Twofish 加密算法及其应用

Principle and Application of Twofish Block Cipher

简清明　（四川理工学院网管中心 643000）

摘要：Twofish 算法是 128 位块加密算法，采用多达 256 位的可变密钥，具备一流的可靠性和抗攻击能力。本文对这一算法的加密过程和应用进行了详细地描述。

关键词：Twofish　块加密算法　源代码

1　引言

在目前计算机处理能力快速增长的情况下，56 位 DES 的安全性已受到公众的质疑。有鉴于此，NIST 于 1997 年开始公开征求下一代的标准加密算法（AES），Twofish 就是其中五种候选加密算法之一。虽然它没有最终被选为新一代的加密标准，但是其优异的加解密性能使其成为一种非常有前途的加密技术。

2　Twofish 算法

Twofish 算法是典型的分组加密算法，即对固定长度的一组明文进行加密的算法。它将明文按一定的位长分组，明文组和密钥组的全部经过加密运算得到密文组。解密时密文组和密钥组经过解密运算（加密运算的逆运算），还原成明文组。Twofish 算法采用的明密文分组长度为 128 比特，支持 128、192、256 比特的密钥长度。Twofish 算法在 Blowfish 算法的基础上吸收了 Square 密码中的 MDS 码矩阵以及 Safer 系列密码中的 PHT 结构而设计的，具有快速、紧凑、简单和可变安全性等特点，已在许多产品中得到实现并进行了大量的分析研究.

2.1　Twofish 的基本组件

Twofish 一般包括以下的基本组件：

（1）Feistel Networks。Feistel Network 的架构于 1973 年在 Horst Feistel 所设计的 Lucifer 加解密算法中提出，至今已广泛的运用在大多数加解密算法中，包含 DES、FEAL、GOST、LOKI、CAST－128 等知名的加解密算法。Feistel Network 常被用来将一个函数（通常称为 F 函数）转换成一个排列，可用非线性函数来表式：

$$F:\{0,1\}n/2 \times \{0,1\}m \rightarrow \{0,1\}n/2$$

其中 n 是 Feistel network 处理的每一个分组大小，F 函数则是用每一分组中 n/2 位数据及 m 位的密钥当作输入，产生长度为 n/2 位的输出。在每一回合中，分组数据会被分成两半，一半称为"来源分组"输入至 F 函数中，F 函数的结果再与另一半的"目标分组"进行 XOR 后输出，然后这二分组对换位置再进行下一个回合运算。通过重复多个回合的操作，我们可以将原本较弱的加密方法增大强度。

Twofish 采用的是一个 16 回合的 Feistel network，使用了一个双射 F 函数。F 函数是一个与密钥相关 64 bits 的排列运算。它包含了三个部份：R_0，R_1 和回合数 r。R_0 经过函数 g 的运算后成为 T_0，R_1 先左旋 8 bits 后再经过函数 g 的运算后成为 T_1，接着 T_0，T_1 再经过 PHT 的组合运算后得到函数 F 的输出值 F_0，F_1：

$$T_0 = g(R_0)$$

$$T_1 = g(ROL(R_1,8))$$

$$F_0 = (T_0 + T_1 + K_{2r+8}) \bmod 2^{32}$$

$$F_1 = (T_0 + 2T_1 + K_{2r+9}) \bmod 2^{32}$$

每一回合的核心 F 函数，均由两个 g 函数构成。函数 g 是整个 Twofish 最重要的部分，其输入 X 是 32 位的数据，分成 4 个字节，每一个字节运算时都有属于自己的 S－boxes，运算完成后得到的结果再输入到一个 4？4 的 MDS 矩阵，可得到一个 32 位的输出结果 Z。整个数学表示式如下：

$$X_i = \lfloor X / 2^{8i} \rfloor \bmod 2^8 \quad i = 0,\ldots,3$$

$$y_i = s_i[X_i] \quad i = 0,\ldots,3$$

$$\begin{bmatrix} Z_0 \\ Z_1 \\ Z_2 \\ Z_3 \end{bmatrix} = \begin{bmatrix} \cdot & \cdots & \cdot \\ \cdot & & \cdot \\ \cdot & MDS & \cdot \\ \cdot & & \cdot \\ \cdot & \cdots & \cdot \end{bmatrix} \cdot \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

$$Z = \sum_{i=0}^{3} Z_i \cdot 2^{8i}$$

其中 MDS 矩阵如下所示(数值用十六进制表示):

$$MDS = \begin{bmatrix} 01 & EF & 5B & 5B \\ 5B & EF & EF & 01 \\ EF & 5B & 01 & EF \\ EF & 01 & EF & 5B \end{bmatrix}$$

(2) S – boxes。S – boxes 是一种非线性的置换运算,常见于分组密码算法中,可以用表格来表示。不同的 S – box 可由随机的方式产生,或用特定的算法产生出来。S – box 输入和输出个数,随分组密码算法的不同而有所不同。Twofish 使用四个 8 ×8 位的 S – boxes。这些 S – boxes 是由两个固定的 8 ×8 位的置换,再加上密钥的数据所产生出来的。

(3) MDS 矩阵。MDS 是一个作用在域上的线性映射,从一个包含 a 个元素的向量映射到有 b 个元素的向量,会产生一个含有 a + b 个元素的合成向量,而这个向量有一个性质就是:任何非零的向量它的非零元素个数至少有 b +1 个。换句话说,假如有两组不同的向量是经由 MDS 运算后所产生出来的,则这两组向量的元素至少有 b +1 个是不同的,并且也能够证明不会有其它种的线性映射能够使得两个向量的最小差值大于 b +1。MDS 通常表示成含有 a × b 个元素的矩阵型态,而 Twofish 本身就使用了一个作用在 GF(2^8) 上 4 ×4 的 MDS 矩阵。

(4) PHT。PHT 是一种可以快速执行的简单混合操作。假设给定两个输入 a 和 b,则 32 位的 PHT 定义为:

$a^1 = a + b \bmod 2^{32}$;

$b^1 = a + 2b \bmod 2^{32}$

Twofish 使用 32 位长度的 PHT 来混合二个并行的 g 函数输出的各 32 位的数据。

(5) Whitening。Whitening 是在第一个回合之前和最后一个回合之后,将密钥的数据和分组数据进行 XOR 的操作。因为通常攻击都是针对第一个 round 的输入到最后一个 round 的输出作分析,而第一个 round 的输入与最后一个 round 输出的实际值已经被隐藏,因而该操作能大大增加攻击的难度。

(6) Key shceduling。Key shceduling 的作用是从源密钥产生 $K_0, \ldots, K_{39}$ 共 40 个长度为 4 字节的扩展密钥以及 4 组相关的 S – boxes. 这些 S – boxes 由 g 函数使用。Twofish 定义了 128、192、256 位三种源密钥长度,短于 256 位的输入密钥用 0 填充至就近长度。Key shceduling 的基本流程如下:设 k = N/64,则源密钥 M 可分成 2k 个 32 位的字,构成两个长度为 k 的字向量,

$M_e = (M_0, M_2, \ldots, M_{2K-2})$, $M_o = (M_1, M_3, \ldots, M_{2K-1})$

第三个字向量 $S = (S_{K-1}, S_{K-2}, \ldots, S_0)$ 由下式确定:

$$\begin{bmatrix} S_{i,0} \\ S_{i,1} \\ S_{i,2} \\ S_{i,3} \end{bmatrix} = \begin{bmatrix} \cdot & \cdots & \cdot \\ \cdot & RS & \cdot \\ \cdot & \cdots & \cdot \end{bmatrix} \cdot \begin{bmatrix} m_{8i} \\ m_{8i+1} \\ m_{8i+2} \\ m_{8i+3} \\ m_{8i+4} \\ m_{8i+5} \\ m_{8i+6} \\ m_{8i+7} \end{bmatrix}, S_i = \sum_{j=0}^{3} S_{i,j} \cdot 2^{8j}$$

其中向量 $(m_0, m_1, \cdots, m_{8k-1})$ 的每个元素由源密钥的相邻 8 位组成,RS 矩阵定义如下:

$$\begin{bmatrix} 01 & A4 & 55 & 87 & 5A & 58 & DB & 9E \\ A4 & 56 & 82 & F3 & 1E & C6 & 68 & E5 \\ 02 & A1 & FC & C1 & 47 & AE & 3D & 19 \\ A4 & 55 & 87 & 5A & 58 & DB & 9E & 03 \end{bmatrix}$$

Me、Mo、S 就是产生扩展密钥的三个要素。$K_0, \ldots, K_{39}$ 由下面的表达式确定:

$\rho = 2^{24} + 2^{16} + 2^8 + 2^0$

$A_i = h(2i\rho, M_e)$

$B_i = ROL(h((2i+1)\rho, M_o), 8)$

$K_{2i} = (A_i + B_i) \bmod 2^{32}$

$K_{2i+1} = ROL((A_i + 2B_i) \bmod 2^{32}, 9)$

一般而言一个区块加解密算法在每个 round 都需要所谓的子金匙(sub – key),这些 sub – key 的总长度往往远大于 key 的长度。我们希望有一个算法能够将 key 分成很多 sub – key,这些 sub – key 是相对于每个 round 进行运算的,彼此互相独立,就好比我们拥有一把长度为 sub – key 总长度的 key 一般,这种将 key 分

成很多 sub – key 的算法就称为? Key Schedule?。由于 Twofish 须要产生许多与 key 相关的数据，因此 key schedule 相当复杂，为了促进分析，Twofish 的 key schedule 使用相同的不可分解函数做为回合函数。

2.2 Twofish 算法的流程

Twofish 算法如图 1 所示，包括十六回合的 Feistel 操作，以及额外的输入/输出部分的 whitening 操作。

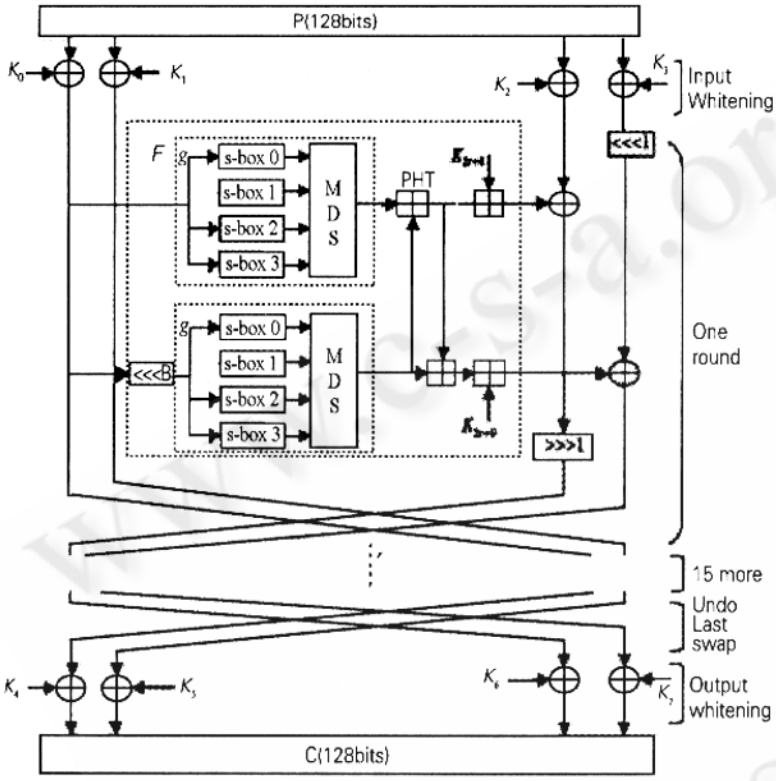

图 1　Twofish 的算法过程

与 DES 的加密与解密的算法基本上一样不同，Twofish 的加密与解密的算法稍微有点不同，但是使用的基本组件是一样的。以加密为例，首先将 128 位长度的明文分成四个 32 位长度的字组（P_0, P_1, P_2, P_3），并分别与长度是 32 位的四个子密钥 $K_0 \sim K_3$ 做 XOR 运算，得到 $R_{0,0} \sim R_{0,3}$ 等四组结果，此即输入 whitening 步骤。

$$R_{0,i} = P_i \oplus K_i \quad i = 0, \cdots, 3$$

接着从事十六个回合的加密动作。首先计算 $T_{r,0}$ 和 $T_{r,1}$

$$T_{r,0} = g(R_{r,0})$$

$$T_{r,1} = g(ROL(R_{r,1}, 8)),$$

$$r = 0, 1, \cdots 15$$

这里 r 代表当前的回合数。

然后将 $T_{r,0}$ 与 $T_{r,1}$ 经由 PHT 之后，再分别加上子密钥 K_{2r+8} 与 K_{2r+9}，得到 $F_{r,0}$ 与 $F_{r,1}$：

$$F_{r,0} = (T_{r,0} + T_{r,1} + K_{2r+8}) \bmod 2^{32}$$

$$F_{r,1} = (T_{r,0} + 2T_{r,1} + K_{2r+9}) \bmod 2^{32}$$

接着再将 $F_{r,0}$ 与 $F_{r,1}$ 分别和 $R_{r,2}$ 与 $(R_{r,3} << 1)$ 进行 XOR 运算，最后将 $R_{r,2}$ 再右移 1 位。如此便得到下一回合的输入数据。

$$(F_{r,0}, F_{r,1}) = F(R_{r,0}, R_{r,1}, r)$$

$$R_{r+1,0} = ROR(R_{r,2} \oplus F_{r,0}, 1)$$

$$R_{r+1,1} = ROL(R_{r,3}, 1) \oplus F_{r,1}$$

$$R_{r+1,2} = R_{r,0}$$

$$R_{r+1,3} = R_{r,1}$$

在最后一个回合之后，进行输出 whitening 操作，亦即将输出的四组 32 位数据分别和 $K_4 \sim K_7$ 进行 XOR 操作，如下所示：

$$C_i = R_{16,(i+2) \bmod 4} \oplus K_{i+4} \quad i = 0, \ldots, 3。$$

其中 ROR 与 ROL 分别代表向右及向左移位的运算。(C_0, C_1, C_2, C_3) 代表 16 个字节的输出密文。

解密的算法与加密的类似，都是采用相同的十六回合动作，用到的 F 函数也是一样的，不过解密时所有子密钥的使用顺序与加密时相反，而且每一回合的动作略有修改。

twofish 算法可以用多种编程语言实现，在 http://www.schneier.com/resources.html 网站上可以得到几种典型的的源代码实现。

3　结束语

twofish 算法以其良好的的快速加解密能力和优异的保密性和抗攻击性在数据加密和网络安全通信等方面得到了广泛的应用。在实际的应用系统中，通常和其他加密技术共同构成一个完整的加密体系。

参考文献

1　Bruce Schneier, John Kelsey, Doug W hiting, David W agner, Chris Hall. Twofish: A 128 – bit block cipher [EB/OL]. http://www.counterpane.com/twofish.html.