Symmetric Encryption

Viet Tung Hoang

The slides are loosely based on those of Prof. Mihir Bellare, UC San Diego.
1. Modes of Encryption: ECB, CBC, CTR
2. Formalizing Security
3. Stream Ciphers
Encryption Syntax

Key Gen

\[ \mathcal{K} \xrightarrow{\$} K \]

Encrypt

\[ M \xrightarrow{\$} \mathcal{E} \xrightarrow{K} C \]

Encryption can be probabilistic

Decrypt

\[ C \xrightarrow{} \mathcal{D} \xrightarrow{K} \begin{cases} M \text{ or } \bot \end{cases} \]
(Bad) Encryption Using Blockcipher: ECB

\[ E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n \]

Can encrypt any message whose length is a multiple of \( n \)
ECB Is **Insecure**

Message  
ECB ciphertext  
Properly encrypted ciphertext
Why Is ECB So Bad?

If $M_i = M_j$ then $C_i = C_j$
ECB Horror Stories

Half the apps in Android used ECB to encrypt data

An Empirical Study of Cryptographic Misuse in Android Applications

Adobe used ECB to encrypt passwords

Zoom concedes custom encryption is substandard as Citizen Lab pokes holes in it

Zoom used ECB to encrypt video conferencing
Randomized Encryption: CBC

randomly chosen

\[ C_0 \rightarrow M_1 \rightarrow E_K \rightarrow C_1 \]
\[ M_2 \rightarrow E_K \rightarrow C_2 \]
\[ M_3 \rightarrow E_K \rightarrow C_3 \]
\[ M_4 \rightarrow E_K \rightarrow C_4 \]
Decryption of CBC
Dealing with Fragmentary Data

**Naive solution:** Pad with \(10^*\)

**Example:** Suppose that the block length is 16 bytes.

Padding is required, otherwise can’t decrypt

**Problem:** Waste bandwidth, and for full-length msg, waste a blockcipher call
Ciphertext Stealing in CBC

Exercise: How to use ciphertext stealing if msg is shorter than 1 block?
Randomized Encryption: CTR

\[ C_0 + 1 \]
\[ E_K \]
\[ C_0 + 2 \]
\[ E_K \]
\[ C_0 + 3 \]
\[ E_K \]

randomly chosen

\[ C_0 \]
\[ M_1 \]
\[ C_1 \]
\[ M_2 \]
\[ C_2 \]
\[ M_3 \]
\[ C_3 \]

fully parallelizable
Dealing with Fragmentary Data

randomly chosen

$C_0$ → $M_1$ → $E_K$ → $C_1$

$C_0 + 1$

$E_K$ → $C_0 + 2$

$C_0 + 3$

truncation

$M_3$ → $E_K$ → $C_3$
1. Modes of Encryption: ECB, CBC, CTR

2. Formalizing Security

3. Stream Ciphers
Formalizing Security: Intuition

Should hide all partial information about the plaintexts

Except message length

CBC trivially leaks message length
Formalizing Security: Informal Definition

Adversary can’t even distinguish the encryption of its **own chosen messages**

“A good disguise should not allow a mother to distinguish her own children”

Goldwasser and Micali
Formalizing Security: Left-or-Right

Left $\mathcal{E}$

procedure $\text{Enc}(M_0, M_1)$
Return $\mathcal{E}_K(M_0)$

Right $\mathcal{E}$

procedure $\text{Enc}(M_0, M_1)$
Return $\mathcal{E}_K(M_1)$

In each query, the two messages must have the same length

$$\text{Adv}^\text{lr}_{\mathcal{E}}(A) = \Pr[\text{Right}_A^\mathcal{E} \Rightarrow 1] - \Pr[\text{Left}_A^\mathcal{E} \Rightarrow 1]$$
Formalizing Security: Real-or-Random

\[
\text{Real } \mathcal{E} \\
\text{procedure } \text{Enc}(M) \\
\text{Return } \mathcal{E}_K(M) \\
\]

\[
\text{Rand } \mathcal{E} \\
\text{procedure } \text{Enc}(M) \\
C \leftrightarrow \mathcal{E}_K(M'); C' \leftrightarrow \{0, 1\}^{|C|}; \text{ Return } C' \\
\]

\[
\text{Adv}_{\mathcal{E}}^{rr}(A) = \Pr[\text{Real}_{\mathcal{E}}^A \Rightarrow 1] - \Pr[\text{Rand}_{\mathcal{E}}^A \Rightarrow 1] \\
\]
Exercise: Break LR Security of ECB

\[ M_1 \xrightarrow{E_K} C_1 \] 
\[ M_2 \xrightarrow{E_K} C_2 \] 
\[ M_3 \xrightarrow{E_K} C_3 \] 
\[ M_4 \xrightarrow{E_K} C_4 \]
Question: Break the real-or-random security of this scheme using a single query of a 2-block message.
1. Modes of Encryption: ECB, CBC, CTR

2. Formalizing Security

3. Stream Ciphers
Real-world Broken Stream Ciphers

RC4

Encryption scheme for Web traffic and Wifi traffic

A5/1 Stream cipher

Cellular encryption in GSM networks

E0 Stream cipher

Bluetooth encryption
Syntax

Initialization
- IV
- $K$

Generation
- $s$
- Gen
- $s'$
- $P$

Optional
Use of Stream Cipher

Producing A Stream of One-Time Pad

Question: Formalize a security notion for stream cipher
A Wrong Construction
Linear Feedback Shift Register (LFSR)

Question: Given $n$ bits of output, recover subsequent bits
Case Study: DVD Encryption System

- $K$: 40 bits
- $s_0$: 16 bits
- $s_1$: 24 bits

**Init**

- 17-bit LFSR
- 8-bit $X$

**Gen**

- 25-bit LFSR
- 8-bit $Y$

**Comb**

- 8-bit $Z$
Property of Combiner To Exploit

\[
\begin{align*}
&X 
\rightarrow \text{Comb} 
\rightarrow Z \\
&Y 
\rightarrow \text{Comb} 
\rightarrow Z
\end{align*}
\]

**Invertibility**: Given \(Z\) and \(X\), it’s trivial to compute \(Y\)
Question: Given the first 128 bits of output, recover subsequent bits using $O(2^{16})$ time by guessing the initial $s_0$. 
Building Stream Cipher From Blockcipher

Init sets $IV = 0$ and outputs $(IV, K)$ as the initial state
How Encryption Looks Like: Stateful CTR

Ciphertext doesn’t include IV
Sender and receiver update $IV \leftarrow IV + 3$ for the next encryption