Software Security

Mohammadreza Hazhirpasand
Software Composition Group
What I want to share

- Secure software
  - Development process and security development lifecycle
  - Threat modeling
  - Security requirements
  - Microsoft SDL

- Mycompany’s demo
  - Dynamic/static analyses
  - Client-side and server-side penetration tests
  - Network Sniffing
  - Denial of service attacks
  - & more...

- Wrap up
  - More attacks
  - Cyber security resources
Secure software

• Flawed software  → Ignoring security during the design phase
  Add security when functional requirements are satisfied

• Better approach  → Incorporate security into all phases of the development process
Development process

Four common phases:

1. Requirements + security
2. Design + security
3. Implementation + security
4. Testing + security

Where is security?
Development process

Four common phases:

1. Requirements + security
   • Security requirements
   • Abuse cases
2. Design + security
   • Architectural risk analysis
   • Security-oriented design
3. Implementation + security
   • Code review (manual + tools)
4. Testing + security
   • Risk-based security tests
   • Penetration testing

Where is security?
Secure software vs. hardware

- Software
  - Easily changeable
  - Can be weak at security

- Hardware
  - Hard to change
  - Exploiting hardware is not easy
  - Intel SGX (encrypted computation)
Threat modeling

• A threat model is a structured representation of all the information that affects the security of an application, IoT, distributed system and so on

• The threat model is critically important

• This is part of architectural risk analysis

• STRIDE is a model of threats (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of service, Elevation of privilege)
Example: Snooping user

As a malicious user who is connected to a network where others are also working:

1. Read others’ messages
2. Intercept, modify, and duplicate messages
3. Flood the network with invalid packets

...
Example: Co-located user

As a malicious user who has installed a malware on a user’s machine:

1. Read/write users’ files
2. Read/write users’ memory
3. Record user’s keystroke

... Confidential information theft, Encrypting users’ data (Ransomware)
Threat-driven design

• Different threat models can provide different aspects of your software security

• E.g. In the snooping attack scenario encrypting user traffic is important (IPsec, SSL, WPA3, …)

• E.g. In the co-located user scenario providing users with additional means of authentication is necessary
Security requirements

• Software requirements: what the software should do?

• Security requirements
  • Security goals or policies (secrecy of user’s bank balance)
  • Security mechanisms (passwords)
Kinds of requirements

• Policies
  • Confidentiality: sensitive information should not be leaked
  • Integrity: changing the content of a network packet
  • Availability: DoS to weaken availability

• Mechanisms
  • Authentication: password, biometrics, multi-factor auth..
  • Authorization: access controls, role-based or user-based permissions
  • Auditability: logging every event in the system - backups
Security principles

- **Prevention**
  - Eliminate software defects completely
  - Heartbleed bug would have been prevented by a type safe language

- **Mitigation**
  - Reduce the damage from unknown exploitation possibilities
  - Run each browser’s tab in a separate process

- **Detection**
  - Identify the attack and undo the damage
  - Monitoring and taking snapshots periodically
Microsoft SDL – Security Development Lifecycle

• The Microsoft SDL incorporate security and privacy considerations into all phases of the development process, supporting developers build highly secure software, address security compliance requirements, and reduce development costs.

Mycompany – A very bad example!

Password + User’s name

Checks the password + Stores the name

Client side – C++ application

Server side – PHP application

Security through obscurity
Mycompany - demos

• Static/Dynamic analyses
• Traffic analysis
• SQL injection
• DoS
Terminology – DoS attacks

• A Denial-of-Service (DoS) or Distributed DoS attacks meant to shut down or slow down a machine or a network

• DoS attacks accomplish this by flooding the target with traffic, which is commonly useless. Sometimes the traffic triggers a crash in the remote program

• The attack is easy to perform for attackers

• ICMP and SYN flood
Terminology – SQL injection

• SQL injection makes it possible to execute malicious SQL statements

• An attacker can insert, update, or delete a record

• The problem is rooted in unchecked inputs
Mycompany – lessons learned

• Obfuscate the code / do not use hard-coded secrets

• Validate the inputs / use web-based firewalls

• Limit the number of request per machine + firewalls/IPS/IDS

• Encryption + TLS

• Not made-up approaches such as concatenation of weak random numbers
Hmmm..

More demo?
Pcap file analysis

• Pcap files are commonly data files generated by network packet capturing programs

• They normally contain the packet data of a network

• Many hands-On packet analysis courses exist....
Android analysis

• Reverse engineering Android applications
• Exploit Android vulnerabilities
• Discover hard-coded secrets
• ...

Cryptography

• Hashing algorithms (MD5, SHA-1 (160-bit), SHA-2, ....)

• Symmetric encryption

  1. Employs a single cryptographic key to encrypt and decrypt data
  2. It is fast compared to its counterpart (asymmetric)
  3. DES – 3DES – AES (all are block ciphers)
Block ciphers...

• ... employs various modes of encryption (ECB, CBC, CTR, ...)
• ... necessitates the input to be an exact multiple of the block size
• **ECB** jeopardizes the security of your software system!
• **Duplicated plaintext** blocks give the same ciphertext block!
ECB

Electronic Codebook (ECB) mode decryption
ECB

Unencrypted  ECB mode encrypted  CBC mode encrypted
The broken system...

- The cookie contains a username and password encrypted by an algorithm and ECB mode
- Base64 is an encoding algorithm used in the cookie
- The problem is encryption provides confidentiality but not integrity
- Integrity checks for data tampering
The broken system...

If the username is test and password is password:
DNS rebinding attack

• Cross-origin policy (SOP) in browsers restricts how a document or script loaded by one origin can interact with a resource from another origin

• However, SOP only checks the domain names!

• DNS rebinding effortlessly circumvents SOP!
DNS rebinding attack

• Ubuntu – the attacker – uses a DNS changer

• Kali Linux – the victim – only has a localhost on his machine

• The attacker wants to read the victim's "oh.txt" file, locating in the victim's localhost

• The victim visits the attacker's malicious website

• The malicious website continuously checks for an update in the victim's browser cache
What else?

• Reconnaissance

• Network scanning

• System hacking

• Malware threats

• Hacking mobile platforms
What else?

• Social engineering

• Session hijacking

• Evading IDS or firewalls

• Hacking web servers

• Cryptography
Certifications in cyber security

• Security+
• Certified Information Systems Security Professional (CISSP)
• Certified Ethical Hacker (CEH)
• Offensive Security Certified Professional (OSCP)
• Offensive Security Wireless Professional (OSWP)
• Offensive Security Experienced Penetration Tester (OSEP)
• Offensive Security Exploit Developer (OSED)
• eLearnSecurity Certified Professional Penetration Tester (eCPPT)
• www.pentesterlab.com
Now you should know

• What is a secure software?

• What is Microsoft SDL?

• How a vulnerability can be exploited?

• What security aspects must be taken into account when writing software?