3/19

Table of Contents

Block Cipher（分组密码）

How it works?

$$\text{Plain Text Block(n bits)} \rightarrow (E, D) \rightarrow \text{Cipher Text Block(n bits)} \\ \uparrow \text{Key(k bits)}$$

Canonical Examples:

1. 3DES: n = 64 bits, k = 168 bits; ($\left|\frac{k}{3}\right| \leq \left|m\right|$)
2. AES(International Standard): n = 128 bits, k = 128, 192, 196 bits(Flexible, Different Levels of Security).

Built by Iteration

$$\text{key } k \rightarrow \text{key expansion} \Rightarrow \left\{ \begin{array}{ll} & \quad m \\ & \quad \downarrow \\ k_1 \rightarrow & R(k_1,\cdot) \\ & \quad \downarrow \\ k_2 \rightarrow & R(k_2,\cdot) \\ & \quad \downarrow \\ k_3 \rightarrow & R(k_3,\cdot) \\ \vdots & \quad \downarrow \\ \vdots & \quad \vdots \\ \vdots & \quad \downarrow \\ k_n \rightarrow & R(k_n,\cdot) \\ & \quad \downarrow \\ & \quad c \end{array} \right.$$

$R(k,m)$ is called a **Round Function**, iterated by a "small" function.

In this case, $n = 48$ for 3-DES while $n = 10$ for AES-128.

Every bit of input can affect output and every bit of output is related to input.

The performance is not as fast as stream cipher, so it's not appropriate for real time encryption.

Recommending Crypto++, a open-source repository

PRP(Pseudo Random Permutaion) & PRF(Pseudo Random Function)

$\textbf{Def}$: PRF defined over$(K,X,Y) : F : K \times X \rightarrow Y$
such that exists "efficient" algorithm to evaluate $F(k,x)$

$\textbf{Def}$: PRP defined over$(K,X) : F : K \times X \rightarrow X$
such that:
1. Exists "efficient" determininstic algorithm to evaluate $E(k,x)$.
2. The function $E(k,\cdot)$ is one-to-one.
3. Exists "efficient" inversion algorithm $D(k,y)$.

Functionally, any PRP is also a PRF. A PRP is a PRF where X = Y and is efficiently invertible(可逆).

Secure PRFs

$$\text{PRF: } F: K \times X \to Y$$

$$\begin{cases}\text{Funs}[X,Y]: \text{the set of all functions from } X \text{ to } Y\\ S_F = \{F(k,\cdot) \text{ s.t. } k \in K\} \subseteq \text{Funs}[X,Y]\end{cases}$$

A PRF is **secure** if a random function in $\text{Funs}[X,Y]$ (with a size of $|Y|^{|X|}$) is indistinguishable from a random function in S_F (with a size of $|K|$), so that the attacker can't distinguish $f(x)$ from $F(k,x)$.

Secure PRPs (secure block cipher)

$$\text{PRP: } E: K \times X \to Y$$

$$\begin{cases}\text{Perms}[X]: \text{the set of all one-to-one functions from } X \text{ to } Y\\ S_F = \{E(k,\cdot) \text{ s.t. } k \in K\} \subseteq \text{Perms}[X,Y]\end{cases}$$

A PRP is **secure** if a random function in $\text{Perms}[X]$ (with a size of $|Y|^{|X|}$) is indistinguishable from a random function in S_F , so that the attacker can't distinguish $\pi(x)$ from $E(k,x)$.

Let $F: K \times X \to \{0,1\}^{128}$ be a secure PRF.
Is the following G a secure PRF?

$$G(k,x) = \begin{cases}0^{128} & \text{if } x = 0\\ F(k,x) & \text{otherwise}\end{cases}$$

- NO, it is easy to distinguish G from a random function.

Application: PRF $\Rightarrow$ PRG

Let $F: K \times \{0,1\}^n \to \{0,1\}^n$ be a secure PRF.

Then the following $G: K \to \{0,1\}^{nt}$ is a secure PRG:

$$G(k) = F(k,0) \parallel F(k,1) \parallel \cdots \parallel F(k,t-1)$$

Key property: parallelizable（可并行计算，即这里可以并行地计算/生成每个长度为t的F（secure PRF），最后拼接得到需要生成的长度为nt的伪随机序列G（secure PRG））.

Security from PRF property: $F(k,\cdot)$ indist. from random function $f(\cdot)$.（即上面提到的攻击者不可区分 $f(x)$ 和 $F(k,x)$ 从而构成安全的PRF）

这样就有了生成安全PRG的办法，保障了Stream Cipher的安全性。

Review

PRF $\longrightarrow$ Block Cipher $\longrightarrow G(k) \longrightarrow$ PRG $\longrightarrow$ Stream Cipher

DES(Data Encryption Standard)

Core Idea -- Feistel Network

Given $f_1, \ldots, f_d$: $\{0,1\}^n \rightarrow \{0,1\}^n$

Goal: build invertible(可逆) function $F : \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$

$$\begin{cases} R_i = f_i(k_i, R_{i-1}) \oplus L_{i-1} \\ L_i = R_{i-1} \end{cases}$$

$$(R_i : \text{n-bit}, L_i : \text{n-bit})$$

$$\Rightarrow \begin{cases} R_{i-1} = L_i \\ L_{i-1} = R_i \oplus f_i(k_i, R_{i-1}) \end{cases}$$

Like a Round Function.

For DES, $d = 16 (\text{Even Number}), f_1, \ldots, f_{16} : \{0,1\}^{32} \rightarrow \{0,1\}^{32},$
$f_i(x) = F(k_i, x) \text{ where } k_i \text{ is from Key } K.$ Used in many block ciphers but **NOT** in AES(AES不具备 $f = f^{-1}$ 的条件).

$\textbf{Thm}$:(Luby-Rackoff ’85):
$f : K \times \{0,1\}^n \rightarrow \{0,1\}^n \text{ a secure PRF}$
$\Rightarrow \text{3-round Feistel F: } K^3 \times \{0,1\}^{2n} \rightarrow \{0,1\}^{2n} \text{ a secure PRP}$

Michael Luby 和 Charles Rackoff 证明了如果 Round Function 是使用 K_i 为种子的密码安全的伪随机函数，
那么经过三轮操作之后，生成的分组密码就已经是伪随机排列了。(CPA secure)
经过四轮操作可以生成“强”伪随机排列。(CCA secure)

S-Boxes

S-box: function $S_i : \{0,1\}^6 \rightarrow \{0,1\}^4$, implemented as Look-Up Table.

Choosing the S-boxes and P-box *at random* would result in an insecure block cipher(key recovery after $\approx 2^{24}$ outputs) [BS’89]

这里不能

> Choosing the S-boxes and P-box *at random*

意思是不能“随便地”取S-box和P-box，相对地，需要一定程度上地精心设计才能保证其安全性。否则如下例：

Suppose:

$$S_i(x_1, x_2, \ldots, x_6) = (x_2 \oplus x_3, x_1 \oplus x_4 \oplus x_5, x_1 \oplus x_6, x_2 \oplus x_3 \oplus x_6)$$

or written equivalently: $S_i(\mathbf{x}) = \mathbf{A_i} \cdot \mathbf{x} \pmod 2$

$$\begin{pmatrix} x_2 \oplus x_3 \\ x_1 \oplus x_4 \oplus x_5 \\ x_1 \oplus x_6 \\ x_2 \oplus x_3 \oplus x_6 \end{pmatrix} = \begin{pmatrix} 0,1,1,0,0,0 \\ 1,0,0,1,1,0 \\ 1,0,0,0,0,1 \\ 0,1,1,0,0,1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix}$$

We say that S_i is a linear function.

Then entire DES cipher would be linear: $\exists$ fixed binary matrix $\mathbf{B}$ s.t.

$$\mathrm{DES}(k, m) = \begin{pmatrix} b_{1,1}, & b_{1,2}, & \cdots, & b_{1,832} \\ b_{2,1}, & b_{2,2}, & \cdots, & b_{2,832} \\ \vdots & \vdots & \ddots & \vdots \\ b_{64,1}, & b_{64,2}, & \cdots, & b_{64,832} \end{pmatrix} \cdot \begin{pmatrix} m \\ k_1 \\ k_2 \\ \vdots \\ k_{16} \end{pmatrix} = \mathbf{C} \pmod 2$$

But then: $\mathrm{DES}(k, m_1) \oplus \mathrm{DES}(k, m_2) \oplus \mathrm{DES}(k, m_3) = \mathrm{DES}(k, m_1 \oplus m_2 \oplus m_3)$

PROBELMS...

> 容易被差分攻击破解？所以S-Box、P-Box的设计需要满足一定的密码学指标？
> Two-fish $\longrightarrow$ dynamic open details for S-Box?

Several rules used in choice of S and P boxes:

- No output bit should be close to a linear func. of the input bits
- S-boxes are 4-to-1 maps

> S-Box, P-Box, ..., Hardware Implementation $\longleftarrow$ Compact Computing $\longleftarrow$ Permutation(IP, IP^{-1})

Exhaustive Search Attacks(暴力破解DES（已被破解--1997，AES代替--2000）)

以及关联的并行计算对密码算法复杂度的挑战。

$\textbf{Goal:} \text{ Given a few input output pairs} (m_i, c_i = E(k, m_i)), i = 1, ..., 3, \text{find key } k.$

$\textbf{Lemma:}$ Suppose DES is an $\textbf{\textit{ideal cipher}}$
$(2^{56} \text{ random invertible functions } \pi_1, ...\pi_{256} : \{0,1\}^{56} \to \{0,1\}^{64})$
Then $\forall m, c$ there is at most one key k s.t. $c = \mathrm{DES}(k, m)$ with prob. $p \geq 1 - \frac{1}{256} \approx 99.5\%$

$\textbf{Proof:}$
$$Pr\Big[\exists k' \neq k, c = \mathrm{DES}(k, m) = \mathrm{DES}(k', m)\Big]$$
$$\leq \sum_{k' \in \{0,1\}^{56}} Pr[\mathrm{DES}(k, m) = \mathrm{DES}(k', m)] \leq 2^{56} \cdot \frac{1}{2^{64}} = \frac{1}{2^8}$$

For two DES pairs $(m_1, c_1 = \mathrm{DES}(k, m_1)), (m_2, c_2 = \mathrm{DES}(k, m_2)), \text{unicity prob.} \approx 1 - \frac{1}{271}$

For AES--128: given two inp/out pairs, unicity prob. ≈ 1 -- 1/2128
⇒ two input/output pairs are enough for exhaustive key search.

Strengthend DES

3-DES / T-DES(Triple-DES)

Define $3E : K^3 \times M \to M$ as $3E\big((k_1, k_2, k_3), m\big) = E\Big(k_1, D\big(k_2, E(k_3, m)\big)\Big)$

$(k_1 = k_2 = k_3) \Rightarrow \text{Single DES.}$

$\text{key-size} = 3 \times 56 = 168 \text{ bits}$

3 times slower than DES.

Middle Attack--Why not Double-DES

Well, there's no difference between Double-DES and Single-DES for the attacker when using middle attack -- same time cost from both side!

The same way, even numbers of expansion do nothing more than the less largest odd expansion.((2n)-DES = (2n - 1)-DES)

DESX

$E : K \times \{0,1\}^n \to \{0,1\}^n$ a block cipher.

Define EX as $EX\big((k_1, k_2, k_3), m\big) = k_1 \oplus E(k_2, m \oplus k_3)$

For DESX: $\text{key-len} = 64 + 56 + 64 = 184 \text{ bits}$

... but easy attack in time $2^{64+56} = 2^{120}$

Note: $k_1 \oplus E(k_2, m)$ and $E(k_2, m \oplus k_1)$ does nothing!!

Attacks on the implementation

Side Channel Attacks(passive): [Kocher, Jaffe, Jun, 1998]

- Measure **time** to do enc/dec(e.g. if branch in program...)
- Measure **power** for enc/dec

TC 8051 TTL

Fault Attacks(glitch, active):

- Computing errors in the **last round** expose the secret key k.

Ways? Equipments?

And many others such as Linear(Not work for AES and SM4), Differential, Quantum...

Project(?)

从AES、SM4中选择一种，两种及以上模式下的快速实现/轻量化实现，进行指标衡量、横向对比，可基于已有开源代码?