

∨ US

ి Client login



Cyber Security ▶ Research Blog

Emissary Panda – A potential new malicious tool

18 May 2018

By Nikolaos Pantazopoulos









Reverse Engineering

Vulnerability

Threat Intelligence

Introduction

Hacking groups linked t most active along with as TG-3390, APT 27 and related with education,

In the past, Emissary Pa Hacking Team leak. Usu believed to have Chines

Recent research showe compromised machine with the same people w

Attribution

While attribution is always tool based on the follow

- Several code similarities
- Tools were found on co
- ChinaChopper, a web s order to interact with the
- The publicly available n
- A modified mimikatz wh
- The Hunter tool, a web
- Use of DLL Hijacking ted

This website makes use of cookies.

This website uses cookies to ensure you get the best experience on our website. Some of these cookies are essential for the site to function, whilst others are useful to help us to analyse our traffic and collect statistics on how and when the site is being used and how it can be improved. We also share information about your use of our site with our analytics partners.

By clicking "Accept All," you consent to the storing of cookies on your device to enhance site navigation, analyse site usage, and assist in measuring the success of our marketing efforts.

Manage your preferences at any time, on the cookie preference tool at the bottom of the screen, or if you want to learn more, please go to our Cookie policy

Accept all cookies

Reject all cookies

Necessary cookies

Necessary cookies enable core functionality as you browse our website such as session management and remembering your cookie preferences. The website cannot function properly without these cookies and can only be disabled by changing your browser preferences.

Analytical Cookies

Analytical cookies help us to improve our website by collecting and reporting information on

ave tended to be the ida' group, also known ed organisations

e exploits from the a tool which is

eing found in recent ch has possible ties

development of this

ols are: password is required in



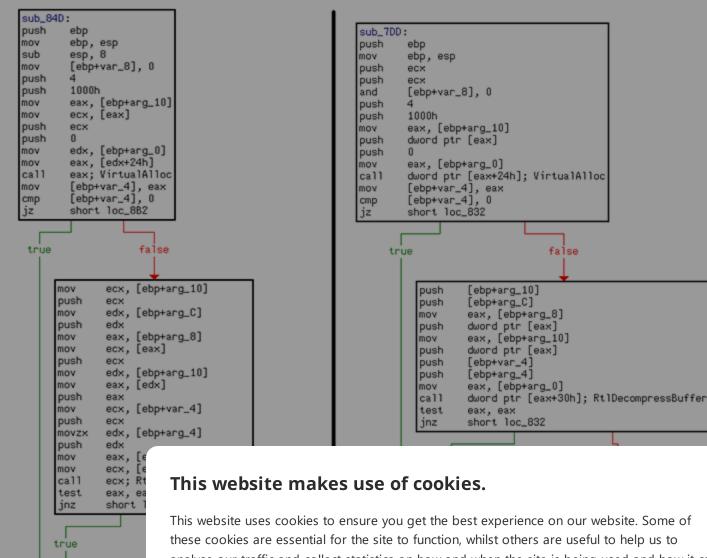


Figure 1: Old sample or

eax, [ebp+var_8]

mov mov

pop retn

```
[ebp+var_2E], cx
mov
pop
        ecx
        44h ; 'D'
push
        [ebp+var_2C], cx
mov
        ecx
pop
        56h; 'V'
push
mov
        [ebp+var_2A], cx
pop
        ecx
        50h ; 'P'
push
mov
        [ebp+var_28], cx
pop
        4Fh; '0'
push
        [ebp+var_26], cx
mov
pop
        ecx
        43h; 'C'
push
        [ebp+var_24], cx
mov
pop
        ecx
        58h ; 'X'
push
mov
        [ebp+var_22], cx
pop
        ecx
        4Fh; '0'
push
        [ebp+var_20], cx
mov
mov
        ecx, eax
        [ebp+var_1E], cx
mov
        ecx
pop
        43h; 'C'
push
mov
pop
        ecx
push
        58h ; X
mov
```

analyse our traffic and collect statistics on how and when the site is being used and how it can be improved. We also share information about your use of our site with our analytics partners.

By clicking "Accept All," you consent to the storing of cookies on your device to enhance site navigation, analyse site usage, and assist in measuring the success of our marketing efforts.

Manage your preferences at any time, on the cookie preference tool at the bottom of the screen, or if you want to learn more, please go to our Cookie policy

Necessary cookies

Necessary cookies enable core functionality as you browse our website such as session management and remembering your cookie preferences. The website cannot function properly without these cookies and can only be disabled by changing your browser preferences.

Analytical Cookies



Analytical cookies help us to improve our website by collecting and reporting information on

```
6Eh ; 'n'
                                                     push
        [ebp+var_1C], cx
                                                     mov
                                                             [ebp+var_1A], cx
                                                             75h; 'u'
                                                     push
        [ebp+var_1A], cx
                                                     mov
                                                             [ebp+var_16], ax
pop
                                                     pop
                                                             eax
        75h ; 'u'
push
                                                             72h ; 'r'
                                                     push
        [ebp+var_16], ax
mov
                                                             [ebp+var_14], ax
                                                     mov
pop
                                                     pop
                                                             eax
        72h; 'r'
push
                                                             6Ch ; '1'
                                                     push
        [ebp+var_14], ax
mov
                                                             [ebp+var_12], ax
pop
        eax
                                                     pop
        6Ch ; '1'
push
                                                             [ebp+var_10], ax
                                                     mov
        [ebp+var_12], ax
mov
                                                     xor
                                                             eax, eax
pop
                                                             [ebp+var_E], ax
        [ebp+var_10], ax
mov
                                                     push
                                                             104h
xor
        eax, eax
                                                             eax, [ebp+var_234]
                                                     lea
        [ebp+var_E], ax
mov
                                                     xor
                                                             edi, edi
push
                                                     push
        eax, [ebp+var_23C]
lea
                                                             edi
                                                     push
        edi, edi
xor
                                                             [ebp+var_18], cx
                                                     mov
push
        eax
                                                     mov
                                                             [ebp+var_8], edi
        edi
push
                                                     call
                                                             dword ptr [esi+10h] ; GetModuleFileNameV
mov
        [ebp+var_18], cx
                                                             eax, eax
                                                     test
mov
        [ebp+var_C], edi
                                                             loc_245
        dword ptr [esi+10h] ; GetModuleFileNameW
call
```

Figure 2: Old sample on the left side, our sample on the right side

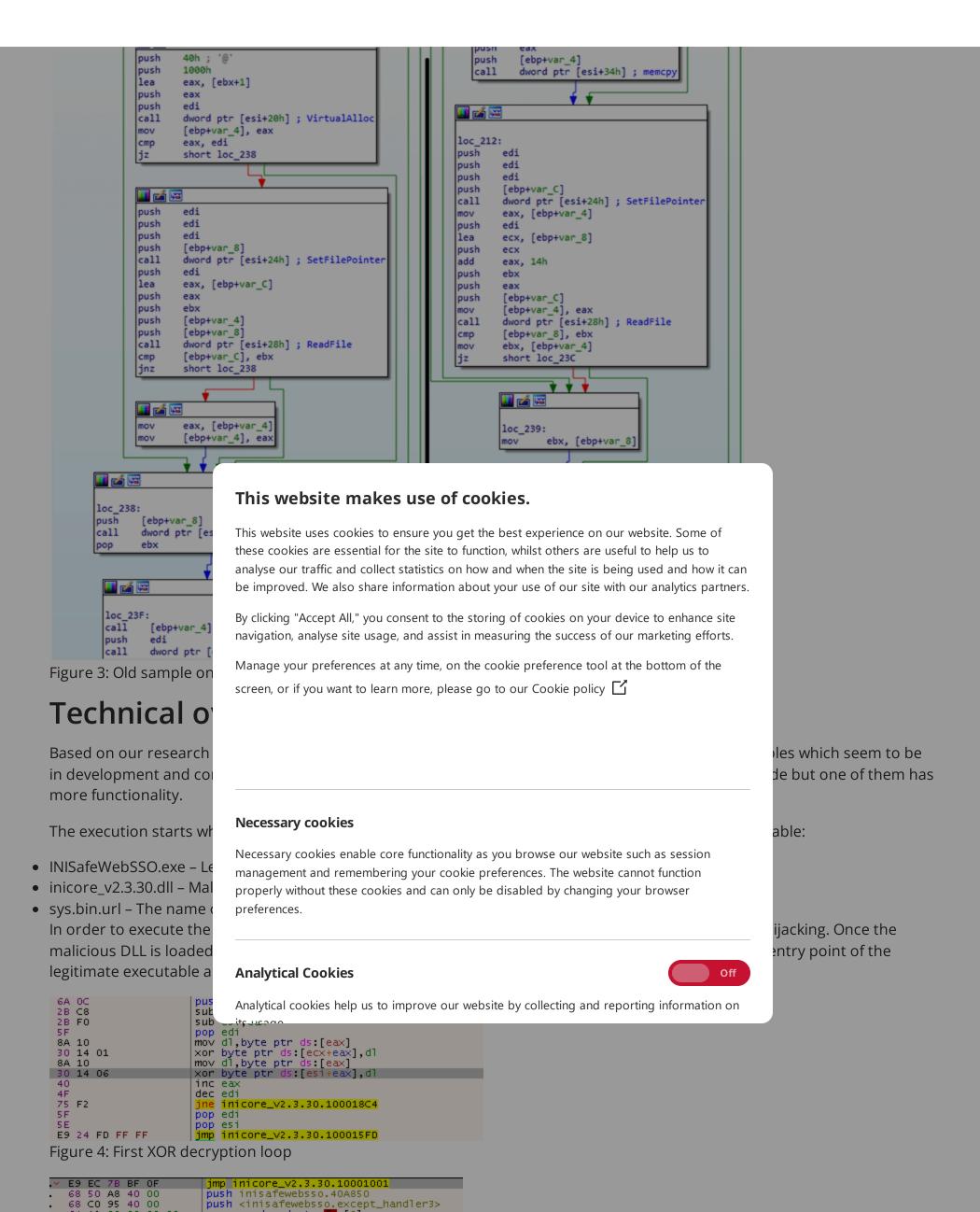


Figure 5: Patched entry point

After jumping back to the DLL, it will repeat the same process to decrypt a part of itself and find the addresses of LoadLibrary and GetProcAddress to load all the necessary functions dynamically.

Lastly, it will read the sys.bin.url file and the execution will transfer to it. Once this is done, it will XOR decrypt the rest of the malicious payload and decompress it using RtlDecompressBuffer.

Payload

We will focus on the payload with the additional functionality (we will describe the differences between the two samples we found later). Entering the payload, we can see some interesting strings which seem to be used for debugging purposes (see Figure 6 and Figure 7). This is one of the reasons we believe that the tool is still in development.

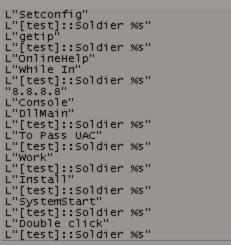


Figure 6: Debug strings

L"mydebugview::Soldier %s" L"Failed to initialize security." L"mydebugview::Soldier %s" L"Failed to create IWbemLocator object." L"mydebugview::Soldier %s"

Figure 7: Debug strings

The action taken is base below.

No. of parameters	
	0
	1
	2
	3
	4

This website makes use of cookies.

This website uses cookies to ensure you get the best experience on our website. Some of these cookies are essential for the site to function, whilst others are useful to help us to analyse our traffic and collect statistics on how and when the site is being used and how it can be improved. We also share information about your use of our site with our analytics partners.

By clicking "Accept All," you consent to the storing of cookies on your device to enhance site navigation, analyse site usage, and assist in measuring the success of our marketing efforts.

Manage your preferences at any time, on the cookie preference tool at the bottom of the screen, or if you want to learn more, please go to our Cookie policy

Option 0 - Tern

When the binary is exec

- It will check if it runs fro
- If it runs from the %APF sys.bin.url to it.
- Otherwise, it will create DLL, sys.bin.url) into it,

Option one - Sv

Necessary cookies

Necessary cookies enable core functionality as you browse our website such as session management and remembering your cookie preferences. The website cannot function properly without these cookies and can only be disabled by changing your browser preferences.

Analytical Cookies

Off

Analytical cookies help us to improve our website by collecting and reporting information on

Where the number of passed parameters is one, the payload will read the sys.bin.url file from %appdata%\systemconfig. It will then spawns a new svchost process as C:\windows\system32\svchost.exe –k update in suspended state and injects the payload. Finally, it patches the entry point of svchost.exe so it can execute the malicious payload after the ResumeThread call.

Option two - Persistence svchost injection again

The method of persistence depends on the access rights. If the payload's process is running from a user with admin rights then it will create a new service. The service name will be taken from the config, in our case the name is systemconfig with 'for system config' as the description of the service. The binary path will be the extracted installer path along with /update as a parameter.

Otherwise, it will add the binary's path to the Software\Microsoft\Windows\CurrentVersion\Run key with —Update as a parameter. If the persistence was done by this method, or not at all, then it will inject into svchost as described in the option one section.

Option three – Core functionality

This is described in detail below in the Core Functionality section.

Option four – UAC Bypass

table and described

will inject the

hree files (executable,

An already public UAC bypass method is included in the binary. It doesn't matter if the method will work or not since the process will exit. This is one more indication that the tool is still in development and there are plans to expand its capabilities.

Core functionality

Currently, the core functionality includes writing the configuration to registry and communicating with the C C server. We did not find any malicious functionality such as uploading or downloading files, or executing attacker's commands.

Config

Each value of the config is written to the registry after encrypting them using the DES algorithm. A new registry key is created under HKEY_CURRENT_USER\Software\Classes using either the SystemProductName value from the HARDWARE\DESCRIPTION\System\BIOS key or the hardcoded string "68A-D3H-B1111 as a name. Additionally, a hardcoded string -HjDWr6vsJqfYb89mxxxx is appended to the name. For example:

- VMware Virtual Service-HjDWr6vsJqfYb89mxxxx or
- Z68A-D3H-B1111-HjDWr6vsJqfYb89mxxxx The key and the IV used in the encryption are based on the first eight bytes of this registry key's name, for example, VMware V.

The encrypted sub-keys are described below. The majority of these sub-keys will not be read from the payload once they have been written. This might suggest that there are plans to expand the functionality of the tool. We wrote a Python script to automate the identification of the registry low and description of the sub-key values. [1] A summary of the decrypted values

can be found in the follo This website makes use of cookies. Key name in sample two Desc This website uses cookies to ensure you get the best experience on our website. Some of these cookies are essential for the site to function, whilst others are useful to help us to Bin Paylo analyse our traffic and collect statistics on how and when the site is being used and how it can be improved. We also share information about your use of our site with our analytics partners. Console N/A By clicking "Accept All," you consent to the storing of cookies on your device to enhance site navigation, analyse site usage, and assist in measuring the success of our marketing efforts. .3.30.dll DII Hijac paylo Manage your preferences at any time, on the cookie preference tool at the bottom of the screen, or if you want to learn more, please go to our Cookie policy Group N/A **GUID** GUID Uniq CoCr MD5 N/A qfYb89mxxxx **Necessary cookies** OnlineHelp Store 157:443 Necessary cookies enable core functionality as you browse our website such as session **Path** Path management and remembering your cookie preferences. The website cannot function nData\systemconfig\ properly without these cookies and can only be disabled by changing your browser PE preferences. bSSO.exe Legit load Periodic N/A **Analytical Cookies** Off Analytical cookies help us to improve our website by collecting and reporting information on **Process** Proce Serv Service name systemconfig systemconfig Service description for systemconfig for systemconfig ServDis

Differences between the two samples

As mentioned before, the two samples share a lot of code but there are many differences between them. Two important differences which should be highlighted are:

- Each sample has different debug strings.
- The sample with less functionality needs to read and decrypt the stored registry values in order to communicate with the C C or to inject to svchost. This is because the config is not included in the binary.
 A summary of the differences can be found below:

Functionality	Sample one	Sample two
Persistence	✓	
UAC bypass	✓	
C&C Communication	✓	✓
Write config to registry	✓	
Read registry values	✓	✓
Inject to svchost	✓	✓
x64 Injection		✓
Debug strings	✓	✓
Execution based on params	✓	✓
WMI execution	✓	

Conclusion

Emissary Panda is still a have any malicious fund

References

[1] https://github.com/r

Previous work carried

https://www.securewor

https://www.securewor

IOCs

CCIP

103.59.144.183

159.65.80.157

Registry value

HjDWr6vsJqfYb89mx

Z68A-D3H-B1111

C:ProgramData

This website makes use of cookies.

This website uses cookies to ensure you get the best experience on our website. Some of these cookies are essential for the site to function, whilst others are useful to help us to analyse our traffic and collect statistics on how and when the site is being used and how it can be improved. We also share information about your use of our site with our analytics partners.

By clicking "Accept All," you consent to the storing of cookies on your device to enhance site navigation, analyse site usage, and assist in measuring the success of our marketing efforts.

Manage your preferences at any time, on the cookie preference tool at the bottom of the screen, or if you want to learn more, please go to our Cookie policy

Necessary cookies

Necessary cookies enable core functionality as you browse our website such as session management and remembering your cookie preferences. The website cannot function properly without these cookies and can only be disabled by changing your browser preferences.

Analytical Cookies

Off

Analytical cookies help us to improve our website by collecting and reporting information on its usage

systemconfig Check for the presence of this key in: SoftwareMicrosoftWindowsCurrentVersionRun

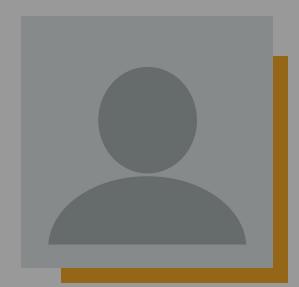
Systemconfig Check for this service with description: "for systemconfig"

File Name	SHA-256
INISafeWebSSO.exe	C501203FF3335FBFC258B2729A72E82638719F60F7E6361FC1CA3C8560365A0E
inicore_v2.3.30.dll	4D65D371A789AABE1BEADCC10B38DA1F998CD3EC87D4CC1CFBF0AF014B783822
sys.bin.url	2B2BB4C132D808572F180FE4DB3A0A3143A37FDECE667F8E78778EE1E9717606
sys.bin.url	3E718F39DFB2F6B8FBA366FEFA8B7C127DB1E6795F3CAAD2D4A9F3753EEA0ADC

Published date: 18 May 2018

Written by: Nikolaos Pantazopoulos and Thomas Henry

ed samples do not attacks.



Nikolaos Pantazopoulos

Terms and Conditions Privacy Policy Contact Us

This website makes use of cookies.

This website uses cookies to ensure you get the best experience on our website. Some of these cookies are essential for the site to function, whilst others are useful to help us to analyse our traffic and collect statistics on how and when the site is being used and how it can be improved. We also share information about your use of our site with our analytics partners.

By clicking "Accept All," you consent to the storing of cookies on your device to enhance site navigation, analyse site usage, and assist in measuring the success of our marketing efforts.

Manage your preferences at any time, on the cookie preference tool at the bottom of the screen, or if you want to learn more, please go to our Cookie policy

Necessary cookies

Necessary cookies enable core functionality as you browse our website such as session management and remembering your cookie preferences. The website cannot function properly without these cookies and can only be disabled by changing your browser preferences.

Analytical Cookies



Analytical cookies help us to improve our website by collecting and reporting information on its usage

onse Hotline or cirt@nccgroup.com