一种双层入侵检测方法的研究

Investigation of a Kind of double – deck Intrusion Detection Method

刘解放 董方敏 刘 勇 岳宇君 （三峡大学电气信息学院 湖北宜昌 443002）

摘 要： 传统入侵检测系统采用单一在用户层或核心层技术对网络数据包进行捕获和分析均存在缺陷，本文在分析比较了 Windows 系统网络数据包捕获机制的基础上，设计了一个基于中间层驱动技术与 SPI 技术相结合的双层入侵检测系统模型，并对其关键技术进行了实现。实验结果表明，该双层入侵检测模型是比较有效的。并在实际应用中取得了较好的性能。

关键词： 中间层驱动 入侵检测系统 服务提供者接口

随着现代通信技术和互联网的迅速发展与普及，信息技术的应用日益广泛深入，使得网络安全问题更加突出。如何高效地检测和防止网络攻击已是十分重要。传统入侵检测系统[1]一般都是单纯的在用户层或者在核心层对数据包进行监控，这样不可能监控整个多层网络体系，很多非法入侵者就容易被漏检。对于用户层的入侵检测系统，它只能在 Winsock 层之上进行，而对于网络协议栈中底层协议的数据包无法给与处理（如 Ping to Death[2]）。而对于内核层的入侵检测系统，它有一个弱点，就是编程接口复杂，而且编写出来的软件自动化安装太困难，很容易造成整个网络瘫痪。

本文提出一种基于 NDIS 中间层驱动技术[3]与 SPI[4]技术相结合的双层入侵检测模型，利用应用层和核心层双层检测来彻底阻止有害数据攻击。下面将重点研究如何在 Windows 操作系统下开发一个双层入侵检测系统。

1 技术背景

1.1 入侵检测系统分类

1.1.1 基于主机、网络和分布式的 IDS

按照入侵检测的数据来源和系统结构来看，入侵检测系统可以分为基于主机的 IDS（HIDS）、基于网络的 IDS（NIDS）和分布式 IDS（DIDS）。

（1）HIDS 的数据来源于主机系统的审计日志。通过对系统日志和审计记录的不断监控和分析来发现攻击后操作。优点是：①能确定攻击是否成功。②能监视特定的系统活动。③适用被加密和交换的环境。但它只能保护本地单一主机，而且占用大量存储资源和 CPU 资源，实时性差。

（2）NIDS 的数据来源于网络的信息流，该类系统一般被动地在网络上监听整个网段上的信息流，通过捕获网络数据包，进行分析，能够检测该网段上发生的网络入侵。优点是：①实时检测和响应。②攻击者转移证据很困难。③对主机资源消耗少。④操作系统无关性。其最大的缺点是，NIDS 本身也会受到攻击。

（3）DIDS 是同时分析来自主机系统审计日志和网络数据流的入侵检测系统。一般由多个部件组成，分布在网络的各个部分，完成相应的功能。通过中心的控制部件进行数据汇总、分析、产生入侵警报等。在这种结构下，不仅可以检测到针对单独主机的入侵，同时也可以检测到针对整个网络上的主机的入侵。

1.1.2 基本介绍异常检测和误用检测的 IDS

按照入侵检测系统所采用的技术来看，入侵检测系统可以分为误用检测与异常检测两种。

（1）误用检测设定一些入侵检测活动的特征，通过现在的活动是否与这些特征匹配来检测。

（2）异常检测技术则是先定义一组系统“正常”情况的数值，如 CPU 利用率、内存利用率、文件校验和等（这类数据可以人为定义、也可以通过观察系统、并用统计的办法得出），然后将系统运行时的数值与所定义的“正常”情况比较，得出是否有被攻击的迹象。这种检测方式的核心在于如何定义所谓的“正常”情况。对用户要求比较高。

1.2 数据包捕获技术

总的来说，网络数据包捕获技术可以在操作系统的两个状态下进行：用户态捕获技术和内核态捕获技术。

1.2.1 用户态下的数据包捕获

在用户态下进行网络数据包的捕获有三种方法：Windows2000 包过滤接口、替换系统自带的 WINSOCK 动态连接库、Winsock Layered Service Provider（LSP）。其中 WINSOCK2.0 引入的 SPI 技术可以截获所有基于 WINSOCK 的通信，完成诸如传输质量控制（QOS）、扩展 TCP/IP 协议栈，URL 过滤及网络安全控制等功能。用户态下的数据包捕获工作在应用层，针对性强，控制粒度细，但对于网络协议栈中底层协议的数据包无法进行处理。

1.2.2 内核态下的数据包捕获

内核态下进行网络数据包捕获有以下几种方法：

第一，TDI 传输层过滤驱动程序。通过开发过滤驱动来捕获协议驱动所提供的与应用程序交互的接口可实现网络数据包的捕获。TDI 层的网络数据捕获还可以得到操作网络数据包的进程详细信息。

第二，Win2k Filter－Hook Driver。该驱动程序主要是利用 Ipfiltdrv. sys 所提供的功能来捕获网络数据包。由于它的简易性及对 Ipfiltdrv. sys 的依赖性，微软不推荐使用 Filter－Hook Driver。

第三，NDIS Hook Driver。该方法对平台的依赖性比较大，需要在程序中判断不同的操作系统版本而使用不同的方法。

第四，NDIS 中间层驱动程序（IMD）。它可以提供多种服务，能够捕获所有的网络数据包（以太帧），过滤微端口驱动程序，实现特定的协议或其他诸如数据包加密、认证等功能。NDIS 中间层驱动程序的安装虽然在 Windows NT 下比较复杂，但在 Windows 2000 下可以实现自动安装。中间层驱动程序功能强大，是入侵检测技术的趋势所在。

1.3 日志

日志是每个入侵检测软件所不能少的主要的功能，它记录着入侵检测软件监听到发生的一切事件，日志记录需要的内容为源/目的 IP、源/目的的端口、开始/结束时间、进出流量、应用程序、协议、连接方向、管制动作及封包的特殊信息等。

2 入侵检测系统设计

由上述分析可知，传统入侵检测系统单独采用用户层或者核心层技术对数据包捕获均存在缺陷，因此可以用两种模式结合的方法来避免各自的问题，同时发挥各自的长处。根据用户态与内核态的各自方法的比较，本系统采用 NDIS 中间层驱动技术与 WINSOCK SPI 技术相结合的方案实施。采用以下基本策略：NDIS 中间层驱动程序对进出网络的封包进行检查，并根据匹配规则进行第一级检测，主要完成最基本的安全设置，如传输层及以下层协议分析，IP 地址、端口检测等，网络恶劣状况下的断网操作，以及 SPI 无法完成的操作如捕获 ICMP 数据包等。被 NDIS 中间层驱动程序放行的网络数据的检测由 SPI 实现，主要完成针对应用程序和 WEB 网址的第二级检测。本入侵检测系统分为三个模块：

（1）KERIDS. SYS。本模块是位于核心层的驱动程序，根据定义的模式匹配规则进行操作，同时将产生的日志信息发送上层模块。本模块处于操作系统核心，采用 DDK[3,5] 开发。

（2）APPIDS. DLL。本模块是处于应用层的动态链接库，位于 SPI，捕获所有基于 WINSOCK 的网络通讯，根据定义的模式匹配规则进行操作，同时产生日志信息发送上层模块。本模块采用 VC6.0 开发。

（3）IDS. EXE。本模块是一个普通的应用程序，提供用户接口。用户在此设置模式匹配规则；收集并保存前两个模块产生的日志信息；向用户提供日志查询功能。

入侵检测系统的总体结构如图 1 所示：

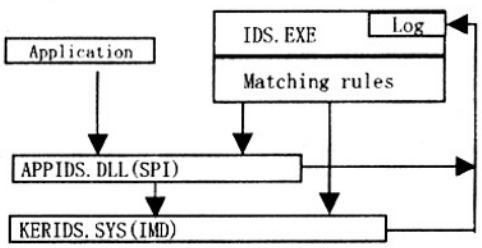

图 1　入侵检测系统模块结构图

网络应用程序的数据都要经过下两层的处理，IDS. EXE 负责模式匹配规则设置，日志的读取，而具体的匹配规则的实施以及安全功能的实现和日志的生成

在 APPIDS. DLL 和 KERIDS. SYS 中。

3 关键技术的实现

在上述的三个模块中，由于 KERIDS. SYS 处于核心层，而且要实现与应用界面的交互，对代码的质量要求较高，因此本文主要针对这两方面的实现作介绍。主要捕获进入主机的 ICMP 数据包为例进行说明。

3.1 捕获 ICMP 数据包

以 Microsoft 驱动开发工具 Win2000 DDK 提供的中间层驱动源代码 Passthru 为框架，对来自外部网络的数据包的捕获、分析及处理，可在 Passthru 的 PtReceive 及 PtReceivePac – ket 函数中实现，流程如图 2 所示。（可以用 DriverMonitor 输出的调试宏信息跟踪确定）。

实现对 ICMP 数据包检测的主要代码结构如下：

（1）定义检查数据包头的函数

```
/ * 根据 IP 包结构，协议标记位为 1 是 ICMP 数据包 * /
UINT Check_Packet( char * pPacket)
```

（2）定义实现将包描符描述的数据包的内容复制到指定的缓存中的函数

```
void CopyPacketBuffer( IN PNDIS_PACKET pPacket, IN OUTPUCHAR pBuff, IN OUTPUINT pLength)
```

（3）插入捕获代码

```
If( monitorflag =1)
{
CopyPacketBuffer ( MyPacket, pPacketContent, &pLength) ;
if( Check_Packet( ( char * ) pPacketContent) )
{
NdisFreeMemory( pPacketContent, 2000, 0) ;//释放复制数据包的缓存
return NDIS_STATUS_NOT_ACCEPTED;
}
}
```

3.2 驱动程序与应用程序的交互

实现驱动程序与应用程序交互的流程如图 3 所示。

（1）应用程序传送数据给设备驱动程序

应用程序通过 CreateFile( ) 函数获取设备驱动程序的句柄后，使用 DeviceIoControl( )、ReadFile( ) 或 WriteFile( ) 等 Win32 函数来实现与设备驱动程序之间的通信，完成控制策略的下传等操作。

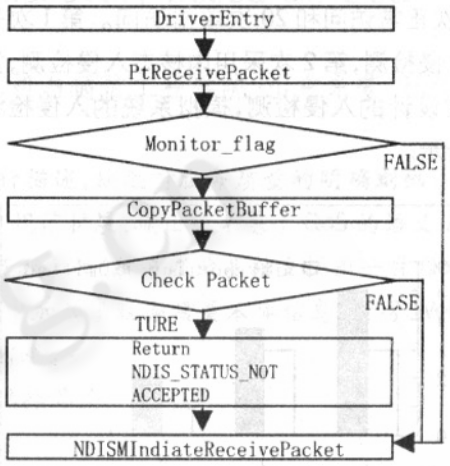

图 2　ICMP 数据包过滤流程图

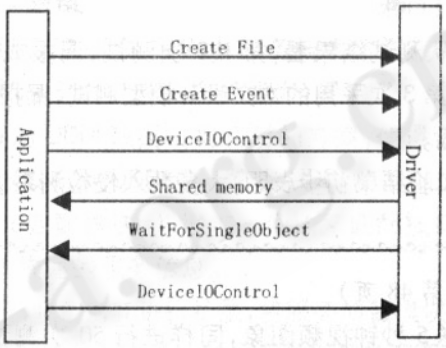

图 3　驱动程序与应用程序通

（2）设备驱动程序给应用程序发送消息

采用 Win32 事件通知方式。应用程序首先创建一个事件，然后将该事件句柄传给设备驱动程序，接着创建一个辅助线程。设备驱动程序获得该事件的句柄后，将它转换成能够使用的事件指针，并存储起来以便后续使用。当设备驱动程序有事件告诉应用程序时就将事件设置为有信号状态，这样应用程序的辅助线程立即知道这个消息并进行相应的处理。

4 实验及结果分析

实验的目的是将单独的用户态入侵检测、单独的

内核态入侵检测和本文的入侵检测进行对比。数据来源于 GIAC(全球信息安全认证,http://www.giac.org),选取了 20 个正常数据集,20 个异常数据集,然后分别对这三种入侵检测系统进行测试。每种测试均进行 20 次正常访问和 20 次攻击访问。第 1 次是采用用户态入侵检测,第 2 次采用内核态入侵检测,第 3 次采用本文设计的入侵检测,模拟系统的入侵检测结果如图 4 所示。

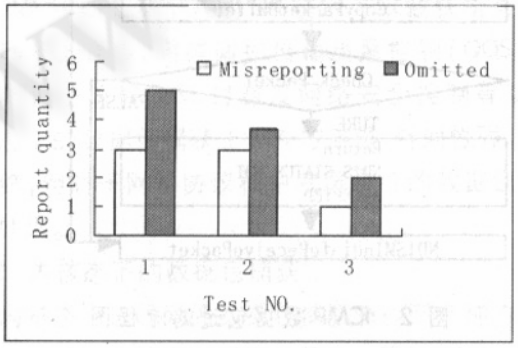

图 4　模拟系统的入侵检测结果

从测试结果看,第 1、2 次测试,漏报或误报数较高;而第 3 次采用的本文设计模型测试,漏报和误报数都较低。

实验结果初步表明,大多数入侵检测系统采用单一的检测策略可能会造成严重的漏报或误报,而采用本文提出的双层检测策略,综合了各层的长处,这样可以降低漏报率和误报率。

5　结束语

本文讨论了在 Windows 2000 下如何利用 NDIS IMD 技术与 SPI 技术实现入侵检测系统,深入讨论了它的原理和实现过程。在实际的实现过程中,NDIS IMD 用于核心层而 SPI 用于应用层,这样就可以相互利用两者的长处,互补两者的短处,使入侵检测系统的设计更高效、合理。

参考文献

1　蒋建春、马恒太、任党恩、卿斯汉,网络安全入侵检测研究综述[J],软件学报,2000,11(11).

2　张振国、张楠,基于 ICMP 的网络攻击与防范[J],微计算机信息,2005,21(7).

3　Microsoft Corporation. Microsoft Windows2000 DDK [DB/OL]. 2001.

4　熊案萍,基于 Winsock 技术的数据包解析研究[J],计算机科学,2006,33(12).

5　微软. Windows 2000 驱动程序开发大全第 1 卷设计指南[M]. 北京:机械工业出版社,2001.