# 密码技术

本文主要介绍为了保证信息机密性、完整性和可用性（CIA三要素）而采用的密码技术。

通过加密技术可以保证信息的机密性。加密技术从密钥类型上来分可以分为对称密码（加解密用相同密钥）和公钥密码（加解密用不同密钥），从密码算法处理数据的模式上来分可以分组密码（分组处理数据）和流密码（一次性处理数据）。这里主要介绍对称密码、公钥密码和分组密码模式（传输大量数据时需要用到的分组加密模式）。

通过认证技术中的消息认证码和数字签名技术可以保证信息的完整性。而贯穿其中的单向散列函数也是我们讲解的重点。通过认证技术中的证书可以保证信息的可用性。

而最后介绍的随机数在以上的各种技术中一般担当随机密钥的角色。因此密钥的不可预测性也显得至关重要。

## 1、对称密码

对称密码中加密和解密时都是使用的同一个密钥（密钥是对称的），因此获得了对称密码的称

号。这里我们主要介绍三种对称密码算法：一次性密码本、DES、AES。

### 1.1、一次性密码本

一次性密码本属于流密码，并被称为绝对不会被破译的密码（即便使用暴力破解也无法破译），我们可以带着这个结论去看下面的介绍。

#### 数学原理

一次性密码本算法的基本运算就是异或运算，其中规定了密钥的长度要和明文的长度一致，因此加密后的密文与明文长度是一致的。

|  |  |
| --- | --- |
| **加密** | **解密** |
| 密文 = 明文⊕密钥 | 明文 = 密文⊕密钥 |

#### 不可破译性

那么为什么一次性密码本是不会被破译的呢？试想一下我们使用暴力破解去破解密钥的过程。首先，密钥长度和明文长度一致而明文长度和密文长度一致，这就可以通过密文长度确定密钥空间。接着，我们开始使用每一个可能的密钥去解密，这时会发现每一个密钥都可以解密出对应的明文，但是我们却无法确认哪一个明文是正确的明文，这也就造成了一次性密码本的不可破译特性。

#### 算法分析

|  |  |
| --- | --- |
| **优点** | **缺点** |
| 1.无法破译，安全性极高 | 1.存在密钥配送问题 |
| 2.效率高 | 2.密钥保存代价高 |
|  | 3.密钥不可重用 |
|  | 4.密钥同步易出错 |
|  | 5.密钥生成难度高 |

该算法的核心运算是异或运算，这就使得一次性密码本的效率非常高；而由于其独特的不可破译特性，也让一次性密码本有了非常高的安全性。

但是一次性密码本却很少被使用，这又是为什么呢？算法本身已经证明是无法破译的，那么安全的关键就在于密钥的安全性，那么应该是密钥的安全性方面的问题导致了一次性密码本的不实用。

首先，存在密钥配送问题，要保证密钥不泄露就必须保证具有一个合理的方式来让双方商议密钥，这个在后续的介绍中可以了解到公钥加密、Diffie-Hellman密钥交换等算法来解决。

其次，存在密钥保存问题，密钥的长度要与明文的长度一致，那么保存明文变成了保存长度一致的密钥，并没有从本质上解决问题。

接着，由于一次性密码本每次通信都要用到新的密钥（这也是一次性的意义），导致了密钥不可重用的局限性。

然后，在通信过程当中需要传输与明文长度一致的密钥，那么当明文很长时，密钥对应的也会很长，这就大大增加了密钥在同步过程当中出错的概率，而一旦密钥出错，也就无法加密了。

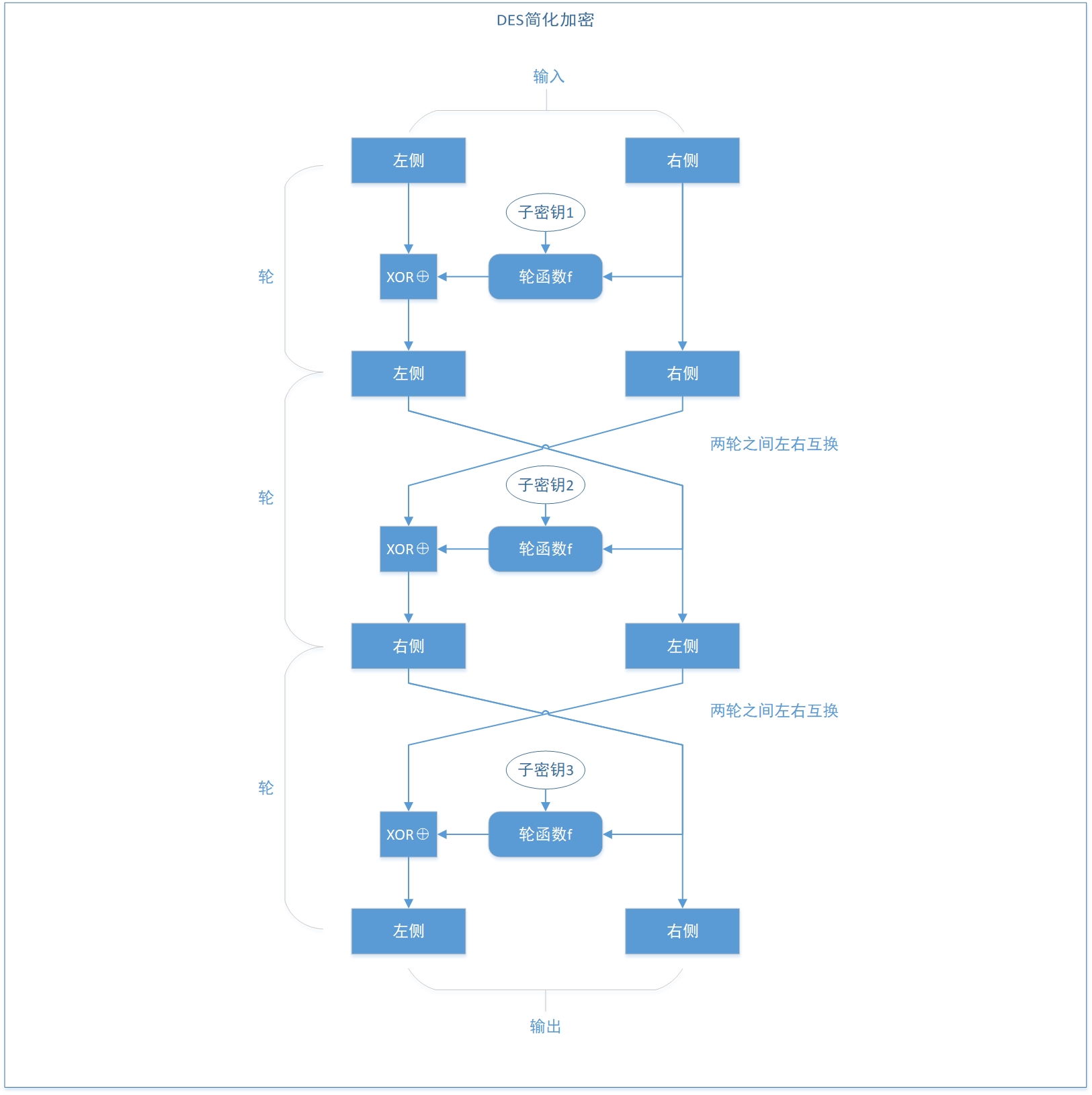
最后，一次性密码本中需要生成大量的随机数，而这里的随机数并不是指程序生成的伪随机数，而必须是无重现的真随机数。

### 1.2、DES

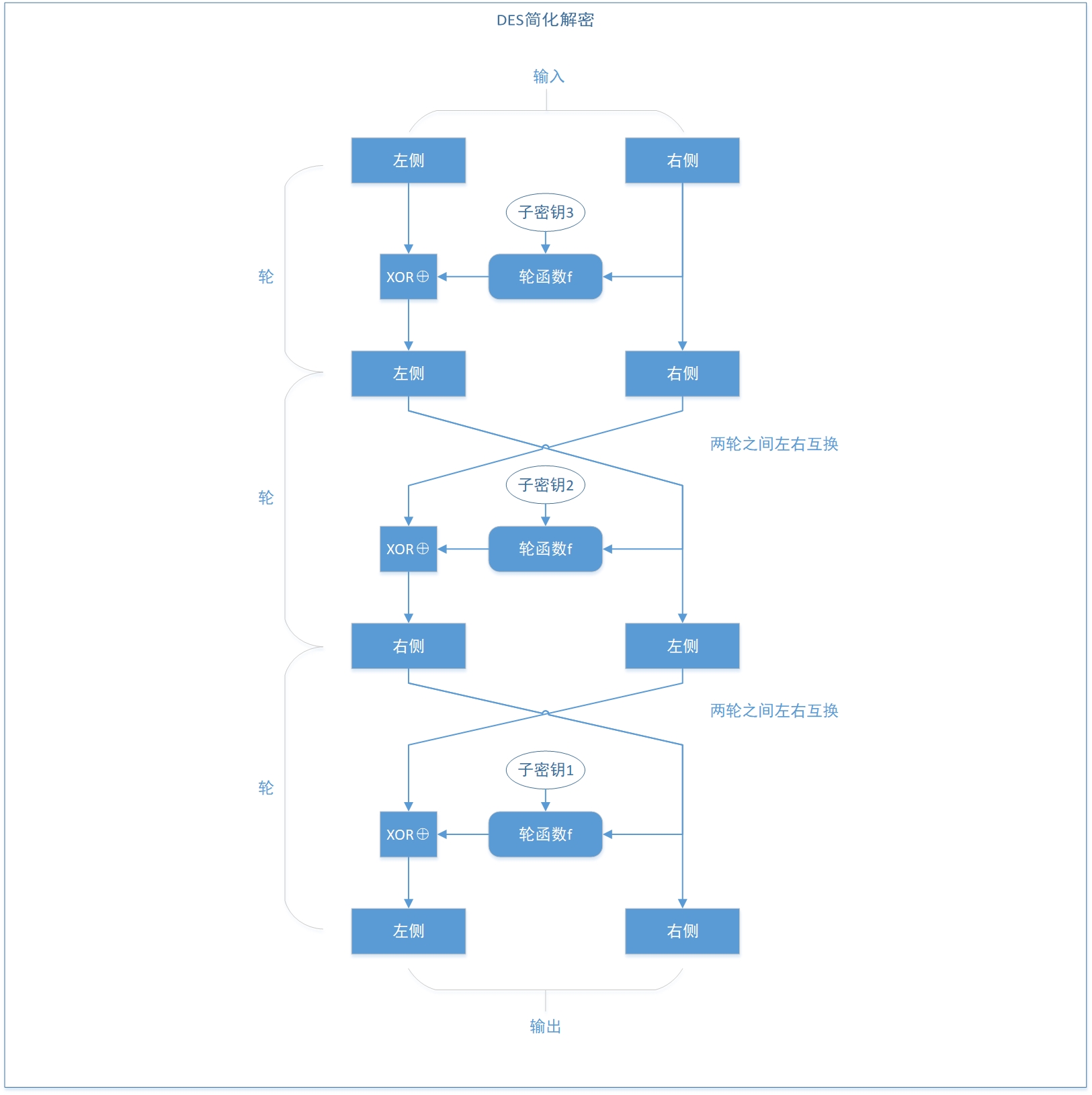
DES属于分组密码，并且每次只能加密64比特的数据，密钥也为64位（只用到了56位）如果要加密长明文就需要对DES机密进行迭代。因此密文长度比明文长度达0到63个比特。DES加密算法如今已经可以在短时间内破译了，因此除了兼容旧系统，都不用DES加密。

#### 数学原理

DES的基本结构时Feistel网络，在Feistel网络中加密的各个步骤称为轮，整个加密过程就是若干次轮的循环，而DES是一种16轮的循环的Feistel网络。下面以三轮循环的Feistel网络为例介绍DES的加密、解密过程。



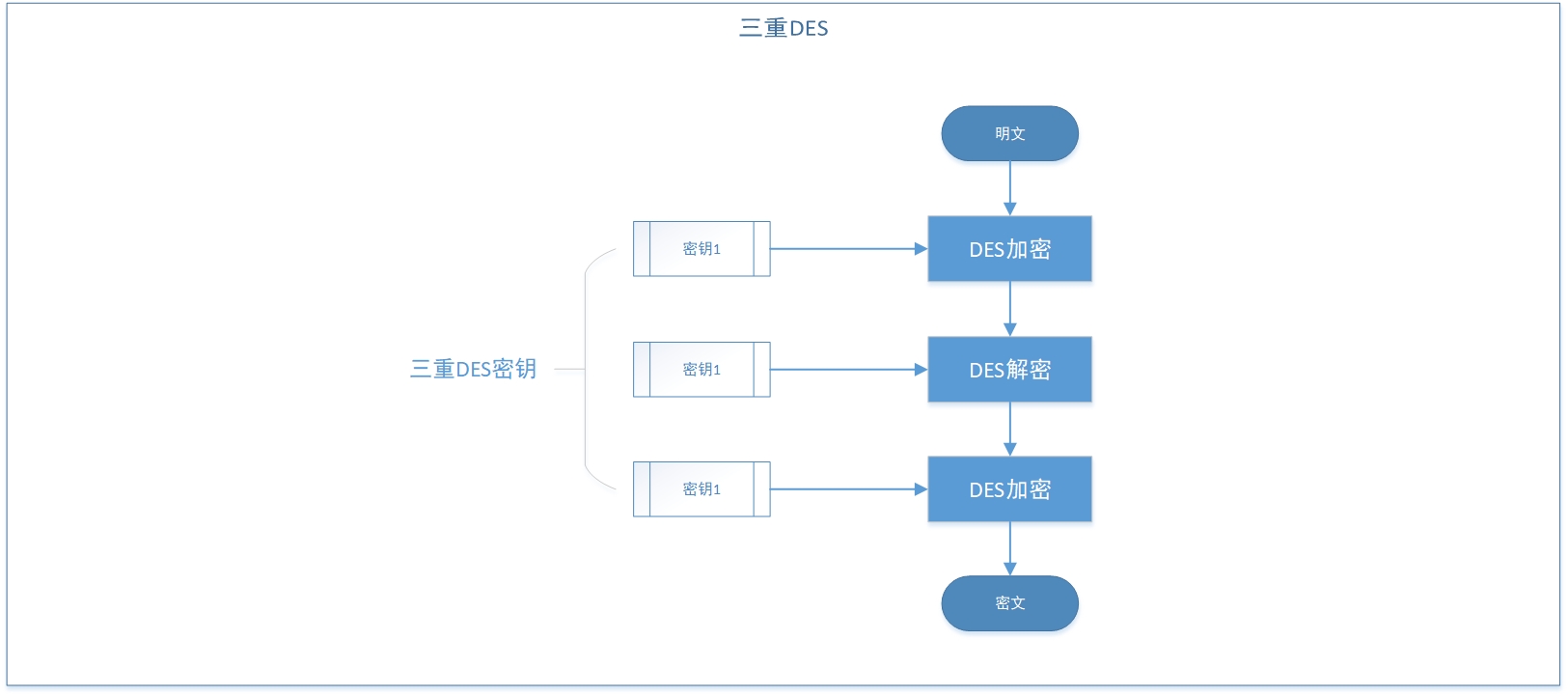
如上图所示，加密过程中每一轮的输入会被分为两部分，其中只有左侧会被处理而右侧则直接和处理后的左侧拼接在一起，作为一轮处理的结果。处理左侧的方法就是通过论函数f和每一轮的子密钥（每一轮的子密钥都不同，子密钥是以DES密钥为种子的伪随机数）对右侧数据进行加密再与左侧数据进行异或操作得到一轮的处理结果。然后一轮结束后就互换左右侧作为下一轮的输入，做到左右侧轮流加密，并且最后一轮不用互换，直接作为加密结果输出。



如上图所示，解密过程和加密过程是非常类似的，两者的区别就是加密使用的子密钥顺序和解密使用的子密钥顺序是相反的，也就是说解密时用的也是同一个算法，只要控制子密钥的顺序与加密时相反就可以了。

#### 三重DES

由于DES目前已经有可行的破译方法，而为了解决这个问题衍生出了三重DES。三重DES可以兼容DES，但是在多了两重加密的情况下，效率比较低。因此除了必须兼容DES外，一般不推荐使用。



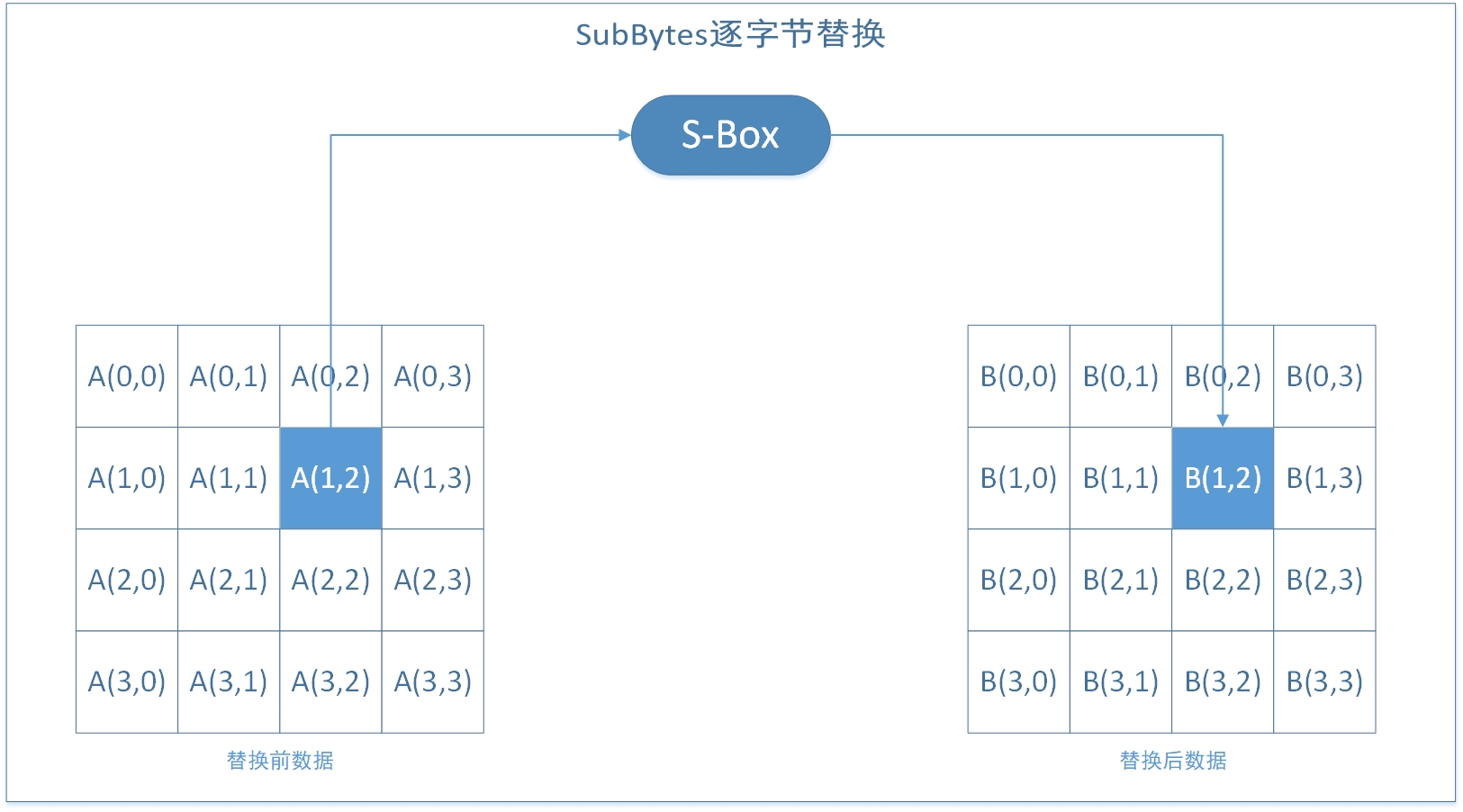
如上图所示，三重DES加密其实就是进行DES加密、解密、加密的过程（解密时为解密、加密、解密），但是三个过程都是使用的不一样的密钥，若要兼容DES只需要把三个密钥都设置为一样就可以了。因此三重DES密钥的长度也是DES的三倍。

### 1.3、AES

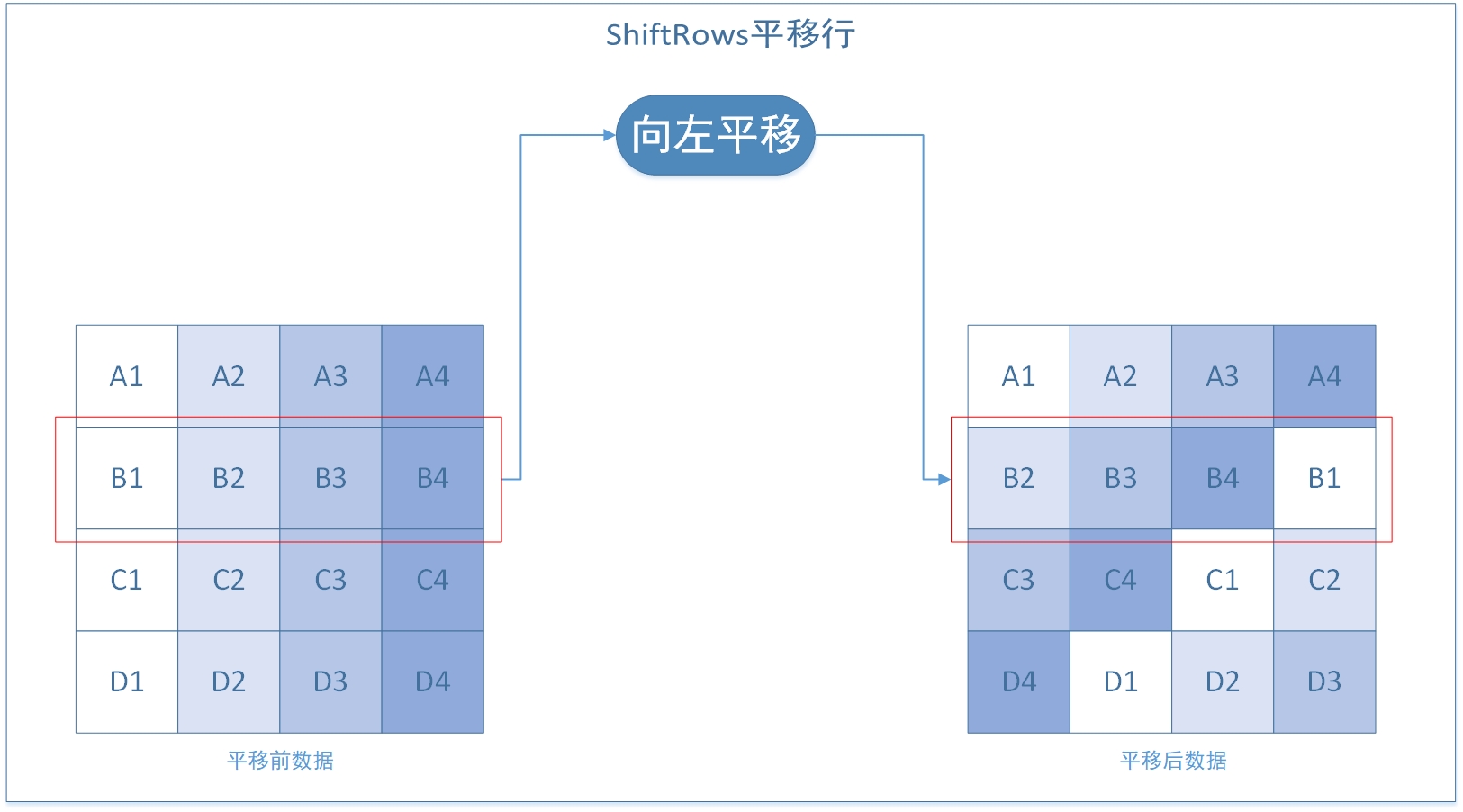
AES算法属于分组密码，其分组长度为128比特，密钥长度有128、192、256比特三种。AES算法是采用Rijndael算法实现的，并且Rijndael算法采用的是SPN网络（代替置换网络），而不是DES采用的Feistel网络。

#### 数学原理

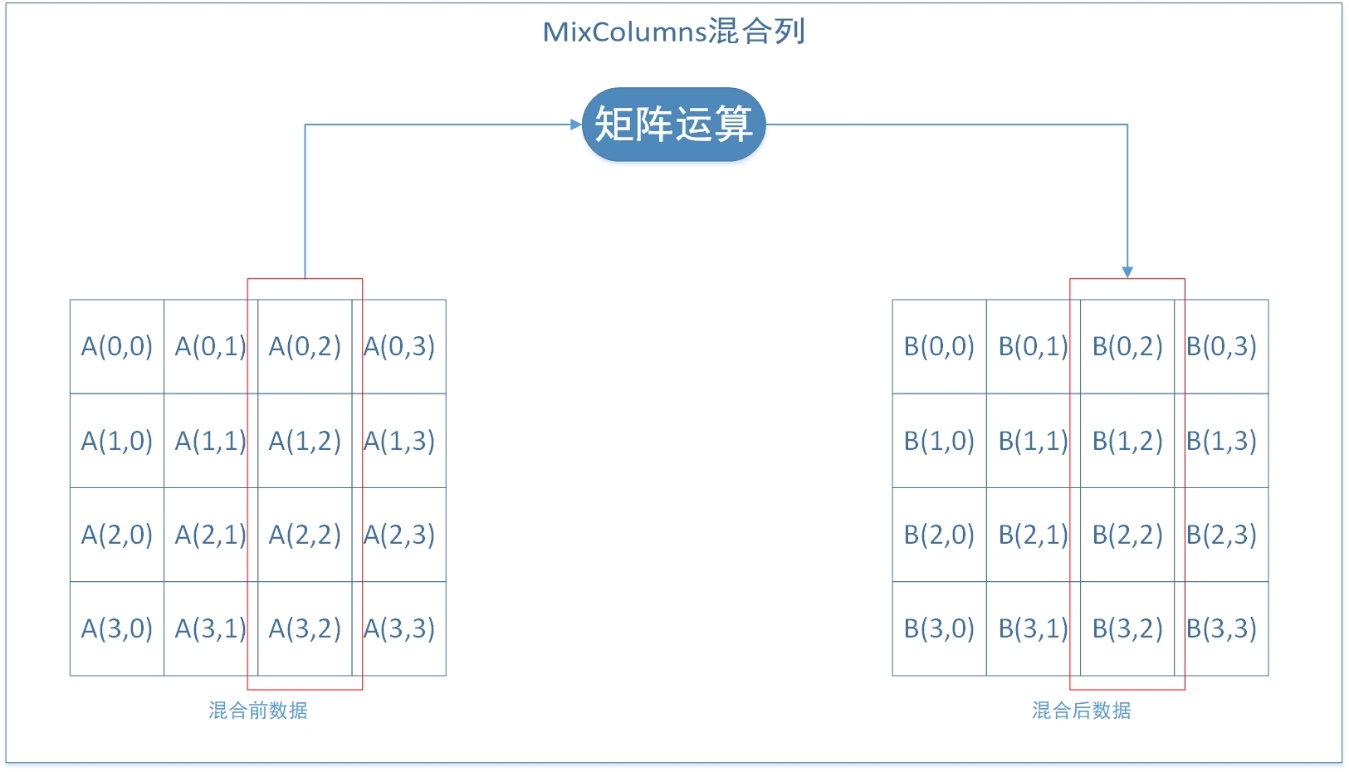
Rijndael算法由10个轮构成，每个轮中包括了SubBytes（逐字节替换）、ShiftRows（平移行）、MixColumns（混合列）、AddRoundKey（与轮密钥进行XOR）四个环节。下面以图例介绍一下加密过程中各环节的操作，而解密过程与加密过程类似，只做简单介绍。



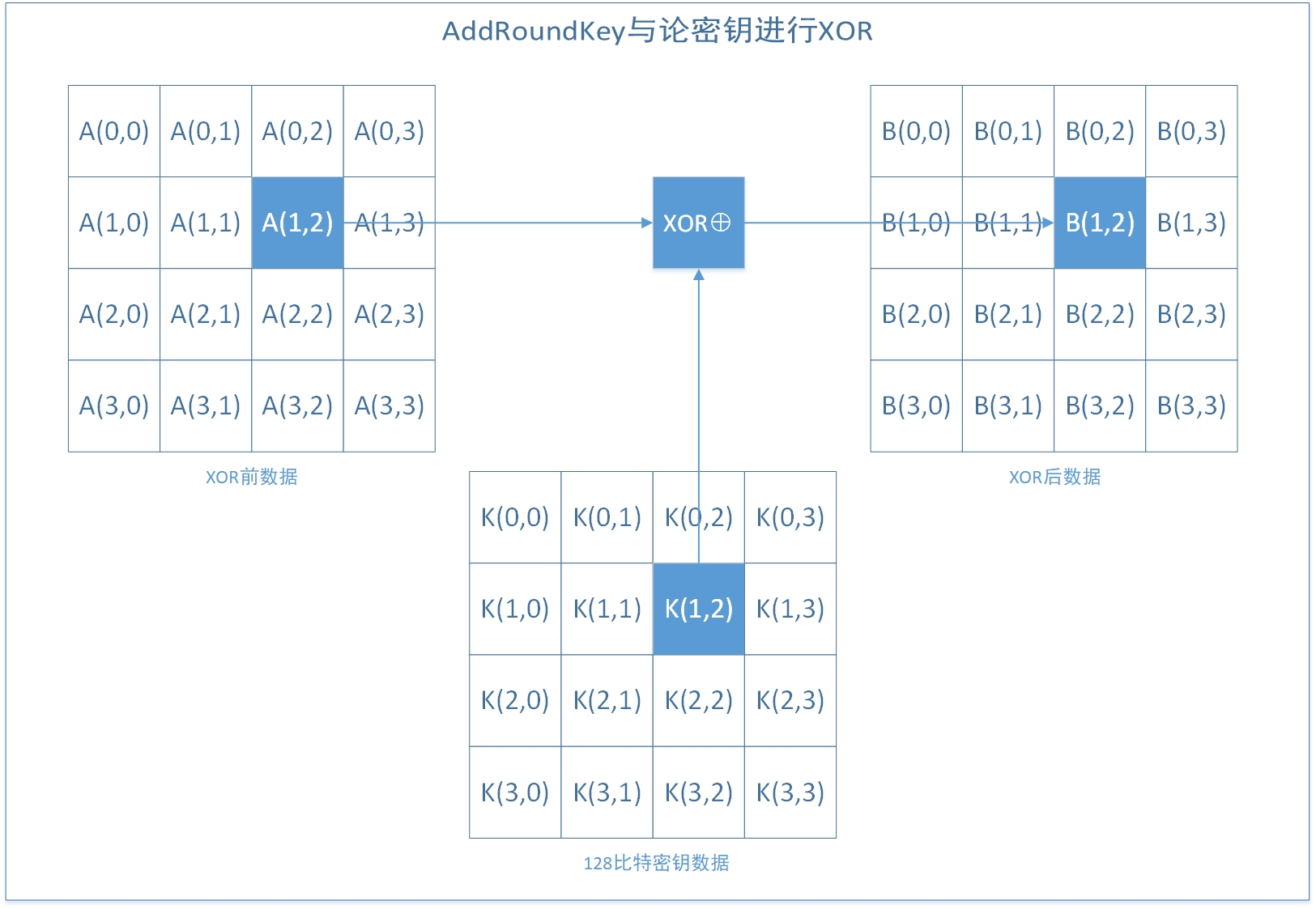
SubBytes环节中内置有S-Box表，其中有256个值。在加密过程中，算法会以每个字节的值（0-255）为索引，从S-Box表中找到索引对应的值，然后进行替换。而解密过程中，算法会以每个字节的值为值，从S-Box表中找到值对应的索引，然后进行替换。



ShifRows环节就是对16个字节中同一行的4个字节进行以字节为基本单位的横向平移。加密过程中，第1到4行都分别向左平移0、1、2、3个单位。解密过程中，第1到4行都分别向右平移0、1、2、3个单位。



MixColumns环节就是对16个字节中同一列的4个字节进行矩阵运算/逆矩阵运算。加密过程中第1到4列通过矩阵运算得到运算后的值。解密过程中，第1到4列通过逆矩阵运算得到运算后的值。



AddRoundKey环节中，加密解密过程都是通过与轮密钥（每一轮轮密钥都不一样）进行异或操作完成的。

在加密过程中，第一轮开始前要先AddRoundKey，然后进行9轮SubBytes -> ShiftRows

-> MixColumns -> AddRoundKey，最后进行第10轮SubBytes -> ShiftRows -> AddRoundKey（此轮没有MixColumns）。解密过程就是加密过程的逆过程。并且加解密都是以AddRoundKey开始、以AddRoundKey结束。

### 1.4、总结

对称密码主要介绍了四种算法：一次性密码本、DES、三重DES、AES。一次性密码本的不实用性前面已经介绍了，下面比较一下后三者的特性。

|  |  |  |  |
| --- | --- | --- | --- |
|  | **DES** | **三重DES** | **AES** |
| **分组长度** | 64 bit | 64 bit | 128 bit |
| **密钥长度** | 64 bit | 196 bit | 128、192、256 bit |
| **密文长度** | 比明文大0-64 bit | 比明文大0-64 bit | 比明文大0-128 bit |
| **破译难度** | 可在现实的时间内破译 | 破译较难 | 暂时无法破译 |
| **运算速度** | 中 | 慢 | 快 |

DES目前已经可在现实的时间内破译，不应该作为加密算法的选择。

三重DES作为DES的改进版，做了三次的DES，密钥的长度也相应的增加了3倍，也就大大增加了破译的难度，但是三重DES由于进行了三次DES，所以运算速度也是比较慢的。因此除了必须要兼容DES的情况，都不使用三重DES。

AES不管是在安全性还是运算速度方面都是这三者中的最优，因此应该优先选择。

## 2、分组密码模式

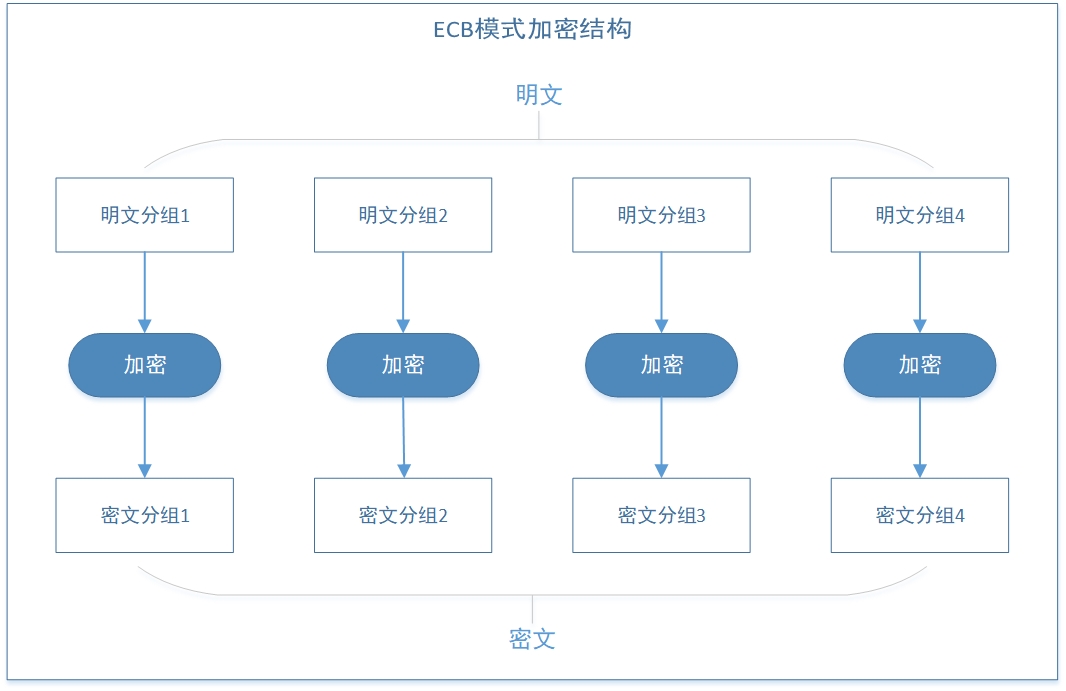
顾名思义，分组密码模式就是分组密码算法在加解密过程中使用的分组模式，而分组模式就是指每一轮的迭代模式。在这里主要介绍5中模式：ECB模式、CBC模式、CFB模式、OFB模式、CTR模式。在本部分我们简化分组密码算法的具体加解密过程，简称为加密和解密。

### 2.1、ECB模式

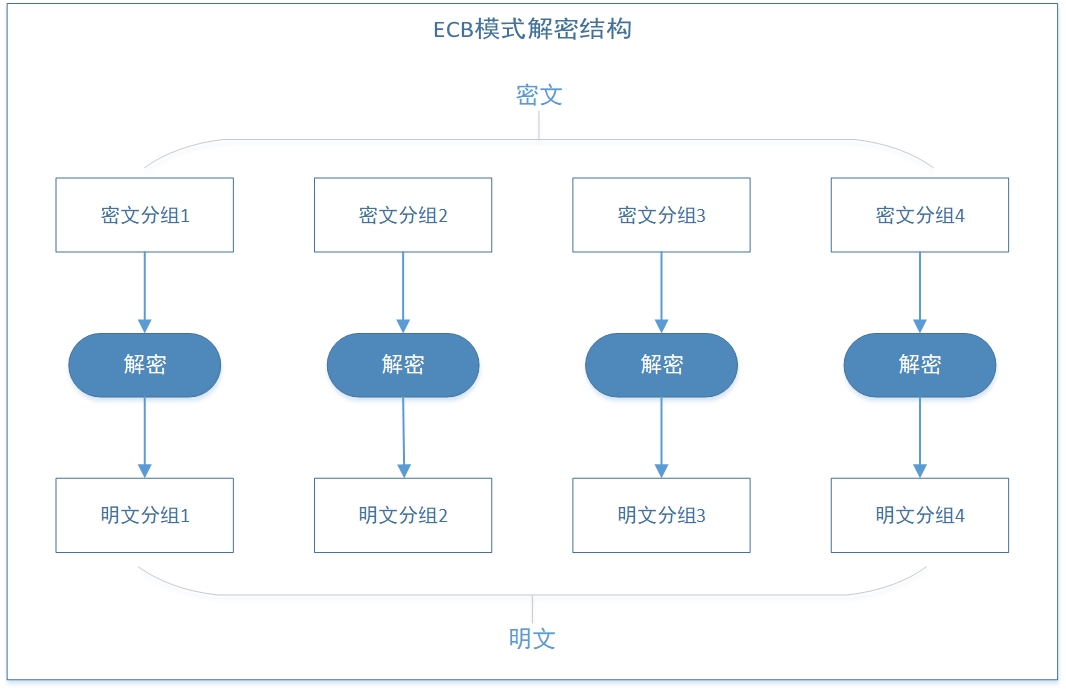
ECB模式的全称为Electronic CodeBook mode，即电子密码本模式。该模式是最简单的分组模式，但由于其显在的弱点，通常不推荐使用。

#### 模式结构

该模式就是将数据分组，然后分别用分组密码算法处理，最后合并的过程。



加密过程中，首先将明文根据分组长度分为若干个明文分组（不足分组长度部分明文进行补码），然后每个明文分组分别进行加密，得到密文分组，最后把密文分组拼接形成密文。



解密过程中，首先将密文根据分组长度分为若干个密文分组，然后把每个密文分组分别进行解密，得到明文分组，在祛除最后一个分组补码后，将明文分组拼接形成明文。

### 2.2、CBC模式

### 2.3、CFB模式

### 2.4、OFB模式

### 2.5、CTR模式

### 2.6、总结

## 3、公钥密码

## 4、单向散列函数

## 5、消息认证码

## 6、数字签名

## 7、证书

## 8、随机数

## 9、总结