有些密钥只产生 4 个不同的子密钥，这些密钥叫作可能的弱密钥，参见表 5-3。

表 5-3　可能的弱密钥列表

可能的弱密钥（十六进制，带奇偶校验位）	
1F 1F 01 01 0E 0E 01 01	E0 01 01 E0 F1 01 01 F1
01 1F 1F 01 01 0E 0E 01	FE 1F 01 E0 FE 0E 01 F1
1F 01 01 1F 0E 01 01 0E	FE 01 1F E0 FE 01 0E F1
01 01 1F 1F 01 01 0E 0E	E0 1F 1F E0 F1 0E 0E F1
E0 E0 01 01 F1 F1 01 01	FE 01 01 FE FE 01 01 FE
FE FE 01 01 FE FE 01 01	E0 1F 01 FE F1 0E 01 FE
FE E0 1F 01 FE F1 0E 01	E0 01 1F FE F1 01 0E FE
E0 FE 1F 01 F1 FE 0E 01	FE 1F 1F FE FE 0E 0E FE
FE E0 01 1F FE F1 01 0E	1F FE 01 E0 0E FE 01 F1
E0 FE 01 1F F1 FE 01 0E	01 FE 1F E0 01 FE 0E F1
E0 E0 1F 1F F1 F1 0E 0E	1F E0 01 FE 0E F1 01 FE
FE FE 1F 1F FE FE 0E 0E	01 E0 1F FE 01 F1 0E FE
FE 1F E0 01 FE 0E F1 01	01 01 E0 E0 01 01 F1 F1
E0 1F FE 01 F1 0E FE 01	1F 1F E0 E0 0E 0E F1 F1
FE 01 E0 1F FE 01 F1 0E	1F 01 FE E0 0E 01 FE F1
E0 01 FE 1F F1 01 FE 0E	01 1F FE E0 01 0E FE F1
01 E0 E0 01 01 F1 F1 01	1F 01 E0 FE 0E 01 F1 FE
1F FE E0 01 0E FE F1 01	01 1F E0 FE 01 0E F1 FE
1F E0 FE 01 0E F1 FE 01	01 01 FE FE 01 01 FE FE
01 FE FE 01 01 FE FE 01	1F 1F FE FE 0E 0E FE FE
1F E0 E0 1F 0E F1 F1 0E	FE FE E0 E0 FE FE F1 F1
01 FE E0 1F 01 FE F1 0E	E0 FE FE E0 F1 FE FE F1
01 E0 FE 1F 01 F1 FE 0E	FE E0 E0 FE FE F1 F1 FE
1F FE FE 1F 0E FE FE 0E	E0 E0 FE FE F1 F1 FE FE

2．3DES

3DES 又称 Triple DES，是 DES 算法的一种变种，它使用 3 个 56 位的密钥对数据进行 3 次加密。3DES 加密/解密过程如图 5-2 所示。

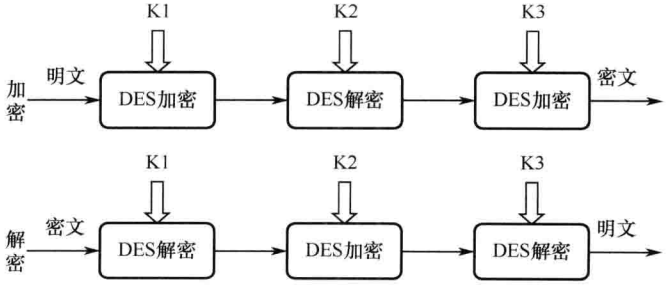

图 5-2　3DES 加密/解密过程

K1、K2、K3 决定了 3DES 算法的安全性。若 3 个密钥互不相同，则 3DES 算法的密钥长度变为 168 位，该算法又称为三倍长密钥的 3DES。如 K1=K3，则 3DES 算法的密钥长度变为 112 位，该算法又称为 2 倍长密钥的 3DES。

3. AES 简介

1997 年 4 月，美国国家标准和技术委员会（NIST）开始征集“高级加密标准”（AES）算法，以便替代 DES 算法。1998 年 5 月，NIST 宣布接受 15 个新的候选算法并提请全世界密码研究界协助分析这些候选算法，包括对每个算法的安全性和效率特性进行初步检验。NIST 考察了这些初步的研究结果，并选定 MARS、RC6、Rijndael、Serpent 和 Twofish 等 5 个算法作为参加决赛的算法。经公众对决赛算法进行更进一步的分析评论，2000 年 10 月，NIST 推荐 Rijndael 作为高级加密标准（AES），并于 2001 年 11 月 26 日发布于 FIPS PUB 197，2002 年 5 月 26 日成为正式标准。

Rijndael 是一种迭代分组密码，采用的是代替 / 置换网络（spn），它对一个 128 位的数据块进行加密操作。加密时，首先将输入的 128 位数据排成 4×4 的字节矩阵，然后根据不同的密钥长度，进行 10（128 位密钥）、12（192 位密钥）或 14（256 位密钥）轮的运算。其 128 位密钥的加密流程如图 5-3 所示。

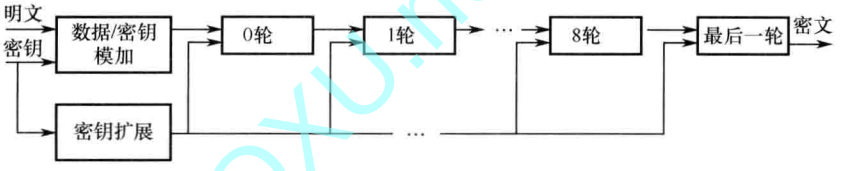

图 5-3 Rijndael 算法 128 位密钥的加密流程（10 轮）

其中，每个轮函数由 4 层组成：第 1 层（盒变换）为非线性层，将一个 8×8 的 s 盒应用于每一个字节；第 2 层（行移位变换）和第 3 层（列混合）是线性混合层，将 4×4 的阵列按行位移，按列混合；在第 4 层（加密钥变换），轮密钥异或到阵列的每个字节。其中，除最后一轮中没有列混合外，其他轮次均相同。密钥扩展的过程根据密钥长度的不同会有所差别。解密过程只是和加密过程稍有不同，除了 s 盒，其他过程分别是加密过程的逆运算。每一轮的流程如图 5-4 所示。

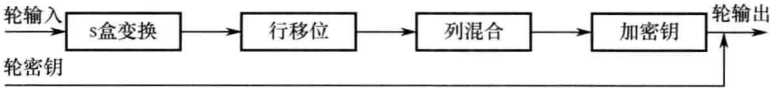

图 5-4 轮变换的过程

4. SM4（原 SMS4）

SM4 算法是由中国国家密码管理局于 2006 年 1 月 6 日发布，在无线局域网产品中批准使用的对称密码算法。

SM4 密码算法是一个迭代分组密码算法。该算法的信息块长度为 128 位，密钥长度为 128 位。加密算法与密钥扩展算法都采用 32 轮非线性迭代结构。SM4 数据解密和数据加密的算法结构相同，只是子密钥的使用顺序相反，解密子密钥是加密子密钥的逆序。

5.1.3 非对称密码算法

本节主要介绍 RSA、ECC、SM2 算法。

1. RSA

RSA 算法是由美国三位科学家 Rivest、Shamir 和 Adleman 于 1976 年提出并在 1978 年正式发表的公开密码算法，其命名取自三位创始人名字的首字母缩写。该算法基于数论中的大数分解难题，即：根据数论，寻求两个大素数比较简单，而将它们的乘积分解开则极其困难。

该算法中，用户有两个密钥：公钥 PK={e，n}和私钥 SK={d，n}，n 为两个大素数 p 和 q 的乘积（素数 p 和 q 一般为 100 位以上的十进制数），e 和 d 满足一定的关系，如只知道 e 和 n 并不能求出 d。

（1）加密/解密过程

若用整数 X 表示明文，用整数 Y 表示密文（X 和 Y 均小于 n），则加密和解密运算为：

加密：　$Y=X^e \bmod n$

解密：　$X=Y^d \bmod n$

（2）密钥的产生

① 计算 n。用户秘密选择两个大素数 p 和 q，计算出 $n=pq$。n 称为 RSA 算法的模数。明文必须能够用小于 n 的数来表示。

② 计算 $\varphi(n)$。用户计算出 n 的欧拉函数 $\varphi(n)=(p-1)(q-1)$。$\varphi(n)$定义为不超过 n 并与 n 互素的数的个数。

③ 选择 e。用户从[0, $\varphi(n)-1$]中选择一个与 $\varphi(n)$互素的数 e 作为公开的加密指数。

④ 计算 d。用户计算出满足公式 $ed=1 \bmod \varphi(n)$的 d 作为解密指数。

⑤ 得出所需要的公钥和私钥：

公钥　PK={e，n}

私钥　SK={d，n}

2. ECC

ECC 是椭圆曲线密码（Elliptic Curve Cryptography）的缩写。1985 年，Koblitz 和 Miller 提出基于椭圆曲线离散对数问题（ECDLP，Elliptic Curve Discrete Logarith Problem）的公钥密码体制，即椭圆曲线密码体制 ECC。它是用椭圆曲线有限群代替基于有限域上离散对数问题公钥密码中的有限循环群所得到的一类密码体制。由于在一般的椭圆曲线群（除了个别特殊的椭圆曲线以外）中没有亚指数时间算法解，所以椭圆曲线密码成了目前最流行的公钥密码体制。

椭圆曲线密码体制 ECC 的安全性基于椭圆曲线点群上离散对数问题 ECDLP 的难解性。而解 ECDLP 最有效的算法则要依靠指数时间的算法。目前对椭圆曲线密码体制最有威胁的方法是 Pollard's rho 方法和 Pohlig Hellman 方法。离散对数的求解是非常困难的，而椭圆曲线离散对数问题比有限域上离散对数问题更难求解，这意味着 ECC 能以更小的密钥长度来产生与其他公钥密码算法相同等级的安全性。为了达到对称密钥 128 位的安全水平，美国国家标准技术研究所（NIST）推荐使用 3072 位的 RSA 密钥。而对 ECC 来说，256 位就可以达到同等的安全水平。

ECC 还具有以下优点：

① 计算量小，处理速度快。虽然 RSA 可以通过选取较小公钥的方法提高公钥处理速度，即提高加密和签名验证的速度，使其在加密和签名验证速度上与 ECC 有可比性，但在私钥的处理速度上（解密和签名），ECC 远比 RSA、DSA（Digital Signature Algorithm）快得多，因此 ECC 总的速度比 RSA、DSA 要快得多。

② 存储空间占用少。ECC 的密钥尺寸和系统参数与 RSA、DSA 相比要小得多，意味着它所占的存储空间要小得多。

③ 带宽要求低。对长消息进行加、解密时，ECC 与 DSA/RSA 密码算法具有相同的带宽要求，但应用于短消息时 ECC 带宽要求却低得多。而公钥密码算法多用于短消息（如用于数字签名和密钥交换），带宽要求低使 ECC 在无线网络领域具有广泛的应用前景。

ECC 算法的基本原理如下。

（1）有限域 Fp 与椭圆曲线

有限域 Fp 是由小于素数 p 的非负整数组成的集合$\{0, 1, 2, \cdots, p-1\}$，其上的运算是模 p 的算术运算。

Fp 上的椭圆曲线是满足方程 $y^2=x^3+ax+b \bmod p$ 的 Fp 上的点（x, y）组成的集合，其中常量 a 和 b 也是 Fp 中的元素。

为了详细描述一条 Fp 上的椭圆曲线，应该给出如下参数：

① 素数 p 的值，为满足 ECC 的安全性要求，p 应该为 160 以上位长的素数；

② 常数 a 和 b 的值，其中 a 可以取−3 以提高点运算效率；

③ 椭圆曲线上的一个基点 G（也称为生成元）；

④ 基点 G 的阶为 n，一般情况下要求 n 为素数且等于椭圆曲线上的点数。

（2）ECC 密钥对

对于给定的椭圆曲线参数$\{p, a, b, G, n\}$，ECC 的私钥 d 为满足 $1<d<n$ 的随机数，相应的公钥 P 为 $P=dG$。为了满足安全性要求，d 必须通过随机或强伪随机数发生器产生。

（3）ECC 签名机制

ECDSA（Elliptic Curve Digital Signature Algorithm）是典型的 ECC 签名机制。

（4）ECC 加密机制

ECIES（Elliptic Curve Integrated Encryption Scheme）是典型的 ECC 加密机制，请参见 ISO/IEC 18033-2。

3．SM2

SM2 算法是由国家密码管理局于 2012 年 12 月 17 日发布的椭圆曲线公钥密码算法，主要满足电子认证服务系统等应用需求。

该算法的主要参数如下：

① 推荐使用素数域 256 位椭圆曲线。

② 椭圆曲线方程：$y^2 = x^3 + ax + b$。

③ 曲线参数：

p=FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 FFFFFFFF FFFFFFFF

a=FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 FFFFFFFF FFFFFFFC

b=28E9FA9E 9D9F5E34 4D5A9E4B CF6509A7 F39789F5 15AB8F92 DDBCBD41 4D940E93
n=FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF 7203DF6B 21C6052B 53BBF409 39D54123
Gx=32C4AE2C 1F198119 5F990446 6A39C994 8FE30BBF F2660BE1 715A4589 334C74C7
Gy=BC3736A2 F4F6779C 59BDCEE3 6B692153 D0A9877C C62A4740 02DF32E5 2139F0A0

5.1.4 摘要算法

本节主要介绍 MD5、SHA1、SM3 算法。

1. MD5

MD5 是 MD4 的改进版，以 512 位分组来处理输入的信息，产生 128 位散列值。

MD5 算法的计算过程主要包括以下步骤。

（1）预填充

首先，对信息进行填充，使其位长对 512 求余的结果等于 448，即信息的位长（Bits Length）被扩展至 *N*×512+448 或 *N*×64+56 个字节（*N* 为一个正整数）。填充的方法为：在信息的后面填充一个 1 和多个 0，直到满足上面的条件。然后，再附加一个 64 位二进制表示的填充前信息长度。经过这两步处理，使消息长度恰好是 512 位的整数倍。

（2）主循环

主循环的次数为信息中 512 位分组的数目。MD5 中需要 4 个 32 位的链接变量：A=0x01234567，B=0x89abcdef，C=0xfedcba98，D=0x76543210。

每个主循环有 4 轮（MD4 只有 3 轮），每轮循环都很相似。在主循环开始前，先将 4 个链接变量复制到另外 4 个变量中：A 到 a，B 到 b，C 到 c，D 到 d。第一轮进行 16 次操作。每次操作对 a、b、c 和 d 中的其中 3 个进行一次非线性函数运算，然后将所得结果加上第 4 个变量、512 位分组数据和一个常数。再将所得结果向右环移一个不定的数，并加上 a、b、c 或 d 中之一。最后用该结果取代 a、b、c 或 d 中之一。

每个主循环完成以后，将 A、B、C、D 分别加上 a、b、c、d，然后用下一个 512 位分组数据继续进行主循环运算，直至所有分组都完成主循环运算。

（3）输出处理

将最后一个主循环生成的 A、B、C、D 这四个 32 位值进行级联，生成一个 128 位的摘要值。

MD5 算法的详细步骤请参见 RFC 1321（The MD5 Message-Digest Algorithm）。

验证 MD5 算法正确性的测试用例如下：

MD5 ("") = d41d8cd98f00b204e9800998ecf8427e。
MD5 ("a") = 0cc175b9c0f1b6a831c399e269772661。
MD5 ("abc") = 900150983cd24fb0d6963f7d28e17f72。
MD5 ("message digest") = f96b697d7cb7938d525a2f31aaf161d0。
MD5 ("abcdefghijklmnopqrstuvwxyz") = c3fcd3d76192e4007dfb496cca67e13b。
MD5("ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789") = d174ab98d277d9f5a5611c2c9f419d9f。
MD5 ("12345678901234567890123456789012345678901234567890123456789012345678901234567890") = 57edf4a22be3c955ac49da2e2107b67a。