信息安全与密码技术

目录

1. 概述

信息安全是软件开发中的一个重要的概念，是每一个商业软件都要重点思考的问题。区块链技术主要探讨的也是安全和信任问题，本文旨在对密码学的相关算法作简单介绍，供大家对密码学体系有一个大致的印象，开发中可根据实际需求搜索相关算法的实现。

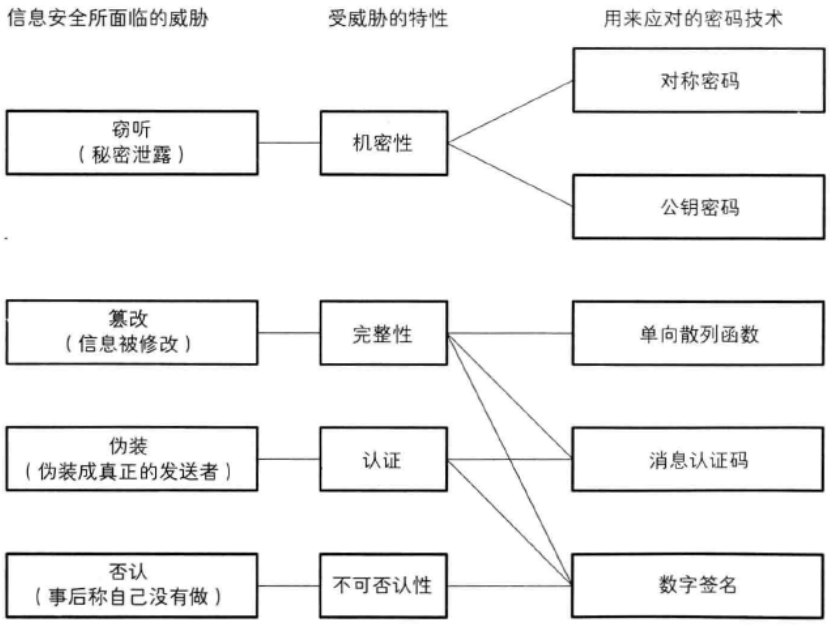
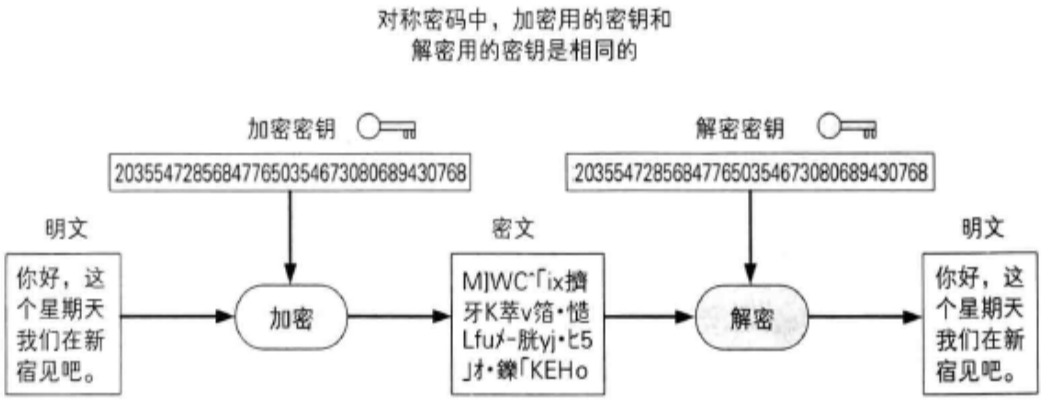


图 1 信息安全所面临的威胁与应对这些威胁的技术

2. 算法工具箱

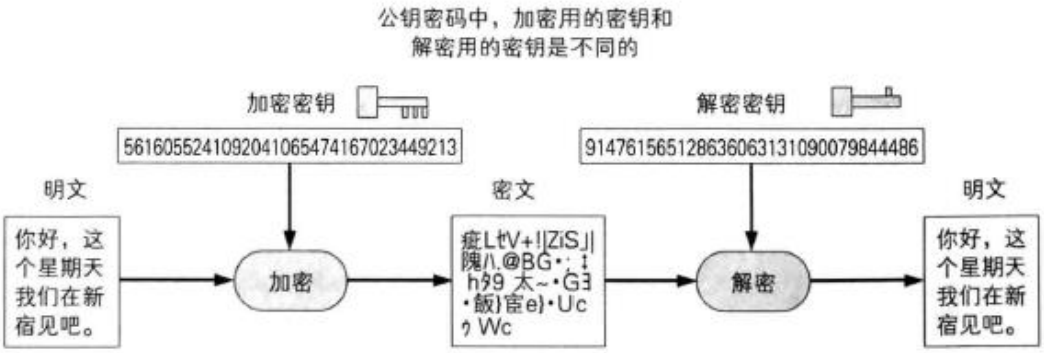
2.1. 对称算法

加密和解密密钥相同，常用算法如 DES、TDES、AES、SM4 等



2.2. 非对称算法

加密和解密密钥不相同，常用算法如 RSA、ECC、SM2 等。



密码技术所提供的并不仅仅是基于密码的机密性，用于检验消息是否被篡改的完整性，以及用于确认对方是否是本人的认证等都是密码技术的重要部分。

2.3. 数据校验算法

数据校验是为保证数据的完整性，用一种指定的算法对原始数据计算出的一个校验值。接收方用同样的算法计算一次校验值，如果和接受数据提供的校验值一样，说明数据是完整的。常见的算法有

- 奇偶校验（Parity Check）
- BCC 异或校验法(block check character，块校验码)
- LRC 纵向冗余校验（Longitudinal Redundancy Check）
- CRC（Cyclic Redundancy Check，循环冗余校验）
- MD5、SHA、MAC 等散列摘要算法。

2.4. 消息认证码 MAC

消息认证码不但能够保证数据完整性，还能提供认证机制（确认消息来自期待的对象）。常用的算法如 HMAC 算法

2.5. 数字签名

保证数据完整性、提供认证、防止否认的密码技术。常见如 RSA 签名校验

2.6. 分散算法

密钥分算算法简称 Diversify，是指将一个双长度(一个长度密钥为 8 个字节)的主密钥(MK)，对数据进行分散处理，推导出一个双长度的 DES 加密密钥(DK)。

该算法广泛应用于现在的金融 IC 卡和其他对于安全要求高的行业。

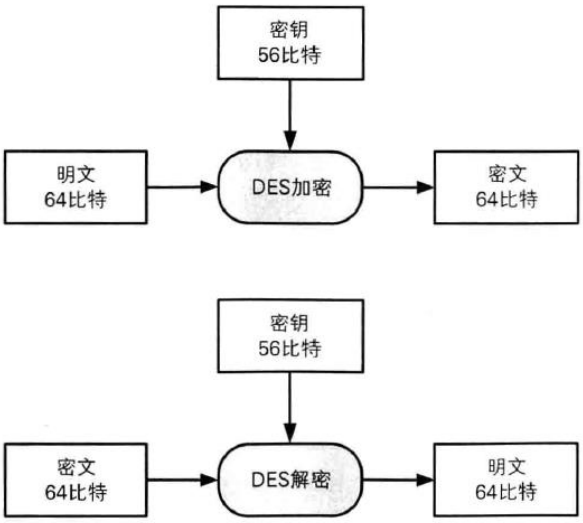
3. 对称算法

3.1. DES 算法

DES 是一种将 64bit 的明文加密成 64bit 的密文的对称密码算法，它的密钥长度是 56bit。尽管从规格上来说，DES 的密钥长度是 64bit，但由于每隔 7bit 会设置一个用于错误校验的 bit，因此实质上其密钥长度是 56bit。

DES 是以 64bit 明文为一个单位来进行加密的，这个 64bit 的单位成为分组。一般来说，以分组为单位进行处理的密码算法成为分组密码（block cipher），DES 就是分组密码的一种。

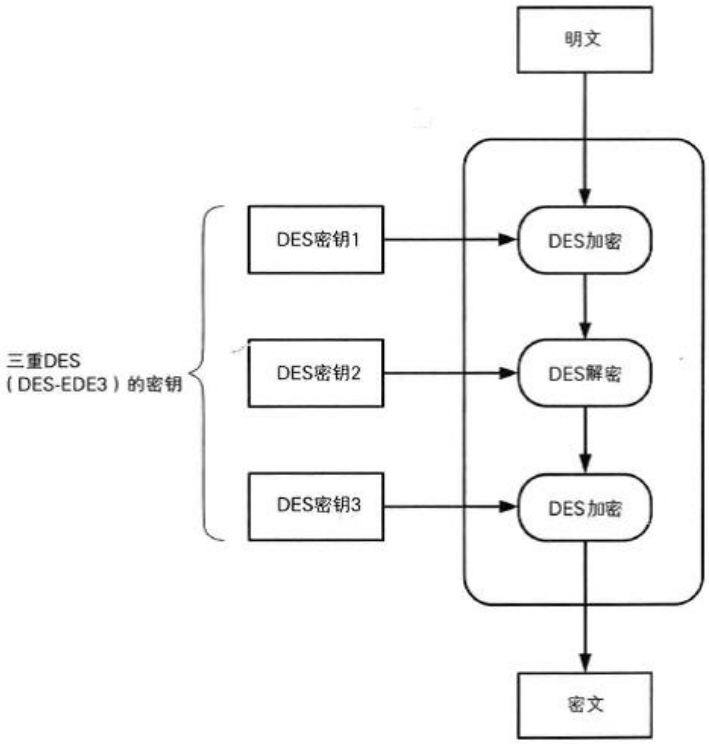
DES 每次加密只能加密 64bit 的数据，如果要加密的明文比较长，就需要对 DES 加密进行迭代，而迭代的具体方式就称为模式（mode）。关于模式在第四章节描述。



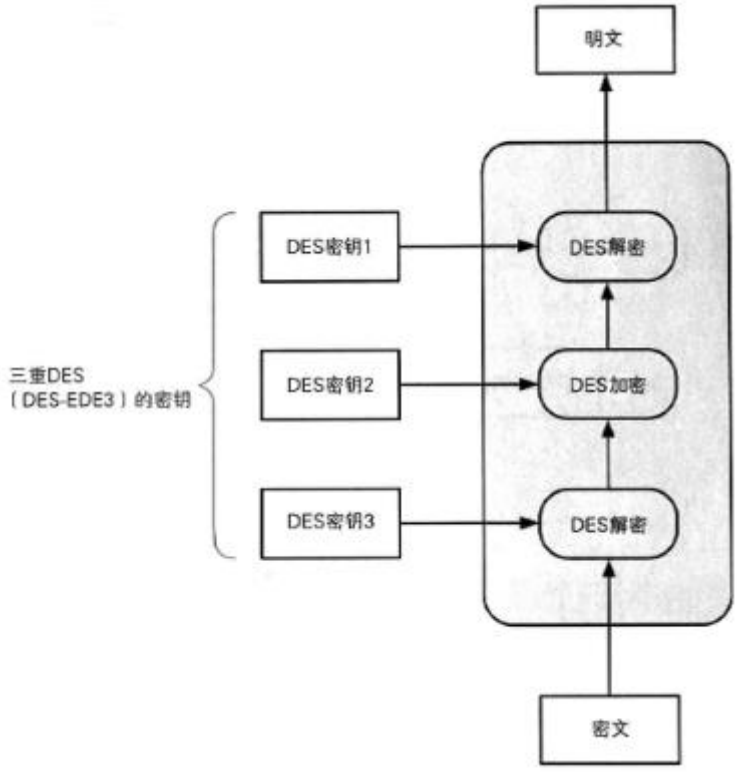
DES 加解密过程

3.2. TDES 算法

DES 现在可以在现实的时间内暴力破解，因此出现了 TDES。TDES（Triple-DES）是为了增加 DES 的强度，将 DES 重复 3 次所得到的一种密码算法，通常缩写为 3DES。尽管 TDES 目前还被银行等机构使用，但其处理速度不高，而且在安全性方面也逐渐显现出一些问题。



TDES 加密过程



TDES 解密过程

3.3. AES 算法

AES（Advanced Encryption Standard）是取代前任标准 DES 而成为新标准的一种对称密码算法。全世界企业和密码学家提交了多个对称密码算法作为 AES 候选，最终 2000 年从这些候选算法汇总选出了一种名为 Rijndael 的对称密码算法。AES 的选定过程中并不仅仅考虑一种算法是否存在弱点，算法的速度、实现的容易性等也都在考虑范围内。

Rijndael 是由比利时密码学家 Joan Daemen 和 Vincent Rijmen 设计的分组密码算法。分组长度为 128bit，AES 规格中，密钥长度只有 128/192/256bit 三种。

4. 分组密码模式

分组密码（block cipher）是每次只能处理特定长度的一块数据的一类密码算法，这里的一块就是分组。一个分组的比特数就是分组长度（block length）。例如，DES 和 TDES 的分组长度都是 64bit，一次只能加密 64bit 长度的明文，并生成 64 比特的密文。AES 分组长度是 128bit、192bit 和 256bit。

流密码（stream cipher）是对数据流进行连续处理的一类密码算法。流密码一般以 1 比特、8 比特或 32 比特为单位进行加密和解密。

分组密码处理完一个分组就结束了，不需要通过内部状态来记录加密的进度；相对的，流密码时对一串数据流进行连续处理，因此需要保持内部状态。

分组密码算法只能加密固定长度的分组，但是我们需要加密的明文长度可能会超过分组密码的分组长度，这时就需要对分组密码算法进行迭代，以便将一段很长的明文全部加密。而迭代的方法就是分组密码的模式（mode）。

模式有很多种，分组密码的主要模式有以下 5 中：

ECB 模式：Electronic Code Book mode 电子密码本模式

CBC 模式：Cipher Block Chaining mode 密码分组链接模式

CFB 模式：Cipher Feedback mode 密文反馈模式

OFB 模式：Output FeedBack mode 输出反馈模式

CTR 模式：Counter mode 计数模式

模式	名称	优点	缺点	备注
ECB 模式	Electronic CodeBook 电子密码本模式	• 简单 • 快速 • 支持并行计算（加密、解密）	• 明文中的重复排列会反映在密文中 • 通过删除、替换密文分组可以对明文进行操作 • 对包含某些比特错误的密文进行解密时，对应的分组会出错 • 不能抵御重放攻击	不应使用
CBC 模式	Cipher Block Chaining 密文分组链接模式	• 明文的重复排列不会反映在密文中 • 支持并行计算（仅解密） • 能够解密任意密文分组	• 对包含某些错误比特的密文进行解密时，第一个分组的全部比特以及后一个分组的相应比特会出错 • 加密不支持并行计算	推荐使用
CFB 模式	Cipher-FeedBack 密文反馈模式	• 不需要填充（padding） • 支持并行计算（仅解密） • 能够解密任意密文分组	• 加密不支持并行计算 • 对包含某些错误比特的密文进行解密时，第一个分组的全部比特以及后一个分组的相应比特会出错 • 不能抵御重放攻击	• 现在已不使用 • 推荐用 CTR 模式代替

模式	名称	优点	缺点	备注
OFB 模式	Output-FeedBack 输出反馈模式	• 不需要填充（padding） • 可事先进行加密、解密的准备 • 加密、解密使用相同结构 • 对包含某些错误比特的密文进行解密时，只有明文中相对应的比特会出错	• 不支持并行计算 • 主动攻击者反转密文分组中的某些比特时，明文分组中相对应的比特也会被反转	推荐用 CTR 模式代替
CTR 模式	CounTeR 计数器模式	• 不需要填充（padding） • 可事先进行加密、解密的准备 • 加密、解密使用相同结构 • 对包含某些错误比特的密文进行解密时，只有明文中相对应的比特会出错 • 支持并行计算（加密、解密）	主动攻击者反转密文分组中的某些比特时，明文分组中相对应的比特也会被反转	推荐使用

5. 非对称算法

5.1. RSA 算法

RSA 是一种公钥密码算法，它的名字是由它的三位开发者，即 Ron Rivest、Adi Shamir 和 Leonard Adleman 的姓氏的首字母组成的。1983 年，RSA 公司为 RSA 算法在美国申请取得专利，现在已过期。

密文 = 明文 E mod N　　（RSA 加密）

也就是说，RSA 的密文是对代表明文的数字的 E 次方求 mod N 的结果。换句话说，就是将明文和自己做 E 次乘法，然后将其结果除以 N 求余数，这个余数就是密文。

RSA 加密公式

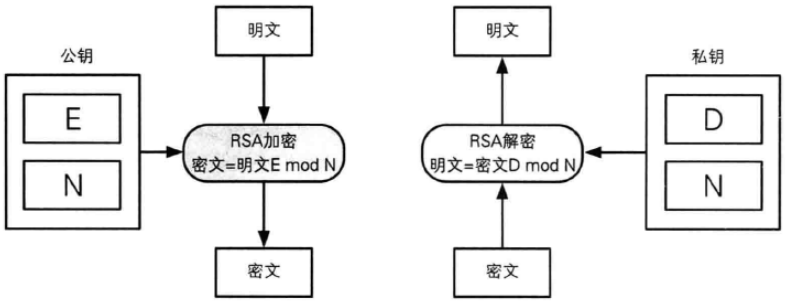
E 和 N 的组合就是公钥。

明文 = 密文 D mod N　　（RSA 解密）

也就是说，对表示密文的数字的 D 次方求 mod N 就可以得到明文。换句话说，将密文和自己做 D 次乘法，再对其结果除以 N 求余数，就可以得到明文。

RSA 解密公式

D 和 N 的组合就是私钥。



RSA 的加密和解密

跟 DES，AES 一样, RSA 也是一个块加密算法（ block cipher algorithm），总是在一个固定长度的块上进行操作。但跟 AES 等不同的是，block length 是跟 key length 以及所使用的填充模式有关的。

RSA 常用的填充模式：

- RSA_PKCS1_PADDING
- RSA_PKCS1_OAEP_PADDING
- RSA_NO_PADDING

5.2. ECC 算法

椭圆曲线密码（Elliptic Curve Cryptosystems,ECC）是最近备受关注的一种公钥密码算法。特点是所需密钥长度比 RSA 短。

6. 数据校验算法

6.1. 奇偶校验

在数据存储和传输中，字节中额外增加一个比特位，用来检验错误。校验位可以通过数据位异或计算出来。

6.2. BCC 异或校验法

很多基于串口的通讯都用这种既简单又相当准确的方法。它就是把所有数据都和一个指定的初始值（通常是 0）异或一次，最后的结果就是校验值，通常把它附在通讯数据的最后一起发送出去。接收方收到数据后自己也计算一次异或和校验值，如果和收到的校验值一致就说明收到的数据是完整的。

6.3. LRC 纵向冗余校验

实现方式：将 ASCII 码帧中的头和尾去掉，将串中的每个字节变成 16 进制相加，再将结果取反加 1（补码），就是 VRC（vertical redundant code，垂直冗余码）了。

6.4. CRC 循环冗余校验

它是利用除法及余数的原理来作错误侦测（Error Detecting）的。实际应用时，发送装置计算出 CRC 值并随数据一同发送给接收装置，接收装置对收到的数据重新计算 CRC 并与收到的 CRC 相比较，若两个 CRC 值不同，则说明数据通讯出现错误。

根据应用环境与习惯的不同，CRC 又可分为以下几种标准：

CRC-12 码；

CRC-16 码；

CRC-CCITT 码；

CRC-32 码。

6.5. 单向散列函数

功能在于获取消息的指纹。散列值的长度和消息的长度无关。单向散列函数的性质：

- 根据任意长度的消息计算出固定长度的散列值；
- 能够快速计算出散列值；
- 消息不同散列值也不同；
- 具有单向性。

实际应用场合：

- 检测软件是否被篡改；
- 基于口令的加密，将口令和盐混合后计算散列值，散列值作为密钥，可防字典攻击；
- 消息认证码，将共享密钥和消息混合后计算出散列值；
- 数字签名，对散列值施加数字签名；
- 伪随机数生成器；
- 一次性口令。

MD4/MD5:消息摘要 Message Digest 的缩写，可产生 128bit 的散列值，抗碰撞性已被攻破，不再安全。

SHA1/SHA256/SHA512:哈希算法 Secure Hash Algorithm 的缩写，SHA1 产生 160bit 的散列值，抗碰撞性已被攻克。SHA256 产生 256bit 散列值，尚未被攻克。

7. 消息认证码

消息认证码（message authentication code）是一种确认完整性并进行认证的技术。输入包括任意长度的消息和一个发送者与接受者之间共享的密钥，它可以

输出固定长度的数据，这个数据称为 MAC 值。

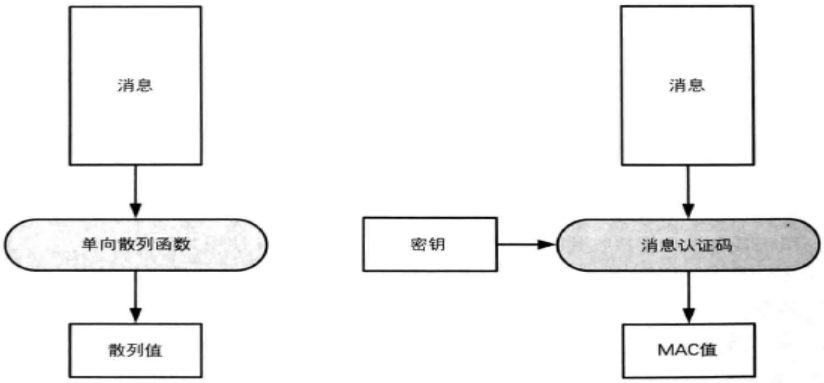


图 8-1　单向散列函数与消息认证码的比较

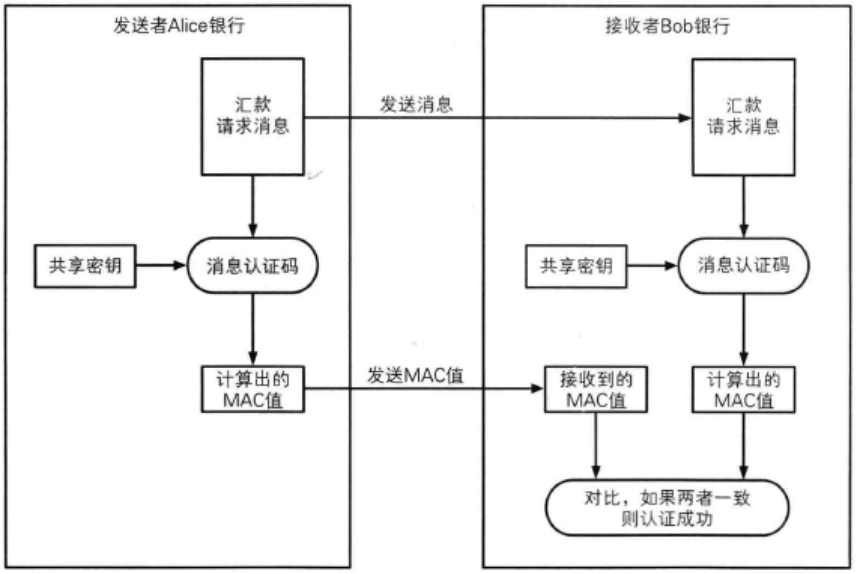


图 8-2　消息认证码的使用步骤

7.1. 消息认证码实现方法

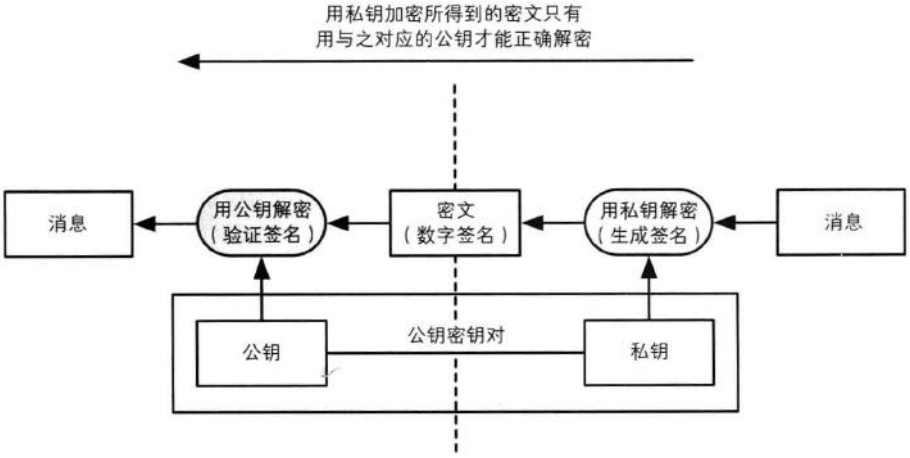
单向散列函数实现，其中一种方法称为 HMAC。

7.2. 分组密码实现

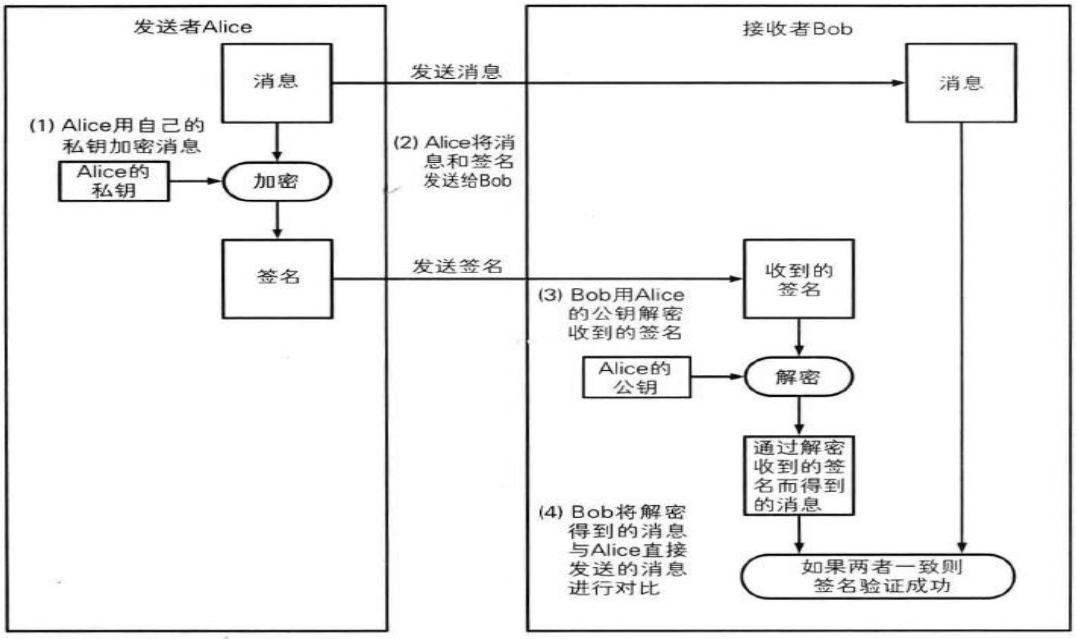
- MAC-ECB
- MAC-PBOC-3DES
- MAC-X9.9(CBC)
- MAC-X9.19(CBC)

8. 数字签名

数字签名就是通过将公钥密码反过来用而实现的，如下图所示：

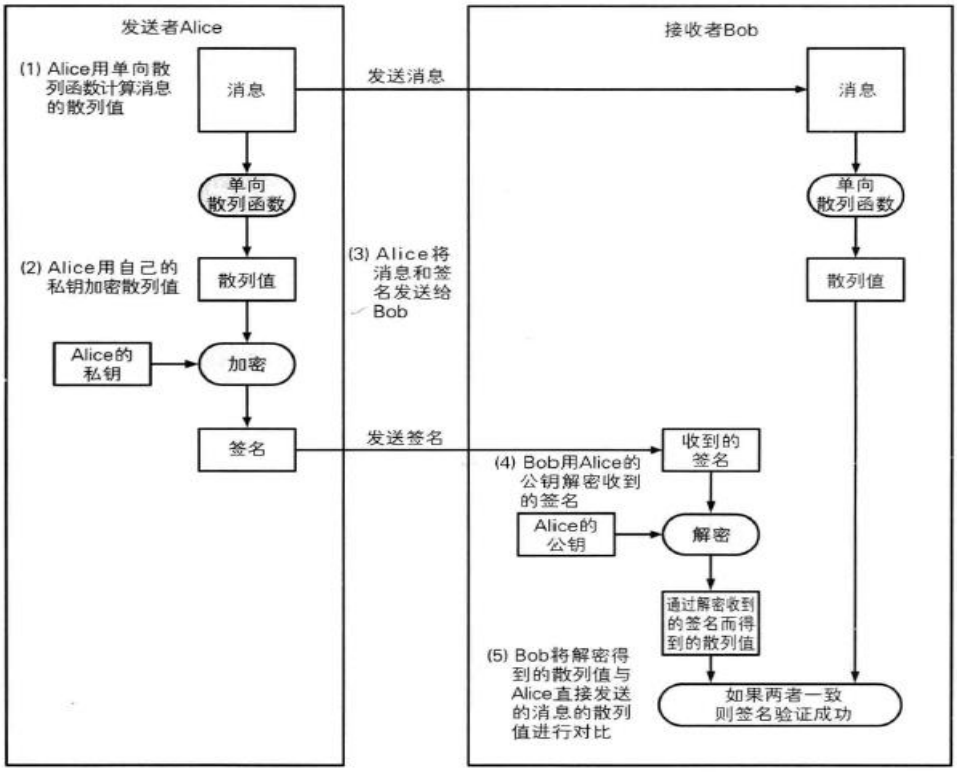


8.1. 直接对消息签名



8.2. 对消息的散列值签名

对整个消息签名时，如果消息很长，签名是很耗时的，所以可以先计算一个散列值，再对散列值签名。



8.3. 用 RSA 实现数字签名

RSA 签名过程：

签名 = 消息 ^{D} mod N　　（用 RSA 生成签名）

这里所使用的 D 和 N 就是签名者的私钥。签名就是对消息的 D 次方求 mod N 的结果，也就是说将消息和自己相乘 D 次，然后再除以 N 求余数，最后求得的余数就是签名。

RSA 的签名验证过程可用下列公式来表述：

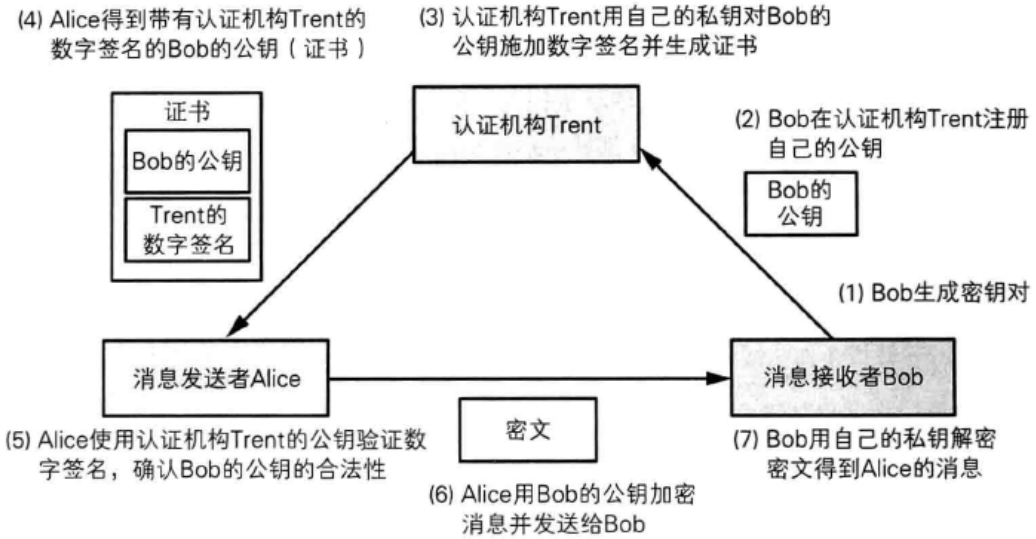
由签名求得的消息 = 签名 ^{E} mod N　　（用 RSA 验证签名）

这里所使用的 E 和 N 就是签名者的公钥。接收者计算签名的 E 次方并求 mod N，得到“由签名求得的消息”，并将其与发送者直接发送过来的“消息”内容进行对比。如果两者一致则签名验证成功，否则签名验证失败。

9. 证书

公钥证书（Public-Key Certificate，PKC）里面记录了姓名、组织、邮箱地址等个人信息，以及属于此人的公钥，并由认证机构（Certification Authority，CA）

施加数字签名。认证机构就是能够认定“公钥确实属于此人”并能够生成数字签名的个人或组织。认证机构中有国际性组织和政府所设立的组织，也有通过提供认证服务来盈利的一般企业，此外个人也可以成立认证机构。世界上最有名的认证机构当属 VeriSign 公司。



9.1.X509

使用最广泛的证书规范是 ITU（国际电信联盟）和 ISO（国际标准化组织）制定的 X.509 规范。

X.509 证书大体上包含以下 3 部分内容：

- 签名前的证书，就是签名对象信息
- 数字签名算法，对证书签名时所使用的算法
- 数字签名，对证书施加的数字签名

```
Certificate :
  DATA :
    Version : 3
    SerialNumber : 26:0c:3c:ea:3b:16:98:63:de:51:3e:2a:4f:e4:d6:2a:
    Signature Algorithm: SHA1WithRSAEncryption
    Issuer  :
      C=US, O=VeriSign, Inc., OU=VeriSign Trust Network, OU=Terms of use at https://www.verisign.
com/rpa (c)05, OU=Persona Not Validated, CN=VeriSign Class 1 Individual Subscriber CA - G2,
    Validity :
      notBefore : Jul 15 00:00:00 2008 UTC
      notAfter  : Sep 13 23:59:59 2008 UTC
    Subject :
      O=VeriSign, Inc., OU=VeriSign Trust Network, OU=www.verisign.com/repository/RPA Incorp. by
 Ref.,LIAB.LTD(c)98, OU=Persona Not Validated, OU=Digital ID Class 1 - Microsoft, CN=Bob Doe,
/Email=bobby@hyuki.com

    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (1024 bit)
        Modulus (1024 bit):
          f1:24:04:25:50:8d:6c:eb:dd:4f:1a:2b:d8:5b:14:83:15:bd:90:3b:
          4f:ed:47:d3:39:dd:23:a9:c5:88:84:68:7a:1a:54:31:33:35:eb:57:
          c3:d7:a4:8f:44:a4:b9:d1:5c:83:7c:89:18:ca:a0:a2:f8:47:e2:80:
          af:94:ac:83:a1:c9:24:5e:22:29:e1:eb:2e:75:1c:d6:89:4d:7b:13:
          b1:ea:b5:b4:f8:28:1e:82:9c:da:06:fb:9f:fd:06:aa:dd:63:6d:23:
          73:66:a0:18:c8:4e:63:c6:29:db:81:ac:1f:a3:7e:72:9e:91:6b:d0:
          d1:ab:7e:97:96:bb:2b:31:
        Exponent:
          00:01:00:01:

    X509v3 extensions:
      x509 Basic Constraints:
        CA:FALSE
        PathLenConstraint:NULL
      x509 Certificate Policies:
        policyID = 2.16.840.1.113733.1.7.23.3
          qualifierID = pkix-id-qt CPSurl
          qualifier = https://www.verisign.com/rpa
      x509 Key Usage:
        digitalSignature, keyEncipherment, (0xa0)
      x509 Ext Key Usage:
        1.3.6.1.5.5.7.3.4 = PKIX-IDKP-EmailProtection
        1.3.6.1.5.5.7.3.2 = PKIX-IDKP-ClientAuth
      x509 CRL Distribution Points:
        [0] dist-point :
          [0] fullName :
          [6] http://IndC1DigitalID-crl.verisign.com/IndC1DigitalID.crl

  Signature Algorithm: SHA1WithRSAEncryption
    3a:75:44:3b:50:b2:da:e5:a9:4b:3e:72:65:c9:be:fa:84:5b:
    03:b3:e4:11:7e:99:7d:90:18:c2:2f:ba:75:77:f6:48:7e:c5:
    d8:9f:8a:a6:ff:44:08:04:b4:0c:36:81:cf:b6:6a:b2:9e:26:
    aa:e4:11:39:a4:b6:20:e1:a7:e5:ad:4b:c6:88:36:4b:51:0a:
    7d:3f:17:b9:29:de:b2:c3:cd:53:f5:2b:1a:78:e4:d6:a5:fd:
    80:50:91:91:93:97:00:53:f6:d1:2a:96:18:ff:75:4e:fa:2a:
    ac:27:35:37:a3:8c:59:14:91:24:fd:8a:c5:dd:92:50:07:a0:
    7a:1a:8b:16:d1:04:78:c1:50:46:a3:54:fe:1d:dc:35:eb:33:
    29:e1:8d:30:ea:66:5a:b0:3a:2f:ea:9c:4c:7b:a3:88:02:a0:
    10:85:a9:d3:97:6a:4f:0e:a1:af:dd:05:3a:43:7f:2f:ce:62:
    c4:fa:e7:8b:3f:a5:57:6e:e2:a6:6b:7c:bc:69:78:4e:94:b4:
    eb:74:35:f0:7a:34:58:91:1a:4d:56:f2:be:4f:4d:37:1a:98:
    97:d2:4b:f4:1d:1b:29:f9:fe:a8:ea:3a:13:e5:93:c8:d0:4a:
    9f:40:76:aa:be:64:d9:c8:d9:31:93:94:91:00:ce:b7:2f:bb:
    3c:13:d9:2d:
```

签名前的证书	
规范版本	3
证书序列号	26:0c:3c:ea:3b:16:98:63:de:51:3e:2a:4f:e4:d6:2a:
数字签名算法	SHA1WithRSAEncryption
证书颁发者	VeriSign Class 1 Individual Subscriber CA - G2
有效期起始时间	Jul 15 00:00:00 2008 UTC
有效期结束时间	Sep 13 23:59:59 2008 UTC
公钥所有人	Bob Doe, /Email=bobby@hyuki.com
公钥算法	rsaEncryption
公钥内容	RSA Public Key: (1024 bit) Modulus (1024 bit): f1:24:04:25:50:8d:6c:eb:dd:4f:1a:2b:d8:5b:14:83:15:bd:90:3b: 4f:ed:47:d3:39:dd:23:a9:c5:88:84:68:7a:1a:54:31:33:35:eb:57: c3:d7:a4:8f:44:a4:b9:d1:5c:83:7c:89:18:ca:a0:a2:f8:47:e2:80: af:94:ac:83:a1:c9:24:5e:22:29:e1:eb:2e:75:1c:d6:89:4d:7b:13: b1:ea:b5:b4:f8:28:1e:82:9c:da:06:fb:9f:fd:06:aa:dd:63:6d:23: 73:66:ae:18:c8:4e:63:c6:29:db:81:ac:1f:a3:7e:72:9e:91:6b:d0: d1:ab:7e:97:96:bb:2b:31: Exponent: 00:01:00:01:
扩展属性[省略]	
数字签名算法	SHA1WithRSAEncryption
数字签名内容	3a:75:44:3b:58:b2:da:e5:a9:4b:3e:72:65:c9:be:fa:84:5b: 03:b3:e4:11:7e:99:7d:90:18:c2:2f:ba:75:77:f6:48:7e:c5: d8:9f:8a:a6:ff:44:08:04:b4:0c:36:81:cf:b6:6a:b2:9e:26: aa:e4:11:39:a4:b6:20:e1:a7:e5:ad:4b:c6:88:36:4b:51:0a: 7d:3f:17:b9:29:de:b2:c3:cd:53:f5:2b:1a:78:e4:d6:a5:fd: 80:50:91:91:93:97:00:53:f6:d1:2a:96:18:ff:75:4e:fa:2a: ac:27:35:37:a3:8c:59:14:91:24:fd:8a:c5:dd:92:50:07:a0: 7a:1a:8b:16:d1:04:78:c1:50:46:a3:54:fe:1d:dc:35:eb:33: 29:e1:8d:30:ea:66:5a:b0:3a:2f:ea:9c:4c:7b:a3:88:02:a0: 10:85:a9:d3:97:6a:4f:0e:a1:af:dd:05:3a:43:7f:2f:ce:62: c4:fa:e7:8b:3f:a5:57:6e:e2:a6:6b:7c:bc:69:78:4e:94:b4: eb:74:35:f0:7a:34:58:91:1a:4d:56:f2:be:4f:4d:37:1a:98: 97:d2:4b:f4:1d:1b:29:f9:fe:a8:ea:3a:13:e5:93:c8:d0:4a: 9f:40:76:aa:be:64:d9:c8:d9:31:93:94:91:00:ce:b7:2f:bb: 3c:13:d9:2d:

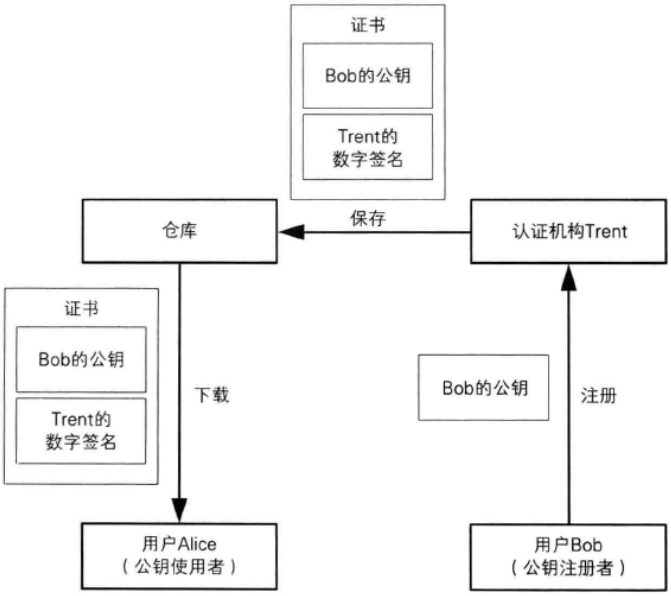
9.2. 公钥基础设施

公钥基础设施（Public-Key Infrastructure）是为了能够更有效的运用公钥而制定的一系列规范和规格的总称，简称 PKI。

RSA 公司制定的 PKCS（Public-Key Cryptography Standards，公钥密码标准）系列规范也是 PKI 的一种。X.509 也是 PKI 的一种。

KPI 的组成要素：

- 用户，使用 PKI 的人
- 认证机构，颁发证书的人
- 仓库，保存证书的数据库



注册公钥的用户所进行的操作：

- 生成密钥对
- 在认证机构注册公钥
- 向认证机构申请公钥
- 根据需要申请作废已注册的公钥
- 解密接收到的密文
- 对消息进行数字签名

使用已注册的公钥的用户所进行的操作：

- 将消息加密后发给接收者
- 验证数字签名

认证机构(CA)

- 生成密钥对（也可以由用户自己生成），传送私钥需要使用 PKCS#12 规范。
- 在注册公钥时对本人身份进行认证，申请证书使用 PKCS#10 规范。
- 生成并颁发证书，生成的证书使用 PKCS#6 和 X.509 规范。
- 作废证书

仓库

保存证书的数据库，PKI 用户在需要的时候可以从中获取证书。

10. 国密算法

国密算法是我国定制的商用加密算，用于金融行业。包括 SM1、SM2、SM3、SM4。秘钥长度与分组长度均为 128 位。

10.1. SM1

SM1 算法为对称加密。其加密强度与 AES 相当。该算法不公开，调用该算

法时，需要通过加密芯片的接口进行调用。

10.2. SM2

SM2 为非对称加密算法，基于 ECC 椭圆曲线密码机制，其签名、秘钥交换方面不同于 ECDSA、ECDH 等国际标准，而是采取更安全的机制。同时由于该算法基于 ECC，故其签名速度与秘钥生成速度都快于 RSA 算法。ECC 256 位(SM2 采用的就是 ECC 256 位的一种)安全强度比 RSA 2048 位高，但运算速度快于 RSA 2048。

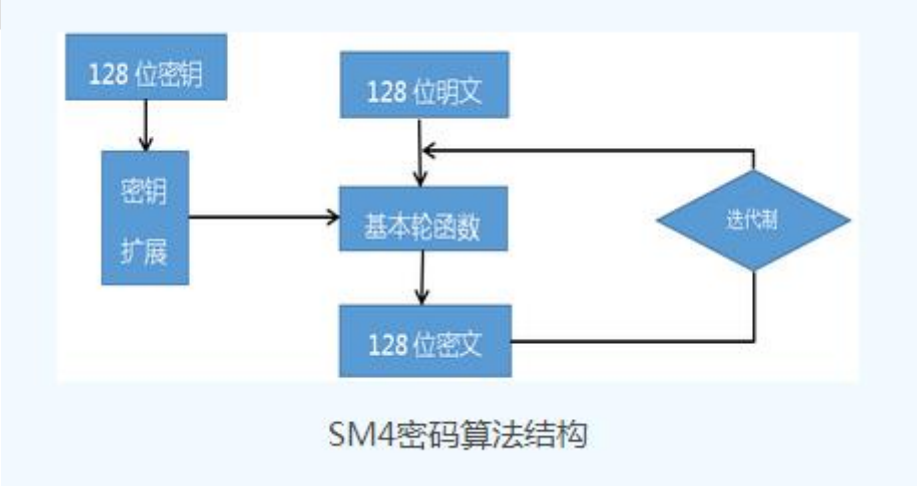
10.3. SM3

SM3 算法是一种消息摘要，类似单向散列函数(MD5/SHA1)，但是其校验结果为 256 比特，因此该算法的安全性高于 128 比特的 MD5 算法与 160 比特的 SHA1 算法。

10.4. SM4

SM4 为分组对称加密算法，用于无线局域网产品。该算法数据分组长度为 128bit，秘钥长度为 128bit。加密算法与密钥扩展算法均采取 32 轮迭代结构。SM4 密码算法以字节(8 位)和字节(32 位)作为单位进行数据处理。

SM4 密码算法是对合运算，因此解密算法与加密算法的结构相同，只是轮密钥的使用顺序相反，解密轮密钥是加密轮密钥的逆序。



SM4密码算法结构

11. 总结

信息安全是一门需要深度思考的复杂的学问，世界上没有一个系统是绝对安全的，如何设计一个相对安全的系统（就是让破解更加麻烦，时间足够长，成本足够高）需要大家不断的思考和不断的改进。实际开发中，一种密码技术并不

足以保证系统安全，一般是上面提到的密码技术综合使用，如采用多级秘钥加密体系——（非对称秘钥->一级对称秘钥（结合真随机数）->二级对称秘钥（各种消息验证算法））。后续工作中大家有秘钥体系设计方面的问题可以相互交流