Lecture 2: Secret Key Cryptography

Helger Lipmaa
Helsinki University of Technology
helger@tcs.hut.fi
Reminder: Communication Model

\[
E
\text{Cipher, Encryption}
\]
\[
E^{-1}
\text{Inverse cipher, Decryption}
\]

- \(M\) is the plaintext
- \(C = E_K(M)\) is the ciphertext
- \(M = E_K^{-1}(E_K(M))\)
- \(K\) is the preshared key

Alice: Sender
Bob: Receiver
Eve: Adversary

Public channel
Private channel
Block Ciphers

- A function $E : \mathcal{K} \times \mathcal{P} \rightarrow \mathcal{C}$

- $\mathcal{K}$—the key space, $\mathcal{P}$—the plaintext space, $\mathcal{C}$—the ciphertext space

- $E(k, x)$ is often denoted as $E_k(x)$

- $E_k$ is permutation: $(\forall x)E_k^{-1}(E_k(x)) = x$. 
Block Ciphers, cont.

- Usually $\mathcal{P} = \mathcal{C} = \{0, 1\}^n$, $\mathcal{K} = \{0, 1\}^k$

- $n$ is the block length, $k$ is the key length

- If $k$ is small, then key can be found by exhaustive search

- If $n$ is small, one can use known-plaintext attack (store all seen plaintext-ciphertext pairs)
Block Ciphers, cont.

- Exhaustively searching $k$-bit keys takes $2^k$ time units

- Storing sufficient amount of plaintext-ciphertext pairs takes $2^n$ memory units

- Birthday attack: $2^{n/2}$ memory units sufficient

- Recommendations: key $k \geq 80$ bits

- Recommendations: block $n \geq 128$ bits
Reminder: Substitution ciphers

- Input and output belong to some set $A$ with $|A| = n$

- Key is a permutation $\pi$ on $(1, 2, 3, \ldots, n)$

- Different “letters” are permuted, according to the key: $A \rightarrow C$, $B \rightarrow X$, $C \rightarrow R$, ...

- Examples: Caesar cipher, shift ciphers, …
Substitution ciphers, cont.

- There are $2^n!$ permutations

- Storing an arbitrary permutation takes $\log_2(2^n!)$ bits

- By Stirling formula, $x! \approx \sqrt{2\pi x} \left(\frac{x}{e}\right)^x$

- Thus, the key length would be $k = \log_2(2^{128}!) \approx 2^{134}$ bits, if $n = 128$

- Clearly impractical! (Compare with the lower bound of 80 bits)
Ultimate goal: pseudorandom permutations

- Have a small key of $k$-bits ($80 \leq k \leq 256$)

- Cipher $E$ should consist of a set of $2^k$ permutations $\{E_k\}$ out of the total $2^n!$ permutations

- For an attacker who does not know the key, the permutation $E_k$ should look “random”

- That is, deciding whether some permutation $\pi$ is one of the chosen $2^k$ permutations should be hard (take $\approx 2^k$ steps)
Permutation ciphers

- Input belongs to $A^n$ for some set $A$.

- Key is a permutation $\pi$ on $(1, 2, 3, \ldots, n)$

- Different “letters” are permuted, according to the key.

- Decryption: apply inverse permutation

- Very weak by itself!
Example

\[ A = \mathbb{Z}_{26}, \, n = 2, \, \text{and} \, \pi(1) = 2, \, \pi(2) = 1. \, \text{A simple example:} \]

willwehaveabreak

XXXXXXXXX

iwllewahevbaerka

T-79.159 Cryptography and Data Security, 28.01.2004 Lecture 2: Secret Key Cryptography, Helger Lipmaa
Product ciphers

Idea: combine two weak ciphers to get a stronger cipher

Tweak: Use the SAME cipher but with different keys (Question: Why this is not a good idea with the already shown ciphers?)

Tweak II: generate $K'$ from $K$ by using some sophisticated key extension algorithm.
Substitution-Permutation Networks

Divide the block into small $s$-bit chunks
Apply a fixed substitution to every small chunk
Apply a (key-dependent) permutation to the combined output
Do this in $r$ rounds
The bit-permutations mix outputs from different S-boxes
Some cleverness should be involved to guarantee reversibility

Hybrid: Round = Substitutions + Permutation, and then multiple rounds
Feistel ciphers

Round 1

\[ L_0, R_0 \]

- \[ f \) — “suitable” function
- \[ K_1 \) — round key
- \[ L_i = R_{i-1} \]
- \[ R_i = L_{i-1} \oplus f(K_i, R_{i-1}) \]

Ciphertext: \( (R_r, L_r) \)

Decryption: same but with the order of round keys reversed

Round 2

\[ L_1, R_1 \]

Round \( r \)

\[ L_{r-1}, R_{r-1} \]

- \[ K_r \) — round key
- \[ f \) — “suitable” function

It is proven that a Feistel cipher with many rounds is secure if \( f \) is a pseudorandom function

T-79.159 Cryptography and Data Security, 28.01.2004 Lecture 2: Secret Key Cryptography, Helger Lipmaa
DES (1/2)

- In 1973, NBS published a solicitation for a cryptosystems

- One suitable candidate raised: DES (by IBM)

- DES first published in 1975

- Adapted as a standard for “unclassified” communication on January 15, 1977.

- Now superseded by AES


DES (2/2)

• Being the first ever published government-endorsed cryptosystem, DES sparkled a great controversy but also genuine interest

• Wide user-base

• Birth of public cryptanalysis of block ciphers: new methods developed in early 90s to break DES have been used to break many other ciphers

• It seems that DES is essentially secure: best attack requires $\approx 2^{40}$ known plaintext-ciphertext pairs

• Is $2^{40}$ secure? Is $2^{56}$ secure?
DES: Description

- A block cipher with 56-bit key, 64-bit block
- Apply a fixed permutation IP to the plaintext $x$
- Apply a 16-round Feistel cipher to IP($x$)
- Apply the inverse permutation $IP^{-1}$
- Keys $K_i$ are derived from $K$ by using key extension algorithm
General Scheme: Function $f(A, J)$, where $A = R_i$
### DES Components

- $E : \{0, 1\}^{32} \rightarrow \{0, 1\}^{48}$: Expansion function. Permutes 32 bits with duplicating half of them.

- $S_i : \{0, 1\}^6 \rightarrow \{0, 1\}^4$: $i$th S-box. A nonlinear function.

- $P$: Bit Permutation. Changes bit locations.

- Note that $E$, $S_i$, $P$ do not depend on the key!
DES: Quick evaluation (1/2)

- Suffers from short key-length: $2^{56}$ DES operations (for exhaustive search) is currently feasible.

- Key complementation property, $E_K(x) = E_K(\overline{x})$, decreases this to $2^{55}$

- ... DES key has been found by using special hardware in 3.5 hours (1999, see http://www.eff.org/descracker/)
DES: Quick evaluation (2/2)

- Best attack: linear cryptanalysis (Matsui 1994, later improved by others), requires $\approx 2^{40}$ known plaintext-ciphertext pairs

- Relatively slow in software: 18 MByte/s on a 800 MHz Pentium

- Very fast in hardware: multi-gigabyte range (designed for hardware)
Differential Cryptanalysis: History

- The first publicly known successful attack against DES (Biham and Shamir, 1990)

- ... who found DES to be surprisingly strong against the DC

- Don Coppersmith (IBM) later admitted that the designers knew this attack when they designed DES and took it into consideration
Differential Cryptanalysis

- A chosen plaintext attack: $n$ plaintext pairs $(x[i], x^*[i]), i \in [1, n]$ are chosen, so that $x[i] \oplus x^*[i] = \Delta x$

- If $\Delta x$ is well chosen then for some $\Delta y$, $E_K(x[i]) \oplus E_K(x^*[i]) = \Delta y$ with a high probability $p$

- We say that $(\Delta x \rightarrow \Delta y)$ has a differential probability $p$

- Use most probable differentials to select some keys as more probable

- Protection: design cipher not to have highly probable differentials
AES

- A competition for the new standard was announced in 1997

- This time, an open competition and 15 candidates participated

- MARS (IBM), RC6 (RSA Labs), Rijndael (Joan Daemen and Vincent Rijmen), Serpent (Anderson, Biham, Knudsen) and Twofish (Counter-pane) were selected to the second round

- All five ciphers were found to be sufficiently secure and in late 2000, Rijndael was selected as a winner based on its versatility and clear design principles
AES algorithm (Rijndael): Overview

- Has 128-bit blocks and 128, 192 or 256-bit keys

- The number of rounds depends on the key-length, being 10, 12 or 14

- Specifically designed to be secure against the differential and linear cryptanalysis

- Fast: more than 53 MByte/s on a 800 MHz Pentium

- See http://www.nist.gov/aes for more
AES: Description

- DES: main operations are XOR, bit permutations and S-boxes (fast in hardware, slow in software)

- AES: main operations are operations in finite field GF(2^8) and S-boxes (fast in both hardware and software)

- One round consists of the next operations: SubBytes (S-box), ShiftRows, MixColumns (make up the permutation) and AddRoundKey
AES: High Level Overview

Like general SPN
AddRoundKey — only dependence on keys
SubBytes: $8 \times 8$ S-box (byte substitution)
ShiftRow: permutation of bytes
MixColumns: matrix multiplication of 8-bit finite field elements
$P$ consists of ShiftRow and MixColumns
Last row is slightly different
Decryption has InverseMixColumns (different matrix)

Hybrid: Round = Substitutions + Permutation, and then multiple rounds
One-time pad

\[
\begin{array}{ccccccccccccccc}
0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
\oplus \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\
\end{array}
\]

All these key bits are random!

\[
0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1
\]

Plaintext \( x \)

Key \( k \)

Ciphertext \( y \)

Perfectly secure: if key is random then ciphertext is random. For every key there exists a plaintext that encrypts to this ciphertext. Thus, no information about plaintext is leaked

Bad: every perfectly secure cipher requires \(|x| = |k| = |y|\). Impractical!

How to improve?

T-79.159 Cryptography and Data Security, 28.01.2004 Lecture 2: Secret Key Cryptography, Helger Lipmaa
Stream cipher

Idea: generate a long pseudorandom (random-looking) sequence out of the short seed
Stream cipher

<table>
<thead>
<tr>
<th>Already seen plaintext ( x )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 0 1 1 0 0 1 0 1 0 0</td>
</tr>
<tr>
<td>0 0 1 0 1 1 0 0 0 0 1</td>
</tr>
</tbody>
</table>

\[
\text{Random } k \quad \text{Already generated key } G(k, x)
\]

\[
\begin{array}{cccccccccc}
0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
\end{array}
\]

\[
0 \oplus 0 = 0
\]

\[
0 \oplus 1 = 1
\]

That is, key stream might be a function of plaintext.
Stream ciphers: pros

- Do not have to be reversible
  - Block ciphers are reversible. This involves increased cost. Stream ciphers are potentially faster

- Intuitively clear what it means for a stream ciphers to be secure: output string is indistinguishable from a random string

- Stream cipher $\approx$ cryptographically strong pseudo-random number generator
Contemporary stream ciphers

- Classical approach, LFSR (Linear Feedback Shift Register), insecure

- Combine two LFSRs by using a well-chosen non-linear function (seen in many ciphers)

- Contemporary ciphers use very different approaches

- While some of stream ciphers are in wide use (RC4, e.g.,), they are far less studied than block ciphers
Contemporary stream ciphers

- RC4: ‘broken” (must discard at least 1024 bytes of the generated key stream), Seal: broken, etc.

- NESSIE project issued a call for stream ciphers. All candidates are broken

- Most efficient attack against the NESSIE candidate LILI128 is by Markku-Juhani Saarinen

- Some secure(?) stream ciphers: Wake, and some new proposals
Why such an situation? (1/2)

- Design philosophy: it’s secure if it is not broken!

- The game of cats and mice between cryptographers and cryptanalysts

- ...Attack, Correct, Attack, Correct, ...

T-79.159 Cryptography and Data Security, 28.01.2004 Lecture 2: Secret Key Cryptography, Helger Lipmaa
Why such an situation? (2/2)

- It would be desirable to have a provably secure cipher

- Unfortunately, provably secure ciphers tend

  1. to have a long key: OTP; or

  2. are very slow (public-key cryptosystems are 1000x slower than AES, RC4, . . . )

- Ciphers, provably secure in some situations are very weak in some others