第33卷 第15期 计 算 机 工 程 2007年8月
Vol.33 No.15 Computer Engineering August 2007
·安全技术· 文章编号：1000—3428(2007)15—0165—02 文献标识码：A 中图分类号：TP393

基于端口复用技术的木马研究

罗改龙[1]，程胜利[2]
(1. 江西蓝天学院计算机系，南昌 330098；2. 武汉理工大学计算机学院，武汉 430000)

摘 要：隐藏技术是木马的关键技术之一，其直接决定木马的生存能力。针对目前木马在隐藏技术方面的普遍缺陷，提出了端口复用的概念，并在实验室研制出了一个具有无进程和端口复用特征的木马。该木马通过利用 Windows 的 WinSock 2 的新特性服务提供者技术，在操作系统中插入了一个分层式服务提供者，每当网络应用程序进行 WinSock 2 的调用时，木马将能获得系统控制权，并通过一个系统已经打开的合法端口与外界进行网络通信。由于该木马不需要打开一个新的端口，因此具有更强的隐藏性。

关键词：木马；端口复用；服务提供者

Research on Trojan Horse Based on Port Reuse Technology

LUO Gai-long[1], CHENG Sheng-li[2]
(1. Computer Department, Jiangxi Bluesky University, Nanchang 330098;
2. Computer Department, Wuhan University of Technology, Wuhan 430000)

【Abstract】 Concealment technology is critical to a Trojan Horse, which decides the survivability ability of the Trojan Horse. The concept of port reuse is brought forward for the common disfigurement of the Trojan Horse in concealment technology. And a Trojan Horse without process or port reuse is made in the lab. The Trojan Horse can insert a layered service provider in operating system by making use of new characteristics of WinSock 2 of Windows, thus it will get the control of the operating system and communicate with outside in an open legal port when network application program makes a call of WinSock 2. For the Trojan Horse which does not have to open a new port in comparison with traditional ones, its concealment ability is stronger.

【Key words】 Trojan Horse; port reuse; service provider

1 概述

木马是驻留在目标机器上的程序，该程序主要在目标机器上执行一些未经授权的操作，如窃取口令、银行帐号和密码以及控制目标计算机等，因此，木马程序普遍具有非法性。木马程序为了避免被发觉，大多会采用各种方法隐藏自己。最近几年，木马技术发展很快，其隐蔽性也越来越强。下面是一些新型的木马技术：

(1)反弹端口木马[1]。此类木马主要是改传统木马的被动监听方式为主动连接。

(2)基于窗口 HOOK、挂接 API 或远程线程技术动态嵌入式木马。

(3)基于动态链接库转发的木马[2]。

(4)协同隐藏型木马[3]。该类木马主要是综合利用多种隐藏技术。

上述 4 类木马技术是当前非常流行的木马技术，特别是类型(2)和类型(3)，可以将木马线程植入合法进程体内，利用合法进程与外界通信，很具迷惑性。但是上述木马均有一个共同的弱点，那就是要打开一个新的端口。所以加强对端口的监控仍然是对付这些木马的有效方法。对于类型(2)和类型(3)，还可以采用应用程序和端口相关联的技术来进行检测。如果一个合法的进程打开了一个它不属于它的端口，那么一定是感染了木马。

端口监测是当前防火墙普遍采用的核心技术之一，如果木马程序能够不打开新的端口，而是直接利用系统已经打开的端口(笔者把这种情况称为端口复用)与外界进行通信，那么隐蔽性将更强，也就更容易穿过防火墙。本文以 SPI 为基础，实现了一个具有端口复用特征的木马。

2 端口复用的实现原理

对于基于 UDP 或 TCP 的网络应用程序进行通信时，首先必须将本地 IP 和一个端口绑定在一个套接字上，然后利用该套接字进行通信。不同的网络应用程序不会打开相同的端口。当系统收到一个数据包时，会根据数据包指示的端口号找到对应的应用程序并转交该数据包。如果对某个端口采用了复用技术，那么系统收到数据包时，就不能够直接将它转交给相应的网络应用程序，而是应该对系统行为作出适当的修改。图 1 展示了端口复用的原理。数据包归属判断模块主要是判断该复用端口收到的数据包是应该转发给网络应用程序，还是转发给端口复用模块。

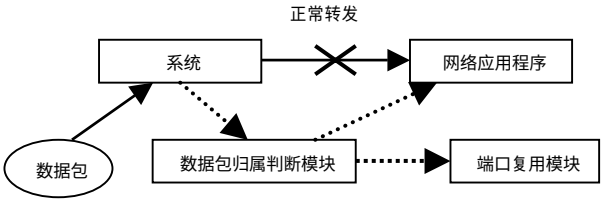

图 1 端口复用原理

3 SPI 的基本原理

服务提供者接口(service provider interface, SPI)是 Windows的WinSock 2 的新特性。服务提供者接口代表着另一

作者简介：罗改龙(1977－)，男，硕士研究生，主研方向：网络与信息安全；程胜利，副教授

收稿日期：2006-08-09 **E-mail：**luogailong@sohu.com

端的Wi n s o c k编程(和Winsock 2API相对应)。Winsock 2 不仅提供了一个供应用程序访问网络服务的Windows socket应用程序编程接口(API)，还包含了由传输服务提供者和名字解析服务提供者实现的Winsock服务提供者接口(SPI)和ws2_32.dll。图 2 展示了应用程序、Ws2_32.dll和传输服务提供者接口之间的层关系[4]。

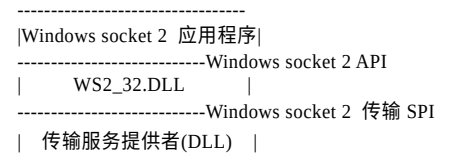

图 2　网络应用程序工作原理

传输服务提供者是以DLL的形式存在的，它对外只有一个入口函数：WSPStartup，其中的参数LPWSPPROC_TABLE结构指针指向函数派遣表，该函数派遣表里包含 30 个SPI函数指针项。每一个SPI函数有与之相对应的API。如WSPRecv和WSPSend，它们在Ws2_32.dll中的对应函数是WSARecv和WSASend，再如WSPRecvFrom和WSPSendTo，在Ws2_32.dll中的对应函数是WSARecvFrom和WSASendTo.WS2_32.DLL就是通过函数派遣表来实现对传输服务提供者的所有SPI函数的调用的。当网络应用程序调用WSASocket/socket函数创建套接字时，会有 3 个参数：地址族，套接字类型和协议，这 3 个参数共同决定了由哪一个类型的传输服务提供者来实现本应用程序的功能[5]。在整个层次结构中，Ws2_32.dll只起到了媒介的作用，应用程序则是对用户功能的实现，而真正实现网络传输功能的是传输服务提供者接口。当前系统中有一些默认的服务提供者，它们已经实现了大部分基本的功能，所以在书写服务提供者程序时，只须对数据报进行处理后，就可将数据报传送给系统服务提供者来实现剩下的功能。

4　基于 SPI 的端口复用技术

对于端口复用技术，核心问题是对数据包的接收处理。本文只研究基于 SPI 的 UDP 端口的复用问题。当网络应用程序接收一个数据包时，如图 2，会通过 API 函数 RecvFrom 或 WSARecvFrom 调用对应的 SPI 函数 WSPRecvFrom 通过对 WSPRecvFrom 的重新设计可以实现端口复用的功能，相关核心代码如下：

```
int WSPAPI WSPRecvFrom (参数略)
{
//先接收网络数据包
nextproctable.lpWSPRecvFrom(s,lpBuffers,dwBufferCount,lpNumberOfBytesRecvd,lpFlags,lpFrom,lpFromlen,lpOverlapped,lpCompletionRoutine,lpThreadId,lpErrno);
if(是给端口复用模块的)
{   端口复用模块进行相应处理
}
Else 转发给网络应用程序 }
```

对端口复用技术，发送数据包的处理相对简单，在端口复用模块中直接调用系统提供的 SPI 函数 WSPSendTo 即可。

5　构建端口复用型木马

在系统中插入一个木马的服务提供者后，木马将可以拦截所有的 WinSock 调用，从而也就拦截了所有的网络应用程序，如图 2。本木马就是利用所拦截的网络应用程序已经打开的端口进行通信的，也就是端口复用。实现木马的服务提供者需要两个应用程序：(1)安装程序，主要负责木马服务提供者的安装和排序；(2)服务提供者(DLL)，该模块主要实现端口复用功能以及数据包的转发工作。为了防止基于数据内容的检测技术，本木马的通信内容进行了加密处理。

5.1　安装程序的技术实现

当网络应用程序用 WSASocket/socket 创建套接字时，Ws2_32.dll 就会在服务提供者数据库中按顺序搜索和 WSAStartup/socket 提供的 3 个参数相匹配的服务提供者，如果同时有两个相同类型的服务提供者存在于服务提供者数据库中，那么顺序在前的那个服务提供者就会被调用。因此，必须将木马的服务提供者信息写入系统的服务提供者数据库，而且要将其排在所有服务提供者的最前面。传输服务提供者包括基础服务提供者和分层式服务提供者两类，本木马采用的是分层式服务提供者。下面是安装程序的具体流程：

(1)用系统已有的 WSAPROTOCOL_INFOW 目录条目结构初始化分层式服务提供者目录条目。用 WSCInstallProvider 安装分层式服务提供者。

(2)遍历系统服务提供者目录数据库获得分层式服务提供者的目录 ID。

(3)用系统现成的 WSAPROTOCOL_INFOW 目录条目结构和获得的目录 ID 初始化协议链目录条目。

(4)安装协议链。

(5)用 WSCWriteProviderOrder 将协议链和分层式服务提供者排在系统的所有服务提供者的前面。

(6)安装程序自行销毁。

当安装完毕后，当网络应用程序访问网络时便会唤醒木马程序，不再需要其它可执行程序的介入，所以可以销毁安装程序，这样系统中只会留下木马的服务提供者(一个动态链接库文件)，加强了木马的隐藏性。

5.2　传输服务提供者的技术实现

木马的传输服务提供者从功能上可以划分为以下 5 个模块：

(1)控制模块。该模块主要负责控制木马的端口状态的开和关。木马必须复用且只能复用一个端口，如果复用多个端口，则会给木马控制端的通信带来混乱，是不可取的。初始时，木马的端口状态设为关。当一个应用程序打开一个 UDP 端口时，便将木马的端口状态改为开，并保存该端口信息，以后就复用该端口进行通信。同时还需要创建一个新线程，该线程主要实现定时连接功能。复用端口所在的网络应用程序随时可能关闭，此时应将端口状态设为关，当找到新的使用 UDP 端口的应用程序时，再将端口状态设为开。只有在端口状态为开的情况下，木马才能与控制端进行通信。

(2)定时连接模块。该模块主要负责控制木马控制木马的工作状态。工作状态分两种：活跃和潜伏。初始状态为潜伏，对于一个木马，绝大多数时候都是潜伏在体统内部，并且平时活动越少越好，这样能增加其隐藏性。本木马采用定时与控制端进行连接的反弹端口形式，设置为每天中午 12：00 向控制端发起连接，若控制端有指示，则将工作状态置为活跃，否则置为潜伏。

(3)加密解密模块。该模块主要是用 3 DES 对数据进行加密和解密处理。

(4)接收模块。该模块主要重写了 WSPRecvFrom，其流程如图 3。其中对判断是否给木马的数据，可根据数据包所指示的 IP 地址进行判断，也可采用发送特殊格式的数据包。

（下转第 169 页）