COMP 3704 Computer Security

Christian Grothoff
christian@grothoff.org

http://grothoff.org/christian/
README

Overview

- Computer Security \(\equiv\) protecting information

- **Protecting**: Integrity, confidentiality, authenticity, availability
- **Information**: Randomness, entropy, correlation, storage, transmission

- You will write code in C, C++ and Java.

- You must already be able to write systems code in either C, C++ or Java.
Academic dishonesty

- Webpage says what is allowed.
- If in doubt, ask first.
- Cheating can be detected with automated tools.
- Any violation will be reported to the dean.
Expectations

- Read the indicated chapters of the textbook – not every detail is covered in class, but it may still be helpful in exams!

- Study additional material (software documentation, other books, additional textbook chapters) as needed.

- Deliver tested and working versions of projects on time using subversion.

- Answer questions in midterm and final exams.
Programming Assignments: The Rules

- All projects are individual projects.
- You must submit a working version of the code by the deadline to the subversion repository.
- Your projects must compile either by invoking configure; make or just make. You must also include instructions how to run the resulting program.
Questions
Computer Security Overview

- Cryptography (mathematics)
- Network Security (protocols)
- System Security (access control)
- Application Security (bugs)
Terminology (0/6)

- An **adversary** is a subject trying to break the security of a system.
- A **threat** is a mechanism that the adversary can employ to achieve his goals.
- A **risk** is a loss that would occur if the adversary succeeds.
- A **vulnerability** is a flaw creating a threat.
- A **threat model** describes the mechanisms available to the adversary.
- A **trust model** describes subjects that are trusted not to have vulnerabilities.
- A **security model** specifies functional and security goals together with threat and trust models.
Terminology (1/6)

- Plaintext: $P$
- Ciphertext: $C$
- Encryption: $E_K(P) = C$
- Decryption: $D_K(C) = P$
- Cryptography + Cryptanalysis = Cryptology
- Steganography
Terminology (2/6)

• Authentication: receiver ascertains origin of message
• Integrity: verify message was not modified in transit
• Nonrepudiation: sender cannot deny sending message
Terminology (3/6)

- Cipher $= (E, D)$

- restricted algorithm $\equiv$ security based on secrecy of algorithm

- modern algorithm $\equiv$ security based on secrecy of key $K$
Terminology (4/6)

- Symmetric algorithms – same key for $E$ and $D$
- Public-key algorithms – different keys for $E$ and $D$, for example:

$$E_{K_{pub}}(P) = C$$
$$D_{K_{priv}}(C) = P$$
Terminology (5/6)

Types of cryptanalytic attacks:

- Ciphertext-only
- Known-plaintext
- Chosen-plaintext
- Adaptive-chosen-plaintext
- Chosen-ciphertext
- Brute-force
- Rubber-hose cryptanalysis
Attacker limitations:

- Data complexity (how much data required as input to the attack)
- Processing complexity (how much processing is needed)
- Storage requirements (how much memory is needed)
Substitution Ciphers

- Monoalphabetic ciphers $\equiv 1:1$
- Homophonic substitution ciphers $\equiv 1:n$
- Polygram substitution ciphers $\equiv n:m$
- Polyalphabetic substitution ciphers

Famous examples: Caesar Cipher, ROT13, Vigenere, Enigma
Transposition Ciphers

• change the order of the characters, not the characters

• frequency distribution unchanged

• requires buffers in memory
XOR with key

- Vigenere polyalphabetic cipher
- Generally easy to break,
- except: key length = ciphertext length
Question

Why are one-time-pads almost never used in practice?
Questions
General Homework Hints

- $ \texttt{svn add filename} ; \texttt{svn commit -m "logmessage"}$
- $ \texttt{gcc -o binary sourcename.c} ; \texttt{./binary}$
- $ \texttt{latex filename.tex} ; \texttt{xdvi filename.dvi}$
- $ \texttt{javac pack/Type.java} ; \texttt{java pack.Type}$
Homework Summary

Before the next lecture:

• Generate password with `htpasswd` and register account.
• Read the first chapters of the subversion manual
• Install software (or use department machines).
• Implement “Hello World”, test and submit!
• Read Chapters 1 & 2.
Questions

?