

COMP 3704 Computer Security

Christian Grothoff

`christian@grothoff.org`

`http://grothoff.org/christian/`

README

<http://grothoff.org/christian/teaching/2009/3704/>

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, except for assignment “ZERO”, 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

Terminology (6/6)

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)

Kerckhoff's principle (1883)

The only thing the adversary does not know is the secret key.

The design of encryption and decryption algorithms and the protocol is public:

- Allows public scrutiny of the design
- No need to replace system if design is exposed
- Same design can be used for multiple applications
- Focus on security the key!

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

- `$ svn add filename ; svn commit -m "logmessage"`
- `$ gcc -o binary sourcename.c ; ./binary`
- `$ latex filename.tex ; xdvi filename.dvi`
- `$ javac pack/Type.java ; 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



Protocols

- “A **protocol** is a series of steps, involving two or more parties, designed to accomplish a task.”
- Everyone involved must know the steps in advance and agree to follow it.
- The protocol must be complete and unambiguous.
- For cryptographic protocols, it should not be possible to do more or learn more than what is specified in the protocol.

Dramatis Personae¹

- Alice, Bob, Carol and Dave
- Eve – Eavesdropper
- Mallory – Malicious active attacker
- Trent – Trusted arbitrator
- Walter – Warden
- Peggy – Prover
- Victor – Verifier

¹More at http://en.wikipedia.org/wiki/Alice_and_Bob

Efficiency

- Number of steps in protocol
- Size of messages
- Conflict resolution cost:
 1. Involvement of trusted party (arbitrated protocols)
 2. Resolution by trusted party on dispute (adjudicated protocols)
 3. Self-enforcing protocols

Attack Personae

- Eavesdroppers
- Passive cheaters
- Active cheaters
- Real-world adversaries – Mallory

Example: Symmetric Cryptography

1. Alice and Bob agree on a cryptosystem
2. Alice and Bob agree on a key
3. Alice encrypts plaintext with key
4. Alice sends ciphertext to Bob
5. Bob decrypts ciphertext and reads it

One-Way (hash) Functions

- Easy to compute $f(x)$, hard to compute $f^{-1}(y)$
- Trapdoor one-way hash functions: hard to compute f^{-1} without the secret
- Good hash functions are collision-free: it is hard to generate two pre-images with the same hash value

Alternative names

- Contraction function
- Message digest
- Fingerprint
- Cryptographic checksum
- Message integrity check (MIC)
- Manipulation detection code (MDC)
- Message authentication code (MAC) \equiv hash + key

Public-key Cryptography

The mathematical primitive is often similar to trapdoor one-way hash functions.

Canonical use:

1. Alice and Bob agree on a public-key cryptosystem.
2. Bob sends Alice his public key.
3. Alice encrypts her message using Bob's public key.
4. Alice sends the ciphertext to Bob.
5. Bob decrypts Alice's message using his private key.

Hybrid Cryptosystems

- Use public key cipher for key exchange (session key)
- Use symmetric cipher for data exchange
- Use hashing/MAC to verify data integrity

Signatures

Autenticic: The signer deliberately signed the document.

Unforgeable:

Nobody but the signer signed the document.

not reusable:

The signature cannot be moved to another document.

Unalterable:

The document cannot be changed after signing.

not repudiatable:

The signer cannot later claim not to have signed it.

Public-key Signatures

1. Alice encrypts the document with her private key.
2. Alice sends the signed document to Bob.
3. Bob decrypts the document with Alice's public key.

In practice, Alice does not encrypt the document, but a hash of the document. This is implied when we use the notations $S_K(M)$ and $V_K(M)$ for signing and verifying.

Attacks

- Repudiation by intentional key compromise
- Replay attacks
- Signing is decrypting!

Questions



Problem

Alice has an item x , and Bob has a set of five distinct items y_1, \dots, y_5 . Design a protocol through which Alice (but not Bob) finds out whether her x equals any of Bob's five items; Alice should not find out anything other than the answer ("Yes" or "No") to the above question, and Bob should not know that answer. Your solution must always be correct, not just with high probability.