对称加密与伪随机性
Private-Key Encryption and Pseudorandomness

安全目标

无条件安全（unconditional security）

即使攻击者具有无限的计算资源，也无法攻破密码体制，我们称这种密码体制是无条件安全的。

安全目标

计算安全（computational security）

攻击者具有有限的计算资源，很难攻破密码体制，我们称这种密码体制是计算安全的。

如何定义“有限的计算资源”?

如何定义“很难”?

如何定义“有限的计算资源”?

- 有限的计算资源
  - 多少台计算机?
  - 每台计算机有多快的CPU?
  - 每台计算机有多大的内存?
  - 程序允许在其上运行多久?
  - 等等

- 如何给出统一定义呢?

如何定义“有限的计算资源”?

- 算法
  - O表示法
- 有限的计算资源
  - 将敌手视作一个算法。
  - 该算法的时间复杂度是 $O(n^c)$
  - n是安全参数，c是一个常数
- 敌手能够利用有限的计算资源，遍历所有可能的密钥吗?

如何定义“很难”

- 计算安全：攻破密码算法的概率可忽略。
- 无条件安全：无法攻破密码算法。

- 如何定义概率可忽略?
  - 多少算小呢?

  - 安全参数：n（密钥的二进制位数）
  - 当n足够长时，攻破密码算法的概率趋近于0。
  - 形式化定义，利用可忽略函数实现。

可忽略函数

1.安全参数n ： 密钥的二进制位数，简称密钥位数

2.多项式$q(n)$ ： 密钥位数的多项式

如$2n^2 + 3n + 5$

3.可忽略函数$\varepsilon(n)$ ： 极限趋向于0的表达式。通常是分数形式

如 $\frac{3n^3 + 5}{2^n}$ $\frac{3n^3 + 5}{2^n - 2n^4 + n^2}$

可忽略函数

- 两个可忽略函数的和是可忽略函数吗?
- 常数与可忽略函数的积是可忽略函数吗?

加密算法的定义

加密算法包括三个子算法：

1. 密钥生成子算法(Gen)

   算法输入：安全参数n；算法输出：满足特定分布的密钥k

2. 加密子算法(Enc)

   算法输入：密钥k和明文m；算法输出：密文c，$c \leftarrow Enc(k, m)$

3. 解密子算法(Dec)

   算法输入：密钥k和密文c；算法输出：明文m，$m = Dec(k, c)$

   满足：$Dec(k, Enc(k, m)) = m$

敌手能力

唯密文攻击，已知明文攻击，选择明文攻击，选择密文攻击

唯密文攻击实验：

1. 运行密钥生成算法Gen，确定密钥k。
2. 随机加密若干明文，并将密文发送给攻击者。
3. 攻击者发送明文空间中的两条消息m0和m1。
4. 随机加密其中一条明文，并将对应密文发送给攻击者。
5. 随机加密若干明文，并将密文发送给攻击者。
6. 攻击者根据密文猜测对应的明文是m0还是m1。

加密算法

- 唯密文攻击实验：

- 1. 运行密钥生成算法Gen，确定密钥k。
- 2. 随机加密若干明文，并将密文发送给攻击者。
- 3. 攻击者发送明文空间中的两条消息m0和m1。
- 4. 随机加密其中一条明文，并将对应密文发送给攻击者。
- 5. 随机加密若干明文，并将密文发送给攻击者。
- 6. 攻击者根据密文猜测对应的明文是m0还是m1。

- 定理：
- 假定n表示安全参数。如果加密方案在唯密文攻击下是计算安全的，那么攻击者猜对的概率不超过1/2+negl(n)。

加密算法的安全定义

- 安全参数：n
- 唯密文攻击：eav
- 密码算法：Π
- 攻击实验：PrivK

- 定理:
- 如果加密方案在唯密文攻击下是计算安全的，那么攻击者猜对的概率不超过1/2+negl(n)。

$$\Pr\left[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1\right]\le\frac{1}{2}+negl(n)$$

加密算法的安全定义

如何证明下述两个不等式是等价的?

$$\Pr\left[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1\right]\le\frac{1}{2}+negl(n)$$

$$\left|\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,0)\right)=1\right]-\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,1)\right)=1\right]\right|\le negl(n)$$

加密算法的安全定义

如何证明下述两个不等式是等价的?

$$\Pr\left[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1\right]\le\frac{1}{2}+negl(n)$$

$$\left|\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,0)\right)=1\right]-\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,1)\right)=1\right]\right|\le negl(n)$$

$$\begin{aligned}
&\Pr\left[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)=1\right]\\
&=\frac{1}{2}\left(\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,0)\right)=0\right]+\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,1)\right)=1\right]\right)\\
&=\frac{1}{2}\left(1-\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,0)\right)=1\right]+\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,1)\right)=1\right]\right)\\
&=\frac{1}{2}-\frac{1}{2}\left(\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,0)\right)=1\right]-\Pr\left[\mathrm{output}\left(\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n,1)\right)=1\right]\right)
\end{aligned}$$

加密算法设计的两种基本策略

流密码(序列密码，Stream Cipher)

- 将密钥扩展到与密文等长
- 加密算法一般为异或操作
- 其安全性主要在于密钥扩展后的随机性

$M = m_1 m_2 m_3 \ldots, K = k, C = c$

其 $c = Enc_k(M)$

分组密码(Block Cipher)

- 将明文分成等长的分组（比如64位、128位）
- 对每个分组采用相同的加密算法进行加密
- 算法一般是交替使用混乱和扩散技术

- $M = m_1 m_2 m_3 \ldots., K = k_1 k_2 k_3 \ldots, C = c_1 c_2 c_3 \ldots$
- 其 $c_i = Enc_{k_i}(m_i)$

流密码

流密码(序列密码，Stream Cipher)

- 将密钥扩展到与密文等长
- 加密算法一般为异或操作
- 其安全性主要在于密钥扩展后的随机性

$$M = m_1 m_2 m_3 \ldots, K = k, C = c$$

其 $c = Enc_k(M)$

-
- 如何将密钥扩展后具有随机性?
- 判断标准：在有限计算资源下，该数与随机数区分概率可忽略不计。
-
- 伪随机数发生器

伪随机数生成器

- n：　安全参数
- l(n)：　扩展因子，n的多项式
- k：　密钥，长度为n
- G(k)：　确定性多项式时间算法，k→l(n)

- 1.扩展性
- $l(n) > n$
- 2.伪随机性，对任意概率多项式时间算法D

$$\left|\Pr\left[D(r)=1\right]-\Pr\left[D\left(G\left(k\right)\right)=1\right]\right| \le negl(n)$$

序列加密算法的构造

序列加密算法

1. Gen

   n→k，k是一个[0,2^n]均匀分布中随机取出的一个整数。

2. Env

   (k, m) = c;　　$c = G(k) \oplus m$

3. Dec

   (k, c) = m；　　$m = G(k) \oplus c$

明文m和密文c的长度均为l(n)。

证明

理想加密算法

1. Gen

   r，r是一个[0,2^l(n)]均匀分布中随机取出的一个整数。

2. Enc

   (r, m) = c, $c = r \oplus m$

3. Dec

   (r, c) = m； $m = r \oplus c$

r是与明文长度相同的随机数。

证明

唯密文攻击实验：

1. 运行密钥生成算法Gen，确定密钥k。
2. 随机加密若干明文，并将密文发送给攻击者。
3. 攻击者发送明文空间中的两条消息m0和m1。
4. 随机加密其中一条明文，并将对应密文发送给攻击者。
5. 随机加密若干明文，并将密文发送给攻击者。
6. 攻击者根据密文猜测对应的明文是m0还是m1。

在唯密文攻击下，敌手攻击理想加密算法成功概率为1/2。

证明

在唯密文攻击下，敌手攻击理想加密算法成功概率为1/2。

理想加密算法与序列加密算法的区别仅为用随机数代替了伪随机数。

$$\left|\Pr\left[D(r)=1\right]-\Pr\left[D\left(G\left(k\right)\right)=1\right]\right| \le negl\left(n\right)$$

所以，序列加密算法的成功概率不超过1/2+negl(n)

攻击实验

前述方案在选择明文攻击下是安全的吗?

- 1. 运行密钥生成算法Gen，确定密钥k
- 2. 攻击者选择明文空间中的明文
- 3. 将明文加密后生成并发送给攻击者
  - 2和3步 可反复运行安全参数的多项式次
- 4. 攻击者发送明文空间中的两条明文m0和m1
- 5. 随机加密其中一条明文，并将对应密文发送给攻击者
- 6. 攻击者选择明文空间中的明文
- 7. 将明文加密后生成并发送给攻击者
  - 6和7步可反复运行安全参数的多项式次
- 8. 攻击者根据密文猜测对应的明文是m0还是m1

序列加密算法的构造

序列加密算法

1. Gen

   n→k，k是一个[0,2^n]均匀分布中随机取出的一个整数。

2. Enc

   (k, m) → c，$c \leftarrow \left(r, G\left(r \oplus k\right) \oplus m\right)$

3. Dec

   (k, (c1,c2)) → m；$m \leftarrow G\left(c_1 \oplus k\right) \oplus c_2$

已知k，
如何构造 $G\left(k\right)$

明文m，随机数r，密文c的长度均为l(n)。

伪随机数生成器

- 伪随机数生成器的伪代码（初始化）
  - **for** i **from** 0 **to** 255
  - S[i] := i
  - **endfor**
  - j := 0
  - **for** i **from** 0 **to** 255
  - j := (j + S[i] + K[i mod keylength]) mod 256
  - swap values of S[i] and S[j]
  - **endfor**

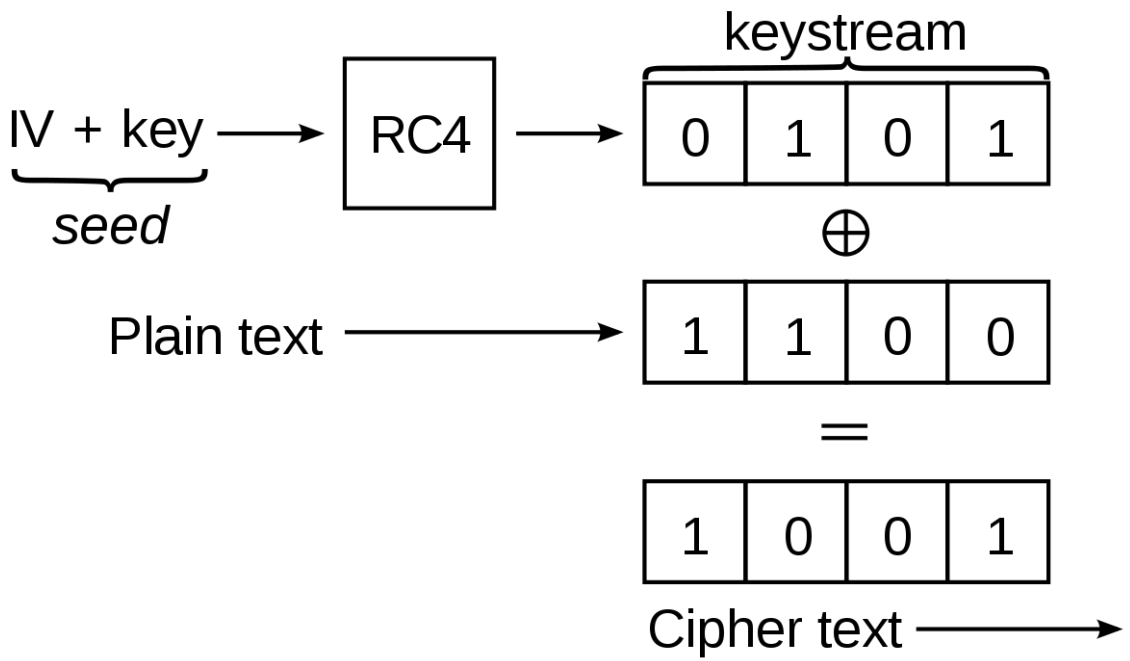
伪随机数发生器

- 伪随机数生成器的伪代码（生成伪随机数）
  - i := j := 0
  - **while** GeneratingOutput:
  - i := (i + 1) mod 256
  - j := (j + S[i]) mod 256
  - swap values of S[i] and S[j]
  - SR:= S[(S[i] + S[j]) mod 256]
  - output SR
  - **endwhile**

RC4序列加密算法

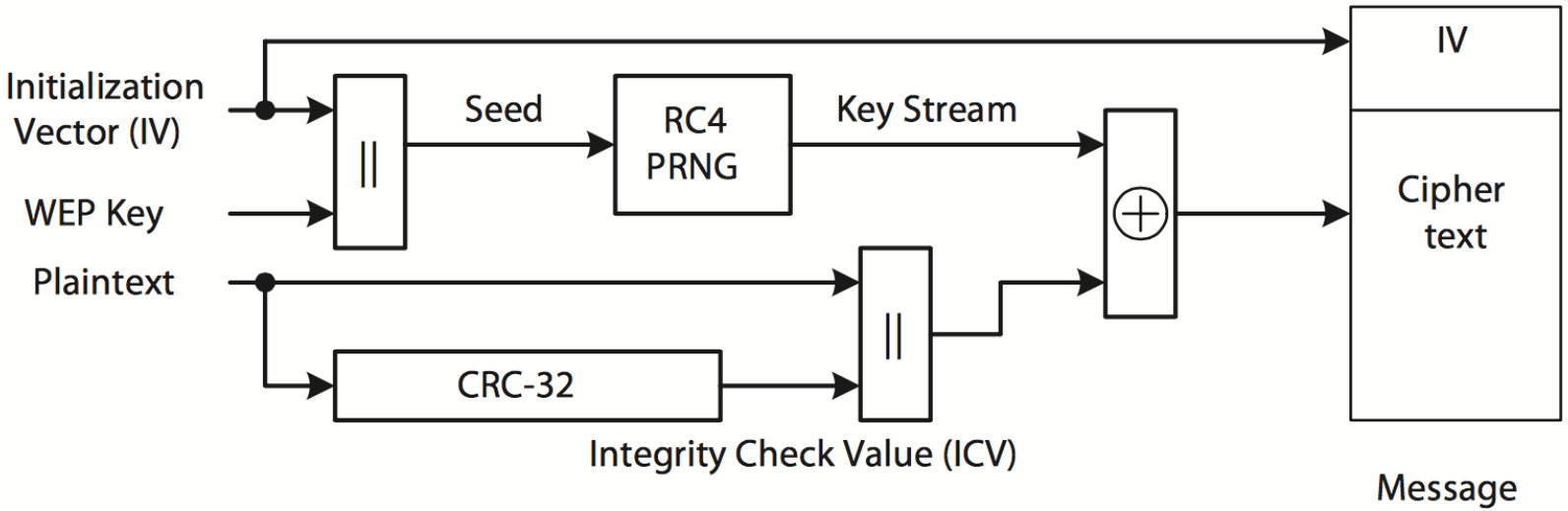
- RC4序列加密算法
  - 1987年由Ronald Rivest设计
  - 全称Rivest Cipher 4
  - 确定性算法
  - 密钥长度
    - 1~256字节
    - 通常取4-16字节（32-128位）

RC4如何变成非确定性算法呢?

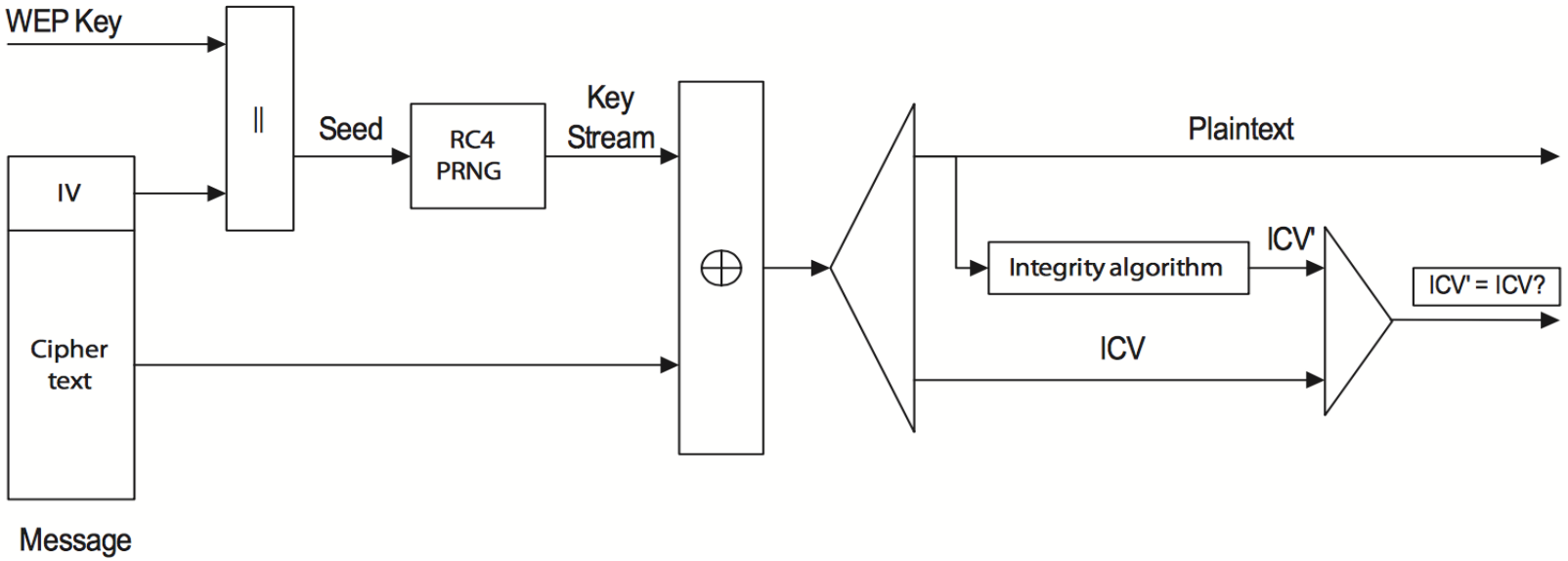


WEP

- WEP使用该方案加密数据
  - 40-bit key + 24-bit IV



WEP方案



- WEP数据帧中第一个字节消息相同，攻击者可以得到第一个字节的G(k)值。
- IV只有24bits，2^12个消息就很可能出现碰撞。
- 利用碰撞，相对容易猜测密文对应的明文。

THANKS