IS - Assignment no. 2 Deadline to submit – 24th Feb 2022

1. **What is Program security? What are the fault tolerance terminology?**

A security program is the entirety of an organization's security policies, procedures, tools, and controls. Essentially, your security program is the full, multi-faceted security strategy and governance that protects your organization's sensitive data and capabilities.

* 1. Error - may lead to a fault
  2. Fault - cause for deviation from intended function
  3. Failure - system malfunction caused by fault
  4. Faults - seen by” insiders” (e.g., programmers)
  5. Failures - seen by “outsiders” (e.g., independent testers, users)
  6. Error/fault/failure example:
     1. Programmer’s indexing error, leads to buffer overflow fault
     2. Buffer overflow fault causes system crash (a failure)
  7. Two categories of faults w.r.t. duration
     1. Permanent faults
     2. Transient faults – can be much more difficult to diagnose

1. **What is non-malicious program error? Explain the types with suitable example**

These errors cause program malfunctions but do not lead to more serious security vulnerabilities

We consider three classic error types:

1. Buffer overflows
   1. A buffer is a space in which data can be held. A buffer’s capacity is finite.
   2. The programmer must declare the buffer’s max. size so that the compiler can set aside that amount of space.
   3. Buffer may overflow into (and change):
      1. User’s own data space
      2. User’s program area
      3. System data space
      4. System program area
   4. Buffer Overflows Security Implication
      1. Attacker replaces code in the system space and takes control back from the operating system
      2. Attacker uses the stack pointer or return register to execute other codes

Graphical user interface, application

Description automatically generated

1. Incomplete mediation
   1. Attackers are exploiting it to cause security problems.
   2. Supplying the wrong type of data being requested.
   3. Supplying the wrong length of data being requested.
   4. Problem
      1. System Fails
      2. Supply of Bad Data
   5. Must be checked by programmer
2. Time-to-check to Time-to-Use Errors
   1. The third programming flaw we investigate involves synchronization.
   2. To improve efficiency, modern processors and operating systems usually change the order in which instructions and procedures are executed.
   3. In particular, instructions that appear to be adjacent may not actually be executed immediately after each other, either because of intentionally changed order or because of the effects of other processes in concurrent execution.
   4. Access control is a fundamental part of computer security; we want to make sure that only those who should access an object are allowed that access.
   5. This flaw concerns mediation that is performed with a "bait and switch" in the middle. It is also known as a serialization or synchronization flaw.
   6. Conditions
      1. Can occur during file I/O
      2. first checking some object and then using it
3. **What are the kinds of malicious code? explain with suitable example**

Malicious code is the general name for unanticipated or undesired effects in programs or program parts caused by an agent intent on damage.

The agent is the writer of the program or a person who causes its distribution.

Virus – code that attaches to another program and copies itself to other programs

Transient virus – life depends on life of its host eg:

Resident virus – locates inside memory eg:

Trojan Horse – malicious effect is hidden from user (Ex: login script)

Logic viruses – triggered by an event and goes off when specific condition occur Eg: 2000, Y2K

Time viruses – triggered by a time or date eg: ILOVEYOU

Trapdoor (backdoor) – feature that allows access to program other than through normal channels eg: Sort of trap doors was used as a plot device / new browser version to suppress the starvation due to workload on browser.

Worm – program that spreads copies of itself through a network, can be standalone program eg: ILOVEYOU virus

Rabbit – virus/worm that self-replicates without bound eg:

1. **How do the viruses attach to program? Explain the types of way to attach virus to program.**

a virus simply inserts a copy of itself into the executable program file before the first executable instruction, so that all the virus instructions execute first, and after the last virus instruction, control flows naturally to what used to be the first program instruction. Most common viruses today are attached to e-mail; when the attachment is opened, virus is active

Three ways:

Appended

* A program virus attaches itself to a program; then, whenever the program is run, the virus is activated. This kind of attachment is usually easy to program.
* This kind of attachment is simple and usually effective.
* The virus writer does not need to know anything about the program to which the virus will attach, and often the attached program simply serves as a carrier for the virus.

Diagram

Description automatically generated

Surrounds programs

An alternative to the attachment is a virus that runs the original program but has control before and after its execution.

Diagram

Description automatically generated

Integrated viruses and replacements

A third situation occurs when the virus replaces some of its target, integrating itself into the original code of the target.

The virus writer has to know the exact structure of the original program to know where to insert which pieces of the virus.

Finally, the virus can replace the entire target, either mimicking the effect of the target or ignoring the expected effect of the target and performing only the virus effect

Diagram

Description automatically generated with low confidence

1. **What is virus signature? How we can prevent from virus.**

* A virus cannot be completely invisible.
* Code must be stored somewhere, and the code must be in memory to execute.
* Moreover, the virus executes in a particular way, using certain methods/characteristics to spread.
* Each of these characteristics yields a pattern, called a **signature,** that can be found by a program that knows to look for it.
* The virus's signature is important for creating a program, called a **virus scanner,** that can automatically detect and, in some cases, remove viruses.
* The scanner searches memory and long-term storage, monitoring execution and watching for the signatures of viruses.
* When the scanner recognizes a known virus's pattern, it can then block the virus, inform the user, and deactivate or remove the virus.
* However, a virus scanner is effective only if it has been kept up-to-date with the latest information on current viruses

1. **Explain the virus patterns in detail.**

A virus is a fragment of code embedded in a legitimate program. Viruses are self-replicating and are designed to infect other programs. They can wreak havoc in a system by modifying or destroying files causing system crashes and program malfunctions. On reaching the target machine a virus dropper(usually a trojan horse) inserts the virus into the system.

1. **What is the control against program threat?**
2. **What is system design? Explain the principles for the design and implementation of security mechanisms.**
3. **What is confinement problem? explain the suitable example of confinement problem**

***The confinement problem deals with prevention of processes from taking disallowed actions. OR***

According to Lampson:

***The confinement problem is the problem of preventing a server from leaking information that the user of the service considers confidential.***

* Consider a client and a server. When the client issues a request to the server, the client sends the server some data.
* The server then uses the data to perform some function and returns a result (or no result) to the client.
* Access control affects the function of the server in two ways.
  + 1. The server must ensure that the resources it accesses on behalf of the client include only those resources that the client is authorized to access.
  + 2. The server must ensure that it does not reveal the client’s data to any other entity not authorized to see the client’s data.

Consider a client/server situation: the client sends a data request to the server; the server uses the data, performs some function, and sends the results (data) back to the client. In this case the confinement problem deals with preventing a server from leaking information that the user of that service considers confidential. Access control affects the function of the server in 2 ways

Goal of service provider

1. The server must ensure that the resources it accesses on behalf of the client include only those resources that the client is authorized to access.

Goal of the service user

1. The server must ensure that it does not reveal the client’s data to any other entity not authorized to see the client’s data.
2. **What is assurance? What are the types and need for assurance? Explain the trusted system.**

Assurance for secure and trusted systems must be an integral part of the development process. Confidence gained as result of evidence.

Assurance is confidence that an entity meets its security requirements based on evidence provided by applying assurance techniques.

Meets security requirements” == Enforces policy

## Types of Assurance

1. *Policy assurance* **is evidence** establishing security requirements in policy is complete, consistent, technically sound.
2. *Design assurance* **is evidence** establishing design sufficient to meet requirements of security policy.
3. *Implementation assurance* **is evidence** establishing implementation consistent with security requirements of security policy
4. *Operational* *assurance* is evidence establishing system sustains the security policy requirements during installation, configuration, and day-to-day operation
   * Also called *administrative assurance*
   * One fundamental operational assurance technique is a thorough review of product or system documentation and procedures, to ensure that the system cannot accidentally be placed into a non-secure state.
   * This emphasizes the importance of proper and *complete* documentation for computer applications, systems, and other entities.

## The need of Assurance

* Applying assurance techniques is time-consuming and expensive.
* Accidental or unintentional failures of computer systems, as well as intentional compromises of security mechanisms, can lead to security failures.

## Trusted System

* A *trusted system* is a system that has been shown to meet well-defined requirements under an evaluation by a credible body of experts who are certified to assign trust ratings to evaluated products and systems.



1. **What are the waterfall lifecycle model for secure design system?**

The waterfall life cycle model is the model of building in stages, whereby one stage is completed before the next stage begins.

Five Steps are:

1. Requirement’s definition and analysis
   1. Functional and non-functional
   2. General (for customer), specifications
2. System and software design
3. Implementation and unit testing
4. Integration and system testing
5. Operation and maintenance

The waterfall life cycle model. The solid arrows represent the flow of development in the model. The dashed arrows represent the paths along which information about errors may be sent.

