CSE 484 / CSE M 584:
Finish Software Security +
Start Cryptography

Fall 2022

Franziska (Franzi) Roesner
franzi@cs

UW Instruction Team: David Kohlbrenner, Yoshi Kohno, Franziska Roesner. Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...
Announcements

• Things Due
  – Lab 1a, due Saturday
  – Research Reading #2 (584M) due Thursday
Password Checker

• Functional requirements
  – PwdCheck(RealPwd, CandidatePwd) should:
    • Return TRUE if RealPwd matches CandidatePwd
    • Return FALSE otherwise
  – RealPwd and CandidatePwd are both 8 characters long
Password Checker

• Functional requirements
  – PwdCheck(RealPwd, CandidatePwd) should:
    • Return TRUE if RealPwd matches CandidatePwd
    • Return FALSE otherwise
  – RealPwd and CandidatePwd are both 8 characters long

• Implementation (like TENEX system)

```c
PwdCheck(RealPwd, CandidatePwd)  // both 8 chars
for i = 1 to 8 do
    if (RealPwd[i] != CandidatePwd[i]) then
        return FALSE
return TRUE
```

• Clearly meets functional description
**Attacker Model**

- Attacker can guess *CandidatePwds* through some standard interface
- Naive: Try all $256^8 = 18,446,744,073,709,551,616$ possibilities
- Is it possible to derive password more quickly?
  - **Time** how long it takes to reject a *CandidatePwd*
  - Then try all possibilities for first character, then second, then third, ....
  - Total tries: $256 \times 8 = 2048$

```plaintext
PwdCheck(RealPwd, CandidatePwd) // both 8 chars
for i = 1 to 8 do
    if (RealPwd[i] != CandidatePwd[i]) then
        return FALSE
return TRUE
```
Timing Attacks

- Assume there are no “typical” bugs in the software
  - No buffer overflow bugs
  - No format string vulnerabilities
  - Good choice of randomness
  - Good design

- The software may still be vulnerable to timing attacks
  - Software exhibits input-dependent timings

- Complex and hard to fully protect against

- Even possible over a network
  - “Remote timing attacks are possible” (Brumley & Boneh, 2005)

- Plenty of other side channels... We’ll return to this later in the course
Software Security:
So, what do we do?
General Principles

- Check inputs
- Check all return values
- Principle of least privilege
- Securely clear memory (passwords, keys, etc.)
- Failsafe defaults
- Defense in depth
  - Also: prevent, detect, respond
General Principles

• Reduce size of trusted computing base (TCB)
• Simplicity, modularity
  – But: Be careful at interface boundaries!
• Minimize attack surface
• Use vetted components
• Security by design
  – But: tension between security and other goals
• Open design? Open source? Closed source?
  – Different perspectives
Vulnerability Analysis and Disclosure

• What do you do if you’ve found a security problem in a real system?
• Say
  – A commercial website?
  – UW grade database?
  – Boeing 787?
  – TSA procedures?

What would you do? What ethical questions come up?
Next major section of the course: Cryptography

Terminology note: “blockchain” and “crypto”
• Rising interest, mostly in the cryptocurrency space
• For this course: crypto means “cryptography”
Common Communication Security Goals

**Privacy of data:**
Prevent exposure of information

**Integrity of data:**
Prevent modification of information
Recall Bigger Picture

• Cryptography only one small piece of a larger system
• Must protect entire system
  – Physical security
  – Operating system security
  – Network security
  – Users
  – Cryptography (following slides)
• Recall the weakest link

• Still, cryptography is a crucial part of our toolbox
XKCD:  http://xkcd.com/538/
History

• Substitution Ciphers
  – Caesar Cipher
• Transposition Ciphers
• Codebooks
• Machines

• Recommended Reading: The Codebreakers by David Kahn and The Code Book by Simon Singh.
History: Caesar Cipher (Shift Cipher)

• Plaintext letters are replaced with letters a fixed shift away in the alphabet.

• Example:
  – Plaintext: The quick brown fox jumps over the lazy dog
  – Key: Shift 3
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    DEFGHIJKLMNOPQRSTUVWXYZABC
  – Ciphertext: WKHTX LFNEU RZQIR AMXPS VRYHU WKHOD CBGRJ
History: Caesar Cipher (Shift Cipher)

• ROT13: shift 13 (encryption and decryption are symmetric)
• What is the key space?
  – 26 possible shifts.
• How to attack shift ciphers?
  – Brute force.