Class 8

Implementation Issues

Deployment and implementation of real security systems

The weakest link properly:

A system is as strong as the weakest element/component of its system

Individual components of the system not all are/were designed for a secure environment.

Attacks to cryptography might be harder compared to attacks in the implementation of system or specific components of the system.

A pressure to design a reliable cryptographic system as successful brakes can lead to an unnoticed.

Implementation Issues

Developing correct programs

A correct program is the one that operates according to its specifications

Specifications

Requirements:

Informal description of the program's goal. General layout.

Functional specification:

A details description and definition of the behavior of the program. The measurement of functionality constrains the specification of the program. A test suite might be design to be use as an external testing mechanism.

Completeness is the goal of the functional specification. Anything not included in the functional specification should not be included.

Implementation design

Specifies how the program works internally. Modularized approach, program split into sub-programs (modules)

Testing approach

From Edsger Dijkstra 1972, testing can only show the presence of bugs, never the absence of bugs.

Some rules that could lead to a better quality of software.

- Per bug finding, implement a test that detects the bug. Fix the bug, check that the test did not find the bug.
- Review what is causing the bug whenever you find it.
- Keep track of every bug event. Statistical analysis of the bugs found.

Testing approach

Software industry Vs Aerospace Industry

Cost vs Quality

Creating Secure Software

Lack of functionality is the difference between correct and secure software.

Secure software should be able to implement it, no matter what Eve does she can not do *X.*

Functionality can be tested on contrary lack of functionality can not be tested.

“Standar implementation techniques are entirely inadequate to create secure code”

Secrets and Memory

Transient secrets for the Secure channel?

Transient secrets are kept/store in memory

Wiping state

Wipe any type of information as soon as it not used or needed.

Transient secrets will involve wiping memory locations. Complicated when considering how the memory is allocated by different software/programs, i.e C Vs C++.

For C++ there is a destructor function for each object. On standard operation of the program the destructor should clean the memory.

Program should also care about that all heap-allocated object are refurbish.

In Java is more complicated as all objects live in Heap memory is garbage collected.

Wiping state

Java’s exception handling makes it difficult to do the wiping state manually.

Ensure that the finalization routing are run at program exit.

Secret data could also land into CPU registers. Not possible to wipe the state of the registers manually

Swap file

Virtual memory system is used by OS like Windows and Linux to increase the number of programs that run in parallel.

Some data is store in a swap file while the program is running.

The Virtual Memory should care about the security of the swap file.

If you can constrain the memory (not to be swapped) which memory should be locked?

Caches

Cache keeps a copy of the most recently used data from the main memory.

Data retention by Memory

Overwriting data in memory does not delete the data. Depends of the type of memory involved.

Static RAM (SRAM). Power cycling the memory could return previous states (old data).

Dynamic RAM (DRAM). Works by storing a small charge on a small capacitor. The insulating material around the capacitor is stressed by resulting field. An attacker with physical access could recover the data.

Slow decay of charge in DRAM capacitors at room temperature.

Data retention by Memory

Partial solution proposed. Instead of storing message ***m*** store message R

R random string

$$m \rightarrow R,\ h(R) \oplus m$$

$$where,$$

$$h\ is\ a\ hash\ function$$

To read from this store you read both parts, hash the first and XOR them together to get ***m.***

Multi-access

Program on the same machine may access to the same data.

Share memory.

User hierarchy in OS like linux. Super user and regular user.

Data Integrity

Assumption the hardware use is reliable. Memory could use an Error-correcting code.

Quality of Code

Simplicity

Complexity is worst any for security.

Programs availability to support many cipher suites, eliminate if possible.

Modularization

A complex model could be manageable is modularized.

Assertions

Professional paranoia.

Assertion errors should lead to an abort of the program.

Quality of Code

Buffer overflow

From Algol 60 array bound checking. Do not disable it.

Testing

Two type of tests

First. Generic functional tests developed from the module's functional specifications

Second. Developed by the programmer of the module itself.

Side channel attacks

Timing information could lead to leak information about the message itself or the underlying encryption key.

An attacker could measure how much current the card draws in time is the cryptography is embedded in a smart card.

Magnetic field, RF emissions, power consumption, timing … can be used for side-channel attacks.