**电子科技大学信息与软件工程学院**

**标 准 实 验 报 告**

**（实验）课程名称 计算机病毒防治**

**电子科技大学教务处制表**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名： 学 号： 指导教师：**

**实验地点： 实验时间：**

**一、实验室名称：**

**二、实验项目名称：PE病毒实验**

**三、实验学时： 4 学时**

**四、实验目的和内容：**

**实验目的**：掌握PE文件格式；理解病毒感染PE文件的原理。

**实验内容：**

1. 了解PE文件格式。
2. 根据实验步骤，编程实现在PE文件中插入病毒代码。运行插入病毒代码后的PE文件。
3. 编程实现在磁盘中搜索.exe文件的操作，对于所有的.exe文件插入病毒代码并运行插入病毒代码后的exe文件。
4. 编程实现在磁盘中搜索(3)中的.exe文件的操作，对于所有的.exe文件删除病毒代码并运行删除病毒代码后的exe文件。
5. **实验原理：**

Win32的可执行文件，如\*.exe、\*.dll、\*.ocx等，都是PE格式文件。感染PE格式文件的win32病毒，简称PE病毒。PE病毒同时也是所有病毒中数量极多、破坏性极大、技巧性最强的一类病毒。为了更好地发展反病毒技术，了解病毒的原理是极为必要的。一个PE病毒基本上需要具有重定位、截获API函数地址、搜索感染目标文件、内存文件映射、实施感染等几个功能，这也是病毒必须解决的几个基本问题。PE病毒感染文件的基本步骤如下：

1. 判断目标文件开始的两个字节是否为“MZ”。
2. 判断PE文件标记“PE”。
3. 判断感染标记，如果已被感染过则跳出继续执行HOST程序，否则继续。
4. 获得Directory(数据目录)的个数，(每个数据目录信息占8个字节)。
5. 得到节表起始位置。(Directory的偏移地址+数据目录占用的字节数=节表起始位置)。
6. 得到目前最后节表的末尾偏移(紧接其后用于写入一个新的病毒节)。

节表起始位置+节的个数\*(每个节表占用的字节数28H)=目前最后节表的末尾偏移。

1. 开始写入节表：
2. 写入节名(8字节)。
3. 写入节的实际字节数(4字节)。
4. 写入新节在内存中的开始偏移地址(4字节)，同时可以计算出病毒入口位置。上节在内存中的开始偏移地址+(上节大小/节对齐+1)×节对齐=本节在内存中的开始偏移地址。
5. 写入本节(即病毒节)在文件中对齐后的大小。
6. 写入本节在文件中的开始位置。上节在文件中的开始位置+上节对齐后的大小=本节(即病毒)在文件中的开始位置。
7. 修改映像文件头中的节表数目。
8. 修改AddressOfEntryPoint(即程序入口点指向病毒入口位置)，同时保存旧的AddressOfEntryPoint，以便返回HOST继续执行。
9. 更新SizeOfImage(内存中整个PE映像尺寸=原SizeOfImage+病毒节经过内存节对齐后的大小)。
10. 写入感染标记(后面例子中是放在PE头中)。
11. 写入病毒代码到新添加的节中：

病毒长度；

病毒代码位置(并不一定等于病毒执行代码开始位置)；

病毒节写入位置(后面例子是在内存映射文件中的相应位置)。

11）将当前文件位置设为文件末尾。

1. **实验器材（设备、元器件）**
2. 学生每人一台PC，安装Windows操作系统。
3. **实验步骤：**

**PE文件病毒的实现**

1. 编制一个输出为“hello world!”的简单程序，运行后得到hello.exe文件。
2. 编制一个简单的病毒代码，例如：弹出消息框。
3. 编制程序在hello.exe文件中插入病毒代码，并运行插入代码后的hello.exe。(结果是弹出消息框后输出“hello world!”。注意：MessageBox()的偏移地址在user32.dll中，需在kenrel32.dll中取得LoadLibraryA（）和GetProcAddress()的加载地址，调用LoadLibraryA()加载user32.dll,再调用GetProcAddress()获取MessageBox()加载地址。)

4. 编制程序搜索D盘中的所有的\*\*\*.exe文件，并对所有\*\*\*.c文件进行步骤3的操作。运行\*\*\*.exe文件。

5. 编制程序实现删除\*\*\*.exe文件中病毒代码操作，运行删除病毒代码后的\*\*\*.exe文件。

1. **实验数据及结果分析：**

**(按实验步骤顺序填写代码、数据或截图)**

1. **实验结论、心得体会和改进建议：**

**六、对本实验过程及方法、手段的改进建议：**

**报告评分：**

**指导教师签字：**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名： 学 号： 指导教师：**

**实验地点： 实验时间：**

**一、实验室名称：**

**二、实验项目名称：剖析木马程序实验**

**三、实验学时： 8学时**

**四、实验原理：**

**实验目的：**通过对木马的练习，理解与掌握木马传播与运行的机制；通过手动删除木马，掌握检查木马和删除木马的技巧，学会防御木马的相关知识，加深对木马的安全防范意识。

**实验内容**：

1. Socket编程实验
2. Mini木马程序剖析实验
3. 经典木马程序实验
4. **实验原理：**

木马的全称为特洛伊木马，来源于古希腊神话。木马是具有远程控制、信息偷取、隐私传输功能的恶意代码，它通过欺骗或者诱骗的方式安装，并在用户的计算机中隐藏以实现控制用户计算机的目的。

1. **木马的特性**

木马程序为了实现其特殊功能，一般应该具有以下性质。

伪装性：木马程序伪装成合法程序，并且诱惑被攻击者执行，使木马代码会在未经授权的情况下装载到系统中并开始运行。

隐藏性：木马程序不被杀毒软件杀掉，会在系统中采用一些隐藏手段，不会让使用者觉察到木马的存在，例如进程在任务管理器中隐藏，文件不会出现在浏览器中等，从而实现长久对目标计算机的控制。

破坏性：通过木马程序的远程控制，攻击者可以对目标计算机中的文件进行删除，修改、远程运行，还可以进行诸如改变系统配置等恶性破坏系统。

窃密性：木马程序最大的特点就是可以偷取被入侵计算机上的所有资料，包括硬盘上的文件，以及屏幕信息，键盘输入信息等。

1. **木马的入侵途径**

木马的入侵主要是通过一些欺骗手段实施的。

捆绑欺骗：如果把木马文件与普通文件捆绑，并更改捆绑后的文件图标，伪造成与原文件类似。再通过电子邮件、QQ、MSN等手段直接发送给用户，或者通过放在网上或者某个服务器中，欺骗被攻击者下载直接执行。

利用网页脚本入侵：木马也可以通过Script, active, ASP, CGI交互脚本的方式入侵，由于微软的浏览器在执行Script脚本上存在一些漏洞，攻击者可以利用这些漏洞实现木马的下载和安装。

利用漏洞入侵：木马还利用一些系统的漏洞入侵，如微软的IIS服务器存在多种溢出漏洞，通过缓冲区溢出攻击程序造成IIS服务器溢出，获得控制权限，然后在被攻击的服务器上安装并运行木马。

和病毒协作入侵：现在的病毒有多种自动感染和传播功能，而木马往往和病毒协同工作，在病毒感染目标计算机后，通过木马对目标计算机进行控制。

1. **木马的种类**

按照发展历程和主流技术的演变，可以将木马分为5个阶段。

第一代木马是出现在20世纪80年代，主要是UNIX环境中通过命令行界面实现远程控制。

第二代木马出现在20世纪90年代，随着WINDOWS系统的普及木马在WINDOWS环境中大量应用，它具备伪装和传播两种功能，具有图形控制界面，可以进行密码窃取、远程控制，例如BO2000和冰河木马。因为放火墙的普遍应用，第二代木马在进入21世纪之后不再有多少用武之地了，由于它采用黑客主动连接用户的方式，对于这种从外网发来的数据包都将被防火墙阻断。

第三代木马在连接方式上比第二代木马有了改进，通过端口反弹技术，可以穿透硬件防火墙，例如灰鸽子木马，但木马进程外联黑客时会被软件防火墙阻挡，经验丰富的网络管理员都可以将其拦截。

第四代木马在进程隐藏方面比第三代木马做了教大改动，木马通过线程插入技术隐藏在系统进程或应用进程中，实现木马运行中没有进程，网络连接也隐藏在系统进程或应用进程中，比如广外男生木马。第四代木马可以实现对硬件防火墙的穿透，同时它隐藏在系统或应用进程中，往往网络管理员很难识别，所以被软件防火墙拦截后往往又被放行，从而实现对软件防火墙的穿透。

第五代木马在隐藏方面比第四代木马又进行了进一步提升，它普遍采用了ROOTKIT技术，通过ROOTKIT技术实现木马运行时进程、文件、服务、端口等的隐藏，采用系统标准诊断工具难以发现它的踪迹。

除了按照技术发展分类之外，从功能上木马又可以分为：破坏型木马，主要功能是破坏并删除文件；密码发送型木马，它可以找到密码并发送到指定的邮箱中；服务型木马，它通过启动FTP服务或者建立共享目录，使黑客可以连接并下载文件；DOS攻击型木马，它将作为被黑客控制的肉鸡实施DOS攻击；代理型木马，可使被入侵的机器作为黑客发起攻击的跳板；远程攻击型木马，可以使攻击者利用客户端软件进行完全控制。

1. **木马的连接方式**

一般的木马都采用C/S运行模式，它分为两部分，即客户端和服务器端木马程序。黑客安装木马的客户端，同时诱骗用户安装木马的服务器端。下面简单介绍木马的传统连接技术、反弹端口技术和线称技术。

1. 木马的传统连接技术

第一代木马和第二代木马均采用传统的连接方式，即由木马的客户端程序主动连接服务器端程序。当服务器端程序在目标计算机上被执行后，一般会打开一个默认的端口进行监听，当客户端向服务器主动提出连接请求，服务器端的木马程序就会自动运行，来应答客户端的请求，从而建立连接。

1. 木马的反端口技术

随着防火墙技术的发展，它可以有效拦截采用传统连接方式从外部主动发起连接的木马程序。但通常硬件防火墙对内部发起的连接请求则认为是正常连接，第三代之后的“反弹式”木马就是利用这个缺点，其服务器端程序主动发起对外连接请求，连接到木马的客户端，就是说“反弹式”木马是服务器端主动发起连接请求 ，而客户端被动的等待连接。

根据客户端的IP地址是静态的还是动态的，反弹端口可以有两种连接方式。

反弹端口的第一种连接方式，在设置服务器断时要设置固定的客户端IP地址和待连接端口，所以这种方式只适用于客户端IP地址是公网IP且是静态IP的情况。

反弹端口的第二种连接方式，可实现服务端根据配置主动连接变动了IP的客户端。入侵者为了避免暴露自己的身份，往往通过跳板计算机控制被入侵用户的计算机，在跳板计算机中安装木马客户端软件，被入侵用户的计算机安装木马的服务端软件。当然，入侵者的跳板计算机有时可能失去入侵者的控制，这时入侵者就需要找到新的跳板计算机，同时使用户计算机上的木马服务端程序，能够连接到新跳板计算机上的木马客户端程序。为此，入侵者利用了一个“代理服务器”保存改变后的客户端IP地址和待连接的端口，只要跳板主机改变了IP地址，入侵者就可以更新“代理服务器”中存放的IP地址和端口号。远程被入侵主机上的服务端程序每次启动后，被设置为先连接“代理服务器”，查询最新木马客户端的IP和端口信息，再按照新的IP地址和客户端进行连接，因此这种连接方式可以适用于客户端和服务器端都是动态IP地址的情况，而且还可以穿透更加严密的防火墙，当然客户端的IP要求是公网IP才行。

1. **木马的隐藏技术**

木马为了防止被杀毒软件查杀，同时也避免被用户计算机发现，往往采用一些隐藏技术，在系统中实现隐身。

1. 线程插入技术

一般一个应用程序在运行之后，都会在系统中产生一个进程，同时，每个进程分别对应了一个不同的PID（progress ID，进程标示符）。系统会分配一个虚拟的内存空间地址段给这个进程，一切相关的程序操作，都会在这个虚拟的空间中进行。一个进程可以对应一个或多个线程，线程之间可以同步执行，一般情况下，线程之间是相互独立的，当一个线程发生错误的时候，并不一定会导致整个线程的崩溃，“线程插入技术”就是运用了线程之间运行的相对独立性，使木马完全溶进了系统的内核。这种技术把木马程序作为一个线程，把自身插入到其他应用程序的地址空间。而这个被插入的应用程序对于系统来说，是一个正常的程序，这样，就达到了彻底的隐藏效果。系统运行时会有许多的进程，而每个进程又有许多的线程，这就导致了查杀利用“线程插入”技术木马程序难度的增加。

1. ROOTKIT技术

ROOTKIT是一种隐藏技术，它使得恶意程序可以逃避操作系统标准诊断程序的查找。早期的ROOTKIT技术通过修改内存中的系统文件映像来逃避检测。这一类ROOTKIT技术主要依赖HOOK技术，比如HOOK API或者系统调用表。目前主流的ROOTKIT技术主要在内核态实现，例如DKOM（直接内核对象操作）技术，通过动态修改系统中的内核数据结构来逃避安全软件的监测。由于这些数据结构随着系统的运行而不断更新变化，因此非常难于检测。

1. **木马的检测**

木马的远程控制功能要实现，必须通过执行一段代码来实现。为此，木马采用的技术再新，也会在操作系统中留下痕迹。如果能够对系统中的文件、注册表做全面的监控，可以实现发现和清楚各种木马的目的。当然现有的监控手段还不一定能够做到全面的监控，但对系统的行为监控也已经非常深入了，通过运用多种监控手段和工具，可以协助发现植入的木马。当然，由于木马的机制不同，所以检测和查杀手段也不尽相同。

1. **实验器材（设备、元器件）**

1、实验人数50～80人，每人1台计算机；两人一组完成本实验。

2、拓扑：（A、B范围中的主机分别简称为A主机和B主机）



**A**

**B**

3、设备：以太网交换机2～4台；计算机50～80台

4、软件：VC++ 6.0软件

1. **实验步骤：**
2. Socket编程实验
3. 使用VC，新建一个“Win32 Console Application”类型的工程。
4. 在这个项目中编写一个基于TCP Socket的服务端“C++ Source File”， 其流程是WSAStartup( )—socket( )—bind( )—accept( )—send( ), 完成的功能是服务端监听主机A的某个端口，一旦有客户端telnet这个端口，就向客户端发送欢迎语句如“hello”等。
5. 在主机A上执行这个程序，使用“netstat –an”命令查看程序中指定的端口处于什么状态。
6. 选择主机B作为客户端，使用“telnet IP port”命令连接主机A，记录运行结果。
7. Mini木马程序剖析实验
8. 在主机A上编译组建执行mini木马程序。

Mini程序体现了木马的基本功能远程控制，无须客户端软件，服务端精简，占用非常少的CPU和内存资源，便于隐藏。但不能自启动，需要第三方软件加载到自启动项目或服务中。

1. 主机A上使用“netstat –an”命令查看端口999处于什么状态。
2. 选择主机B作为客户端，使用“telnet IP 999”命令连接主机A，记录运行结果。
3. 主机A上使用“netstat –an”命令查看主机B的哪个端口和主机A的999端口建立了连接，状态是什么。同时在主机A上
4. 主机B的Telnet窗口中，使用ipconfig 和“net user”命令查看系统的IP地址和用户。
5. 使用“net user 用户名 密码 /add”命令增加一个用户。
6. 使用“net localgroup Administrators 用户名 /add”命令将该用户添加到Administrators组。
7. 使用“net localgroup Administrators”命令查看该组下有哪些用户。
8. 使用exit命令退出telnet连接。
9. 经典木马程序实验
10. 关机程序shutdown，阅读程序代码，执行程序自动关闭系统。
11. 进程查杀程序process
    1. 阅读程序代码。
    2. 打开一个cmd窗口，使用tasklist命令查看系统进程和PID号。记下当前cmd窗口的PID号。
    3. 执行process程序,结果与tasklist命令比较，并在提示语句后输入cmd窗口的PID号，结果会怎样。
12. 获取主机IP地址程序hostip，阅读程序代码，执行程序列出当前主机地址。
13. 单线程TCP扫描程序tcpscanner，阅读程序代码，执行程序列出当前主机端口。
14. 下载者程序download，阅读程序代码，修改程序为ftp协议下载，执行程序，查看是否在主机上下载成功。
15. 执行注册表读取程序read，分析其取得是注册表哪个位置的值？取得的值是否跟注册表里的信息一致？
    1. 注册表是以树状结构储存，每一个节点称为一个键值（key），每个key又包括子键值(subkey)及数据入口的值(value)。读写注册表前，必须先将目标的子键打开，也就是取得一个操作的句柄。
    2. 打开函数

LONG RegOpenKeyEx(

　　HKEY hKey, // 需要打开的主键的句柄

　　LPCTSTR lpSubKey, //需要打开的子键的名称

　　DWORD ulOptions, // 保留，设为0

　　REGSAM samDesired, // 安全访问标记，也就是权限

PHKEY phkResult // 得到的将要打开键的句柄

）；

1. **实验数据及结果分析：**

**(按实验步骤顺序填写代码、数据或截图)**

1. **实验结论、心得体会和改进建议：**

**报告评分：**

**指导教师签字：**