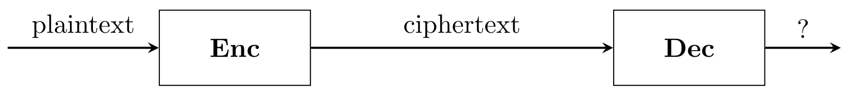
Cryptography

7 Security against chosen-plaintext attacks



increasing adversary power $\downarrow$ $\left.\begin{array}{l}\text{known-plaintext}\\ \text{chosen-plaintext}\\ \text{chosen ciphertext}\end{array}\right\}$ attack

One-time security
- not very useful
- chooses a fresh key per encryption cell
  $\Rightarrow$ relax this!

Definition

A encryption scheme Σ is secure against chosen-plaintext attacks if $L^{\Sigma}_{cpa-L} \approx L^{\Sigma}_{cpa-R}$, where:

L^{Σ}_{cpa-L}	L^{Σ}_{cpa-R}
$k \leftarrow \{0,1\}^{\lambda}$	$k \leftarrow \{0,1\}^{\lambda}$
EAVESDROP(m_L, m_R)	EAVESDROP(m_L, m_R)
if $\mid m_L \mid \neq \mid m_R \mid$	if $\mid m_L \mid \neq \mid m_R \mid$
then return ERROR	then return ERROR
$c := \Sigma.Enc(k, m_L)$	$c := \Sigma.Enc(k, m_R)$
return c	return c

NOTE 1: Lengths must be equal. That allows Σ to be used for plaintext of different lengths:
$\Sigma.M = \{0,1\}^{\star}, \quad \mid m \mid = \textit{length in bits}$

- Traffic analysis reveals information about plaintext sizes
- Steganography hides the existence of a hidden message

NOTE 2: Often called IND-CPA security (indistinguishable CPA)

NOTE 3: Almost same notion for public key crypto.

Lemma

CPA-secure encryption schemes cannot be deterministic.

Proof

Suppose it is:
Then:

$$
\begin{aligned}
&c_x := EAVESDROP(x, x);\\
&c_y := EAVESDROP(x, y);\\
&\textbf{return } c_x \stackrel{?}{=} c_y
\end{aligned}
$$

distinguishes between L^{Σ}_{cpa-L} and L^{Σ}_{cpa-R}.
Need probabilistic encryption!

How to make encryption non-deterministic?

1. Stateful encryption
   - Keep state (counter) inside Σ.Enc()
   - Complex to implement:
     requires synchronisation between Enc() and Dec()
2. Randomization in encryption algorithm
   - Σ.Enc uses randomness r
   - r becomes part of ciphertext:
     ⇒ increases length
   - most popular
3. Nonce-based encryption
   - add a nonce to Σ.Enc()
   - nonce: number used once
   - caller must ensure that Enc() is never called with the same nonce twice

Pseudorandom ciphertext

1. Second notion for CPA-secure symmetric encryption
2. Often more useful than CPA

Definition

An encryption scheme Σ has pseudorandom ciphertexts against chosen-plaintext attacks if $L^{\Sigma}_{cpa\$-real} \approx L^{\Sigma}_{cpa\$-rand}$

$$\boxed{\begin{array}{l} \underline{L^{\Sigma}_{cpa\$-real}} \\ k \leftarrow \{0,1\}^{\lambda} \\ \\ \underline{\text{CTXT}(m)} \\ \quad c := \Sigma.Enc(k,m) \\ \quad \text{return c} \end{array}} \qquad \boxed{\begin{array}{l} \underline{L^{\Sigma}_{cpa\$-rand}} \\ \\ \underline{\text{CTXT}(m_L, m_R)} \\ \quad r \leftarrow \Sigma.C \mid_{length(r)=length(\Sigma.Enc(k,m))} \\ \quad \text{return r} \end{array}}$$

Lemma

CPA\$-security $\Rightarrow$ CPA-security ($\not\Leftarrow$)

Proof

Exercise

CPA-secure encryption from a PRF

<u>Idea:</u> Use $F(k,r)$ with a different r for each encryption call.
<u>How to make r distinct?</u>

- statful encryption: r: counter/state, complex
- Randomized: $r \leftarrow \{0,1\}^{\lambda}$
- Delegate it: use nonce, $r := nonce$

Construction:

CPA-secure Σ from PRF F

$$F : \{0,1\}^{\lambda} \times \{0,1\}^{\lambda} \rightarrow \{0,1\}^{len}$$

$$\begin{array}{lcl} \Sigma.M & = & \{0,1\}^{\lambda} \\ \Sigma.C & = & \{0,1\}^{\lambda} \times \{0,1\}^{len} \\ \Sigma.K & = & \{0,1\}^{\lambda} \end{array} \qquad \begin{array}{l} \underline{\text{KeyGen():}} \\ \quad k \leftarrow \{0,1\}^{\lambda} \\ \quad \textbf{return } k \end{array}$$

$$\begin{array}{l} \underline{\Sigma.\text{Enc(k,m):}} \\ \quad r \leftarrow \{0,1\}^{\lambda} \\ \quad x := F(k,r) \oplus m \\ \quad \textbf{return } \text{F(r,x)} \end{array} \qquad \begin{array}{l} \underline{\Sigma.\text{Dec(r,x):}} \\ \quad \textbf{return } F(k,r) \oplus x \end{array}$$

Lemma

If F use a secure PRF then construction has CPA\$-security.

Proof idea

$$\boxed{\begin{array}{l} \underline{L^{\Sigma}_{cpa\$-real}} \\ k \leftarrow \{0,1\}^{\lambda} \\ \\ \underline{\textsc{CTxt}(m)} \\ \quad r \leftarrow \{0,1\}^{\lambda} \\ \quad x := F(k,r) \oplus m \\ \quad \textbf{return } (r,x) \end{array}} \qquad \boxed{\begin{array}{l} \underline{L^{\Sigma}_{cpa\$-rand}} \\ r \leftarrow \{0,1\}^{\lambda} \\ x \leftarrow \{0,1\}^{len} \\ \textbf{return } (r,x) \end{array}}$$

8 Block ciphers in practice

- Blockcipher as a PRP
- How to encrypt long messages?
- **Mode of operation**
  Needed to implement CPA-secure encryption of long messages.
- **Padding scheme**