewall limits network access between these two security domains.

Intrusion Detection

- Detect attempts to intrude into computer systems.
- Detection methods:
  - Auditing and logging.
  - Tripwire (UNIX software that checks if certain files and directories have been altered – I.e. password files)
- System call monitoring

Encryption

- Encrypt clear text into cipher text.
- Properties of good encryption technique:
  - Relatively simple for authorized users to encrypt and decrypt data.
  - Encryption scheme depends not on the secrecy of the algorithm but on a parameter of the algorithm called the encryption key.
  - Extremely difficult for an intruder to determine the encryption key.
- *Data Encryption Standard* substitutes characters and rearranges their order on the basis of an encryption key provided to authorized users via a secure mechanism.

Encryption

- Public-key encryption based on each user having two keys:
  - public key – published key used to encrypt data.
  - private key – key known only to individual user used to decrypt data.
- Must be an encryption scheme that can be made public without making it easy to figure out the decryption scheme.
  - Efficient algorithm for testing whether or not a number is prime.
  - No efficient algorithm is know for finding the prime factors of a number.