北京郵電大學
BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

《现代密码学》第七章

公钥加密体制

信息安全中心

上讲内容回顾

- Hash函数的定义及安全目标
- Hash函数的发展现状
- Hash函数的构造
- 消息鉴别码的定义及安全目标
- 消息鉴别码的发展现状
- 消息鉴别码的构造
- 认证加密模式

本章主要内容

- 对称密码体制面临的问题
- 公钥密码体制的发展
- RSA单向陷门函数及其应用
- ElGmal单向陷门函数
- 椭圆曲线单向陷门函数简介
- DL/ECIES(IEEE 1363a-2004)

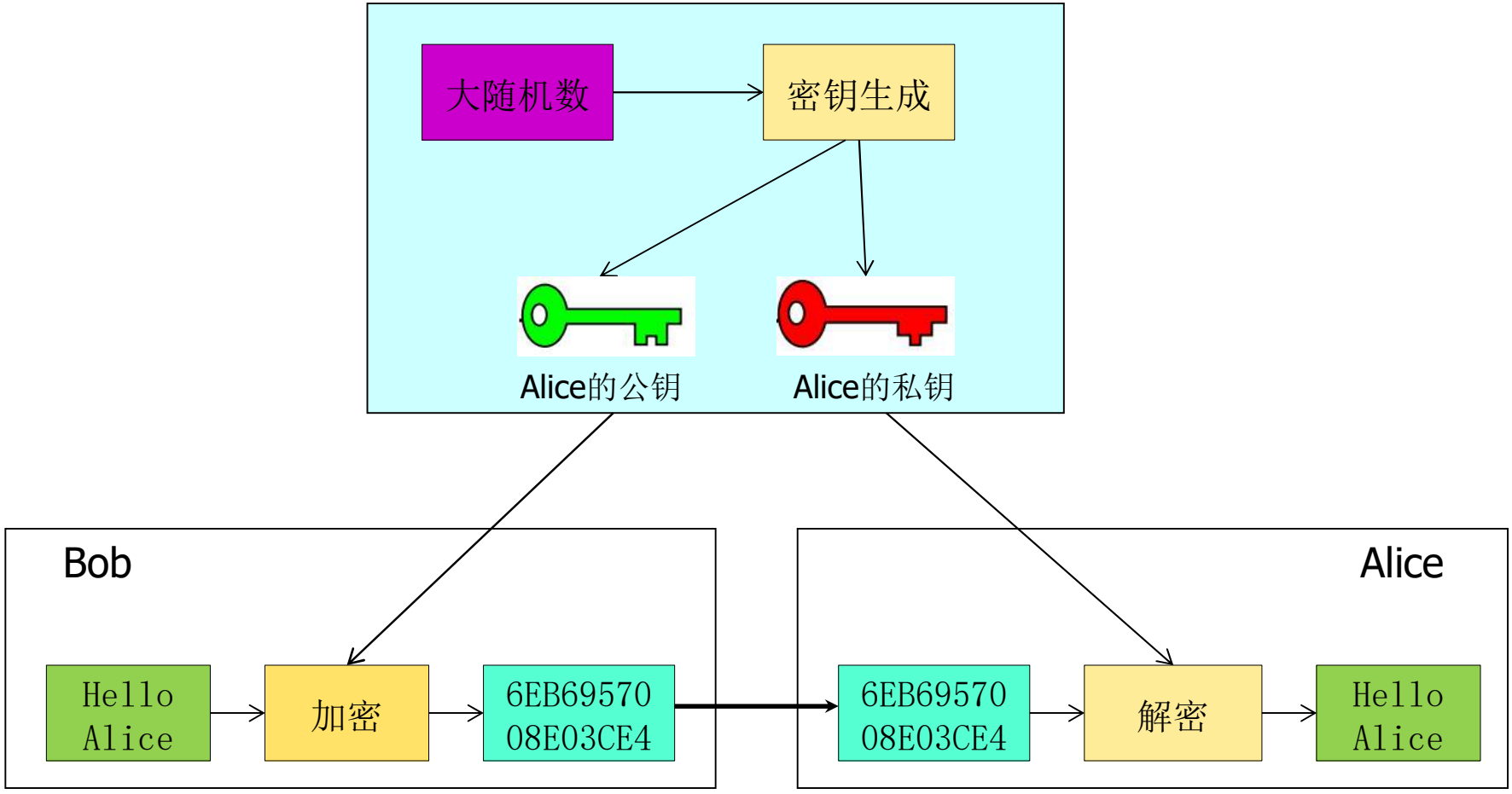
本章主要内容

- 对称密码体制面临的问题
- 公钥密码体制的发展
- RSA单向陷门函数及其应用
- ElGmal单向陷门函数
- 椭圆曲线单向陷门函数简介

对称密码体制面临问题

- 初始密钥分配：对称密码体制，发送方指定一个（种子）密钥后，必须得想方设法把密钥告知接收方，怎样确保“告知过程”密钥不泄露？
- 不可抵赖性：当主体A收到主体B的电子文挡（电子数据）时，无法向第三方证明此电子文档确实来源于B。
- 密钥管理：在有n个用户的网络中，若需要两两用户安全通信，则每对用户需要共享独立的秘密密钥，网络中需要管理的密钥总数是 $n*(n-1)/2$

公钥加密模型


大随机数
密钥生成
Alice的公钥
Alice的私钥
Bob
Hello
Alice
加密
6EB69570
08E03CE4
Alice
6EB69570
08E03CE4
解密
Hello
Alice

公钥加密模型

- 密钥生成过程：接收消息的端系统（如图中的接收者Alice）产生一对密钥(PK_A, SK_A)，PK_A是公开钥（用于加密），SK_A是秘密密钥（用于解密）.

- 加密过程：Bob想向Alice发送消息m，则获取Alice的公开密钥PK_A，加密得密文$c = E_{PK_A}[m]$，其中E是加密算法.

- 解密过程：Alice收到密文c后，用自己的秘密钥SK_A解密，表示为$m = D_{SK_A}[c]$，其中D是解密算法.

单向陷门函数

定义:一个定义在（X,Y）空间上的单向陷门函数(One-way Trapdoor functions, OTF)包含3个“有效”算法（G，F，F^{-1}）

- G(·)：随机算法，输出对(pk, sk)
- F(pk, ·):确定算法，定义X ⟶ Y的运算
- F^{-1}(sk, ·)：F(pk, ·)的逆函数，定义Y ⟶ X的运算

单向陷门函数

对于任意生成的一对密钥$G \rightarrow (pk, sk)$,单向陷门函数满足:

$$\forall x \in X:\ F^{-1}(sk,\ F(pk, x)) = x$$

定义:(G, F, F^{-1}) 是一个安全的的OTF当对于所有有效的算法A:

$$Adv_{OW}[A,F] = \mathbf{Pr[\ A[F(pk, x)]=x]} < \text{可忽略}$$

本章主要内容

- 对称密码体制面临的问题
- 公钥密码体制的发展
- RSA单向陷门函数及其应用
- ElGmal单向陷门函数
- 椭圆曲线单向陷门函数简介

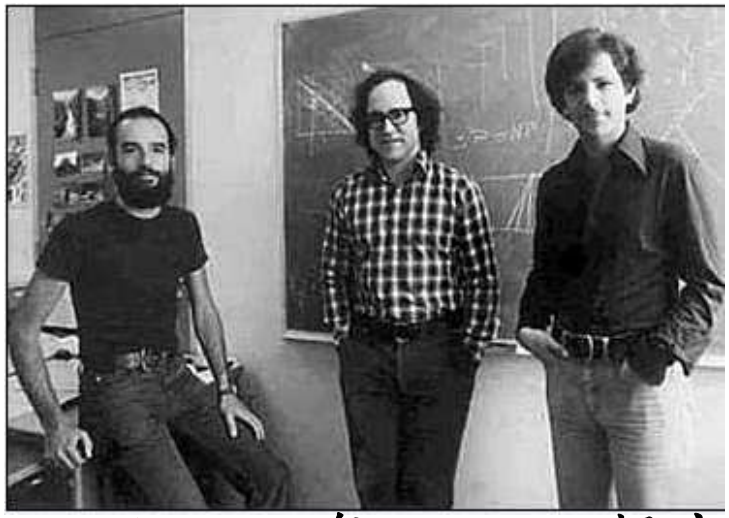
公钥密码体制的发展

- 1976年 Diffie和Hellman 在《密码学的新方向》中首次公开提出了非对称密码算法的思想，但是没有实现加密方案，只给出一个密钥协商协议；

Diffie和Hellman

公钥密码体制的发展

Adi Shamir, Ronald Rivest, and Len Adleman



- 1978年 Rivest,Shamir和Adleman提出应用广泛的RSA算法；
- 1984年 Shamir提出基于身份的密码体制，没有实现加密体制，只给出一个基于身份的数字签名算法；
- 2001年 Boneh,Franklin和Cocks分别独立提出基于身份的加密算法
- 2003年 Al-Riyami提出的无证书的密码体制。

公钥密码体制的发展

后量子密码体制

- 2016年12月，NIST正式面向全球征集具备抗击量子计算机攻击能力的新一代PQC算法。
- 2017年12月，NIST公布PQC标准协议的第一轮预选结果，期间共收到82个基础方案，筛选收录63个完整方案，其中包括44个非对称加密和KEMs方案及19个数字签名方案。

类型	非对称加密和KEMs	数字签名	总计
格密码	21	5	26
编码密码	16	2	18
多变量密码	2	7	9
哈希密码	0	3	3
其它	5	2	7
总计	44	19	63

公钥密码体制的发展

后量子密码体制

- 2019年1月，NIST公布PQC标准协议的第二轮预选结果，共计26个算法进入下一轮进程，其中包括17个非对称加密和KEMs方案
- 2021年1月，NIST公布的第三轮审查共有7个算法入围，其中包括4种非对称加密和KEMs（Classic McEliece、CRYSTALS-KYBER、NTRU、SABER）及3种数字签名算法（CRYSTALS-DILITHIUM、FALCON、Rainbow）。此外，NIST还保留了8个备选算法，包括5种备选公钥加密和密钥生成算法（BIKE、FrodoKEM、HQC、NTRU Prime、SIKE）和3种数字签名算法（GeMSS、Picnic、SPHINCS+）。

公钥密码体制的发展

后量子密码体制

- 2022年7月5日，NIST公布提前选中并将进行标准化的算法，其中包括用于非对称加密和KEMs的CRYSTALS-KYBER、用于数字签名的CRYSTALS-Dilithium、FALCON及SPHINCS+。以上四种算法均在2024年之前支持NIST未来的加密标准。
- 此外，NIST推荐将BIKE、Classic McEliece、HQC、SIKE算法进入第四轮筛选进程。

本章主要内容

- 对称密码体制面临的问题
- 公钥密码体制的发展
- **RSA单向陷门函数及其应用**
- ElGmal单向陷门函数
- 椭圆曲线单向陷门函数简介

RSA单向陷门函数及其应用

RSA算法是1978年由R.Rivest, A.Shamir和L.Adleman 提出的一种用数论难题构造的公钥密码体制。

R L Rivest, A Shamir, L Adleman, "On Digital Signatures and Public Key Cryptosystems", Communications of the ACM, vol 21 no 2, pp120-126, Feb 1978

RSA单向陷门函数及其应用

1） 密钥的产生

① 选两个安全的大素数p和q。

② 计算$n=p\times q$，$\varphi(n)=(p-1)(q-1)$，其中$\varphi(n)$是n的欧拉函数值。

③ 选一整数e，满足$1<e<\varphi(n)$，且$\gcd(\varphi(n),e)=1$。

④ 计算d，满足$d\cdot e\equiv 1 \mod \varphi(n)$，即$d$是$e$在模$\varphi(n)$下的乘法逆元。因$e$与$\varphi(n)$互素，故它的乘法逆元存在且唯一。

⑤ $\{e,n\}$为公开密钥，$\{d,n\}$为秘密密钥。

RSA单向陷门函数及其应用

2） 加密

加密时首先将明文分组，使得每个分组m值小于n，即分组长度小于$\log_2 n$。然后对每个明文分组m，作加密运算：

$$c \equiv m^e \bmod n$$

3） 解密

对每个密文分组的解密运算：$m \equiv c^d \bmod n$

RSA单向陷门函数及其应用

4）正确性

证明： 若m与n互素，由加密过程知$c \equiv m^e \bmod n$，所以

$$c^d \bmod n \equiv m^{ed} \bmod n \equiv m^{k\varphi(n)+1} \bmod n$$

由Euler定理知 $m^{\varphi(n)} \equiv 1 \bmod n$,

所以 $m^{k\varphi(n)} \equiv 1 \bmod n$,

进而 $m^{k\varphi(n)+1} \equiv m \bmod n$,

即 $c^d \bmod n \equiv m$.

RSA单向陷门函数及其应用

5）实例

- ① 密钥产生

Bob选p=7，q=17. 求$n=p\times q$=119，$\varphi(n)=(p-1)(q-1)$=96.
取e=5，满足$1<e<\varphi(n)$，且$\gcd(\varphi(n),e)=1$，
因为$77\times5=385=4\times96+1$，所以满足$d\cdot e\equiv1 \bmod 96$的$d$为77，
因此Bob的公开密钥为{5，119}，秘密密钥为{77，119}.

- ② 加密

设Alice加密明文$m=19$给Bob，则加密过程为
$c\equiv19^5 \bmod 119\equiv2476099 \bmod 119\equiv66$;

- ③ 解密

Bob用私钥解密密文66，过程为 $m\equiv66^{77}\bmod 119\equiv19$.

RSA单向陷门函数及其应用

6）安全性

整数分解问题：已知n是两个大素数的乘积，求n的素分解；

如果RSA加密算法的模数n被成功地分解为$p \times q$，则获得$\varphi(n)=(p-1)(q-1)$，从而攻击者能够从公钥e解出私钥d，即$d \equiv e^{-1} \bmod \varphi(n)$，完全破译.

RSA问题：已知c求明文m，即求c的e次方根？

RSA单向陷门函数及其应用

- 至今还未能证明分解大整数就是NPC问题，也许有尚未发现的多项式时间分解算法.
- 随着人类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.

例如RSA-129（即n为129位十进制数，大约428个比特）已在网络上通过分布式计算历时8个月于1994年4月被成功分解，RSA-130 已于1996年4月被成功分解.

RSA-768 has 232 decimal digits and was factored on December 12, 2009 by Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Pierrick Gaudry, Alexander Kruppa, Peter Montgomery, Joppe W. Bos, Dag Arne Osvik, Herman te Riele, Andrey Timofeev, and Paul Zimmermann

RSA单向陷门函数及其应用

- 分解算法的进一步改进

过去分解算法都采用二次筛法，如对RSA-129的分解. 而对RSA-130的分解则采用了一个新算法，称为推广的数域筛法，该算法在分解RSA-130时所做的计算仅比分解RSA-129多10%.

综上所述，*在使用RSA算法时对其密钥的选取要特别注意其大小。估计在未来一段比较长的时期，密钥长度介于1024比特至2048比特之间的RSA是安全的.*

RSA单向陷门函数及其应用

7）安全参数

- **|p-q|**要大

因为 $\frac{(p+q)^2}{4}-n=\frac{(p+q)^2}{4}-pq=\frac{(p-q)^2}{4}$,

如果$|p-q|$小，则 $(p-q)^2/4$ 也小；因此$(p+q)^2/4$ 稍大于n，即$(p+q)/2$稍大于$n^{1/2}$。那么

① 顺序检查大于$n^{1/2}$的每一整数x，直到找到一个x使得x^2-n是某一整数（记为y）的平方。

② 由$x^2-n=y^2$，得$n=(x+y)(x-y)$，可得n的分解.

RSA单向陷门函数及其应用

- 不选取太小的加密指数

若选取较小的e指数(e. g.，$e=3$)，

如果要加密的明文$m<n^{1/e}$，则加密后的密文$c<n$；

攻击者直接求$c^{1/e}$(密文c的e次方根)，就可恢复真正的明文m。

RSA单向陷门函数及其应用

- 不同用户需使用不同的模数

RSA的共模攻击：

若系统中用户共用一个模数n，而拥有不同的e和d;

若存在同一明文（设为m）分别用不同的公钥（$e1$和$e2$）加密，

$$c1 = m^{e1} \bmod n \quad ; \quad c2 = m^{e2} \bmod n$$

设攻击者截获n、$e1$、$e2$、$c1$和$c2$，　$\gcd(e1,e2)=1$，则他可以恢复m。

RSA单向陷门函数及其应用

算法：因为$e1$和$e2$互质，故用Euclidean算法能找到r和s，满足：

$$r * e1 + s * e2 = 1$$

则

$$(c1)^r * (c2)^s = (m^{e1})^r * (m^{e2})^s \bmod n$$

$$= m^{r * e1 + s * e2} \bmod n = m \bmod n$$

RSA单向陷门函数及其应用

- Th. 设s/r是一个满足$|\frac{s}{r}-\varphi| \le \frac{1}{2r^2}$|的有理数，则$s/r$是的连分式$\varphi$的一个渐进值，且可以用连分式算法在$O(l^3)$个运算之内计算，其中$l$是$\varphi$的长度.
- （Wiener's attack）：

$$d \le N^{0.25}/3 \quad \Rightarrow \left|\frac{k}{d}-\frac{e}{N}\right| \le \frac{1}{2d^2}$$

其中，$e \cdot d = k \cdot \varphi(N) + 1$

- 对e/N使用连分式算法，可以得出 k/d.

  又因为$e \cdot d = 1 \pmod{k}$ $\Rightarrow$ $\gcd(d,k)=1$ $\Rightarrow$ 从k/d可以推导出d.

RSA单向陷门函数及其应用

- 已知N =5025043, $e = 717223$
  - ① $\frac{e}{N} = \frac{717223}{5025043} = \frac{1}{\frac{5025043}{717223}} = \frac{1}{7+\frac{4482}{717223}}$
  - ② 令$\frac{k}{d} = \frac{1}{7}$，又因为gcd(1, 7)=1，故假设$d = 7$, $k = 1$;
  - ③ $\varphi(N) = e \cdot d - 1$ =5020561-1=5020560=$N - p - q + 1$;
  - ④ 故而由$p + q = 4484$, $p \cdot q = 5025043$解得 $p = 2203$, $q = 2281$;
  - ⑤ 验证可知$p \cdot q = 5025043$.

RSA单向陷门函数及其应用

7）为保证RSA算法足够安全，参数须满足下面几个基本要求：

- 需要选择足够大的素数p，q；
- $|p-q|$较大；
- e对所有用户可以是相同的，建议使用$e=2^{16}-1=65535$;
- 解密指数比较大，$d>n^{1/2}$?
- 不同用户不共用模数n.
- $p-1$和$q-1$有大的素因子；
- 。。。

RSA单向陷门函数及其应用

8）RSA的计算问题

- RSA的加密与解密的模幂运算

$66^{77} \bmod 119 =$

$1.27316015002712725024996823827 45e+140 \bmod 119$

首先，用模运算的性质：

$(a \times b) \bmod n = [(a \bmod n) \times (b \bmod n)] \bmod n$

减小中间运算结果.

RSA单向陷门函数及其应用

- 幂运算的快速算法

例如：求x^{16}，直接计算的话需做15次乘法。

然而如果重复对每个部分结果做平方运算即求x，x^2，x^4，x^8，x^{16}则只需4次乘法。

求a^t可如下进行，其中a，t是正整数：

将t表示为二进制形式$b_k\, b_{k-1}\ldots b_0$，即

$$t=b_k 2^k+b_{k-1}2^{k-1}+\ldots+b_1 2+b_0$$

因此

$$a^{\mathrm{t}}=\left(\cdots\left(\left(\left(a^{b_k}\right)^2 a^{b_{k-1}}\right)^2 a^{b_{k-2}}\right)^2\cdots a^{b_1}\right)^2 a^{b_0}$$

RSA单向陷门函数及其应用

例：求a^{19}

$19=1\times2^4+0\times2^3+0\times2^2+1\times2^1+1\times2^0$

所以

$$a^{19}=((((a^1)^2a^0)^2a^0)^2a^1)^2a^1$$

练习：求a^7和a^8，并统计快速运算法的运算次数.

RSA单向陷门函数及其应用

➢ RSA的解密运算

加密很快，指数小；解密比较慢，指数较大. 利用中国剩余定理CRT，加快计算速度.

- CRT 对RSA解密算法生成两个解密方程，即：

$$M_1 = M \bmod p = (C \bmod p)^{d \bmod (p-1)} \bmod p$$

$$M_2 = M \bmod q = (C \bmod q)^{d \bmod (q-1)} \bmod q$$

- 解方程组

$$M = M_1 \bmod p;$$

$$M = M_2 \bmod q.$$

- 利用CRT，具有唯一解：

$$M = [(M_2 + q - M_1)u \bmod q]\, p + M_1.$$

其中 $p.u \bmod q = 1$.

RSA加密算法及其应用

9）RSA的应用--PKCS #1 v2.2: OAEP (RFC8017)

RSA的选择密文攻击：

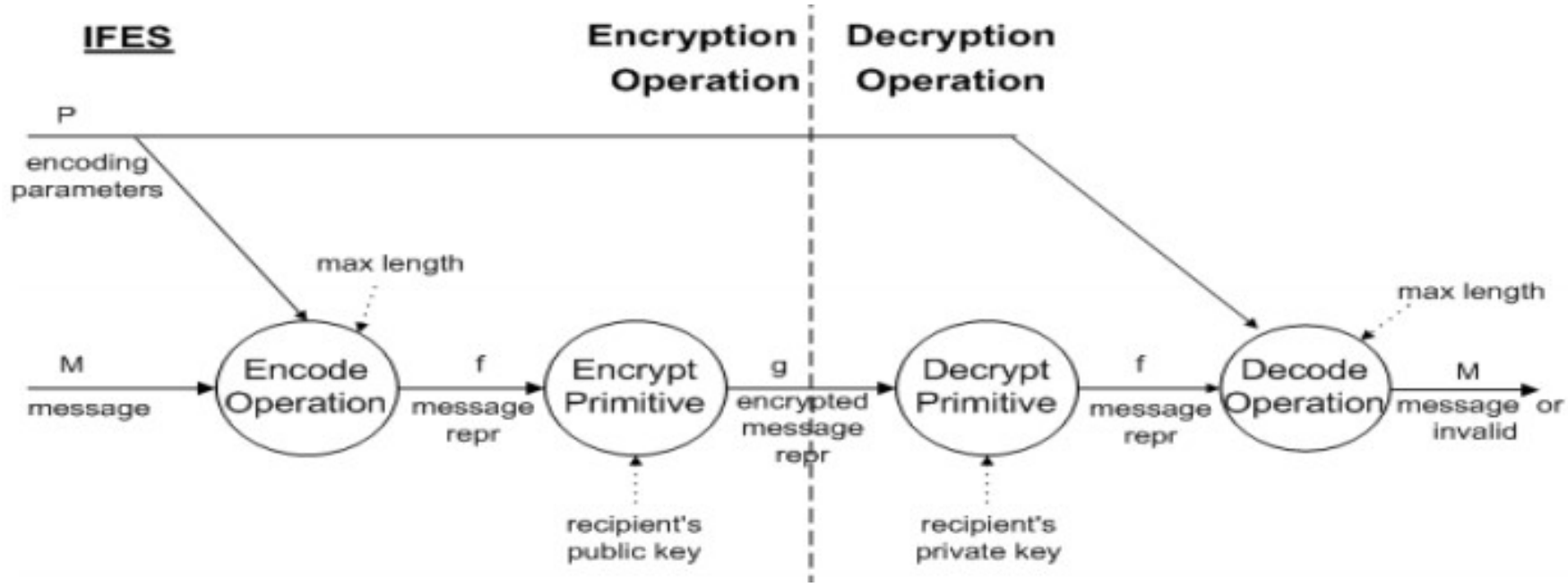
攻击者是将希望解密的目标密文C作一下伪装r^eC，让拥有私钥的实体解密。然后，脱去伪装就可得到目标密文所对应的目标明文：

$(r^eC)^d = r^{ed} * C^d \bmod n = r * M \bmod n$，所以

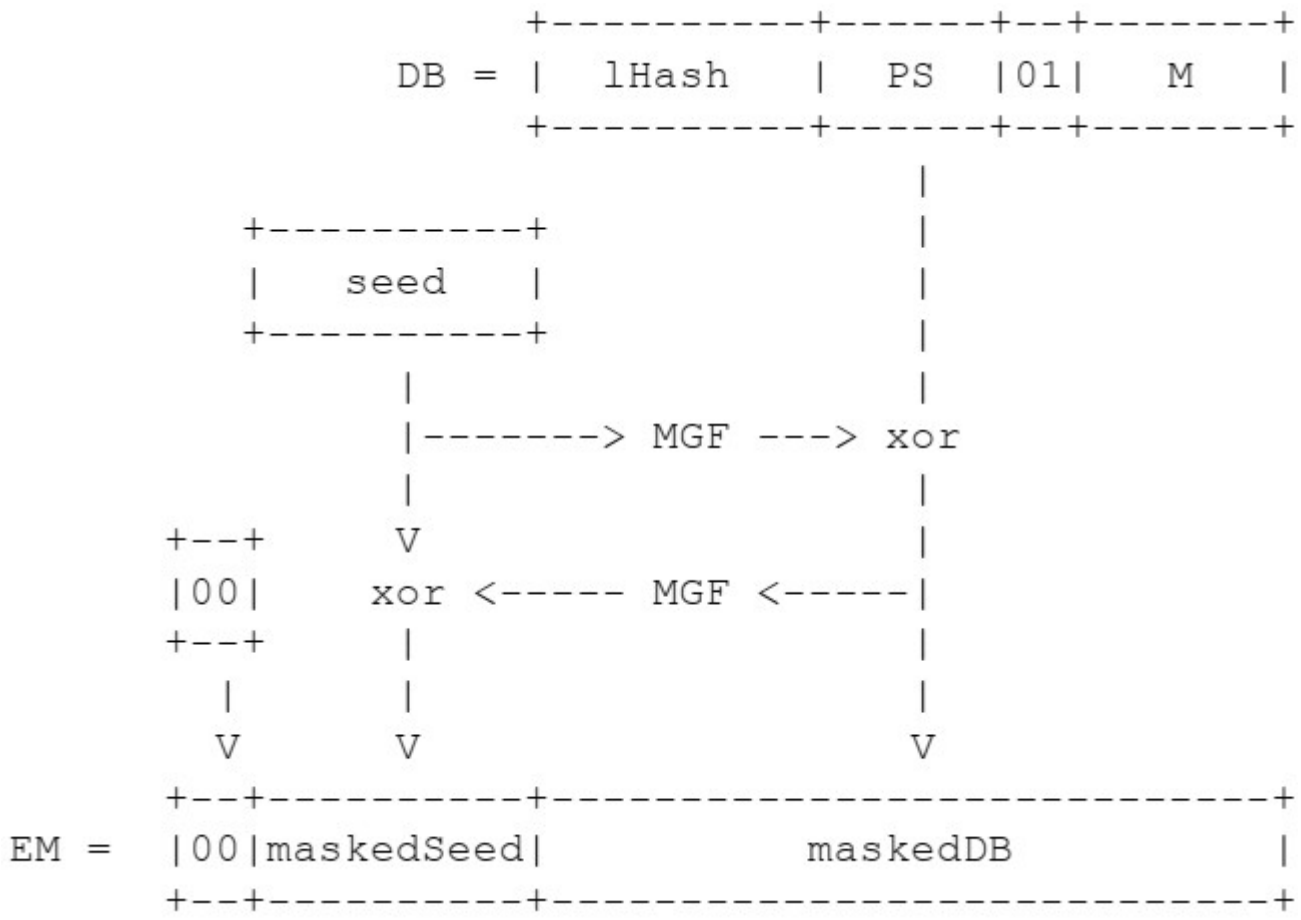
$$M = (r^eC)^d * r^{-1} \bmod n \ .$$

RSA加密算法及其应用

9)RSA的应用--PKCS #1 v2.2: OAEP (RFC8017)



RSA加密算法及其应用



EME-OAEP Encoding Operation

RSA加密算法及其应用

- RSAES-OAEP-ENCRYPT ((n, e), M, L)
  - Hash: hash function (hLen 是hash function 输出的字节数)
  - MGF: mask generation function
- 输入:
  - (n, e):接收者的RSA public key (k 表示RSA 模数n的字节数)
  - M: 明文, 长度为mLen字节, mLen <= k - 2hLen – 2
  - L: optional label to be associated with the message; the default value for L, if L is not provided, is the empty string

RSA加密算法及其应用

- RSAES-OAEP-ENCRYPT ((n, e), M, L)
- 输出:
  - C: 密文, 长度为k字节
  - Errors: "message too long"; "label too long"
- 步骤:

  1. 长度检测:

  a. L小于等于 hash function的限制 (2^{61} - 1 字节 SHA-256), 否则输出"label too long" 并停止.

  b. 如果 mLen > k - 2hLen - 2, 输出 "message too long"并停止.

RSA加密算法及其应用

- RSAES-OAEP-ENCRYPT ((n, e), M, L)
- 步骤:

  2. EME-OAEP 编码:

  a. If the label L is not provided, let L be the empty string. 令lHash = Hash(L), 输出长度为hLen字节.

  b. 生成一个填充PS包含 k - mLen -2hLen - 2 个0 字节. PS 的长度可能为0.

  c. lHash, PS, 十六进制常数字节0x01, 和明文链接在一起组成长度为k - hLen – 1字节的数据块DB:

  DB = lHash || PS || 0x01 || M.

RSA加密算法及其应用

- RSAES-OAEP-ENCRYPT ((n, e), M, L)
- 步骤:

  2. EME-OAEP 编码:

  d. 生成长度为hLen的随机字节串.

  e. 令dbMask = MGF(seed, k - hLen - 1).

  f. 令maskedDB = DB⊕dbMask.

  g. 令 seedMask = MGF(maskedDB, hLen).

  h. 令 maskedSeed = seed⊕seedMask.

  i. 将maskedSeed和maskedDB串起来，并在头部添加0x00构成k字节长的 EM

  EM = 0x00 || maskedSeed || maskedDB.

RSA加密算法及其应用

- RSAES-OAEP-ENCRYPT ((n, e), M, L)
- 步骤:

  3. RSA 加密:

  a. 将二进制字节串EM转化成整数表示:

  $$m = OS2IP\ (EM).$$

  b. 使用公钥(n, e)和 RSAEP 加密原语对m运算得密文c:

  $$c = RSAEP\ ((n, e), m).$$

  c. 将整数表示的密文c 转化为长为k的二进制字节流C:

  $$C = I2OSP\ (c, k).$$

  d.输出密文C.

RSA加密算法及其应用

- RSAES-OAEP-DECRYPT (K, C, L)
- 注释:

- lHash 是可选标签L的长度.
- 如果 L 是空串, lHash 根据选择hash函数不同而赋予如下十六进制值:

SHA-1: (0x)da39a3ee 5e6b4b0d 3255bfef 95601890 afd80709
SHA-256: (0x)e3b0c442 98fc1c14 9afbf4c8 996fb924 27ae41e4 649b934c a495991b 7852b855
SHA-384: (0x)38b060a7 51ac9638 4cd9327e b1b1e36a 21fdb711 14be0743 4c0cc7bf 63f6e1da 274edebf e76f65fb d51ad2f1 4898b95b
SHA-512: (0x)cf83e135 7eefb8bd f1542850 d66d8007 d620e405 0b5715dc 83f4a921 d36ce9ce 47d0d13c 5d85f2b0 ff8318d2 877eec2f 63b931bd 47417a81 a538327a f927da3e

RSA加密算法及其应用

- RSAES-OAEP-DECRYPT (K, C, L)
- 输入:

- Hash：hash function (输出长度为字节)；

- MGF：mask generation function；

- K 接收者的RSA 私钥(k 表示RSA 模数n的字节数，k >= 2hLen + 2)；

- C：密文, 长度为k个字节；

- L：附在消息后的可选标签; 默认值为 L, 若L 未提供，则为空串；

- 输出:
  - M: 明文, 长度为mLen字节mLen <= k - 2hLen – 2
  - Error: "decryption error"

RSA加密算法及其应用

- RSAES-OAEP-DECRYPT (K, C, L)
- 步骤:

1. 长度检验:

a. 如果L 的长度超过相应 hash function的限制 (2^61 - 1 字节 for SHA-256), 输出“decryption error” 并停止.

b. 如果密文C 的长度不是 k 字节, 输出"decryption error" 并停止.

c. 如果 k < 2hLen + 2, 输出 "decryption error"并停止.

2. RSA 解密:

a. 将密文二进制流C转化为整数表示:

$$c = OS2IP (C).$$

RSA加密算法及其应用

- RSAES-OAEP-DECRYPT (K, C, L)
- 步骤:

2. RSA 解密:

a. 将密文二进制流C转化为整数表示:

$$c = OS2IP\ (C).$$

b. 输入RSA私钥K和密文c到RSADP 解密原语生成消息m:

$$m = RSADP\ (K, c).$$

如果RSADP输出“ciphertext representative out of range” utputs (也就是说c >= n), 输出“decryption error” 并停.

c. 将整数表示消息m转化为长度为k的二进制字节流EM:

$$EM = I2OSP\ (m, k).$$

RSA加密算法及其应用

- RSAES-OAEP-DECRYPT (K, C, L)
- 步骤:

3. EME-OAEP 解码:

a. 如果标签L为提供，令L为空字符串；令lHash = Hash(L)为长hLen的字节串；

b. 将EM分解为一个单一的字节Y，一个长为hLen字节的串maskedSeed和长为k - hLen - 1 的字节串maskedDB：

EM = Y || maskedSeed || maskedDB.

c. 令 seedMask = MGF(maskedDB, hLen).

d. 令 seed = maskedSeed ⊕ seedMask.

e. 令 dbMask = MGF(seed, k - hLen - 1).

f. 令 DB = maskedDB ⊕ dbMask.

RSA加密算法及其应用

- RSAES-OAEP-DECRYPT (K, C, L)
- 步骤:

3. EME-OAEP 解码:

g. 将DB分割成长度为hLen字节的二进制串 lHash’ , 一个头为0x00的填充字段PS（可能为空）,和一个消息M ：

DB = lHash’ || PS || 0x01 || M.

如果没有头为0x01的字节来区分PS和M, 如果lHash与lHash’不相等, 或者Y非零，输出“decryption error” 并停止.

4. 输出消息M.

本节主要内容

- 对称密码体制面临的问题
- 公钥密码体制的发展
- RSA单向陷门函数及其应用

THE END!