【域渗透】获取域内机器共享

rcoil / 2019-10-23 09:25:07 / 浏览数 5092 渗透测试 渗透测试 顶(0) 踩(0)

在域内遇到瓶颈时，不妨看看域内机器开放的共享，获取有些收获。

0x00 前言

大多数内网渗透总结在提到使用 WinAPI■■■■ 只是讲述了利用 NetSessionEnum 来找寻■■ sessions， 利用 NetWkstaUserEnum 来枚举登陆的用户，利用 NetShareEnum 来找寻共享，但却未说明其原型理论。由于前篇文章 【域渗透】域内会话收集) 已经针对 NetSessionEnum 和 NetWkstaUserEnum 进行说明，所以本文是对 NetShareEnum 做一个概述及应用的补充。

0x01 NetShareEnum 函数

以当前权限检索有关服务器上每个共享资源的信息。还可以使用 WNetEnumResource 函数来检索资源信息。但是，WNetEnumResource不会枚举隐藏的共享或连接到共享的用户。

该函数原型为：

C++ Copy

```
NET_API_STATUS NET_API_FUNCTION NetShareEnum(
  LMSTR   servername,
  DWORD   level,
  LPBYTE  *bufptr,
  DWORD   prefmaxlen,
  LPDWORD entriesread,
  LPDWORD totalentries,
  LPDWORD resume_handle
);
```

它需要 7 个参数。

servername：指向一个字符串的指针，该字符串指定要在其上执行该功能的远程服务器的DNS或NetBIOS名称。如果此参数为NULL，则使用本地计算机

level：指定数据的信息级别。

值	含义
0	返回共享名称。所述bufptr参数指向的数组 SHARE_INFO_0结构。
1	返回有关共享资源的信息，包括资源的名称和类型以及与资源关联的注释。所述bufptr参数指向的数组 SHARE_INFO_1结构。
2	返回有关共享资源的信息，包括资源名称，类型和权限，密码以及连接数。所述bufptr参数指向的数组 SHARE_INFO_2结构。
502	返回有关共享资源的信息，包括资源名称，类型和权限，连接数以及其他相关信息。所述bufptr参数指向的数组 SHARE_INFO_502结构。不返回来自不同范围的共享。有关范围界定的更多信息，请参见NetServerTransportAddEx函数的文档的"备注"部分。
503	返回有关共享资源的信息，包括资源名称，类型和权限，连接数以及其他相关信息。所述bufptr参数指向的数组SHARE_INFO_503结构。返回所有范围的共享。如果此结*"，则没有配置的服务器名称，并且NetShareEnum函数枚举所有未作用域名称的共享。Windows Server 2003和Windows XP： 不支持此信息级别。

bufptr：向接收数据的缓冲区的指针。该数据的格式取决于 level 参数的值。

prefmaxlen：指定返回数据的首选最大长度，以字节为单位。如果指定MAX_PREFERRED_LENGTH，则该函数分配数据所需的内存量。如果在此参数中指定另一个值，

- entriesread：指向一个值的指针，该值接收实际枚举的元素数。

  totalentries：指向一个值的值，该值接收可能已经枚举的条目总数。

  resume_handle：指向包含恢复句柄的值的指针，该恢复句柄用于继续现有的共享搜索。

而此 API 的调用示例为：

```
string server = "rcoil.me";
int ret = NetShareEnum(server, 1, ref bufPtr, MAX_PREFERRED_LENGTH, ref entriesread, ref totalentries, ref resume_handle);
```

它会返回如下内容：

```
shi1_netname - ADMIN$
shi1_remark - Remote management
```

```
shi1_netname - C$
shi1_remark - Default share
....
```

关键源码如下：

```
/// <summary>
/// ■■■■■■■■■■■■■■■■■■ SHARE_INFO_1[] ■■■■■
/// https://www.pinvoke.net/default.aspx/netapi32/netshareenum.html
/// </summary>
/// <param name="Server"></param>
/// <returns></returns>
public static SHARE_INFO_1[] EnumNetShares(string Server)
{
   List<SHARE_INFO_1> ShareInfos = new List<SHARE_INFO_1>();
   int entriesread = 0;
   int totalentries = 0;
   int resume_handle = 0;
   int nStructSize = Marshal.SizeOf(typeof(SHARE_INFO_1));
   IntPtr bufPtr = IntPtr.Zero;
   StringBuilder server = new StringBuilder(Server);
   int ret = NetShareEnum(server, 1, ref bufPtr, MAX_PREFERRED_LENGTH, ref entriesread, ref totalentries, ref resume_handle);
   if (ret == NERR_Success)
   {
       IntPtr currentPtr = bufPtr;
       for (int i = 0; i < entriesread; i++)
       {
           SHARE_INFO_1 shi1 = (SHARE_INFO_1)Marshal.PtrToStructure(currentPtr, typeof(SHARE_INFO_1));
           ShareInfos.Add(shi1);
           currentPtr += nStructSize;
       }
       NetApiBufferFree(bufPtr);
       return ShareInfos.ToArray();
   }
   else
   {
       ShareInfos.Add(new SHARE_INFO_1("ERROR=" + ret.ToString(), 10, string.Empty));
       return ShareInfos.ToArray();
   }
}
```

演示结果：

```
beacon> execute-assembly /Users/rcoil/Desktop/Github/CSharp_Tools/SharpShares/Share_NetWorkConnectIPC/bin/Debug/Share_NetWorkConnectIPC.exe 127.0.0.1
[*] Tasked beacon to run .NET program: Share_NetWorkConnectIPC.exe 127.0.0.1
[+] host called home, sent: 109117 bytes
[+] received output:
[+] 127.0.0.1 机器开启的共享如下:
  [>] ADMIN$
  [>] C$
  [>] IPC$
  [>] Users
[*] Processing completed

beacon> execute-assembly /Users/rcoil/Desktop/Github/CSharp_Tools/SharpShares/Share_NetWorkConnectIPC/bin/Debug/Share_NetWorkConnectIPC.exe 192.10.20.233
[*] Tasked beacon to run .NET program: Share_NetWorkConnectIPC.exe 192.10.20.233
[+] host called home, sent: 109125 bytes
[+] received output:
[+] 192.10.20.233 机器开启的共享如下:
  [>] ADMIN$
  [>] C$
  [>] IPC$
  [>] NETLOGON
  [>] SYSVOL
[*] Processing completed

beacon> execute-assembly /Users/rcoil/Desktop/Github/CSharp_Tools/SharpShares/Share_NetWorkConnectIPC/bin/Debug/Share_NetWorkConnectIPC.exe 192.10.20.232
[*] Tasked beacon to run .NET program: Share_NetWorkConnectIPC.exe 192.10.20.232
[+] host called home, sent: 109125 bytes
[+] received output:
[+] 192.10.20.232 机器开启的共享如下:
  [>] ERROR=5
[*] Processing completed
```

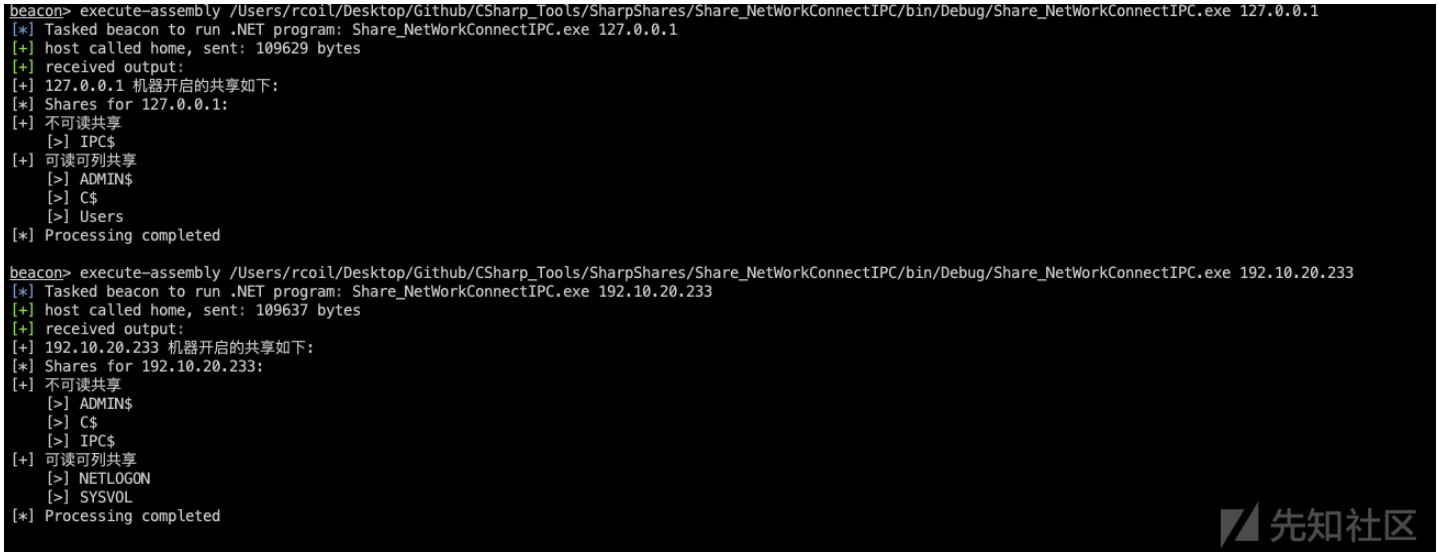

0X02 进一步拓展

2.1、判断是否可读

判断可读，是根据当前用户权限进行判断的。

```
string path = String.Format("\\\\{0}\\{1}", computer, share.shi1_netname);
var files = System.IO.Directory.GetFiles(path);
```

直接根据以上方法进行访问测试即可。效果如下



接下来就是以当前权限，对可访问的共享进行遍历即可。获取文件名、文件大小，再进行下一步的筛选。

2.2、遍历

```
/// <summary>
/// ■■■■■■■
/// </summary>
/// <param name="info">■■■■■■</param>
public static void ListFiles(FileSystemInfo info)
{
 if (!info.Exists) return;
 DirectoryInfo dir = info as DirectoryInfo;
 //■■■■
 if (dir == null) return;
 try
 {
   FileSystemInfo[] files = dir.GetFileSystemInfos();
   for (int i = 0; i < files.Length; i++)
   {
     FileInfo file = files[i] as FileInfo;
     //■■■
     if (file != null)
       Console.WriteLine(file.FullName);
     //■■■■■■■■■■■■
     else
       ListFiles(files[i]);
   }
 }
 catch { }
}
```

```
        Console.WriteLine();
        Console.WriteLine("[*] {0} 机器开启的共享如下:", computer);
        try
        {
            List<string> readableShares = AllSharp.GetShares(computer);
            readableShares.Remove(readableShares.First(obj => { return obj.IndexOf("ADMIN$") != -1; }));
            foreach (string nameShares in readableShares)
            {
                string pathShares = String.Format("\\\\{0}\\{1}", computer, nameShares);
                ListFiles(new DirectoryInfo(pathShares));
            }
```

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

[*] 127.0.0.1 机器开启的共享如下:
[+] 不可读共享
    [>] IPC$
[+] 可读可列共享
    [>] ADMIN$    远程管理
    [>] C$    默认共享
    [>] D$    默认共享
    [>] E$    默认共享
\\127.0.0.1\C$\$Recycle.Bin\S-1-5-21-3807771969-3534507594-3355041475-1001\$I019DJL
\\127.0.0.1\C$\$Recycle.Bin\S-1-5-21-3807771969-3534507594-3355041475-1001\$I080UD1.exe
\\127.0.0.1\C$\$Recycle.Bin\S-1-5-21-3807771969-3534507594-3355041475-1001\$I0KT1S0.jpg
```


到此，整个过程就可以结束了。

上一篇：记一次webshell的获取 下一篇：[红日安全]Web安全Day9 -...

1. 0 条回复
   - 动动手指，沙发就是你的了！