**网络安全课程项目报告**



**项目名称: 基于木马的远程访问系统**

**指导教师: 廖鑫**

**小组成员: 柴博宇、黄卫江、杨焱鑫、李骏嘉**

目录

[引言 3](#_Toc9259286)

[项目介绍 3](#_Toc9259287)

[分工情况 4](#_Toc9259288)

[项目说明 4](#_Toc9259289)

[木马系统所能实现的功能 4](#_Toc9259290)

[木马的植入、开机自起、隐藏 4](#_Toc9259291)

[DLL木马的行为 5](#_Toc9259292)

[控制服务器部分 6](#_Toc9259293)

[主要控制类说明 6](#_Toc9259294)

[控制服务器UI界面 9](#_Toc9259295)

[总结 11](#_Toc9259296)

# 引言

病毒这个话题可谓是老生常谈，在没有计算机的时候指的是生物病毒，被病毒感染的生物往往会生病甚至死亡。计算机出现后就引用了生物界的概念，有了计算机病毒的概念。互联网高速发展的今天，很多心术不正的人利用高超的计算机技术制作计算机病毒，通过传播病毒来达到个人利益。

渐渐的就衍生出计算机安全相关的职业职业和专业，欲防守就得先学会攻击，我们就是通过制作这样一个病毒来加强计算机安全方面的知识和认识。

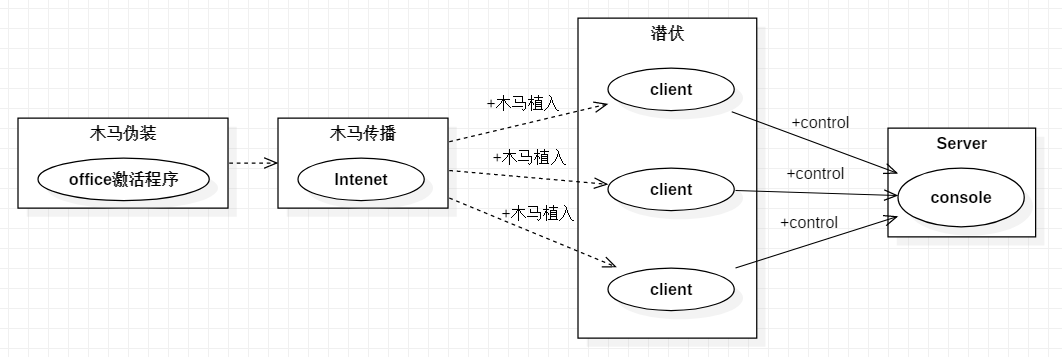
# 项目介绍

本项目是基于木马的远程访问系统。具体的，利用DLL劫持、反弹端口等技术，实现了在目标主机上注入木马，并能通过控制服务器操作木马以进行在目标机执行SHELL指令、从目标机上拉取文件到服务器、从服务器发送文件到目标机等操作。

此外，我们实现了木马开机自起、断线重连，并且通过将Window系统服务作为劫持对象，木马程序实现了很高级别的隐藏。

控制服务器使用了UI图形界面，能够形象的看到已经与控制服务器建立连接的所有被控主机，并能选中任意在线的主机进行操控，可以手动操作服务器进行刷新以清除断开连接的主机。

项目的架构如下：



木马程序被伪装为office激活程序，通过投入互联网的形式进行传播，在木马被运行时会进行植入操作，之后再木马启动时，会主动向服务器创建一个连接，服务器在公网具有固定IP，且能够维护多个被控机的连接，服务器可以向任意一个被控制发送控制指令。

# 分工情况

|  |  |
| --- | --- |
| **成员姓名** | **所做工作** |
| 柴博宇 | 1，木马注入；2，开机自启实现；3，DLL木马部分绝大多数逻辑；4、控制服务器主要类的编写；4，项目整理；5，视频录制；6，部分ppt制作 |
| 黄卫江 | 1，服务器UI界面，及部分socket通信逻辑；2，开机自启（未采用）；3，socket大文件传输；4，编写说明文档；5视频录制；6，部分ppt制作 |
| 杨焱鑫 | 1，控制端和被控端部分功能代码实现（部分功能由于木马端的问题未实际采用）；2，视频录制；3，答辩视频演示；4，系统功能测试 |
| 李骏嘉 | 1，控制端和被控端部分功能代码实现（部分功能由于木马端的问题未实际采用）；2，视频录制；3，功能测试；4、为项目提出了很多建设性的意见 |

# 项目说明

## 木马系统所能实现的功能

系统功能主要是针对控制端服务器可以对被控制端执行的操作来说的，主要分为两个部分，我们通过控制端主机可以在被控制端主机上执行SHELL指令和自定义功能：

1. SHELL指令：

由于我们的劫持对象是Windows系统服务，而Windows系统服务从Vista版本开始就禁止与用户界面交互，我们也无法突破该Session隔离，因此，这里的SHELL指令是指windows中cmd所能执行的除与用户图形界面交互外的所有指令，借助于我们劫持的对象是系统服务，我们的木马也拥有与系统服务相同的执行特权指令的权限。

1. 自定义指令：

由于windows下shell指令的局限性，所以我们通过预置一些指令来实现更强大的功能，这里主要的文件收发功能。控制端可以获取被控端储存设备上任意文件，也能向被控端的任一文件夹发送任意文件。

有了这两部分功能，结合使用就能实现很多功能了，比如有些繁杂的功能通过shell执行会比较麻烦，所以可以先向被控端发送一个exe文件，在通过shell指令在被控端上执行这个exe，就能实现我们预想的功能了。

值得一提的是我们的文件传输功能有很好的稳健性，经过我们的测试，实际传输过程中不会出现格式错乱等问题且对于大型的文件也不会出现崩溃的情况。

## 木马的植入、开机自起、隐藏

起初我们使用了DLL注入的方式实现木马程序的影藏，具体表现为程序通过修改自身身份令牌获取debug权限，之后通过远程创建线程的函数随机选取当前的活动进程并注入线程以调用我们的木马程序。但在实际测试中，我们发现这类程序由于技术出现很早，已能被绝大多数杀毒软件发现，在Windows8扥更高版本的系统中几乎没有成功运行的可能。因此我们封存了该方案。

之后我们通过查阅资料以及相关安全技术博客，最终参考了HOOK007系列木马的行为，利用Windows系统服务的漏洞，劫持名为Print Spooler的系统服务所加载的DLL，使其在运行时启动了我们的木马程序。由于该服务为所有Windows10系统自带，因此，我们能够以最少的改动达到注入木马的目的，此外由于劫持了系统服务，一方面，木马程序能够拥有与系统服务同等的系统权限，另一方面，木马注入前后目标机的任务管理器中不会有丝毫不同，达到了隐藏的目的。

木马植入的整个过程如下：

1. 判断系统的位数

对系统的位数进行判断，由于不同位数的系统对应的Print Spooler系统服务的位数也不同，因此为保证服务能加载正确的DLL需要先判断系统的位数以决定装载的DLL的位数。

1. 创建文件夹并拷贝DLL

在系统服务的数据文件夹下创建单独文件夹以存放用于进行劫持的DLL，之后从我们的DLL分发服务器上下载对应位数的DLL。

1. 修改注册表(重定位目标的dll路径)

Print Spooler服务是通过注册表来定位要加载的DLL的位置的，因此我们需要修改注册表使得该服务在我们指定的文件夹下搜索DLL以加载我们的DLL木马。

1. 将Print Spooler系统服务设为自起

虽然在Windows系统下，Print Spooler服务是默认开机自起的，但为了以防万一，我们仍然需要定位该服务并将该服务器设置为自起的状态。

## DLL木马的行为

DLL木马的行为相对比较简单，大致就是创建一个SOKCET套接字，从被控主机建立一条到我们的控制服务器的连接，之后等待控制服务器发送的控制指令，在收到控制指令后，木马根据指令的类型调取相应的响应处理函数以执行服务器所要求的功能并向服务器返回处理结果。

唯一需要注意的是可能会由于网络断开连接等问题造成木马与控制服务器断开连接，在木马部分我们通过捕获异常来进行处理，当连接断开时，每隔30秒钟，就会尝试进行一次重连。

具体为：

catch (...) //不管什么类型的异常, 重连就完事了

{

closesocket(clientSocket);

clientSocket = socket(AF\_INET, SOCK\_STREAM, 0);

continue;

}

DLL木马的部分功能函数列表如下：

/\*\*\*\*\* 功能函数表 \*\*\*\*\*/

void MakePath(string address, bool hidden); //创建文件夹目录

void SendBrief(SOCKET socket, string message); //使用socket发送信息

string ReadBrief(SOCKET socket); //使用套接字读取数据

void FetchFileHandler(SOCKET socket); //处理获取指定文件请求

void SendFileHandler(SOCKET socket);// 处理发送指定文件请求

void ShellHandler(SOCKET socket); //处理执行shell指令请求

void ScreenShotHandler(SOCKET socket); //处理截图指令请求

## 控制服务器部分

### 主要控制类说明

**SocketNode类**

由于Java的原生Socket类不能很好的满足我们对于控制服务器的需求，因此我们自己编写了一个简单的SocketNode类，同时封装了链接状态检测函数以便我们的其他控制类进行处理。



**DllLoadServer类**

为了保证控制的隐蔽性（不在被控端滞留过多文件），以及针对不同位数的系统植入不同位数的DLL木马，我们把要注入的DLL放在了服务器中，“office激活程序”只负责修改注册表保证自启动以及从服务器下载DLL实现注入。测试中，我们使用的是6666端口，和ControlServer的5547端口不同。



**ControlServer类**

ControlServer主要用来接收来自木马服务器的连接，每当有新的连接到来是，ControlServer将接受该链接并将该链接放入连接列表，以供操作单元从列表中选取并进行控制。此外ControlServer中还提供了刷新函数以从连接列表中删除失效的链接。



**ControlUnit类**

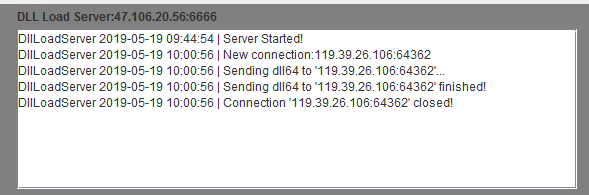
ControlUnit是服务器的基本控制单元，主要用于针对选中的连接进行操控，类中包含了与被控连接进行收发文件、向被控连接发送指令等函数，同时提供了ExecuteCommand函数，该函数解析输入的字符串指令并根据解析的结果向连接发送相应的指令。



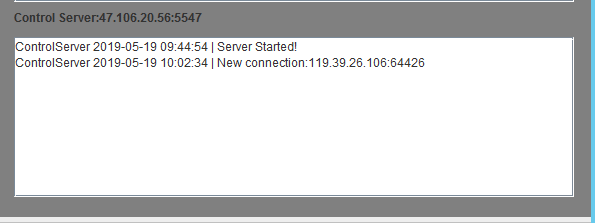
### 控制服务器UI界面

UI界面主要分为DllLoadServer、ControlServer以及ControlUnit三个模块。

DllLoadServer模块用于显示Dll分发服务器的输出信息，能够很直观的看到服务器在什么时候向哪台主机发送了什么版本的dll木马。



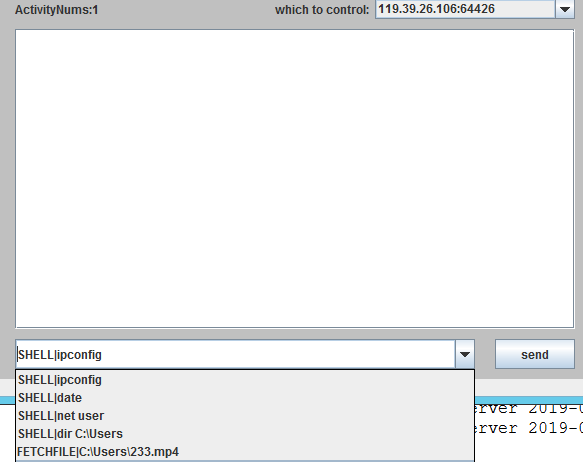
ControlServer模块用于显示控制服务器的输出信息，在该界面能够看到服务器在什么时候接收到了哪台被控机的连接。



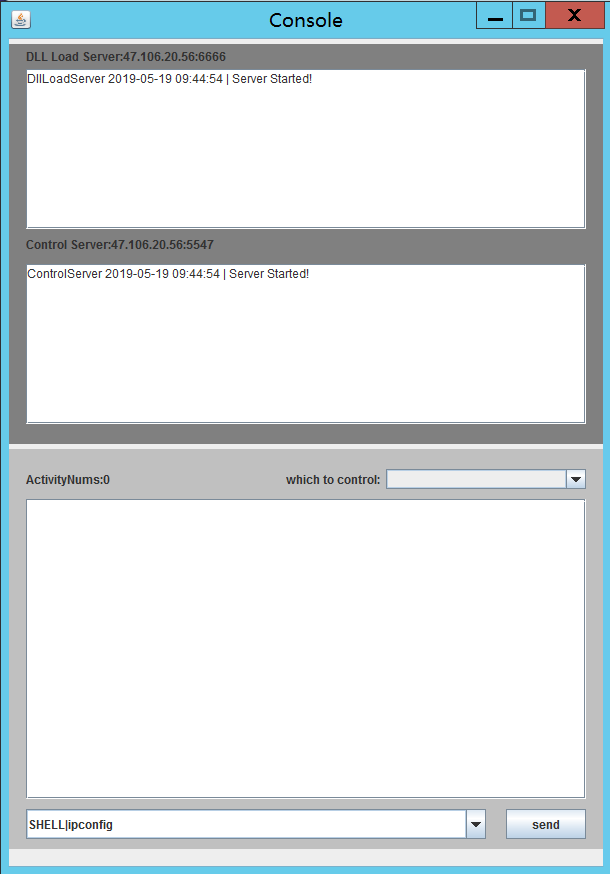
ControlUnit界面主要负责显示当与被控主机进行交互的情况，在右上角的下拉列表中可以根据IP地址和端口号选取任意的连接进行操控。

在正下方的输入栏可以手动输入指令进行执行或者通过下拉的方式选取预定义的模板指令。

中间部分则用于显示当前的交互信息，包括指令的执行情况，被控机的返回信息等。



服务器整体UI界面如下所示：



# 总结

本项目工作量较大，组内每个人都积极参与了项目，虽然在项目的实现过程中由于自身的技术较低以及对相关方面的不了解，走了很多弯路，写了很多没必要的代码，但最终的成果还是让我们感到很满意的，通过本项目，我们组内的每个成员都在windows系统的安全机制、socket通信等方面积累了不少经验，总之，这是一个很有意义的项目。