What is Software Security?

Software Security

- Software security is a kind of computer security that focuses on the **secure design and implementation of software**
  - Using the best languages, tools, methods
- ***Focus*** *of study:*

`the code`

- By contrast: Many popular approaches to security treat software as a *black box* (ignoring the code)
  - OS security, anti-virus, firewalls, etc.

Why Software Security?

Firewalls and anti-virus are like building walls around a weak interior

Attackers often can bypass outer defenses to attack weaknesses within

Software Security aims to address weaknesses directly

Operating System Security

- Operating systems **mediate a program's actions**
  - *Aka* **system calls**
  - such as reading and writing files,
  - sending and receiving network packets,
  - starting new programs, etc.
- Enforceable policies control actions
  - programs run by Alice cannot read files owned by Bob
  - programs run by Bob cannot use TCP port 80
  - programs run in directory D cannot access files outside of D

Limitations of OS Security

- **Cannot enforce application-specific policies**, which can be too fine-grained
  - Example: database management system (DBMS)
- **Cannot** (precisely) **enforce info-flow policies**
  - An operating system typically implements an **execution monitor**: decisions are based on *past and current actions*
  - **Information flow policies**: A *non*-action may reveal something about a secret without leaking it directly

Firewalls and IDSs

- Firewalls and intrusion detection systems (IDSs) **observe**, **block**, and **filter messages** exchanged by programs
  - Based on their origin, content, frequency, etc.
- Examples:
  - Firewall could block all traffic from particular hosts, or to particular TCP ports
  - An IDS could filter packets it recognizes are part of a known exploit pattern

Filtering misses attacks

- **Firewall filtering** is coarse-grained, and **unsound**
  - Port 80 is assumed to be HTTP (web) traffic, which is assumed benign, but can layer arbitrary traffic over HTTP, e.g., SOAP
  - Previously benign sources can become malicious
    - E.g., due to malware infection
- **IDS patterns** fine-grained, but **still unsound**
  - Attack traffic can be slightly modified to work around IDS filters (which are often *syntactic*, not *semantic*)
  - Making filters too fine-grained can hurt performance
    - Thus **compromising availability**

Anti-virus Scanners

- Anti-virus scanners look for **signs of malicious behavior** in local **files**
- In many ways, anti-virus is related to IDS in looking for patterns
- Newer forms of anti-virus scanners are sophisticated, but i**n practice are frequently bypassed**
  - **Trade off precision and performance** (latter could compromise availability)

Ex: Heartbleed

- SSL/TLS is a core **protocol** for **encrypted communications** used by the web
- Heartbleed is a **bug** in the commonly used **OpenSSL** implementation of SSL/TLS, v1.0.1 - 1.0.1f
  - Discovered in March 2014, it has been in released code since March 2012 (**2 years old!**)
- A carefully crafted packet causes OpenSSL to read and return portions of a vulnerable server's memory
  - Leaking passwords, keys, and other private information

Heartbleed, meet SoftSec

- **Black box security is incomplete against Heartbleed exploits**
  - Issue is not at the level of system calls or deposited files: nothing the OS or antivirus can do
  - Basic attack packets could be blocked by IDS, but
    - "Packet chunking" may bypass basic filters
    - Exfiltrated data on the encrypted channel; invisible to forensics
- **Software security** methods attack the **source** of the problem: **the buggy code**