一种基于多模式加密算法的文件保护方案*

杨晋吉[1,2]，李东明[1]
(1. 华南师范大学 计算机学院，广东 广州 510631；2. 中山大学 计算机科学系，广东 广州 510275)

摘　要：结合对称密码算法中的 DES、IDEA、AES 和单向散列算法中的 MD5、SHA-1、SHA-256 等算法，提出了一种在同一文件内部采用多模式加密的方案，该方案比传统的单一模式加密的方案能更好地保证数据的完整性和安全性。详细描述了该方案的算法实现，通过试验验证了其实用性，同时分析了该方案的优点和不足。

关键词：对称密码；散列算法；文件加密
中图分类号：TP309.7　　**文献标志码**：A　　**文章编号**：1001-3695(2007)06-0121-02

Solution to File Encryption Based on Multiple Mode Encryption Algorithm

YANG Jin-ji[1,2]，LI Dong-ming[1]
(1. College of Computer Science, South China Normal University, Guangzhou Guangdong 510631, China; 2. Dept. of Computer Science, Zhong Shan University, Guangzhou Guangdong 510275, China)

Abstract: This paper presented a solution to the file encryption which use symmetric encryption methods such as DES, IDEA, AES and hash algorithms such as MD5, SHA-1, SHA-256. Many encryptions can be used to encrypt a file in the solution. Files encrypted by this solution are more secure than those traditional ones. An implement method based this solution was given, its practicability is proofed by experiments. At last this paper discussed some strongpoints and shortages about this solution.

Key words: symmetric encryption; hash algorithm; file encryption

信息技术和网络通信的发展，尤其是 Internet 的高速发展，大大加快了全球信息化的进程，同时信息安全方面的问题越来越多，这也引起人们的高度重视。不解决信息安全问题，信息化将不可能健康地发展。数据加密是保证信息安全的基本技术。它以很小的代价为信息提供了一种有效的安全保护，国内外学者对比进行了有效的研究[1,2]。本文提出了一个基于多模式加密算法的文件保护方案来保证计算机中信息的安全。为了保护文件的保密性和完整性，防止信息被窜改、伪造和假冒，传统的方法是对同一个文件采用单一模式的加密方案。本文提出的文件保护方案是对同一文件不同内容用不同加密方法的多模式加密：采用 128bits(16Bytes) 密钥，使用对称密码算法 3DES、IDEA、AES 中的一种(可选)和类似 CBC(密码分组链接)的加密模式对原文件进行加密以保证数据的安全；另外使用散列算法 MD5、SHA-1、SHA-256 中的一种(可选)对原文件生成散列值，通过校对散列值检测文件是否被更改，从而保证文件的完整性。该方案较传统的方案有更高的安全性。

1　本文涉及的加密和散列函数算法

本文主要使用了对称密码算法 3DES、IDEA、AES 和类似 CBC(密码分组链接)的加密模式，散列算法采用了 MD5、SHA-1、SHA-256 等对原文件生成散列值，通过校对散列值检测文件是否被更改，下面分别进行简单介绍。

1.1　加密算法

DES 用 56bits 密钥将 64bits 的明文转换成 64bits 的密文。其中，密钥总长为 64bits，另外 8bits 是奇偶校验位，详见文献[3]。DES 的密钥存在弱密钥、半弱密钥[3]等。实用中较多是它的改进型，即用最小密钥进行三重加密[4]。IDEA 算法基于“相异代数群上的混合运算”设计思想[5]。其密钥长度为 128bits，是 DES 密钥的两倍多，分组长度为 64bits。一些文献讨论 IDEA 算法的不足[3,5]，但目前尚未有进行有效攻击的方法。AES 算法[1,7]其设计策略是宽轨迹策略(Wide Trail Strategy)。其使用 128、192 或 256bits 长度的密钥，对于 128、192 或 256bits 的加密分组进行加密(密钥跟加密分组长度无须相等共九种组合)。

1.2　密码模式

分组加密算法在加密数据时可以分为四个模式[3]。ECB 是最简单和最快的分组密码模式，当然也是最弱的。本文加密文件头使用该模式。在 CBC 模式下，每一个分组加密之前必须与前一个分组的密文进行一次 XOR 运算，之后再进行加密。因此，每一个区块的加密结果均会受到之前所有分组内容的影响。本文加密文件体系用该模式。

1.3　单向散列算法

本文所用的单向散列函数是消息摘要算法 MD5 和安全散列算法 SHA。MD5 以 512bits 分组来处理输入的消息，且每一分组又被划分为 16 个 32bits 子分组，经过一系列的处理后算法的输出由 4 个 32bits 分组组成，将这 4 个 32bits 分组级联后将生成一个 128bits 的散列值。其运算速度非常快[4]。SHA[7]设计思想和 MD5 相似，但是比 MD5 具有更长的散列值，因此更能够抵抗攻击。SHA-1 有 160bits 散列值。具备扩展转换，并且为产生更快的雪崩效应而将上一轮的输出送至下一轮。

收稿日期：2006-03-22；修返日期：2006-07-03　　基金项目：国家自然科学基金资助项目(60473004)
作者简介：杨晋吉(1968-)，男，副教授，博士，主要研究方向为网络安全、多媒体技术等(yangjj@scnu.edu.cn).

SHA-256 是 SHA-1 的改进: 具有更大的数字指纹、更复杂的非线性函数和压缩函数、每一步均有唯一的加法常数等。因此, SHA-256 具有更高的安全性。

2 强加密算法文件保护方案的实现

下面介绍该文件保护方案的具体实现思想。主要包括文件的加密和解密两个过程。

2.1 文件的加密过程(如图 1, *IV* 为初始向量)

用户选择好要保护的文件、加密受保护文件头部信息的密码算法、加密原文件的密码算法、散列算法、加密密钥。

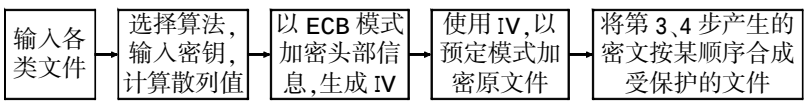

图 1 受保护文件的制作过程

使用用户选择的散列算法求出原文件的散列值, 然后连同原文件的长度、加密原文件的密码算法、散列算法, 这四部分作为受保护文件的头部信息加密后写入受保护文件。头部信息的内容如下: 前 16Bytes 包含了原文件的长度、加密原文件的密码算法、散列算法, 接下来的 16 或 32Bytes 则是原文件的散列值。其中 MD5(128bits) 为 16Bytes, SHA-256(256bits) 为 32Bytes, 而 SHA-1(160bits) 为 16Bytes, 剩余的 32bits 放到前 16Bytes 空闲的第 13 ~16Bytes 中。

使用用户选择的加密受保护文件头部信息的密码算法对头部信息以 ECB(电子密码本) 模式进行加密。在这里需要注意的是无论选择 3DES、IDEA, 还是 AES, 其密钥长度均为 128bits。IDEA 的密钥长度为 128bits, 可直接使用; AES 的密钥长度和分组长度都是可选的, 在本方案中它使用 128bits 的密钥和分组; 对于 3DES, 它使用了文献[3] 介绍的方法, 先用 128bits 的密钥产生三个 64bits 的密钥, 再使用 DES 算法加密。在本方案里, T_1、T_2、T_3 如下所示:

T_1: 0x01, 0x23, 0x45, 0x67, 0x89, 0xab, 0xcd, 0xef;

T_2: 0xfe, 0xdc, 0xba, 0x98, 0x76, 0x54, 0x32, 0x10;

T_3: 0x89, 0xab, 0xcd, 0xef, 0x01, 0x23, 0x45, 0x67。

接下来, 本文使用用户选择的加密原文件的密码算法对原文件进行加密。在加密时用到了一个基于 CBC 模式, 它可以下面的公式表示: $C_i = E(P_i \oplus IV)$; $IV = IV \oplus C_i$; 在这个加密模式中, 第 i 个密文分组是由第 i 个明文分组与初始向量 *IV* 异或后再使用具体的密码算法加密得到的, 然后 *IV* 与第 i 个密文异或得到新的 *IV* 以备对第 $i+1$ 个明文分组加密时使用。最初的初始向量 *IV* 是由受保护文件头部的密码算法产生的子密钥的异或和, 不同的密钥产生不同的 *IV*, 不同的密码算法产生不同的 *IV*。在本方案中, 各个密码算法先产生 128bits 的 *IV*, 在加密原文件时再灵活使用(在加密原文件时, 若用户选择了 AES 算法则 128bits 的 *IV* 可直接使用; 若选择了 3DES 或 IDEA, 则要将 128bits 的 *IV* 分为前后两个 64bits, 再将它们异或得到一个 64bits 的 *IV* 来使用)。

各个密码算法产生 128bits 初始向量 *IV* 的方法: AES 产生 44 个子密钥, 长度为 32bits。将其按顺序分成四组异或得到 128bits 的初始向量 *IV*, 如图 2 所示。DES 先将 128bits 的密钥分成两个 64bits 的密钥, 生成 64 个子密钥(DES 产生 16 个子密钥, 长度为 48bits, 在本方案中一个 48bits 的子密钥被分成两个 24bits 的子密钥), 接着将 64 个子密钥平均分成四组异或得到 128bits 的初始向量 *IV*(图略)。IDEA 产生 52 个子密钥, 长度为 16bits。在本方案中, 使用前 48 个子密钥将其合并成 24 个 32bits 的分组, 然后按顺序分成四组异或得到 128bits 的初始向量 *IV*(图略)。

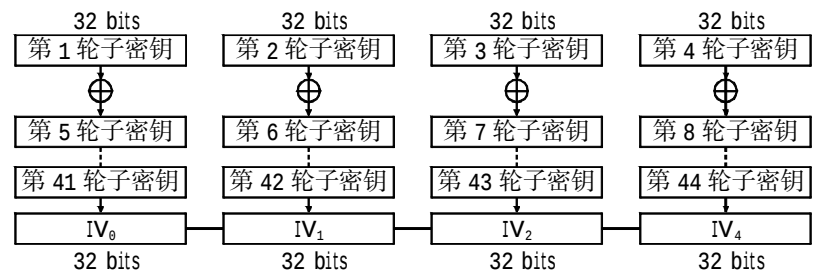

图 2 AES 的 128bits 初始向量 *IV* 的生成

最后要说明的是, 由于 DES、IDEA 的分组长度为 64bits, AES 为 128bits, 若将原文件制作为受保护文件后, 受保护文件的长度是 64bits 的倍数而不是 128bits 的倍数, 那么密码分析者即可确定加密原文件的密码算法不是 DES 就是 IDEA, 这样会降低破译的难度。在加密完原文件后, 将检测受保护文件的长度是否为 128bits 的倍数, 若不是 128bits 的倍数则再加上一些随机数据使其为 128bits 的倍数(是 128bits 的倍数必然是 64bits 的倍数, 那么密码分析者将不能从文件长度中确定密码算法了)。

2.2 文件的解密过程

(1) 软件根据用户提供的解密密钥、受保护文件的头部信息密码算法解密受保护文件的前 16Bytes, 获取原文件的长度、加密算法、散列算法信息;

(2) 根据散列算法产生的散列值的长度读取受保护文件中相应的字节, 将其解密得到原文件的散列值 H_1;

(3) 根据解密受保护文件的前 16Bytes 时得到的原文件加密算法信息和原文件的长度, 解密受保护文件剩余部分, 将解密得到数据写入文件;

(4) 对上一步得到文件用解密受保护文件的前 16Bytes 时得到的散列算法信息求出该文件的散列值 H_2;

(5) 比较 H_1 与 H_2, 若两者相等则表示文件未被修改; 若不等则表示文件已修改。

3 试验结果和该方案的特点

对该方案的实用性进行如下试验: 分别对一个 txt 文件(49.7KB), 一个 MP3 文件(4.95MB), 一个 exe 文件(11.2MB) 进行加密和解密试验, 使用的散列算法: SHA256, 文件头部用 3DES 加密, 文件体分别用 3DES、IDEA、AES 进行加密, 从试验结果可看出该方案的执行时间并没有大幅度的提高, 但安全性有很大提高, 数据如表 1、2(CPU1.1GHz, 256MB 内存)。

表 1 文件加密 单位: ms

	Head 3DES //加密头部信息使用的算法		
	3DES	IDEA	AES
TXT/49.7KB	31	16	31
MP3/4.95MB	3 047	1 266	2 766
EXE/11.2MB	6 875	2 891	6 234

表 2 文件解密 单位: ms

	Head 3DES //加密头部信息使用的算法		
	3DES	IDEA	AES
TXT/49.7KB	31	16	32
MP3/4.95MB	3 109	1 391	2 828
EXE/11.2MB	7 171	3 266	6 422

该方案有以下优点: 本方案的加密模式采用了与 CBC 类似的模式, 如下所示: $C_i = E(P_i \oplus IV)$; $IV = IV \oplus C_i$。(下转第 126 页)

生成和发送新的报文(可选);
}

本文避免了人工实现协议容易出现的一些弱点:首先,自动生成代码保证对所有的 I/O 动作都检查缓冲区是否可能溢出;其次,保证对接收到的所有数据均会有验证代码,这避免了 False Input Attack;再则,通过文献[12]介绍的消息 tag 机制,避免了 Type Flaw Attack;最后,对资源泄露等手写代码常见的非安全性 bug 也通过析构函数 guard 机制进行了保护。

6 总结与展望

Hlpsl2Cpp 的目标代码行为忠实于协议描述,避免了常见的实现弱点,并对常见的通用攻击手段作了防范。目标代码结构清晰、功能明确、可扩展性强,为二次开发打下了良好的基础。基于 ASN.1 的协议报文体系,使得目标代码可以与非 Hlpsl2Cpp 的协议实现互连互通。相对于基于 Java 目标代码的工具,Hlpsl2Cpp 在应用方面具有更大价值。

Hlpsl2Cpp 工具为安全协议的设计、测试、评估工作提供了一个良好的平台,为基于安全协议的应用开发提供了可用的代码框架。笔者对 HLPSL 协议库[1]中的协议进行了测试,证明了该工具是确实可用的。

Hlpsl2Cpp 目前仅支持C++一种目标语言,扩展该工具支持更多的目标输出语言,甚至提供一种方便的目标代码描述模板是十分有用的,这也是笔者下一步的研究方向。同时也将在下一步的研究中,尝试对 Hlspl2Cpp 的转换正确性进行理论证明的探讨。

参考文献:

[1] utomated validation of Internet security protocols and applications FET Open Project IST-2001-39252[EB/OL]. [2005-01]. http://www.avispa-project.org.

[2] PERRIG A, SONG D. A first step toward the automatic generation of security protocols: proc. of Network and Distributed System Security Symposium (NDSS)[C]. San Diego: ISOC, 2000: 73-83.

[3] MULLER F, MILLEN J. Cryptographic protocol generation from CAPSL SRI International[EB/OL]. [2001]. http://www.csl.sri.com/papers/generate/.

[4] MILLEN J K. Common authentication protocol specification language: proc. of the 1996 Workshop on New Security Paradigms[C]. New York: ACM Press, 1996: 132-134.

[5] DIDELOT X. COSP-J: A compiler for security protocols[D]. Oxford: Oxford University Computing Laboratory, 2003.

[6] LOWE G. Casper: A compiler for the analysis of security protocols: proc. of the 10th IEEE Computer Security Foundations Workshop[C]. Los Angeles: IEEE CS Press, 1997: 18-30.

[7] POZZA D, SISTO R, DURANTE L. Spi2Java: automatic cryptographic protocol java gode generation from spicalculus: proc. of the 18th Int. Conf. on Advanced Information Networking and Applications (AINA)[C]. Los Angeles: IEEE CS Press, 2004: 400-405.

[8] TOBLER B, HUTCHISON A. Generating network security protocol implementations from formal specification: proc. of World Comp. Congress Workshop on Certification and Security in Inter-Organizational E-Services (CSES)[C]. Toulouse: Kluwer Academic Publisher, 2004: 102-122.

[9] LUKELL S, VELDMAN C, HUTCHISON A. Automated attack analysis and code generation in a multi-dimensional security protocol engineering framework (SPEAR Ⅱ): proc. of Southern African Telecommunications Networks and Applications Conference[C]. George: Telkom, 2003: 265-270.

[10] BASIN D, M DERSHEIM S, VIGAN L. OFMC: a symbolic model checker for security protocols[J]. International Journal of Information Security, 2004, 4(3): 181-208.

[11] International Telecommunication Union. ITU-T Recommendation X.680 International Standard 8824-1: Abstract Syntax Notation One (ASN.1)[S].

[12] HEATHER J, LOWE G, SCHNEIDER S. How to prevent type flaw attacks on security protocols: proc. of the 13th IEEE Computer Security Foundations Workshop[C]. Los Angeles: IEEE CS Press, 2000: 255-268.

(上接第 122 页)其中,*IV* 是受保护文件头部的密码算法产生的子密钥的异或和。这种加密模式保留了 CBC 的优点——增加了安全性,而且一旦密文遭到修改其错误将会扩散到修改点以后的全部密文,这样可以更加容易地检测到改动。这些方法较好保证了数据的完整性。在一定程度上将三种密码算法混合起来,增加了破译的难度。密文遭修改后,在解密时该错误会被扩大,可更加容易发现密文被改动。

该方案目前也存在着一些不足:密钥过长(128bits),解密时需提供受保护文件头部信息的密码算法。加密保护文件头部信息时只用了电子密码本模式(ECB),降低了安全性。密文被修改后无法正确解密恢复原文,容错性不理想。

4 结束语

该方案用多种模式加密算法对同一文件进行加密。它是对已有的加密方案的有效改进和整合。虽然该方案还有一些不足之处,但较传统单一模式的加密方案其能更好保证文件的安全性和完整性。

参考文献:

[1] 崔国华, 唐国富, 洪帆. AES 算法的实现研究[J]. 计算机应用研究, 2004, 21(8): 99-100.

[2] 张猛, 杨可新, 鞠九滨. 改进加密算法实现的性能[J]. 软件学报, 2001, 12(6): 878-883.

[3] SCHNEIER B. Applied cryptography, protocols, algorithms, and source code in C[M]. Beijing: China Machine Press, 2000.

[4] KNUDSEN L R. Block ciphers-analysis design, applications[D]. Aarhus: Ph. D, dissertation, Aarhus University, 1994.

[5] LAI X. On the design and security of block cipher: proc. of ETH Series in Information Processing[C]. Konstanz: Hartung-Gore Verlag, 1992: 225-248.

[6] RIVEST R L. The MD5 message digest algorithm[S]. [S. l.]: [s. n.], RFC 1321, 1992.

[7] National Institute of Standards and Technology. NIST FIPS PUB 185, Digital Signature Standard[S]. Washington: U. S. Department of Commerce, 1994.