SSH 缓冲区溢出漏洞与安全防范探讨

宋阳秋
(广东女子职业技术学院 广州 511450)

摘 要 安全 Shell (SSH:Secure Shell)是一种应用层的安全通信协议,提供通信双方相互间身份的认证、通信数据的加解密处理、数据完整性校验等多种安全服务,按照其实现的功能,可归为一种应用层的虚拟专用网(VPN)协议。本文概要说明了 SSH 协议的基本概念,然后对 SSH 会话从发起到结束的整个过程,以及扩展协议进行了深入细致的研究和分析,归纳总结了 SSH 协议本身存在的若干缺陷和不足,并提出一系列实际操作过程中可以参考的建议;对部分安全漏洞,特别是对缓冲区溢出漏洞,进行了研究、分析,修改了其脆弱性,弥补了漏洞,并嵌入了质询-响应认证方法;同时,根据实际应用的需求,基于 Windows 平台,实现了 SSH 协议,方便了系统管理和提高了 SSH 的可用性。

关键词 SSH,网络安全,Windows,密码协议

Discussion of SSH Buffer Overflow Loopholes and Security Ward

SONG Yang-Qiu
(Guangdong Women's Polytechnic College, Guangzhou 511450)

Abstract Safety Shell (SSH : Secure Shell) is an application layer security protocol, provides communication between the two sides of identity authentication, communications data encryption, data integrity verification, and other security services, according to its function, falls into an Application Layer Virtual Private Network (VPN) protocol. This Paper outlines the SSH protocol basic concepts, then launchs from SSH sessions to the end of the whole process and the expansion of the agreement in detail the research and analysis; SSH summarizes the agreement, there is a number of shortcomings and deficiencies, and related work experience, and proposes the proposal that a series of actual work process can refer to;researchs and analyzes some security loopholes, especially for Buffer Overflow Vulnerability, revises its vulnerability to make up for the loopholes, and has embedded authentication methods in response to a question the. In addition, according to the actual needs of the application, based on the Windows platform for the SSH protocol to facilitate management and improve the availability of SSH.

Keywords SSH, Network security, Windows, Cryptographic protocol

1 引言

SSH(Secure Shell)传输层协议是底层的安全传输协议,通常都是运行于 TCP/IP 之上,以该协议为基础,其上可以建立起多种网络安全服务。SSH 传输层协议提供高强度的数据通信加密处理、加密的主机身份认证、数据完整性校验以及数据压缩等多项安全服务;协商确定双方通信所需要的密钥交换方式、公钥密码算法、对称密钥密码算法、消息认证算法和哈希算法等。SSH 传输层协议中的认证是基于主机的(Host-based),并不涉及客户端用户的身份认证。该协议的设计目标简洁、灵活,允许参数协商并力争来回握手次数最少(争取在绝大多数情况下,只需要两个来回(round trips)就完成所有的密钥交换、服务器认证、服务请求及其应答等,最糟糕的情况也只需 3 个来回)。

2 SSH 漏洞及缓冲区溢出问题

SSH 是实现网络安全的非常轻型的工具,不占用客户及服务器的大量资源,便于用户理解和使用,灵活且功能强大等,但 SSH 仅是系统安全的一部分,也有其不足,主要表现在以下一些方面:

(1)主机欺骗危险:允许客户第一次连接一台主机(服务器)时可以直接接受其主机密钥而不检查该密钥是否真正属于该主机。

(2)依赖于主机系统的安全:对主机系统的安全性没有任何提高,一旦主机系统被入侵或控制,SSH 无法保证传输安全。

(3)基于下层的可靠传输:SSH 建立在假设可靠的传输之上(如 TCP),无法避免 Wire Cutter 式的 Dos 攻击。

(4)没有消除隐含信道问题:因为在填充域、SHeMSG-weIGNORE 消息和其他一些地方都可以被利用来传递隐含消息,无法防止夹带非法的数据进出敏感区域。

(5)用户认证协议对底层传输协议的支持要求:用户认证协议要求下面的 SSH 传输层提供数据加密和完整性校验保护,如果系统或者用户配置选项关闭了这些缺省的安全功能,那么其上进行的任何对用户身份的认证很可能没有达到预期的安全效果。用户认证前应检验加密、完整性。

(6)调试消息可能泄漏敏感信息:当设计调试消息的时候要注意它们也许会泄漏大量的主机敏感信息。缺省时可不显示调试消息,或屏蔽一些敏感信息。

(7)远程执行安全:协议允许服务器向客户机,或者客户

宋阳秋 讲师,主要从事网络安全、网络应用开发方面的研究。

机向服务器请求远程执行一条命令或一个程序。

(8)用户非完全透明:尽管 SSH 协议从应用程序的角度来说是应用无关的(只要是面向连接的应用),但是对于用户来说却不是完全透明的,因为它要求使用者产生和分发自己的密钥、选择合适的客户端软件,修改相应的配置,事先知道自己要连接主机的 IP 地址和转发端口等。

(9)密钥管理与分发操作复杂:协议中对用户认证用的密钥没有规定具体的产生、分发和废除等操作,大多数的实现也都是采取人为手工的方式来完成。这样做不够灵活和方便,安全性也不够,在规模加大时更是明显。

(10)安全与管理的矛盾:端口转发可能使入侵者绕过边界安全防护,如防火墙。因为信道是加密的,不能看到其中的数据,所以很难做出反应。

3 缓冲区溢出攻击的防范措施

3.1 缓冲区溢出的攻击方法

程序代码和数据的存储可以放在堆栈中,程序执行前的指令寄存器地址也暂时被放在堆栈中,它们被依次压入堆栈。执行过程中调用参数和函数时,如果数据输入过长,就有可能把预先存储地址的代码区覆盖,在系统再次调用时也就存在错误调用的威胁,执行恶意代码,进而成为一种攻击手段。缓冲区溢出之所以成为远程攻击的主要手段,更因为它能够扰乱具有某些特权运行的程序,使攻击者取得该程序的控制权。如果该程序具有足够的权限,那么有可能控制整个主机,这对 SSH 而言尤其危险,因为 SSH 通信的安全首先取决于系统的安全。一般而言,攻击者攻击 root 程序,然后执行类似"exec(sh)"的执行代码来获得 root 的 Shell。为此,攻击者必须要做的是:在程序的地址空间安排适当的代码;通过适当地初始化寄存器和存储器,让程序跳转到预定的地址空间执行。

在被攻击程序地址空间里设置攻击代码的方法有两种:一是攻击者向被攻击的程序输入一个字符串,程序会把这个字符串放到缓冲区里。这个字符串包含的数据是可以在这个被攻击的硬件平台上运行的指令序列。在这里攻击者用被攻击程序的缓冲区来存放攻击代码。另一种是利用被攻击程序中已经存在的代码,攻击者所要做的只是对代码传递一些参数,然后使程序跳转到目标。

跳转到攻击代码最基本的方法,就是溢出一个没有边界检查或者其他弱点的缓仲区,扰乱程序的正常执行顺序,攻击者用近乎暴力的方法改写相邻的程序空间而直接跳过了系统的检查。

最简单和常见的缓冲区溢出攻击类型就是在一个字符串里综合了代码植入和激活记录。攻击者定位一个可供溢出的自动变量,然后向程序传递一个很大的字符串,在引发缓冲区溢出改变激活记录的同时植入了代码。代码植入和缓冲区溢出不一定要在一次动作内完成。攻击者可以在一个缓冲区内放置代码,只是不能溢出缓冲区;然后通过溢出另外一个缓冲区来转移程序的指针。这种方法一般用来解决可供溢出的缓冲区不够大(不能放下全部的代码)的情况。

如果攻击者试图使用已经常驻的代码而不是从外部植入代码,他们通常必须把代码作为参数。攻击者使用缓冲区溢出改变程序的参数,然后利用另一个缓冲区溢出使程序指针指向 libc 中特定的代码段。

3.2 缓冲区溢出的防范方法

3.2.1 编写正确的代码

编写正确的代码是一件非常有意义但耗时的工作,可以使用一些工具和技术编写安全、正确的程序。最简单的方法就是用 grep 来搜索源代码中容易产生漏洞的库的调用,比如对 strcpy 和 sprintf 的调用,这两个函数都没有检查输入参数的长度,因此用 strncpy 和 snprintf 等替代函数防止缓冲区溢出的发生;利用一些高级的查错工具和静态分析工具来侦测缓冲区溢出的存在。

3.2.2 非执行的缓冲区

非执行的缓冲区技术通过使被攻击程序的数据段地址空间不可执行,从而使得攻击者不可能执行被植入的被攻击程序输入缓冲区的代码。事实上,很多旧的 Unix 系统都是这样设计的,但是近来的 Unix 和 MS Windows 系统为实现更好的性能和功能,往往在数据段中动态地放入可执行的代码,所以为了保持程序的兼容性不可能使得所有程序的数据段不可执行。

非执行堆栈的保护可以有效地对付把代码植入自动变量的缓冲区溢出攻击,而对于其他形式的攻击则没有效果。通过引用一个驻留的程序的指针,就可以跳过这种保护措施。其他的攻击可以采用把代码植入堆或者静态数据段中来跳过保护。

3.2.3 数组边界检查

数组边界检查完全防止了缓冲区溢出的产生和攻击。只要数组不能被溢出,溢出攻击也就无从谈起。为了实现数组边界检查,所有对数组的读写操作都应当被检查以确保对数组的操作在正确的范围内。通常可以采用一些优化的技术来减少检查的次数。目前的检查方法包括:Compaq C 编译器,Jones&Kelly C 语言的数组边界检查,存储器存取检查 Purify,以及安全语言等。

3.2.4 程序指针完整性检查

程序指针完钻性检查是在程序指针被引用之前检测它是否被改变。因此,即便一个攻击者成功地改变了程序的指针,由于系统事先检测到了指针的改变,因此这个指针将不会被使用。与数组边界检查相比,这种方法不能解决所有的缓冲区溢出问题,但是在性能上有很大的优势,而且兼容性很好。

最普通的缓冲区溢出形式是攻击活动纪录,然后在堆栈中植入代码。非执行堆栈可以防范所有把代码植入堆栈的攻击方法,堆栈保护可以防范所有改变活动纪录的方法。这两种方法相互兼容,可以同时防范多种可能的攻击。

剩下的攻击基本上可以用指针保护的方法来防范,但是在某些特殊的场合需要用手工来实现指针保护。全自动的指针保护需要对每个变量加入附加字节,这样使得指针边界检查在某些情况下具有优势。

另外,利用一些工具检测特征代码等,及早发现并针对不同的程序段应用动态连接检查缓冲区溢出漏洞,最大程度地防止溢出攻击的发生。

4 SSH 协议的改进和实现

4.1 SSH 协议中质询-响应认证方式嵌入

在 SSH 协议用户认证过程中,一般采用口令认证或公钥认证。口令认证比较简单,但存在传输过程中数据劫取的威胁,虽然利用 SSH 协议通信过程中建立起来的加密信道可以保护口令数据,但依然潜在被攻破的威胁,而公钥认证也存在两个弊端:一是公钥的生成要消耗和占用系统资源;二是公钥的管理和存储问题一直没能很好地解决。而质询-响应

(CHAP)认证方式可以较好地解决上述问题。

所谓质询-响应认证方式,就是对等实体通过共享一个普通文本密钥来实现网络安全性。这个密钥不会在传输链路上发送,通过如下步骤执行:

(1)链路建立后,主机(认证者)首先发送质询消息给客户(被认证者),该质询消息一般包括标识符(ID)、随机数和本地设备的主机名和远程用户的用户名等;

(2)客户接收到质询消息后,使用单向哈希函数计算出一个质询响应值,密钥是单向哈希函数的输入;

(3)客户发送该质询响应值,包括 ID 的加密版本、计算出的哈希值(作为秘密口令)、随机数及远程设备的主机名和用户名等;

(4)主机接收到响应,查询响应中给定的名称并执行相同的加密操作,确认密钥是否一致,并把得到的哈希值与自己期望的哈希值比较;

(5)如果密钥与哈希值都匹配,则认证成功,发送成功认证消息并建立连接,否则认证失败。在质询-响应认证方式中,远程和本地设备上的秘密口令必须相同,并以安全的方案得到认可、生成和交换。由于秘密口令从不进行传输,因此其他设备无法得到。通过使用一个增量变化的标识和可变质询值,质询-响应认证方式可以防止重放攻击。另外可以使用重复的质询,以限制任何一次攻击的暴露时间,认证者可以控制质询的频率和时间。

质询-响应认证方式中一般使用 MD5 作为单向哈希函数,而共享秘密口令通常以普通文本存储。质询响应认证方式的流程见图 1。

4.2 Windows 平台上 SSH 协议的实现

4.2.1 SSH 协议应用存在的主要问题

现有 SSH 界面多是在 Unix 环境下运行,存在用户界面不友好、配置复杂等问题。而 Windows 是大多数用户比较熟悉的操作环境,在这一界面下完成协议的功能,将能进一步推动其获得更广泛的应用。其功能模块主要包括:

(1)认证方式选择;

(2)密钥产生(包括密钥选择,如 DSA\RSA);

(3)密钥代理(选取密钥、修改、增删密钥);

(4)SSH 配置管理(协议版本(SSH 1/2)及认证方式选择,是否压缩等);

(5)会话管理(已有连接会话的载入,新会话的建立)。

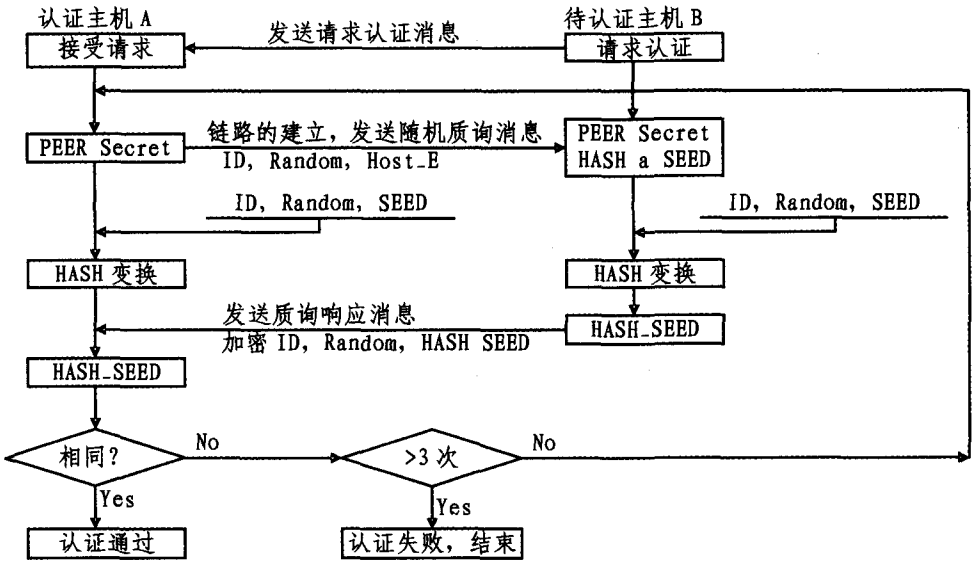

图 1 质询响应认证方式流程

4.2.2 系统设计目标及存在的问题

基于 Windows 平台实现 SSH 协议,要解决的主要问题涉及系统功能定义、终端字符支持、用户界面设定等。关键的几点内容包括:

(1)系统功能包括远程连接、文件的远程传输(SCP 及 SFTP)、端口转发、Xll 转发,以及相关的密钥生成、导入与删除管理等;

(2)终端字符集的支持,包括终端类型、显示区域、字体与编码选择等;

(3)用户界面的设定,主要包括功能选择、系统缺省设置,以及快捷键设置等。

4.2.3 工作实施

分客户端和服务器两个实施阶段,利用已有的条件和资源,分别在客户端和服务器端实施,最后实现系统的整合。为保证接口的一致性,统一定义全局变量和局部变量,以顺利实现变量传递和保证接口的顺利衔接。考虑到为保证服务器端有较高的系统安全性,服务器端的操作系统可以使用 Linux,这要在设计时考虑两种系统之间的互通。

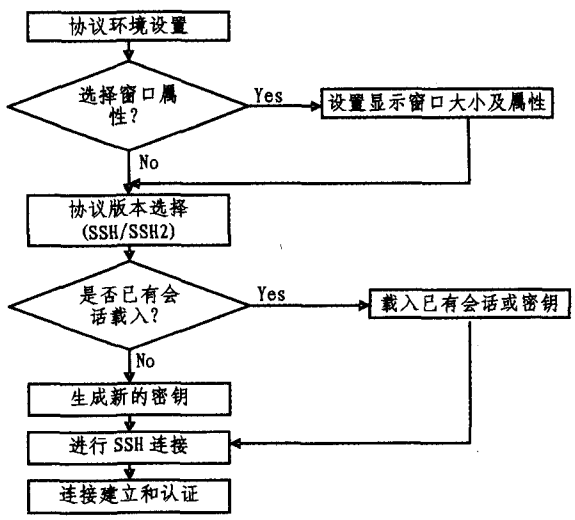

图 2 Windows 环境下 SSH 协议的初始设置流程

(下转第 90 页)

好的 QoS 保障。这是因为，传统的方案只考虑了缓冲队列中的数据长度；Mukul 的自适应方案只考虑了过去的带宽分配情况，而没有考虑到缓冲队列中的数据长度；本文的自适应方案不但考虑过去的带宽分配情况，同时也考虑了当前系统实时业务的缓冲队列中的数据长度，以此来估计发出带宽请求到得到请求回应之间所需的实际带宽，因此可以使更多的数据在相同的时间内发送出去，增加系统吞吐量。

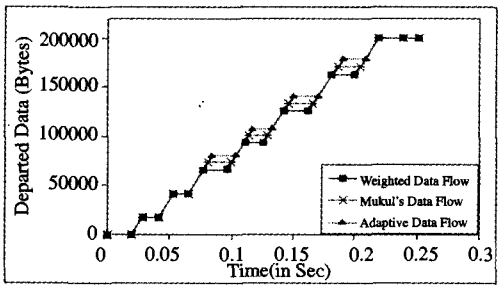

图 1　数据量吞吐量

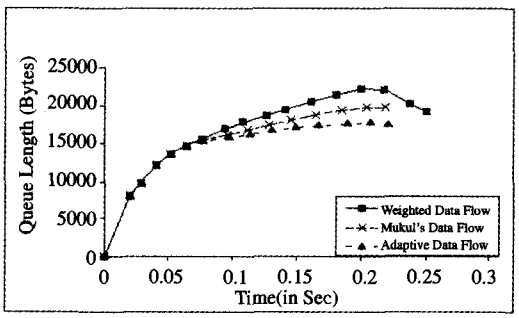

图 2　平均队列长度

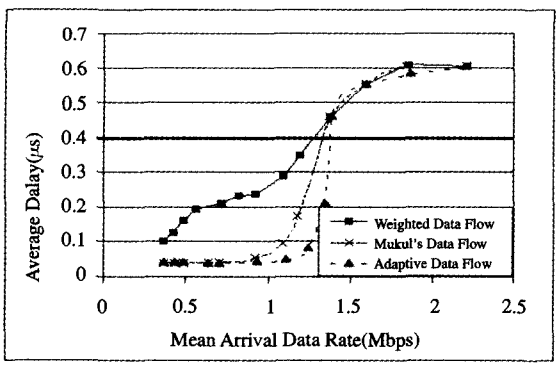

图 3　数据包延迟

图 2 是系统中的实时业务队列的平均长度。可以看出，本文提出的自适应方案比传统方案和 Mukul 的自适应方案需要的队列长度要小。这是因为如上所述，本文自适应方案中数据可以更快地发送出去。也可以看出，同样的数据进入系统时，数据在传统方案和 Mukul 方案的系统中停留的时间更长，而且因为队列长度小，自适应方案需要的缓冲大小也更小。

图 3 显示的是三种方案下的平均时延。平均数据到达速率在 0.5Mbps 到 2Mbps 之间，因为数据到达速率比数据发送速率小，本文的自适应方案与其他两种方案相比拥有更小的时时延。当速率为 1 Mbps 时，时延比传统方案减少了 70%。当数据到达速率接近最大时达到饱和时，三种方案的时延相差无几。

结束语　本文提出了一种 WiMAX 无线城域网系统中的实时业务自适应带宽调度方案，此方案根据当前系统中队列缓冲的数据量与以往 BS 分配的带宽信息，预测在发送请求和应答请求之间所到达的实际数据量，从而有效减少实时业务时延，并提高了系统吞吐量，减少了系统队列长度需求。今后的研究将集中在支持多会话的实时业务与多业务带宽优化等方面。

参 考 文 献

1　Draft of WiMAX Standard. IEEE Standard for Local and Metropolitan Area Networks - Part 16: Air Interface for Fixed Broadband Wireless Access Systems. IEEE 802.16 forum

2　Eklund C, Markas R B, Stanwood K L. IEEE Standard 802.16: A technical overview of the wireless MAN Air Interface for Broadband Wireless Access. IEEE Communications Magazine, June 2002. 98～107

3　Chu GuoSong, Wang Deng, Mei Shunliang. A QoS architecture for the MAC protocol of IEEE 802.16 BWA system. Communications, Circuits and Systems and West Sino Expositions, IEEE 2002. 435～439

4　Cho Dong-Hoon, Song Jung-Hoon, Kim Min-Su, et al. Performance analysis of the IEEE 802.16 wireless Metropolitan Area Network. In: Proceedings of the first International Conference on Distributed Frameworks for Multimedia Applications, IEEE Computer Society, 2005

5　Wongthavarawat K, Ganz A. IEEE 802.16 based last mile wireless military networks with quality of service support. 2003. 779～784

6　Lee Howon, Kwon Taesoo, Cho Dong-Ho. An efficient uplink scheduling algorithm for VoIP Services in IEEE 802.16 BWA Systems. IEEE, 2004. 3070～3074

7　Kumar A, Manjunath D, Kury J. Communication Networking: An analytical approach. 1st Edition. Elsevier, 2004

8　Le Boudec J Y, Thiran P. A theory of deterministic queuing systems for the Internet. Open book, May 2004

9　Mukul R, Singh P, Jayaram D. An adaptive bandwidth request mechanism for QoS enhancement in WiMAX real time communication. In: IEEE Wireless and Optical Communications Networks International Conference, April 2006

10　罗军，袁满，胡建平，等. 下一代网络自适 QoS 研究. 计算机科学，2003(4)

(上接第 87 页)

Windows 环境下 SSH 协议的初始设置流程如图 2 所示。

完成后在 Windows 环境下实现了基于 SSH 协议的安全通信，可以把命令、配置等以简捷的界面给用户提供安全通信。在实践中也证明，Windows 环境下实现了 SSH 协议，既保证了网络的安全性，又实现了操作的简便性，达到了安全的目的。

结束语　根据当前网络技术推广和发展过程中遇到的安全问题，本文有针对性地选择应用层网络安全协议 SSH 作为研究对象，通过分析 SSH 协议的三个层次及其扩展协议，指出了协议本身和各种实现版本中存在的主要问题，并提出了嵌入了质询-响应认证方式，来弥补协议中的漏洞。在 Windows 环境下实现 SSH 协议，以增强 Windows 系统的安全性。

参 考 文 献

1　圣安妮. 卡利斯科著. SSH: UNIX Secure Shell 工具. 张蓬，匡巍，张建杰，等译. 北京：机械工业出版社，2004

2　戴关侠，连一峰，正航. 系统安全与入侵检测. 北京：清华大学出版社，2004

3　李俊照，王浩，徐栋哲. 基于 SSH 协议的集群构建与性能测试[J]. 计算机工程与应用，2005

4　丁晓峰，李周贤，刘炳华，顾巍，吴楠宁. 在 SSH 协议下的入侵检测[J]. 现代图书情报技术，2005

5　赵秀文，罗平，陈强，张宁. 基于 SSH 和 LDAP 的分布式安全文件系统[J]. 计算机应用研究，2005