Emissary Panda – A potential new malicious tool

18 May 2018 By Nikolaos Pantazopoulos

Research Reverse Engineering Vulnerability Threat Intelligence

Introduction

Hacking groups linked t ave tended to be the most active along with I da' group, also known as TG-3390, APT 27 and ed organisations related with education,

In the past, Emissary Pa e exploits from the Hacking Team leak. Usu a tool which is believed to have Chines

Recent research showe eing found in recent compromised machine ch has possible ties with the same people w

Attribution

While attribution is alwa development of this tool based on the follow

- Several code similaritie
- Tools were found on co ols are:
- ChinaChopper, a web sl password is required in order to interact with th
- The publicly available n
- A modified mimikatz wh
- The Hunter tool, a web
- Use of DLL Hijacking tec

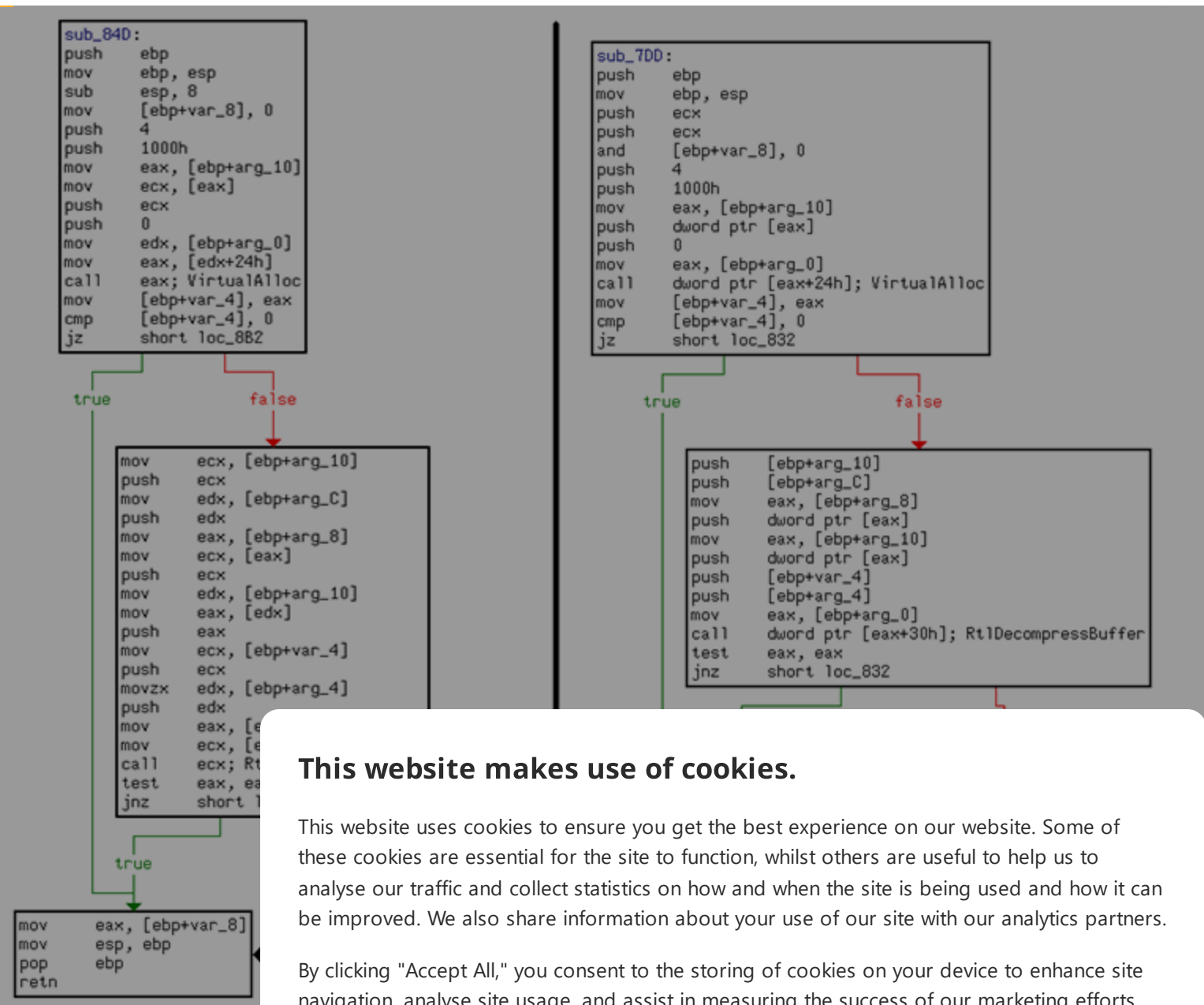


Figure 1: Old sample on

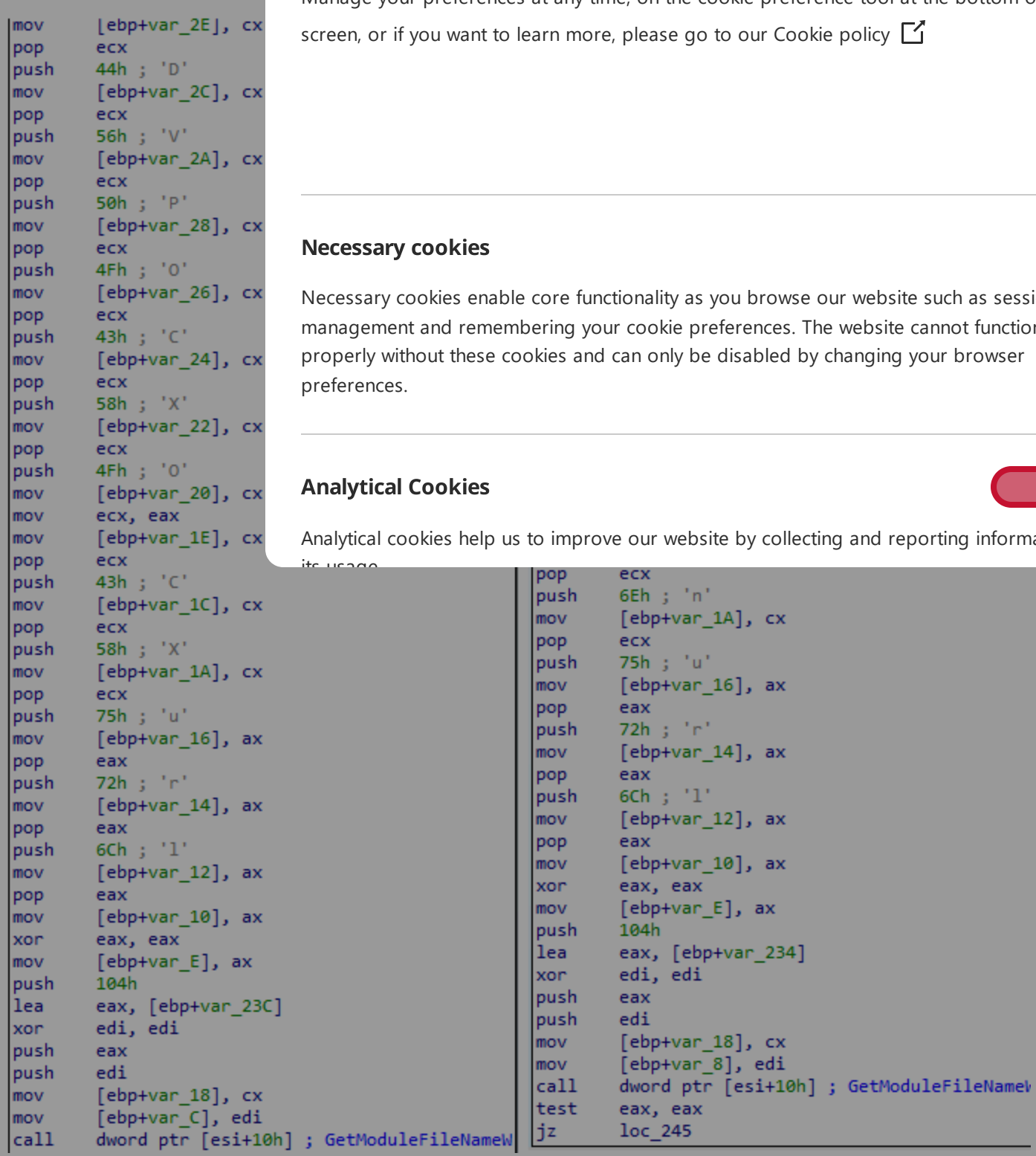


Figure 2: Old sample on the left side, our sample on the right side

Figure 3: Old sample on

Technical o

Based on our research les which seem to be in development and co de but one of them has more functionality.

The execution starts wh able:

- INISafeWebSSO.exe – Le
- inicore_v2.3.30.dll – Mal
- sys.bin.url – The name

In order to execute the ijacking. Once the malicious DLL is loaded entry point of the legitimate executable a

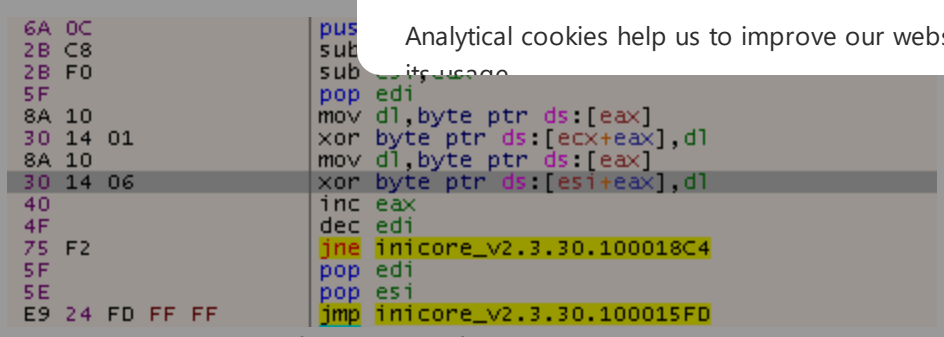


Figure 4: First XOR decryption loop

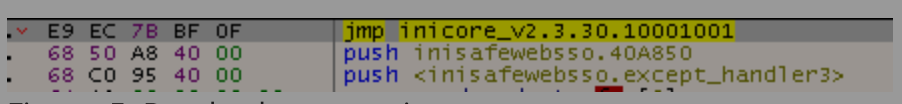


Figure 5: Patched entry point

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.

```
L"Setconfig"
L"[test]::Soldier %s"
L"getip"
L"[test]::Soldier %s"
L"OnlineHelp"
L"While In"
L"[test]::Soldier %s"
"8.8.8.8"
L"Console"
L"DllMain"
L"[test]::Soldier %s"
L"To Pass UAC"
L"[test]::Soldier %s"
L"Work"
L"[test]::Soldier %s"
L"Install"
L"[test]::Soldier %s"
L"SystemStart"
L"[test]::Soldier %s"
L"Double click"
L"[test]::Soldier %s"
```

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 table and described below.

No. of parameters
0
1
2
3
4

Option 0 – Tern

When the binary is exec

- It will check if it runs fro
- If it runs from the %APP will inject the sys.bin.url to it.
- Otherwise, it will create hree files (executable, DLL, sys.bin.url) into it,

Option one – Sv

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

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 key and decryption of the sub-key values [1]. A summary of the decrypted values can be found in the foll

Key name	Desc		in sample two
Bin	Paylo		
Console	N/A		
Dll	Hijac paylo		.3.30.dll
Group	N/A		
GUID	Uniq CoCr		GUID
MD5	N/A		qfYb89mxxxx
OnlineHelp	Store		157:443
Path	Path		mData\systemconfig\
PE	Legiti load		bSSO.exe
Periodic	N/A		
Process	Proce		e
Serv	Service name	systemconfig	systemconfig
ServDis	Service description	for systemconfig	for systemconfig

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 ... ed samples do not have any malicious func ... attacks.

References

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

Previous work carried o

https://www.securewor

https://www.securewor

IOCs

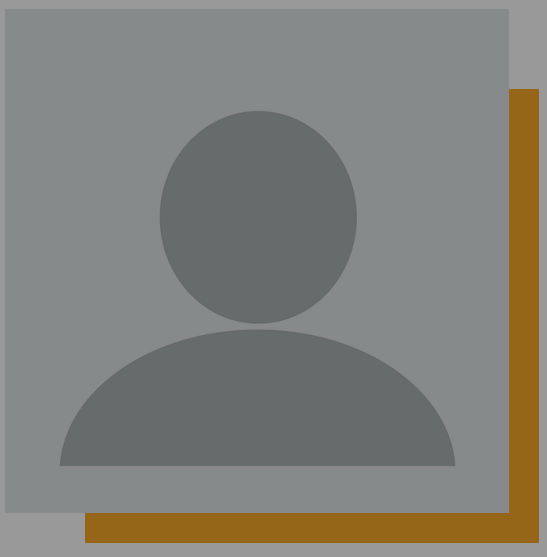
C C IP
103.59.144.183
159.65.80.157

Registry value	
HjDWr6vsJqfYb89mx	
Z68A-D3H-B1111	
C:ProgramData	
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



Nikolaos Pantazopoulos