



第3章 密码学基础

- 3.1 数据加密概述
- 3.2 传统密码技术
- 3.3 对称密码算法
- 3.4 公钥密码体制
- 3.5 数字签名技术
- 3.6 密钥分发管理



3.1 数据加密概述

- n 密码技术是保护信息安全的主要手段之一，是结合数学、计算机科学、电子与通信等诸多学科于一身的交叉学科。它不仅具有信息加密功能，而且具有数字签名、身份验证、秘密分存、系统安全等功能。所以使用密码技术不仅可以保证信息的机密性，而且可以保证信息的完整性和正确性，防止信息被篡改、伪造或假冒。
 - n 用户在计算机网络的信道上相互通信，其主要危险是被非法窃听。
 - n 采用搭线窃听，对线路上传输的信息进行截获；
 - n 采用电磁窃听，对用无线电传输的信息进行截获等。
- 因此，对网络传输的报文进行数据加密，是一种很有效的反窃听手段。通常是采用一定算法对原文进行软加密，然后将密码电文进行传输，即使被截获一般也是一时难以破译的。

两美元的“听诊器”就能破解Apple Pay?

“作案工具”包括以下四个：

- n 一个价值2美元的电磁波探测头
- n 一个 USB 声卡
- n 一台电脑
- n 一名掌握解密技术的黑客

真能破解？





口令攻击程序

- n 口令攻击程序是对原来加密的口令进行解密并使得口令显露出来的程序。如破解UNIX, NT等用户密码, zip, pdf, rar文件密码, 软件注册码等。
- 1. 强加密算法 单向加密 散列 通常采用穷举法, 即采用同样的加密算法对试探口令加密, 并对结果进行分析比较, 需较长时间, UNIX, ZIP等
- 2. 弱加密算法 非单向, 可逆推或很快就可穷举出来, 如CMOS密码, 一些软件的注册码, 注册机



基本概念

- n 明文：信息的原始形式（plaintext，记为P）。
- n 密文：明文经过变换加密后的形式（ciphertext，记为C）。
- n 加密：由明文变成密文的过程称为加密（enciphering，记为E），加密通常是由加密算法来实现的。
- n 解密：由密文还原成明文的过程称为解密（deciphering，记为D），解密通常是由解密算法来实现。
- n 密钥：为了有效地控制加密和解密算法的实现，在其处理过程中要有通信双方掌握的专门信息参与，这种专门信息称为密钥（key,记为K）。



基本概念

- n 密码技术通过信息的变换或编码，将机密的敏感信息变换成黑客难以读懂的乱码型文字，以此达到两个目的：
 - n 其一，使不知道如何解密的黑客不可能从其截获的乱码中得到任何有意义的信息；
 - n 其二，使黑客不可能伪造任何乱码型的信息。
- n 一般把要加密的报文（称为明文，plaintext），按照以密钥（key）为参数的函数进行变换，通过加密过程而产生的输出称为密文（ciphertext）或密码文件（cryptogram），破译密码的技术称为密码分析（cryptanalysis）。
- n 把设计密码的技术（加密技术）和破译密码的技术（密码分析）总称为密码技术（cryptology）。
- n 加密算法和解密算法是在密钥的控制下进行的，加密和解密过程中使用的密钥分别称为加密密钥和解密密钥。



基本概念

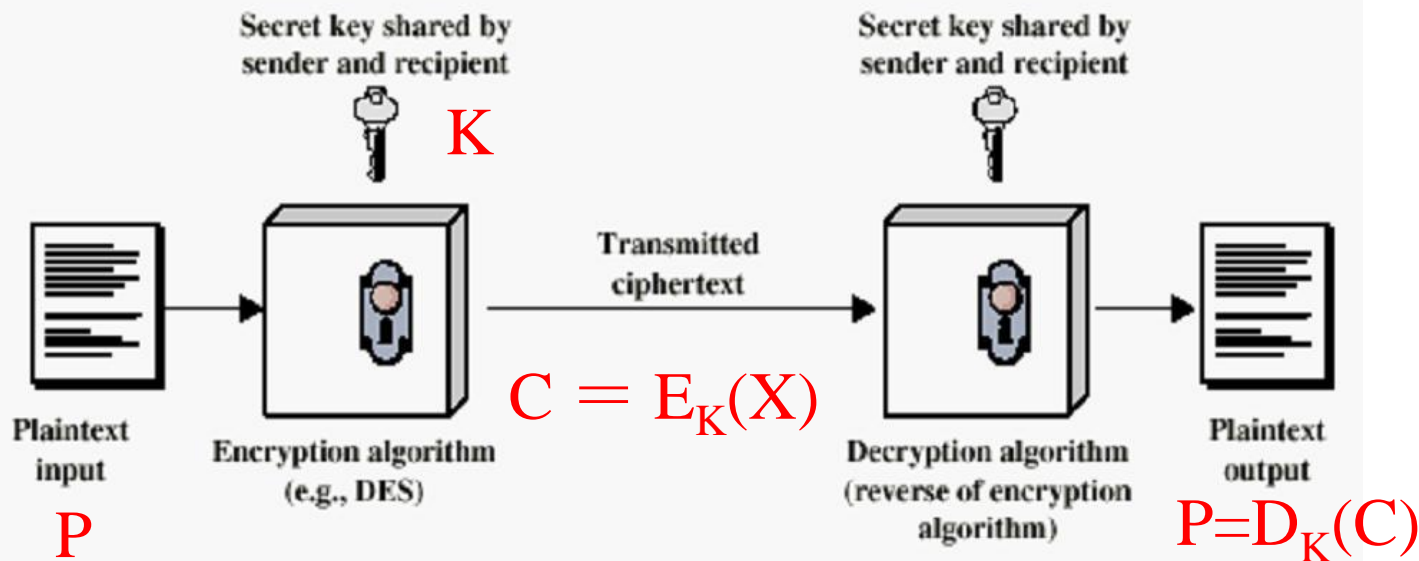
- n 用于将明文转换为密文操作的类型
所有的加密算法基于两个基本原则：
1. 替代 2. 置换
- n 使用密钥的数量
两种形式：常规加密（对称）经典技术和现代技术**DES** 单密钥加密，秘密密钥加密
公钥加密（非对称）**RSA** 双密钥加密，公开密钥加密
- n 明文处理的方式：分组加密输入块，流加密输入元素



3.2 传统密码技术

- n 对称，单密钥加密
- n An encryption scheme has five ingredients:
 - n Plaintext
 - n Encryption algorithm
 - n **Secret Key (the most important thing)**
 - n Ciphertext
 - n Decryption algorithm
- n Security depends on the secrecy of the key, not the secrecy of the algorithm

常规加密模型（简化的）



加密/解密算法公开，可用硬件实现，密钥不公开

Figure Simplified Model of Conventional Encryption



密码分析

- n 嗅探者能观察到C，但得不到K和P，可以复原P或K；
- n 若能得到K的一个估计 K' ，则可解密未来的消息；若只对特定的消息感兴趣，则产生一个明文估计 P' 来复原P。
- n 试图发现P或K或这两者的过程称为密码分析cryptology cryptanalysis，使用的策略取决于加密方案的性质以及可用的信息



密码分析

n 密文——>算法——明文

统计分析 穷举密钥

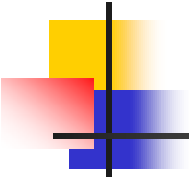
n 安全:

1 破译成本超过加密信息的价值

2 破译时间超过加密信息的生命周期

n 无条件安全:

(one-time pad) 不可解密, 一次性口令



Average time required for exhaustive key search 穷举估时

Key Size (bits)	Number of Alternative Keys	Time required at 10^6 Decryption/ μ s
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years



加密技术的基本构造模块

1 替代 substitution ciphers

明文中的每个元素（比特，字母，比特组合或字母组合）被映射为另一个元素，密钥字母表，可以由一个明文字母表构成，也可以由多个明文字母表构成。

- a. 凯撒密码（已知最早的替代密码）字母表的每个字母用该字母的后面第三个字母进行替代
plaintext --- ciphertext

Caesar cipher (Julis Caesar)

$$c = E(p) = (p+3) \bmod (26)$$

$$p = D(c) = (c-3) \bmod (26)$$



1 替代 substitution ciphers

- n 以英文26个字母为例，它用D表示a，用E表示b，用F表示c，...，用C表示z，也就是说密文字母相对明文字母循环左移了3位，因此，又称为循环移位密码。这种映射关系表示为如下函数：

$$F(a) = (a + k) \bmod n$$

其中：a表示明文字母，n为字符集中字母个数，k为密钥。

- n 假设k=3，对于明文 P=COMPUTER SYSTEMS 则

$$f(C) = (2 + 3) \bmod 26 = 5 = F, \quad f(O) = R, \quad f(M) = P, \\ f(M) = P, \quad f(S) = V$$

所以，密文 $C = EK(P) = FRPSXWHUVBVWHPV$ 。



1 替代 substitution ciphers

- n 通用的Caesar cipher移位 $k = 1 \sim 25$
 - n $c = E(p) = (p+k) \bmod (26)$
 - n $p = D(c) = (c-k) \bmod (26)$
- n 强行攻击: $k = 1 \sim 25$ 密钥进行试探, 分析解密结果
- n 具备的条件
 - n 1 加解密算法已知
 - n 2 密钥仅有25个
 - n 3 明文语言已知并容易识别
- n 单一字母替代法: monoalphabetic 非常不安全
- n 简单扩展字母替代法: 任意排列, 密钥个数为 $26!$ 个, 比 $DES2^{56}$ 还大 10^{10}
- n 例如, 在字母表中首先排列出密钥中出现的字母, 然后在密钥后面填上剩余的字母。
密 钥 JOY, A用J替代、B用O替代...Z用Z替代, 对于明文
 $P=COMPUTER\ SYSTEMS$ 则密文为: YMKNTSBQ RXRSBKR。



1 替代 substitution ciphers

- n 替代法进行加密或解密可以看成是直接查映射表来实现的。变换一个字符只需要一个固定的时间，这样加密 n 个字符的时间与 n 成正比。短字、有重复模式的单词，以及常用的起始和结束字母都给出猜测字母表排列的线索。
- n 频率分析+试探：单字母频率，双字母频率分析 digrams, 三字母频率分析 trigrams
- n 例如，在英语中，最常用的字母是e，其次是t，再其次是a，o，n，i。组合or，an，the，and，that等。



1 替代 substitution ciphers

b. 多字母加密密码

按照明文的位置，用不同的密文字母代替明文字母，例如，the这种三字母组合，将根据它们在明文中的位置，在密文中会映射出不同的三字母组合

n playfair密码 5x5字母矩阵 25个字母(I, J算一个)

频率分布较平坦，但仍可分析

n 维吉尼亚Vigenere 密码 替代规则由 26 个 Caesar cipher组成 表格

密钥长度 = 明文长度 = 密文长度

密钥重复



Vigenere密码

- n 密码表是以字母表移位为基础，把26个英文字母进行循环移位，排列在一起，形成 26×26 的方阵，该方阵被称为Vigenere代换表。
- n 加密过程很简单，就是给定密钥字母x和明文字母y，密文字母是位于x行和y列的那个字母。这样就决定了加密一条明文需要与明文一样长的密钥字符串，通常，密钥字符串是密钥词的重复。

密钥词是deceptive，明文是“we are discovered save yourself”，那么加密过程如下：

deceptivedeceptivedeceptive（密钥字符串）

wearediscoveredsaveyourself（明文）

ZICVTWQNGRZGVTWAVZHCQYGLMGJ（密文）

- n 扩展一下，以上加密仅能对26个字母进行加密，而且不能区分大小写。但其实英文中除了字母外，还有标点符号，还有空格。如果假设能被加密的字符有N个，如果把这N个字符建成一个环，那么加密过程就是模N的过程，即， $C(i) = (K(i) + P(i)) \bmod N$ ，其中K、C、P分别代表的是密钥空间、密文空间、明文空间。

ABCDEFGHIJKLMNOPQRSTUVWXYZ
BCDEFGHIJKLMNOPQRSTUVWXYZA
CDEFGHIJKLMNOPQRSTUVWXYZAB
DEFGHIJKLMNOPQRSTUVWXYZABC
EFGHIJKLMNOPQRSTUVWXYZABCD
FGHIJKLMNOPQRSTUVWXYZABCDE
GHIJKLMNOPQRSTUVWXYZABCDEF
HIJKLMNOPQRSTUVWXYZABCDEFG
IJKLMNOPQRSTUVWXYZABCDEFGH
JKLMNOPQRSTUVWXYZABCDEFGHI
LMNOPQRSTUVWXYZABCDEFGHIJK
MNOPQRSTUVWXYZABCDEFGHIJKL
NOPQRSTUVWXYZABCDEFGHIJKLM
OPQRSTUVWXYZABCDEFGHIJKLMN
PQRSTUVWXYZABCDEFGHIJKLMNO
QRSTUVWXYZABCDEFGHIJKLMNOP
RSTUVWXYZABCDEFGHIJKLMNO
STUVWXYZABCDEFGHIJKLMNO
TUVWXYZABCDEFGHIJKLMNO
UVWXYZABCDEFGHIJKLMNO
VWXYZABCDEFGHIJKLMNO
WXYZABCDEFGHIJKLMNO
XYZABCDEFGHIJKLMNO
YZABCDEFGHIJKLMNO
ZABCDEFGHIJKLMNO



1 替代 substitution ciphers

- n 多字母密码要比单字母密码好，但只要给密码分析员以足够数量的密文，总还是可以进行破译的，这里的加密关键在于密钥。通常进一步采用的方法是：加长密钥长度或采用随机的二进制串作为密钥。
- n 不一定是每次都只研究一个字母。例如，坡他密码(Portacipher)，采用 26×26 的表。每次把明文看成两个字符(偶对)的密码，由第一个字符指示行，第二个字符指示列，由此产生的交叉点的数字或字母偶对就是译出的密码值。



2. 置换技术 permutation

Transposition ciphers (permutation)

Computer Network Andrew S. Tanenbaum

- n 明文中的元素被重排列，本身不变，但位置变了，而不是隐藏元素。所有操作可逆转
- n 最简单的例子：把明文中的字母的顺序倒过来写，然后以固定长度的字母组发送或记录，如：
明文： computer systems
密文： smetsys retupmoc
- n 栅栏技术，对角线写下明文，以行的顺序输出，如深度2
meet me after the toga party
Mematrh...
etefete...
- n 易破解



2. 置换技术 permutation

- n 复杂方案：矩阵
- n 逐行写出明文，不够填其他字母，再逐列读出，得密文。
列规则为密钥 可以为一单词
4312567 7列
attackp
ostpone
duntilt
woamxyz
- n 调整列，再逐行读出，得密文，密钥就是矩阵的行数 m 和列数 n ，即 $m \times n$ ，以及给定的列置换矩阵 f ，也就是 $k = (m \times n, f)$

2. 置换技术 permutation

n 明文ENGINEERING按行排在 3×4 矩阵中，如最后一行不全可用A, B, C...填充，如下所示。

n	1	2	3	4
n	E	N	G	I
n	N	E	E	R
n	I	N	G	A

n 给定一个置换

n $f = ((1234)(2413))$

2	4	1	3
N	I	E	G
E	R	N	E
N	A	I	G

得到密文

NIEGERNENAIG。

- 字母频率分析 + 试探列规则
- 多次置换，分析较困难

c. 异或操作



3.3 对称密码算法

1 分组密码原理

n 分组长度太短 = 古典替代密码

n Feistel 分组密码 替代，置换交替进行

Claude Shannon

n 扰乱 (**confusion**) 试图使得密文的统计特性与加密密钥的取值之间的关系尽量复杂，用一个复杂的替代算法使得攻击者即使掌握了密文的统计特性也难于推测出密钥

n 扩散 (**diffusion**) 明文的统计结构被扩散消失到密文的长程统计特性中，让明文的每个数字影响许多密文数字的取值，等价地说，每个密文数字被许多明文数字影响，如平均操作，加密重复置换



Feistel 密码结构

- n 分组大小：越大越安全
- n 密钥大小：越大越安全
- n 加密轮数：多轮加密可提高安全强度
- n 子密钥产生算法：复杂性越大越难破译
- n 软件加密/解密速度要快



2 数据加密标准DES

- n 数据加密标准 DES(Data Encryption Standard)是美国国家标准局开始研究除国防部以外的其他部门的计算机系统的数据加密标准，于1972年和1974年美国国家标准局(NBS)先后两次向公众发出了征求加密算法的公告。
- n 对加密算法要求要达到以下几点：
 - (1) 必须提供高度的安全性；
 - (2) 具有相当高的复杂性，使得破译的开销超过可能获得的利益，同时又便于理解和掌握；
 - (3) 安全性应不依赖于算法的保密，其加密的安全性仅以加密密钥的保密为基础；
 - (4) 必须适用于不同的用户和不同的场合；
 - (5) 实现经济、运行有效；
 - (6) 必须能够验证，允许出口。
- n 1977年1月，美国政府采纳IBM公司设计的方案作为非机密数据的正式数据加密标准(DES)。DES被授权用于所有公开的和私人的非保密通信场合，后来它又曾被国际标准组织采纳为国际标准。



数据加密标准DES

数据加密标准DES - feistel结构

单钥密码算法，是一种典型的按分组方式加密
内部结构

- n S盒子（替代）设计标准保密

- n P盒子（置换）

- n 雪崩效应

明文或密钥的一点小的变动使密文发生一个大的变化，一个比特变化引起密文许多比特变化

- n DES算法是对称的，既可用于加密又可用于解密

- n 当用56位密钥时，可能的组合大于 7.2×10^6 种

- n DES算法现已在VLSI芯片上实现了



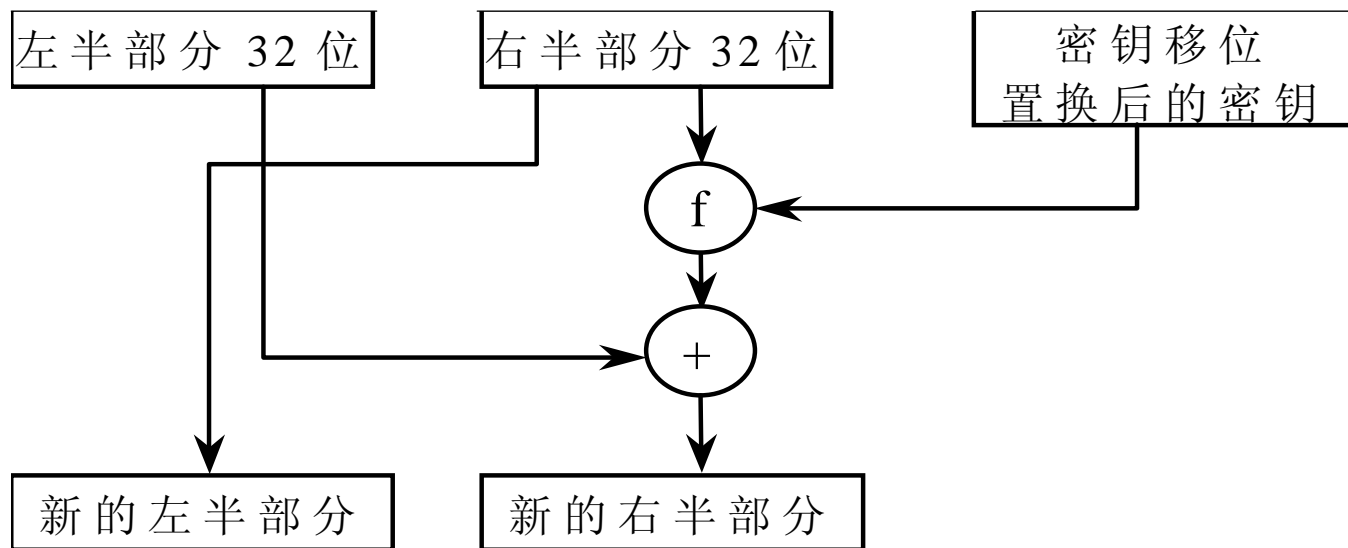
Data Encryption Standard (DES)

- n 分组加密算法，以64位为分组，先要经过初始置换处理
- n 扰乱和扩散的组合，一次加密或解密总共有16轮，也就是要完成一次加密（解密）过程，必须在明文（密文）分组上实施16次相同的组合技术
- n 在最后一轮之后，进行逆初始置换运算，它是初始置换的逆，最后得到64位密文。
- n 密钥长度：56位，密钥通常表示为64位的数，但每个字节的第8位都用作奇偶校验，可以忽略

初始置换IP和初始逆置换IP⁻¹

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

Data Encryption Standard (DES)



每一轮DES加密原理示意图



Data Encryption Standard (DES)

n 每一轮DES:

n $L_i = R_{i-1}$

n $R_i = L_{i-1} \otimes F(R_{i-1}, K_i)$

n 函数f的细节：数据右半部分通过扩展由32位扩展为48位，并通过一个异或操作与经过移位和置换的48位密钥结合，其结果通过8个S-盒（substitution box）将这48位替代成新的32位数据，再将其置换一次

n 生成每一轮的密钥——子密钥

n 解密算法和加密算法相同，但子密钥使用次序相反

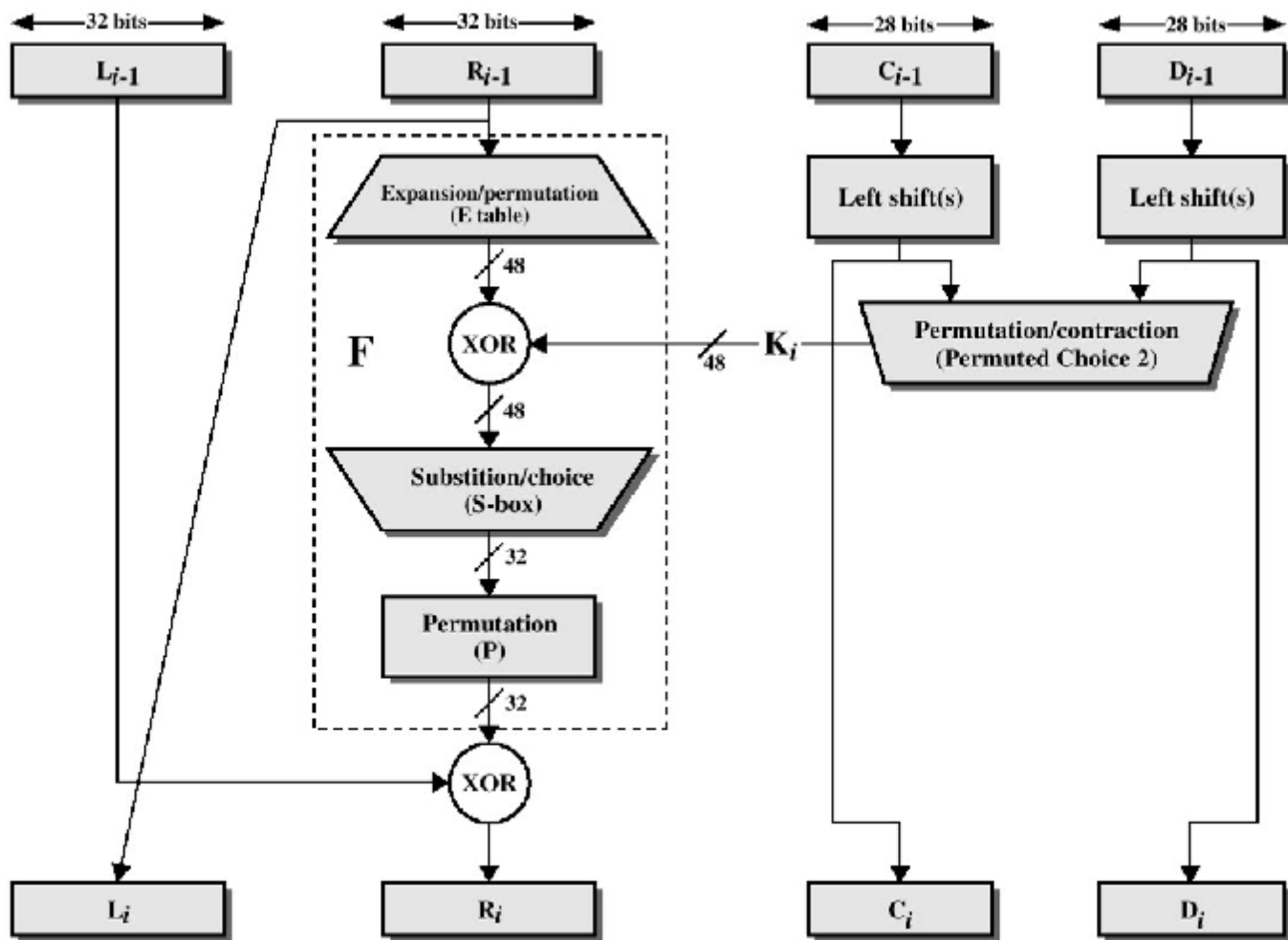
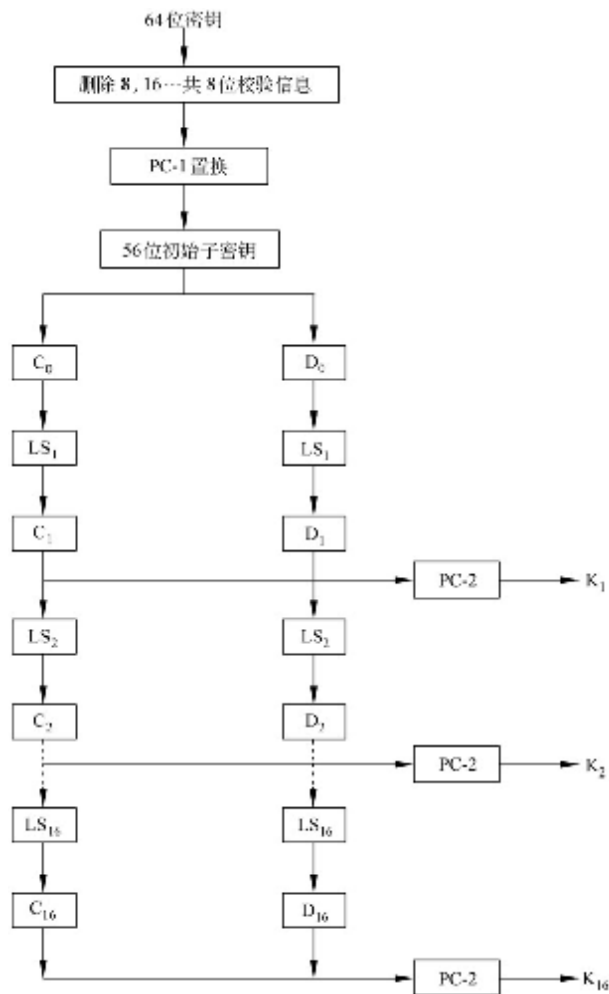
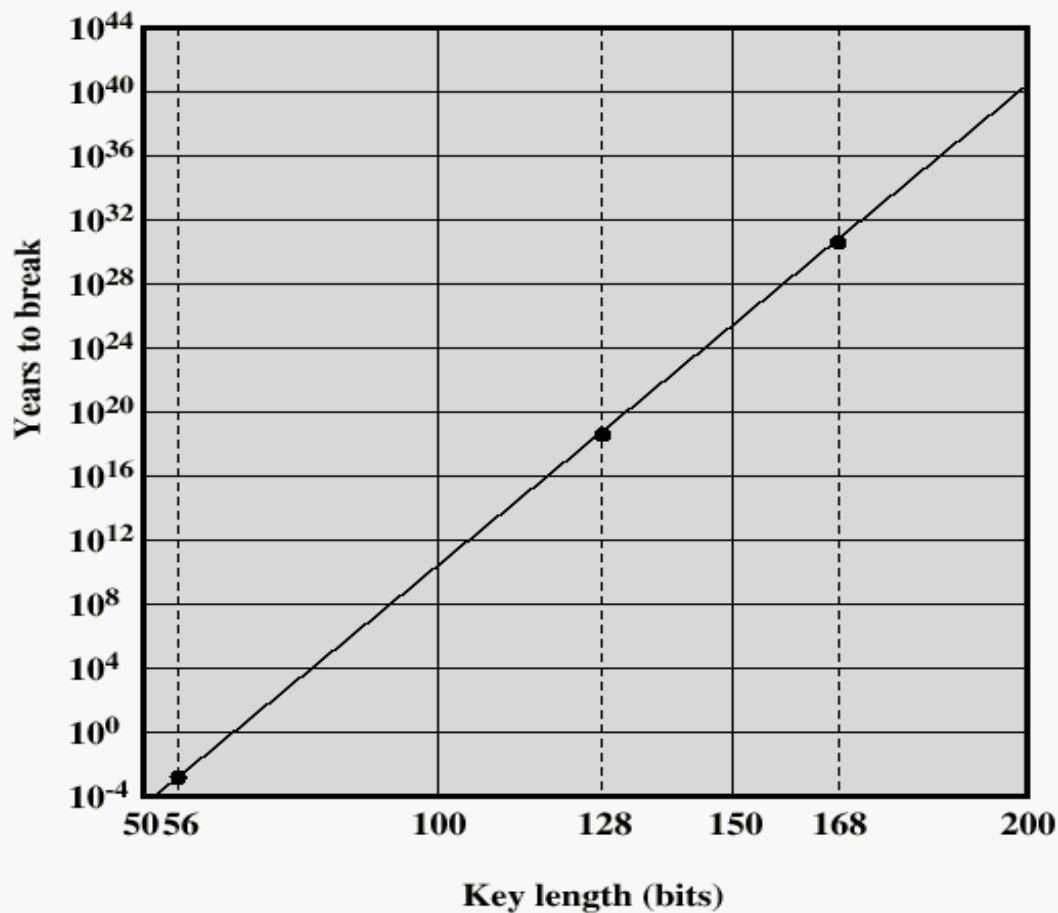


Figure Single Round of DES Algorithm



16轮子密钥

破译时间 (10^6 次解密/ μs)





DES强度

- n DES的强度：56比特的密钥长度
 - n 1997年1月提出挑战，DESCHALL计划，开始时只有几百人参与，最终吸引了数万名志愿者参加。利用Internet的分布式计算能力，在96天后攻破。
 - n DES的全部密钥穷举量为 $7.2e16$ ，DESCHALL计划完成时，搜索的密钥量为 $1.773e16$ ，占全部密钥穷举量的24.6%，平均每天最多搜索 $6.0e14$ 个，每秒最多搜索 $7.0e9$ 个，其中最后24小时搜索了 $5.59e14$ 个，占全部穷举量的0.7%，假若一开始就以这个速度搜索，则DESCHALL计划只需32天即可完成。
 - n 2008-11 德国FPGA并行计算机 平均破译时间小于1天
 - n 计算能力的增长和新的攻击算法，要求算法密钥的长度必须增加。
- n 最近的一次评估是在1994年1月，当时决定1998年12月以后，DES不再作为联邦加密标准。



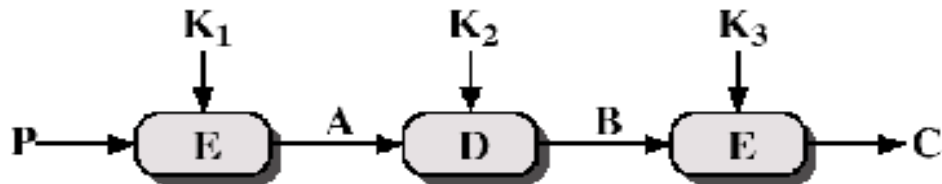
Triple DES

- n 1997 DES破解比赛，1999 3DES过渡
- n 采用三个密钥，执行三次DES
(加密-解密-加密)

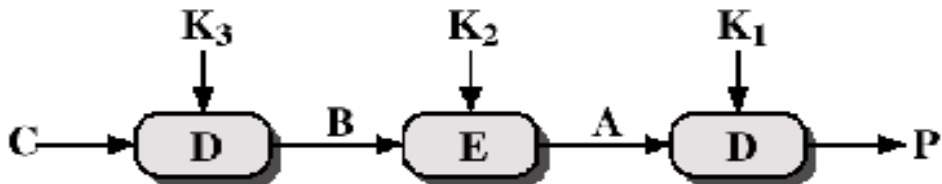
$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$

- n C = 密文
- n P = 明文
- n $E_K[X]$ = 用密钥 K 加密 X
- n $D_K[Y]$ = 用密钥 K 解密 Y
- n 有效密钥长度168位

Triple DES



(a) 加密



(b) 解密

Triple DEA



Advanced Encryption Standard (AES)

- n DES的设计主要针对硬件实现，目前已不安全。而当今许多领域的问题需要用软件方法来实现，AES加密方法在这种需求下应运而生。
- n AES 算法是128位块密码，支持三种不同大小的密钥：128，192和256位。
- n AES密码算法采用的是替代-置换网络结构，每一轮操作由4层组成：
 - n 第1层（字节替换）为非线性层，用S盒对每一轮中的单个字节分别进行替换；
 - n 第2层（行移位）和第3层（列混合）是线性混合层，对当前的状态按行移位，按列混合；
 - n 第4层（密钥加层）用子密钥与当前状态进行字节上的异或。



其他对称加密算法

n International Data Encryption Algorithm (IDEA)

IDEA算法又叫国际数据加密算法，是瑞士联邦技术学院开发的一种面向数据分组块的数据加密标准。相对于DES的56位密钥，它使用128位密钥，每次加密一个64位的数据块。这个算法被加强以防止一种特殊类型的攻击，即微分密码分析攻击。任何人都可以得到这个算法，它的安全与DES算法一样不在隐藏算法本身，而在于保存好密钥。

- n 密钥长度128位，采用混乱和扩散等操作：异或、模加、模乘，容易用软件和硬件实现。
- n 安全性相对DES算法有很大的提高
- n 用在 PGP

n Blowfish

- n 实现容易
- n 速度快
- n 内存要求小于5K



对称密钥密码的分析方法

- n 密码编码学与密码分析学的对立性促进了密码学的发展。根据密码分析者对明文、密文等信息掌握的多少，可以将密码分析分为以下五种情形：
 - n 唯一密文攻击
 - n 已知明文攻击
 - n 选择明文攻击
 - n 选择密文攻击
 - n 选择文本攻击



对称密钥密码的分析方法

n 具体的分析方法主要包括：

1. 强力攻击法

- n 强力攻击可用于任何分组密码，且攻击的复杂度仅依赖于分组长度和密钥长度。工作效率包括加/解密速度、密钥扩展速度、存储空间等。

2. 差分密码分析

- n 已知最有效的攻击迭代密码的方法之一。基本思想是通过分析明文对的差值对密文对的差值的影响来恢复某些密钥比特。
- n 差分密码分析最初是针对DES加密提出的一种攻击方法，能成功破解轮数较低的DES。



对称密钥密码的分析方法

3. 线性密码分析

- 本质上是一种已知明文攻击法，是对DES加密方法进行破译的主要方法。基本思想是通过寻找一个给定密码算法的有效的线性近似表达式来破译密码系统。

4. 差分-线性密码分析

- 对差分密码分析和线性密码分析进行改进，是降低它们复杂度的众多改进之一。它利用的是差分密码分析和线性密码分析相结合的技术。

5. 插值攻击

- 利用了拉格朗日插值公式的思想。如果一个密码算法是固定的密钥的低次多项式函数，或项数较少的多项式，其项数可以估算出来，则通过插值法可以得到其代数表达式，从而可能恢复出密钥。

6. 微分密码分析法

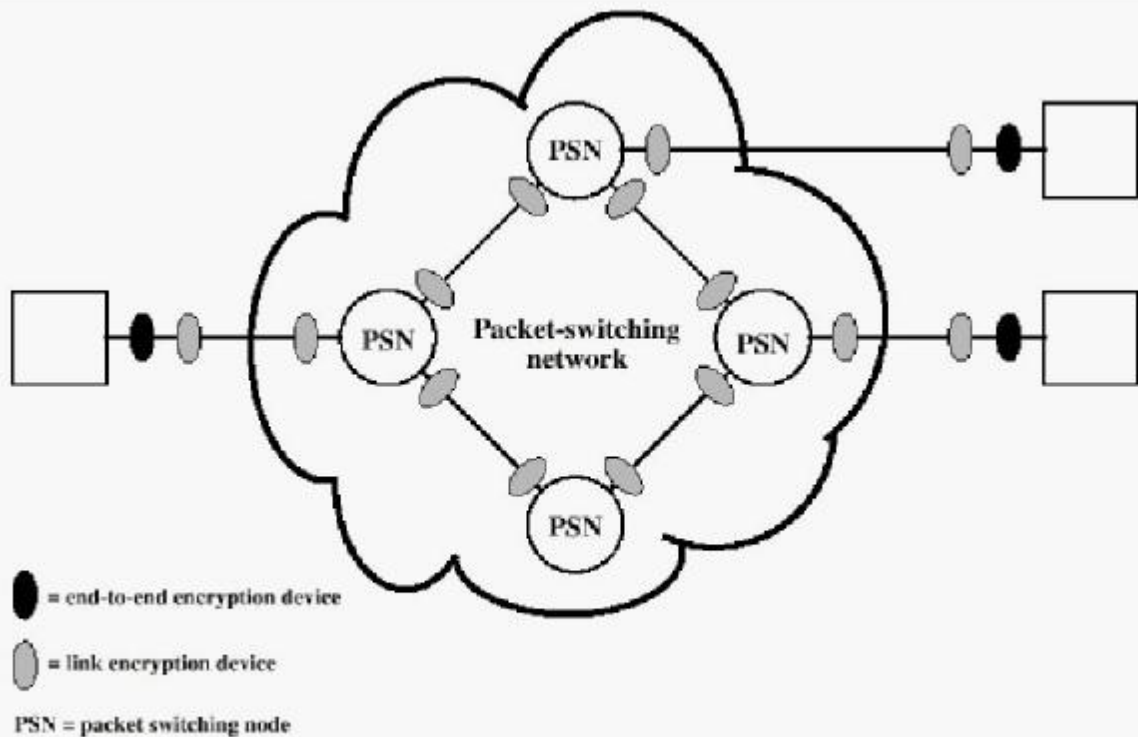
- 1990年，两名以色列密码专家发明，破解DES。按照这一方法只需要对特殊的明文和密文成对采样247对，通过短时间的分析便可以得到正确的key.



使用常规加密进行保密通信

加密功能的位置 - 三种加密方式

- n 链路加密方式 Link encryption:
 - n 加密设备多
 - n 保密性好
 - n 在每一个交换设备都需加解密
- n 端对端加密方式 End-to-end encryption
 - n 发送端加密，接收端解密
 - n 数据加密
 - n 头部不加密
- n 高保密 High Security: 前两种方法结合



Encryption Across a Packet-Switching Network



3.4 公钥密码体制

- n 在公开的计算机网络上安全地传送和保管密钥是一个严峻的问题。1976年，Diffie和Hellman为解决密钥管理问题，在他们具有奠基性意义的“密码学的新方向”论文中，提出了一种密钥交换协议，允许在不安全的媒体上通信双方交换信息，安全地达成一致的密钥。在此新思想的基础上，很快出现了“不对称密钥密码体制”，即“公开密钥密码体制”。其算法使用一对密钥，一个密钥公之于众，谁都可以用；而另一个密钥只有一个人知道。它们分别称为“公开密钥”(public-key)和“私有密钥”(private-key)。
- n 1978年后Rivest, Shamir和Adleman提出的RSA算法体现了公钥算法的思想。
- n 公钥密码体制是现代密码学的一个标志，是目前密码学史上最大也是唯一真正的革命。
- n 公钥密码体制是建立在数学函数基础上的

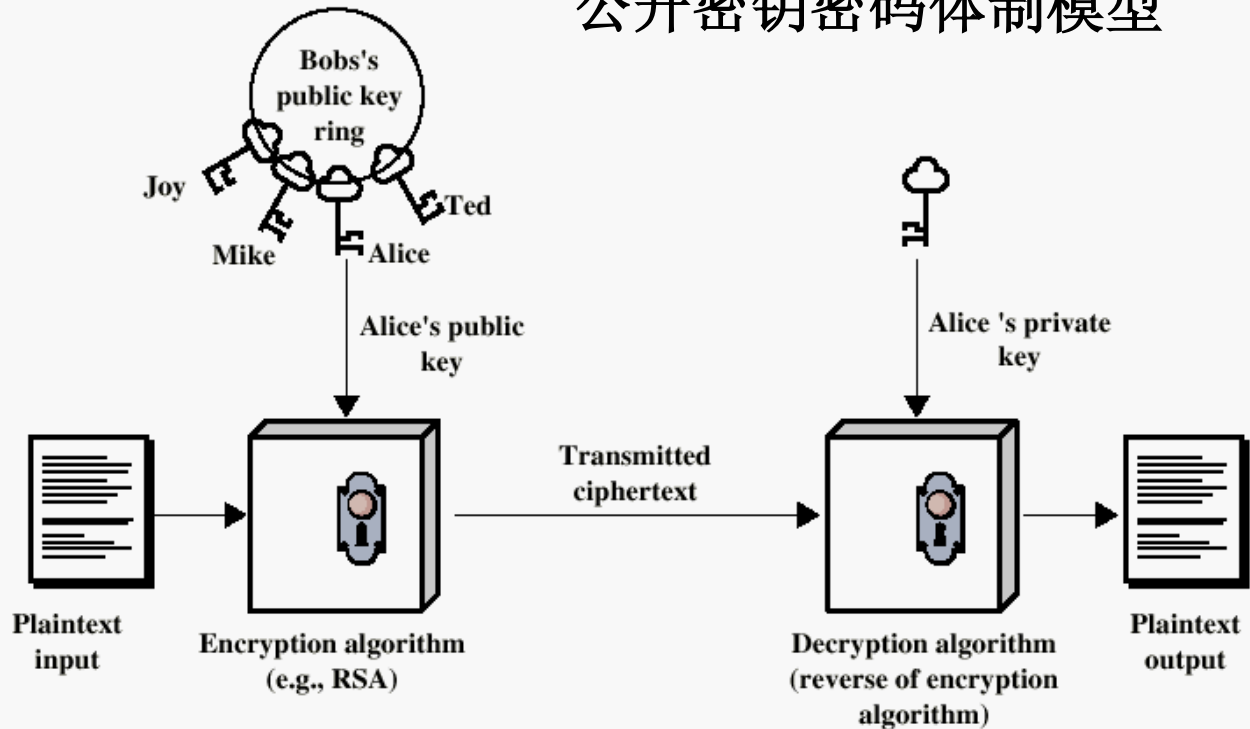


1. 公开密钥加密算法的特点

- n DES加密算法及其类似算法属于传统密码体制，要求加密和解密的密钥是相同的，因此密钥必须保密。而Diffie和Hellman研究出的公开密钥密码体制新算法：使用一个加密算法 $E(K_a)$ 和一个解密算法 $D(K_b)$ ，它们彼此完全不同，根据已选定的 E 和 D ，即使已知 E 的完整描述，也不可能推导出 D 。这给密码技术带来了新的变革。
- n 此种新算法需有以下3个条件：
 - (1) $D(E(P))=P$;
 - (2) 由 E 来推断 D 极其困难;
 - (3) 用已选定的明文进行分析，不能破译 E 。
- n 第1条说明，采用解密算法 D 用于密码报文 $E(P)$ 上，可以得到原来的明文 P ；第2个条件，显而易见，必需的；第3个条件，也是必需的。在满足这3条的情况下，加密算法 E 可以公开。

Encryption using Public-Key system

公开密钥密码体制模型





公开密钥算法的基本工作过程

- n 每一个终端产生其自身用于通信的加密和解密密钥
- n 各个终端将其自身的加密密钥通过某种方式向外界公布
 - n 目录服务、公共服务器、公共文件等
 - n 被公布的加密密钥称为公开密钥
 - n 另一个被个终端自己保存的解密密钥称为私有密钥
- n 通常，公开密钥和私有密钥应可以互为加密/解密密钥
- n 若A 需向B 发送消息，可用B 的公开密钥对消息进行加密
 - n B 收到消息后，可用其私有密钥对消息进行解密
 - n 由于公钥密码算法的特性，可以保证只有B 才能正确地解读消息
- n 任何人都可以使用B 的公开密钥来向B 发送加密消息，但只有B 能够解密



1. 公开密钥加密算法的特点

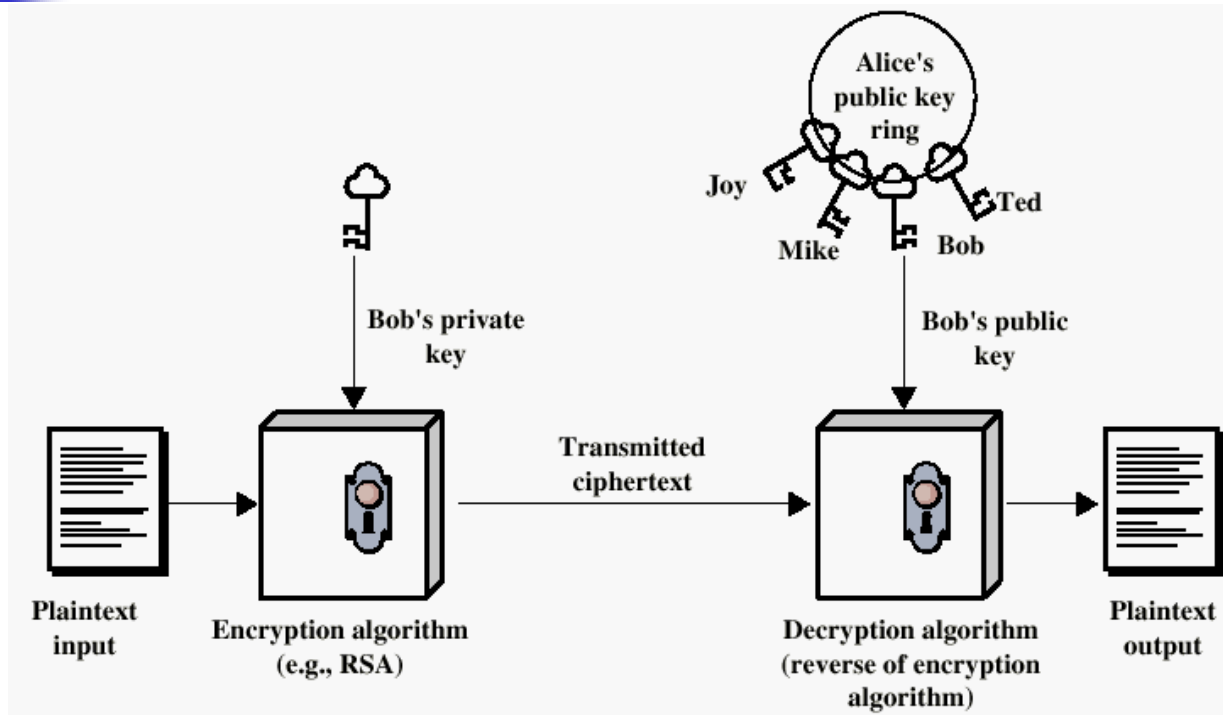
- n 公钥体系包括六个元素：
 - n 明文 Plaintext
 - n 加密算法 Encryption algorithm
 - n 公钥/私钥对 Public and private key
 - n 解密算法 Decryption algorithm
 - n 密文 Ciphertext



Applications for Public-Key Cryptosystems

- n 公钥加密系统的应用 三类:
 - n 加/解密: 发送方用接受方的公钥加密.
 - n 数字签名 **Digital signature**: 发送方用自己的私钥给消息加密或产生认证码.
 - n 密钥交换: 通信双方交换一个会话密钥

Authentication using Public-Key System



采用公钥系统的认证



Requirements for Public-Key Cryptography

算法要求

- n 容易计算出B的密钥对 (公开密钥 K_{Ub} , 私钥密钥 K_{Rb})
- n 发送者容易产生密文:

$$C = E_{K_{Ub}}(M)$$

- n 接受者容易用私钥解密密文:

$$M = D_{K_{Rb}}(C) = D_{K_{Rb}}[E_{K_{Ub}}(M)]$$



Requirements for Public-Key Cryptography

- n 知道公钥 (KUb) 计算不出私钥 (KRb)
- n 知道公钥 KUb 和密文 C 不能恢复原来的消息M即明文
- n 密钥对的任何一个都可以用作加密密钥，另一个用作解密密钥：

$$M = D_{KRb}[E_{KUb}(M)] = D_{KUb}[E_{KRb}(M)]$$



公钥加密算法

RSA and Diffie-Hellman

- n RSA - Ron Rivest, Adi Shamir and Len Adleman at MIT, in 1977.
 - n RSA 是数据块加密，基于素数理论，欧拉定理
 - n 使用最广泛的公钥算法
- n Diffie-Hellman
 - n 安全交换密钥
 - n 基于离散对数



The RSA Algorithm – Key Generation

- n Select p, q both prime 素数 (secret)
欧拉函数 $\Phi(n) = (p-1)(q-1)$
- n Calculate $n = p \times q$ (n public)
- n Select integer e $\gcd(\Phi(n), e) = 1; 1 < e < \Phi(n)$
- n Calculate d $d = e^{-1} \bmod \Phi(n)$
- n Public Key $KU = \{e, n\}$
- n Private key $KR = \{d, n\}$



RSA的加解密过程

n Encryption 加密

- n Plaintext消息: $m < n$

- n 加密算法

- n Ciphertext密文: $C = E(m) = m^e \pmod n$

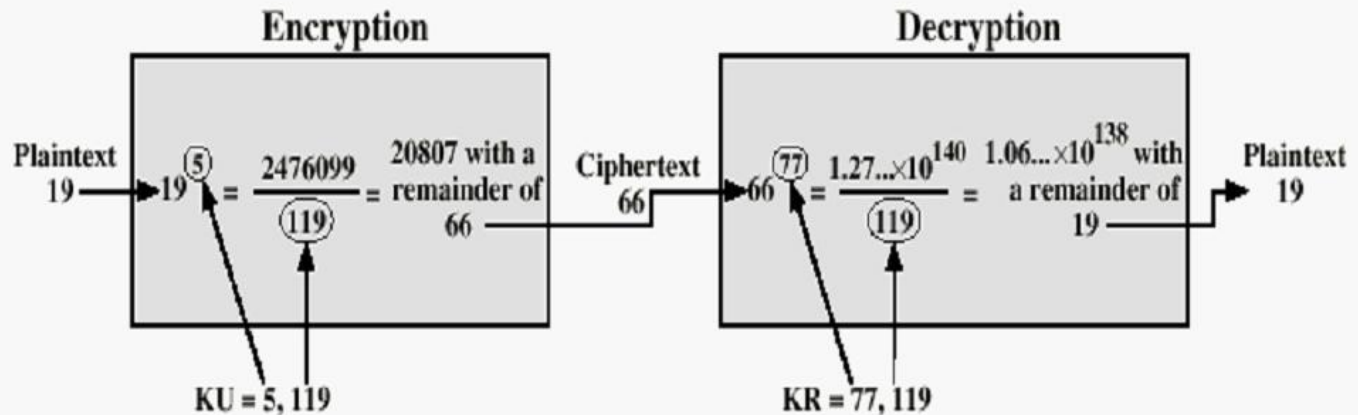
n Decryption 解密:

- n Ciphertext密文: C

- n 解密算法: $m = D(c) = c^d \pmod n$

- n Plaintext: $m = C^d \pmod n$

Example of RSA Algorithm



$$p=7, q=17$$

$$\Phi(n) = (p-1)(q-1) = 96$$

$$n = p \times q = 119$$

$$e=5, d=77, \text{exd mod } 96 = 1$$

Example of RSA Algorithm



RSA算法的安全性

- n 对RSA算法的数学攻击实际上等效于对模 n 乘积因子的分解。
- n 对素数 p 和 q 的选取的一些限制：
 - n p 和 q 的长度相差不能太大
 - n $p-1$ 和 $q-1$ 都应有大的素数因子
 - n $\gcd(p-1, q-1)$ 应该偏小
- n 随着计算机计算能力的不断提高，原来被认为是不可能分解的大数已被成功分解。目前密钥长度不小于1024bit的RSA算法是安全的，有些场合则要求2048bit。



RSA numbers

From Wikipedia

RSA Factoring Challenge - RSA Laboratories March 18, 1991, ended in 2007

- n RSA-100: 100 decimal digits (330 bits). April 1, 1991 by Arjen K. Lenstra, a few days using the multiple-polynomial quadratic sieve algorithm on a MasPar parallel computer, or four hours using the program Msieve on a 2200 MHz Athlon 64 processor, or 72 minutes on overclocked to 3.5 GHz Intel Core2 Quad q9300, using GGNFS and Msieve binaries running by distributed version of the factmsieve Perl script
- n RSA-110: 110 decimal digits (364 bits), April 1992 by Arjen K. Lenstra and Mark S. Manasse in approximately one month.
or less than 4 hours on overclocked to 3.5 GHz Intel Core2 Quad q9300, using GGNFS and Msieve binaries running by distributed version of the factmsieve Perl script
- n RSA-120: 120 decimal digits (397 bits), June 1993 by Thomas Denny, Bruce Dodson, Arjen K. Lenstra, and Mark S. Manasse 3 months
- n RSA-129: decimal digits (426 bits), April 1994 by a team led by Derek Atkins, Michael Graff, Arjen K. Lenstra and Paul Leyland, using approximately 1600 computers from around 600 volunteers connected over the Internet
- n RSA-130: 130 decimal digits (430 bits), April 10, 1996
- n RSA-140: 140 decimal digits (463 bits), February 2, 1999
- n RSA-150: 150 decimal digits (496 bits), 2004 (was withdrawn from the challenge by RSA Security)

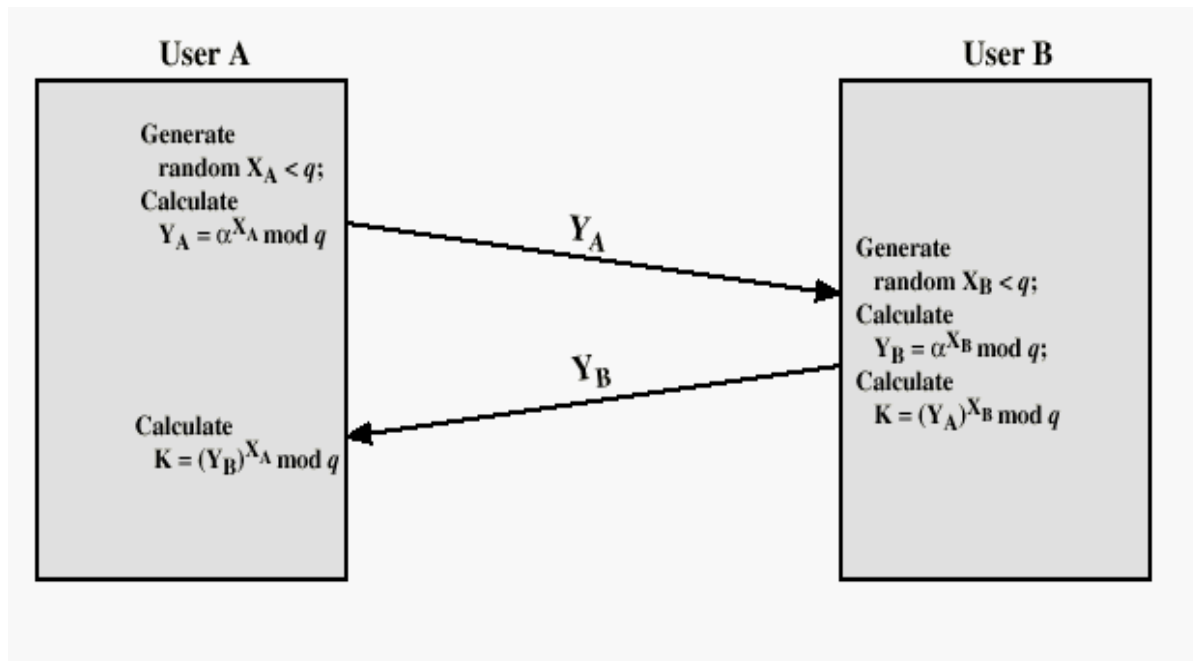


RSA numbers

From Wikipedia

- n RSA-155: 155 decimal digits (512 bits), August 22, 1999 in a span of 6 months
- n RSA-160: 160 decimal digits (530 bits), April 1, 2003
- n RSA-170: 170 decimal digits (563 bits), December 29, 2009
- n RSA-576: 174 decimal digits (576 bits), December 3, 2003
- n RSA-180: 180 decimal digits (596 bits), May 8, 2010
- n RSA-190: 190 decimal digits (629 bits), Russia and Netherlands.
- n RSA-640: 640 bits (193 decimal digits), November 2, 2005 5 months on 80 2.2 GHz AMD Opteron CPUs
- n RSA-200: 200 decimal digits (663 bits), May 9, 2005 by a collection of parallel computers amounted – very approximately – to the equivalent of 75 years work for a single 2.2 GHz Opteron-based computer
- n RSA-210: 210 decimal digits (696 bits), September 2013
- n RSA-704: 704 bits (212 decimal digits), July 2, 2012
- n RSA-220: decimal digits (729 bits), RSA-230: 230 decimal digits (762 bits), RSA-232 : 232 decimal digits (768 bits)
- n **RSA-768 : 232 decimal digits (768 bits), December 12, 2009** over the span of 2 years, by a collection of parallel computers amounted approximately to the equivalent of almost 2000 years of computing on a single-core 2.2 GHz AMD Opteron-based computer
- n RSA-240: 240 decimal digits (795 bits), RSA-250: 250 decimal digits (829 bits), RSA-309: 309 decimal digits (1,024 bits), RSA-1024: 1,024 bits (309 decimal digits), RSA-310: 310 decimal digits (1,028 bits),.....RSA-617: 617 decimal digits (2,048 bits), RSA-2048: 617 decimal digits (2,048 bits).

Diffie-Hellman 密钥交换



n Diffie-Hellman (D-H) 密钥交换协议中的公钥加密算法



Diffie-Hellman密钥交换算法

- n Diffie-Hellman密钥交换算法的有效性依赖于计算离散对数的难度。
- n 没有提供双方身份的任何信息。
- n 它是计算密集性的，因此容易遭受阻塞性攻击，即对手请求大量的密钥，受攻击者花费了相对多的计算资源来求解无用的幂系数而不是在做真正的工作
- n 没办法防止重演攻击

- n Oakley算法是对Diffie-Hellman密钥交换算法的优化，它保留了后者的优点，同时克服了其弱点
- n Oakley算法具有五个重要特征：
 - n 采用称为cookie程序的机制来对抗阻塞攻击
 - n 使双方能够协商一个全局参数集合
 - n 使用时效性来保证抵抗重演攻击
 - n 能够交换Diffie-Hellman公开密钥
 - n 对Diffie-Hellman交换进行鉴别以对抗中间人的攻击



其他公钥加密算法

- n 背包算法、Rabin(Rabin的加密法可以说是RSA方法的特例)
- n ElGamal 数字签名标准 Digital Signature Standard (DSS)
 - n 使用 Secure Hash Algorithm SHA-1
 - n 不适用加密和密钥交换
- n 椭圆曲线加密算法
Elliptic-Curve Cryptography (ECC)
 - n 同样的安全强度，ECC算法所需的密钥长度远比RSA算法低
 - n 目前应用较少，Microsoft
 - n 非常复杂



公钥密码体制的优缺点

n 优点:

- n 解决了密钥传递的问题
- n 大大减少密钥持有量
- n 提供了对称密码技术无法或很难提供的服务（数字签名）

n 缺点:

- n 计算复杂、耗用资源大 几百倍、几千倍甚至上万倍。
- n 非对称会导致得到的密文变长

- n 使用RSA不对称密钥密码体制来传送某个对称密钥密码体制(例如DES)的密钥，然后使用该密钥加密信息。



DES密钥	ECC密钥	RSA密钥
56	112	420
80	160	768
96	192	1024
128	256	1620

DES、ECC和RSA同等安全强度时密钥长度的比较



3.5 数字签名技术

- n 认证又称为鉴别，是防止主动攻击（如篡改、伪造信息等）的一项重要技术，解决网络数据传输过程中可能出的非法访问与篡改、假冒伪造、拒绝服务、抵赖等安全问题，确保网络中传送数据的机密性、访问可控制性、数据完整性、抗抵赖性等方面的安全需求。
- n 认证的目的包括：消息完整性认证和身份认证。



3.5 数字签名技术

基本概念

- n 数字签名（Digital Signature，又称公钥数字签名、电子签章）是一种使用了公钥加密技术，用于鉴别数字信息的方法。
- n 一套数字签名通常定义为两种互补的运算，一个用于签名，另一个用于验证。



3.5 数字签名技术

- n 通常的方法是数据单元上附加一些数据，或是对数据单元进行密码变换，使得接收者能够确认数据单元的来源和数据单元的完整性并保护数据，防止被人进行伪造。
- n 公钥密码体制和对称密码体制都可以用于数字签名，但目前数字签名主要是基于公钥密码体制，包括普通数字签名和特殊数字签名。



3.5 数字签名技术

- n 普通数字签名算法有RSA、ElGamal、Fiat-Shamir、Guillou-Quisquater、Schnorr、Ong-Schnorr-Shamir数字签名算法、DES/DSA，椭圆曲线数字签名算法和有限自动机数字签名算法等。
- n 特殊数字签名有盲签名、代理签名、群签名、不可否认签名、公平盲签名、门限签名、具有消息恢复功能的签名等，它与具体应用环境密切相关。



3.5 数字签名技术

- n 数字签名技术是不对称加密算法的典型应用，是在网络系统虚拟环境中确认身份的重要技术。数字签名应用中，发送者的公钥可以很方便地得到，私钥则需要严格保密。

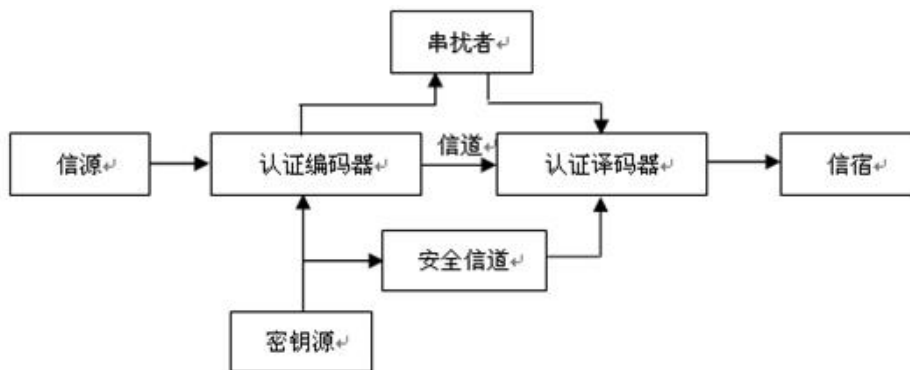


信息认证技术概述

- n 一个安全的认证体制至少应该满足以下要求：
 - 1) 接收者能够检验和证实消息的合法性、真实性和完整性。
 - 2) 消息的发送者对所发的消息不能抵赖，某些场合也要求消息的接收者不能否认收到的消息。
 - 3) 除了合法的消息发送者外，其他人不能伪造发送消息。

信息认证技术概述

- n 认证和保密通常是相对独立的，一个纯认证系统的模型如下：



一个纯认证系统的模型



信息认证技术概述

- n 信息认证是指通过对消息或者消息有关的信息进行加密或签名变换进行的认证，目的是为了防止传输和存储的消息被有意/无意的篡改，包括消息内容认证（即消息完整性认证）、消息的源和宿认证（即身份认证）、以及消息的序号和操作时间认证等。它在票据防伪中具有重要应用。信息认证主要用于防止信息被篡改。



Hash函数与消息认证

1. Hash函数

- n Hash函数是把可变长度的输入串转换成固定长度的输出串的一种函数。
- n Hash函数具备以下性质：
 - 1) Hash函数 H 可适用于任意长度的输入数据块，产生固定长度的Hash值。
 - 2) 对于每一个给定输入数据 M ，都能很容易计算出它的Hash值 $H(M)$ 。

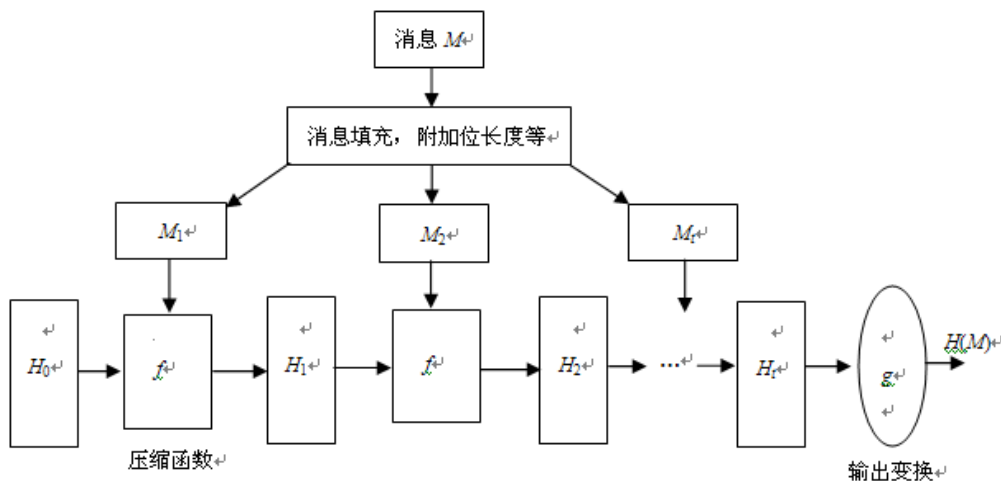


Hash函数与消息认证

- 3) 如果给定Hash值 h ，要逆向推出输入数据 M 在计算上不可行，即Hash函数具备单向性。
- 4) 对于给定的消息 $M1$ 和其Hash值 $H(M1)$ ，找到满足 $M2 \neq M1$ ，且 $H(M2) = H(M1)$ 的 $M2$ 在计算上是不可行的，即抗弱碰撞性。
- 5) 要找到任何满足 $H(M1) = H(M2)$ 且 $M1 \neq M2$ 的消息对 $(M1, M2)$ 在计算上是不可行的，即抗强碰撞性。

Hash函数与消息认证

安全单向Hash函数的一般结构如图：



安全单向 Hash 函数的一般结构



Hash函数与消息认证

2. 消息认证

- n 消息鉴别码也叫密码校验和，是鉴别函数的一种。
- n 其原理是：用公开函数和密钥产生一个固定长度的值作为认证标识，用这个标识鉴别消息的完整性。
- n 消息认证码的安全性取决于两点：
 - ① 采用的加密算法；
 - ② 待加密数据块的生成方法。



Hash函数与消息认证

- n 消息认证不支持可逆性，是多对一的函数，其定义域由任意长的消息组成，而值域则是由远小于消息长度的比特构成。必须要找到一种足够单向和抗强碰撞性的方法，消息认证才是安全的。
 - 1) 利用校验码加密的方式构造认证码，实现数据完整性。
 - 2) 对于用单向Hash函数构造认证码的方式来说，摘要的长度是一个关键的因素。



Hash函数与消息认证

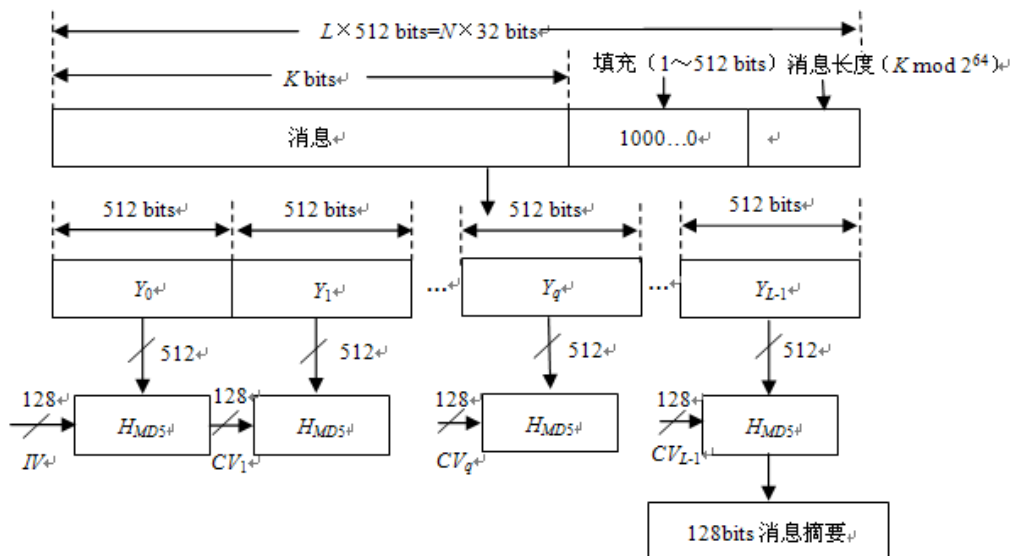
n 常见的单向Hash函数

1. MD5

- n MD5是RSA数据安全公司开发的一种单向Hash算法，MD5被广泛使用，可以用来把不同长度的数据块进行运算处理生成一个128bit的数据块。
- n MD5以512 bit分组来处理输入的信息，且每一分组又被划分为16个32 bit的子分组，经过了一系列的处理后，算法的输出由4个32 bits分组组成，最终将这4个32位分组级联后将生成一个128位Hash值。

Hash函数与消息认证

MD5算法的总体框架如图所示：



MD5 算法的总体框架图



Hash函数与消息认证

- n MD5算法中，首先需要对信息进行填充，使其位长度满足模512等于448。信息的位长度被扩展至 $N \times 512 + 448$ 。填充的方法为在信息的后面填充一个1和无数个0，直到满足上面的条件时才停止用0对信息的填充。再在这个结果后面附加一个以64位二进制表示的填充前信息长度。经过这两步的处理，信息位长度为

$$N \times 512 + 448 + 64 = (N + 1) \times 512。$$



Hash函数与消息认证

n MD5中有四个32位链接变量:

$A=0x01234567$ 、

$B=0x89abcdef$ 、 $C=0xfedcba98$ 、

$D=0x76543210$ 。

设置好链接变量后，进入算法的四轮循环运算。

Hash函数与消息认证

n 第一轮进行16次操作。每次操作对A、B、C和D中的其中3个作一次非线性函数运算，然后将所得结果加上第4个变量，文本的一个子分组和一个常数。再将所得结果向右环移一个不定的数，并加上A、B、C或D中之一。最后用该结果取代A、B、C或D中之一。

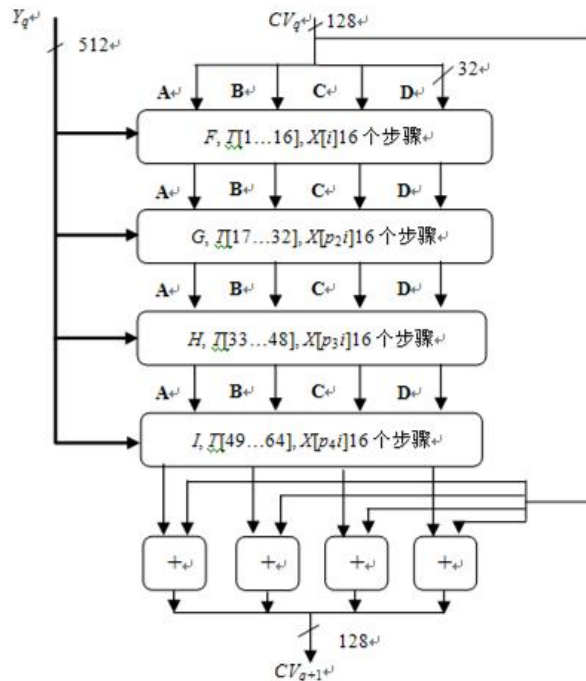
n 使用的四个非线性函数为：

$$F(X, Y, Z) = (X \& Y) | (\overline{X} \& Z)$$

$$G(X, Y, Z) = (X \& Z) | (Y \& \overline{Z})$$

$$H(X, Y, Z) = X \oplus Y \oplus Z$$

$$I(X, Y, Z) = Y \oplus (X | \overline{Z})$$



MD5 对单个 512 位分组的处理过程



Hash函数与消息认证

- n 在MD5算法中，核心是压缩函数HMD5。MD5的压缩函数中有4次循环，每一次循环包含对缓冲区ABCD的16步操作，每一循环的形式为：

$$(a,b,c,d)=(d,b+((a+g(b,c,d)+X[k]+T[i]<<s),b,c)$$



MD5的安全性

- n 2004年8月17日的美国加州圣巴巴拉的国际密码学会议（Crypto'2004）上，来自中国山东大学的王小云教授做了破译MD5、HAVAL-128、MD4和RIPEMD算法的报告，公布了MD系列算法的破解结果。宣告了固若金汤的世界通行密码标准MD5的堡垒轰然倒塌，引发了密码学界的轩然大波。
- n 2009年，冯登国、谢涛二人利用差分攻击，将MD5的碰撞算法复杂度从王小云的 2^{42} 进一步降低到 2^{21} ，极端情况下甚至可以降低至 2^{10} 。仅仅 2^{21} 的复杂度意味着即便是在2008年的计算机上，也只要几秒便可以找到一对碰撞。
- n 注意:并非是真正的破解，只是加速了杂凑冲撞



单向Hash函数

2. SHA-1 Secure Hash Algorithm

- n SHA (Secure Hash Algorithm, 安全散列算法) 是美国国家安全局 (NSA) 设计, 美国国家标准与技术研究院 (NIST) 发布的一系列密码散列函数。正式名称为 SHA 的家族第一个成员 SHA-0 发布于 1993 年。由于存在密码安全性的错误 (1998 年被证实), 两年之后, 发布 SHA-1。另外还有四种变体, 提升输出的范围和变更一些细微设计: SHA-224, SHA-256, SHA-384 和 SHA-512 (这些有时候也被称做 SHA-2)。
- n SHA-0 和 SHA-1 对于长度小于 2^{64} 位的消息, SHA-1 会产生一个 160 位的消息摘要。当接收到消息的时候, 这个消息摘要可以用来验证数据的完整性。
- n SHA-1 的原理类似 MD4 及 MD5 (MIT 教授 Ronald L. Rivest) 算法。主要适用于数字签名标准里面定义的数字签名算法。



单向Hash函数

n SHA-1算法处理步骤如下：

1. 添加填充位。和MD5采用的办法完全一样。
2. 添加长度。一个64位的数据块，表示原始消息的长度。
3. 初始化消息摘要的缓冲区。
4. 以512位数据块作为单位来对消息进行处理。
算法的核心是一个包含四个循环的模块，每个循环由20个处理步骤组成。



Problem of SHA-1

- n Weakness of the SHA-1 - collision attacks
- n Such attacks could allow an attacker to generate additional certificates that have the same digital signature as an original.
- n 2005年8月，王小云、姚期智等在国际密码讨论年会提出SHA-1杂凑函数杂凑冲撞演算法的改良版，使破解SHA-1时间缩短。
- n The use of SHA-1 certificates for specific purposes that require resistance against these attacks has been discouraged.
- n Microsoft - no longer use the SHA-1 hashing algorithm as a default functionality in its software, such as Windows 7, replace with SHA-2 patch, KB3033929, March 2015



单向Hash函数

3. Tiger Hash

- n Tiger Hash结构比MD5和SHA-1更复杂，接近于分组密码。为了适应64位处理器，Tiger选择输出位数是192位。使用了4个S盒，每个S盒将8位映射成64位。还应用了密钥扩展算法，对输入分组进行扩展。



单向Hash函数

4. CRC (循环冗余校验码)

- n CRC由于实现简单，检错能力强，被广泛使用在各种数据校验应用中。占用系统资源少，用软硬件均能实现，是进行数据传输差错检测的一种很好的手段。
- n 生成CRC码的基本原理：任意一个由二进制位串组成的代码都可以和一个系数仅为0和1取值的多项式一一对应。
- n CRC校验码软件生成方法：借助多项式除法，余数为校验字段。



消息认证码MAC算法

- n 常见的消息认证码算法 message authentication code
MAC 实质上是一个将双方共享的密钥 k 和消息 m 作为输入的函数，如果将函数值记为 $MAC_k(m)$ ，这个函数值就是一个认证标记，用 δ 表示。
- n 如果攻击者可以找到一个消息 m ， m 不在 m_1, \dots, m_q 之中，并且能够得到正确的认证标记 $d = MAC_k(m)$ 就说明攻击成功了。攻击者成功的概率就是其攻破 MAC 的概率。



消息认证码MAC算法

- n MAC的构造方法有很多，主要类型是基于带密钥的Hash函数和基于流密码的构造方法。
- n 基于带密钥的 Hash 函数的构造方法最早是由 M.Bellare等人提出的。它要求所使用的 Hash函数具有迭代结构（如 MD5，SHA-1 等），即反复地使用压缩函数 f 将长消息映射为短消息。



消息认证码MAC算法

- n 和同类型的MAC算法相比，HMAC将MAC的安全性归结到所使用Hash函数上。同时具有免费和黑盒的优点。
- n 基于分组密码设计的MAC主要有：CBC-MAC、XOR-MAC。EMAC（加密的 CBC-MAC）、PMAC、XECB-MAC等。



3.5 数字签名技术

常用的数字签名体制

DSS/DSA算法。

1. DSS (Digital Signature Standard)

- n DSS使用的是只提供数字签名的算法，与RSA不同，DSS是一种公钥方法，但不能用于加密或密钥分配。



3.5 数字签名技术

- n DSS方法也是用Hash函数，它产生的Hash值和为此次签名而产生的随机数 k 作为签名函数的输入，签名函数依赖于发送方的私钥和一组通信多方所共有的参数（可以看作全局公钥）。签名由两部分组成，分别记为 s 和 r 。
- n 接收方对接收到的消息产生Hash码，这个Hash码和签名一起作为验证函数的输入，验证函数依赖于全局公钥和发送方公钥，若验证函数的输出等于签名中的 r 成分，则签名是有效的。

3.5 数字签名技术

2. DSA算法

n DSA建立在求离散对数的困难性以及ElGamal和Schnorr最初提出的方法之上。

全局公钥组成

P 为素数, 其中 $2^{L-1} < P < 2^L, 2^9 \leq P \leq 2^{10}$,
且 L 是 64 的倍数

$q(p-1)$ 的素因子, 其中 $2^{159} \leq q \leq 2^{160}$, 即位长为 160 位

$g = h^{(P-1)/q} \bmod P$, 其中 h 是满足 $1 < h < (P-1)$
并且 $h^{(P-1)/q} \bmod P > 1$ 的任何整数

用户的私钥

x 为随机或伪随机整数且 $0 < x < q$

用户的公钥

$y = g^x \bmod P$

与用户每条消息相关的秘密值

$k =$ 随机或伪随机整数且 $0 < k < q$

签名

$r = (g^k \bmod P) \bmod q$

$s = [k^{-1}(H(M) + xr)] \bmod q$

签名 $= (r, s)$

验证

$w = (s')^{-1} \bmod q$

$u_1 = [H(M')^w] \bmod q$

$u_2 = (r')^w \bmod q$

$v = [(g^{u_1} y^{u_2}) \bmod P] \bmod q$

检验: $v = r'$

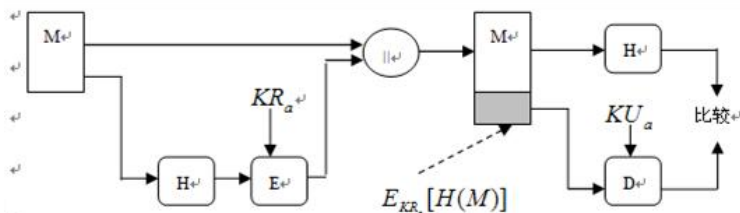
M : 要签名的消息

$H(M)$ 使用 SHA-1 计算的 M 的 Hash 值

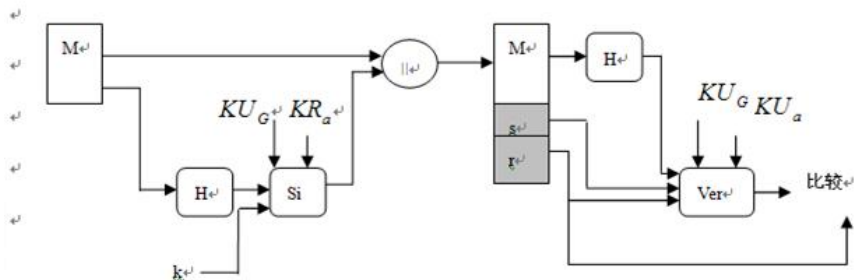
M', r', s' 接收到的 M, r, s

3.5 数字签名技术

RSA与DSS的比较如图所示：



(a) RSA 方法



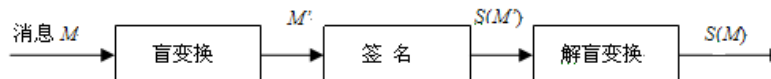
(b) DSS 方法

两种数字签名方法

盲签名和群签名

1. 盲签名

- 所谓盲签名，就是先将隐蔽的文件放进信封里，而除去盲因子的过程就是打开这个信封。
- 盲签名是接收者在不让签名者获取所签署消息具体内容的情况下所采取的一种特殊的数字签名技术。在电子商务和电子选举等领域有着广泛的应用。
- 它必须满足以下两条性质：
 - 1) 签名者对其所签署的消息是不可见的。
 - 2) 签名消息不可追踪。



盲签名过程



盲签名和群签名

n 一个好的盲签名应该具有以下性质：

- 1) 不可伪造性
- 2) 不可抵赖性
- 3) 盲性
- 4) 不可跟踪性



盲签名和群签名

2. 群签名

- n 在一个群签名方案中，一个群体中的任意一个成员可以以匿名的方式代表整个群体对消息进行签名。与其他数字签名一样，群签名是可以公开验证的，而且可以只用单个群公钥来验证。也可以作为群标志来展示群的主要用途、种类等。
- n 群签名在军事、政治及经济等多个方面有着广泛的应用。



盲签名和群签名

n 群签名具有以下几个特点：

- ① 只有群体中的成员能代表群体签名。
- ② 接收到签名的人可以用公钥验证群签名，但是不可知道由群体中哪个成员所签。
- ③ 发生争议时可由群体中的成员或者可信赖机构识别群中的签名者。



盲签名和群签名

- n 群签名有如下几个研究方向：
 - 1) 如何安全有效的废除群成员。
 - 2) 如何设计高效的打开签名的算法。
 - 3) 寻找一些安全高效的新的群签名算法。
 - 4) 如何在电子商务等领域更广泛的使用群签名。
 - 5) 对于群签名相关的数字签名及其应用的研究。



3.6 密钥管理与分配

- n 在加密标准DES和公开密钥加密算法中，由于加密算法的公开，网络安全完全基于密钥的安全保护上，因此密钥的管理非常重要。
- n 对称密钥加密方法的一个致命弱点就是它的密钥管理十分困难，因此它很难在现代电子商务的实践中得到广泛的应用。在这一点上，公开密钥加密方法占有绝对的优势。不过，无论实施哪种方案，密钥管理都是要考虑的问题。当网络扩展得更大、用户增加更多时尤其如此。
- n 密钥分配是密钥管理中最大的问题。密钥必须通过最安全的通路进行分配。例如，可以派非常可靠的信使携带密钥的分配给相互通信的各用户，这种方法称为网络外分配方式。如果网络中通信的用户很多且密钥更换很频繁，则要求采用网络通信进行网络内分配方式，即对密钥自动分配。



3.6 密钥管理与分配

密钥分配

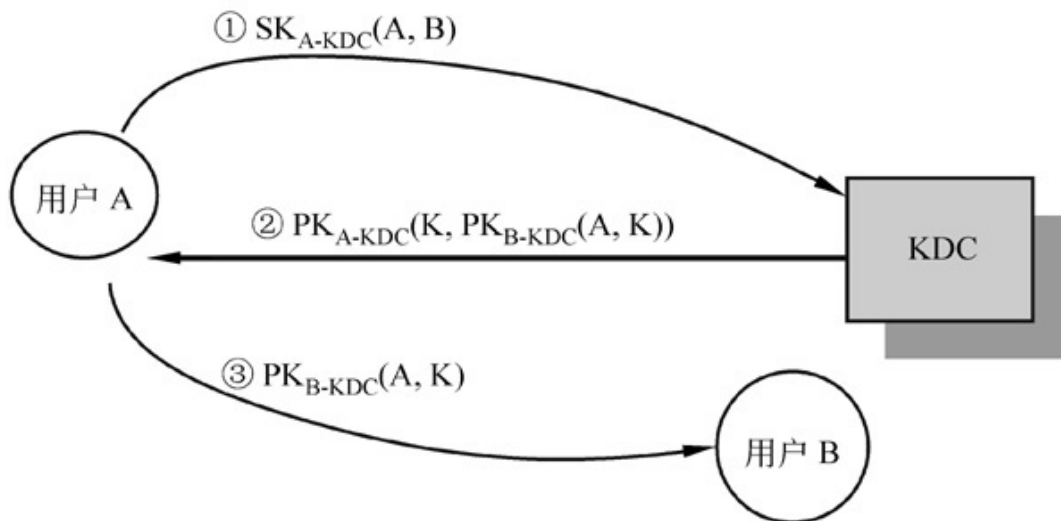
- n 密钥可由A选择并分发给B.
- n 由第三方选择密钥并分发给A和B。
- n 如果A和B以前用过一个密钥，一方可以用这个密钥将新密钥加密后，发送到另一方。
- n 如果A和B都有加密线路到第三方C，C可以通过此加密链路分发密钥给A和B。



3.6 密钥管理与分配

- n 目前，密钥分配公认的有效方法是通过密钥分配中心KDC来管理和分配公开密钥。每个用户只保存自己的私有密钥SK和KDC的公开密钥PK。用户可以通过KDC获得任何其他用户的公开密钥或者某一次通信采用的对称密钥加密算法的临时密钥。
- n 假设有两个用户A和B都是KDC的注册用户，他们拥有与KDC通信的私有密钥SKA-KDC和SKB-KDC，现在A想与B通过对称密钥加密算法通信，要求KDC分配这次通信的临时密钥，则KDC分配密钥的过程如图所示。

对称密钥加密算法的临时密钥



公钥密码体制KDC分配密钥的过程



对称密钥加密算法的临时密钥

- n 首先，A向KDC发送用自己的私有密钥 SK_{A-KDC} 加密的信息报文 $SK_{A-KDC}(A, B)$ ，说明想和用户B通信。KDC根据某个算法随机产生一个密钥K供A和B支持通信使用，然后返回给A一个应答报文 $PK_{A-KDC}(K, PK_{B-KDC}(A, K))$ ，此报文用A的公开密钥 PK_{A-KDC} 加密，报文中有K和请A转给B的报文 $PK_{B-KDC}(A, K)$ ，此报文是用B的公开密钥 PK_{B-KDC} 加密的，当B收到A转来的报文 $PK_{B-KDC}(A, K)$ 后，就知道A要和自己通信且密钥为K。此后，A就可利用密钥K与B通信。
- n 此外KDC可使用其私有密钥 SK_{KDC} 对发给A的应答报文进行数字签名，以防止伪造，还可在报文中加入时间戳防止重放攻击。由于密钥K的使用是一次性的，保密非常高。



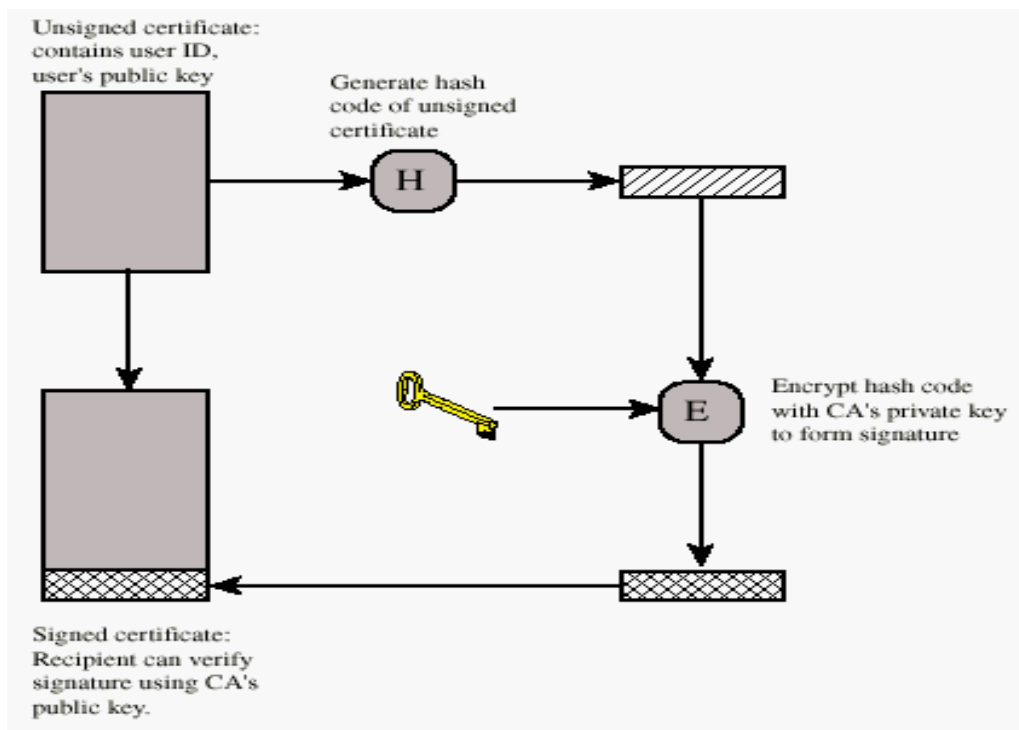
认证中心(certification authority, CA)

- n 在公开密钥体制中，为使各用户更加安全的通信，必须有一个机构把用户的公开密钥与用户的实体(人或计算机)绑定联系起来，这样才能防止用自己的私有密钥签名报文伪造成别的用户，这个机构被称为认证中心(certification authority, CA)。

例如，用户A想欺骗用户B，用户A可以伪造一份是C发送的报文发给B，A用自己的私有密钥签名，并附上A自己的公开密钥，谎称这个公开密钥是C的，B有时很难确定这个公开密钥是否是C的。这就需要值得信赖的机构CA来确定。

- n CA可由政府或信誉良好的组织出资建立，每个实体都有CA颁发的证书(certificate)，里面包含有公开密钥及其拥有者的标示信息(用户名、域名、软件信息等)，此证书由CA进行数字签名。任何用户都可以从可信的地方获得CA的公开密钥，此公开密钥可以用来验证某个用户的公开密钥是否为该实体所拥有。

数字证书的生成





数字证书的生成和验证

CA的数字证书（公钥）已安装到所有计算机中

生成ABC持有的数字证书（由CA签署）：

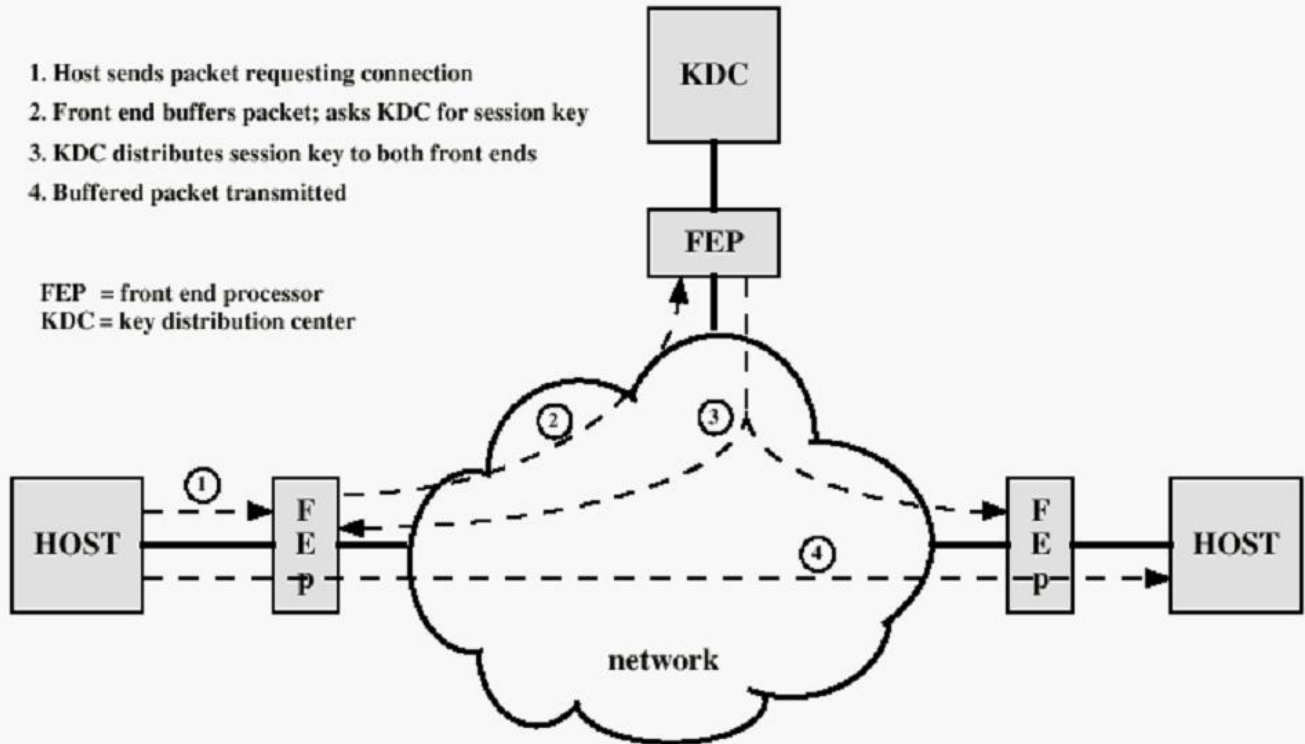
- n 公司ABC在本地生成一对公/私钥
- n ABC发送身份信息和公钥，及用CA的公钥加密身份信息和公钥的HASH值，给CA
- n CA将收到的加密后的HASH值用自己的私钥解密，并与收到的ABC身份信息和公钥的HASH值比较，一致后，核实ABC身份及其公钥（防止冒充ABC及公钥）https
- n CA生成ABC持有的证书，内容为ABC身份信息（例如版本号、主题、日期、加密算法、指纹算法等）和其公钥，及用CA 私钥加密的ABC身份信息和公钥的HASH值（加密的指纹）即数字签名，CA名称等。
- n CA 将数字证书颁发给公司ABC

ABC的客户验证该证书有效：

- n 客户端获得该证书（可通过浏览器https访问ABC网站自动下载，或其他方式如ftp）
- n 客户端提取该证书中的CA，检查本机中已安装的受信任证书颁发机构，找到该CA
- n 找到CA的证书，从该证书中得到CA的公钥
- n 使用CA的公钥解密ABC的证书的签名，得到指纹
- n 客户端根据指纹算法对ABC的证书进行hash计算得到 hash 值
- n 比较该hash值和解密的指纹
- n 二者一致，说明ABC证书确实是CA机构颁发的可信任证书，证书中的内容没被修改，其中的ABC公钥有效。

1. Host sends packet requesting connection
2. Front end buffers packet; asks KDC for session key
3. KDC distributes session key to both front ends
4. Buffered packet transmitted

FEP = front end processor
KDC = key distribution center



Automatic Key Distribution for Connection-Oriented Protocol



3.6 密钥管理与分配

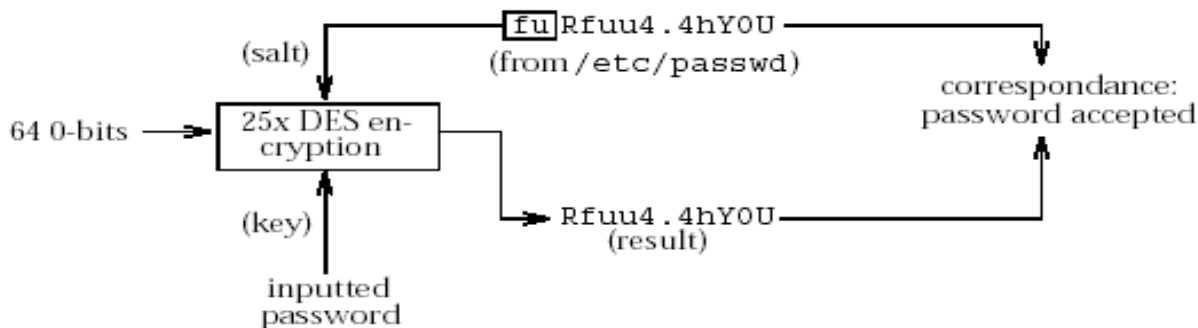
- n 会话密钥 Session key:
 - n 加密一次会话中的数据，会话结束后，密钥销毁
- n 永久密钥 Permanent key:
 - n 为了分发会话密钥而在各通信实体间共享的密钥
- n PKI（Public Key Infrastructure）公钥基础设施，遵循标准的公钥加密技术

常见口令破解

- n UNIX 口令文件 通常位于 /etc/passwd, /etc/shadow
用户名:加密过的口令:uid:gid:info:homedir:shell
2个字节+11个字节
12位 salt 4096种选择 salt 16位或更长的salt值

Char *crypt(char *salt, char *passwd)

64bit0 à DES à DES à out





常见口令破解

- n Windows NT口令文件 SAM数据库
14个字符hash MD4 l0pht crack
- n BIOS口令CMOS
- n 文件系列号等 ZIP PDF DOC
- n CISCO IOS口令
- n 苹果口令，电子辐射破译，防止重放则不太可能破解