ECEN 5022 Cryptography

Introduction

Peter Mathys

University of Colorado

Spring 2008
Historically, cryptography is the science and study of secret writing (Greek: “kryptos” = hidden, “graphein” = to write).

Modern cryptography also includes such topics as authentication, message integrity, digital signatures, and cryptographic protocols.

Classical cryptography is typically concerned with patterns in languages and how to conceal them. For English the starting point is usually the 26 letter alphabet (often converted to numbers 0...25).

The original goal of cryptography is encryption for secrecy.

Alice wants to send a message $m$ to Bob, but Eve listens in.

The use of names (Alice and Bob for the “good guys” and Eve for the eavesdropper) is traditional in cryptography.
Encryption

Alice and Bob use encryption to keep their communication secret from Eve.

Alice and Bob agree on a secret encryption key $K_e$, using a secure communication channel.

When Alice sends a message $m$ to Bob she encrypts it as $c = E(K_e, m)$. $m$ is also called plaintext and $c$ is also called ciphertext.

Bob decrypts the message as $m = D(K_e, c)$.

Eve does not have the secret key $K_e$ and cannot decrypt the ciphertext.
Kerckhoffs’ Principle

▶ To decrypt the ciphertext $c$, two things must be known: (i) the decryption algorithm $D$ and (ii) the key $K_e$.
▶ It is tempting (but foolish) to argue that the most secure strategy is to keep both $D$ and $K_e$ secret.
▶ **Kerckhoffs’ Principle.** The security of the encryption scheme must depend only on the secrecy of the key $K_e$ and **not** on the secrecy of the algorithm(s) $D$ (and $E$).
▶ **Reasons:**

▶ Cryptographic systems are built for many users and are used for many years. Thus, changing algorithms (if they are compromised) is expensive and difficult to do.
▶ Because many users use the same algorithm, it is infeasible to keep it secret.
▶ In fact, the algorithms should be published, so that everybody can try to find flaws. By publishing your algorithm you can have it analyzed for free!
Need for Authentication

- Eve can do more than just listen in. She can delete a message so that Bob never receives it.
- She can also try to alter the message $m$ (or invent a new message) so that Bob receives message $m'$.
- Problem: Suppose Bob just received a message. How does he know it came from Alice and not from Eve?
- Solution: Use a message authentication code (MAC).
Authentication

$h$ is a MAC function (often a hash function).

Alice now sends both $m$ and $a$ to Bob. He recomputes $a$ from $m$ and checks against received $a$.

If Eve sends $m'$ instead of $m$, $h(K_a, m') \neq a$ for a good MAC function.
Authentication

- If Eve does not know $K_a$, then she cannot send $m'$ and corresponding $a'$ to Bob.
- But Eve can still delete messages, or delay messages, or change the order of messages.
- Thus, some form of message integrity is also needed. A simple strategy is to use time stamps and/or number the messages sequentially.
- Note that it is possible to combine secrecy and authentication.
To use (conventional) encryption and/or authentication, Alice and Bob must share secret keys $K_e$ and/or $K_a$.

To exchange keys there must be a secure channel.

Distributing and managing keys is one of the difficult problems of cryptography.

Alice and Bob can meet for dinner once a month, but if there are $N = 100$ people, then $N(N - 1)/2 = 100 \times 99/2 = 4950$ pairs of keys need to be distributed securely.
Public Key Cryptography

- $K_e$ is public encryption key for Bob, $K_d$ is Bob’s secret decryption key.
- For public-key cryptography $K_e \neq K_d$. Moreover, it must be infeasible to compute $K_d$ from knowledge of $K_e$.
- Necessary to recover $m$: $D(K_d, E(K_e, m)) = m$.
- Another name for public-key encryption is asymmetric-key encryption.
- Simplification of key distribution problem: Bob now only has to distribute/publish his public key $K_e$ that everybody can use.
Digital Signatures

A digital signature must have the property that everyone can check it, but only one person can generate it.

Alice computes signature $s$ for message $m$ as $s = \sigma(K_d, m)$ using her private key $K_d$. She sends $m, s$ to Bob.

Bob receives $m, s$ and uses Alice’s public key $K_e$ to verify the signature with $v(K_e, m, s)$. This works like a MAC, except that it is verified with a public key, whereas the private key is needed to generate $s$. 
Public Key Infrastructure

- Public-key cryptography simplifies the key management problem, but Alice still needs to be able to find Bob’s key and be sure that it’s not Malice who pretends to be Bob.
- The general solution is to use a public key infrastructure (PKI).
- The main idea is to set up a central authority, called certificate authority (CA).
- Each user then takes their public key to the CA and the CA verifies their identity (e.g., using a passport or a fingerprint).
- The CA then signs the user’s public, saying something to the effect: The CA has verified that this key is Bob’s public key.
- Some problems: The CA must be trusted by everybody. And what if the CA issues a false certificate (e.g., based on a forged ID)? Who is liable?
Ciphertext-Only Attack

- The cryptanalyst has (one or more) cryptograms available that were encrypted with the same key and tries to find the corresponding plaintext(s).
- The cryptanalyst may also try to deduce the key from the cryptograms.
- This is the most difficult type of attack.
- A ciphertext-only attack is what most people tend to think of when they hear talk about breaking an encryption system.
Known Plaintext Attack

- The cryptanalyst has one (or more) plaintext(s) and the corresponding ciphertext(s), all based on the same key, available.

- The goal is to find the encryption and/or decryption key or an algorithm that can decrypt any further messages encrypted with the same key.

- Plaintext/ciphertext pairs may be obtained from standard messages or message headers (e.g., e-mail autoreply), or from mailings that are sent encrypted to several people, including a cryptanalyst.
Chosen Plaintext Attack

➤ Here it is assumed that the cryptanalyst has obtained (temporary) access to an encryption device. This is clearly easily possible for public key encryption where the encryption key and algorithm are public knowledge.

➤ The cryptanalyst chooses plaintexts with desirable properties and computes the corresponding ciphertexts. The goal is to find the decryption key or an algorithm that can decrypt all messages that are encrypted with the same key.

➤ A big advantage of a chosen plaintext attack is that the cryptanalyst can repeatedly encrypt new plaintexts with modifications derived from previous encryptions.
Chosen Ciphertext Attack

- In this case the cryptanalyst has gained (temporary) access to a decryption device and can compute the plaintext for any ciphertext.
- Often the cryptanalyst gets to choose both plaintext and ciphertext values.
- The goal is again to obtain the decryption key or an algorithm for computing the plaintext from the ciphertext.
- The chosen ciphertext attack is the most powerful of the cryptanalytic attacks discussed here.
Birthday Attacks

- If there are 23 people at a party, the probability that two have the same birthday is a little larger than 0.5.

- In general, if an element can take on $n$ values and there are $r$ such elements with independently and uniformly assigned values, then the probability that they all have different values is (assume $r \ll n$ for the approximations)

$$
(1 - \frac{1}{n}) (1 - \frac{2}{n}) \cdots (1 - \frac{r-1}{n}) \approx \prod_{i=1}^{r-1} e^{-i/n} = e^{-\frac{r(r-1)}{2n}} \approx e^{-\frac{r^2}{2n}}.
$$

- Thus, the probability that two or more items out of $r$ have the same value is $p \approx 1 - e^{-\frac{r^2}{2n}}$ and if $r \approx 1.18 \sqrt{n}$ then $p \approx 0.5$.

- If a function $h(.)$ can take on $n$ values, then a collision ($h(x_1) = h(x_2), x_1 \neq x_2$) occurs with probability 0.5 after $\approx \sqrt{n}$ trials.
Yuval’s Birthday Attack

- **INP:**  
  \( m_1 \) legitimate message.  
  \( m_2 \) fraudulent message.  
  \( h \) \( b \)-bit hash function.

- **OUTP:**  
  \( m'_1, m'_2 \) resulting from minor modifications of \( m_1, m_2 \) with \( h(m'_1) = h(m'_2) \).

1. Generate \( t = 2^{b/2} \) minor modifications \( m'_1 \) of \( m_1 \).
2. Compute and store \( h(m'_1) \) for all \( m'_1 \).
3. Generate minor modifications \( m'_2 \) of \( m_2 \) and compute \( h(m'_2) \) until match in \( h(m'_1) \) table is found.

- \( p(\text{success}) \geq 0.5 \) if approximately \( \sqrt{2^b} = 2^{b/2} \) hash functions \( h(m'_1) \) and \( h(m'_2) \) are checked.

- \( \Rightarrow \) effort \( \approx 2 \times 2^{b/2} \) rather than \( 2^b \) required.

- Example: \( 2^{128} = 3.4 \times 10^{38} \) but \( 2 \times 2^{64} = 3.7 \times 10^{19} \). One year = \( 3.15 \times 10^{16} \) ns.
Meet in the Middle Attacks

▶ A meet in the middle attack is more flexible than a birthday attack.

▶ Birthday attack example (for comparison): Consider a system that uses a MAC with a 64-bit key on messages with standard headers (e.g., “People’s Bank of South Dakota, this is transaction number ...”). Each message uses a new randomly chosen key. The attacker stores the MAC values of all headers and the messages that follow in a table. After about $2^{32}$ messages there is a $\geq 50\%$ chance that two header MAC values are the same, and thus the two corresponding messages (which use the same MAC key as the header) can be exchanged without detection, thereby compromising the MAC system.
Meet in the Middle Attacks

- Meet in the middle attack example: Consider the same MAC system as for the birthday attack. Now the attacker chooses $2^{32}$ MAC keys at random, computes the MAC for the standard header and stores both the MAC and the key in a table. Then the attacker listens to each message and checks the MAC of the standard header against the stored MACs. If the MAC is in the table, then it was most likely computed using the same key as the one stored in the table. Now the attacker can insert an arbitrary message since he knows the MAC key. The attacker has precomputed the MAC on 1 out of every $2^{32}$ keys on the average. Thus, after listening to $2^{32}$ transactions, he can expect to see a MAC that matches one of the stored MACs and thus the effort is about $2 \times 2^{32}$. 
**Meet in the Middle Attack on 2-DES**

Double encryption with two keys $K_1$, $K_2$. Does this increase effective key size?

Key observation: $x$ can be written as $x = E(K_1, m)$ and as $x = D(K_2, c)$ $\implies$ use known plaintext meet in middle attack.

As shown on next slide, effective key length for 2-DES is only increased from 56 to 57 bits. $\implies$ Need to use at least 3-DES to see improvement over 1-DES.
Meet in the Middle Attack on 2-DES

Assume two plaintext/ciphertext pairs \((m_a, c_a)\) and \((m_b, c_b)\) are known and keylength of \(K_1, K_2\) is \(L\) for each. Algorithm:

1. For each possible \(K_1\) compute \(x_{K_1} = E(K_1, m_a)\) and store \((x_{K_1}, K_1)\). Requires \(2^L\) storage locations.

2. For each \(K_2\) compute \(x_{K_2} = D(K_2, c_a)\) and look for \(x_{K_2} = x_{K_1}\) in table. When a match is found, check that \(K_1, K_2\) were used for encryption by computing \(E(K_2, E(K_1, m_b))\) and comparing it to \(c_b\). If check fails continue computing \(D(K_2, c_a)\) and checking in table.

Need to compute at most \(2 \times 2^L = 2^{L+1}\) encryptions/decryptions. Thus, effective keyspace size only increased from \(L\) to \(L + 1\) (from 56 to 57 bits for 2-DES).
Cryptographic Protocols

- A protocol is a series of steps, involving two or more parties, designed to accomplish a task.
- A protocol has a well-defined sequence from start to finish. Every step must be executed in turn, and no step can be taken before the previous step is finished.
- At least two people are required to complete the protocol, one person alone does not make a protocol.
- A cryptographic protocol is a protocol that uses cryptography. The point of using cryptography is to prevent or detect eavesdropping or cheating.
Coin Flipping over Telephone

Alice and Bob have agreed on

(i) \( x \) having property \( P \) means “heads”, otherwise result is “tails”.
(ii) One-way function \( x \rightarrow h(x) \), \( x \leftrightarrow h(x) \) such that \( h(x) \) does not reveal property \( P \).

Protocol:

1. Alice picks random \( x \), computes \( h(x) \) and reads it to Bob over the phone. This is Alice’s commitment. She knows whether \( x \) has property \( P \) or not.
2. Bob tells Alice his guess whether \( x \) has property \( P \) or not. This is Bob’s commitment.
3. Alice reads \( x \) to Bob. This is the first verification step.
4. Bob checks \( x \) for property \( P \) and verifies \( h(x) \). This is the second verification step.
Coin Flipping over Telephone

- For Alice’s committment to work it must be impossible to find $x'$ with property $P^c(x)$ such that $h(x') = h(x)$.
- Note that Alice could try to influence the result by biasing “heads” and “tails”.
- Examples for $P$ and $h(x)$:
  - $P$: $x$ is even. $h(x) = x^2 \pmod{n}$ for $n = pq$, $p, q, p \neq q$, large primes with $p = 3 \pmod{4}$ and $q = 3 \pmod{4}$.
  - $h(x) = pq$ where $p, q$ are large primes, with $p = 1 \pmod{4}$ and $q = 3 \pmod{4}$. $x$ has property $P$ if $p < q$. 
Identification or Entity Authentication

- Basis:
  1) Something known, like password, PIN, or secret encryption key.
  2) Something possessed, like passport, chip card, or magnetic striped card.
  3) Something inherent, like a fingerprint or a retinal pattern.

- Identification protocol can provide unilateral or mutual identification.

- Difference between entity and message authentication:
  - Entity authentication is real-time, does not need meaningful message.
  - Message authentication applies to meaningful message, not timeliness.
Passwords (Weak Authentication)

- Claimant presents password $pw$ and verifier checks correctness against $pw$ stored in database.
- Susceptible to dictionary attacks, replay attacks, database attacks.
- To prevent database attack (stealing passwords directly from database) use one-way function $h$ and store only $h(pw)$ in database.
- To make dictionary attacks (trying all the strings that people typically use as passwords) more difficult, use random number $s$ (“salt”) and store $s$ and $h(pw, s)$ in database.
- To prevent replay attacks, use sequence of one-time passwords.
Challenge-Response Identification

- To obtain strong authentication, real-time interaction between claimant \((C)\) and verifier \((V)\) is needed. Example using public-key cryptography:

\[
\begin{align*}
C & \quad a_1 \quad V \\
& \quad a_1
\end{align*}
\]

\[
a_1 = E(K^V_e, r_1, C)
\]

\[
C \quad a_2 \quad V
\]

\[
r_1, r_2 = D(K^C_d, a_2)
\]

\[
\text{verify } r_1
\]

\[
C \quad r_2 \quad V
\]

\[
a_2 = E(K^C_e, r_1, r_2)
\]

\[
\text{verify } r_2
\]