Lecture 5 Formalism of Cryptosystems §1.7

Def A symmetric cryptosystem is a tuple

$(\mathcal{K}, \mathcal{M}, \mathcal{C}, e, d)$ where

$\mathcal{K}$: Set of possible keys

$\mathcal{M}$: Set of possible plaintexts (messages)

$\mathcal{C}$: Set of possible ciphertexts

$e : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ encryption function

$e_k : \mathcal{M} \to \mathcal{C}$ for fixed key

$d : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$ decryption function

$d_k : \mathcal{C} \to \mathcal{M}$

s.t. $d_k(e_k(m)) = m$ for all $k \in \mathcal{K}$, $m \in \mathcal{M}$.

Properties Required for $(\mathcal{K}, \mathcal{M}, \mathcal{C}, e, d)$ to be a "good cryptosystem":

Practical

1. e_k is easy to compute.
2. d_k is easy to compute
3. $\mathcal{K}$ is a reasonable size

Secure

3. Given $c_1, \ldots, c_n$ messages encrypted w/ a key k hard to compute $d_k(c_1), \ldots, d_k(c_n)$ without knowing k.

4. (Known plaintext attack)

   Given $(m_1, c_1), \ldots, (m_n, c_n)$ plaintext, ciphertext pairs $(c_i = e_k(m_i)$ for a fixed $k \in \mathcal{K})$ hard to compute $d_k(c)$ if c not in the list.

eg Substitution cipher doesn't satisfy 4 b/c if every letter is contained in some m_i, know the key.

5. (chosen plaintext attack)

For any chosen $m_1, \dots, m_n$ and

$c_1 = e_k(m_1), \dots, c_n = e_k(m_n)$,

hard to compute $d_k(c)$ for $c \notin \{c_1, \dots, c_n\}$,

e.g. substitution cipher doesn't satisfy this,

Choose $m = ab \dots z$.

Same with Vignere cipher:

look at $m = aaa \dots a$

Encodings

On a computer, everything is a lump of numbers. to allow text to be represented. Most common encodings are ascii & utf-8.

ascii is only good for english, utf-8 supports all languages.

ascii is simple: each character is a single byte.

65	66	---	91
a	b		z
97	98	---	122
A	B		Z

So

$a \rightsquigarrow 0100 \quad 0001$

$b \rightsquigarrow 0100 \quad 0010$

$\vdots$

$A \rightsquigarrow 0110 \quad 0001$

$B \rightsquigarrow 0110 \quad 0010.$

$\vdots$

text $\rightsquigarrow$ ascii encoding (or utf-8, utf-16, ...)

Cut into blocks of B bits.

to prevent brute force attacks pick $B_k \geq 160$.

<u>e.g.</u> $B=4$, $m=$ "bed" $\longleftrightarrow$

$$\overbrace{\underset{4}{0100}\quad \underset{2}{0010}}^{b}\qquad \overbrace{\underset{4}{0100}\quad \underset{5}{0101}}^{e}\qquad \overbrace{\underset{4}{0100}\quad \underset{4}{0100}}^{d}$$

Each block is a string of B bits.

Examples of symmetric ciphers

Pick a prime $p \sim 2^{160}$ (public info)

① (Addition mod p / shift cipher)

$\mathcal{K} = \mathbb{Z}/p\mathbb{Z} = \mathcal{M} = \mathcal{C}$.

$e_k(m) = m + k$

$d_k(c) = c - k$

Satisfies

1. Easy to compute e_k
2. — — — — d_k
3. Given c, hard to find m w/o k.

For any $m, c \in \mathbb{Z}/p\mathbb{Z}$, $k = m - c$ makes $e_k(m) = c$. So one c by itself could come from any message.

4. Known Plaintext attack

Vulnerable: $m, c \longrightarrow k = m - c$.

(2) Multiplication mod p

$$\mathcal{K} = (\mathbb{Z}/p\mathbb{Z})^\times = \mathcal{M} = \mathcal{C}$$

$$e_k(m) = km$$

$$d_k(m) = k^{-1}m.$$

Similar to (1), vulnerable to known plaintext.

Better than (1) at

(3) Affine transform → combination of (1) & (2).

$$\mathcal{K} = (\mathbb{Z}/p\mathbb{Z})^\times \times \mathbb{Z}/p\mathbb{Z} \ni k = (k_1, k_2)$$

$$\mathcal{M} = \mathcal{C} = \mathbb{Z}/p\mathbb{Z}$$

$$e_{(k_1,k_2)}(m) = k_1 m + k_2$$

$$d_{(k_1,k_2)}(c) = k_1^{-1}(m - k_2)$$

ex $p = 37$, $(k_1, k_2) = (2, 3)$ $m = 4$

$$c = k_1 m + k_2 = 2 \cdot 4 + 3 = 11 \in \mathbb{Z}/37\mathbb{Z}$$

$$m = k_1^{-1}(c - k_2) = 2^{-1}(11 - 3)$$

$$= 19 \cdot 8 = 152 = 4$$

Still vulnerable to known plaintext, just need two instead of one. also doubled key size.

④ Hill Cipher (vector version of ③)

$$K = \underbrace{GL_n(\mathbb{F}_p)}_{\substack{n\times n \text{ invertible} \\ \text{matrices w/} \\ \text{entries in } \mathbb{F}_p}} \times \mathbb{F}_p^n \ni (k_1, k_2)$$

Vulnerable to known plaintext attack if we have $n+1$ (m_i, c_i) pairs.

$$\begin{cases} c_1 = k_1 m_1 + k_2 \\ \vdots \\ c_{n+1} = k_1 m_{n+1} + k_2 \end{cases} \Rightarrow \begin{cases} c_2 - c_1 = k_1(m_1 - m_{n+1}) \\ \vdots \\ c_{n+1} - c_1 = k_1(m_n - m_{n+1}) \end{cases}$$

if $(c_2 - c_1), \ldots, (c_{n+1} - c_1)$ form a basis for $\mathbb{F}_p^n$, can solve for k_1.