IT securtiy Coding and implementation

# CPU

* Fetching

Gets instruction from System memory

The pointer keeps track of the instructions(performed/not performed and stored data)

* Decoding

Deciphers the instruction.

* Execution

Control moves from control unit to the ALU and mathematical operations occur

* Storing

ALU stores the information in a register/memory

Releases the result to output

# Compiled languages

Compilation: Cannot be executed unless the all necessary files are provided.

Linking: The process where the required depencies are linked together to run the program.

The output is machine code.

# Interpreted languages

Reads line by line.

Is not compiled.

Slow execution quicker changes.

# Hyrbrid languages

Compiled.

Required independet host program JVM for java

At runt time Just In Time JIT compiler converts it into machine code.

# Buffer overflow

Stack overflow: Target is the stored that mostly Corrupts it. When length is not checked in array/input

# Injection flaw

SQL: When attacker inputs sql commands in an input field etc DROP database; Sanitize

OS injection: Adressbar where you can put %20 to a space and reach administrator pages without previlegies, XML injection

# Cross-Site Scripting (XSS)

Often about stealing cookies

# Sensitive Data Exposure

* + Lack of appropriate confidentiality controls lead to data exposure
  + Common causes of data exposure are
    - Insufficient data-in-motion protection(Man in the middle)
    - Insufficient data-at-rest protection
    - Electronic social engineering

# Cross-Site Request Forgery (CSRF)

Hacker provides a forged link wich has the intentions of stealing information HTTP

# Race condition

When two threads are executing at the same time, but only one thread can alter

# Side Channel Attacks

* + Timing attacks
    - Attacker measure how long each computational operation takes and uses that side channel information to discover other information about the internal makeup of the system
  + Power Analysis attacks
    - The attacker measures the varying degrees of power (peak power consumption can be interpreted that CPU is doing multiplication)
  + TEMPEST attacks
    - Attacker uses leaked electromagnetic radiations to reveal infomation

# DEVELOPMENT AND CODING

# Input validation

Server side validation is a must, both recommended

# Canonicalization

The process of converting data that has more than one presentation

URL for example

Canonical form <http://www.google.se>

Other representive <http://google.se>

Etc

# Sanitization

Remove of illegal characters from input

# Error handling

Generalizated error messages

Repeated error should lock out

Error 122”#12321 has occured contact admin

# Memory management

# Locality of Reference

Its predictable when the data gets referenced

# Temporal locality

Referenced memory location are more likely to get used again

# Spatial locality

The memory beside the referenced memory location are likely to get used again

# Dangling pointers

The pointer references to a objects memory space. Dangling pointers are when the pointers references to a location where the object has been removed but not the pointer.

# Garbage collector

Reclaims unreachable memory objects.

Slow/fast death

# Type safety

Cannot access memory address space.

C# prevents this during JIT compilation

İn C++ this isnt possible and you can reach beyond any array if not declared correctly

# Excepton management

SAFESH

# Session management

Always authenticate users only from an encrypted source (web page).

Update renew session

# Concurrency

Race window: ex two Word running at the same time

# Atomic operations

If running cannot be altered by another object

# Mutual excursion

Two mutual processes running simultaneous but exclusive to each other.

Resource locking

# Tokenization

Replacing sensitive data with tokens

# Sandboxing

Google chromes extensions is a kind of sandboxing

# Anti Tampering

# Code Analysis

# Code/Peer review