程序员密码学基础导览

本导览主要面向**程序员(programmer)**（而非系统管理员、运维人员、IT维护人员、密码学家），初略介绍日常编程中可能会用到的密码学知识。

一、基本概念和术语介绍

场景一：Alice需要向Bob发送邮件。她不想让别人看到邮件的内容，于是她决定将邮件进行**加密(encrypt)**后再发送出去。加密之前的消息称为**明文(plaintext)**，加密之后的消息称为**密文(ciphertext)**。

Bob收到了Alice的加密邮件后，需要对密文进行**解密(decrypt)**之后才能阅读其内容。解密就是将密文恢复成明文的过程。

从明文生成密文的步骤，称为“加密算法”；从密文还原为明文的步骤，称为“解密算法”。加密、解密的算法合在一起统称为**密码算法(cryptographic algorithm)**。

密码算法中一般需要用到**密钥(key)**，就如同现实生活中需要用钥匙来锁门和开锁一样。

通过引入密码算法，Alice保护了其邮件的**机密性(confidentiality)**，第三方窃听者只能得到密文，需要采用某些困难的手段将其**破译(cryptanalysis)**为明文才能获取有价值的信息。

场景二：Bob收到了一封来自Alice的邮件，内容是“以100万元的价格购买该商品”。此时谨慎的Bob需要考虑三个问题：

- 这封邮件是否真的是Alice所发送？邮件的发送者(From: 一栏的内容)很容易被伪造，存在别人**伪装(spoofing)**成Alice的风险。

- 假定这封邮件真的是Alice所发，但是否有可能，Alice当初写的内容是“1万元”，而在邮件传输过程中被某些别有用心的人进行了**篡改(tamper)**，将“1万元”改为了“100万元”呢?
- 最后，即使Alice真的向Bob发送过“以100万元购买该商品”的邮件，但后来她反悔不想买了，便慌称根本没有发送过这样的邮件。这种事后推翻自己先前主张的行为，称为**否认(repudiation)**。

为了防止伪装，需要引入**认证(authentication)**机制；

为了防止篡改，需要引入**完整性(integrity)** 检查机制；

为了防止否认，需要引入**数字签名**机制；

二、信息安全所面临的威胁及用来应对的密码技术

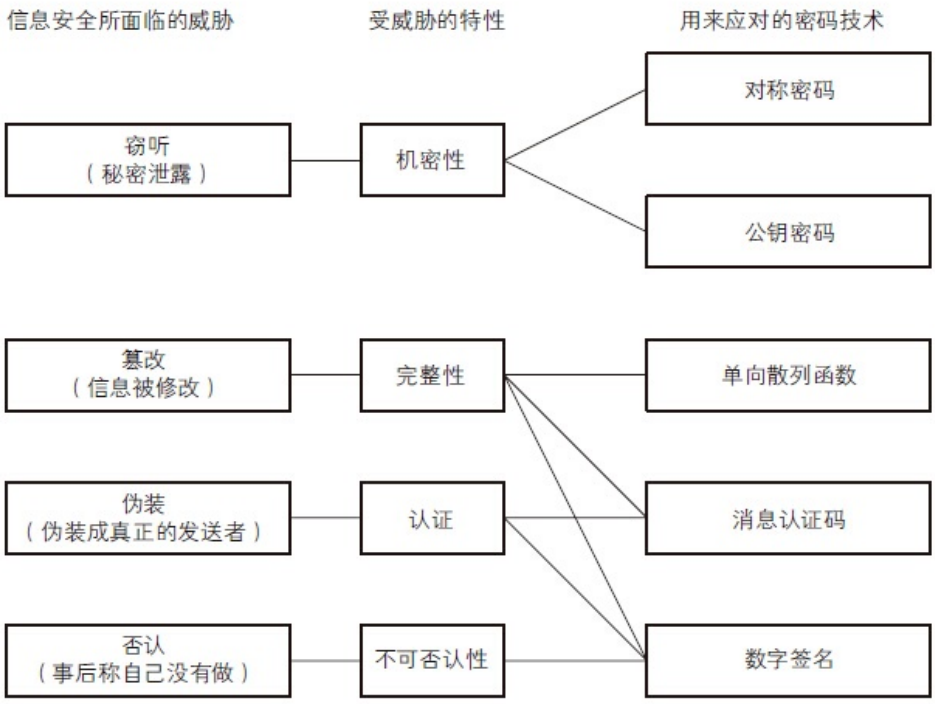


图 1-8　信息安全所面临的威胁与应对这些威胁的密码技术

三、密码算法分类

分组密码和流密码

- **分组密码(block cipher)**是每次只能处理特定长度数据块的一类密码算法，这里的一块就被称为一个**分组(block)**，每个块的比特数被称为**分组长度(block length)**。
- **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比特为单位进行加解密处理。

对称密码和非对称密码

- **对称密码(symmetric cryptograph)**是指在加密和解密时使用同一密钥的一类密码算法。
- **非对称密码(asymmetric cryptograph)**是指加密和解密时使用不同的密钥的一类密码算法。非对称密码又被称为**公钥密码(public-key cryptograph)**。

四、对称密码

XOR和一次性密码本

一次性密码本由维纳(G.S. Vernam)于1917年提出，其原理是"将明文和一串随机的比特序列进行XOR运算"。

香农在1949年证明了一次性密码本是无法破译的(theoretically unbreakable)，因为没有办法在暴力破解过程中判断当前所解出的结果是否为正确的明文。

但它不具备实用性，因为：

- 密钥生成需要真随机数；
- 密钥和明文等长；
- 为了安全的传输n个比特的明文，需要先安全的配送n个比特长的密钥；但如果有办法安全的配送n个比特长的密钥的话，直接用它来传输明文好了，根本不需要搞什么加密；

一次性密码本的思路孕育出了流密码(stream cipher)算法。

DES

DES(Data Encryption Standard)由IBM发明，于1977年被美国联邦信息处理标准(FIPS)所采纳。

DES是一种将64比特明文转换为64比特密文的分组密码(block cipher)算法，其密钥长度为56比特（规格上是64比特，但每隔7个比特会设置一个用于错误检测的比特，因此实质为56比特)。

现代计算机的计算能力已经可以暴力破解DES，不应再继续使用。

Triple-DES

3重DES(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。

明文经过三次DES处理才能变成最后的密文，由于DES的密钥长度为56比特，因此3重DES的密钥长度为 56 x 3 = 168比特。

3重DES对普通DES具备向下兼容性，因为其3次处理过程为：加密->解密->加密。当三部分的密钥设置为相同值时，第一步和第二步相当于空操作，因此实质就退化为普通的DES了。

其处理速度慢，安全性一般，不建议使用。

AES

AES(Advanced Encryption Standard)是取代其前任标准DES而成为新标准的一种对称加密算法。全世界的企业和密码学家提交了多个对称密码算法作为AES的候选，最终在2000年从这些候选算法中选定了Rijndael，并将其确定为了AES。

Rijndael是由比利时密码学家Joan Daemem和Vincent Rijmem设计的分组密码算法，其分组长度为128比特，密钥长度可以以32比特为单位在128比特到256比特的范围内进行选择（不过在AES的规则中，密钥长度只有128、192和256比特三种）。

AES经过了许多安全性审核验证，其执行速度也比较快(特别是现代CPU内置了专门的AES加速指令)，推荐使用。

五、非对称密码

RSA

TODO

六、分组密码的模式

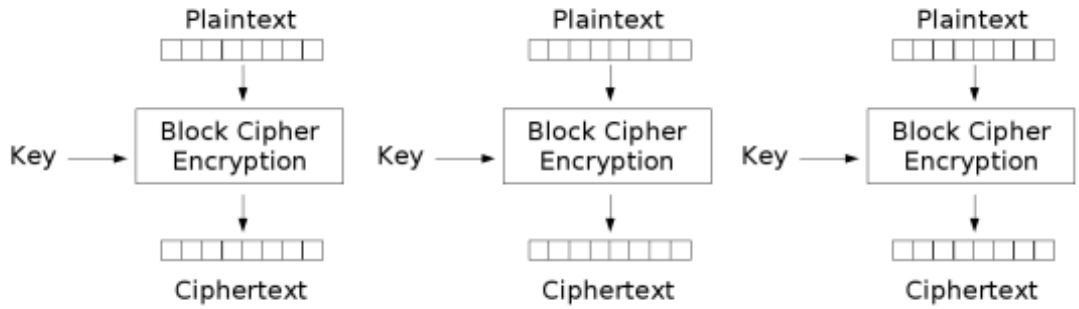
分组密码只能加密固定长度的明文。如果需要加密任意长度的明文，则需要对分组密码进行迭代，而分组密码的迭代方法就称为分组密码的"模式"。

分组模式对加密强度有很大影响。

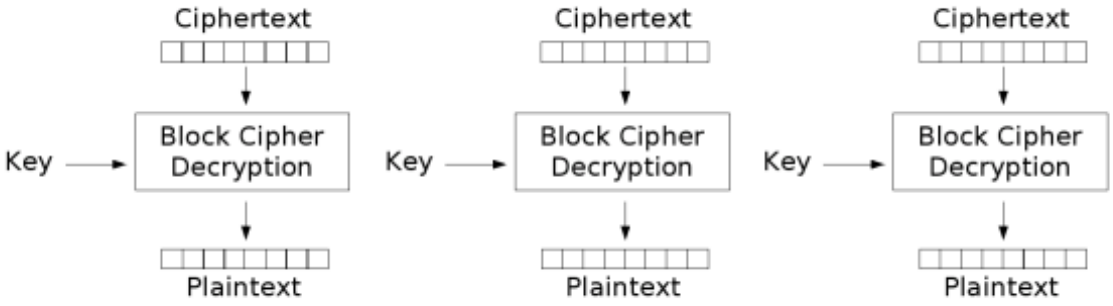
ECB模式

ECB模式的全称是Electronic　CodeBook模式(电子密码本模式)。在ECB模式中，明文分组加密之后的结果直接成为密文分组。

当最后一个明文分组的内容小于分组长度时，需要用一些特定的数据进行**填充(padding)**后再加密。



Electronic Codebook (ECB) mode encryption



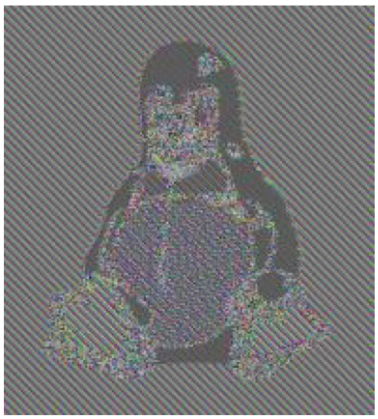
Electronic Codebook (ECB) mode decryption

如果明文中存在多个相同的明文分组，则这些明文分组将被转换为相同的密文分组。因此简单的观察密文，即可知道明文中存在怎样的重复组合，并以此为线索来破译密码。

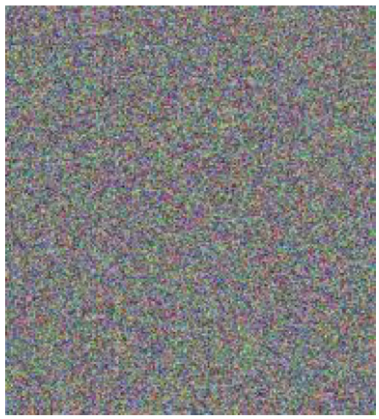
另外，攻击者无需破译密码就能够操纵明文，例如将两个密文分组对调、删除某个密文分组、复制某个密文分组，所解密出来的明文就会发生对应的改变。

一个用ECB模式加密的示例：

原图



使用ECB模式加密

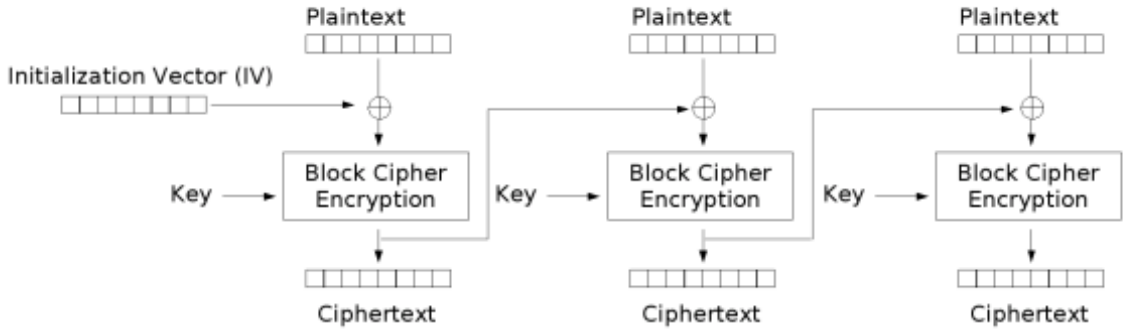


提供了伪随机性的非ECB模式

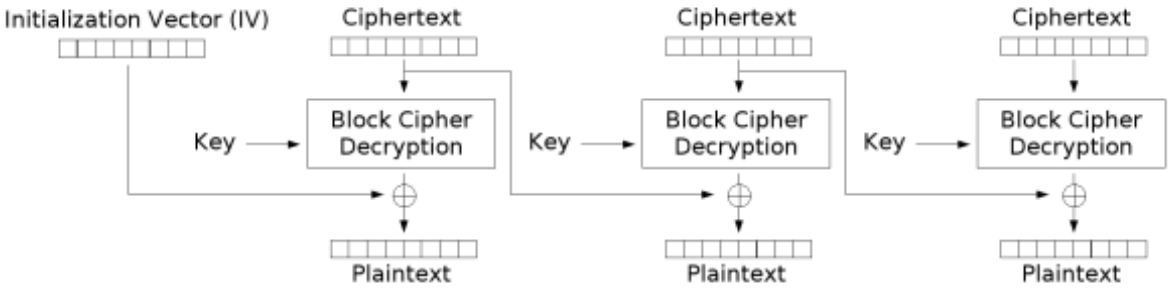
CBC模式

CBC模式的全称是Cipher Block Chaining模式(密文分组链接模式)。在CBC模式中，首先将明文分组与前一个密文分组进行XOR运算，然后再进行加密处理。

在加密第一个明文分组时，由于不存在“前一个密文分组”，因此需要事先准备一个长度为一个分组长度的比特序列来代替“前一个密文分组”，这个比特序列称为**初始化向量(initialization vector)**，通常缩写为IV。



Cipher Block Chaining (CBC) mode encryption

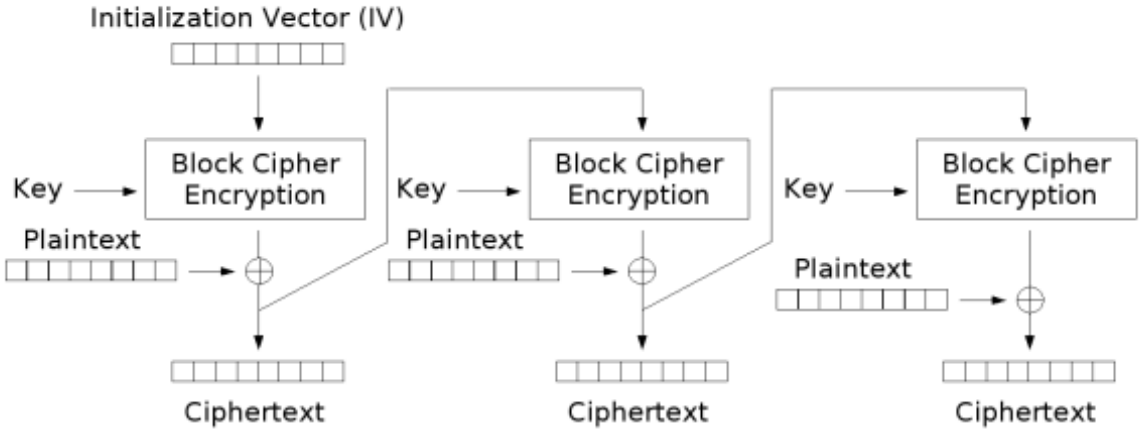


Cipher Block Chaining (CBC) mode decryption

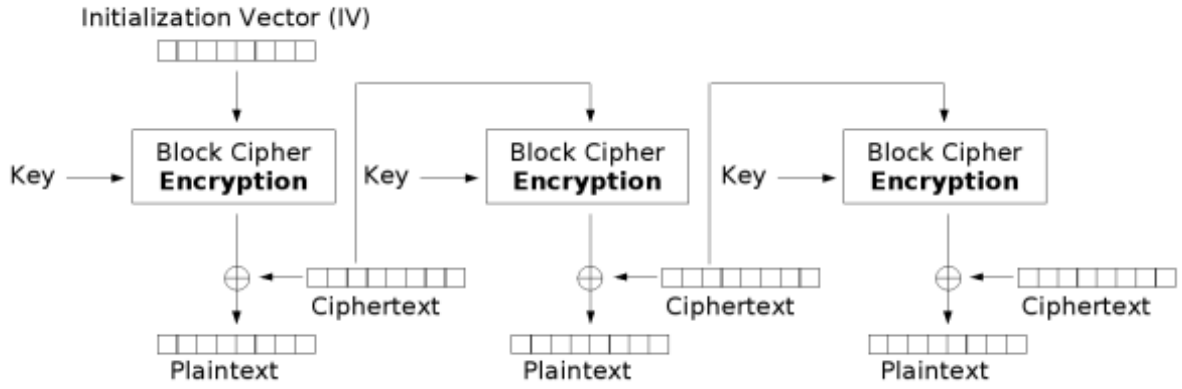
CBC的加密过程是串形的，无法被并行化。解密时，由于两个相邻的密文分组可以解出一个明文分组，因此可以并行化。

CFB模式

CFB模式的全称是Cipher FeedBack模式(密文反馈模式)。在CFB模式中，前一个密文分组作为输入数据进行一次加密运算，再与当前明文分组进行XOR操作，生成当前密文分组。



Cipher Feedback (CFB) mode encryption



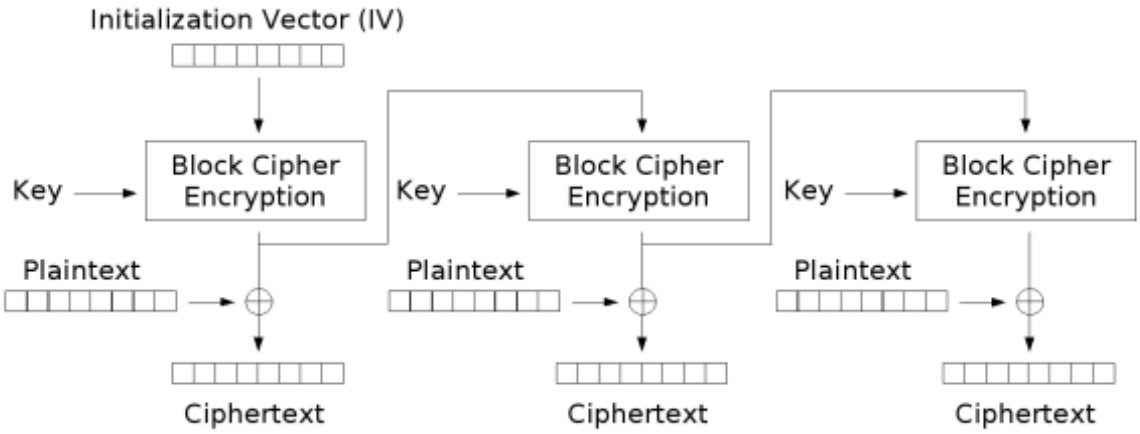
Cipher Feedback (CFB) mode decryption

CFB中，密码运算用于生成**密钥流(key stream)**比特序列，再与明文分组进行XOR操作。明文分组可以逐字节加密，无需padding，因此我们可以将CFB模式看出是一种使用分组密码来实现流密码的方式。

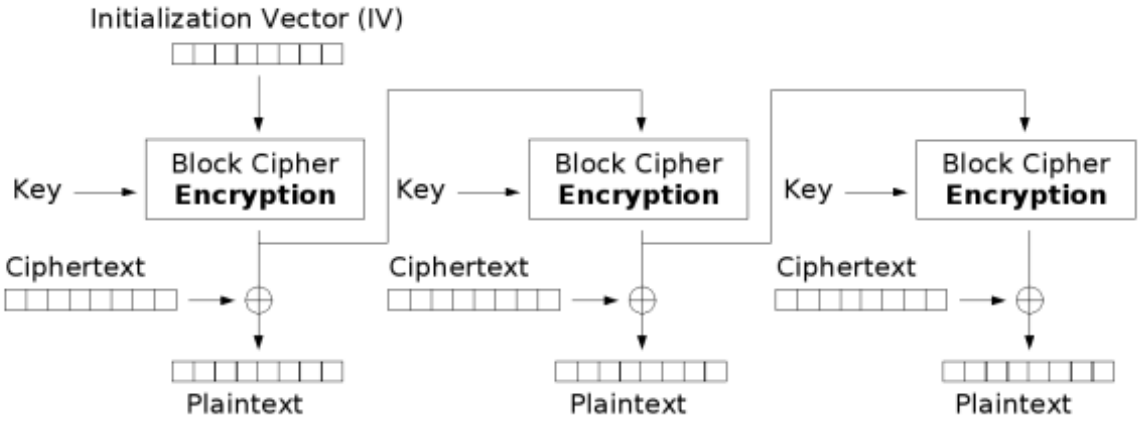
与CBC类似，CFB的加密过程是串行的，解密过程则可以并行化。

OFB模式

OFB模式的全称是Output Feedback模式(输出反馈模式)。在OFB模式中，前一次密码运算的输出会被反馈到当前密码算法的输入，所得到的结果再与当前明文分组进行XOR，得到当前的密文分组。



Output Feedback (OFB) mode encryption

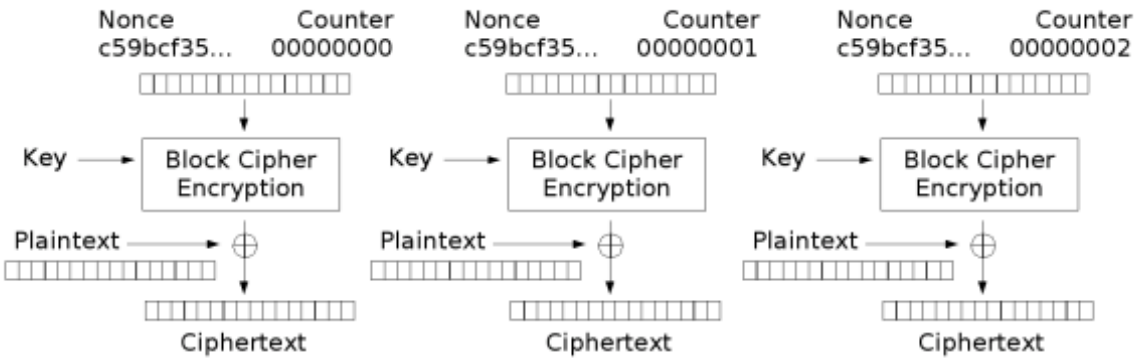


Output Feedback (OFB) mode decryption

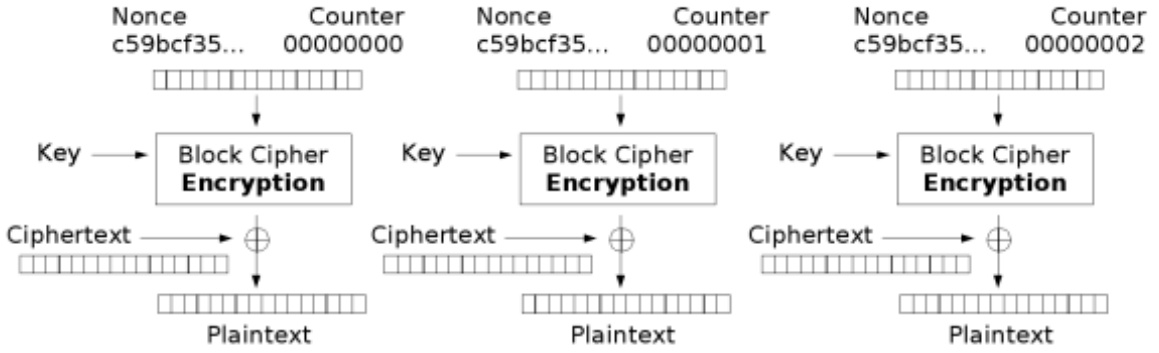
OFB与CFB一样，密码运算生成的都是密钥流，再与明文进行XOR操作。区别是，OFB的密钥流生成，和所有的明文分组都完全无关，因此可以提前生成密钥流。这种情况下的加密过程，仅仅是简单的XOR操作，是非常快的。

CTR模式

CTR模式的全称是CounTeR模式(计数器模式)。CTR模式通过递增一个计数器，用密码算法对此计数器进行加密，来生成密钥流，再与明文分组进行XOR处理。



Counter (CTR) mode encryption



Counter (CTR) mode decryption

CTR模式中计数器的初始值，被称为**nonce**。

由于加解密运算时所需要的"计数器"的值可以经由nonce和分组序号直接算出，因此CTR模式下可以以任意顺序对分组进行加密和解密，完全的并行化。

使用建议

不应使用ECB。

建议使用CBC和CTR。

七、单向散列函数

TODO

八、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，一般简称为MAC。

消息认证码的输入为任意长度的**消息**和一个在发送者与接收者之间的**共享密钥**，其输出为固定长度的数据，这个数据被称为**MAC值**。在合理的实现中，输入数据（消息、密钥）的任意改变几乎都会导致不同的MAC值输出。

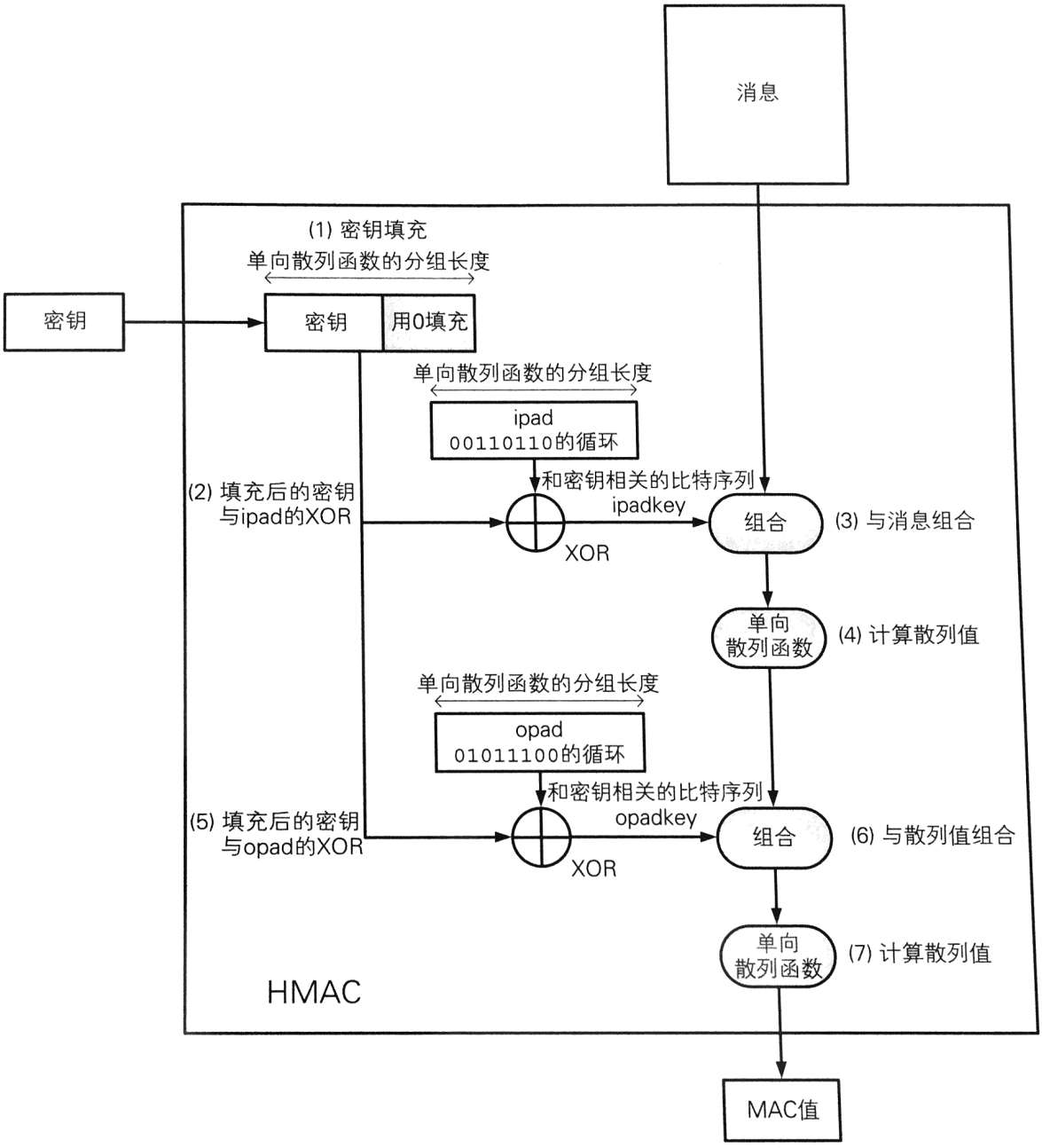
假设发送者Alice要向接收者Bob发送消息，如果使用了消息认证码，接收者Bob就能判断出所接收到的消息与发送者Alice所发出去的消息是否一致，因为**消息中的MAC值只有用Alice和Bob之间共享的密钥才能计算出来**。

消息认证码无法解决如下问题：

- 向第三方证明所收到的消息是对方发送的；
- 防止发送者否认；

HMAC是一种使用单向散列函数来构造消息认证码的方法（RFC2104），常见的HMAC构造有HMAC-MD5、HMAC-SHA-1等。

HMAC 中是按照下列步骤来计算 MAC 值的（图 8-3）。



九、AEAD构造

Authenticated Encryption with Associated Data (AEAD) is a form of encryption which simultaneously provides confidentiality, integrity, and authenticity

assurances on the data; decryption is combined in single step with integrity verification.

A typical programming interface for AEAD mode implementation would provide the following functions:

1. Encryption
   - Input: plaintext, key, and optionally a header in plaintext that will not be encrypted, but will be covered by authenticity protection.
   - Output: ciphertext and authentication tag (Message Authentication Code).
2. Decryption
   - Input: ciphertext, key, authentication tag, and optionally a header.
   - Output: plaintext, or an error if the authentication tag does not match the supplied ciphertext or header.

基本上AEAD就是综合了加密和消息验证码，同时能支持部分数据不被加密但会被消息验证码保护。最典型的一种用况是网络协议包通讯：明文数据需要被加密以保证机密性；整个数据包要能够校验完整性以防被第三方篡改；数据包的包头中包含了用于做协议包定界的数据（例如包长度）希望不被加密但也希望保证其完整性。

常见实现包括AES-GCM和ChaCha20-Poly1305。在AES-GCM中，密码算法使用的是AES in Counter mode(AES-CTR)，MAC使用的是通过AES-CTR加密的GHASH。在ChaCha20-Poly1305中，ChaCha20用作密码算法，Poly1305用作MAC算法。

我们用到了一个ChaCha20-Poly1305的变种，其大致流程为：

1. 整个协议包依次由4字节的包长度、ciphertext、MAC组成；
2. 以给定的key, nonce和0值的counter运行1次ChaCha20，64字节输出的前32字节保留下来作为Poly1305的一次性密钥，其它数据丢弃。

3. 以给定的key，nonce并以1作为counter的起始值来对plaintext进行加密，得到ciphertext。
4. 将包长度和ciphertext视为一个首位相连的的虚拟buffer，在这个buffer上以第1步计算出的Poly1305 key为密钥，计算出Poly1305 MAC值。
5. 将MAC拼接在ciphertext之后，整个协议包构造完成。