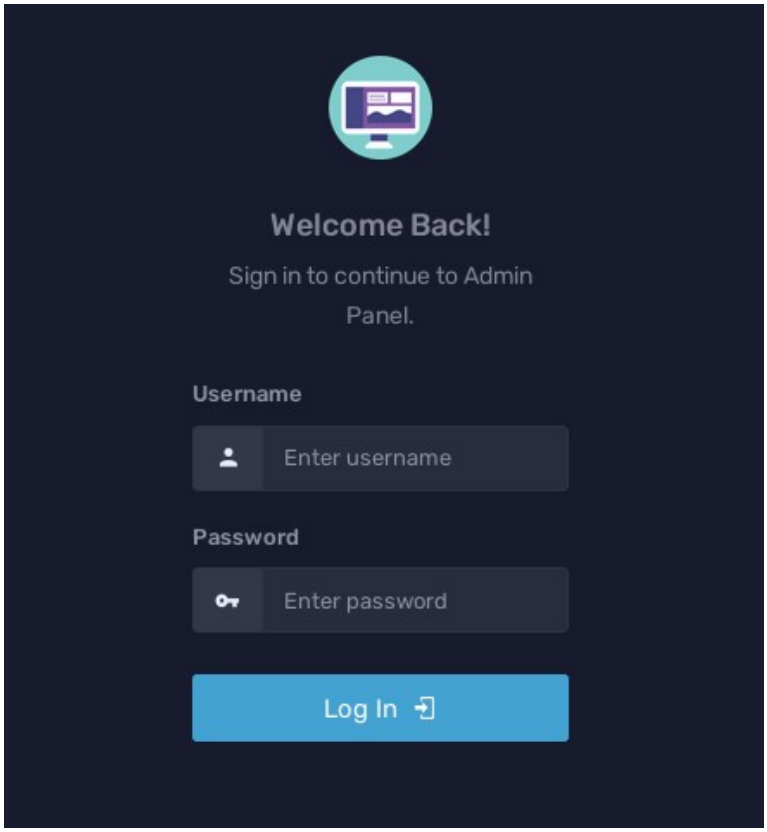
PsiXBot: The Evolution Of A Modular .NET Bot

blog.fox-it.com/2019/03/27/psixbot-the-evolution-of-a-modular-net-bot/

March 27, 2019



Summary

In this blog we will share our analysis of a modular piece of malware which is referred to by the author as PsiXBot. The malware first surfaced in 2017 but has recently undergone significant developments of its core and modules, which include the logging of keystrokes and stealing of Outlook and browser credentials. With these new developments done and the first large scale distributions observed in the wild, PsiXBot has officially made its debut in the malware ecosystem.

Introduction

Fox-IT actively monitors cyber criminal activity on a daily basis in order to proactively identify threats that are relevant to our customers. On the 21st of February 2019 we noticed *SmokeLoader*, a popular bot used to install additional malware on infected machines for a fee, push a task in order to distribute a .NET malware sample. Further research on the sample revealed a bot with a modular nature and capabilities, such as stealing data from infected hosts as well as receiving download & execute tasks. Our interest was further peaked when the Spelevo Exploit Kit started distributing the same malware on the 16th of March at which point we decided to further investigate this piece of malware, resulting in the findings below.

Having seen it evolve since 2017 to now getting out from beta versioning, we observe its being distributed by multiple infection vectors, such as exploit kits and malware loaders.

Analysis

During routine threat research activity, we stumbled upon a tweet on what seemed to be an early version of the malware known as *PsiXBot*. In the same Twitter thread a link was also shared to a very early version of that same malware.

An overview of the versions are found below:

- Mid 2017: First version spotted in the wild (*SHA256: d2ee07bf04947cac64cc372123174900725525c20211e221110b9f91b7806332*);
- August 2018: Updated version spotted (*SHA256: ce0e46fa1c5b463ed4a070a05594a79203ed2dd5df96cece9f875e2957fda4fa*);
- Early 2019: The latest version is now being distributed via different infection vectors (*SHA256: ca30c42334fcc693320772b4ce1df26fe5f1d0110bc454ec6388d79dffea4ae8*).

The illustration below displays the code structure of the versions: the first version to the left and the most recent one to the right.

Note that the name *PsiX* is derived from the name of the assembly. Taking a look to the PDB path, the same naming is again present: `G:\WORK\MONEY\BOT\NoName\PsiX\obj\Release\NoNameBot.pdb`

Additionally, taking a look at the sample displayed in the center of the image above, the name “Radius” can be spotted as well. Part or whole name can be also observed among the C&Cs such as: `radcall.bit`, `radbot.bit` and `rrradiusspace.bit`.

The differences we observed across the versions are mostly:

- New commands supported by adding different modules;
- Encryption of the strings with AES;

- Version number updated from `Beta 1.0.0` to `1.0.1`.

The analysis below is centered on the most recent version.

As you could have already seen, the malware is written in .NET and it is not obfuscated. Typically it's distributed within a dropper which hides the main payload. Once executed the *Main()* function is invoked. It first verifies that it is the only running instance by looking to a hardcoded mutex (for the sample analyzed: `gfdhfyf543543cdsdfsdf`), then it executes a loop to mimic sleep function before activating.

Most of the strings are encrypted with AES by a hardcoded string key. For this sample the key is: `1243hewqr8n1220g321&^*&^Tb0`

The malware also checks the language settings of the victim, if the language is set to `ru-RU` (Russian) the malware will exit. For all other language settings the malware will continue its malicious activity.

An additional check is relating the filename to ensure it is matching the one configured. In instances where it is not, it proceeds with the installation process, which is done by invoking the *CopyEx* method via *WMI* and invoking the copied binary again via WMI. The installation path is: `Local\Microsoft\.exe` in the `%APPDATA%` folder.

After the installation the malware contacts the configured C&Cs, that are initialized by the following code:

[code lang=c]
public static string[] valid = new string[]
{
"pppoe.bit",
"weather0.bit",
"mygranny.bit",
"six6.bit",
"learncpp.bit"
};
[/code]

In order to communicate with the *.bit* domains it uses hardcoded DNS servers (`193.37.213.223` for the sample used in this analysis). Upon the DNS resolution, it sends a ping to the C&Cs in order to identify the first one that is up.

The bot reports to its C&C some information gathered from the infected host. An example of the string used in the request is:

[code lang=text]
action=call&user_name=john&bot_id=D63BAFF79F3A3504C70DC3298EE74C68&av=N/A&os_major=Microsoft Windows 7 Home Basic N &permissions=User&os_bit=64&cpu=Intel(R) Core(TM) i7-6820HQ CPU @ 2.70GHz&gpu=Standard VGA Graphics Adapter&ram=2048&hdd=C:12345/67890&version=1.0.1&user_group=Admin&pc_net=4.0
[/code]

The meaning of the parameters are:

- *action*: the purpose of the request;
- *user_name*: username of the victim's host;
- *bot_id*: unique string to identify the infected host;
- *av*: name of the AntiVirus software installed;

- *os_major*: name of the OS installed;
- *permissions*: username's permissions;
- *os_bit*: OS architecture;
- *cpu*: CPU model;
- *gpu*: GPU model;
- *ram*: RAM available;
- *hdd*: HDD Serial Number;
- *version*: version number of the malware (latest one is 1.0.1);
- *user_group*: User group name the username is part of;
- *pc_net*: version of .NET framework installed.

The data transfer is encrypted with RC4 using a hardcoded key which for this sample is `63a6a2eea47f74b9d25d50879214997a`. It is interesting to note that the author encrypted most of the strings, except the RC4 encryption key, C&Cs and DNS servers which are in plain-text.

The C&C answers with a JSON string, like the following one:

[code lang=c]
{
result_code: [
{
"result_code": "200",
}
]
}
[/code]

If the server returns a valid response, the malware sleeps for 95 seconds before requesting a new command to execute. This is done by sending the data:

[code lang=text]
action=command&bot_id=D63BAFF79F3A3504C70DC3298EE74C68
[/code]

An example of a response is the following one:

[code lang=c]
{
"result_code": [
{
"result_code": "200"
}
],
"commands": [
{
"command_id": "1485",
"command_action": "GetSteallerPasswords",
"command_data": "",
"command_arg": ""
},
{

"command_id": "1486",
"command_action": "StartSchedulerModule",
"command_data": "",
"command_arg": ""
}
]
}
[/code]

In the example above the C&C asks for the execution of two commands. The `command_action` value is the exact name of the method that must be invoked. The malware will resolve the method dynamically in accordance to this value. In order to avoid hardcoded strings, the author implemented an easy way to upgrade the malware: if the method name sent by the C&C is not present in the malware during execution, the method/call is simply ignored.
For one particular method invocation the malware uses a type named *SukaBlyat*, which is an offensive term used as Russian slang.

The two received commands are used in order to ask the C&C server for additional modules. The data transfered requesting the module is:

[code lang=text]
action=module&bot_id=D63BAFF79F3A3504C70DC3298EE74C68&module_action=SchedulerModule
[/code]

Subsequently, the module is downloaded and executed, while sending the following command:

[code lang=text]
action=result&command_id=1485&command_result=6E756C6C
[/code]

The commands currently supported are:

[code lang=text]
Download
DownloadAndExecute
Execute
GetInstalledSoft
GetKeylogs
GetOutlook
GetProcessesList
GetScreenShot
GetSteallerCookies
GetSteallerPasswords
StartAndroidModule
StartBTC
StartComplexModule
StartKeylogger
StartNewComplexModule
StartSchedulerModule
StopProcess
[/code]

Modules

The modules available for this recent version of the bot are:

- *BrowserModule (assembly name stMod.exe)*: used to dump passwords or cookies from a variety of browsers as well as from FileZilla FTP client. It accepts an argument to specify which is the data to be dumped: `-passes` for the password or `-cookies` for the cookies. The program returns a string with all the stolen information. It seems to be based on the QuasarRAT project;
- *BTCModule (assembly name LESHI.exe)*: accepts an argument and a cryptocurrency address. The supported address types are: `-btc` , `-ether` , `-ltc` , `-monero` and `-ripple` . Once such an address is configured, the program proceeds to monitor the clipboard every 3 seconds and verifies if the copied text is a valid address and of which type. If the check is successful the malware replaces the text with one of the configured wallet addresses;
- *ComplexModule (assembly name Client.exe)*: an old version of the open source rat QuasarRAT. In particular the *xclient* string (which is part of this module namespace) was present in a fork from 2016 (see `https://github.com/GeekGalaxy/QuasarRAT`). Also within the decompiled source code we retrieved the type name *QuasarClient*;
- *KeyLoggerModule (assembly name KeyLoggerModule.exe)*: uses the *SetWindowsHookEx* API in order to set a global hook and intercept keystrokes. The intercepted keys are saved in a file named `Logger.log` ;
- *NewComplexModule (assembly name RemoteClient.exe)*: implements a remote desktop like program. It allows streaming the desktop user, interacting with it and starting the browser. The code does not seem to be anything publicly available;
- *OutlookModule (assembly name OutlookPasswordRecovery.dll)*: dumps the Outlook passwords and returns a string with the information retrieved.
- *SchedulerModule (assembly name Scheduler.exe)*: used to ensure persistence. It just creates a scheduled task to run the bot each 60 seconds.

Distribution

Typically, there are two ways of spreading such a strain of malware: infecting new unwitting victims and leveraging existing compromised systems. Fox-IT observes the PsiXBot actors are able to do both – delivering their malware via malspam campaigns or exploit kits (such as the Spelevo Exploit Kit) as well as using services offered on underground markets to load malware on (pre-infected) devices such as SmokeLoader.

SmokeLoader

The SmokeLoader bot from which we received the task to distribute the PsiXBot malware was configured with the below metadata:

[code lang=text]
server_rc4_key_recv e097b3a6
server_rc4_key 40e5223b
bot_version 2018
seller new1

c&c http://5gssghhs2w.org/2/
c&c http://dvhwzq.ru/2/
c&c http://hdxaet.ru/2/
c&c http://hghwwgh6.info/2/
c&c http://jdcbhs.ru/2/
c&c http://kdcbst.ru/2/
c&c http://kkted54d.ru/2/
c&c http://si3213gher.com/2/
c&c http://vshmesz.com/2/
c&c http://vygxxhh.bit/2/
[/code]

The distribution URL sent by *SmokeLoader*'s task is:

`hxxp://favoritfile.in/7-8.exe`

From the referenced distribution URL we managed to download the sample with *SHA256: 9b8c0c82fe79ae15e0f723d6aa267d38d359a7260613a091a2d70d770488e919*

The C&Cs of this sample are:

[code lang=text]
myauto.bit
sokoban.bit
[/code]

Spelevo Exploit Kit

With regards to the Spelevo Exploit Kit, the sample distributed is identified by *SHA256: ca30c42334fcc693320772b4ce1df26fe5f1d0110bc454ec6388d79dffea4ae8*

The C&C servers of this sample are:

[code lang=text]
learncpp.bit
mygranny.bit
pppoe.bit
six6.bit
weather0.bit
[/code]

Another distribution vector observed is spam mailings. One of the spam campaigns we managed to identify is Italian themed, with the following metadata:

[code lang=text]
Receiver from pecfe04.sogei.it (pecfe04.sogei.it [26.2.42.237]) by PECP-BE02 (lmtpd) with LMTP id 28663.002; Tue, 8 Jan 2019 16:22:51 +0100 (CET)
Receiver from PECP-FE04 ([127.0.0.1]) by pecfe04.sogei.it (Dovecot) with LMTP id 474fM6e/NFysCAAAxEz/xA ; Tue, 08 Jan 2019 16:22:51 +0100
Receiver from mx.pec.sogei.it (localhost [127.0.0.1]) by smtps.pec.sogei.it (Postfix) with ESMTP id 43YwxQ6pm5zgYCT for <protocollo@pec.agenziariscossione.gov.it>; Tue, 8 Jan 2019 16:22:50 +0100 (CET)

Receiver from smtps.pec.aruba.it (smtpecgo01.pec.aruba.it [80.88.94.21]) by mx.pec.sogei.it (Postfix) with ESMTPS for <protocollo@pec.agenziariscossione.gov.it>; Tue, 8 Jan 2019 16:22:50 +0100 (CET)
Receiver from avvocatismcv.com (ipvspec1.pec.ad.aruba.it [62.149.152.1]) by smtps.pec.aruba.it (Postfix) with ESMTPSA id 43YwxQ2V8Sz2L7hcc; Tue, 8 Jan 2019 16:22:50 +0100 (CET)
Reply-To luigi.ferrandino@avvocatismcv.com
Return-Path luigi.ferrandino@avvocatismcv.com
Attachments ["daticert.xml", "Nuovi_contratti_2019__145038.zip", "smime.p7s"]
Number_of_attachments 3
Date 2019-01-08 15:22:50 (UTC)
To ["luigi.ferrandino@avvocatismcv.com"]
From "Per conto di: luigi.ferrandino@avvocatismcv.com" <posta-certificata@pec.aruba.it>
Subject POSTA CERTIFICATA: Re: Notificazione ai sensi della legge n. 53 del 1994
[/code]

The zip contained a JavaScript (*SHA256: e4006cde4a96048ff25727459abfae0ffd37985d04581793be784d7cf50e32d7*) which, once executed, tried to fetch the next stage from the following URL:

`hxxp://img.martatovaglieri.it/index?83836`

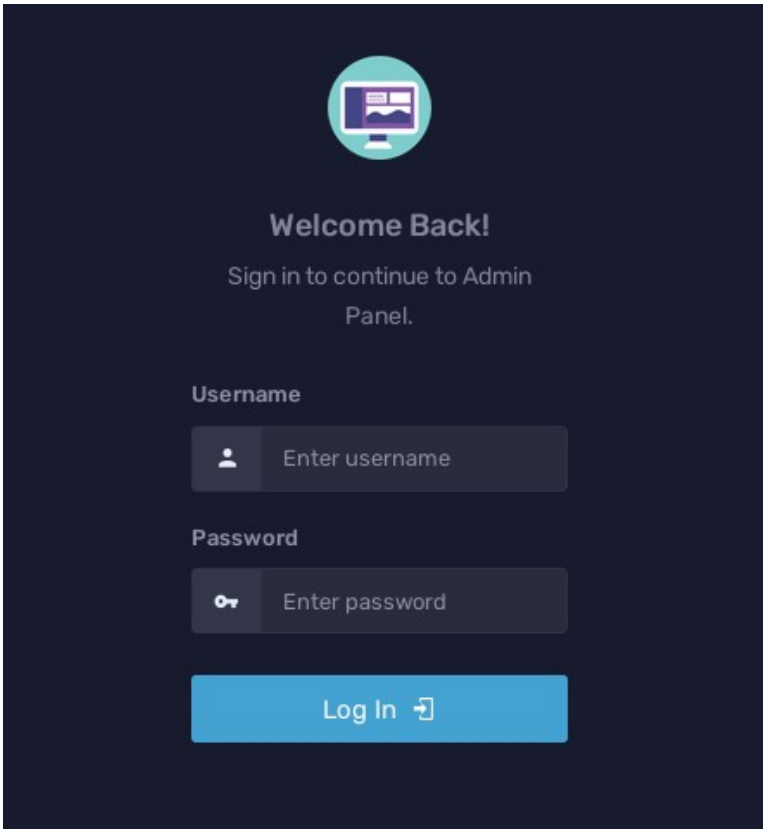
The binary downloaded from this URL can be identified with *SHA256: db1f57ffd6c58e1d40823e2c8834e45a67271557ceaa1b3bcccf4feab83243a1*.

The C&C of this sample is:

[code lang=text]
anyname.bit
[/code]

Panel

The screenshot below shows PsiXBot's login panel:



The following noteworthy code is inside the HTML:

```
<!-- saved from url=(0043)hxxps://kyrkymalol.000webhostapp.com/admin/ -->
```

IOCs

The most relevant malware hashes can be found below:

[code lang=text]
PsiXBot first version: d2ee07bf04947cac64cc372123174900725525c20211e221110b9f91b7806332
PsiXBot updated version: ce0e46fa1c5b463ed4a070a05594a79203ed2dd5df96cece9f875e2957fda4fa
PsiXBot latest version: ca30c42334fcc693320772b4ce1df26fe5f1d0110bc454ec6388d79dffea4ae8
BrowserModule: 6a9841b7e19024c4909d0a0356a2eeff6389dcc1e2ac863e7421cca88b94e7e0
SchedulerModule: 6e123ce5c7c48132f057428c202638eb9d0e4daa690523619316a9f72b69d17f
BTCModule: 3846fcfdc6414685efd217a88613ed3383a61f8313a0aa7ecdcde8ed99c8ebac
KeyLoggerModule: 7bac9b3b5598059db770cdeee74f0b1cf3078c2cb54cc2fcd18ae811d42a5d63
ComplexModule: 0f931fec3fd436d974d767f84f66b44f6f2fc168d9e6d77b2aa2e9d3bf4cd604
NewComplexModule: a5edab1596346358c8899b9f81ec49b0560da929327e0ef08ab51dcb277c9b70
OutlookModule: b01fbb8cfeb16c4232fddea6dea53212a57e73ef32ee20056cd69d29570bf55c
[/code]

The full set of indicators of compromise can be found on our GitHub page.
More information on the threat actors behind PsiXBot is available for InTELL customers on Fox-IT's cybercrime portal.

Stefano Antenucci & Antonio Parata (Fox-IT InTELL)