[逆向][Writeup]ISG2015 flagfinder - .NET程序逆向

转载

weixin_34355715 于 2017-02-16 11:35:00 发布 53 收藏
原文链接：http://www.cnblogs.com/gsharpsh00ter/p/6404940.html

这个题目同样是一道.NET的逆向题，.NET的逆向方法在之前的博文中已经介绍过，这里不做重复的说明。本题的源程序可以在我的github上下载：https://github.com/gsharpsh00ter/reverse

0x01 逆向

flagfinder为.NET编译的PE文件，用dnSpy反编译后，得到如下源码：

```
using System;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Threading;
using Microsoft.VisualBasic.CompilerServices;

namespace flagfinder
{
    // Token: 0x02000007 RID: 7
    [StandardModule]
    internal sealed class Module1
    {
        // Token: 0x06000012 RID: 18 RVA: 0x000020D4 File Offset: 0x000002D4
        [STAThread]
        //遍历所有驱动器上的所有文件
        public static void Main()
        {
            DriveInfo[] drives = DriveInfo.GetDrives();
            checked
            {
                for (int i = 0; i < drives.Length; i++)
                {
                    DriveInfo driveInfo = drives[i];
                    if (driveInfo.IsReady)
                    {
                        Module1.SearchDir(driveInfo.RootDirectory);
                    }
                }
            }
        }

        // Token: 0x06000013 RID: 19 RVA: 0x0000210C File Offset: 0x0000030C
        public static void SearchDir(DirectoryInfo dir)
        {
            checked
            {
                try
                {
                    FileInfo[] files = dir.GetFiles();
                    for (int i = 0; i < files.Length; i++)
                    {
                        FileInfo file = files[i];
```

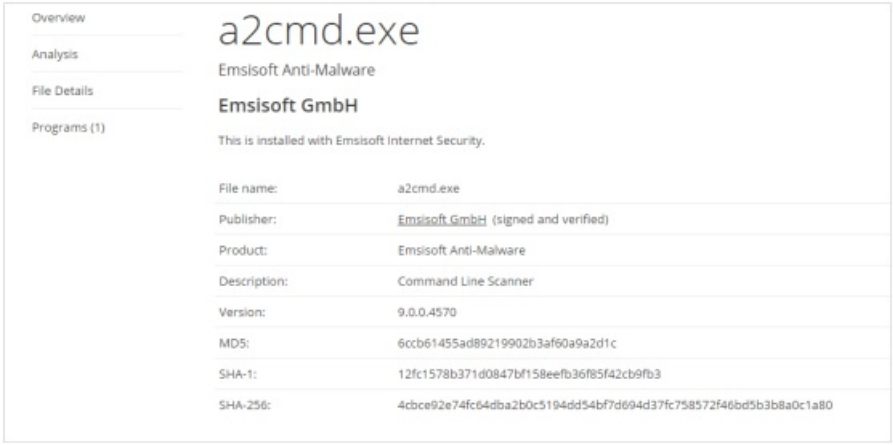
```
                        //对于每一个文件，会进行检查，找到符合条件的文件会打印flag
                        Module1.CheckFile(file);
                    }
                    DirectoryInfo[] directories = dir.GetDirectories();
                    for (int j = 0; j < directories.Length; j++)
                    {
                        dir = directories[j];
                        Module1.SearchDir(dir);
                    }
                }
                catch (Exception expr_49)
                {
                    ProjectData.SetProjectError(expr_49);
                    Console.WriteLine("Unable to search: " + dir.FullName);
                    ProjectData.ClearProjectError();
                }
            }
        }

        // Token: 0x06000014 RID: 20 RVA: 0x00002198 File Offset: 0x00000398
        public static void CheckFile(FileInfo file)
        {
            try
            {
                Console.WriteLine("Analyzing " + file.FullName + " ...");
                //此处为判断条件，文件内容的MD5为指定值后会计算sha256哈希值，并打印flag
                MD5CryptoServiceProvider mD5CryptoServiceProvider = new MD5CryptoServiceProvider();
                if (mD5CryptoServiceProvider.ComputeHash(file.OpenRead()).SequenceEqual(Module1.target))
                {
                    SHA256CryptoServiceProvider sHA256CryptoServiceProvider = new
SHA256CryptoServiceProvider();
                    Console.WriteLine("We've found the flag on your hard drive:");
                    Console.WriteLine("ISG{" +
BitConverter.ToString(sHA256CryptoServiceProvider.ComputeHash(file.OpenRead())).ToLower() + "}");
                    Environment.Exit(0);
                }
                Thread.Sleep(100);
            }
            catch (Exception expr_81)
            {
                ProjectData.SetProjectError(expr_81);
                Console.WriteLine("Unable to read: " + file.FullName);
                ProjectData.ClearProjectError();
            }
        }

        // Token: 0x04000006 RID: 6
        private static byte[] target = new byte[]
        {
            108,
            203,
            97,
            69,
            90,
            216,
            146,
            25,
            144,
            43,
```

```
101             58,
102             246,
103             10,
104             154,
105             45,
106             28
107         };
108     }
109 }
```

0x02 分析

程序逻辑比较简单，运行后会遍历驱动器上的所有文件，并对每一个文件进行检查。如果某个文件的内容经过MD5哈希后是指定的值(target)，则该文件为flag文件，此时会计算器sha256的哈希值，并打印flag。

显然我们不太可能根据已有的MD5值去逆向破解出文件的内容。可以用google查一下对应的MD5值。在http://www.herdprotect.com/a2cmd.exe-12fc1578b371d0847bf158eefb36f85f42cb9fb3.aspx这个页面，我们发现相关的信息：



MD5: 6ccb61455ad89219902b3af60a9a2d1c

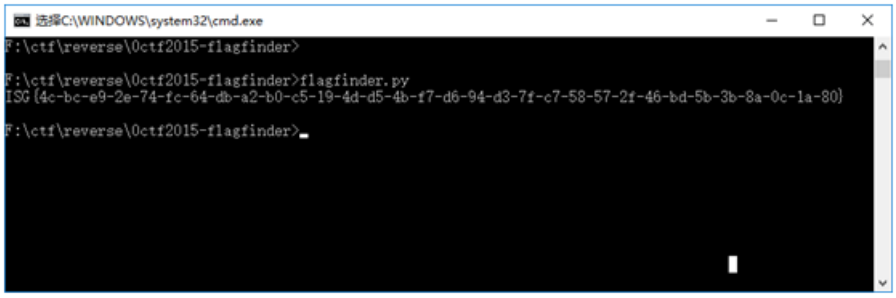
Sha256: 4cbce92e74fc64dba2b0c5194dd54bf7d694d37fc758572f46bd5b3b8a0c1a80

这应该是一个恶意文件的MD5值，虽然我们没有该文件，但是没关系，我们已经得到了sha256的哈希值，根据程序逻辑，我们已经可以自己打印flag了。

打印flag的python代码如下：

```python
#!/usr/bin/python2

#targets = [108, 203, 97, 69, 90, 216, 146, 25, 144, 43, 58, 246, 10, 154, 45, 28]
md5="6ccb61455ad89219902b3af60a9a2d1c"
sha256="4cbce92e74fc64dba2b0c5194dd54bf7d694d37fc758572f46bd5b3b8a0c1a80"
flag="ISG{"

for i in range(0, len(sha256), 2):
    flag = flag + sha256[i:i+2] + "-"
flag += "}"
flag = flag.replace("-}", "}")
print flag
```

运行结果如下：



Flag为：

ISG{4c-bc-e9-2e-74-fc-64-db-a2-b0-c5-19-4d-d5-4b-f7-d6-94-d3-7f-c7-58-57-2f-46-bd-5b-3b-8a-0c-1a-80}

转载于:https://www.cnblogs.com/gsharpsh00ter/p/6404940.html