



Sykipot variant hijacks DOD and Windows smart cards

January 12th, 2012 | Posted by [jaimeliasco](#) in [Attacks](#) | [Malware](#) | [News](#) | [Windows](#)

Defenses of any sort, virtual or physical, are a means of forcing your attacker to attack you on your terms, not theirs. As we build more elaborate defenses within information security, we force our attacker's hand. For instance, in many cases, implementing multi-factor authentication systems just forces the attacker to go after that system directly to achieve their goals. Take the breach at RSA, for example. It has been attributed to attackers who needed the SecurID information to go after their real targets in the defense industry.

Recently, our lab has been talking about Sykipot:

- [Are the Sykipot's authors obsessed with next generation US drones?](#)
- [Another Sykipot sample likely targeting US federal agencies](#)

As we discussed, this malware has been used to launch targeted attacks via "spear phishing" campaigns against targets mainly in the US, since around 2007. According to our research, these attacks originate from servers in China with what appears to be the purpose of obtaining information from the defense sector: the same sector that makes extensive use of PC/SC x509 Smartcards for authentication.

Smartcards have a long history of usage in the Defense Sector, for both physical and information access management, and historically have merely forced attackers to route around the smartcard authentication system through other, more vulnerable attack vectors.

It should come as no surprise, then, that we recently discovered a variant of Sykipot with some new, interesting features that allow it to effectively hijack DOD and Windows smart cards. This variant, which appears to have been compiled in March 2011, has been seen in dozens of attack samples from the past year.

Like we have shown with previous Sykipot attacks, the attackers use a spear phishing campaign to get their targets to open a PDF attachment which then deposits the Sykipot malware onto their machine (the attackers here took advantage of a zero-day exploit in Adobe). Then, unlike previous strains, the malware uses a keylogger to steal PINs for the cards. When a card is inserted into the reader, the malware then acts as the authenticated user and can access sensitive information. The malware is controlled by the attackers from the command & control center.

Here is more detail on the attack:

Smartcard access

The first one is that it creates a new thread with a keylogger routine. The code is very basic, it stores the window name and the keys pressed under a file named MSF5F0.dat on an unencrypted format, example:

```
Title:Internet Explorer
www.google.es
Title:My Computer
```

It uses the WIN32 API's functions [GetKeyState, GetAsyncKeyState, GetForegroundWindow, GetWindowTextA].

It also saves the information contained in the clipboard using the native functions: OpenClipboard, GetClipboardDataand CloseClipboard.

This code is very similar to other pieces of APT's like:

<http://contagiodump.blogspot.com/2010/07/apt-activity-monitor-keylogger.html>

Apart from this we found two more modules that attracted our attention. The first one is capable of listing all the certificates that are stored on the windows key store:

```
loc_1000845C:
lea   eax, [esp+818h+pszNameString]
push  ebx           ; cchNameString
push  eax           ; pszNameString
push  0             ; puTypePara
push  1             ; dwFlags
push  4             ; dwType
push  esi           ; pCertContext
call  edi ; CertGetNameStringA
inc   [esp+818h+var_808]
lea   eax, [esp+818h+pszNameString]
push  eax
push  [esp+81Ch+var_808]
push  offset aD_IssuerS ; "%d.Issuer=%s\r\n"
push  File          ; File
call  ebp ; fprintf
add   esp, 10h
lea   eax, [esp+818h+var_400]
push  ebx           ; cchNameString
push  eax           ; pszNameString
push  0             ; puTypePara
push  0             ; dwFlags
push  4             ; dwType
push  esi           ; pCertContext
call  edi ; CertGetNameStringA
lea   eax, [esp+818h+var_400]
push  eax
push  offset aSubjectS ; "Subject: %s\r\n"
push  File          ; File
call  ebp ; fprintf
add   esp, 0Ch
push  esi           ; pPrevCertContext
push  [esp+81Ch+hCertStore] ; hCertStore
call  ds:CertEnumCertificatesInStore
mov   esi, eax
test  esi, esi
jnz   short loc_1000845C
```

This next routine is called if the command "ci" is present on the config file fetched from the C&C.

```
.text:10005E89 loc_10005E89:          ; CODE XREF: sub_10004E37+F29fj
.text:10005E89          push     2           ; MaxCount
.text:10005E8B          lea     eax, [esp+0EB3Ch+Dst]
.text:10005E92          push    eax         ; "c1"
.text:10005E97          push    eax         ; Str1
.text:10005E98          call   esi ; _strnicmp
.text:10005E9A          add     esp, 0Ch
.text:10005E9D          test   eax, eax
.text:10005E9F          jnz    short loc_10005EAB
.text:10005EA1          call   getCertStoreInfo
.text:10005EA6          jmp    loc_10006CF8
.text:10005EAB :
```

When you insert a smart card into a reader attached to a Windows computer, the certificate on the smart card is registered to the local certificate store on the client computer.

The second one is even more interesting:

```
push  offset a2           ; "2\r\n"
push  File               ; File
call  edi ; fprintf
call  ebx ; Size
push  0 ; Val
push  esi ; Dst
call  nmemset
push  offset aCProgramFilesA ; "C:\\Program Files\\ActivIdentity\\ActivClient"
push  esi ; Dest
call  strcpy
add   esp, 1Ch
push  esi ; lpLibFileName
call  ebp ; LoadLibraryA
test  eax, eax
mov   dword ptr unk_100B53E4, eax
jnz   short loc_100085F8
```

```
lea   eax, [esp+18h+Size]
push  eax
push  ebx
call  [esp+20h+var_4]
mov   eax, [esp+18h+Size]
dec   eax
push  eax ; Size
call  esi ; malloc
mov   esi, eax
mov   eax, [esp+1Ch+Size]
dec   eax
push  eax ; Size
push  ebx ; Src
push  esi ; Dst
call  nmemcpy
mov   eax, [esp+28h+Size]
add   esp, 10h
push  offset aAc_xsi_utilg_0 ; "AC_XSI_UtilGetCardStatus"
lea   ebx, [eax-1]
push  dword ptr unk_100B53E4 ; hModule
call  ebp ; GetProcAddress
test  eax, eax
jnz   short loc_1000869D
```

```
loc_100085F8:
mov   ebp, ds:GetProcAddress
and   [esp+18h+Size], 0
push  offset aAc_xsi_utilg_0 ; "AC_XSI_UtilGetReaderList"
push  eax ; hModule
call  ebp ; GetProcAddress
xor   esi, esi
mov   [esp+18h+var_4], eax
cmp   eax, esi
jnz   short loc_10008629
```

It loads:

```
C:\Program Files\ActivIdentity\ActivClient\acpkcs201.dll
```

(a module that handles some of the functions related with ActivIdentity's ActivClient solution.)

ActivClient is a smart card-based PKI authentication solution for compliance with:

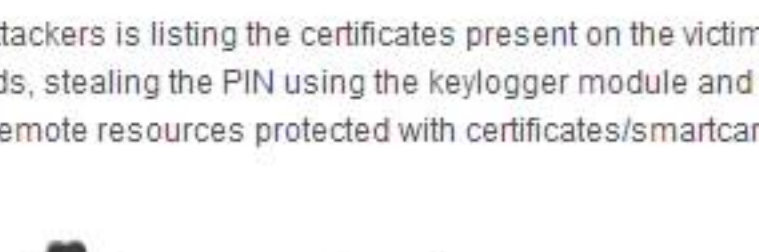
- [U.S. Government Smart Card Interoperability Specifications GSC-IS 2.1](#)
- [U.S. General Services Administration \(GSA\) Basic Services Interface \(BSI\)](#)

(In fact it is one of the platforms used to support the Department of Defense common access card - DoD CoAC)

This routine is called if the command "cm" is present on the config file fetched from the C&C:

```
.text:10005EAB :
.text:10005EAB loc_10005EAB:          ; CODE XREF: sub_10004E37+1068fj
.text:10005EAB          push     2           ; MaxCount
.text:10005EAB          lea     eax, [esp+0EB3Ch+Dst]
.text:10005EAB          push    offset aCn ; "cn"
.text:10005EAB          push    eax         ; Str1
.text:10005EAB          call   esi ; _strnicmp
.text:10005EAB          add     esp, 0Ch
.text:10005EAB          test   eax, eax
.text:10005EAB          jnz    short loc_10005ECD
.text:10005EAB          call   get_CMonitor
.text:10005EAB          jmp    loc_10006CF8
.text:10005ECD :
```

So, the modus operandi of the attackers is listing the certificates present on the victim's computer included the smartcards, stealing the PIN using the keylogger/smartcard and then use this information to log onto remote resources protected with certificates/smartcards.



To log onto protected resources they have implemented the command "krundll", if the C&C sends that command, the victim receives a new dll that implements the required code to login using the certificate and the stolen PIN. This DLL implements the "LoginFunc" and "GetFunc". These methods will contain all the code depending on the application used:

```
loc_10001CD4:
push  400h ; Size
push  edi ; Val
push  offset byte_100B2A80 ; Dst
call  nmemset
push  100h ; Size
mov   esi, offset Dest
push  edi ; Val
push  esi ; Dst
call  nmemset
push  [esp+28h+Source] ; Source
push  esi ; Dest
call  strcpy
add   esp, 20h
push  dword_100B4BCC, 1
push  offset ProcName ; "LoginFunc"
push  hModule ; hModule
call  esi ; GetProcAddress
cmp   eax, edi
mov   dword_100B4BD0, eax
mov   ebp, offset aA ; "a"
mov   ebx, offset FileName
jnz   short loc_10001D72
```

```
loc_10001D8E:
push  offset aPutFunc ; "PutFunc"
push  hModule ; hModule
call  esi ; GetProcAddress
test  eax, eax
mov   dword_100B4BD0, eax
jnz   short loc_10001E0A
```

Summary

We have seen how the attackers are implementing new techniques to bypass two-factor authentication with smartcard/PIN to access protected resources on the victim's network. By capturing the PIN for the smartcard and binding the certificate, malware can silently use the card to authenticate to secure resources, so long as the card remains physically present in the card reader. This is similar to what Mandiant described on the 2011 M-Trends report as a "Smart Card Proxy". While trojans that have targeted smartcards are not new, there is obvious significance to the targeting of a particular smartcard system in wide deployment by the US DoD and other government agencies, particularly given the nature of the information the attackers seem to be [targeting for exfiltration](#).

Implications

As defenses get better, attackers will continue to change their tactics to adapt, and as seen here, will hijack the very systems designed to provide more security, if necessary. An interesting by-product of this malware's necessity of having the card physically present is that attackers can only leverage it for secure authentication to target systems, during times that the user them is physically present at the workstation, making unauthorized activity that much more difficult to discern from legitimate usage. Although smart cards are designed to provide a two factor system of 'chip and pin', again we see that true two-factor authentication is not possible without a physical component that is not accessible digitally.

Navigation

- About
- Alienvault Labs
- Blogs
- Contact
- Docs
 - Articles
 - Presentations
 - Tutorials
- Projects
 - Alienvault Labs Garage
 - Open Source IP Reputation Portal
 - Download IP Reputation Database
 - Latest IPs
 - Real Time Map
 - Open Source Security Event Taxonomy
 - Wireless Intrusion Detection Testing Tool

Recent posts

- New Garage Tool: ClearCutter
- Sykipot variant hijacks DOD and Windows smart cards
- Are the Sykipot's authors obsessed with next generation US drones?
- Another Sykipot sample likely targeting US federal agencies
- Easy entry to SIEM Correlation Rules with Policy Validation