密码系统安全实现及其容侵机制的设计与分析

王玉柱　廖晓峰
（重庆大学计算机学院　重庆 400030）

摘　要　本文分析了密码系统安全实现的动机、任务就及目标，并将容侵思想应用于系统的安全实现，提出了一种基于失败-停止协议的容侵机制的设计方法。

关键词　密码体制，安全实现，容侵，失败-停止协议

Design on Attack-tolerant Scheme in Secure Implementation of Cryptosystems

WANG Yu-zhu　LIAO Xiao-feng
(School of Computer, Chongqing University, Chongqing 400030, China)

Abstract　Secure implementation of cryptosystems has become a new research direction. This paper deals with its motivation, task and goal and then, based on the fail-stop protocol, presents a methold of designing attack-tolerant scheme that active attacks cannot cause the release of secrets within the run of system.

Keywords　Cryptographic scheme, Secure implementation, Attack-tolerant, Fail-stop protocols

1　引言

网络系统最重要的特征之一是信道的不可靠性与通信实体的分散性。随着网络信息交换的日益增加，安全通信已成为全世界关注的问题。现代密码学的发展为网络信息安全提供了理论与技术基础，许多基于密码理论的协议已经用于网络的安全通信，并形成了国际化标准，如会话密钥管理协议IETF RFC 2522，公钥加密标准 IEEE P1363，INTERNET 密钥协商 RFC 2409 等。但是，许多著名的、可证明安全的密码体制被实现后在应用中出现了不安全的情形，如 Bleichenbacher[1]曾经提出了一种对公钥加密标准 PKCS #1 V1.5 的攻击方法。Manger 对 SSL 协议的早期版本提出了一种基于消息篡改的攻击[2]。近年来，这些基于系统实现中存在的漏洞的攻击已经引起研究者的广泛关注，使得密码体制的安全实现问题成为一个新的研究方向。

研究密码体制的安全实现主要出于以下几个理由：

(1)存在理论上安全的密码原语并不等于存在安全的系统实现，因为应用系统的安全性除了依赖于所使用的密码原语之外，还涉及到其它多方面的因素；

(2)理想系统的安全性并不能蕴涵其应用系统的安全性，因为应用系统中不具有理想系统的随机性、独立性和单向性保证，而这些特性正是安全性所依赖的基础，这些特性的部分缺失会对安全性造成什么影响是很难预料的；

(3)密码应用系统的安全性与代码生成、执行过程及运行环境有很大关系。一个算法是安全的并不等于算法的执行过程也是安全的，一个系统在独立运行下是安全的并不等于它在并行环境下也是安全的；

(4)系统安全实现也是对理论安全性的一种补充，利用软件实现的灵活性与可控性使某些理论上可行的攻击在现实中是不可行的，例如在时间、空间以及流程方面强行限制攻击者，或采用专门的检测机制来抵御攻击。

2　密码系统安全实现的任务与目标

密码应用系统与普通应用系统有着本质的区别。传统的应用系统是以需求为牵引的，而且需求是一个确定的、可以明确表达的概念，系统实现的正确性主要体现在它能够满足需求，系统实现的任务就是按照软件工程的理论和方法完成系统的设计、分析、编码、测试等工作。而密码应用系统是以安全为目标的系统，安全性是一个相对的概念，本身具有不确定性和对抗性，这就使得密码应用系统的正确实现变得更加困难。目前，关于密码系统的安全实现技术正处于研究发展阶段，如 Lucks 等人提出了一种密码编译器的思想与体系结构[3]，但是，还没有形成一套成熟的理论和方法。

2.1　安全实现的主要任务

密码系统的安全实现主要包含以下 4 项任务：

(1)建立系统的模型

建立系统模型就是给出系统的一种形式化描述，包括系统的功能、系统中传递的消息、执行流程、安全要求等。能够清楚、准确地描述系统的功能和安全需求是系统实现的基础，为了便于分析系统的安全性，一般采用高度抽象的形式化描述语言，如通用验证协议说明语言 CAPSL[4]。

(2)基于模型的安全性分析

系统实现中的安全性分析是在假设所使用的加密算法是安全的前提下进行的。所谓加密算法是安全的意味着只有知道密钥才能生成密文，只有借助合适的密钥才能对密文解密，不知道密钥既不能生成密文也不能解密密文。基于模型的安全性分析主要运用安全逻辑(如 GNY 等)分析系统的信任情况，检查系统执行中的漏洞和可能存在的攻击，以便进一步修正系统模型，保证系统实现的正确性。

(3)性能分析

统计分析系统的计算量、内存需求量、通信量、多客户环境网络服务器的负载情况等，并将分析结构反馈给模型设计。

王玉柱　博士研究生，主要研究方向为信息安全。

(4)代码生成与验证

根据模型生成相应程序代码。因为在编码实现中仍然可能植入错误,所以,程序的正确性也需要验证。理想的方法是能够自动地由系统描述语言生成程序代码,并进行机器证明。

2.2 系统安全实现的主要目标

系统安全实现的主要目标包括以下几个方面:

(1)系统中交换的消息是可靠的。消息的可靠性是指消息的来源、去向正确,消息未被篡改与重放,而且在规定的时间内传送等。

(2)密钥独立性。密钥独立性是指不能根据已知密钥导出其它密钥,特别是对由 Diffie-Hellman 密钥交换协议派生的共享密钥必须具备密钥独立性。

(3)协议执行的独立性。一个协议的多个执行应相互屏蔽,以防止并行攻击。

(4)临时数据的保密性。对协议执行过程中产生的临时秘密数据要建立保护措施,防止攻击者通过读内存或读磁盘技术来窃取秘密。

(5)执行逻辑的正确性。保证系统完成预定功能,并且遵循了预定的执行逻辑。

(6)容侵性。建立系统安全的最后一道防线,即使受到攻击也不致于泄露秘密。

(7)安全性与效率的合理平衡。在达到所需安全要求的基础上要充分考虑系统的效率,系统的效率决定了系统的生命力。

3 容侵机制的必要性与可行性

3.1 必要性

尽管一个密码应用系统从设计到实现要经过两个层面的安全性分析,即密码算法的计算复杂性分析和系统实现与执行过程的正确性分析,但这些分析都是建立在某些假设基础之上的,如果这些假设是不现实的,则仍然存在攻击的可能。此外,密码应用系统一般都被嵌入到一个更大的开放的系统中运行,对某些未知的和潜在的攻击是难以防范的,通用的安全系统是不存在的。因此,建立密码系统的容侵机制,使得系统即使受到攻击也能将损失降到最低,是必要的也是非常有用的。

3.2 可行性

Gong 和 Syverson 提出了失败停-止协议(Fail-Sto Pprotocol)的概念[5],为容侵机制的实现提供了技术支持。按照 Gong 和 Syverson 的定义,一个失败-停止协议必须满足下列条件:

(1)每个消息都包含一个头部,其中包括消息的发送者与接收者的身份、协议识别码与版本号、消息序列号和以及刷新识别码等;

(2)每个消息都使用通信双方的共享密钥加密;

(3)诚实方遵循协议流程而忽视所有不期望的消息;

(4)如果一个期望的消息不能在指定时间内到达,则终止该协议的运行。

定义 一个协议是一个失败-停止协议,如果该协议执行中的任何一个消息受到攻击干扰,则停止其后的消息接收与发送,终止协议执行。

此外,Gong 和 Syverson 使用类 BAN 逻辑证明了如下两个定理:

定理 1 任何主动攻击都不会使一个失败-停止协议泄露秘密信息。

定理 2 失败-停止协议的性质在顺序执行和并行执行条件下都能够得到保持。

从上面的定义和定理可以看出,失败-停止协议的实质就是强行将主动攻击变为被动攻击。这在一定程度上简化了系统安全实现的难度,实现者可以将更多的精力放到预防被动攻击和内部攻击上,只要系统在被动攻击下不泄露秘密信息,则在主动攻击下也不会泄露秘密信息,这就为容侵功能的实现提供了一个基本方案,通过设计正确的主动攻击行为检测机制,使其中的信息交换满足失败停止协议的条件,就可以达到容侵的目的。

4 基于消息链鉴别的攻击检测方法

基于消息篡改的攻击是最常见的一类主动攻击方法。下面给出基于消息链鉴别的攻击行为检测方法:

对系统的诚实参与方分别建立其内部状态表,其中,状态用安全 HASH 函数值来表示,HASH 函数的输入为该参与者到目前为止与对方交互的信息链,并保持原有的顺序。由于一个协议执行中每条消息的头部除少量数据外都相同,为了减少运算量,消息链只包含一个头部即可,即第一条信息,后面发送的消息只将消息本身加入消息链。发送者每发送一个消息(用 Message 表示)时,除原有的消息内容外,应增加一个状态值域(用 State 表示),用来作存放发送者当前的状态。接收方每收到一条消息时,重新计算当前状态并与发送方的状态进行比较,即检查 $H(key, state, message) = message.State$ 是否成立,如果不成立,则认为消息被篡改。双方的内部状态表由各自进行更新并采取相应的保密措施防止泄露。

关于安全 HASH 函数的条件和构造方法与具体的应用有关,也有许多成熟的技术可供参考,本文不作具体讨论。下面在假设存在安全 HASH 函数条件下来分析该检测方法的正确性,即表明该方法能够正确判定所有主动攻击行为。

根据目前已知的对应用系统的攻击手段和目的,我们可以将一个主动攻击者所能进行的攻击行为归结为以下三类:

(1)获取密钥:因为消息在信道上是以密文形式传送的,要知道其中的内容必须要进行相应的解密操作,所以获取共享的加密密钥是攻击的最佳结果。可是,系统实现阶段的安全性分析的基础是假设所使用加密算法是安全的,所以在这个假设下,攻击者获取共享密钥是不可行的。

(2)篡改密文:主动攻击者最常用的攻击手段是从公共信道上截获密文并进行篡改,如果攻击者能够通过篡改密文而攻击系统,则它必须满足两个条件,第一,篡改后的密文经解密后能够得到有意义的明文,第二,必须通过状态验证。因为加密算法一般是随机的,对任意选择的密文能够得到有意义的明文其概率是很小的,而对于无任何意义的解密来说算不上攻击成功。其次,如果攻击者能够修改密文并通过状态检查,这说明它能够找到 HASH 函数的一个冲突,根据 HASH 函数的抗冲突性这也是不可行的。

(3)消息重放:如果一个攻击者通过重放前面发送过的消息来攻击系统,那么它同时必须能够将接收者的当前状态修改为重放消息的前一个状态,这是非常困难的,因为一方面状态表是诚实通信方的内部数据,可以使用多种技术实施保护,阻止非法访问,另一方面,即使攻击者能够修改接收者的状态,但要将其修改为重放消息的前一状态,其概率大约为(1/

(下转第 110 页)

带来规划求解效率的极大提高，但代价是牺牲了一定的表达能力；PDDL是在STRIPS方法基础上发展起来的，其主要贡献在于给出了一种统一的领域描述语言，并扩充其表达能力，以弥补STRIPS方法在表达能力上的不足；HTN语言则是从任务分解的角度对规划问题进行表示，如果待求解的规划问题很容易进行任务分解，则用HTN语言进行表示，则在规划求解过程中，用HTN规划方法就能快速地产生规划解。但HTN方法的难点在于不容易找到合适的任务分解的方法，通常情况下，对于很多实际规划问题，我们不太容易甚至根本不可能找到合适的任务分解的方法进行求解。

因此，从总体上来看，现代智能规划技术研究的发展趋势主要存在以下两个方向：一是从知识表示的角度出发，寻求新的有效的规划形式表示方法，扩充规划形式表示能力，以表示和刻画复杂的现实规划问题；二是在已有的规划形式表示框架下，寻求高效的智能规划求解算法，并最终设计出性能良好、处理能力强的规划器，以满足实际应用的需要。

参考文献

[1] Green C C. Theorem proving by resolution as a basis for question-answering systems//Meltzer B, Michie D, eds. Machine Intelligence 4, American Elsevier, New York, 1969: 183-205

[2] McCarthy J. Situations, actions and causal laws. Technical report. Stanford University, 1963

[3] Fikes R, Nilsson N. STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 1971, 2(3): 189-203

[4] McDermott D. The 1998 AI planning systems competition. AI Magazine, 2002, 21(2)

[5] McDermott D. The AIPS'98 Planning Competition Committee. PDDL-the planning domain definition language. Technical report. Available at: www.cs.yale.edu/homes/dvm, 1998

[6] Fox M, Long D. PDDL2.1: An extension to pddl for expressing temporal planning domains. Journal of Artificial Intelligence Research, 2003(20): 61-124

[7] Biundo S, Schattenberg B. From abstract crisis to concrete relief——a preliminary report on combining state abstraction and HTN planning // Proceedings of the European Conference on Planning(ECP). 2001: 157-168

(上接第102页)

$2)^{n/2}$，其中，n为HASH函数的输出长度，当n足够大时是可忽略的。除了上面的三种攻击外还有一种是拒绝服务攻击，但由于拒绝服务攻击本身不泄露秘密信息，不影响我们的讨论，因此可以不考虑。

以上的分析与推理说明采用基于消息链的状态检测方法能够准确判定主动攻击行为的发生。因此，只要将上述检测机制附加到交互型密码应用系统中，根据失败-停止协议的原理，就能实现对主动攻击的容侵功能，即使发生攻击也不会泄露秘密信息。

5 基于等待时间的入侵检测方法

在失败-停止协议中，终止协议执行的条件除了发生消息篡改或消息重放行为外，还有一个条件是期望的消息没有在规定时间内到达，因为任何类型的主动攻击都需要一定的时间，纯粹的中间人攻击在很多情况下构不成威胁。因此，可以设计一个消息响应的时间模型，确定协议执行中每一条消息发出后，容许等待回应消息的最长时间值。这个值应根据生成回应消息的计算量、客户或服务器的计算能力、网络的性能与延时等因素来确定。特别是当密码系统用于一个特定的群组时，时间模型能够被较为准确地设计出来。此外，为了防止敌人根据消息的计算时间来发现密钥，例如Kocher在文献[6]中提出了一种通过分析计算指数$m^x \bmod p$所花的时间来发现x的攻击方法，可以对指数运算量进行适当随机化。具体方法如下：

(1)秘密设置初始随机数种子：随机选择$r_1^0 \in Z_p^*$，计算$r_2^0 = (r_1^0)^{-x}$

(2)对$j=0,1,\cdots,N$，第j次计算$m^x \bmod p$的算法为：

① 计算$u=(m \cdot r_1^j) \bmod p, t=u^r \bmod p, t \cdot r_2^j \bmod p (=m^x \bmod p)$。其中，$u$是对消息$m$的盲化，$t \cdot r_2^j \bmod p = m^x \bmod p$是去盲。

② 设置新的种子：$r_1^{j+1}=(r_1^j)^2, r_2^{j+1}=(r_2^j)^2$

该算法的基本原理是，因为r_1^0是秘密选择的随机数种子，对于攻击者来说，在不知道r_1^0的情况下，r_1^1是随机的，同理，如果不知道r_1^0, r_1^1，则r_1^2也是随机的，依次类推，序列$r_1^0, r_1^1, \cdots, r_1^j$具有随机性。当然，在计算中的暂时结果$u, t$和需要保存在内存的种子$r_1^j, r_2^j$不能泄露。

时间模型确定后，通信双方可以内部建立各自的消息容许等待时间表，其中包括对预期消息的开始等待时间，容许等待的最大时间等，用以判断预期接收的消息在到达时间上的合法性，如果在容许范围内没有到达，则自动结束协议，并做相应的处理。这种容侵方案也是对前面基于消息链验证的方案的一种补充，虽然简单，但对那些安全需求较高的应用系统是很有用的，因为这是一种以效率换取安全的使用解决方案。

结束语 密码系统的安全实现是一项复杂的系统工程。本文将容侵思想应用于密码系统的实现，并利用失败-停止协议的基本概念提出了基于消息链鉴别和基于等待时间的容侵实现方法，这种方法的特点是简单实用，计算量的增加也不大，而且具有一般性。不仅可以用于密码协议来保证单个数据包的可靠性，也可用于TCP协议，对整个数据流的可靠性进行检验。需要注意的是应针对具体应用选择安全的HASH函数，并对内存秘密数据进行保护。

参考文献

[1] Bleichenbacher D. Chosen Ciphertext Attacks against Protocols Based on the RSA Encryption Standard PKCS #1 // Hrawczyk. Advances in Cryptology-CRYPTO'98. vol. 1462 of Lecture Notes in Computer Science. Berlin: Springer-Verlag, 1998: 629-660

[2] Manger J. A Chosen ciphertext attack on RSA optimal asymmetric encryption padding(OAEP) as Standardized in PKCS # v2.0 // Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology. London, UK, 2001

[3] Lucks S, Schmoigl N, That E I. The Idea and Architecture of a Cyptographic Compiler // WEWoRC 2005 Coference Records. Leuven, Belgium, 2005

[4] Millen J, Denker G. CAPSL and MuCAPSL. Journal of Telecommunications and Information Technology, 2002, 4: 16-25

[5] Li Gong, Syverson P. Fail-sto Pprotocols: An approach to designing secure protocols // R. K. Iyer, M. Morganti, Fuchs W. K, and V. Gligor, eds. Dependable Computing for Critical Applications. 5. IEEE Computer Society, 1998: 79-100

[6] Kocher P. Cryptanalysis of Diffie-Hellman, RSA, DSS, and other Cryptosystems using timing attacks. In Advances in Cryptology, CRYPTO'95 // 16TH Annual International Cryptology Conference. California, USA, 1995