计算机网络安全技术

- 课程代号：*40240572*
- 课程对象：本科生
- 授课教师：尹　霞
- 开课单位：计算机系网络所

密码学基础

初识密码

古典密码

- 代换技术
- 置换技术
- 破译举例

对称密码算法

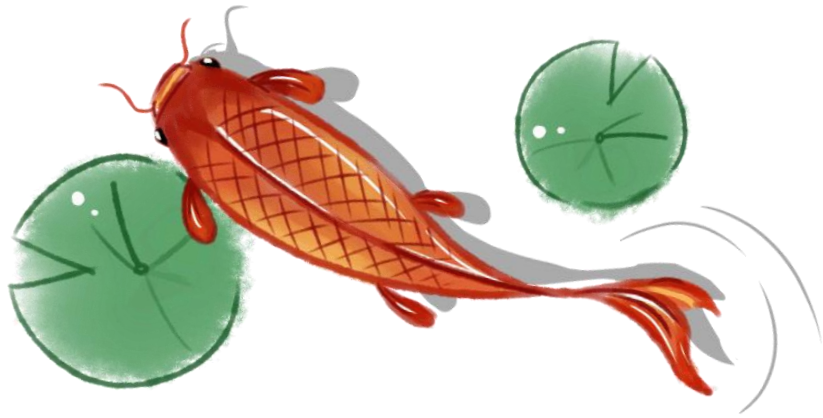
- 简化DES算法
- Feistel密码结构
- DES密码算法
- 常用对称密码

非对称密码算法

- 公钥密码原理
- 数论基础
- RSA算法
- DH密钥交换算法

密钥的分配

非对称密码：算法原理



非对称密码算法

- 对称密钥密码系统的缺陷
  - 密钥必须经过安全的信道分配
  - 无法用于数字签名
  - 密钥管理复杂，密钥的数量：O(n2)
- 1976年，Whitfield Diffie和Martin Hellman提出了非对称密钥密码（一个公钥、一个私钥），也称公钥密码
- 公钥密码和之前四千多年来所使用的所有密码学方法都不同，是密码学中的一个惊人的成就，是密码学历史上唯一的一次真正的革命
- 公钥密码是基于数学函数而不是代换和置换

公钥密码体制

- 公钥密码体制有6个组成部分。
  - 明文：可读的信息，做为加密算法的输入。
  - 加密算法：对明文进行的各种变换。
  - 公钥/私钥：一个用于加密，一个用于解密。
    加密算法执行的变换依赖于公钥和私钥。
  - 密文：加密算法的输出，不可读信息。密文依赖于明文和密钥，不同的密钥产生不同的密文。
  - 解密算法：根据密文和相应的密钥，产生出明文。

符号说明

- 会话密钥Ks
- 用户A的公钥 KUa
- 用户A的私钥 KRa
- $E_{KUa}[P]$：用A的公钥对明文P加密
- $E_{KRa}[P]$：用A的私钥对明文P加密
- $D_{KUa}[C]$：用A的公钥对密文C解密
- $D_{KRa}[C]$：用A的私钥对密文C解密

公钥密码体制

- 每个用户产生一对密钥：用于加密和解密。
  其中一个密钥存于公开的寄存器或者文件中，即公钥；另外一个密钥是私有的，称为私钥
  - 公钥公开，用于加密和验证签名
  - 私钥保密，用作解密和签名

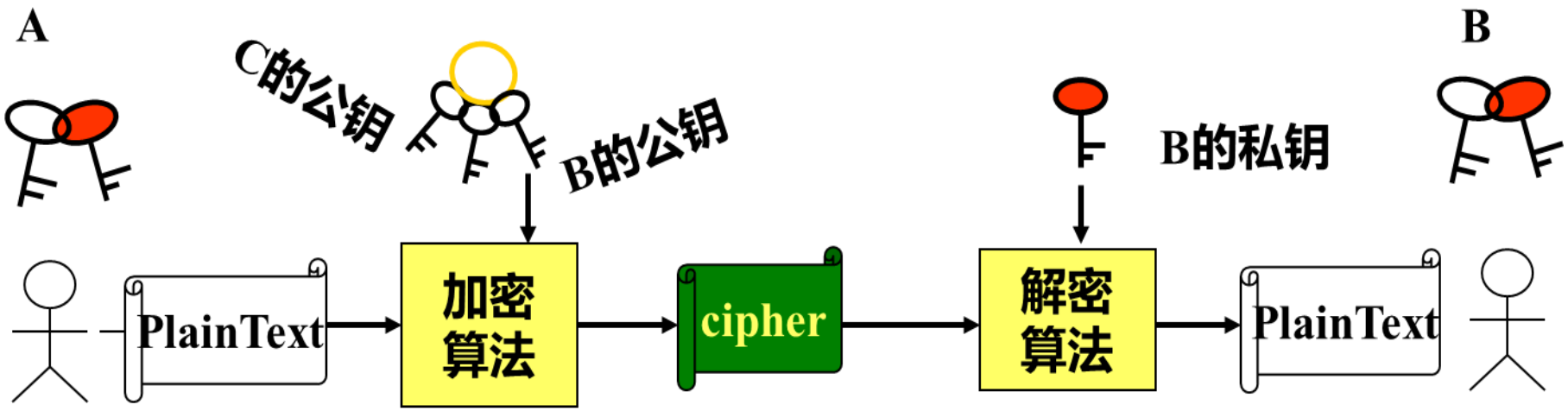
- 例如：
  - Bob给Alice发消息，用Alice的公钥对信息加密
  - Alice收到消息后，用自己的私钥解密
  - 由于只有Alice拥有自己的私钥，所以其他人都不能解密出其信息来

公钥密码体制

- 利用这种方法，通信各方皆可以访问公钥，而私钥是个通信方在本地产生的，所以不必进行分配
- 只要系统控制了私钥，那么它的通信是安全的
- 任何时刻，系统都可以改变自己的私钥，而公布相应的公钥代替原来的公钥

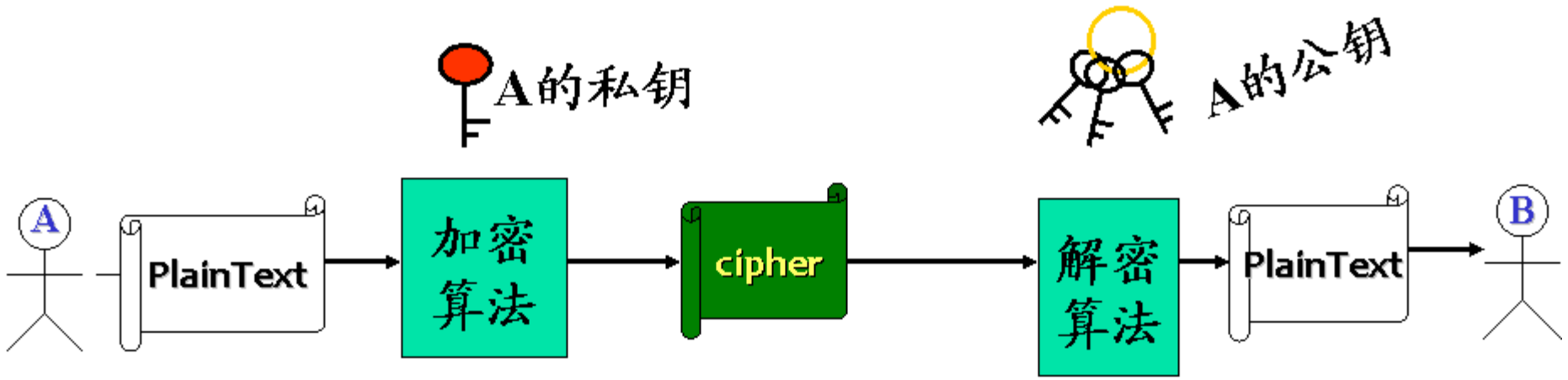
公钥密码系统的加密原理

- 每个通信实体有一对密钥（公钥,私钥）。公钥公开，用于加密和验证签名；私钥保密，用作解密和签名
  - A向B发送消息，用B的公钥加密
  - B收到密文后，用自己的私钥解密
  - 任何人向B发信息都使用同一个密钥(B的公钥)加密。没有人可以得到B的私钥，只有B可以解密



公钥密码系统的签名原理

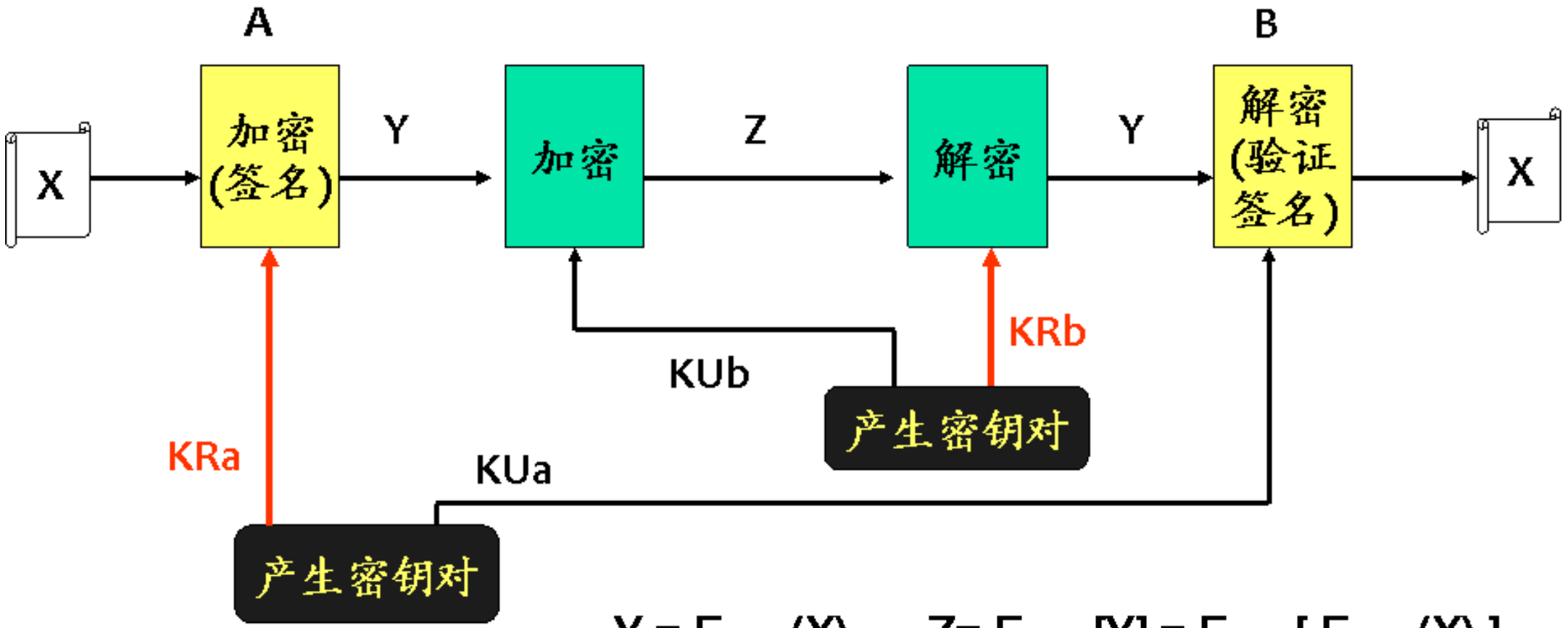
- A向B发送消息，用A的私钥加密(签名)
- B收到密文后，用A的公钥解密(验证)



公钥密码算法的表示

- 对称密钥算法
  - 密钥：会话密钥(Ks)
  - 加密函数：C= E_{Ks}[P]
  - 对密文C，解密函数：D_{Ks}[C]，
- 公开密钥算法
  - A(K_{Ua}，K_{Ra})向B (K_{Ub}，K_{Rb})发送信息：
  - 加密：C= E_{KUb}[P]（用B的公开密钥加密）
  - 解密：P= D_{KRb}[C]
  - 签名：E_{KRa}[P]（用A的私有密钥加密）
  - 验证：D_{KUa}[C]

数字签名和加密同时使用



$$Y = E_{KRa}(X) \quad , \quad Z = E_{KUb}[Y] = E_{KUb}[E_{KRa}(X)]$$

$$Y = D_{KRb}(Z)], \quad X = D_{KUa}[Y] = D_{KUa}[D_{KRb}(Z)]$$

公钥密码的数学原理：陷门单向函数

- 公钥密码系统是基于陷门单向函数的概念
- 单向函数是求逆困难的函数；单向陷门函数，是在不知陷门信息下求逆困难的函数，当知道陷门信息后，求逆是易于实现的
  - 单向陷门函数f(x)，必须满足以下三个条件：

    ①给定x，计算y=f(x)是容易的

    ②给定y, 计算x使y=f(x)是困难的
    （所谓计算$x=f^{-1}(y)$困难是指计算上相当复杂已无实际意义）

    ③存在δ，已知δ时对给定的任何y，若相应的x存在，则计算x使y=f(x)是容易的
- 仅满足①、②两条的为单向函数；第③条为陷门性，δ 称为陷门信息

公钥密码系统的应用

- 公钥密码系统有三种用途：
  - 加密/解密
  - 数字签名：如果电子文件都需要签名，如何能够确保数字签名是出自某个特定人，而且通信双方无异议
    - 发送方用自己的**私钥**签署报文，接收方用对方的公钥验证对方的签名
  - 密钥交换：双方协商会话密钥，用于对称密钥数据加密

公钥密码算法	加密/解密	数字签名	密钥交换
RSA	**Y**	**Y**	**Y**
Diffie-Hellman	**N**	**N**	**Y**
DSA	**N**	**Y**	**N**

公钥密码分析

- 与对称密码相同，公钥密码也容易受到穷举攻击
  - 攻击者可以利用公钥对所以可能的密钥加密，并与所传送的密文对比，从而可以解密任何消息
  - 公钥体制使用的是某种可逆的数学函数，函数值的复杂性可能不是密钥长度的线性增长，而是指数或者更快的增长速度
- 对应穷举攻击，解决方法也是使用长密钥
  - 为了抗击穷举攻击，密钥必须足够长，为了便于实现，密钥又要足够短
- 在实际应用中，公钥密码目前仅局限于密钥管理和数字签名

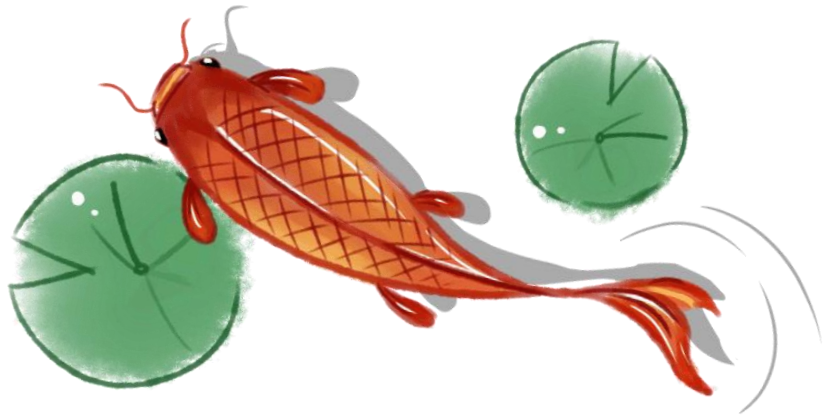
对称密码 vs 非对称密码

- 非对称的公钥密码学，使用两个独立的密钥，这对于密钥分配、数字签名、认证技术等有深远影响；但存在一些误解
- 误解一：从密码分析角度看，公钥密码比传统密码更安全
  - 任何加密方法的安全性依赖于密钥长度和破译密文所需要的计算量
  - 从抗击密码分析的角度看，不能简单地说传统密码和公钥密码那个更安全
- 误解二：公钥密码是一种通用方法，传统密码已经过时
  - 由于公钥密码需要大量计算，仅限于密钥管理和签名这类应用中，所以基本不太可能取代传统密码
- 误解三：传统密码中与密钥分配中心的握手是一件异常麻烦的事情，而公钥密码实现密钥分配则是非常简单的
  - 使用公钥密码也需要某种形式的协议，通常也包括一个中心代理，所包含的处理过程即不比传统密码简单，也不比其更有效

对称密码和公钥密码

	对称密码	公钥密码
一般要求	加密和解密使用相同的密钥	加密和解密使用相同的密码算法，但使用不同的密钥
	收发双发必须共享密钥	发送方拥有加密或解密的密钥，而接收方只拥有另一个密钥。
安全要求	密钥必须是保密的	两个密钥之一必须是保密的
	若没有其它信息，则解密消息是不可能或者至少是不可行的	若没有其他信息，则解密消息是不可能或者至少是不可行的。
	知晓算法和若干密文不足以确定密钥	知道算法和其中一个密钥以及若干密文不足以确定另外一个密钥

非对称密码：数论基础



数论基础：素数

- 素数：整数P>1是素数，当且仅当它只有因子+1和+P。
  - 100以内的素数
  - 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
- 设P是所有素数的集合，则任意正整数可以唯一地表示为：
- a与b的最大公因数：gcd (a, b)
  - gcd(20, 24)=4 , gcd (15, 16)=1
- 如果gcd(a, b)=1， 称a与b 互素

数论基础：模运算

- 模运算 mod
  - a= q n +r 其中 0≤r<n ； q=[a/n] ;
    则有 r = a mod n
    - [x] 表示小于或等于x的最大整数
  - 如果 (a mod n)= (b mod n) ,
    则称a 与b 模n同余,
    记为 $a \equiv b \mod n$
    - 例如， $2^3 \equiv 8 \mod 5$ ， $8 \equiv 1 \mod 7$

数论基础：模运算

- 模运算对加法和乘法是可交换、可结合、可分配的
  - (a+b) mod n = ((a mod n) + (b mod n)) mod n
  - (a-b) mod n = ((a mod n) – (b mod n)) mod n
  - (a×b) mod n = ((a mod n)×b mod n)) mod n
  - (a × (b+c)) mod n = ((a×b) mod n + (a ×c) mod n) mod n
- 幂, 模运算　ma mod n
  - m^2 mod n = (m×m) mod n = (m mod n) ^{2} mod n
  - m^4 mod n = (m^2 mod n) ^{2} mod n
  - m^8 mod n = ((m^2 mod n)^2 mod n)^2 mod n
  - m^{25} mod n = (m × m^8 × m^{16}) mod n

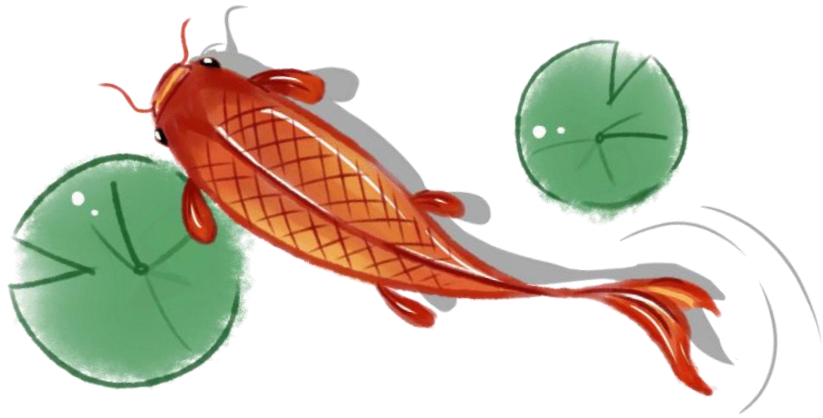
数论基础：欧拉函数

- 欧拉函数φ(n)： n是正整数,φ(n) 是比n小且与n 互素的正整数个数
  - φ(3)=|{1, 2}| =2
  - φ(4)=|{1, 3}| =2
  - φ(5)=|{1, 2, 3, 4 }| =4
  - φ(6)=|{1, 5}| =2
  - φ(7)=|{1, 2, 3, 4, 5, 6}| =6
  - φ(10)=|{1, 3, 7, 9}| =4
- 如果p是素数，则 φ(p)=(p-1)
- 如果p,q 是素数，则 φ(pq)=φ(p) φ(q) = (p-1)(q-1)

数论基础：欧拉定理

- 欧拉定理
  - 若整数m 和n 互素，则 $m^{\phi(n)} \equiv 1 \mod n$
    等价形式 $m^{\phi(n)+1} \equiv m \mod n$
- 例如：
  - m=3，n=10；ϕ(10)=4；$m^{\phi(n)}=3^4=81$；81 mod 10 = 1
  - 即：$81 \equiv 1 \mod 10$ ； $3^{4+1} = 243 \equiv 3 \mod 10$
- 推论：
  - 给定两个素数p, q，p<>q, 两个整数 n, m，使得n=pq, 0<m<n；则对于任意整数k，下列关系成立：$m^{k\phi(n)+1} \equiv m \mod n$

非对称密码：RSA算法



RSA算法简介

- MIT的Ron **R**ivest, Adi **S**hamir，Leonard **A**dleman于1977年提出，于1978年首次发表的RSA算法，是最早满足要求的、被广泛接收、被实现的通用公钥密码算法之一
- RSA体制是一种分组密码，其明文和密文都是0~n-1之间的整数，通常n的大小为1024位二进制数或者309位十进制数

RSA算法：密钥的产生

- 密钥产生
  - 取两个大素数 p，q，保密；
  - 计算n=pq，公开n；
  - 计算欧拉函数φ(n) =(p−1)(q−1)；
  - 任意取一个与φ(n) 互素的小整数e，
    即 gcd (e, φ(n))=1； 1<e< φ(n)
  - 寻找d， d< φ(n)，使得
    de ≡1 mod φ(n)，即de =k φ(n) +1
  - 公开 (e, n)
  - 将d 保密，丢弃p，q。
- 公开密钥:KU={e, n}
- 秘密密钥: KR={d, n}

- 设： p=7,q=17
- 则： n=119
- Φ(n)=6 × 16=96
- 选择e=5
- 5d=k × 96+1
- 令 k=4, 得到 d=77
- 故知道：
  - KU = {5,119}
  - KR = {77,119}

RSA 算法：加密/解密

- 利用： 公钥KU={**e**, n}和私钥KR={**d**, n}
- 加密过程
  - 把待加密的内容分成k比特的分组，k≤ log2n，并写成数字，设为M，则：**$C = M^e \mod n$**
  - 例如：$C=M^5 \mod 119$
- 解密过程
  - **$M = C^d \mod n$**
  - 例如：$M=C^{77} \mod 119$

RSA 算法的证明

- 试证明：解密过程是正确的
- 证明：

$$M = C^d \mod n$$
$$= (M^e \mod n)^d \mod n$$
$$= M^{ed} \mod n$$

即 $M^{ed} \equiv M \mod n$

- 根据欧拉定理推论: $M^{k\phi(n)+1} \equiv M \mod n$，
得到 $ed = k\phi(n)+1$

RSA加密过程举例

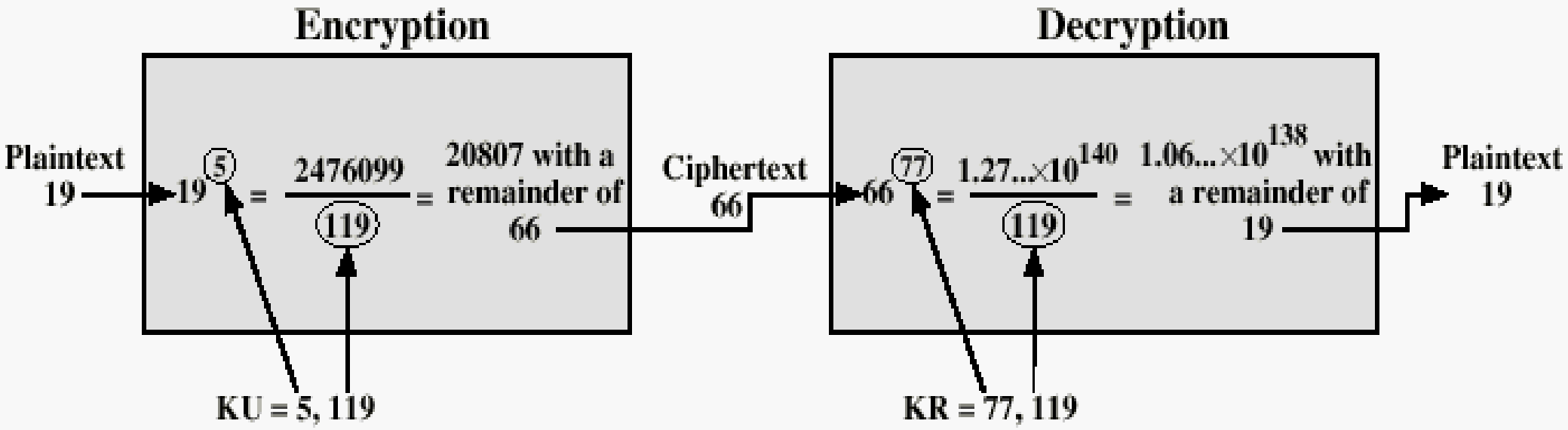


Figure 3.9 Example of RSA Algorithm

RSA算法举例1

- 用RSA算法对下面数据实现加密和解密。
  - p=5 ； q=11 ； e=3 ； M=9

- 密钥产生
  - 取两个大素数 p, q，保密；
  - 计算n=pq，公开n；
  - 计算欧拉函数 φ(n) =(p-1)(q-1)；
  - 任意取一个与φ(n)互素的小整数e，即gcd (e, φ(n))=1; 1<e< φ(n)
  - 寻找d，d< φ(n)，使得 de ≡1 mod φ(n) ，即de =k φ(n) +1
  - 公开 (e,n)
  - 将d 保密，丢弃p, q。
- 公开密钥:KU={e, n}
- 秘密密钥: KR={d, n}

- 设：p=5,q=11
- 则：n=5*11=55
- Φ(n)=(5-1)*(11-1)=4*10=40

- 因为e=3
  根据de =k φ(n) +1
  故得 3d=k × 40+1

- 令 k=2, 得到d=27

- 故知道：
  - KU = {3,55}
  - KR = {27,55}

- 加密过程

$$C=M^e \bmod n$$
$$=9^3 \bmod 55=729 \bmod 55=14$$

- 解密过程

$$M=C^d \bmod n$$
$$=14^{27} \bmod 55=9$$

RSA算法举例2

- 假设Alice需要将明文“key”通过RSA加密后传递给Bob。

- 第一步：设计公钥和密钥
  - 令p=3，q=11，得出n=p×q=3×11=33
  - Φ (n)=(p-1)(q-1)=2×10=20
  - 取e=3，则e×d≡1 mod Φ (n)，即3×d≡1 mod 20；取d=7
  - 从而可以设计出一对公私密钥，
    - 加密密钥（公钥）为：KU =(e,n)=(3,33)
    - 解密密钥（私钥）为：KR =(d,n)=(7,33)

RSA算法举例2

- 第二步：明文信息数字化
  - 假定明文英文字母编码表为按字母顺序排列数值
  - 则得到分组后的key的明文信息为：11，05，25。

字母	A	B	C	D	E	F	G	H	I	J	K	L	M
码值	01	02	03	04	**05**	06	07	08	09	10	**11**	12	13
字母	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
码值	14	15	16	17	18	19	20	21	22	23	24	**25**	26

RSA算法举例2

- 第三步：明文加密
  - Alice用加密密钥(3,33) 将数字化明文分组信息加密成密文
  - 由C≡Me(mod n)得到
    相应的密文信息为：
    11，26，16

$C1 \equiv (M1)^e \pmod{n} = 11^3 \pmod{33} = 11$

$C2 \equiv (M2)^e \pmod{n} = 5^3 \pmod{33} = 26$

$C3 \equiv (M3)^e \pmod{n} = 25^3 \pmod{33} = 16$

- 第四步：密文解密
  - Bob收到密文后，若将其解密，只需要计算 $m_i = C_i^d \bmod n$
    - $M_1=11^7 \pmod{33}=11$； $M_2=26^7 \pmod{33}=05$； $M_3=16^7 \pmod{33}=25$
  - 得到明文信息为：11，05，25；根据编码表将其转换为“key”

RSA 算法的安全性

- 对RSA算法的攻击方法：蛮力攻击、数学攻击、计时攻击
- 蛮力攻击：对所有密钥都进行尝试
- 数学攻击：有多种数学攻击方法，他们的实质是两个素数乘积(n)的因子分解
- 计时攻击：类似于通过观察他人转动保险柜拨号盘的时间长短来猜测密码；攻击者可以通过记录计算机解密消息所用的时间来确定私钥
  - 计时攻击不仅可以用于攻击RSA，还可以用于攻击其它公钥密码系统，由于这种攻击的完全不可预知性以及它仅仅依赖明文，所以计时攻击具有很大的威胁

RSA 算法的安全性

- 大数因子分解是数论中的一个难题，因子分解的进展可用MIPS年描述计算的代价
  - MIPS年是指一台每秒执行百万条指令的处理器运行一年

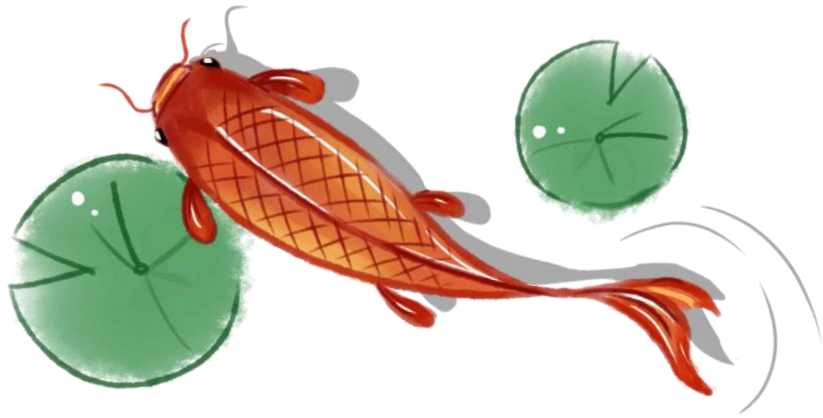
十进制数字位数	近似比特数	完成日期	MIPS年
100	332	1991年4月	7
110	365	1992年4月	75
120	398	1993年6月	830
129	428	1994年4月	5000
130	431	1996年4月	1000
140	465	1999年2月	2000
155	512	1999年8月	8000

RSA 算法的性能

- 与其它密码体制一样，RSA抗穷举攻击的方法也是使用大密钥空间，但是密钥越大，系统运行速度越慢
  - 软件实现比DES慢100倍
  - 硬件实现比DES慢1000倍

	512位	768位	1024位
加密	0.03	0.05	0.08
解密	0.16	0.48	0.93
签名	0.16	0.52	0.97
验证	0.02	0.07	0.08

非对称密码：DH密钥交换算法



Diffie-Hellman密钥交换算法

- 1976年，Diffie-Hellman第一个发表的公开密钥算法，被称为Diffie-Hellman密钥交换算法
- Diffie-Hellman密钥交换算法的目的是使两个用户能够安全地交换密钥，该算法本身也只局限于进行密钥交换
- Diffie-Hellman密钥交换算法的有效性在于计算离散对数非常困难

离散对数

- 对数是指数的反函数： $b= a^i \Rightarrow i= \log_a b$
- 本原根（Primitive Root）
  - a是素数p 的一个本原根，如果
    - a mod p, a^2 mod p , …, a^{p-1} mod p 是1到p-1的排列，即各不相同，是整数1到p-1的一个置换。

p= 19, a^i mod p, i=1,2,3,…,18																	
a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1

离散对数

- 对于整数b（b<p） 和素数p 的一个本原根a，可以找到一个唯一的指数i，使得：
  $b \equiv a^i \bmod p$ ，其中$0 \le i \le (p-1)$

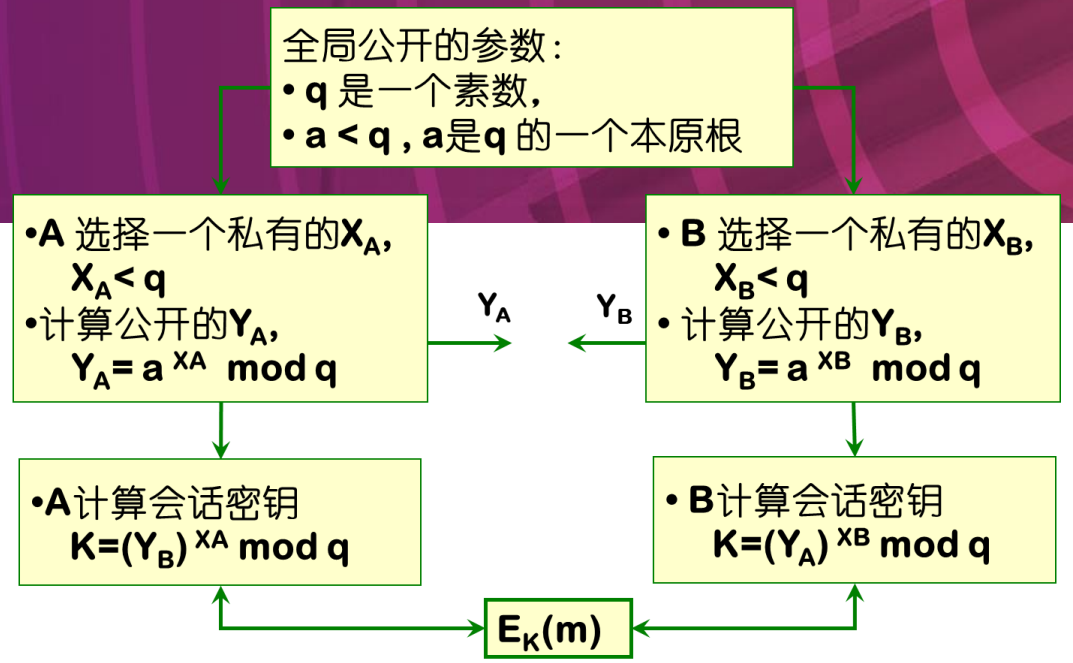
  i 称为 b 的 以a 为底 模p的离散对数或指数，记为 $\text{ind}_{a,p}(b)$
  - $\text{ind}_{a,p}(1)=0$，因为 $a^0 \bmod p = 1 \bmod p = 1$;
  - $\text{ind}_{a,p}(a)=1$，因为 $a^1 \bmod p = a \bmod p = a$;

- 对于$b = a^x \bmod p$
  - 已知a, x, p，计算b 是容易的
  - 已知a, b, p，计算x 是非常困难的

Diffie-Hellman 密钥交换过程

DH算法举例

- 全局公开参数： q=97， a = 5
  (5是97 的本原根)
- A选择私钥 X_A=36
- B 选择私钥 X_B=58



- A 计算公钥 $Y_A = 5^{36} \bmod 97 = 50$
- B 计算公钥 $Y_B = 5^{58} \bmod 97 = 44$
- A 与 B 交换公开密钥

- A 计算会话密钥
  - $K = Y_B^{XA} \bmod q = 44^{36} \bmod 97 = 75$
- B 计算会话密钥
  - $K = Y_A^{XB} \bmod q = 50^{58} \bmod 97 = 75$

其他公钥密码算法

- DSA
  - 1991年, NIST 提出了数字签名算法(DSA),并用于数字签名标准(DSS)
  - DSA只能用于数字签名，算法的安全性是基于计算离散对数的难度
  - 但是DSA招致大量的反对：
    - DSA 不能用于加密或密钥分配
    - DSA是由 NIST研制的，可能有后门
    - DSA的选择过程不公开，提供的分析时间不充分
    - DSA比RSA慢(10—40倍)
    - 密钥长度太小(512位)
    - DSA可能侵犯其他专利

其他公钥密码算法

- 椭圆曲线密码系统
  - 有限域GF(2^n)
  - 运算器容易构造
  - 加密速度快
  - 更小的密钥长度实现同等的安全性

- RSA是事实上的标准

密码学基础

初识密码

古典密码

- 代换技术
- 置换技术
- 破译举例

对称密码算法

- 简化DES算法
- Feistel密码结构
- DES密码算法
- 常用对称密码

非对称密码算法

- 公钥密码原理
- 数论基础
- RSA算法
- DH密钥交换算法

密钥的分配

密码功能的设置

- 对于网络而言，攻击者无处不在
- 加密是所使用的最有力也最常用的方法
- 需要确定的是：加密什么和在什么地方加密
- 存在两种方法：
  - 链路加密
  - 端到端加密

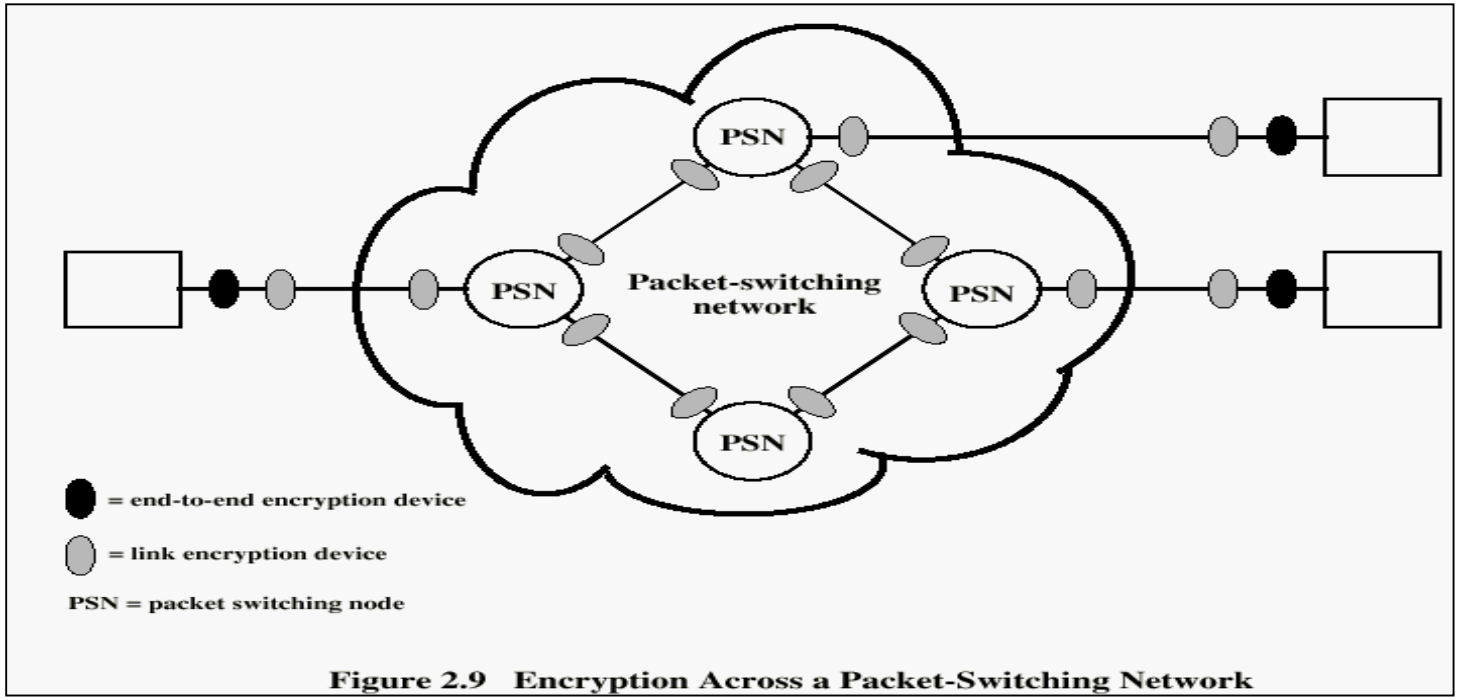


Figure 2.9 Encryption Across a Packet-Switching Network

密钥的分配

- 无论是对称密码还是非对称密码，任何密码系统的强度都和密钥分配方法有关
- 密钥分配方法就是将密钥发放给希望交换数据的双方而不让别人知道的方法
  - 情况一：传统的对称密码分配
  - 情况二：非对称密码中的公钥分配
  - 情况三：公钥密码用于传统密码体制的密钥分配

密钥分配情况一：
传统的对称密码分配

传统的对称密码分配

- 对参与者A和B，传统的对称密钥分配有：
  - 方法一：密钥由A选择，并亲自交给B
  - 方法二：第三方选择密钥后亲自交给A和B
  - 方法三：如果A和B以前或者最近使用过某个密钥，其中一方可用它加密一个新密钥后再发送给另外一方
  - 方法四：A和B与第三方C均有秘密渠道，则C可以将一密钥分别秘密发送给A和B
- 方法一和二需要人工传送密钥，适用于链路加密；
  对于端到端加密，则使用密钥分配中心

密钥分配中心KDC模式

- 假定每个用户与密钥分配中心KDC共享唯一的一个主密钥
  - 设A要和B建立一个逻辑连接
  - A有一个除了自己只有KDC知道的主密钥Ka
  - B有一个除了自己只有KDC知道的主密钥Kb

密钥分配中心KDC模式

- A和B获得一次性会话密钥Ks的工作过程
  - STEP1：A向KDC请求一个会话密钥以保护与B的逻辑连接；消息中有A和B的标识以及一个临时交互号N1
  - STEP 2：KDC用Ka加密的消息做出响应；此时，A可知：
    - 一次性会话密钥Ks，用于会话
    - 原始请求消息，含N1，使A可以做出反映
    - 此外，还含有两项给B的内容，用Kb加密了
      - 一次性会话密钥Ks 和 A的标识符IDa
  - STEP 3： A存下会话密钥Ks备用，将KDC消息中的后两项内容发给B，B也知道会话密钥了
- 这样会话密钥就安全地发给了A和B

面向连接协议的自动密钥分配

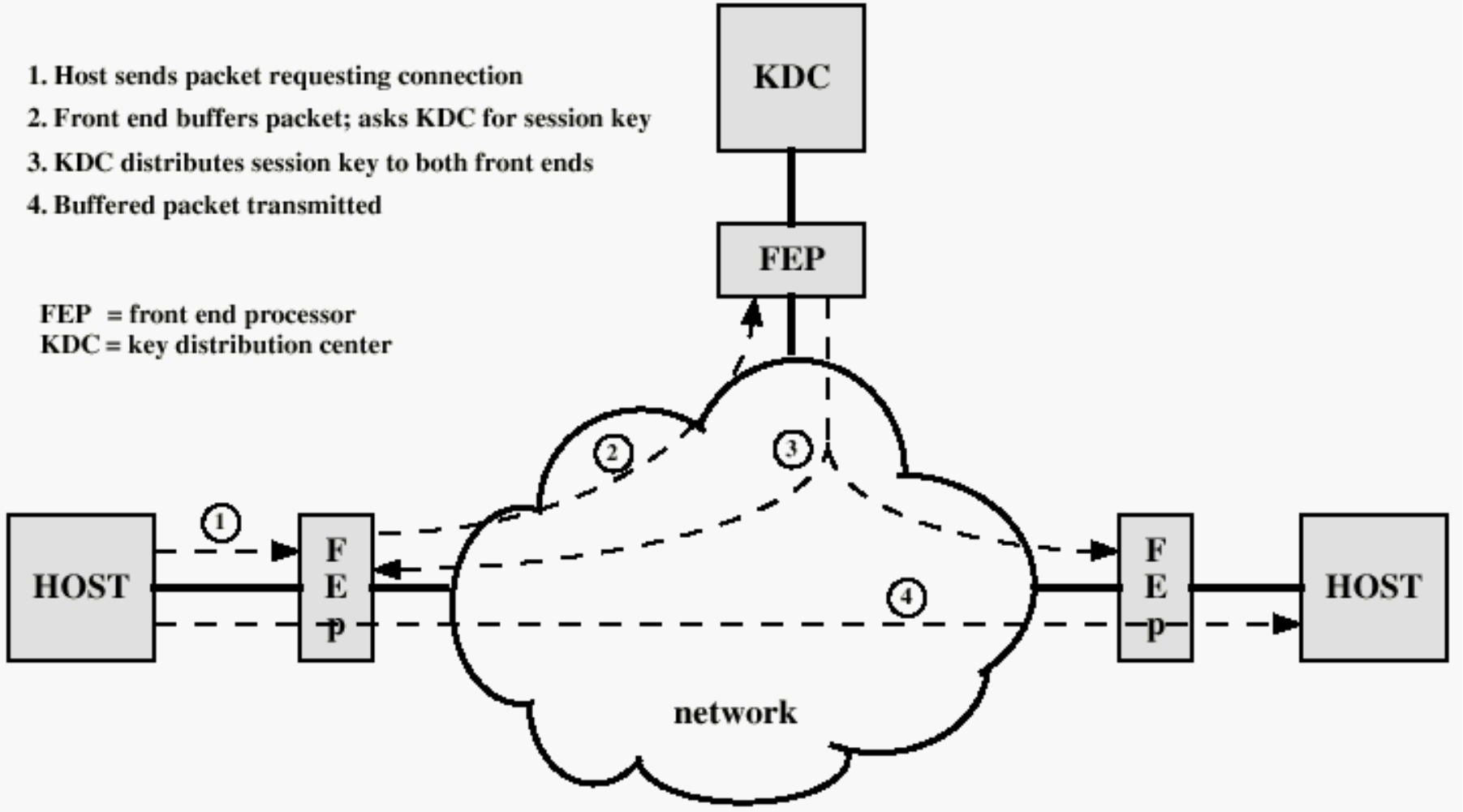


Figure 2.10 Automatic Key Distribution for Connection-Oriented Protocol

密钥的控制

- 在网络规模很大的时候，密钥的分配功能不限定在单个的KDC上面，而是使用层次式的KDC

- 层次式KDC密钥分配使得主密钥分配的代价变小了。而且，如果一个本地KDC出错，或者被攻击了，破坏只是集中在一个区域中，不会影响全局

密钥分配情况二：公钥的分配

公钥的分配

- 公钥密码的主要作用之一就是解决密钥分配问题，公钥密码可用于下面两个方面：
  - 公钥的分配
  - 公钥密码用于传统密码体制的密钥分配
- 常用的公钥分配方法有四种：
  - 公开发布
  - 公开可访问目录
  - 公钥授权
  - 公钥证书

方法一：公开发布

- 任一通信方可以将他的公钥发送给另外一个通讯方或者广播给通信各方
  - 例如：目前在电子邮件的认证和保密性方面广泛应用的PGP(pretty good privacy)协议，很多PGP用户就在自己发送的信息之后加入自己的公钥

- 这种方法比较简便，但是它有一个大缺点，就是任何人都可以伪造这种公钥的公开发布

方法二：公开可访问目录

- 维护一个动态可访问的公钥目录可以获得更大程度的安全性；某个可信的实体或组织负责这个公开目录的维护和分配
  - 管理员通过对每个通讯方建立一个目录项<姓名，公钥>来维护目录，管理员定期发布或者更新该目录
  - 每个通讯方通过目录管理员注册一个公钥；通讯方在任何时候都可以用新的密钥替代当前密钥
  - 通讯方也可以访问电子目录；当然，它必须拥有从管理员到他的安全认证通道
- 这种方法比公开公钥要安全，但是也存在缺点：一旦攻击者获得或者计算出目录管理员的私钥，带来的危险将很大

方法三：公钥授权

- 公钥授权是一种更严格的方法，它的工作过程：
  - A发送一条带有时间戳的消息给公钥管理员，请求B的当前公钥
  - 管理员给A发送一条用其私钥KR加密的消息，A可以用管理员的公钥解密。这条消息中包含：B的公钥、原始请求、原始时间戳
  - A保存B公钥，并将A的表示和临时交互号N1发给B
  - 与A一样，B使用同样的方法从管理员那里得到A的公钥。
  - A与B已经可以通信了，再通过核对临时交互号确认各自的身份后，安全的通信机制就建立了
- 公钥授权的缺点在于公钥管理员就会成为系统的瓶颈
  - 只要用户之间通信，就必须向目录管理员申请对方的公钥

方法四：公钥证书

- Kohnfelder最早提出不通过管理员，而用整数来交换密钥；此方法与公钥授权的安全性相同
  - 证书包含者公钥和其它一些信息，由证书管理员产生，并发给拥有相应私钥的通讯方
  - 通信一方通过传递证书将密钥信息传递给另外一方，其它通信各方可以验证该证书确实是由证书管理员发出的
- 这种方法需要满足下面的要求：
  - 任何通信方可以读取证书并确定证书拥有者的姓名和公钥
  - 任何通信方可以验证该证书出自证书管理员，而不是伪造的
  - 只有证书管理员才可以产生并更新证书
  - 任何通信方可以验证证书的当前性

密钥分配情况三：
利用公钥分配传统密码的密钥

利用公钥分配传统密码的密钥

- 由于公钥密码速度较慢，几乎没有用户愿意在通信中完全使用公钥密码，因此公钥密码更适合作为传统密码中实现密钥分配的一种手段。

- 下面介绍三种密钥分配方法
  - 简单的密钥分配方法
  - 具有保密性和真实性的密钥分配方法
  - 混合方法

简单的密钥分配

- Merkle提出了一种极其简单的方法
- 例如：A要和B通信，则执行：
  - A产生公/私钥对{KUa,KRa}，并将含有KUa和A表示的消息发给B。
  - B产生秘密钥Ks，用A的公钥对Ks加密后传给A。
  - A计算DKRa[EKUa[Ks]]得到秘密钥Ks。
  - 这样A和B就可以利用Ks和对称密码进行安全通信。
- 不过，这种协议容易受到主动攻击

具有保密性和真实性的密钥分配

- 下面这种方法即可以抗击主动攻击也可以抗击被动攻击
  - 假设A和B已经交换了公钥
  - A用B的公钥对含有A标识和临时交互号N1的消息加密，并发给B
  - B发送一条用A的公钥加密的消息，包括A的N1和B的临时交互号N2
    - 因为只有B能够解密A发来的消息，所有本条消息中的N1可以使A确信其通信伙伴是B
  - A用B的公钥对N2加密，并返回B，这样可使B确信其通信伙伴是A

具有保密性和真实性的密钥分配

- A选择密钥Ks，并将M=EKUb[EKRa[Ks]]发送给B
- B计算DKUa[DKRb[M]]得到密钥Ks
- A和B可以利用对称密码进行安全通信了

- 利用公钥密码进行密钥分配，也需要密钥分配中心KDC，KDC与每个用户共享一个主密钥，通过该主密钥加密实现会话密钥的分配


Thanks a lot !

Activity is the only road to knowledge!

Computer Network Security @ 2022Fall