文章编号:1001 - 9081(2006)11 - 2544 - 03

基于网络隔离系统的业务代理设计

李 捷,汪海航,谭成翔
(同济大学 计算机科学与工程系,上海 201804)
(skyuniverse2002@yahoo.com.cn)

摘 要:基于网络隔离技术,给出了一种网络隔离系统的体系结构。着重阐述了在这一特殊架构下的业务代理模型设计,并说明了在网络隔离环境下的代理模型与一般代理服务器的区别,最后实现了 HTTP 协议透明代理的功能。

关键词:网络隔离系统;透明代理;HTTP 应用代理
中图分类号: TP393.02 **文献标识码**:A

Design of proxy based on network isolation system

LI Jie, WANG Hai-hang, TAN Cheng-xiang
(*Department of Computer Science and Engineering, Tongji University, Shanghai* 201804, *China*)

Abstract: The architecture of network isolation system based on network isolation technique was proposed. And design of its proxy part was discussed in detail. Then the difference between the proxy pattern and traditional proxy server was specified. The HTTP transparent proxy was realized in the end.

Key words: network isolation system; transparent proxy; HTTP proxy

0 引言

面对新型网络攻击手段的出现和高安全度网络对安全的特殊需求,全新安全防护防范理念的网络安全技术——“网络隔离技术”应运而生。该技术的应用十分广泛,尤其在政府部门、银行和重要的商业机构中更有着其特殊的需求。因此国家建立了 2005 信息安全专项,专门研究“网络隔离技术”在隔离设备中的应用。本文便是在此背景下,研究在隔离设备中实现业务代理功能,满足隔离网络之间进行可控数据交换的需求。

本文将首先介绍“网络隔离技术”,然后根据自身课题背景着重阐述在“网络隔离系统”这一特殊架构下的业务代理模型设计,并说明在“网络隔离”环境下的代理模型与一般代理服务器的区别,最后以 HTTP 协议代理的实现作为实例进行详细的论述。

1 网络隔离技术

网络隔离技术的目标是确保把有害的攻击隔离在可信网络之外和保证可信网络内部信息不外泄的前提下,完成网间数据的安全交换。该技术的发展经历了 5 代:完全的隔离、硬件卡隔离、数据转播隔离、空气开关隔离和安全通道隔离。其中前 3 种属于桌面级防护,对单客户机或服务器进行隔离保护;后两种则是企业级防护,一般以透明防毒网关的形式对企业或政府内部网络进行保护和监控。

目前,第 5 代的“安全通道隔离”技术是国内外研究的主流技术,此技术通过专用通信硬件和专有安全协议等安全机制,来实现内外部网络的隔离和数据交换,不仅解决了以前隔离技术存在的问题,有效地把内外部网络隔离开来,而且高效地实现了内外网数据的安全交换,透明支持多种网络应用。基于安全通道隔离技术而研发出的“网络安全隔离与信息交换系统”便是一种具有代表性的网络隔离系统,这也是我们课题研究核心的模块之一。

2 系统架构

该网络隔离与信息交换系统采用了三层次的体系结构——内网主机、控制单元和外网主机。如图 1 所示,每一个部分都采用独立的安全硬件和安全操作系统,主机之间则采用非 TCP/IP 的安全通信协议进行通信。内网主机由网络接口、防火墙和内网代理服务器等几部分构成,其主要功能是与受保护的可信任网络内的客户端进行通信,接受内网客户端发起与外网进行数据交换的受控业务请求,将检查后的安全请求信息摆渡到控制单元处理,将控制单元返回的安全响应信息发送给客户端。控制单元是由嵌入式的硬件与经过裁剪的 Linux 操作系统组成,其作用是控制与内网主机和外网主机之间的链路通断,并实现纯数据交换,确保在内网无连接的情况下,内网主机保持与外网的隔离状态。外网主机由外网代理服务器、防火墙和网络接口几个部分组成,直接与 Internet 相连,部署在不可信任域中。其主要功能是向外网应用服务器发起业务请求,并将响应信息进行安全检查后发送给控制单元;具有防火墙和病毒扫描的功能等。

这种三层次的防御架构具有较强的安全保障。首先其自身拥有较高的安全性,外网接口与内网接口是由两个独立的操作系统控制,且控制单元又能保证内外网主机间的通断。这样即使外网主机系统完全被攻陷,黑客也无法控制和破坏内网。其次,内外网主机与中间交换控制单元使用的非 TCP/IP 协议进行通信和链路的通断控制,这样外网就不能取得内

收稿日期:2006 - 05 - 23;**修订日期**:2006 - 07 - 10 **基金项目**:国家信息安全专项基金资助项目([2005]1879)
作者简介:李捷(1983 -),男,江西弋阳人,硕士研究生,主要研究方向:网络安全、电子商务; 汪海航(1965 -),男,浙江奉化人,教授,博士生导师,主要研究方向:网络安全、电子商务; 谭成翔(1965 -),男,湖北荆州人,研究员,博士生导师,主要研究方向:网络安全、电子商务.

网的路由信息,从而确保网络之间是隔离的。再次,这种设计最关键的部分保证了内外网主机与中间的控制单元是纯应用数据交换,代理模块将进行详细的协议分析,将连接信息与应用数据分离传送给控制单元,从而隔断了外网向内网发起的协议攻击,这样诸如 TearDrop,Land,Smurf 和 SYN Flood 等网络攻击包都不能到达内网。最后,中间控制单元是内外网之间数据交换的唯一通道,它控制着连接的通断,而且对应用数据进行详细的内容检查;提供管理控制台的配置和日志记录接口,可配置安全策略规则,这样就确保每次数据交换都是可信的和可控制的。

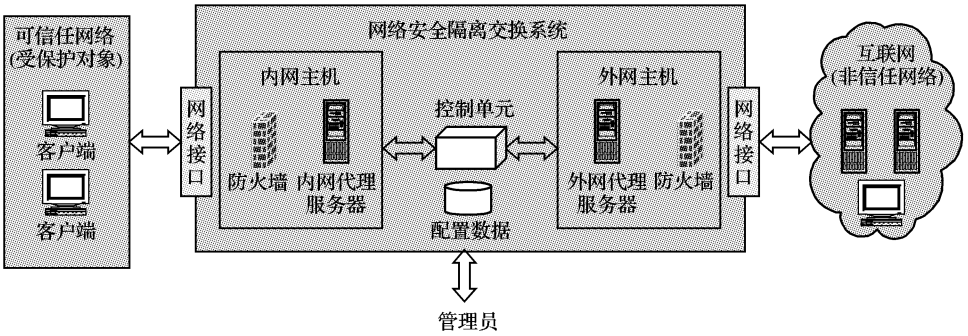

图 1　隔离系统总体架构

3　代理模块设计

代理模块的总体目标是:实现内网与外网之间可控的业务数据交换,对基于 TCP 和 UDP 的应用层协议进行协议分析,保证系统内部主机之间是纯应用数据交换。

根据隔离器的三层安全体系结构,代理模块与一般的代理防火墙设计有着比较大的区别,分为两部分:内网代理子模块和外网代理子模块,它们分别部署在内网主机和外网主机上。整个代理的功能需要多机之间协同工作完成,并且内外网主机之间是非 TCP/IP 通信的方式,因此内外网的代理模块间不能通过 SOCKET 进行通信,这将是代理模块设计的重点和难点,为此采用图 2 所描述的设计结构进行处理。

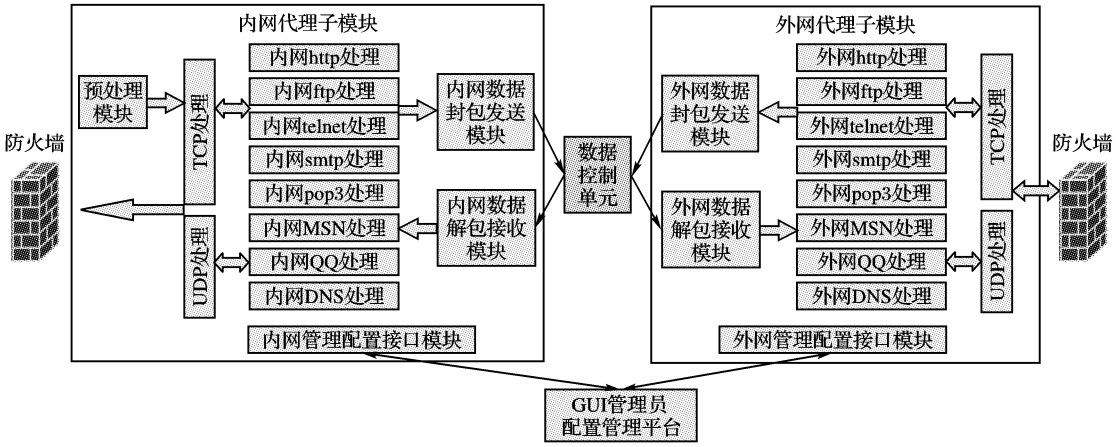

图 2　代理模块整体结构

基本设计思想如下:

在内网代理子模块中,首先通过一个预处理模块将基于面向连接的 TCP 协议和无连接的 UDP 协议的应用层协议分开进行处理,TCP/UDP 处理模块分别处理各自的客户端的请求数据。对于 TCP 协议,模拟服务器响应客户请求,与客户端建立连接;对于 UDP 协议,则直接接受客户端的 UDP 报文处理。然后这两个处理模块都要根据端口号判断出请求的服务是属于哪一种应用协议,再将数据传给对应的应用协议分析模块处理。不同的应用协议处理模块将根据自身协议的特点和控制台的安全策略,对应用协议命令字进行分析和过滤等操作,并将客户端的连接信息和请求内容发送给内网数据封包发送模块处理。最后内网数据封包发送模块将根据自定义的封包格式和内部自定义的传输协议将数据交给数据控制单元处理。同样的内网数据解包接收模块等待接受控制单元发送回的应用服务响应数据,并将数据还原发送给内网特定的应用协议处理模块处理,最后通过 TCP/UDP 处理模块将数据发送给内网客户端。

在外网代理子模块中,外网数据解包接收模块一直在等待接数据控制单元摆渡来的自定义封包的应用数据信息,并将其解包还原转入对应的应用协议代处理块进行处理。各应用协议进行相应处理后将请求信息和连接信息发送给 TCP/UDP 处理模块处理,对于 TCP 协议,其模拟客户端向 Internet 中的应用服务器发起请求,建立外网主机与应用服务器的连接,并等待接收应用服务器的响应信息;对于 UDP 协议,则直接将请求信息包转发给对应服务的应用服务器,并等待接收其返回的数据。一旦服务器的响应数据到达后则转入对应应用协议处理模块处理,各协议处理模块将根据管理控制台的安全策略对响应的数据进行进一步的协议分析和安全检查,并将合法的响应内容和外网连接交给外网数据封包发送模块。最后外网数据封包发送模块将应用数据封包给数据控制单元处理。

下面将以 HTTP 应用协议代理的实现为例,对代理的设计进行详细的说明。

4　HTTP 代理实现

4.1　访问流程描述

以内网客户端访问外网的一个网站为例,说明客户端从发出访问请求到获得服务器发回的响应数据整个流程中,隔

离器是如何处理这些数据包的。

一次完整的网页访问流程如下:1)客户端发出 DNS 包,请求获得域名为 www. aaa. com 的 IP 地址; 2)隔离器的代理模块接受 DNS 包,并代理向 DNS 服务器发出 DNS 请求; 3)隔离器接收 DNS 服务器返回的包含 Web 服务器地址的 DNS 包; 4)隔离器发送 DNS 响应包给客户端; 5)然后客户端将返回的 IP 地址加到 IP 包头上,并向服务器发起连接请求; 6)隔离器模拟 Web 服务器完成与客户端的 TCP/IP 握手,建立连接; 7)隔离器接收客户端的 HTTP 的请求,GET www. aaa. com/a. htm; 8)隔离器向目的 Web 服务器(202. 96. 209. 11)发出连接请求; 9)模拟客户端和服务器完成 TCP/IP 握手,建立连接; 10)向 Web 服务器发送 HTTP 请求; 11)Web 服务器响应隔离器请求,发送 a. htm 文件给隔离器; 12)隔离器获得服务器响应后,对响应内容进行安全检查,并发送给客户端。

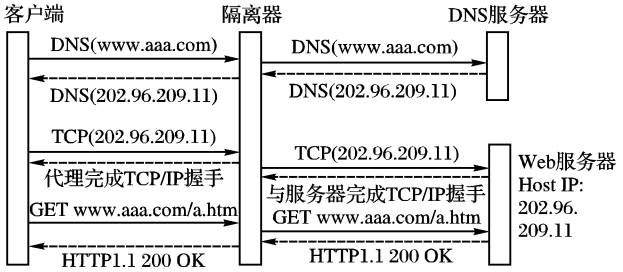

图 3 一次完整的网页访问流程

4.2 内网代理子模块的实现

根据图 3 的 HTTP 业务流程描述,内网代理子模块需要预处理、TCP 处理、UDP 处理、HTTP 处理、DNS 处理以及数据封包和数据解包等几个部分协同工作完成,模块结构如图 2 所示。代理本身运行在 Linux 操作系统上,利用 IPTABLES 的 NAT 机制实现客户端的透明模式访问。

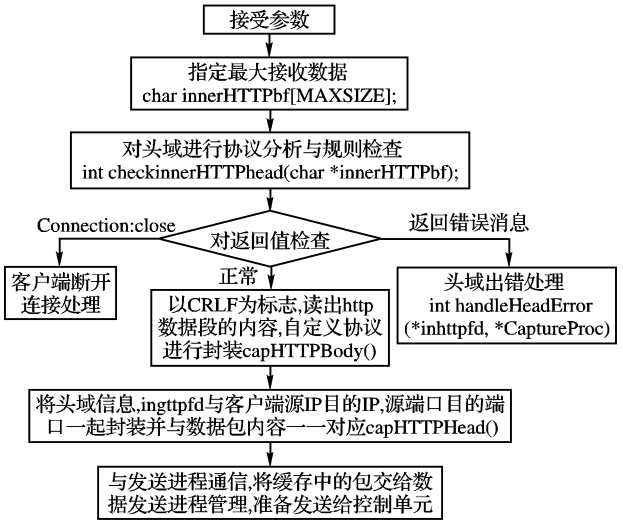

图 4 内网 HTTP 处理流程

预处理: 1) 利用 IPTABLES 脚本,将所有内网的基于 TCP 的请求重定向到系统指定监听端口 INIT_PORT。Iptables -t nat -A PREROUTING -p tcp -j REDIRECT --to-ports INIT_PORT。2) 系统监听并绑定 INIT_PORT 端口,等待处理连接请求。3) 系统同时绑定 DNS(53)端口,利用 select 多路复用的机制并行处理 TCP 和 UDP 请求。4) 若有新的请求到来,则转入相应的 TCP 处理或 UDP 处理模块。

TCP 处理: 1) 循环接受处理新的请求,同时完成与客户端连接的建立。2) 记录客户端的源 IP 和源端口号。3) 利用自定义的函数 self_getsockname(int, struct sockaddr, socklen_t),来获取客户端真实访问的服务器地址和端口。4) 对真实的端口类型进行判断,若是 HTTP 请求,则开辟新的线程进行 HTTP 协议分析。

UDP 处理:直接使用 recvfrom( ) 函数取得 UDP 报文和客户端源 IP 和 PORT,对 UDP 报文安全检查后直接交给数据封包模块处理;然后用 sendto( ) 函数将外网摆渡来的响应数据发送给客户端。

HTTP 处理和封包处理:在 HTTP 处理中主要是对其请求头域信息进行协议分析和安全检查,然后将其头域、纯数据段信息和连接信息一起按照自定义的协议进行封包,等待发送。

解包接收处理:该模块将以守护进程的方式运行,以共享内存的方式与内网代理进程进行通信;一旦控制单元传来响应数据,该模块就将数据进行还原,分解出应用数据和连接信息,并交给代理模块处理。

4.3 外网代理子模块的实现

外网的解包接收与封包发送模块与内网单元的功能类似,只是数据接收和发送的对象不同,下面主要介绍外网 HTTP 处理的流程。

1) 外网 HTTP 处理

接收到内网摆渡来的数据后,对其连接信息要进行判断。若外网主机已经与 Web 服务器建立了连接,则直接将内网请求的信息发向 Web 服务器;若是新的连接请求,则交给 TCP 处理其连接,用 connect( ) 函数向目的服务器发起连接。

当 Web 服务器有响应数据时,首先提取与服务器的连接信息,然后将响应的数据进行头域分析和安全检查,最后将数据与连接信息传给外网数据封包模块处理。

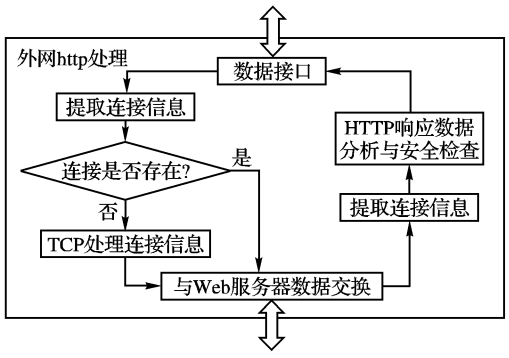

图 5 外网 HTTP 处理

2) UDP 处理

直接使用 sendto( ) 函数将 DNS 的 UDP 报文发向指定 DNS 服务器,并用 recvfrom( ) 等待接收响应的数据,对响应的 UDP 报文安全检查后直接交给外网数据封包模块处理。

参考文献:

[1] 公安部计算机信息系统安全产品质量监督检验中心. MSCTC-GFJ-04,信息技术网闸产品安全检验规范[S], 2003.

[2] GB/T 17900-1999,网络代理服务器的安全技术要求[S],1999.

[3] BERNERS-LEE T. RFC1945, Hypertext Transfer Protocol HTTP/1.0[S], 1996.

[4] FIELDING R, GETTYS J, MOGUL JC, *et al*. RFC2616, Hypertext Transfer Protocol HTTP/l.1[S], 1999.

[5] MOCKAPETRIS P. RFC1035, Domain Names Implementation and Specification[S], 1987.

[6] STEVENS WR. UNIX 网络编程:第二卷 进程间通信[M]. 第 2 版. 杨继张, 译. 北京:清华大学出版社, 2000.