中山大學
SUN YAT-SEN UNIVERSITY

Module I. Fundamentals of Information Security

Chapter 2
Cryptographic Techniques

Web Security: *Principles & Applications*

School of Data & Computer Science, Sun Yat-sen University

Outline

- 2.1 Introduction to Cryptology
- **2.2 Symmetric Key Cryptographic Algorithms**
  - Introduction
  - Types & Modes
  - Data Encryption Standard (DES)
  - Advanced Encryption Standard (AES)
- 2.3 Mathematical Foundations of Public-Key Cryptography
- 2.4 Asymmetric Key Cryptographic Algorithms
- 2.5 Hashing Algorithms
- 2.6 Typical Applications

2.2 Symmetric Key Crypt. Algorithms

2.2.1 Introduction

- Symmetric-key cryptography is sometimes called *secret-key cryptography*. It is a kind of encryption system in which the sender and receiver of a message share a single, common key that is used to encrypt and decrypt the message.
  - Symmetric-key systems are simpler and faster. The two parties must somehow exchange the key in a secure way.
  - The most popular symmetric-key system is the *DES, Data Encryption Standard*

2.2 Symmetric Key Crypt. Algorithms

2.2.1 Introduction

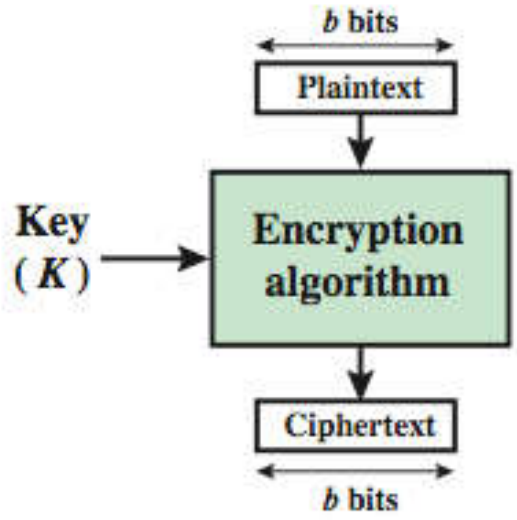
- 对称加密 (也叫私钥制加密) 指加密和解密使用相同密钥的加密算法，有时又叫传统密码算法。
  - 对称密码系统的加密密钥能够从解密密钥中推算出来，同时解密密钥也可以从加密密钥中推算出来。在大多数的对称算法中，采用相同的加密密钥和解密密钥，所以也称这种加密算法为秘密密钥算法或单密钥算法。
  - 对称加密要求发送方和接收方在开始安全通信之前先商定一个密钥。对称算法的安全性依赖于密钥，任何一方泄漏密钥都会导致双方传输的加密消息被解密，所以密钥的保密性至关重要。

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Types： Stream Cipher & Block Cipher**
  - Block Cipher
    - ✧ M is a plain text and separated into M_1、M_2、...、M_n segments of length *b* bits.

      $E(M, K) = E(M_1, K)E(M_2, K) \ldots E(M_n, K)$.
    - ✧ Slow but safer



(b) Block Cipher

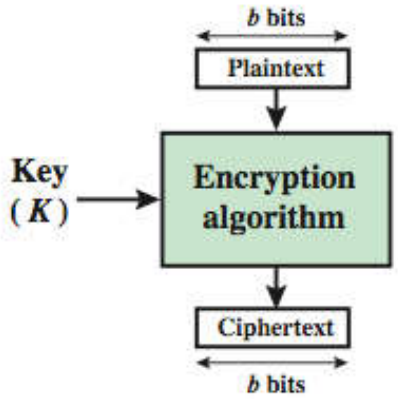
2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Types：Stream Cipher & Block Cipher**
  - 广义上从明文生成密文的算法类型有两种：块加密 (block cipher) 和流加密 (stream cipher)。
  - 块加密
    - ✧ 将明文 M 分割成 M_1、M_2 ... M_n 区段，每一个区段的长度为 *b*，消息应用相同的演算法则和钥匙，数学表示为

      $$E(M, K) = E(M_1, K)E(M_2, K) \ldots E(M_n, K)$$

    - ✧ 加密速度慢，但相对较安全。

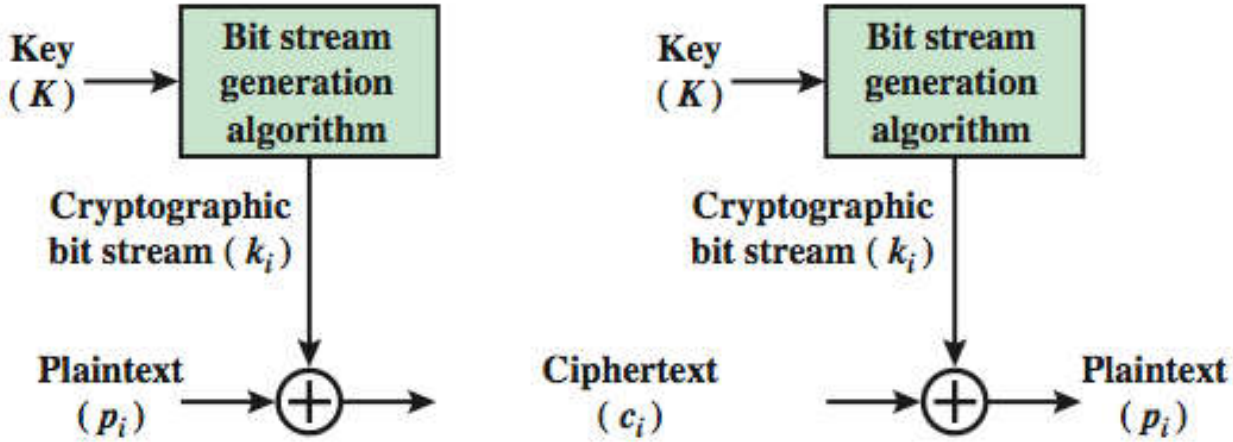


(b) Block Cipher

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Types： Stream Cipher & Block Cipher**
  - Stream Cipher

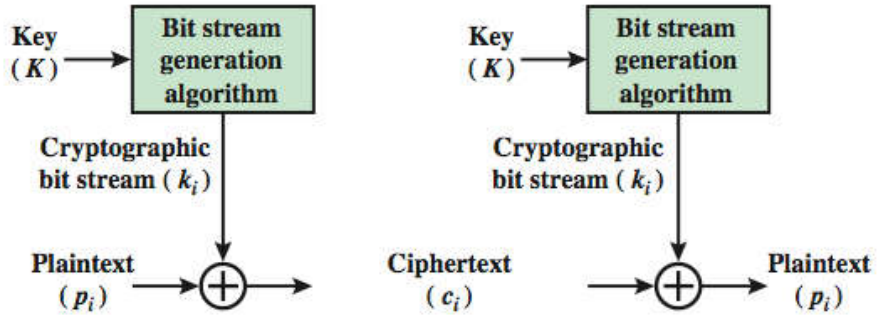


(a) Stream Cipher Using Algorithmic Bit Stream Generator

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Types：Stream Cipher & Block Cipher**
  - 流加密
    - ✧ 流加密不将明文切分为区段，而是一次加密资料流的一个位元或一个位元组。常见的作法是将较短的加密钥匙延展成为无限长、近似乱码的一长串密钥串流 (key-stream)，再将密钥串流和明文 (plain text) 经过 XOR 运算后，产生密文 (cipher text)。加密速度快，但相对容易被破解。

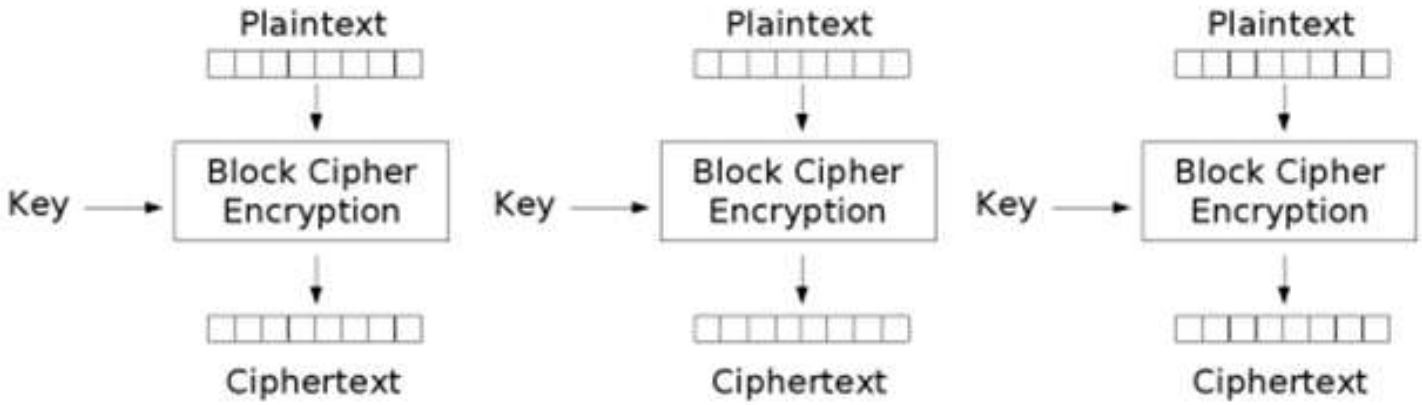


(a) Stream Cipher Using Algorithmic Bit Stream Generator

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Electronic Code Book (ECB) Mode 电子密码本模式
    - ✧ ECB - Encryption

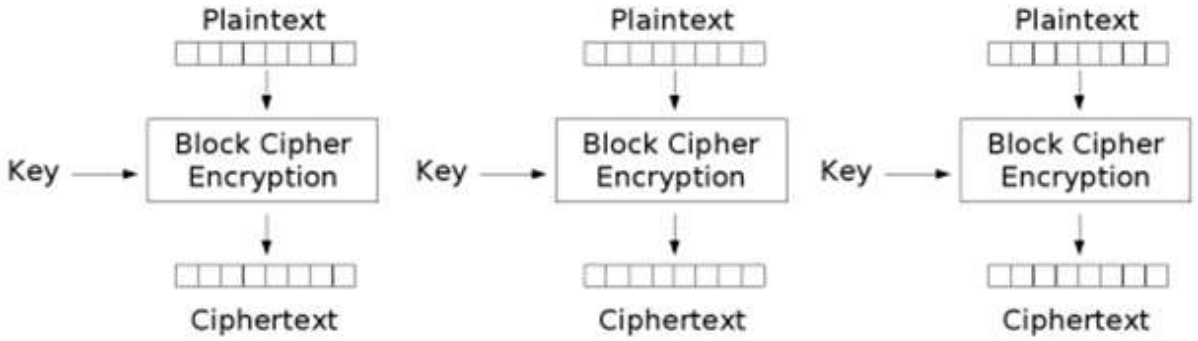


Electronic Codebook (ECB) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Electronic Code Book (ECB) Mode
    - ECB 是最早采用和最简单的加密模式，它将加密的数据分成若干组，每组的大小跟加密密钥长度相同，然后每组都用相同的密钥进行加密。比如 DES 算法，采用一个64位的密钥，明文分成每组64位的数据，最后一组补齐64位，然后每组数据都采用 DES 算法的64位密钥进行加密。

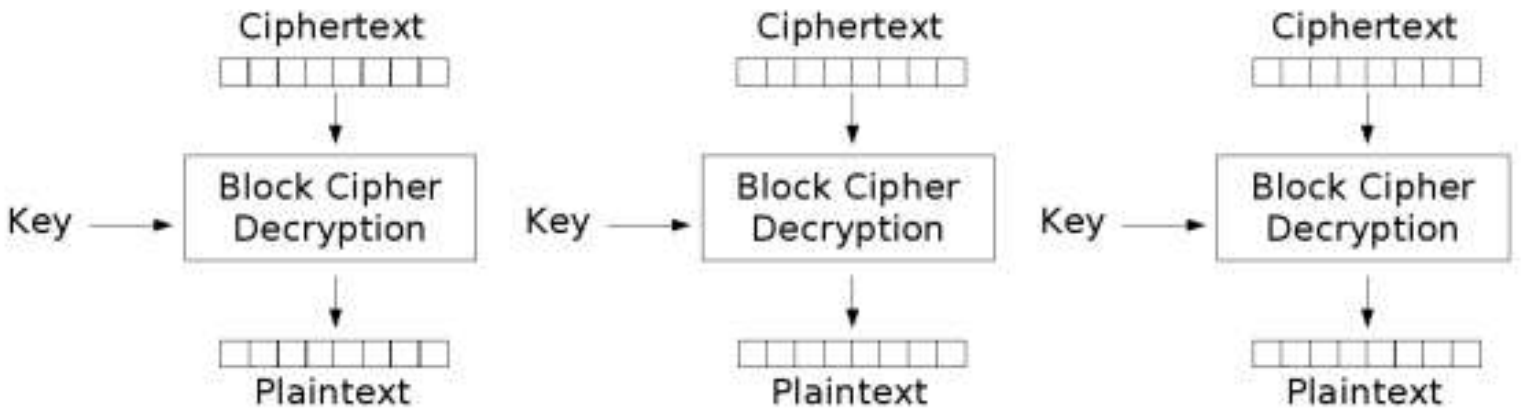


Electronic Codebook (ECB) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Electronic Code Book (ECB) Mode
    - ✧ ECB - Decryption



Electronic Codebook (ECB) mode decryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

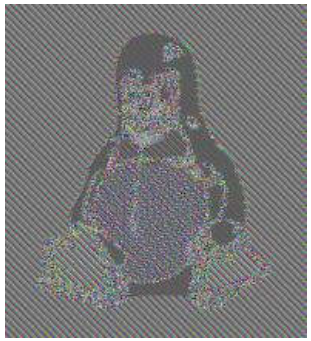
- **Algorithm Modes**
  - Electronic Code Book (ECB) Mode
    - ✧ 例：My name is DragonKing 的每8个字符 (64位) 作为一块，使用一个相同的64位的密钥对每个块进行加密，最后一块不足64位，则补齐为64位后再进行加密。

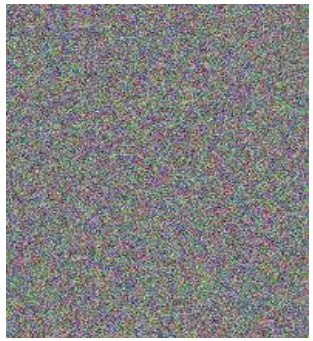
    - ✧ 可以看到，ECB 方式每64位使用的密钥都是相同的，相对容易获得密文进行破解。此外，因为每64位是相互独立的，黑客有时候甚至不用破解密码，只要简单的将其中一块替换就可以达到目的。

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Electronic Code Book (ECB) Mode
    - ✧ 同样的明文块会被 ECB 加密成相同的密文块，因此它不能很好地隐藏数据模式。下面的例子显示了ECB在密文中显示明文的模式的程度：该图像的一个位图版本 (左图) 通过 ECB 模式可能会被加密成中图，而某些非 ECB 模式会将其加密成右图。

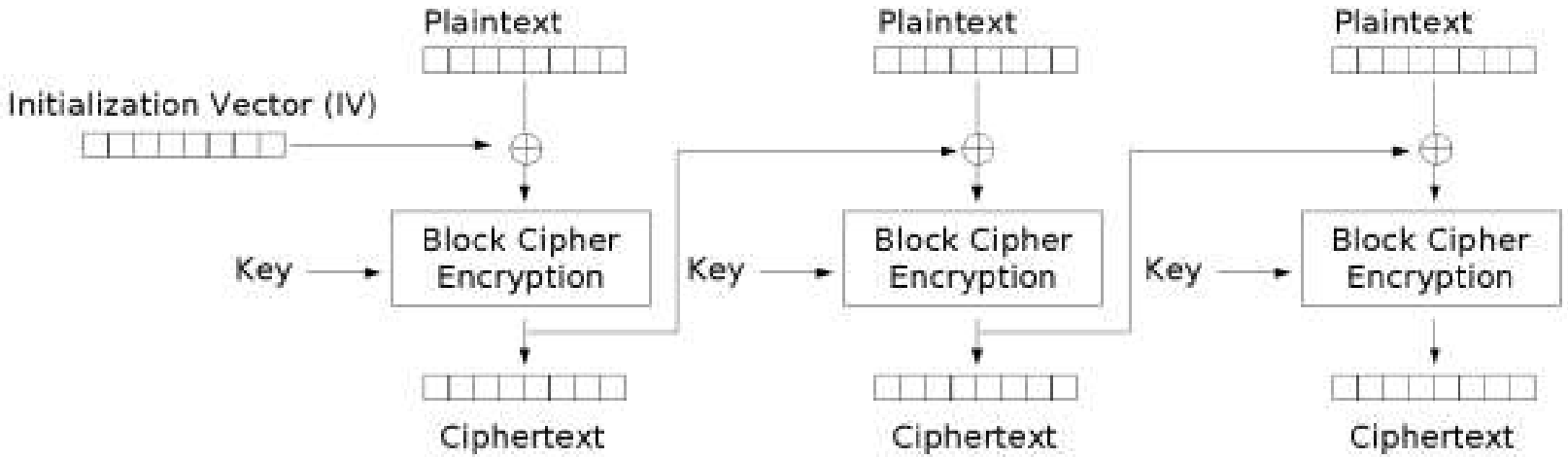




2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Block Chaining (CBC) Mode 密码块链接模式 (IBM,1976)
    - ✧ CBC - Encryption

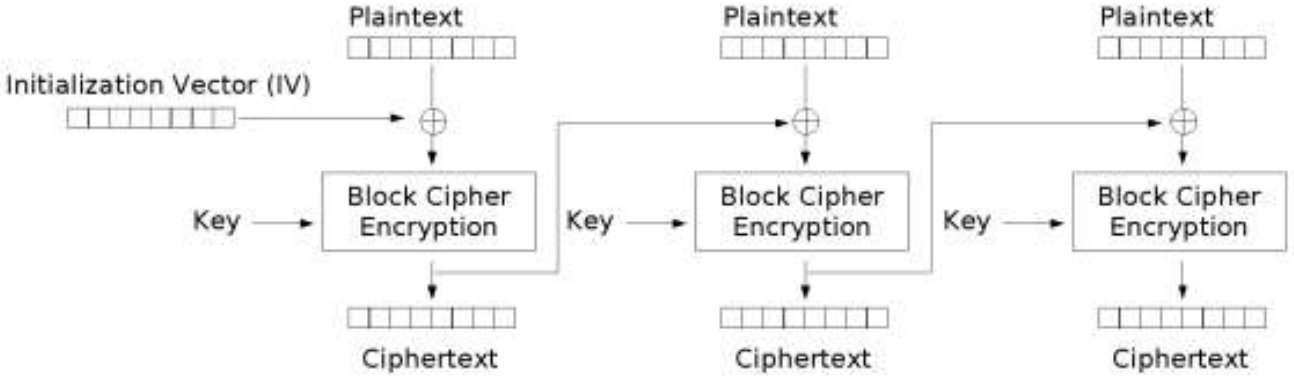


Cipher Block Chaining (CBC) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Block Chaining (CBC) Mode
    - ✧ CBC 模式首先也是将明文分成固定长度 (64位) 的块 (P_0, P_1, ...)，然后将前面一个加密块输出的密文与当前要加密的明文块进行 XOR 操作计算，将计算结果再用密钥进行加密得到当前块的密文。第一明文块加密的时候，因为前面没有加密的密文，所以需要一个初始化向量 (IV)。

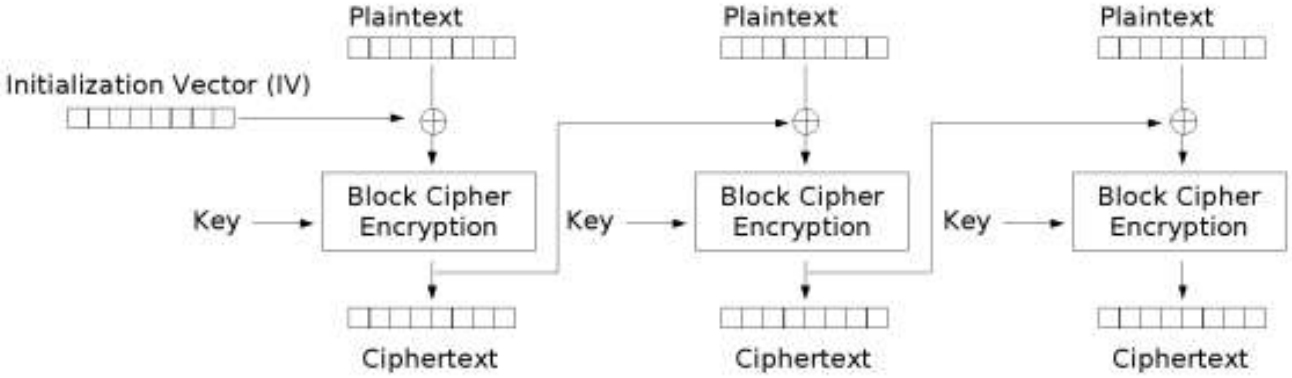


Cipher Block Chaining (CBC) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Block Chaining (CBC) Mode
    - ✧ 跟 ECB 模式不一样的是，CBC 模式通过链接关系使得密文跟明文不再一一对应，破解起来更困难，而且可以抵抗只要简单调换密文块就可能达到目的的攻击。缺点是不能实时解密，每一个密文块必须等到8个字节收齐后才能开始解密，不太适合实时性要求比较高的场合。

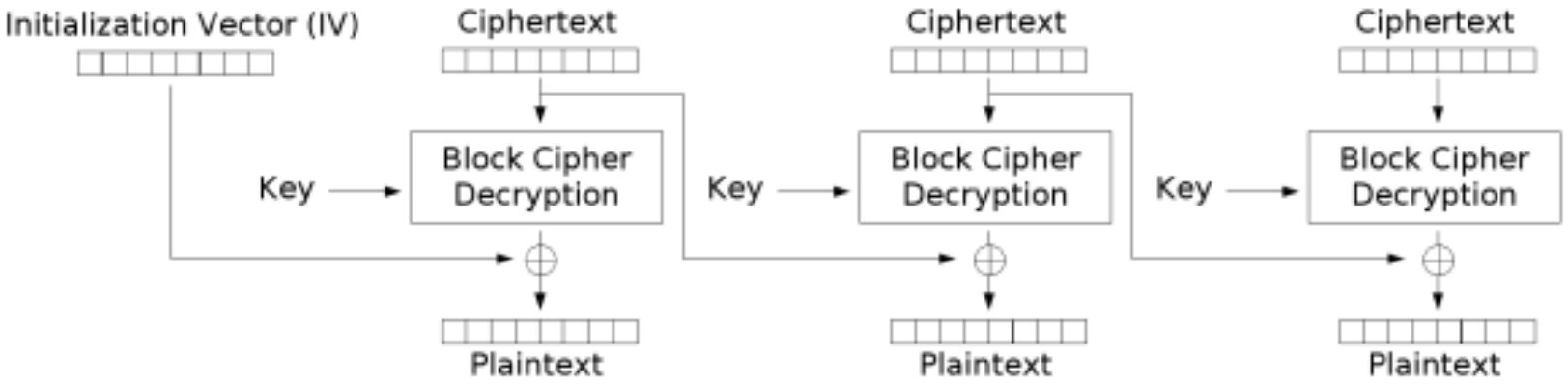


Cipher Block Chaining (CBC) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Block Chaining (CBC) Mode
    - ✧ CBC - Decryption

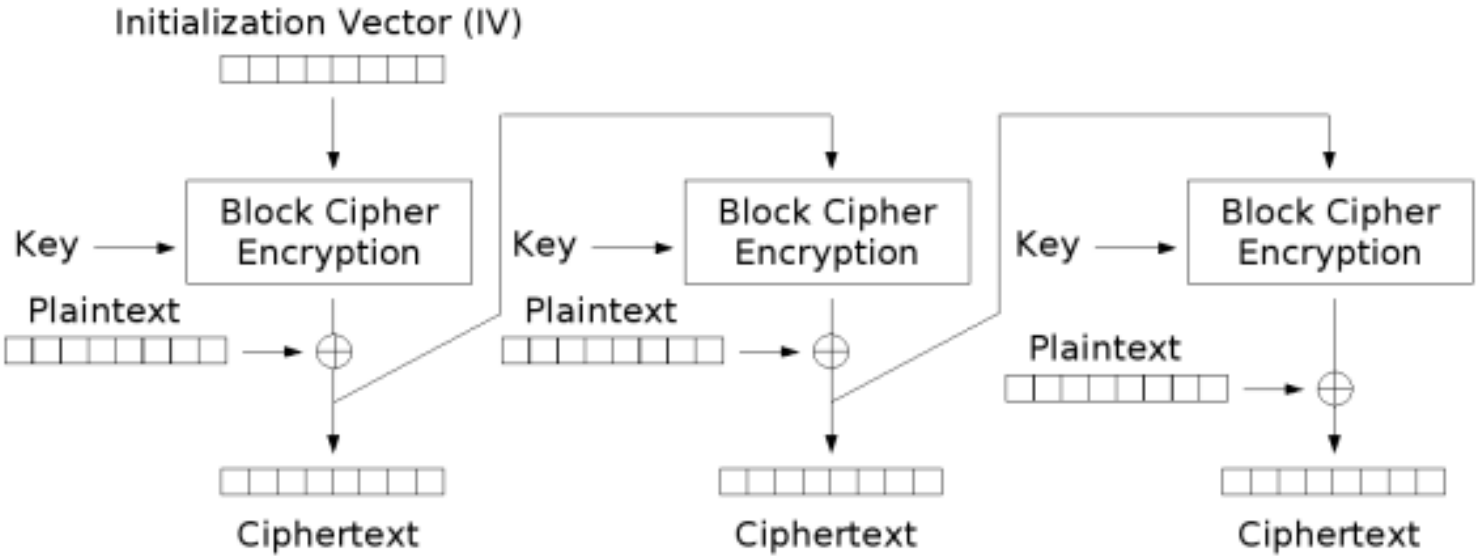


Cipher Block Chaining (CBC) mode decryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Feedback (CFB) Mode 密文反馈模式
    - ✧ CFB - Encryption

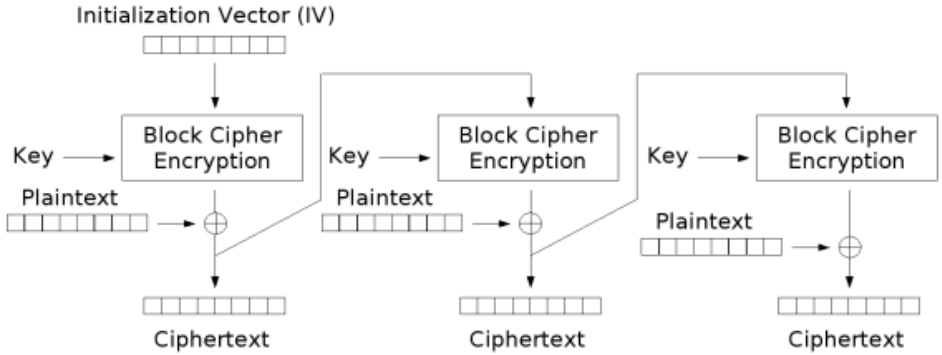


Cipher Feedback (CFB) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Feedback (CFB) Mode
    - ✧ CFB 模式为了克服必须等到收齐8个字节才能进行解密的缺点，采用了一个64位 (8个字节) 的移位寄存器来获得密文。例如，当前源文字节是 P_{10}，$C_2, C_3, \ldots, C_9$ 是移位寄存器数据，加密时从移位寄存器取 C_2~ C_9 用密钥施行加密运算，取加密数据最左边的一个字节跟输入的明文 P_{10} 进行 XOR 操作，得到的值作为输出密文 C_{10}，同时将 C_{10} 送入到移位寄存器中。

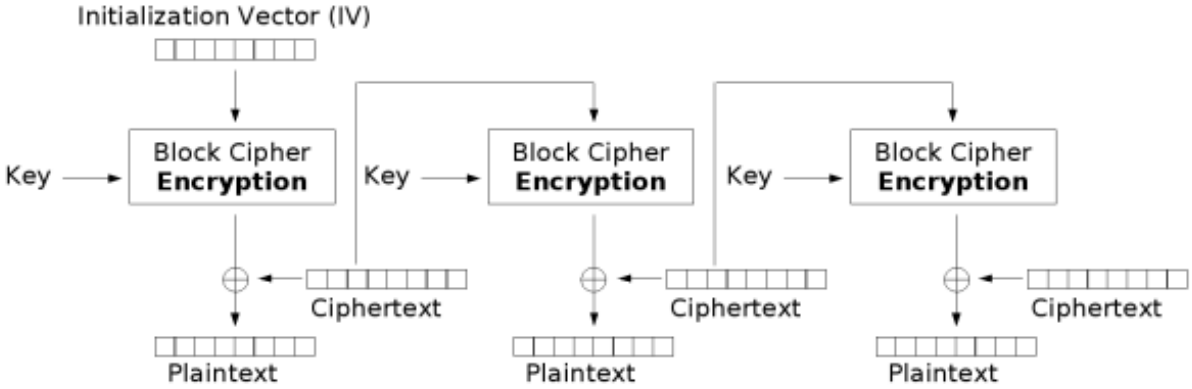


Cipher Feedback (CFB) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Cipher Feedback (CFB) Mode
    - ✧ CFB - Decryption
    - ✧ 从 C_{10} 获得 P_{10} 的解密过程：从移位寄存器取 C_2~ C_9 用密钥施行加密运算，取结果数据最左边的一个字节跟输入的密文 C_{10} 进行 XOR 操作，得到的值就是源文的 P_{10}。

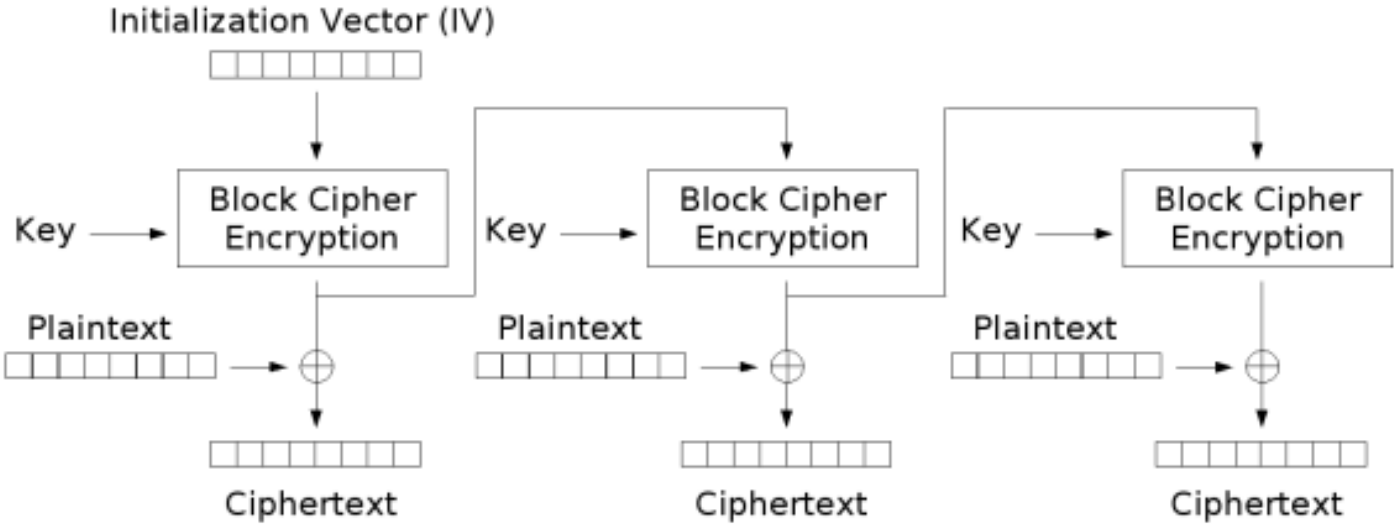


Cipher Feedback (CFB) mode decryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Output Feedback (OFB) Mode 输出反馈模式
    - ✧ OFB – Encryption
      - ◌ OFB 跟 CFB 的不同在于移位寄存器的数据来源

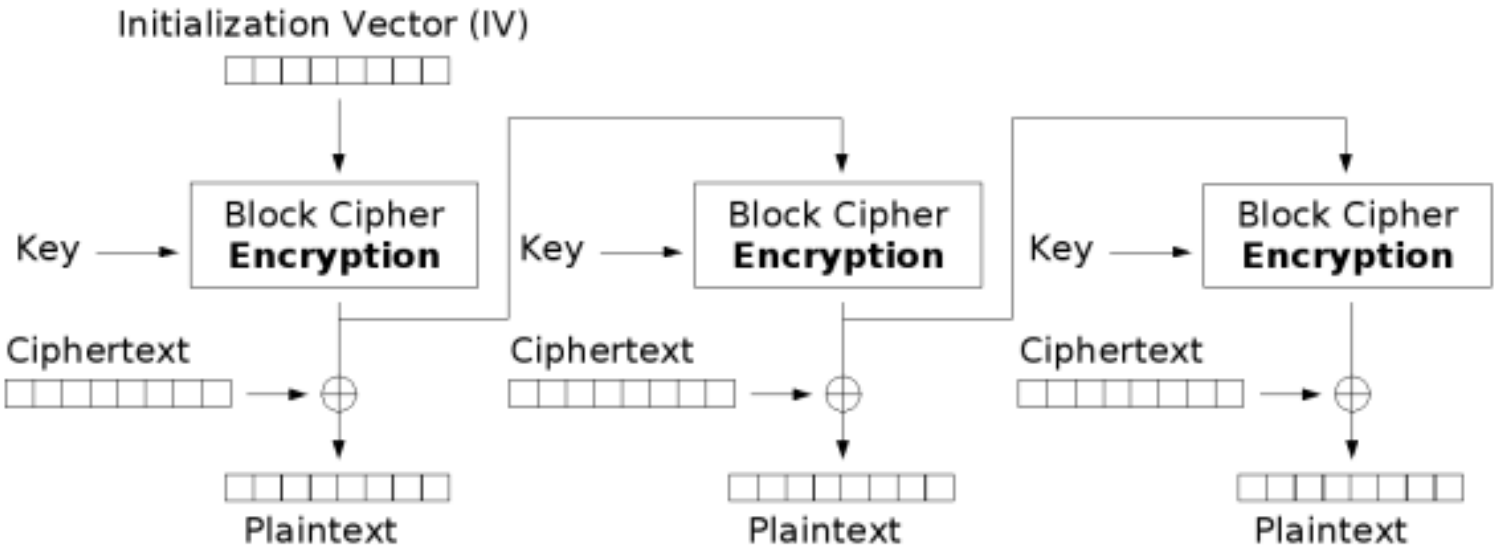


Output Feedback (OFB) mode encryption

2.2 Symmetric Key Crypt. Algorithms

2.2.2 Algorithm Types and Modes

- **Algorithm Modes**
  - Output Feedback (OFB) Mode
    - ✧ OFB - Decryption



Output Feedback (OFB) mode decryption

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **Background and History**
  - Development of DES
    - ✧ DES (Data Encryption Standard) was developed in the early 1970s at IBM and based on an earlier design by *Horst Feistel* . The algorithm was submitted to the National Bureau of Standards (NBS) following the agency's invitation to propose a candidate for the protection of sensitive, unclassified electronic government data. In 1976, after consultation with the National Security Agency (NSA), the NBS eventually selected a slightly modified version (strengthened against differential cryptanalysis, but weakened against brute force attacks), which was published as an official Federal Information Processing Standard (FIPS) for the United States in 1977.

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **Background and History**
  - Development of DES
    - ✧ 数据加密标准 (DES, Data Encryption Standard) 是一种使用密钥加密的块密码，1976年被美国国家标准局 (NBS, National Bureau of Standards，1988年改名为 NIST) 确定为联邦信息处理标准 (FIPS)，随后在国际上获得广泛采用。
    - ✧ DES 基于56位密钥的对称算法，这个算法因为包含一些机密设计元素，相对短的密钥长度以及被怀疑内含美国国家安全局 (NSA) 的后门而在开始时备受争议。DES 因此受到了学院派式的严格审查，并以此推动了现代块密码及其密码分析技术的发展。

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **Background and History**
  - DES 现状
    - ✧ DES 现在已经不被视为一种安全的加密算法，主要原因是它使用的56位密钥过短。1999年1月，distributed.net 与电子前线基金会 (Electronic Frontier Foundation) 合作，在22小时15分钟内公开破解了一个 DES 密钥。有一些分析报告提出了该算法的理论上的弱点。为了提供实用所需的安全性，可以使用 DES 的派生算法 3DES 来进行加密 (虽然 3DES 也存在理论上的攻击方法)。
    - ✧ 2001年，DES 被高级加密标准 (AES) 所取代。另外，DES 已经不再作为 NIST 的标准。

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - DES 特点概述
    - ✧ DES 是一种典型的块加密方法：它以64位为分组长度，64位一组的明文作为算法的输入，通过一系列复杂的操作，输出同样64位长度的密文。
    - ✧ DES 使用加密密钥定义变换过程，因此算法认为只有持有加密所用的密钥的用户才能解密密文。
    - ✧ DES 采用64位密钥，但由于每8位中的最后1位用于奇偶校验，实际有效密钥长度为56位。密钥可以是任意的56位的数，且可随时改变。其中极少量的数被认为是弱密钥，但能容易地避开它们。所有的保密性依赖于密钥。
    - ✧ DES 算法的基本过程是换位和置换。

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 对 DES 的一般讨论包括：
    - ✧ 总体结构
    - ✧ *Feistel* 轮函数
    - ✧ 子密钥生成
    - ✧ 解密过程
    - ✧ 有效性证明

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 信息空间
    - ✧ 信息空间由 {0, 1} 组成的字符串构成，原始明文消息和经过 DES 加密的密文信息是8个字节 (64位) 的分组，密钥也是64位。
    - ✧ 原始明文消息按 PKCS#5 规范进行填充：
      - ◌ 原始明文消息最后的分组不够8个字节 (64位) 时，在末尾以字节填满，填入的字节取值相同，都是填充的字节数目；
      - ◌ 原始明文消息刚好分组完全时，在末尾填充8个字节 (即增加一个完整分组)，字节取值都是08。
    - ✧ 明文分组结构：$M = m_1 m_2 \ldots m_{64}$, $m_i \in \{0, 1\}$, $i = 1 .. 64$.
    - ✧ 密文分组结构：$C = c_1 c_2 \ldots c_{64}$, $c_i \in \{0, 1\}$, $i = 1 .. 64$.
    - ✧ 密钥结构：$K = k_1 k_2 \ldots k_{64}$, $k_i \in \{0, 1\}$, $i = 1 .. 64$.
      - ◌ 除去 $k_8, k_{16}, \ldots, k_{64}$ 共8位奇偶校验位，起作用的仅为56位。

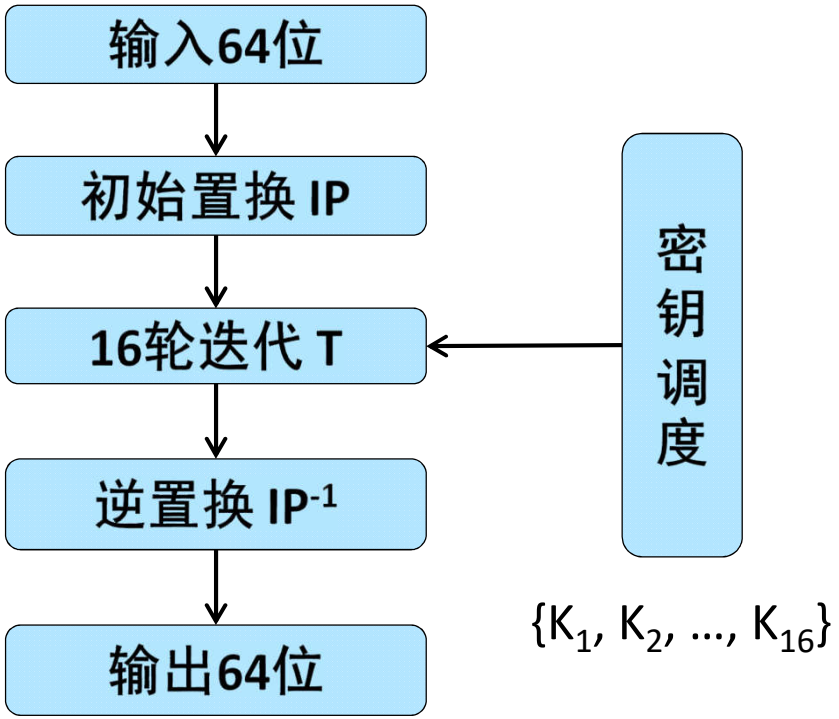
2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 加密过程
    - $C = E_k(M) = IP^{-1} \cdot T_{16} \cdot T_{15} \cdot \ldots \cdot T_1 \cdot IP(M)$.
      - IP 为初始置换；
      - IP^{-1} 是 IP 的逆置换；
      - $T_1, T_2, \ldots, T_{16}$ 是一系列的迭代变换。
  - 解密过程
    - $M = D_k(C) = IP^{-1} \cdot T_1 \cdot T_2 \cdot \ldots \cdot T_{16} \cdot IP\ (C)$.

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - DES 算法的总体结构 — *Feistel* 结构



- ✧ 输入64位明文 M 时，子密钥按 $(K_1K_2 \ldots K_{16})$ 次序调度，是加密过程。
- ✧ 输入64位密文 C 时，子密钥按 $(K_{16}K_{15} \ldots K_1)$ 次序调度，是解密过程。

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 初始置换 IP
    - ✧ 给定64位明文块 M，通过一个固定的初始置换 IP 来重排 M 中的二进制位，得到二进制串 $M_0 = IP(M) = L_0R_0$，这里 L_0 和 R_0 分别是 M_0 的前32位和后32位。下表给出 IP 置换后的下标编号序列。

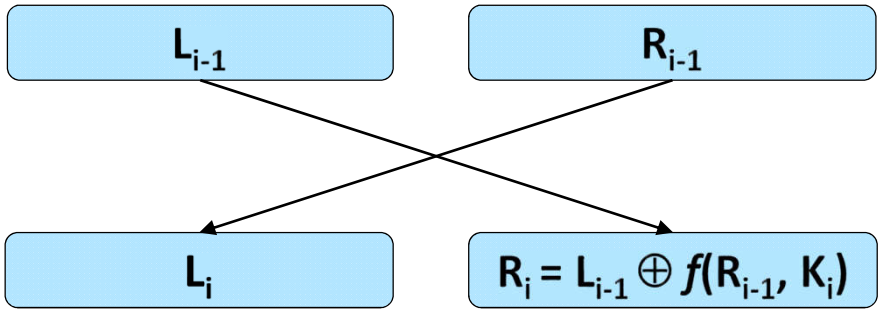
IP 置换表 (64位)							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 迭代 T
    - ✧ 根据 L_0R_0 按下述规则进行16次迭代，即

      $$L_i = R_{i-1},\ R_i = L_{i-1} \oplus f(R_{i-1}, K_i),\ i = 1 .. 16.$$

    - ✧ 这里 ⊕ 是32位二进制串按位异或运算，f 是输出32位的 *Feistel* 轮函数；
    - ✧ 16个长度为48位的子密钥 K_i (i = 1 .. 16) 由密钥 K 生成；
    - ✧ 16次迭代后得到 $L_{16}R_{16}$ ；
    - ✧ 左右交换输出 $R_{16}L_{16}$。



2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 逆置换 IP^{-1}
    - ✧ 对迭代 T 输出的二进制串 $R_{16}L_{16}$ 使用初始置换的逆置换 IP^{-1} 得到密文 C，即： $C = IP^{-1}(R_{16}L_{16})$.

IP 置换表 (64位)							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1} 置换表 (64位)							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$

    (1) 将长度为32位的串 R_{i-1} 作 E-扩展，成为48位的串 $E(R_{i-1})$；

    (2) 将 $E(R_{i-1})$ 和长度为48位的子密钥 K_i 作48位二进制串按位异或运算，K_i 由密钥 K 生成；

    (3) 将 (2) 得到的结果平均分成8个分组 (每个分组长度6位)，各个分组分别经过8个不同的 S-盒进行 6-4 转换，得到8个长度分别为4位的分组；

    (4) 将 (3) 得到的分组结果顺序连接得到长度为32位的串；

    (5) 将 (4) 的结果经过 P-置换，得到的结果作为轮函数 $f(R_{i-1}, K_i)$ 的最终32位输出。

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ E-扩展规则 (表中给出32位二进制串扩展后的下标编号序列)

E-扩展规则 (比特-选择表)					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ E-扩展规则 (表中给出32位二进制串扩展后的下标编号序列)

E-扩展规则 (比特-选择表)					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒
      - ◌ S-盒是一类选择函数，用于二进制 6-4 转换。 *Feistel* 轮函数使用 8个 S-盒 $S_1, \ldots, S_8$，每个 S-盒是一个4行 (编号 0-3)、16列 (编号 0-15) 的表，表中的每个元素是一个十进制数，取值在0-15之间，用于表示一个4位二进制数。
      - ◌ 假设 S_i 的6位输入为 $b_1b_2b_3b_4b_5b_6$，则由 $n = (b_1b_6)_{10}$ 确定行号，由 $m = (b_2b_3b_4b_5)_{10}$ 确定列号，$[S_i]_{n,m}$ 元素的值的二进制形式即为所要的 S_i 的输出。

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒
      - ◌ 例1：设 S_1 的输入 $b_1b_2b_3b_4b_5b_6 = 101100$，则

$$n = (b_1b_6)_{10} = (10)_{10} = 2,$$

$$m = (b_2b_3b_4b_5)_{10} = (0110)_{10} = 6$$

查表得到 $[S_1]_{2,6} = 2 = (0010)_2$ 即为所要的输出。

S_1-BOX															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	15	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒
      - ◌ 例1：设 S_1 的输入 $b_1b_2b_3b_4b_5b_6 = 101100$，则

$$n = (b_1b_6)_{10} = (10)_{10} = 2,$$

$$m = (b_2b_3b_4b_5)_{10} = (0110)_{10} = 6$$

查表得到 $[S_1]_{2,6} = 2 = (0010)_2$ 即为所要的输出。

S_1-BOX															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	15	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒
      - ◌ 例2：设 S_1 的输入 $b_1b_2b_3b_4b_5b_6 = 111001$，则

$$n = (b_1b_6)_{10} = (11)_{10} = 3,$$

$$m = (b_2b_3b_4b_5)_{10} = (1100)_{10} = 12$$

查表得到 $[S_1]_{3,12} = 10 = (1010)_2$ 即为所要的输出。

S_1-BOX															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	15	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒
      - ◌ 例2：设 S_1 的输入 $b_1b_2b_3b_4b_5b_6 = 111001$，则

$$n = (b_1b_6)_{10} = (11)_{10} = 3,$$

$$m = (b_2b_3b_4b_5)_{10} = (1100)_{10} = 12$$

查表得到 $[S_1]_{3,12} = 10 = (1010)_2$ 即为所要的输出。

S_1-BOX															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	15	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒 S_1-S_4

S_1-BOX															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	15	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2-BOX															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3-BOX															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4-BOX															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
12	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ S-盒 S_5-S_8

S_5-BOX															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6-BOX															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7-BOX															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8-BOX															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - *Feistel* 轮函数 $f(R_{i-1}, K_i)$
    - ✧ P-置换 (下表给出32位二进制串 P-置换后的下标编号序列)

P–置换表			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 子密钥生成
    - ✧ 子密钥生成过程根据给定的64位密钥 K 生成 *Feistel* 轮函数的每轮中使用的子密钥 K_i 。

(1) 对 K 的56个非校验位实行置换 PC-1，得到 C_0D_0，其中 C_0 和 D_0 分别由 PC-1 置换后的前28位和后28位组成。

PC-1 置换表						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

注意到密钥 K 的8个校验位的下标不参与置换。

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 子密钥生成
    - ✧ 子密钥生成过程根据给定的64位密钥 K 生成 *Feistel* 轮函数的每轮中使用的子密钥 K_i 。

(1) 对 K 的56个非校验位实行置换 PC-1，得到 C_0D_0，其中 C_0 和 D_0 分别由 PC-1 置换后的前28位和后28位组成。

PC-1 置换表

C_0	57	49	41	33	25	17	9
	1	58	50	42	34	26	18
	10	2	59	51	43	35	27
	19	11	3	60	52	44	36
D_0	63	55	47	39	31	23	15
	7	62	54	46	38	30	22
	14	6	61	53	45	37	29
	21	13	5	28	20	12	4

注意到密钥 K 的8个校验位的下标不参与置换。

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 子密钥生成
    - ✧ 子密钥生成过程根据给定的64位密钥 K 生成 *Feistel* 轮函数的每轮中使用的子密钥 K_i 。

    (1) 对 K 的56个非校验位实行置换 PC-1，得到 C_0D_0，其中 C_0 和 D_0 分别由 PC-1 置换后的前28位和后28位组成。i = 1。

    (2) 计算 $C_i = LS_i(C_{i-1})$ 和 $D_i = LS_i(D_{i-1})$

      - ◌ 当 i =1, 2, 9, 16 时，$LS_i(A)$ 表示将二进制串 A 循环左移一个位置；否则循环左移两个位置。

    (3) 对 56位的 C_iD_i 实行 PC-2 压缩置换，得到48位的 K_i 。i = i+1。

    (4) 如果已经得到 K_{16}，密钥调度过程结束；否则转 (2)。

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

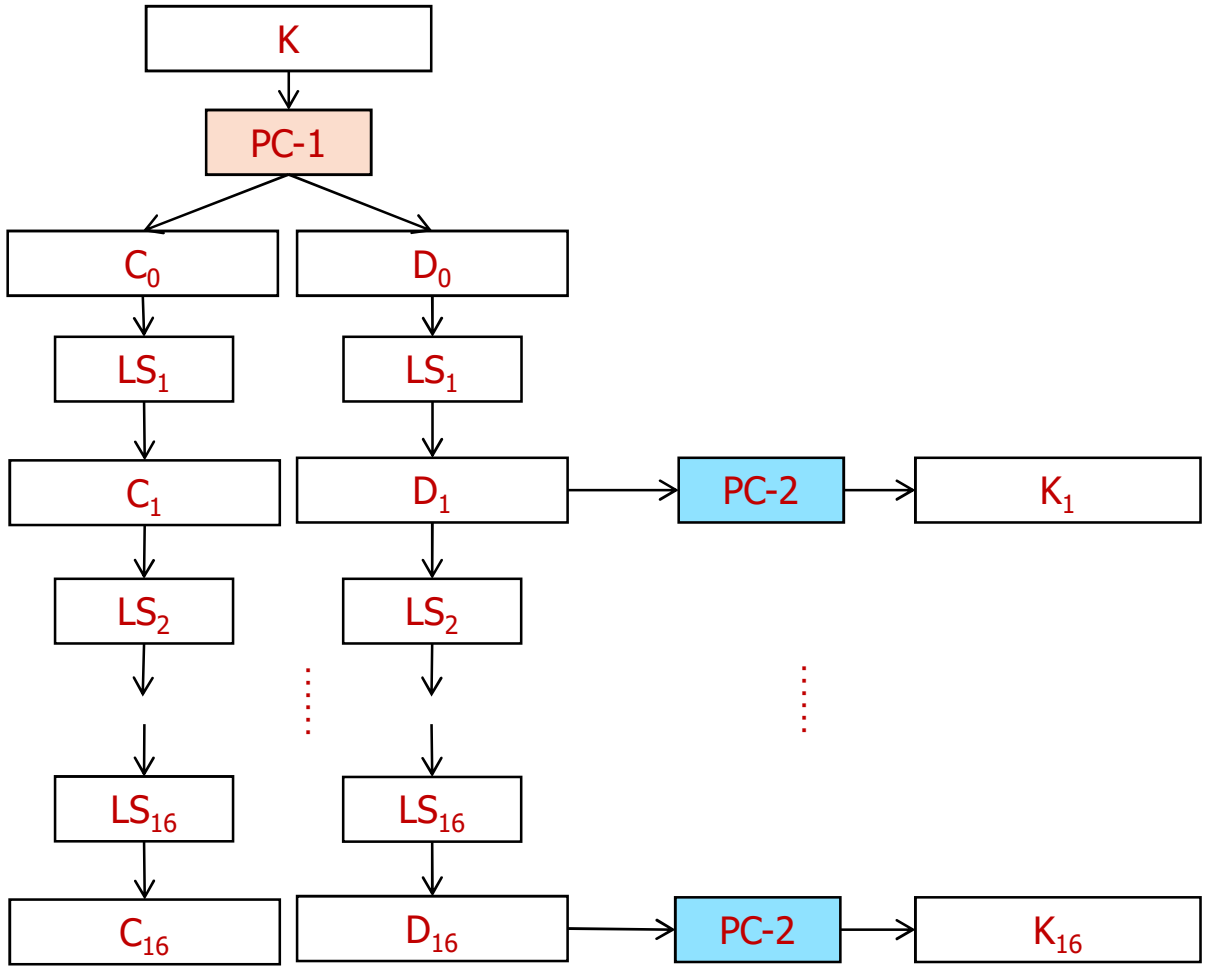
- **DES 算法概要**
  - 子密钥生成
    - PC-2 压缩置换：从56位的 C_iD_i 中去掉第 9, 18, 22, 25, 35, 38, 43, 54位，将剩下的48位按照 PC-2 置换表作置换，得到 K_i 。

PC-2 压缩置换表					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - 子密钥生成



2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - DES 的解密
    - ✧ 分析所有的代替、置换、异或和循环移动过程，获得一个非常有用的性质：DES 的加密和解密可使用相同的算法和密钥。
    - ✧ DES 的过程设计使得用相同的函数来加密或解密每个分组成为可能。加解密过程中使用由同一个密钥 K 经过相同的子密钥生成算法得到的子密钥序列，唯一不同之处是加解密过程中子密钥的调度次序恰好相反。
      - ◌ 加密过程的子密钥按 $(K_1\ K_2 \ldots K_{15}\ K_{16})$ 次序调度
      - ◌ 解密过程的子密钥按 $(K_{16}\ K_{15} \ldots K_2\ K_1)$ 次序调度

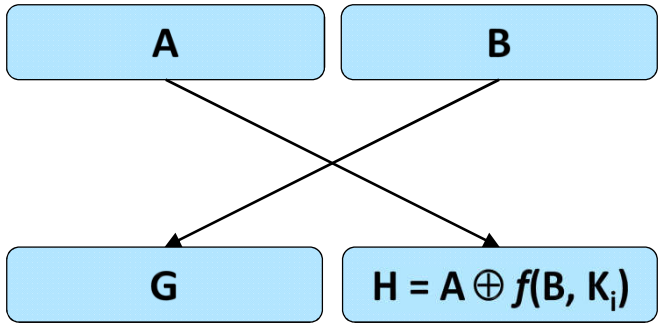
2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - DES 的有效性证明
    - ✧ 64位密文 C 输入 DES 过程，IP 置换后得到加密过程中的 $R_{16}L_{16}$。
    - ✧ 解密过程对 $R_{16}L_{16}$ 实行16轮迭代，过程中 *Fiestel* 轮函数按照相反次序引用子密钥 $K_{16}, K_{15}, \ldots, K_1$。

      $A = R_{16}$，$B = L_{16}$

      $G = B = L_{16} = R_{15}$

      $H = A \oplus f(B, K_{16}) = R_{16} \oplus f(L_{16}, K_{16})$



2.2.3 Data Encryption Standard (DES)

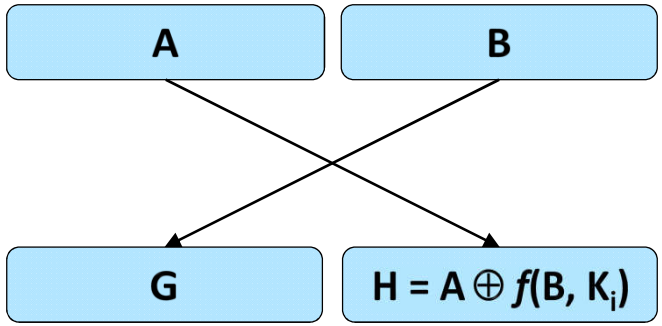
- **DES 算法概要**
  - DES 的有效性证明
    - ✧ 64位密文 C 输入 DES 过程，IP 置换后得到加密过程中的 $R_{16}L_{16}$。
    - ✧ 对 $R_{16}L_{16}$ 实行16轮迭代，过程中 Fiestel 轮函数按照相反次序引用子密钥 $K_{16}, K_{15}, \ldots, K_1$。

$A = R_{16}$，$B = L_{16}$

$G = B = L_{16} = R_{15}$

$H = A \oplus f(B, K_{16}) = R_{16} \oplus f(L_{16}, K_{16})$

$= L_{15} \oplus f(R_{15}, K_{16}) \oplus f(R_{15}, K_{16}) = L_{15}$



等价性来源于加密过程的迭代：

$L_i = R_{i-1}$，$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$, i = 1, …, 16.

即有：$L_{16} = R_{15}$

$R_{16} = L_{15} \oplus f(R_{15}, K_{16})$

2.2.3 Data Encryption Standard (DES)

- **DES 算法概要**
  - DES 的有效性证明
    - ✧ 64位密文 C 输入 DES 过程，IP 置换后得到加密过程中的 $R_{16}L_{16}$。
    - ✧ 对 $R_{16}L_{16}$ 实行16轮迭代，过程中 Fiestel 轮函数按照相反次序引用子密钥 $K_{16}, K_{15}, \ldots, K_1$。

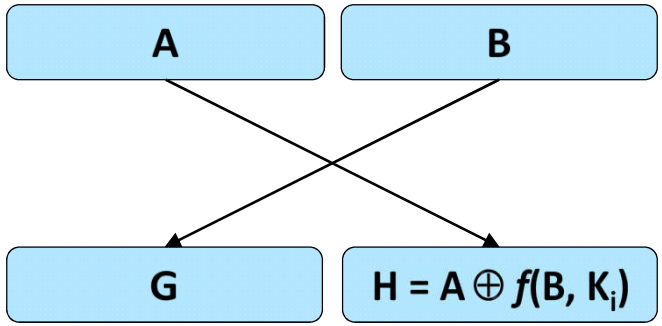
      $A = R_{16}$，$B = L_{16}$

      $G = B = L_{16} = R_{15}$

      $H = A \oplus f(B, K_{16}) = R_{16} \oplus f(L_{16}, K_{16})$

      $= L_{15} \oplus f(R_{15}, K_{16}) \oplus f(R_{15}, K_{16}) = L_{15}$

      ……

    - ✧ 16轮迭代结束时 $G = R_0$, $H = L_0$。按算法过程左右交换得到 L_0R_0，即为加密过程中的 M_0。
    - ✧ M_0 经过 IP^{-1} 置换得到加密前的明文块 M，解密过程结束。

2.2 Symmetric Key Crypt. Algorithms

2.2.3 Data Encryption Standard (DES)

- **DES 的讨论**
  - S-盒的设计
    - ✧ DES 的核心是 S-盒，除此之外的计算是线性的。
      - ◌ S-盒作为该密码体制的非线性组件对安全性至关重要，但 S-盒的设计原理至今未公布，是否存在隐藏陷门 (Hidden Trapdoors) 不得而知 (DES 的半公开性)。
    - ✧ S-盒的设计准则：
      - ◌ S-盒中的每一行是整数0-15的一个置换
      - ◌ S-盒不是它输入变量的线性或仿射函数
      - ◌ S-盒的输入端每改变1位至少要引起输出端改变2位
      - ◌ S(X) 和 S(X+001100) 至少有2位不同
      - ◌ 对6位二进制串 $X = x_1x_2x_3x_4x_5x_6$，$S(X) \neq S(X+11x_5x_600)$
      - ◌ S-盒的输入端保持任1位不变，则其它输入位的变化输出数字中0和1的总数近于相等。

2.2.3 Data Encryption Standard (DES)

- **DES 的讨论**
  - 可攻击性
    - ✧ DES 的实际密钥长度为56位，就目前计算机的计算能力而言，DES 不能抵抗对密钥的穷举搜索攻击。
    - ✧ 1997年克罗拉多州的程序员 *Verser* 在因特网上数万名志愿者的协作下用96天的时间找到了密钥长度为40位和48位的 DES 密钥。1998年7月电子前哨基金会 (EFF) 使用一台价值25万美元的计算机在56小时之内破译了56位的 DES。1999年1月 EFF 通过因特网上的10万台计算机合作，仅用22小时15分就破解了56位的 DES。
    - ✧ 不过这些破译的前提是，破译者能识别出破译的结果确实是明文，也即破译的结果必须容易辩认。如果明文加密之前经过压缩等处理，辩认工作将比较困难。

2.2.3 Data Encryption Standard (DES)

- **DES 的讨论**
  - 香农准则
    - ✧ 充分混淆：密钥、明文以及密文之间的依赖关系相当复杂。
    - ✧ 充分扩散：密钥的每一位数字影响密文的许多位数字，明文的每一位数字也应影响密文的许多位数字。

2.2 Symmetric Key Crypt. Algorithms

2.2.4 Advanced Encryption Standard (AES)

- Introduction
  - 2001, FIPS PUB 197
  - Rijndael - *Joan Daemen* & *Vincent Rijmen*
- Properties
  - Block length: 128 bits
  - Key length: 128/192/256 bits
- Steps
  - AddRoundKey
  - SubBytes
  - ShiftRows
  - MixColumns

2.2 Symmetric Key Crypt. Algorithms

2.2.4 Advanced Encryption Standard (AES)

- **Introduction**
  - Release of AES
    - ✧ 经过五年的甄选流程，美国国家标准与技术研究院 (NIST)于2001年11月26日发布 AES，并在2002年5月26日成为有效的联邦信息处理加密标准 FIPS PUB 197，替代原先的 DES。
    - ✧ AES 在密码学中也被称为 Rijndael 加密法，已经被多方分析且广泛使用。2006年后，AES 已成为对称密钥加密中最流行的算法之一。
    - ✧ Rijndael 算法由比利时密码学家 *Joan Daemen* 和 *Vincent Rijmen* 设计发明，算法也以两位作者的名字结合命名。

2.2 Symmetric Key Crypt. Algorithms

2.2.4 Advanced Encryption Standard (AES)

- **How AES Works**
  - AES 和 Rijndael 加密算法在严格意义上并不完全一样。
    - ✧ AES 的区块长度固定为128 位，密钥长度则可以是128，192或256位。
    - ✧ Rijndael 加密算法可以支持更大范围的区块和密钥长度，其使用的密钥和区块长度可以是32位的整数倍，以128位为下限，256位为上限。加密过程中使用的密钥由 Rijndael 密钥生成方案产生。
    - ✧ AES 加密过程在一个称为 state (体) 的 4×4 字节矩阵上进行，其初值是一个16字节 (128位) 的明文区块。
    - ✧ Rijndael 加密法支持更大的区块，其矩阵行数可视情况增加。

2.2 Symmetric Key Crypt. Algorithms

2.2.4 Advanced Encryption Standard (AES)

- **How AES Works**
  - 加密过程
    - ✧ 各轮 AES 加密循环 (除最后一轮外) 均包含4个步骤：
      - **AddRoundKey**：矩阵中的每一个字节都与该轮密钥 (round key) 做 XOR 运算；每个子密钥由密钥生成方案产生。
      - **SubBytes**：通过一个非线性的替换函数，用查找表的方式把每个字节替换成对应的字节。
      - **ShiftRows**：将矩阵中的每个横列进行循环式移位。
      - **MixColumns**：为了充分混合矩阵中各个直行的操作。这个步骤使用线性转换来混合每内联的四个字节。
    - ✧ 最后一个加密循环中省略 MixColumns 步骤，而以另一个 AddRoundKey 取代。

End of Chapter 2.2