

密码学：
加密，哈希函数，
证明书

加密库 (和 API)

- C :
 - OpenSSL : <https://www.openssl.org>
 - 加密库 : <https://www.cryptlib.com>
 - NaCL (网络和密码学) : <https://nacl.cr.yp.to>
- Java :
 - JCA / JCE 体系结构及其实现 (Bouncy Castle :
<https://www.bouncycastle.org/fr/>)
 - 用于身份验证和授权的 JAAS - 特别是 J2EE Apache Commons Crypto (OpenSSL 包装器) : <https://github.com/apache/commons-crypto>
 - jNaCL : <https://github.com/neilalexander/jnacl>

- 蟒蛇 :
 - NaCL (网络和密码学) : <https://nacl.cr.yp.to>
 - pyOpenSSL : <https://www.pyopenssl.org/en/stable/index.html>
 - 密码学 : <https://github.com/pyca/cryptography>

- 网页 :
 - W3C 低级加密 API (在浏览器中可用) : <https://www.w3.org/TR/WebCryptoAPI/>
 - IETF Javascript 对象签名和加密 (Jose) :
<https://datatracker.ietf.org/wg/jose/about/>

加密

- 将数据（也称为纯文本或明文）转换为密文
 - 密文之外的任何人都无法读取密文
 - 该机密也称为加密或解密密钥
- 加密是一种确保机密性的机制
- 克尔科夫斯原理（1883）：
 - 系统的安全性仅取决于密钥的无知，而不取决于其他参数的无知
 - 没有“默默无闻的安全”



谜机



奥古斯丁·凯尔科夫斯 (Augustin Kerckhoffs)

对称密钥密码术



- 加密和解密是对称的，并且使用相同的密钥
- 对称密钥加密的主要问题：密钥交换
 - 密钥必须在每对通信实体之间交换
 - 实际上，还必须分发将在不被窃听者拦截的情况下使用的秘密密钥。

加密-方法

- 替代密码
 - 字母移位 (旋转)
 - 范例 : 凯撒的密码 (3 个班次)
 - 单字母密码
 - 每个字母都由另一个任意选择的字母替代
 - 弱点 : 频繁分析 (出现字母的频率.....)
 - 多字母替代密码
 - 字母符号的排列
 - 使用多个加密字母 (每个符号 1 个排列) 键选择每个字母使用哪个字母示例
 - : 维格涅尔的密码
 -
- 换位密码 (或置换密码)
 - 字母顺序被修改而不改变其值
 - 排列输入符号
 - 钥匙识别排列
- 产品密码
 - 置换和置换的组合现代加密的基础
 -

加密：转置

1个	2	3	4	加密密钥 : «3 1 4 2» 第一行 : «nrde»
[R	Ë	ñ	d	
Ë	ž	v	Ø	
Ü	s	d	Ё	
嘛		一世	ñ	密文 :
一种	升	ü	ñ	
一世	v	Ё	[R	
s	一世	ň	Ё	«nrdeveozduesimnuanleirvtsei»

加密：替代

对于键中的“B”字母，字母表中使用+1移位。

如果明文包含«R»，且密钥包含«B»，则明文将为«S»。

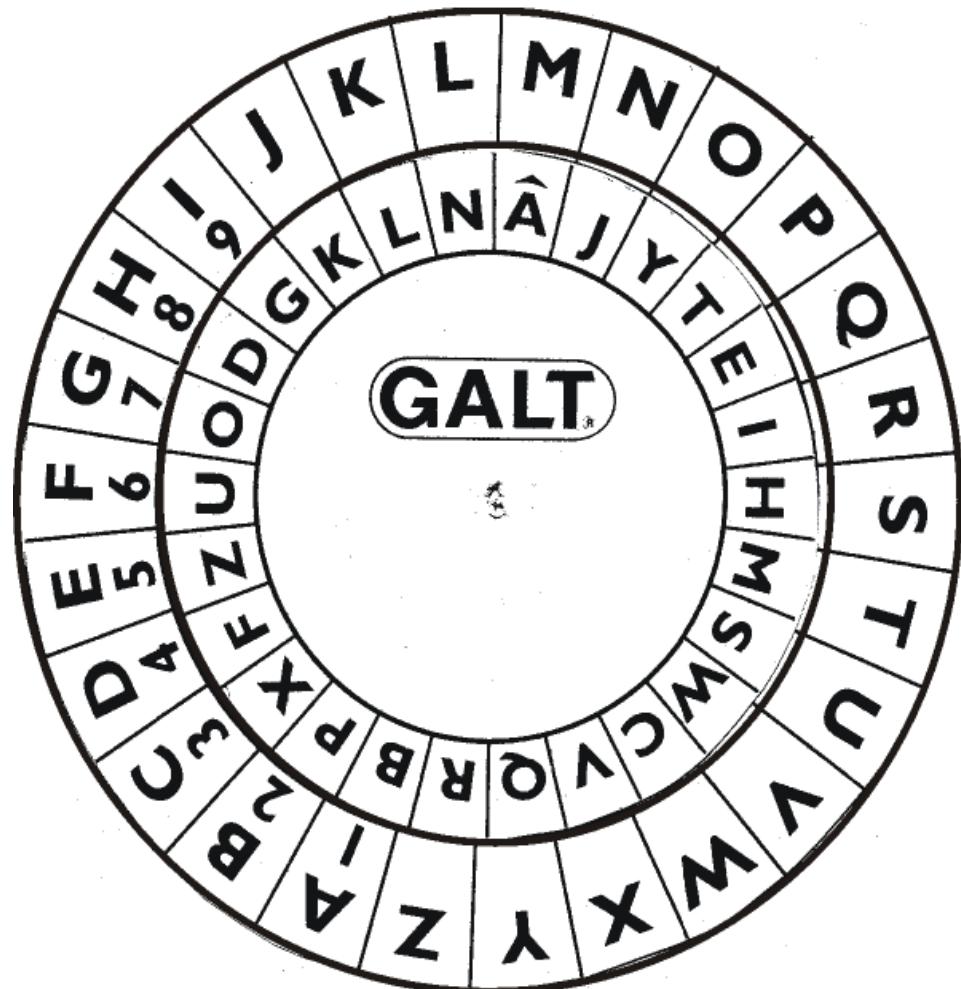
如果明文包含“E”，键包含“A”，则明文将保留为“E”

克莱尔= [R
克莱= 乙 Æ 一种 Æ 一种 ISS 班班 CE
钟琴= SEADJSFDOCR



加密：替代

更复杂
更快的换人
得到了支持
与硬件



分组密码

- 混乱：
 - 明文P和密文C统计信息之间的关系必须过于复杂，无法通过密码分析加以利用
- 基础技术：替代
- 扩散：
 - P和/或K的每个符号必须影响C的几个符号
 - 纯文本冗余必须分布在密文上
 - 基本技术：置换（换位）

DES

1973年，国家标准局美国的发起了对密码系统的呼吁。

1975年，采用了IBM（名称为“Lucifer”）开发的数据加密标准（DES）。

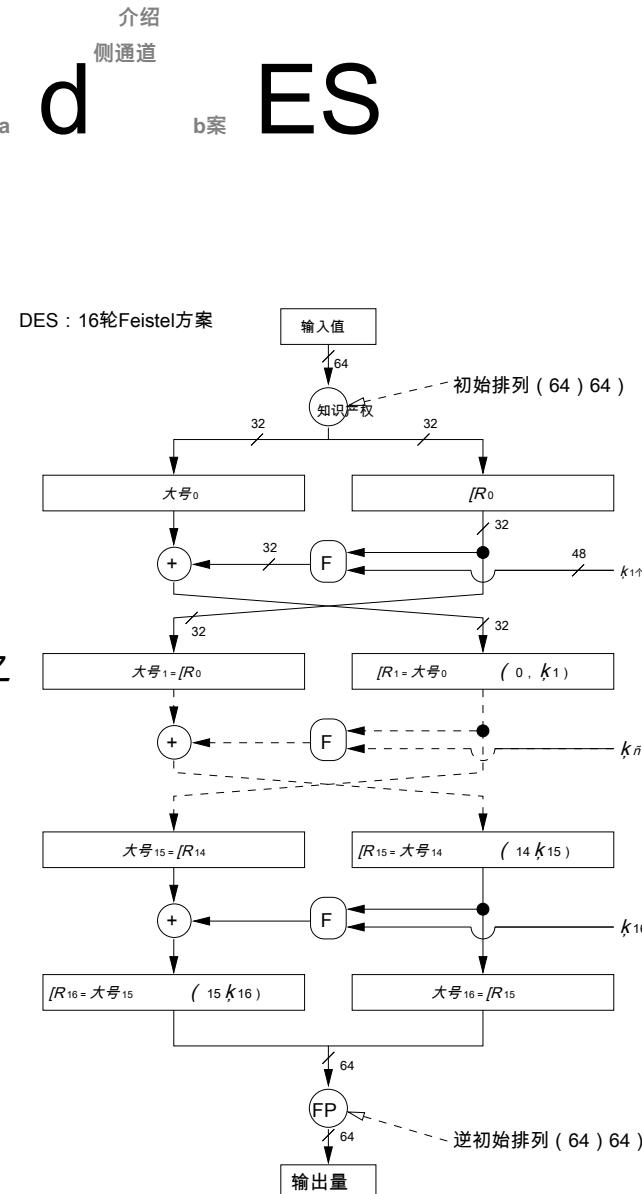
- 64位区块密码
- 56位密钥（72 057 594037037927936密钥）

使用加密

DES
定时攻击，HWSec la

算法 (1/4)

- DES-数据加密标准 (1976)
 - Feistel方案 (迭代块加密) 针对硬件实施进行了优化
 -
 - 56位密钥 , 容易进行详尽的密钥搜索 (2^{56} 可能性)
 - 一台运行 1024 GHz 处理器且频率为 1 GHz 的计算机一天之内可以浏览所有密钥
 - DES 不再安全 , 但现在用作三重DES (DED或EDE)。
- 较新的标准 : AES -高级加密标准 (2000)
 - 128-256位密钥

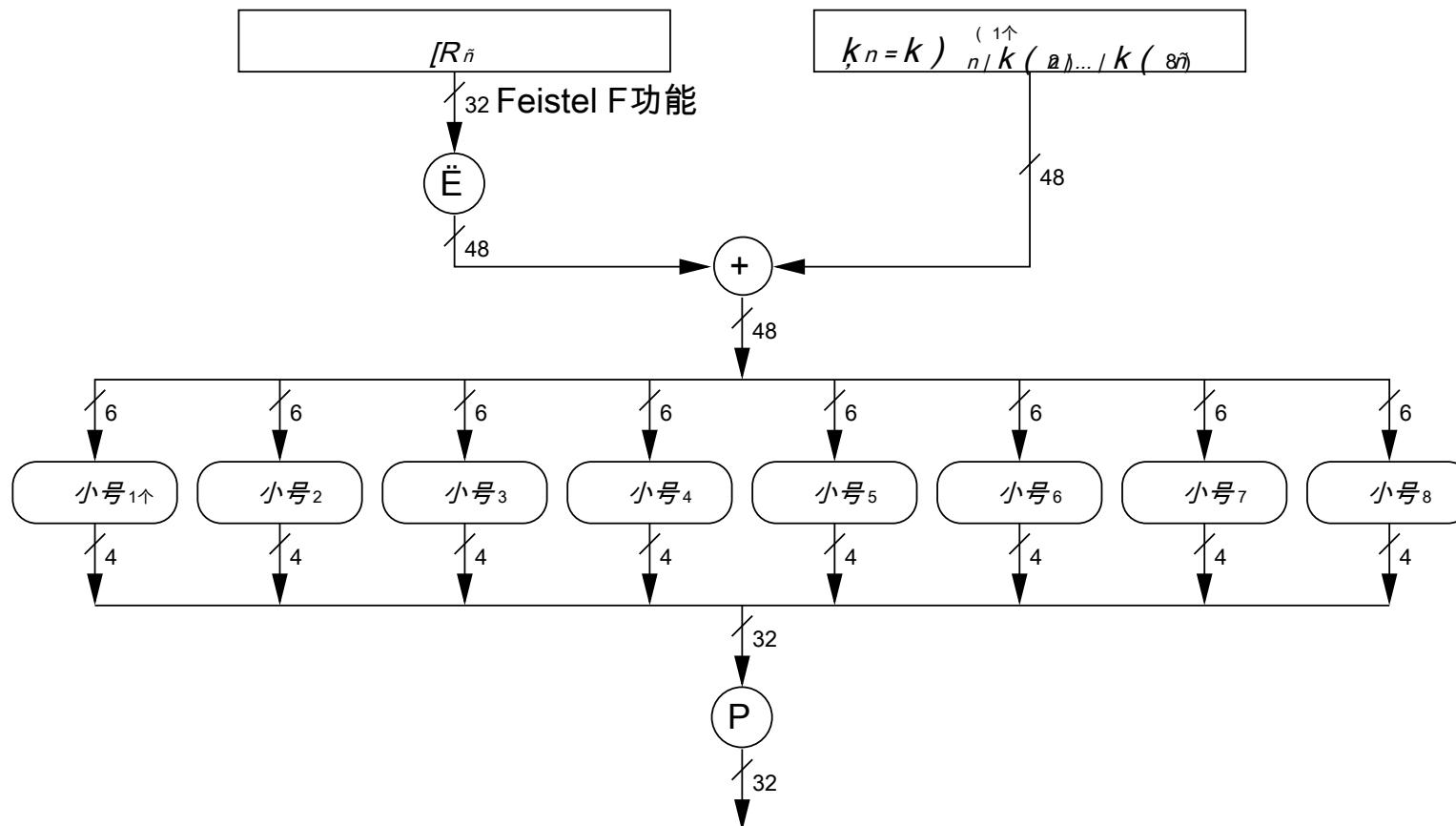


菲斯特尔 FDES 算法中的功能

介绍
侧通道

- 1) 32位移向48位 它是 S 小号
 (在 S-box 表中查找 b, d 如在称为 S-boxes 的表中)
- 2) 预计算 表

盒子 一种 昇 G (p 变量 [R-2世 4 位 H 机构) 进行了广泛讨论



链接模式

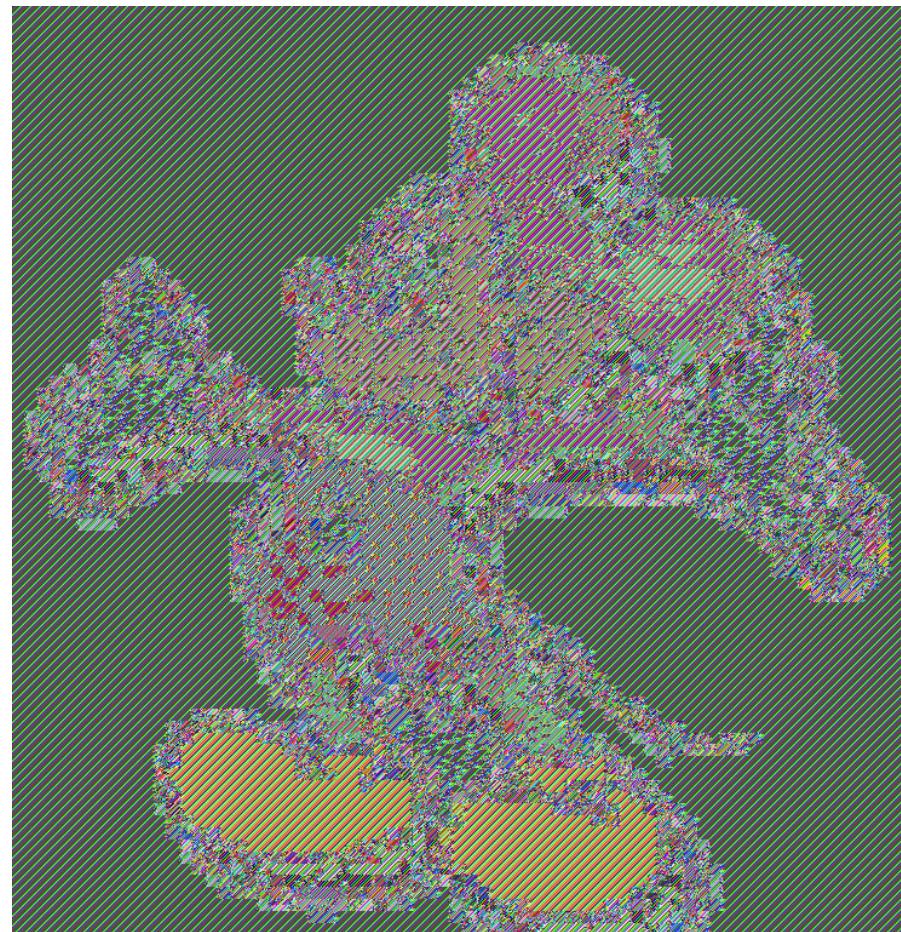
！电子密码书（ ECB ） – 统计攻击 针对DES in
ECB模式！

！密码块链接（ CBC ）
！柜台（ CTR ）
！密码反馈（ CFB ）

链接块引入了有关错误传播和重新同步的问题。



DES-ECB加密

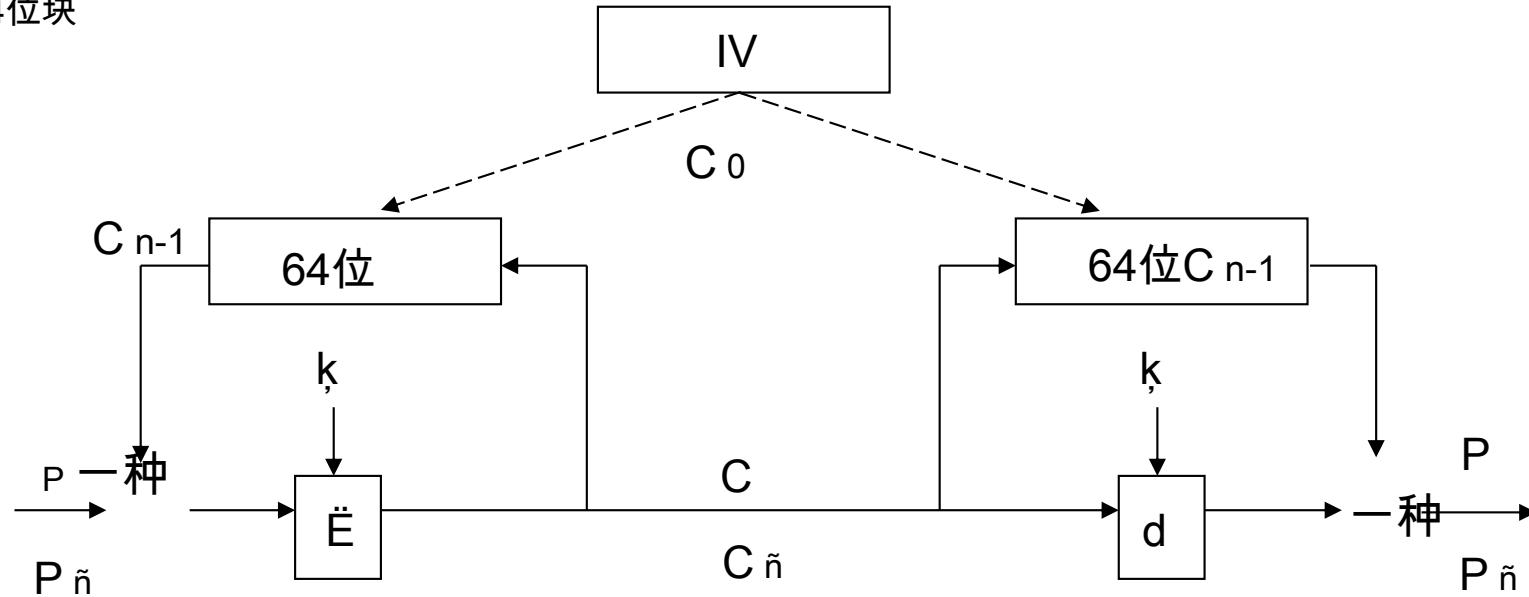




CBC模式

密码块链接

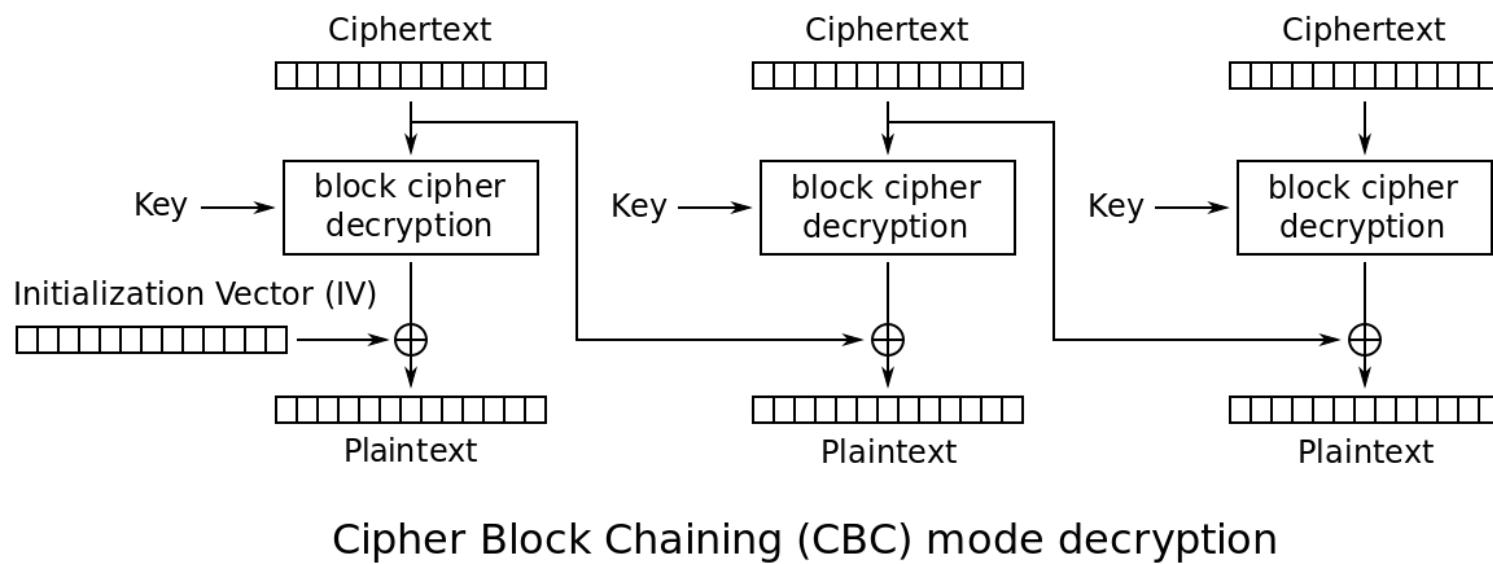
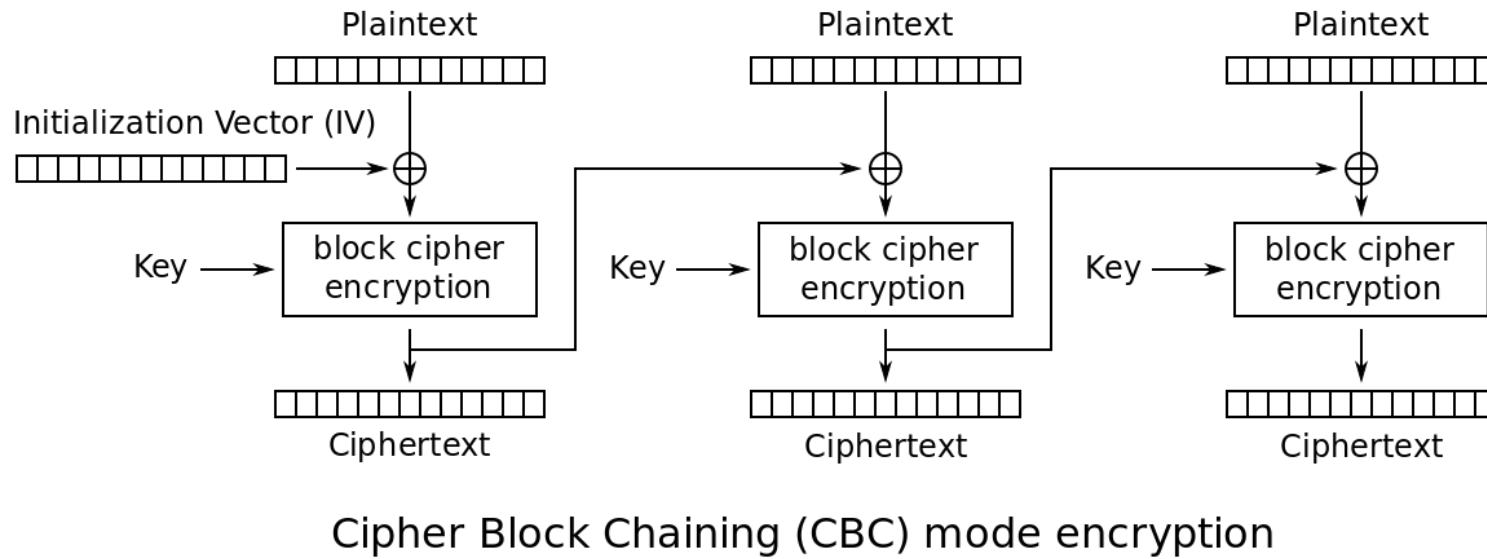
64位块



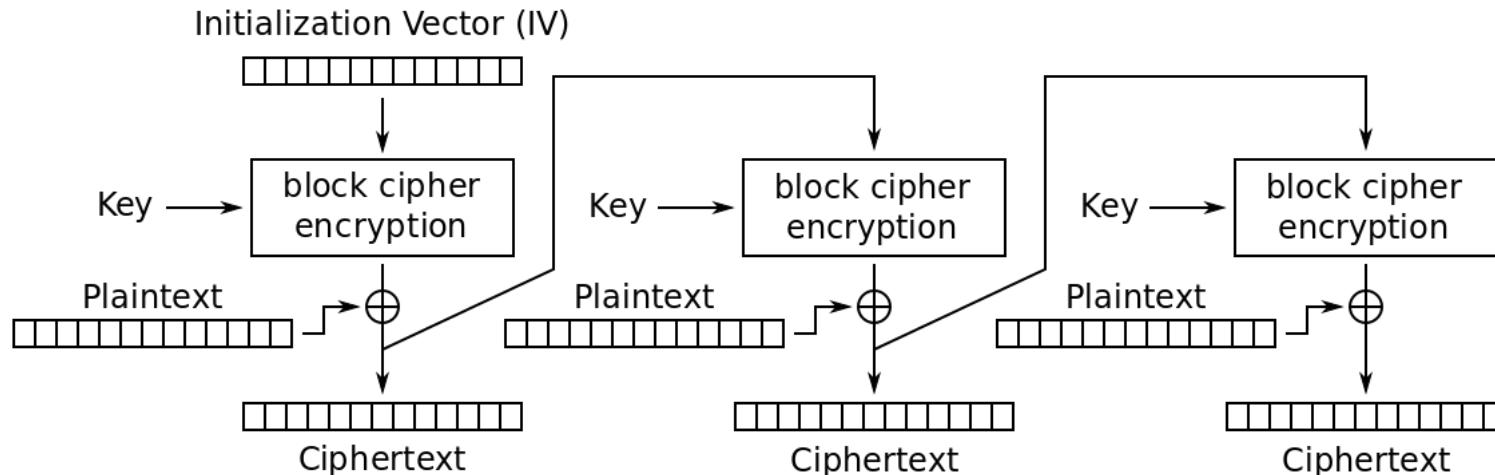
- ! $C_i = \bar{E}_K(P_i - \text{一种} C_{i-1})$
- ! $C_0 = \bar{E}_K(P_0 - \text{一种} IV)$, IV (初始化向量) 清晰传输
- ! $P_i = d_K(C_i - \text{一种} C_{i-1})$

- ! 连锁效果 : C_i 取决于所有 P_j 与 $j \leq i$
- ! C 中的最后一个块 : 取决于所有明文块
- ! 将 DES 转换为流密码
- ! 每 64 位 1 个加密 / 解密操作

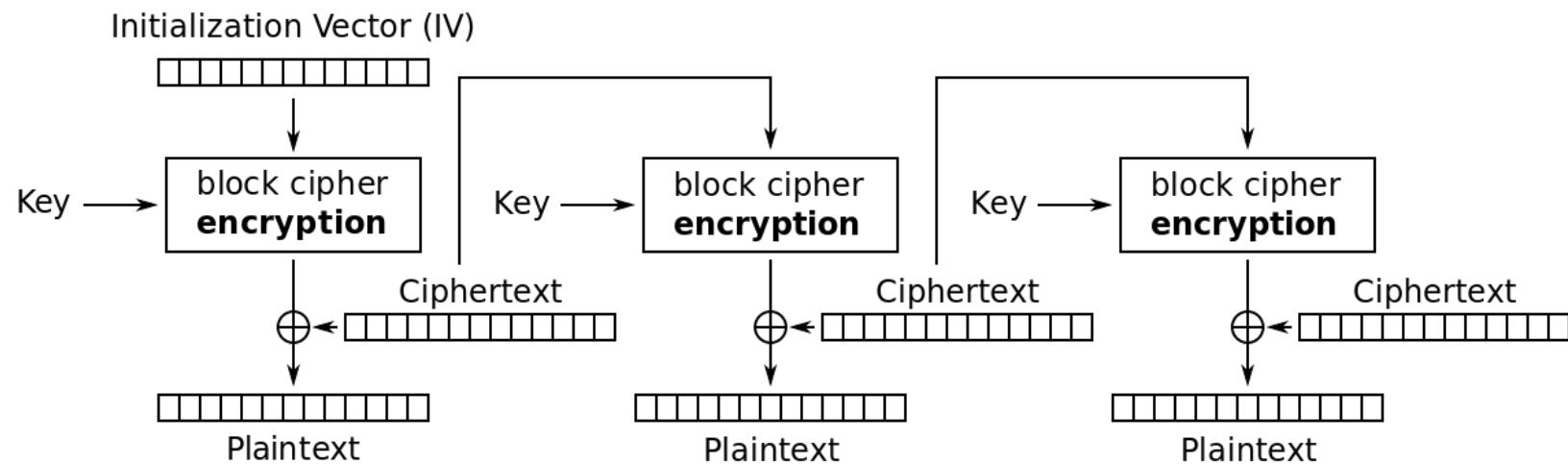
密码块链接 (CBC) 模式



密码反馈 (CFB) 模式

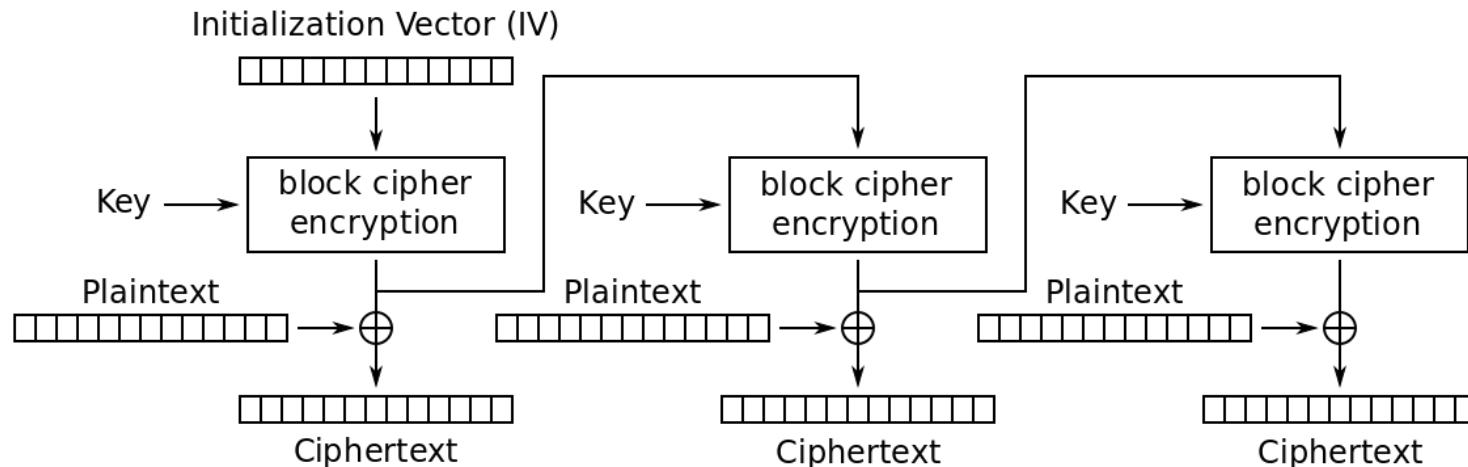


Cipher Feedback (CFB) mode encryption

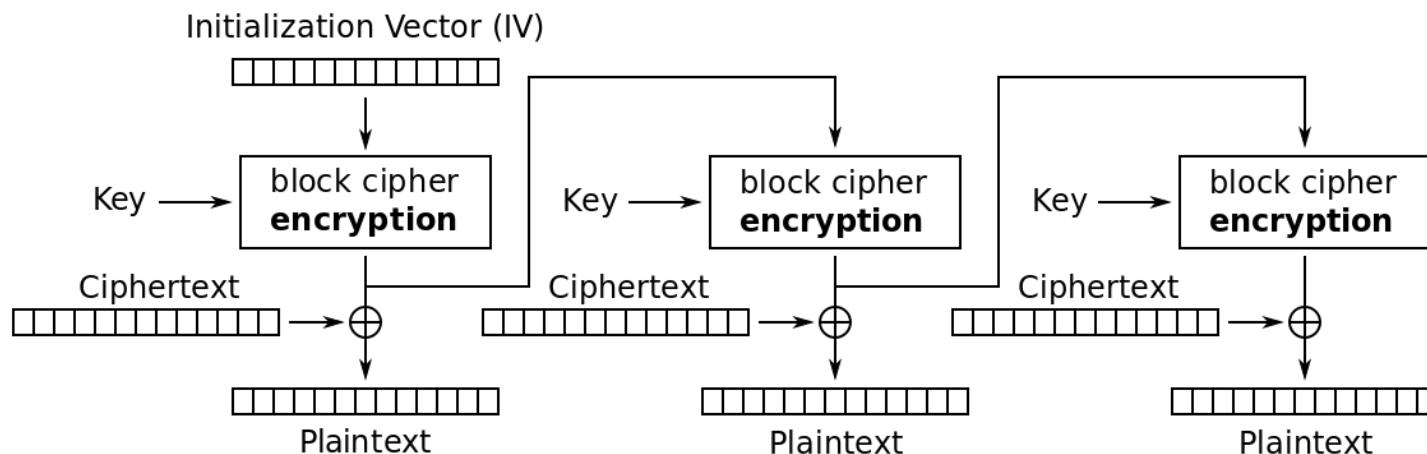


Cipher Feedback (CFB) mode decryption

输出反馈 (OFB) 模式

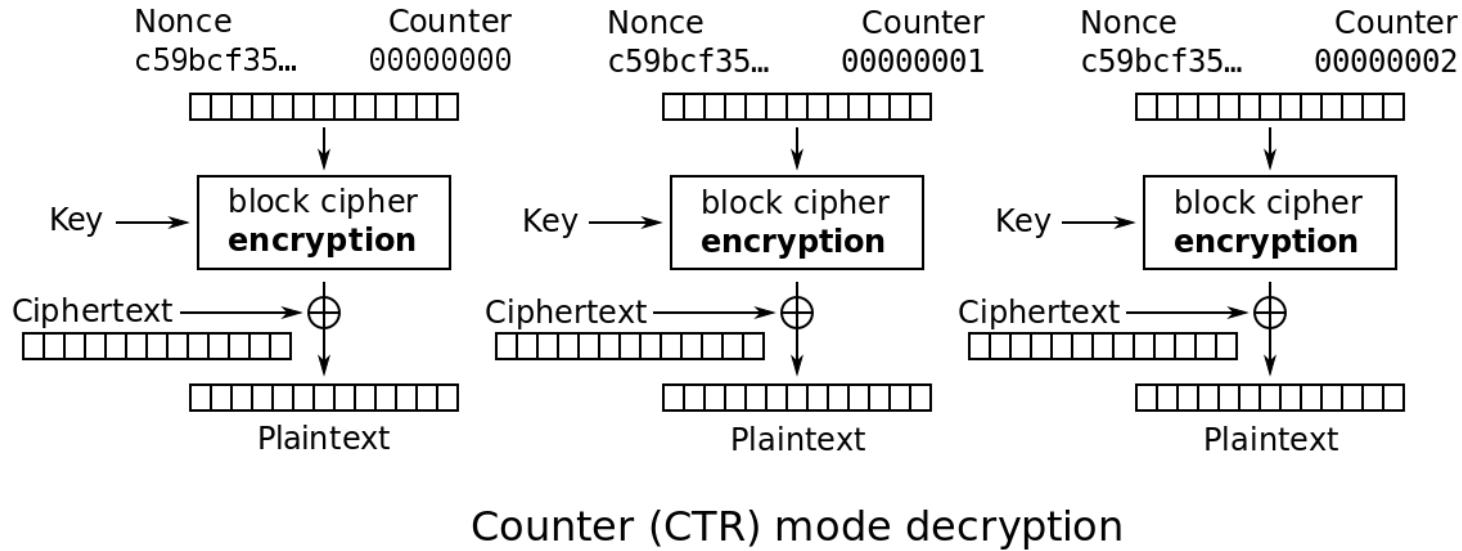
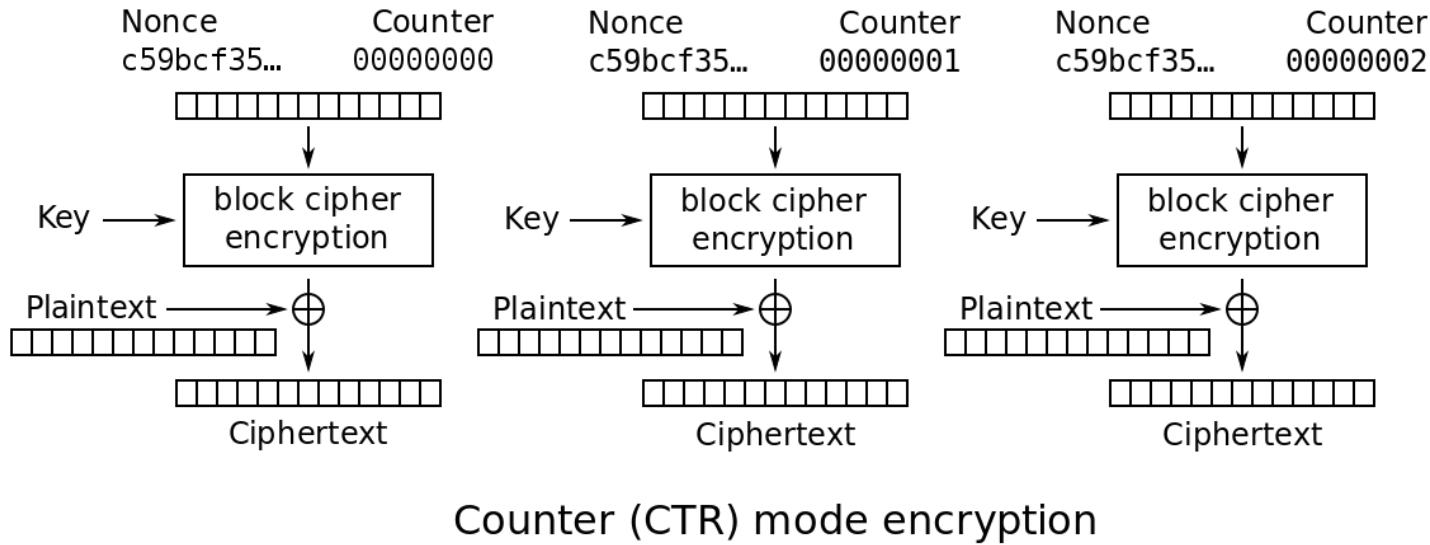


Output Feedback (OFB) mode encryption

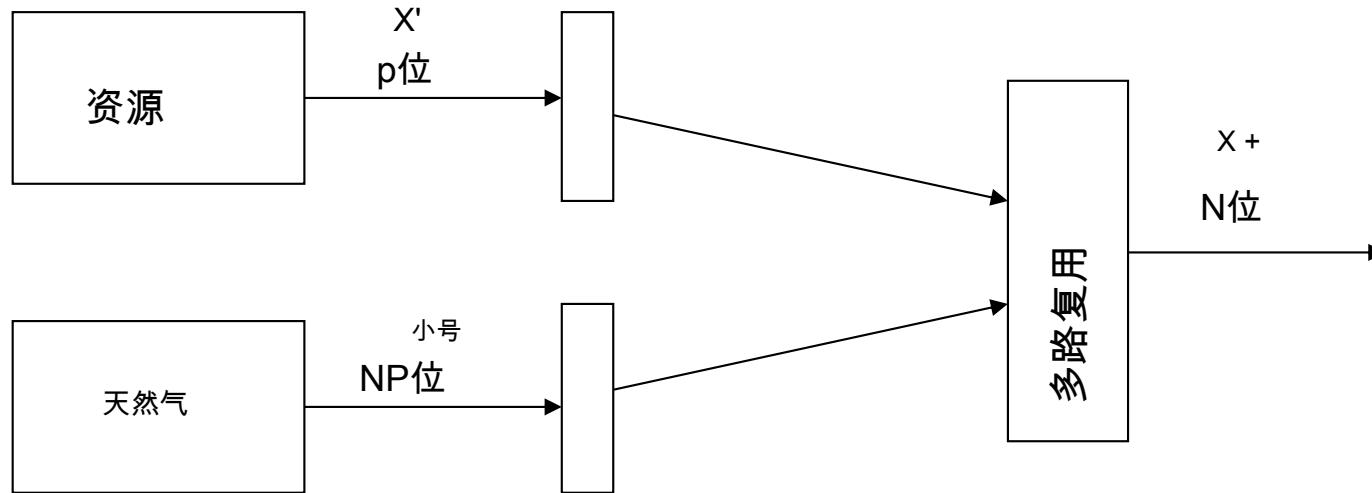


Output Feedback (OFB) mode decryption

计数器 (CTR) 模式



填充



- 两种应用：
 - 对抗密码分析（例如频率分析）
 - 使明文适应算法（输入格式大小，例如DES中的64位）
- 例
 - 加密算法：56位DES密钥， $H(K) = 56$ 位明文：8位英语ASCII编码，
 - $r = 6.7$ 位/字符每个源位的填充以及63个随机位， $p = 1$
 -
- 标准：
 - PKCS # 5和# 7（公钥密码标准），PKCS # 1-RSA-OAEP（最佳非对称加密填充），哈希零填充（ISO / IEC 10118-1）...

攻击类型

- **仅密文攻击：**

攻击者知道 C 等于 $E_K(P)$

目标：找到 P 和 K

- **已知的明文攻击：**

攻击者知道 $(P_{-世}, C_{-世})$ 与 $C_{\text{我}} = E_K(P_{-世})$

目标：找到 K

- **选择明文攻击：**

攻击者可以获得 $C_{-世}$ 从选择的 P 开始 $-世$

目标：找到 K

攻击可以是适应性的

- **选择的密文攻击：**

攻击者可以获得 $P_{-世}$ 对于选定的 $C_{-世}$

目标：找到 K

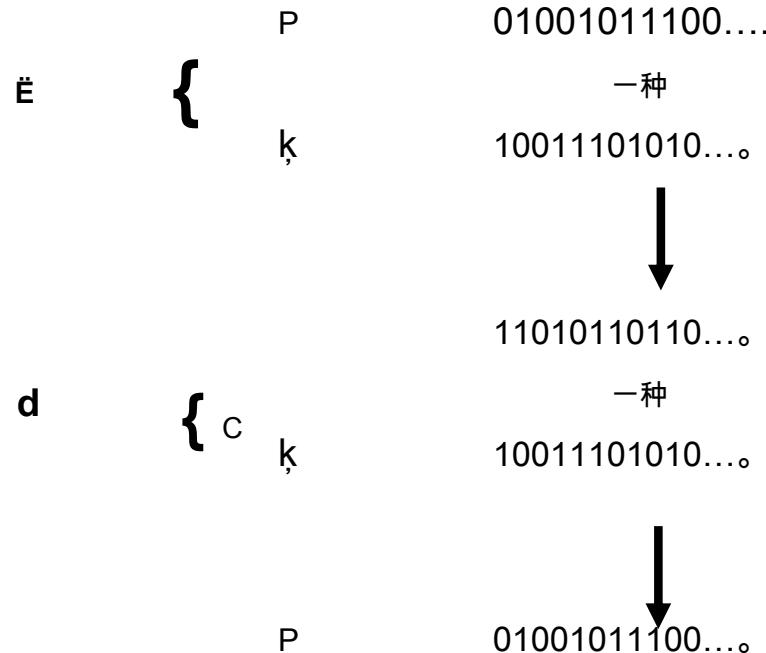
攻击也可以是适应性的

安全评估

- **无条件安全性 (完全保密)**= 该系统不受时间或资源的限制，可以安全地抵御攻击者
 - 如此：是否有足够的信息来危害系统安全性？
- **通过理论复杂性实现安全性**= 证明该系统对具有多项式能力的对手是安全的。
- **可证明的安全性**= 证明损害系统安全性等于解决了一个称为“困难”的问题（例如，离散对数，分解大整数）。
- **计算安全性 (实际安全性)**= 在给定的时间和资源下，系统对于攻击者是安全的。

一次性垫： 完美保密

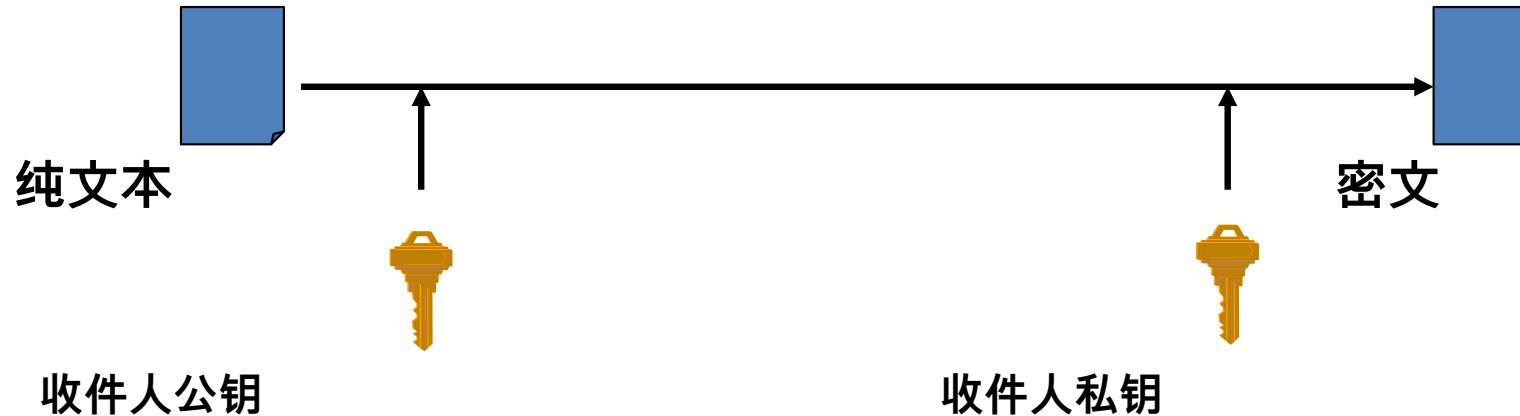
维南密码



吉尔伯特·维尔南 (Gilbert Vernam)

- 附加组操作
- 使用异或 (XOR) 实施

非对称密钥密码术

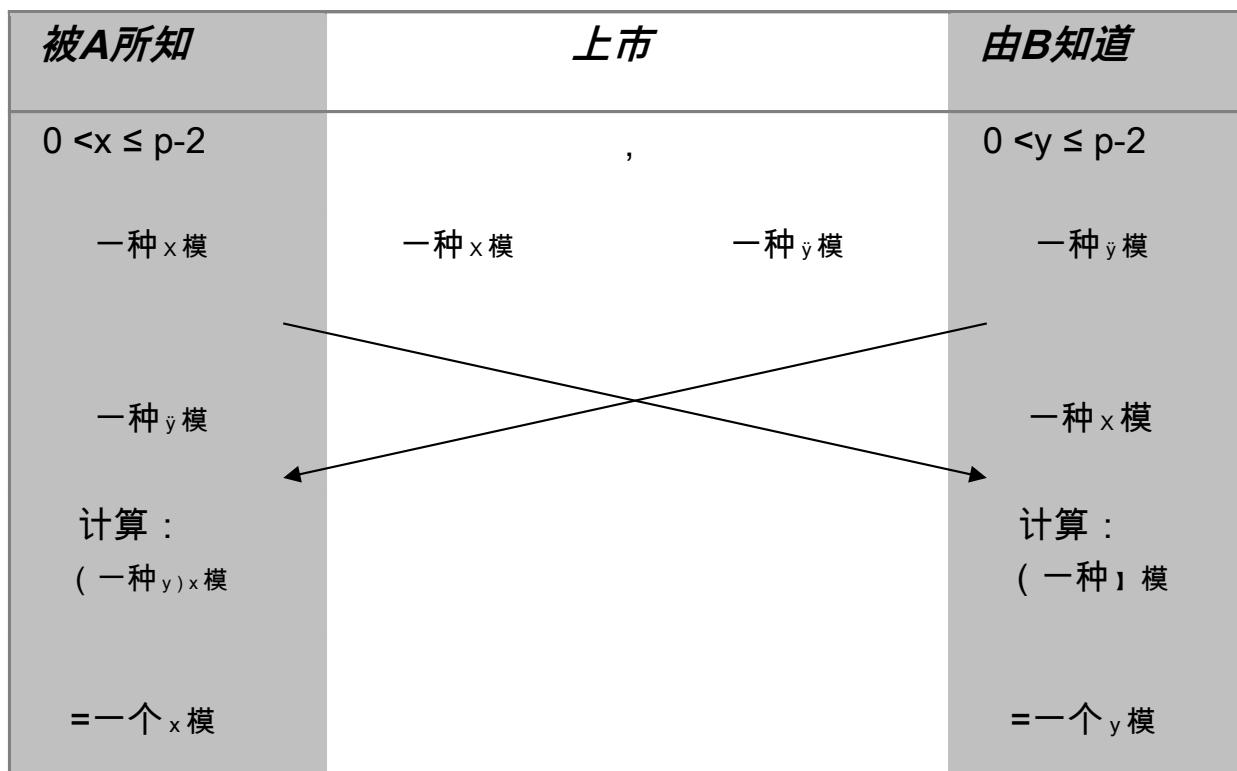


- 根据难点进行加密以找到数学问题的逆解
 - 比对称密钥加密要贵得多
- 迪菲·赫尔曼 (1977)
 - 秘密分享 : $(G_a)_b = (G_b)_a = G_b$ 模
- RSA : 里维斯特·沙米尔·阿德曼 (Rivest-Shamir-Adleman) (1978)
 - 今天的标准解决方案
 - $E_{KP}(P) = M_E$ 模式 等 $d_{KS}(C) = M_d$ 模式
 - 大号 esclésKP = (e , n) 和 KS = (d , n) 通过数学关系连接
- 椭圆曲线 : 新品种

Diffie-Hellman算法

p 是一个大素数 一种的原始元素 \mathbb{Z}^*

p



! A和B建立一个共享的秘密 ($a^y \bmod p$) 无需交换任何秘密信息

! 一种 $y \bmod p$ 可以用作对称密钥算法的秘密密钥，以便对数据进行加密

! 依靠硬度来计算离散对数

公钥加密

逆运算难的运算算法

模量：

X	1个	2	3	4	5	6
$3x$	3	9	27	81	243	729
$3x \text{ Mod } 7$	3	2	6	4	5	1个

RSA : 不可逆性

欧拉函数

对于整数n , $\phi(n)$ 是n的素数。

-如果n是素数 $\phi(n) = n-1$

-如果 $n = p \times q$ 为素数的pq

$$\phi(n) = (p-1)(q-1)$$

欧拉定理

如果a和n分别是我的素数

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

RSA为什么起作用

$$\begin{aligned} d \circ E_k(M) &= (M^e \bmod n)^d \bmod n \\ &= M \bmod n \end{aligned}$$

但是我们选择 $ed = 1 \pmod{\phi(n)}$

因此，存在一个整数j使得 $ed = j\phi(n) + 1$ $M^d \bmod n = (M^e)^d \bmod n = M \bmod n$ 根据

欧拉定理：

$$n = (1)^{\phi(n)} + 1$$

范例 (B.施耐尔)

1) 给定两个素数 $p = 47$, $q = 71$

$$n = pq = 3337$$

2) $z = (p-1) (q-1) = 46 \cdot 70 = 3220$ 让我们选择 $e = 79$ (带 n 的素数)

3) 计算 e 模 z 的逆可能的解决方案 : 欧拉定理

$$\begin{aligned} E_F(n) &= e \cdot e_{F(n)-1} \pmod{z} = 1 \pmod{z} \text{ 因此 } d = \\ e^{-1} &= E_{F(n)-1} \pmod{z} \\ \text{数值} : 79^{78} \pmod{3220} &= 1019 \end{aligned}$$

4) 加密 $M = 6882326879666683$
让我们将 M 分解为值小于 $n = 3337 \Rightarrow$ 3 位数字的块

$$M = 6882326879666683$$

让我们加密 $688 : 688^{79} \pmod{3337} = 1570$ $E(M) = 1570$
2 756 2091 2276 2423 158 让我们解密 $1570 : 1570^{1019} \pmod{3337} = 688$

RSA : 实施方面的弱点

- 切勿使用过小的值 n ,
- 切勿使用过短的私钥
- 仅使用强键，以使 $p-1$ 和 $q-1$ 具有较大的质因子
- 不要加密太短的块（总是将它们完整化为 $n-1$ 位，以破坏任何语法结构）
- 如果这些密钥可用于加密同一条消息，请不要使用这些密钥共有的因子
- 如果是私钥（ d, n ）被盗用，请勿使用其他键使用 n 作为模数
- 切勿在未经修改的情况下加密或验证来自第三方的消息（例如，添加一些随机字节）

加密协议的例子

部署

- 安全传输协议：
 - SSL / TLS (安全套接字层/传输层安全性) –独立于应用程序
 - 服务：服务器身份验证/客户端身份验证/完整性 (检查值) /机密性 (加密)



- 安全有效载荷：
 - S-HTTP : HTTP消息的单独加密
 - 服务：身份验证，完整性，机密性+数字签名 (添加不可否认性)
 - S / MIME (安全的Internet多用途扩展)

散列函数

- 散列函数 h 是与消息 $h(M)$ （也表示为 $\{M\}_H$ ）恒定（通常较短）的长度。
- 虽然 M 可以任意大，而 $\{M\}_H$ 有给定的长度

示例：文件系统哈希码，错误检测代码

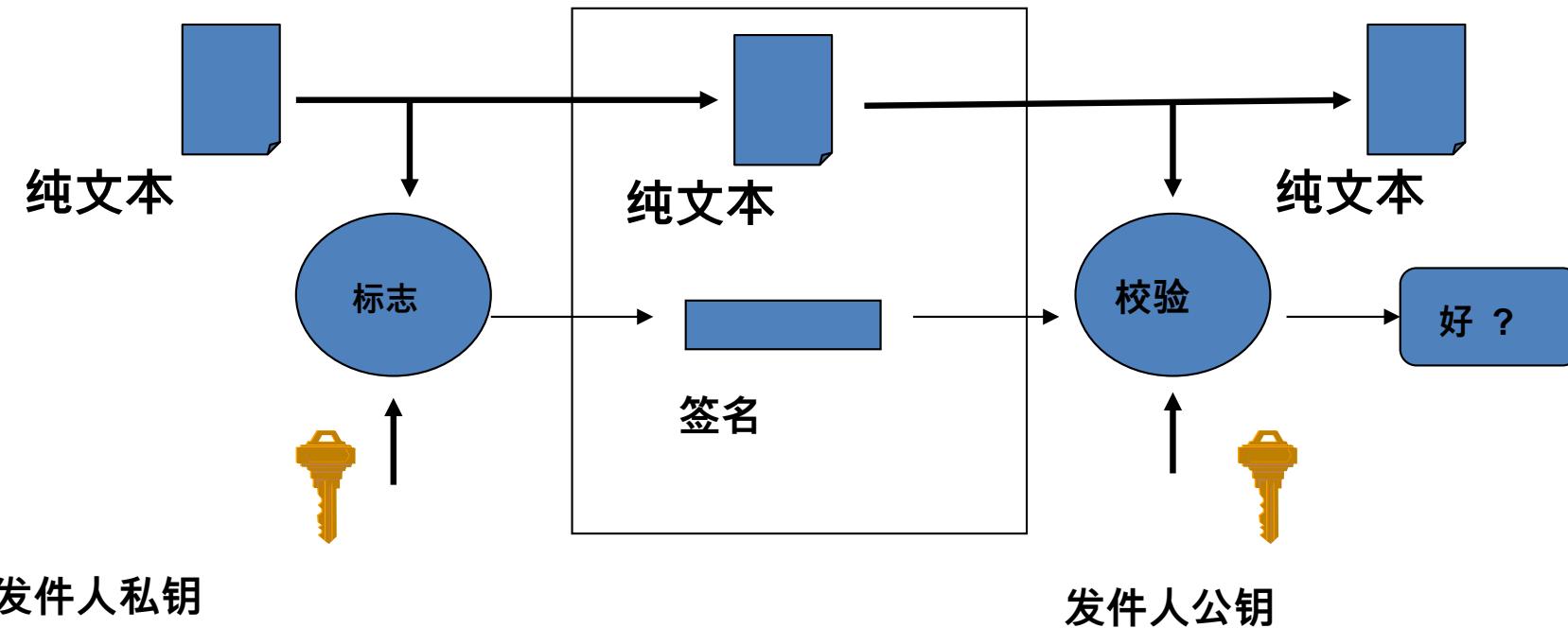
加密哈希函数

显着用法：完整性保护，密码“存储”

MD5（不要使用！），SHA-1（不要使用！），SHA-256，bcrypt，scrypt，Argon2等。

- 3个属性：
- 原像电阻：给定 K ，因此很难找到 M 使得 $h(M) = K$
- 2_{nd} 原像电阻：给定 M ，很难找到与 M 不同的 M' ，使得 $h(M) = h(M') = K$
- 无冲突：很难计算出 M 和与 M 不同的 M' ，使得 $h(M) = h(M')$
 - 生日袭击...
- 如果其计算取决于机密信息（称为MAC –消息验证码），则将其加密

电子签名



- 带有消息 (可以加密或以明文形式)
- 确保：
 - 消息来源的身份验证
 - 保护信息的完整性
 - 不可否认的消息

使用公钥算法签名

E , D : 公钥算法

生成A over消息M的签名 :

$$S = E_{KSa}(h(\text{男}))$$

验证签名 :

-计算 $h(M)$

-验证是否 $d_{千帕}(S) = h(\text{米})$

公钥基础结构 (PKI)

- 公钥证书
 - 验证名称和密钥的关联
 - 有些证书还可以引用授权 (权限)
- 当今安全和商业互联网的基础
- 对身份的保证 , 而不是信任 !
 - 信任=外部知识
- 使用示例 : Thunderbird + Enigmail

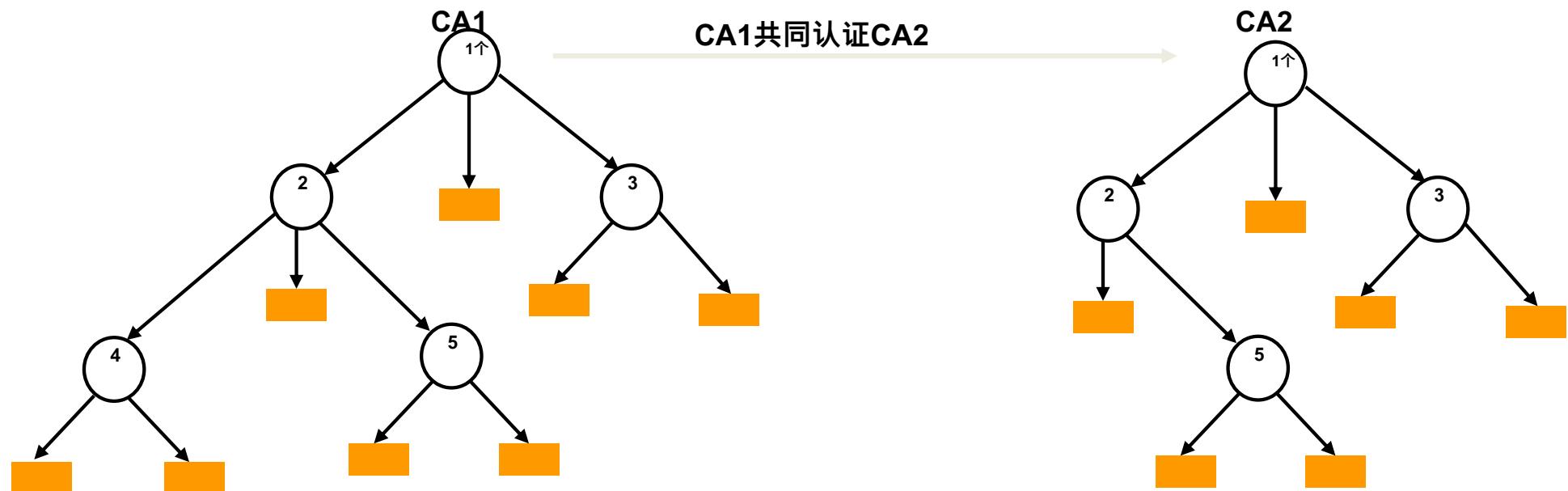


证明书

- 1978年推出[Kohnfelder]
- 证明书 '签名
 - 证书是 使用实现 签名
- 证明书 '认证/授权
 - 身份验证（授权） 可以实现证书
- X.509 –当前“标准”
 - v.1 (1988) 和v.2 –不可扩展
 - v.3 (1997) 当前标准–可选扩展
 - 许多其他提议扩展了X.509
- 其他标准
 - PGP (尤其是OpenPGP) , SPKI等

X.509中的信任模型

- 分层基础架构
- 可以在属于不同树的2个CA之间进行认证（共同认证）



对策和更多漏洞

- 对策可能导致新的漏洞
 - 例如，如果我们只允许三个错误的登录名，以作为对付蛮力攻击（帐户被冻结）的对策，我们会引入哪个新漏洞？
 - 拒绝服务攻击
 - 如果对策依赖新软件，则该新软件中的错误可能意味着
 - 无效，或
 - 更糟糕的是，它引入了更多的弱点
 - 例如，Witty蠕虫出现于2004年3月，它利用ISS安全软件
 - http://en.wikipedia.org/wiki/Witty_%28computer_worm%29

注意事项：加密中的不安全性

- SSH旨在提供安全性，即防范网络窃听的对策，但它是新漏洞的来源
- 密码术不是这些漏洞的原因，并且不能解决/防止这些漏洞
 - 协议，实施错误（例如，WiFi中的WEP）
 - 编程错误（缓冲区溢出）
 - 分发错误（木马）
- 布鲁斯·施耐尔（Bruce Schneier）：“目前，加密是我们拥有的最强大的链接。其他一切都更糟：软件，网络，人员。保持最牢固的联系并使之更加牢固绝对没有任何价值”

认证/识别/
平台完整性

认证方式

- 我所知道的：密码，秘密问题，秘密密钥
- 我拥有的：智能卡，银行卡，RSA密钥，手机
- 我是什么：生物识别-生理学（指纹，虹膜，静脉，面部，声音等）或行为（签名，手势等）
- 越来越多的两个机制的关联
 - 来自不同类别的两因素身份验证

密码

- 严格个人
- 很难找到，容易记住（无需写下来）
 - 最小字符数（包括; / ! %....）
 - 与字典单词不对应
- 避免
 - 亲戚的名字或名字
 - 电话号码...
- 必须定期更换
- 关键字法
 - 说，电影标题...。

FIGURE 11.15 • Passwords, weakest to strongest

Strong passwords use words that are unrelated to your interests and include upper- and lower-case letters, numbers, and symbols.

Weakest
John
Kelley
JohnnieD
yankees
Heresjohnnie
nycoolboy
NYcoolboy#1
Hypertree
nyKOOLB@Y
Hyper#tree9
re@Lpharm#
92Tpo5#cCw
Strongest

Most effective

智能卡

- 智能卡 , IC卡 , 芯片卡 , 密码卡...
- 主要特点 :
 - 便携式对象存储数据和/或过程。
 - 安全对象
 - 防止读取存储在卡存储器中的数据 (秘密密钥...)
 - 在可信空间中执行的代码
 - 可为数亿用户定制的低成本对象。
 - SPOM 1-5 \$ /磁卡0-1,5 \$
 - 不能独自工作 , 需要
 - 读卡器提供能量
 - 一个钟
 - 通讯链接

卡及其技术功能

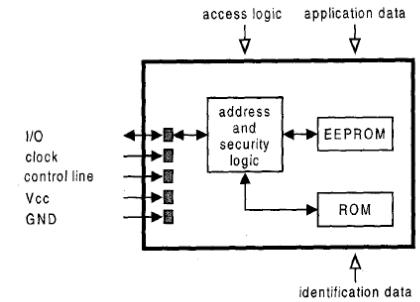
- 存储卡：

- 简单的存储器（读/写）（EPROM / EEPROM）
- 不规范



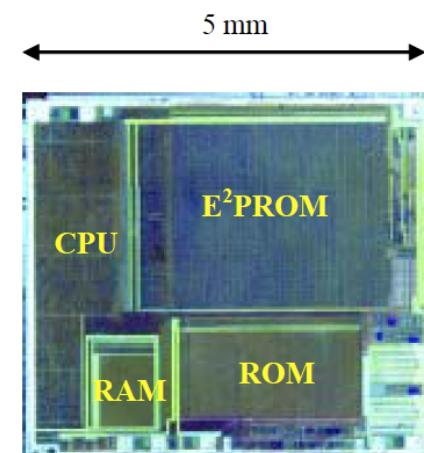
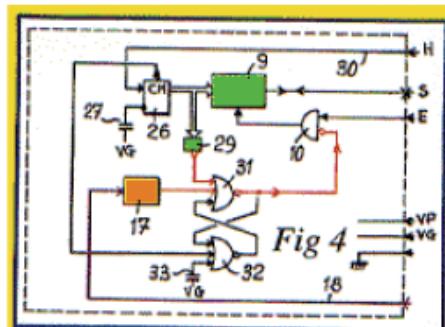
- 有线逻辑卡：

- 该卡包含用于保护数据的有线设备
- 专用电子设备连接输入和输出引脚
- 不规范



- 微处理器卡（SPOM / MAM）：

- 内存+处理器 一种 可编程的
- 安全算法（例如：DES，RSA）
- ISO 7816标准
- 接触式或非接触式卡
- 1个 ST 实施 Bull CP8 : 36b RAM , 1 Kb EPROM , 1.6 Kb ROM



生物识别

- 目标：
 - 用户不承担失去身份验证机制的风险
 - 某些身份验证模式非常完整-例如：指纹 (iPAQ , iPhone 5s)



- 问题：
 - 生物特征因素盗窃：切勿使用一因素识别！
 - 有时很容易造假
(指纹...)
 - 数据应在其所有者的控制下保存 (例如：智能卡)

FIGURE 11.17 • Facial recognition

By taking measurements of 128 facial features and matching them to the measurements of known faces, biometric software can help identify people.



软件平台安全性：Java

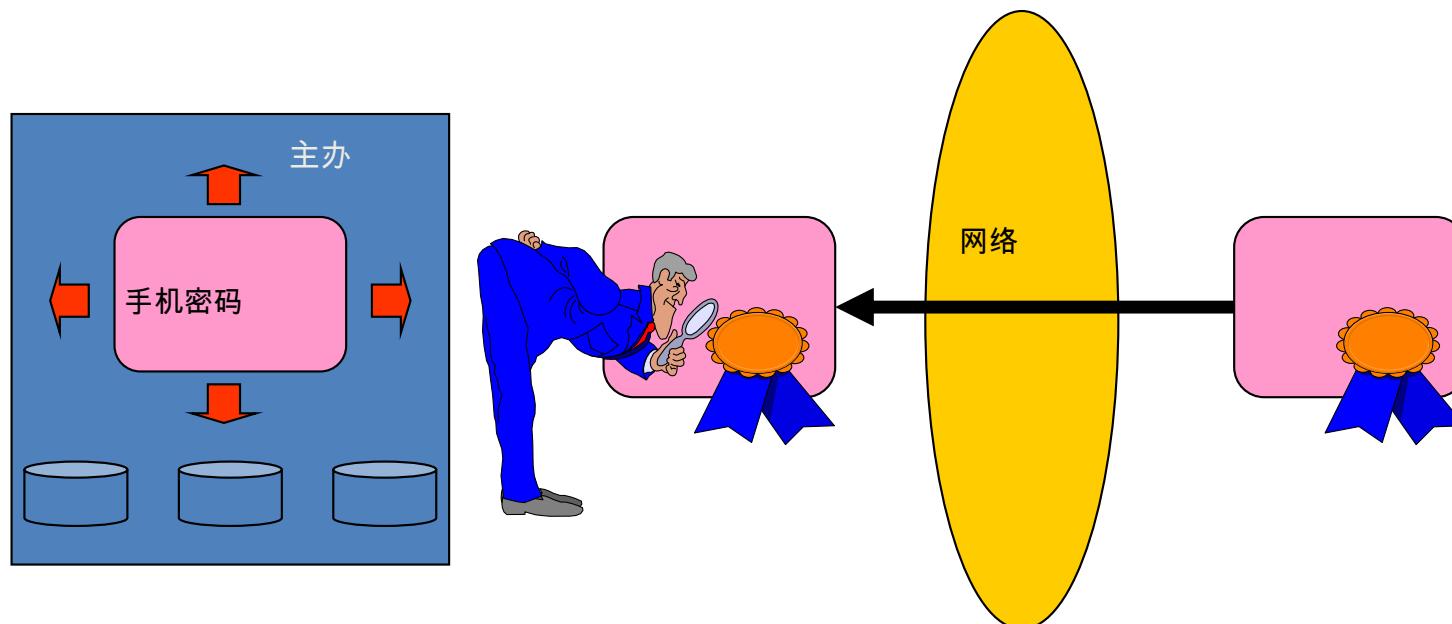
- 主机代码和移动代码具有独立的身份
 - 验证移动代码的来源 _____
- 移动代码通过网络公开
 - 验证移动代码的完整性 _____
- 移动代码由另一方生成
 - 访问控制 _____
 - 语义验证 _____

Java : 安全管理器

- 目的：
 - 建立自定义安全策略
 - 定义沙箱的边界
- 安装应用程序安全管理器：
 - 负责应用程序的生命周期
 - 不可更换
 - 政策可以取决于其他地方收集的信息
 - 例如类加载器
- 要自定义沙箱：
 - 扩展java.lang.SecurityManager类
 - 覆盖授权资源访问的方法

身份验证/完整性-代码 签收

通过数字签名实现强大的身份验证和代码完整性



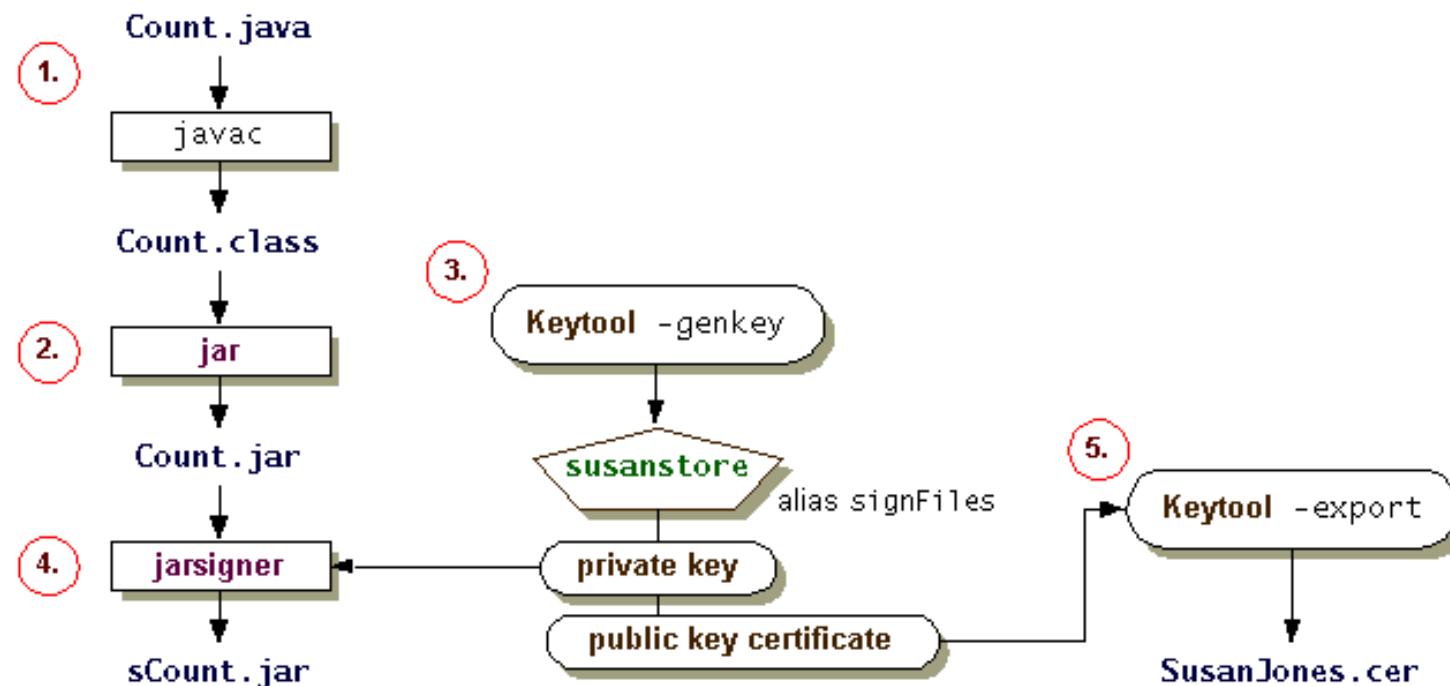
问题：对移动代码操作不负责任

Java签名小程序

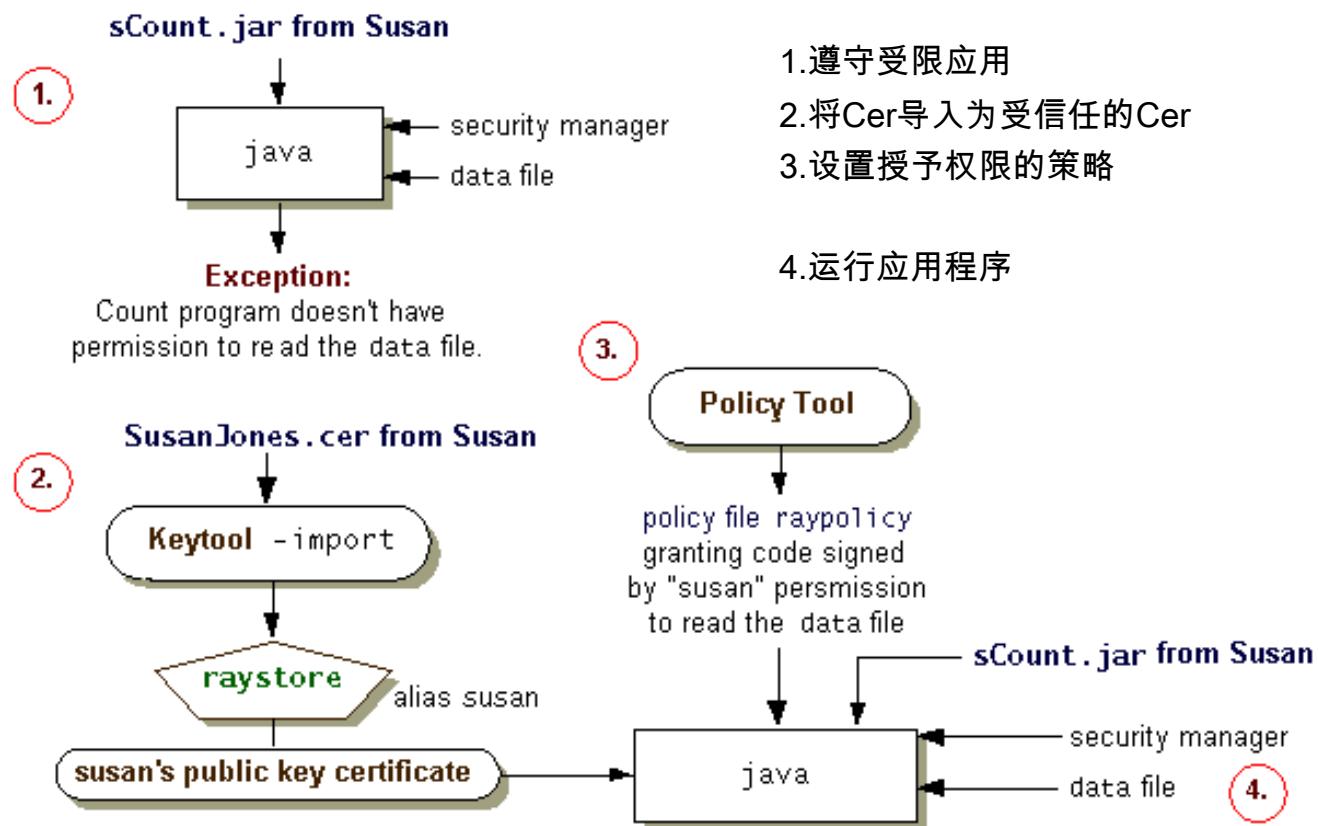
- 从JDK 1.1开始可用
- 类似于已签名的ActiveX组件
- 政策：
 - 小程序已验证 ↳ 完全访问本地资源
 - 没有证书 ↳ 小程序是 沙盒
- 存档 (JAR) 文件包含 [applet (s) + 证书]

签名代码

- 生成密钥 : (3) **关键工具** –genkey –aliassignedFiles –keypass kpi135 –keystore susanstore –storepass ab987c
- (4) **jarsigner** –keystore susanstore –signedjar sCount.jar Count.jar signFiles
- (5) **键盘工具** –export –keystore susanstore –alias signFiles –file SusanJones.cer

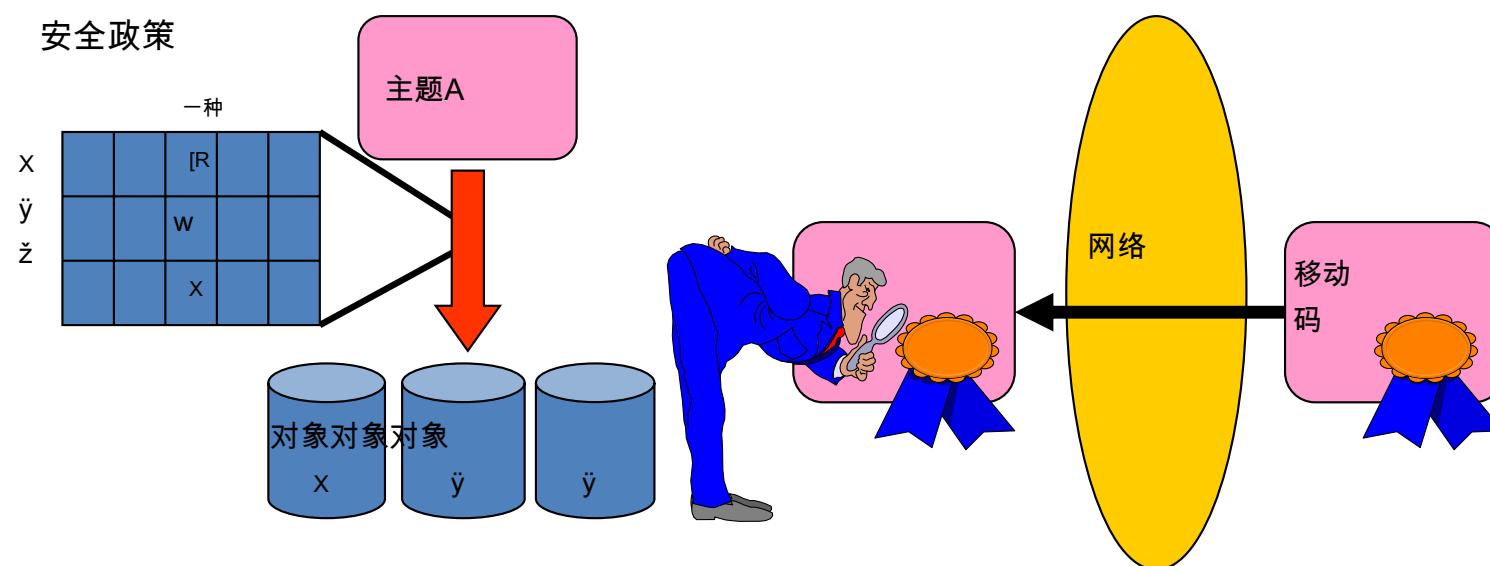


关于接收签名代码

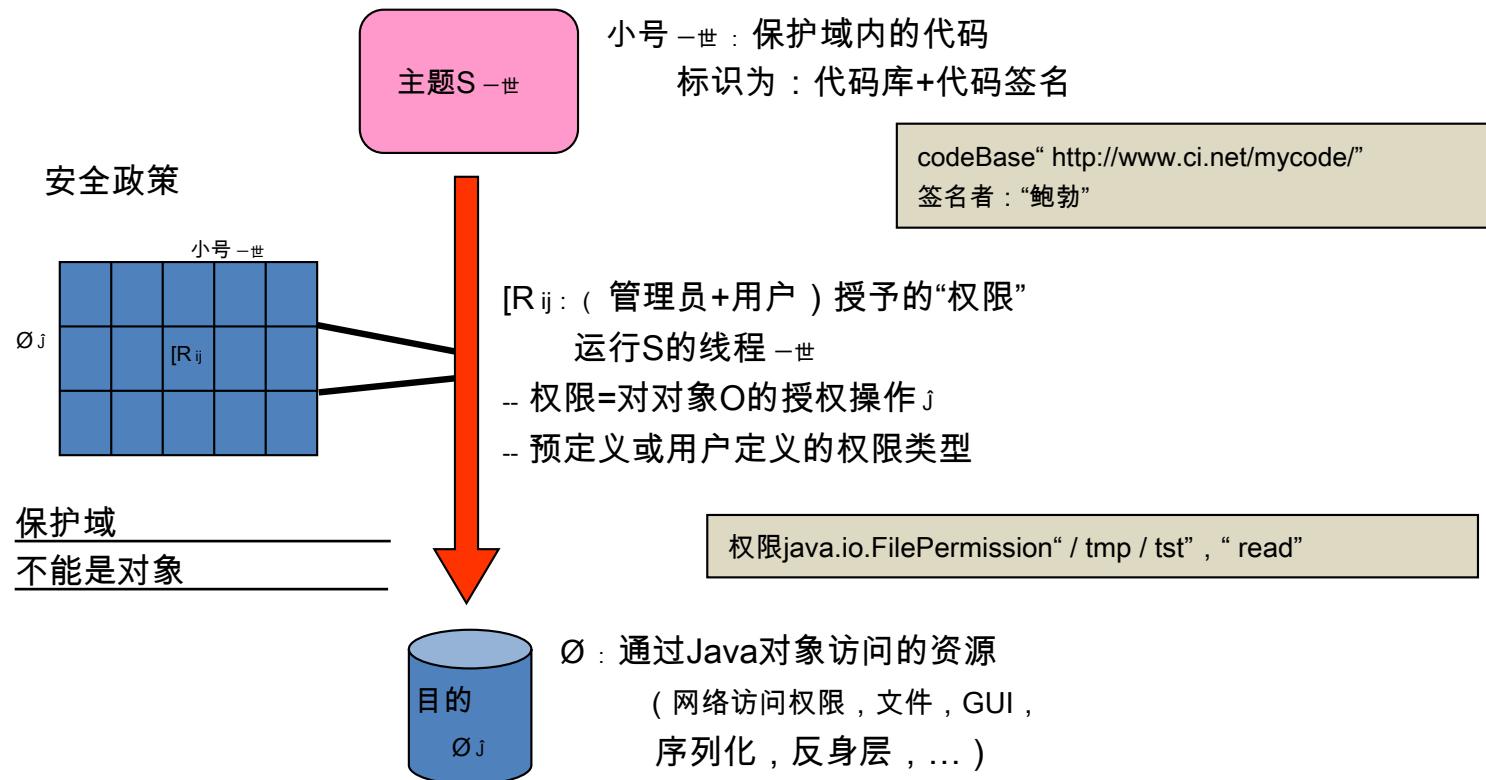


Java访问控制-安全策略

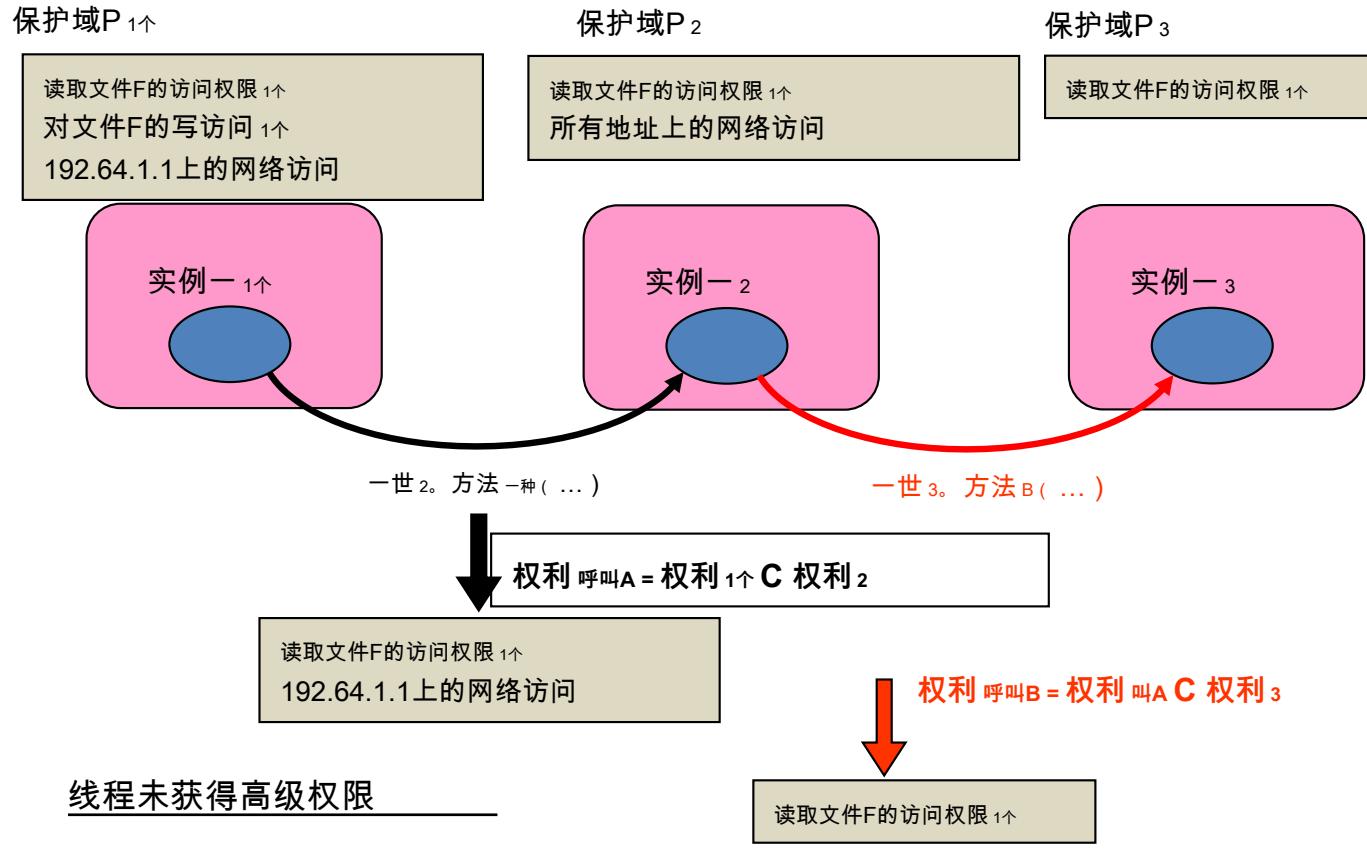
基于可变安全策略的访问控制



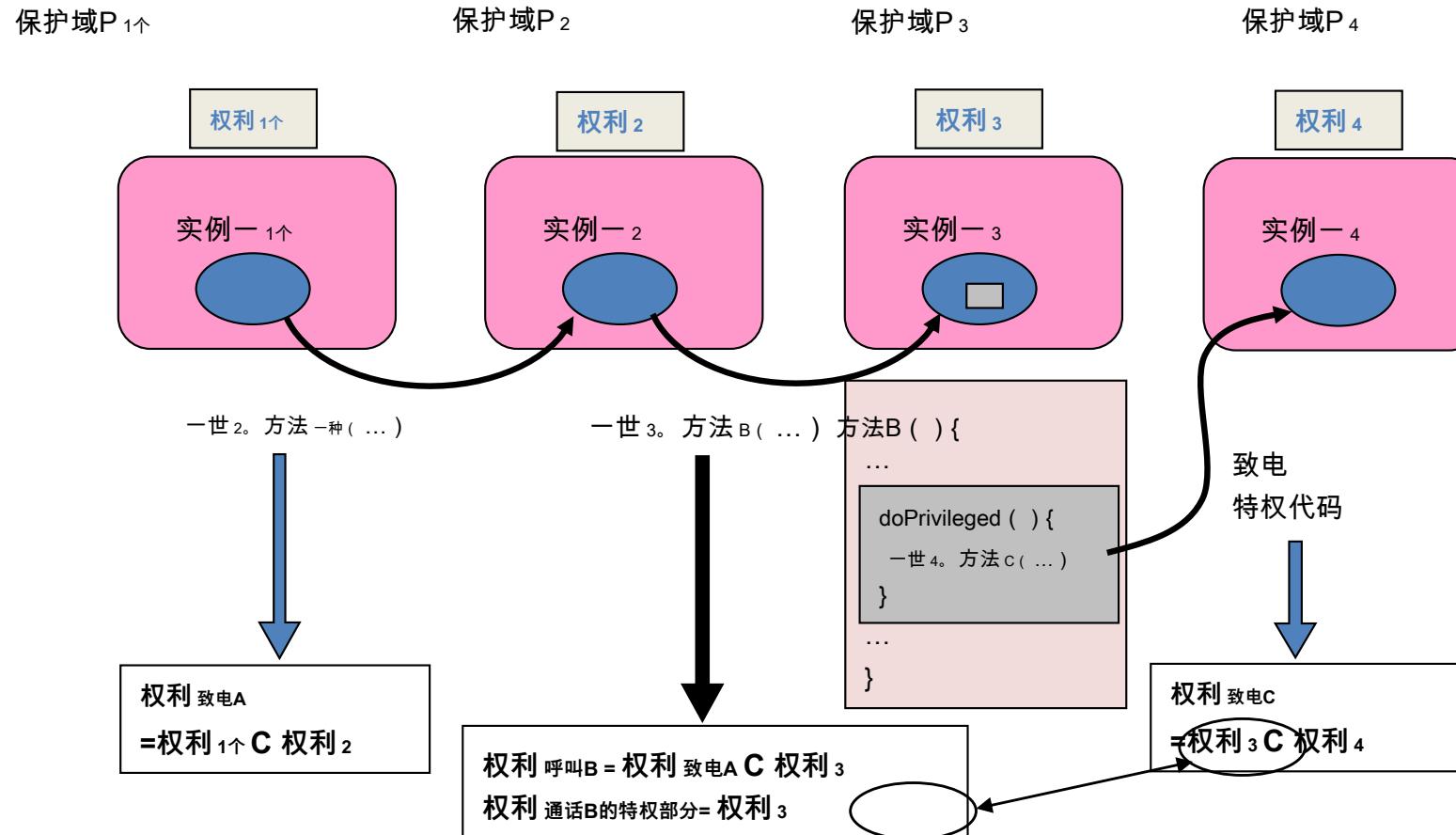
Java访问控制模型



Java : 跨保护域的权利



Java : 特权代码 (POLP)

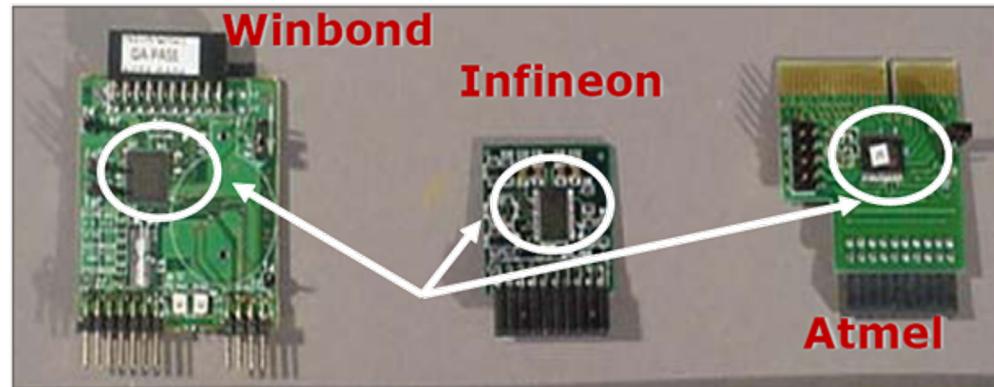


硬件平台完整性： 可信计算

- TCG财团，成立于1999年
 - 主要成员（超过200名）：
 - AMD , HP , IBM , Infineon , Intel , Lenovo , Microsoft , Sun
- 体系结构已被其他供应商采用
 - ARM TrustZone , Apple Secure Enclave , Google Titan-M
- 硬件保护（加密）存储：
 - 只有“授权”软件才能解密数据
 - 例如，保护用于解密文件系统的密钥
- 安全启动：“授权”软件的方法
- 证明：向远程服务器证明我的计算机上正在运行什么软件

建筑

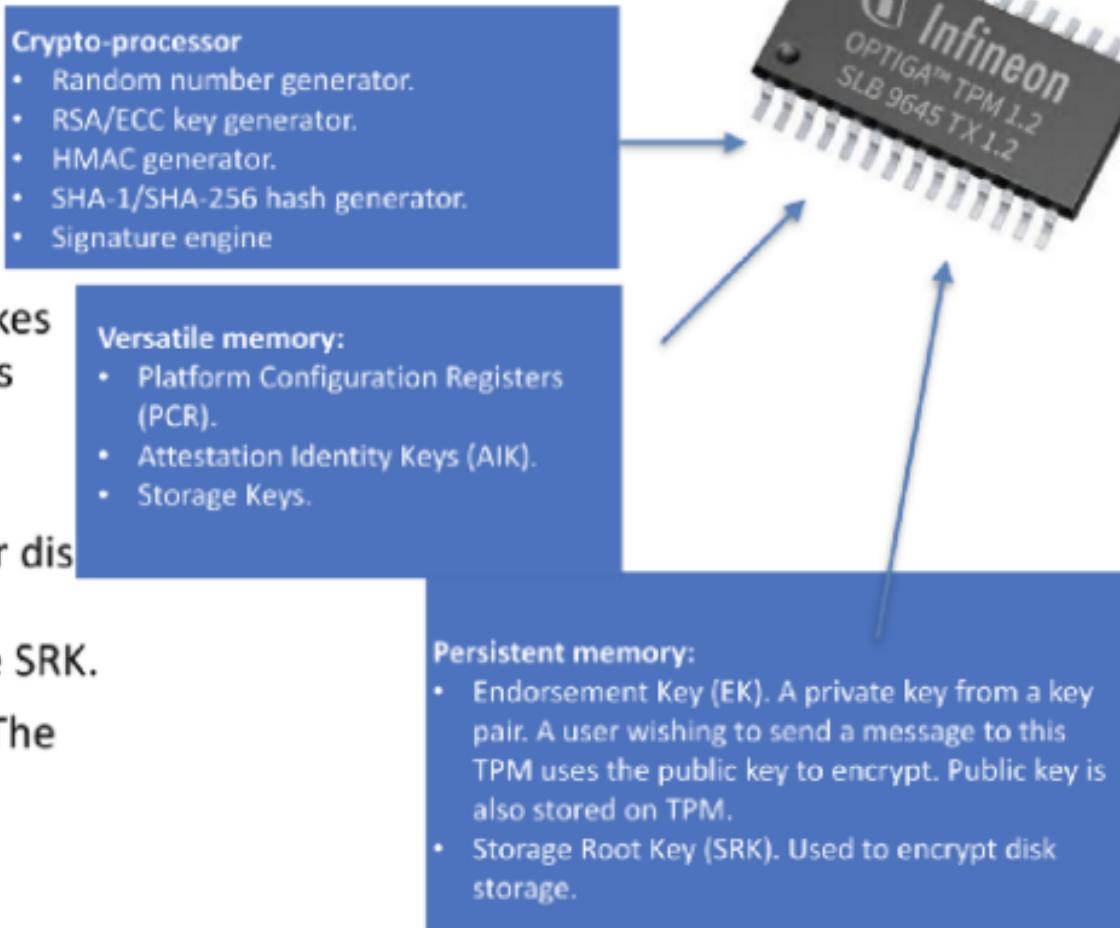
- 硬件协处理器：
 - TCG的TPM芯片（受信任的PlatformModule）
 - 苹果T2
 - Google Titan-M
 - 笔记本电脑，手机.....
- 软件：
 - 对BIOS，OS，应用程序的修改
 - 通常用于文件/磁盘加密（例如Bitlocker）
 - 客户端单点登录链接到工作站
 - 在进行批评操作（登录，签名）之前/一起进行安全的启动/平台认证



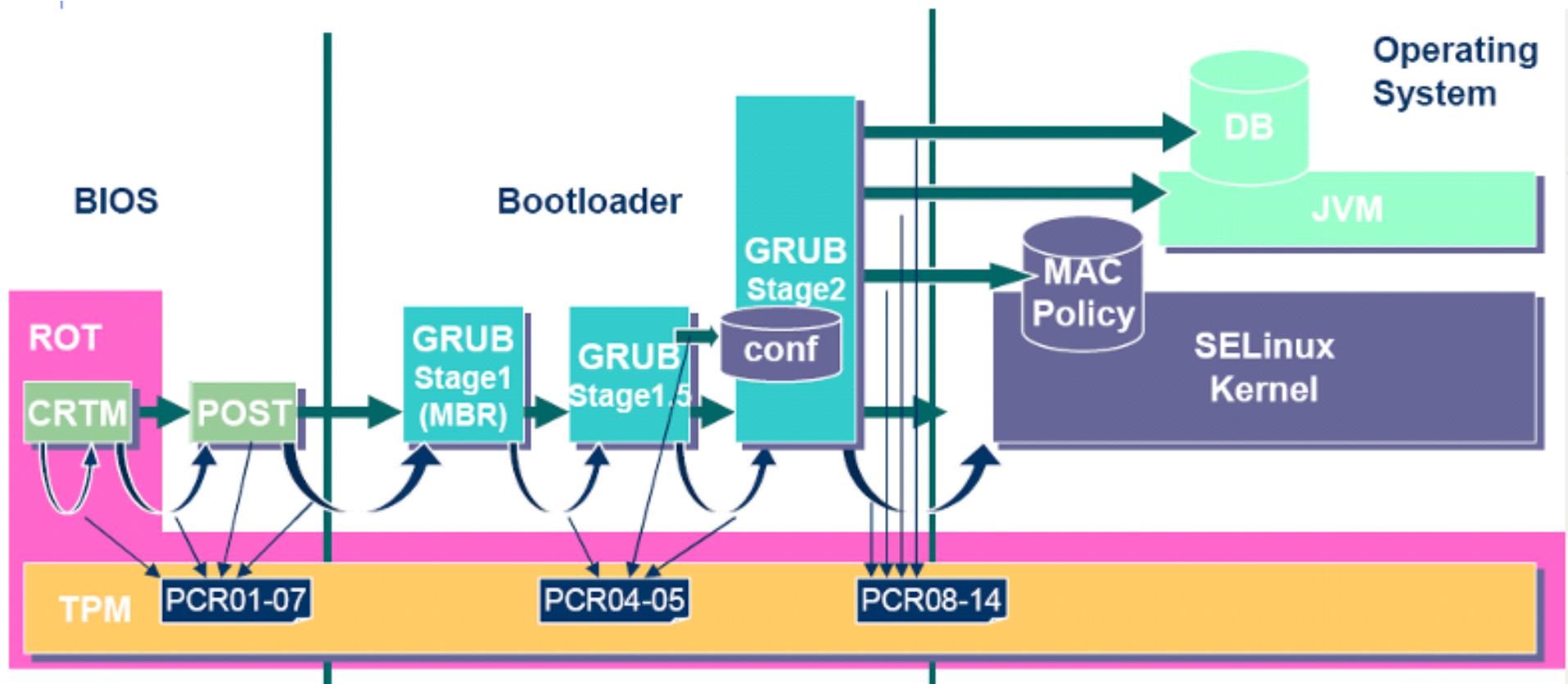
硬件安全模块

TPM 1.2 and 2.0

- **Platform integrity.** Makes sure the boot process is correct.
- **Disk encryption.** The encryption key used for disk encryption is fully or partially defined by the SRK.
- **Password protection.** The storage of the user's password in the chip.



安全启动示例：Trusted Grub (IBM'05)



可信计算：结论

- 今天广泛使用
- 用于访问证明功能和受保护的存储的API：扩展了软件功能
- 但是可能发生攻击：
 - 错误的验证或证明流程（例如CVE-2018-6622，CVE- 2017-16837）
 - 硬件攻击：例如，Bitlocker密钥嗅探（<https://pulsesecurity.co.nz/articles/TPM-嗅探>）

