OBJECTIVES/TOPICS

- Implementing Security Defenses.
- Security Policy.
- Vulnerability Assessment.
- Intrusion Prevention.
- Auditing, Accounting and Logging.
- Firewall to protect systems networks.
- Protection. Principles of protection.
- Protection Rings.
- Domain Protection.
- Access Matrix

IMPLEMENTING SECURITY DEFENSES.

- Just as there are myriad threats to system and network security, there are many security solutions.
    - The solutions range from improved user education, through technology, to writing better software.

- Most security professionals subscribe to the theory of defense in depth, which states that more layers of defense are better than fewer layers.
- Of course, this theory applies to any kind of security.
    - Consider the security of a house without a door lock, with a door lock, and with a lock and an alarm. In this section

SECURITY POLICY.

- The first step toward improving the security of any aspect of computing is to have a security policy.

- Policies vary widely but generally include a statement of what is being secured.
    - For example, a policy might state:
        - that all outside accessible applications must have a code review before being deployed, or
        - that users should not share their passwords, or
        - that all connection points between a company and the outside must have port scans run every six months.

SECURITY POLICY.

- Without a policy in place, it's impossible for users to know what is permissible, what is required,& what is not allowed.

- The policy is a road map to security, & if a site is trying to move from less secure to more secure, it needs a map to know how to get there.

- Once the security policy is in place, the people it affects should know it well. It should be their guide.
  - The policy should also be a living document that is reviewed & updated periodically to ensure that it is still pertinent & still followed.

VULNERABILITY ASSESSMENT.

- How can we determine whether a security policy has been correctly implemented?

- The best way is to execute a vulnerability assessment
  - Such assessments can cover broad ground, from social engineering through risk assessment to port scans.

  - Risk assessment, e.g., attempts to value the assets of the entity in question (a program, a management team, a system, or a facility) and determine the odds that a security incident will affect the entity and decrease its value.

VULNERABILITY ASSESSMENT.

- When the odds of suffering a loss and the amount of the potential loss are known, a value can be placed on trying to secure the entity.

- The core activity of most vulnerability assessments is a penetration test, in which the entity is scanned for known vulnerabilities.
  - Vulnerability scans typically are done at times when computer use is relatively low, to minimize their impact.
  - When appropriate, they are done on test systems rather than production systems, because they can induce unhappy behavior from the target systems or network devices.

VULNERABILITY ASSESSMENT.

- A system scan can check a variety of aspects of the system:
  - Short or easy-to-guess passwords
  - Unauthorized privileged programs, such as setuid programs
  - Unauthorized programs in system directories
  - Unexpectedly long-running processes
  - Improper directory protections on user and system directories
  - Improper protections on system data files, such as the password file, device files, or the operating-system kernel itself
  - Dangerous entries in the program search path, such as the current directory and any easily-written directories such as /tmp
  - Changes to system programs detected with checksum values
  - Unexpected or hidden network daemons

VULNERABILITY ASSESSMENT.

- A system scan can check a variety of aspects of the system:
  - Short or easy-to-guess passwords
  - Unauthorized privileged programs, such as setuid programs
  - Unauthorized programs in system directories
  - Unexpectedly long-running processes
  - Improper directory protections on user and system directories
  - Improper protections on system data files, such as the password file, device files, or the operating-system kernel itself
  - Dangerous entries in the program search path, such as the current directory and any easily-written directories such as /tmp
  - Changes to system programs detected with checksum values
  - Unexpected or hidden network daemons

INTRUSION PREVENTION.

- Securing systems and facilities is intimately linked to intrusion detection and prevention.

- Intrusion prevention, as its name suggests, strives to detect attempted or successful intrusions into computer systems and to initiate appropriate responses to the intrusions.

- Intrusion prevention encompasses a wide array of techniques that vary on a number of axes, including the following:

INTRUSION PREVENTION.

- Securing systems and facilities is intimately linked to intrusion detection and prevention.

- Intrusion prevention, as its name suggests, strives to detect attempted or successful intrusions into computer systems and to initiate appropriate responses to the intrusions.

INTRUSION PREVENTION.

- Intrusion prevention encompasses a wide array of techniques that vary on a number of axes, including the following:
  - The time at which detection occurs. Detection can occur in real time (while the intrusion is occurring) or after the fact.
  - The types of inputs examined to detect intrusive activity. These may include user-shell commands, process system calls etc.
  - The range of response capabilities. Simple forms of response include alerting an administrator to the potential intrusion or somehow halting the potentially intrusive activity

INTRUSION PREVENTION.

- In signature-based detection, system input or network traffic is examined for specific behavior patterns (or signatures) known to indicate attacks.
- In anomaly detection, attempts through various techniques to detect anomalous behavior within computer systems

  - Signature-based detection attempts to characterize dangerous behaviors and to detect when one of these behaviors occurs.
  - Anomaly detection attempts to characterize normal (or nondangerous) behaviors and to detect when something other than these behaviors occurs.

INTRUSION PREVENTION.

- anomaly detection can find previously unknown methods of intrusion (so-called zero-day attacks).

- Signature-based detection, in contrast, will identify only known attacks that can be codified in a recognizable pattern.

AUDITING, ACCOUNTING AND LOGGING.

- Auditing, accounting, and logging can decrease system performance, but they are useful in several areas, including security.
- Logging can be general or specific.
- All system-call executions can be logged for analysis of program behavior (or misbehavior).
- More typically, suspicious events are logged.

AUDITING, ACCOUNTING AND LOGGING.

- Authentication failures and authorization failures can tell us quite a lot about break-in attempts.

- Accounting is another potential tool in a security administrator's kit.
  - It can be used to find performance changes, which in turn can reveal security problems.
  - One of the early UNIX computer break-ins was detected by Cliff Stoll when he was examining accounting logs and spotted an anomaly.

FIREWALL TO PROTECT SYSTEMS NETWORKS.

- How does a trusted computer can be connected safely to an untrustworthy network?
  - 1 solution is the use of a firewall to separate trusted & untrusted systems.

- A firewall is a computer, appliance, process, or router that sits between the trusted and the untrusted

- A network firewall limits network access between the multiple security domains & monitors and logs all connections.

FIREWALL TO PROTECT SYSTEMS NETWORKS.

- It can also limit connections based on source or destination address, source or destination port, or direction of the connection or instance, web servers use HTTP to communicate with web browsers.

  - A firewall therefore may allow only HTTP to pass from all hosts outside the firewall to the web server within the firewall.
  - The first worm, the Morris Internet worm, used the *finger* protocol to break into computers, so *finger* would not be allowed to pass, for example.

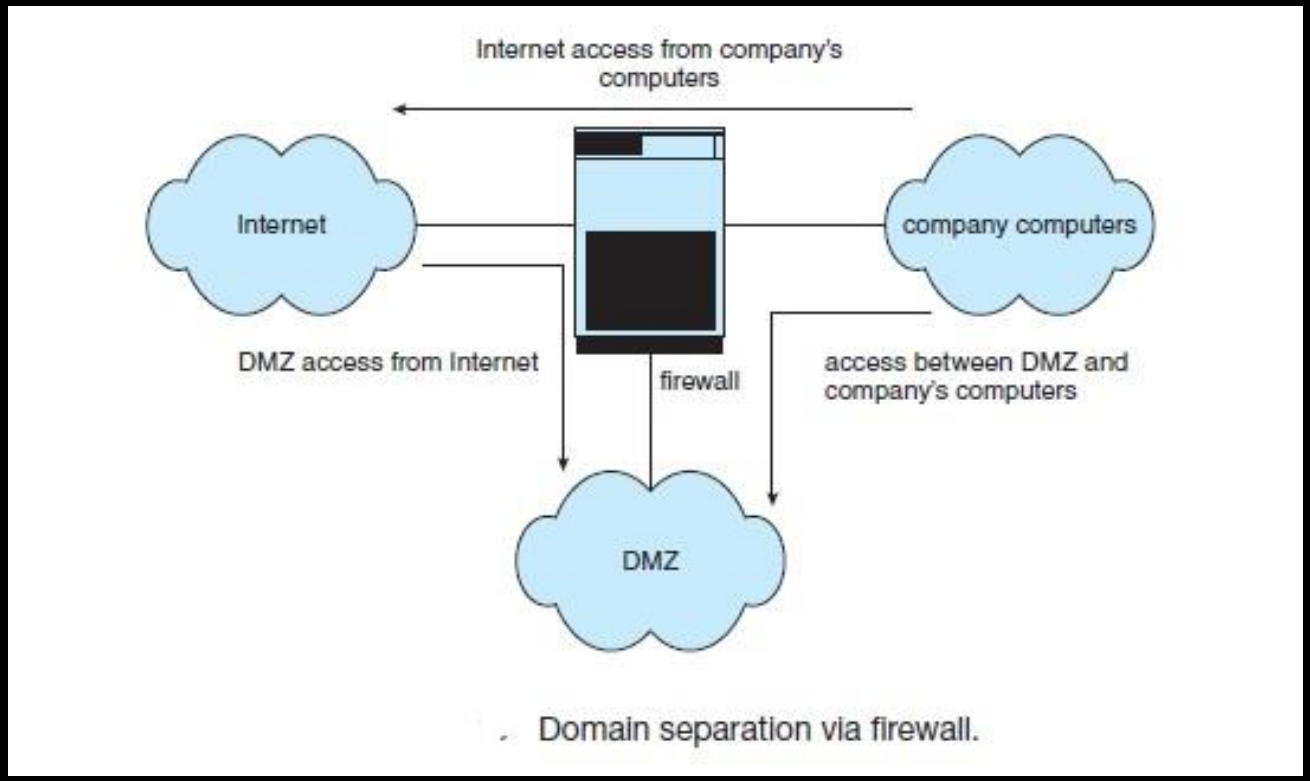
FIREWALL TO PROTECT SYSTEMS NETWORKS.

- A firewall can separate a network into multiple domains.
  - A common implementation has:
    - the Internet as the untrusted domain;
    - a semitrusted and semisecure network, called the demilitarized zone (DMZ), as another domain; &
    - a company's computers as a 3rd domain

- Connections are allowed from:
  - the Internet to the DMZ computers &
  - from the company computers to the Internet
- but connections are not allowed from the Internet or DM computers to the company computers.

FIREWALL TO PROTECT SYSTEMS NETWORKS.

- As shown in the diagram below



Domain separation via firewall.

FIREWALL TO PROTECT SYSTEMS NETWORKS.

- In addition to the most common network firewalls, there are other, newer kinds of firewalls:
  - 1. A personal firewall is a software layer either included with the operating system or added as an application.
    - Rather than limiting communication between security domains, it limits communication to (and possibly from) a given host.
    - A user could add a personal firewall to her PC so that a Trojan horse would be denied access to the network to which the PC is connected, for example.

FIREWALL TO PROTECT SYSTEMS NETWORKS.

- 2. An application proxy fire wall understands the protocols that applications speak across the network.
  - For example, SMTP is used for mail transfer.
  - An application proxy accepts a connection just as an SMTP server would and then initiates a connection to the original destination SMTP server.

- 3. An XML firewall , e.g. , has the specific purpose of analyzing XML traffic and blocking disallowed or malformed XML.

- 3. System-call firewalls sit between applications and the kernel, monitoring system-call execution.

PROTECTION.

- As computer systems have become more sophisticated and pervasive in their applications, the need to protect their integrity has also grown.

- Protection was originally conceived as an adjunct to multiprogramming operating systems, so that untrustworthy users might safely share a common logical name space, such as a directory of files, or a common physical name space, such as memory

PROTECTION.

- Modern protection concepts have evolved to increase the reliability of any complex system that makes use of shared resources and is connected to insecure communications platforms such as the Internet.

- We need to provide protection for several reasons
  - The most obvious is the need to prevent the mischievous, intentional violation of an access restriction by a user.
  - Of more general importance, however, is the need to ensure that each process in a system uses system resources only in ways consistent with stated policies.

PROTECTION.

- The role of protection in a computer system is to provide a mechanism for the enforcement of the policies governing resource use.

- Note that mechanisms are distinct from policies.
  - Mechanisms determine how something will be done; policies decide what will be done.
  - The separation of policy & mechanism is important for flexibility.
  - Policies are likely to change from place to place or time to time. In the worst case, every change in policy would require a change in the underlying mechanism.

PRINCIPLES OF PROTECTION.

- Frequently, a guiding principle can be used throughout a project, such as the design of an operating system.

- Following this principle simplifies design decisions and keeps the system consistent and easy to understand.

- A key, time-tested guiding principle for protection is the principle of least privilege
  - this principle dictates that programs, users, & even systems be given just enough privileges to perform their tasks.

PRINCIPLES OF PROTECTION.

- Consider one of the tenets of UNIX—that a user should not run as root. (In UNIX, only the root user can execute privileged commands.)

- Observing the principle of least privilege would give the system a chance to mitigate the attack—if malicious code cannot obtain root privileges, there is a chance that adequately defined permissions may block all, or at least some, of the damaging operations.

PRINCIPLES OF PROTECTION.

- Another important principle, often seen as a derivative of the principle of least privilege, is compartmentalization.
  - Compartmentalization is the process of protecting each individual system component through the use of specific permissions and access restrictions.
  - Then, if a component is subverted, another line of defense will "kick in" and keep the attacker from compromising the system any further.
  - Compartmentalization is implemented in many forms—from network demilitarized zones (DMZs) through virtualization.

PRINCIPLES OF PROTECTION.

- The careful use of access restrictions can help make a system more secure and can also be beneficial in producing an audit trail, which tracks divergences from allowed accesses.
  - An audit trail is a hard record in the system logs.
  - If monitored closely, it can reveal early warnings of an attack or (if its integrity is maintained despite an attack) provide clues as to which attack vectors were used, as well as accurately assess the damage caused.

PRINCIPLES OF PROTECTION.

- The careful use of access restrictions can help make a system more secure and can also be beneficial in producing an audit trail, which tracks divergences from allowed accesses.
  - An audit trail is a hard record in the system logs.
  - If monitored closely, it can reveal early warnings of an attack or (if its integrity is maintained despite an attack) provide clues as to which attack vectors were used, as well as accurately assess the damage caused.

PROTECTION RINGS.

- the main component of modern operating systems is the kernel, which manages access to system resources and hardware.

- The kernel, by definition, is a trusted and privileged component and therefore must run with a higher level of privileges than user processes.

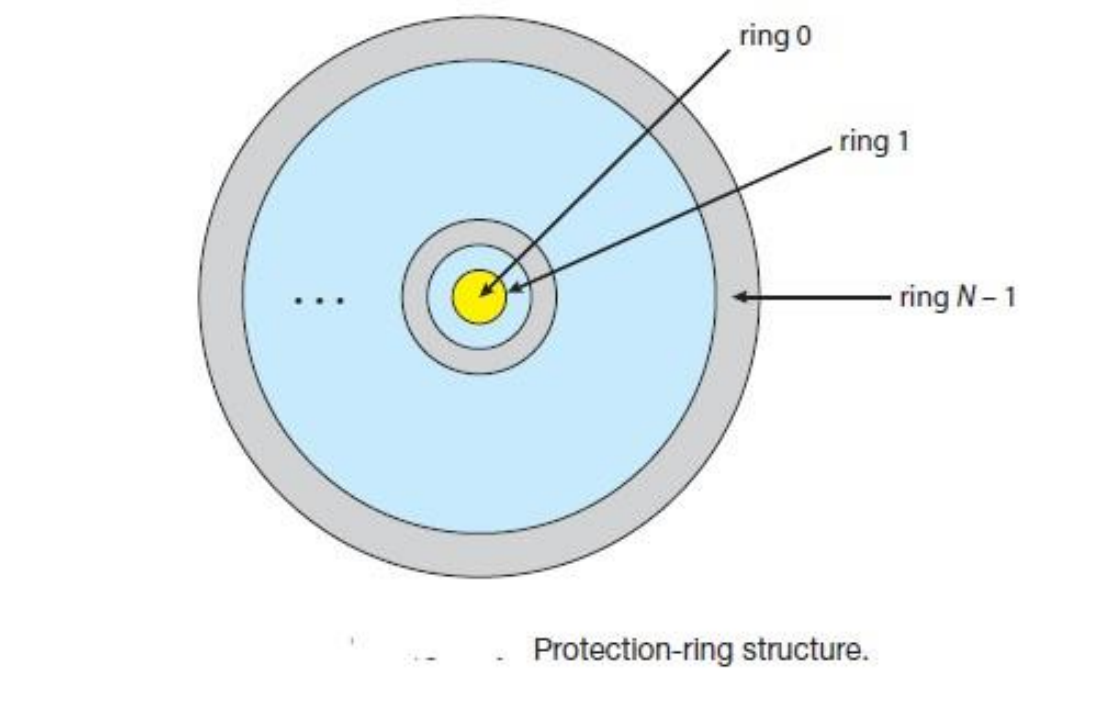
- To carry out this privilege separation, hardware support is required.

PROTECTION RINGS.

- Indeed, all modern hardware supports the notion of separate execution levels, though implementations vary somewhat.

- A popular model of privilege separation is that of protection rings.
  - In this model, execution is defined as a set of concentric rings, with ring i providing a subset of the functionality of ring j for any $j < i$.
  - The innermost ring, ring 0, thus provides the full set of privileges.

PROTECTION RINGS.



Protection-ring structure.

PROTECTION RINGS.

- When the system boots, it boots to the highest privilege level.
  - Code at that level performs necessary initialization before dropping to a less privileged level.

- In order to return to a higher privilege level, code usually calls a special instruction, sometimes referred to as a gate, which provides a portal between rings.
  - The syscall instruction (in Intel) is one example.
  - Calling this instruction shifts execution from user to kernel mode.

PROTECTION RINGS.

- When the system boots, it boots to the highest privilege level.
  - Code at that level performs necessary initialization before dropping to a less privileged level.

- In order to return to a higher privilege level, code usually calls a special instruction, sometimes referred to as a gate, which provides a portal between rings.
  - The syscall instruction (in Intel) is one example.
  - Calling this instruction shifts execution from user to kernel mode.

DOMAIN PROTECTION.

- Rings of protection separate functions into domains and order them hierarchically.

- A generalization of rings is using domains without a hierarchy.

- A computer system can be treated as a collection of processes and objects.

DOMAIN PROTECTION.

- By objects, we mean both hardware objects (such as the CPU, memory segments, printers, disks, and tape drives) and software objects (such as files, programs, and semaphores).

- Each object has a unique name that differentiates it from all other objects in the system, and each can be accessed only through well-defined and meaningful operations.

- Objects are essentially abstract data types

DOMAIN PROTECTION.

- A process should be allowed to access only those objects for which it has authorization.

- Furthermore, at any time, a process should be able to access only those objects that it currently requires to complete its task.
  - This is known as the need-to-know principle, and is useful in limiting the amount of damage a faulty process or an attacker can cause in the system.

DOMAIN PROTECTION.

- A domain can be realized in a variety of ways:
  - Each user may be a domain.
    - In this case, the set of objects that can be accessed depends on the identity of the user. Domain switching occurs when the user is changed—generally when one user logs out and another user logs in.
  - Each process may be a domain.
    - In this case, the set of objects that can be accessed depends on the identity of the process.
    - Domain switching occurs when one process sends a message to another process and then waits for a response.

DOMAIN PROTECTION.

- A domain can be realized in a variety of ways:
  - Each procedure may be a domain.
    - In this case, the set of objects that can be accessed corresponds to the local variables defined within the procedure.
    - Domain switching occurs when a procedure call is made.

ACCESS MATRIX

- The general model of protection can be viewed abstractly as a matrix, called an access matrix.

- The rows of the access matrix represent domains, and the columns represent objects.

- Each entry in the matrix consists of a set of access
- rights.

- Because the column defines objects explicitly, we can omit the object name from the access right.

ACCESS MATRIX

- The entry access(i,j) defines the set of operations that a process executing in domain D_i can invoke on object O_j.

- Consider the following access matrix

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

Access matrix.

ACCESS MATRIX

- In the access matrix, there are 4 domains and 4 objects which are—three files (F_1, F_2, F_3) and one laser printer.

- A process executing in domain D_1 can read files F_1 and F_3.

- A process executing in domain D_4 has the same privileges as one executing in domain D_1; but in addition, it can also write onto files F_1 and F_3.

- The laser printer can be accessed only by a process executing in domain D_2.

ACCESS MATRIX

- The access-matrix scheme provides us with the mechanism for specifying a variety of policies.

- The mechanism consists of implementing the access matrix and ensuring that the semantic properties we have outlined hold.

- More specifically, we must ensure that a process executing in domain D_i can access only those objects specified in row i, and then only as allowed by the access-matrix entries.

THE END