Win32/Lethic Botnet Analysis

resources.infosecinstitute.com/win32lethic-botnet-analysis/

Introduction

Lethic is a spam botnet consisting of an estimated 210 000 – 310 000 individual machines which are mainly involved in pharmaceutical and replica spam. At the peak of its existence, the botnet was responsible for 8-10% of all the spam sent worldwide.

Around early January 2010, the botnet was dismantled by Neustar employees, who contacted various Lethic internet service providers in a bid to take control of the botnet's command and control servers. This move temporarily caused the botnets' spam to decrease to a trickle of its original volume.

In February 2010, the owners of the botnet managed to re-establish control over the botnet, using new command and control servers located in the United States. The takedown has decreased the spam volume of the botnet, however. As of February 2010, the botnets' amount of spam was down to a third of its original. As of April 2010, the botnet has an estimated 1.5% share of the spam market, sending about 2 billion spam messages a day.

This article presents a view on the malware and its capabilities, how it communicates with the CnC, encryption scheme used as well as different protection mechanisms to make the malware analyst job harder.

Tools

- OllyDBG / IDA Pro.
- Lethic sample (MD5 = 23DE74A6122A8AB3B02EFD3B2C481978, password = infected)
- Lethic Unpacked (password = infected)

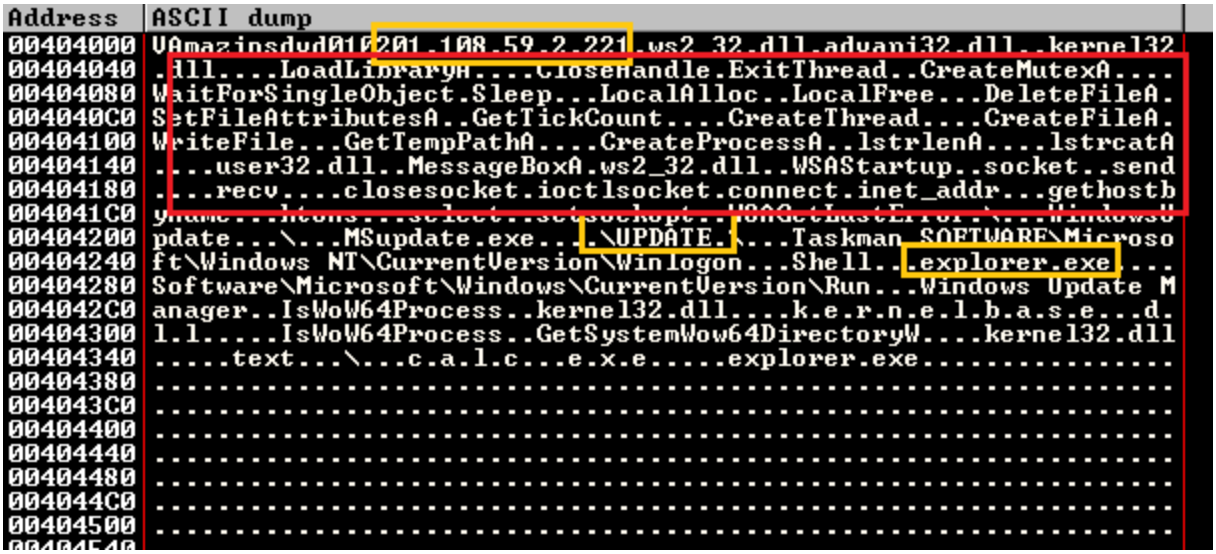
Infection vector

This botnet arrives as attachments to spammed messages disguised as notifications from familiar contacts.

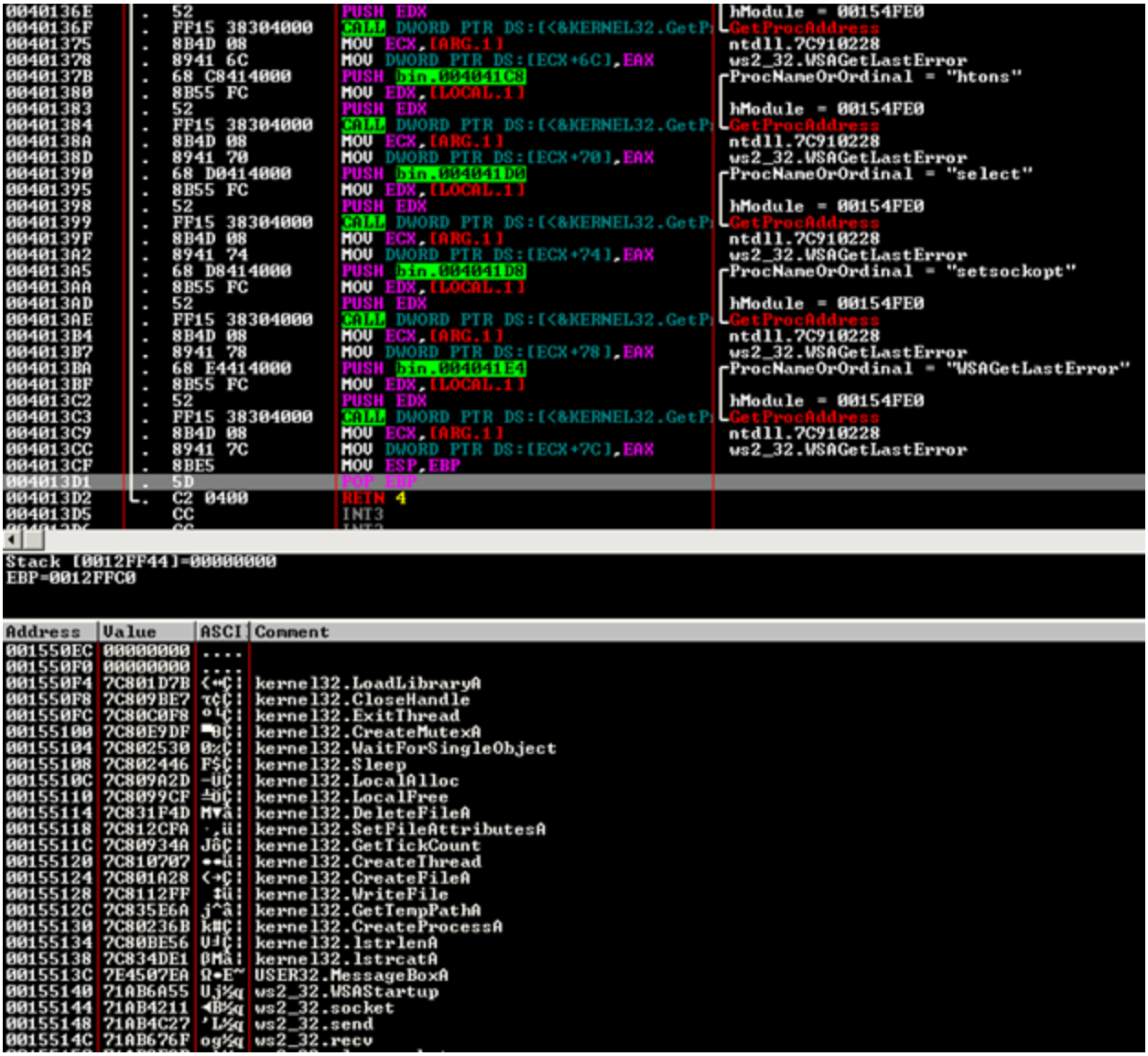
Bot analysis

I will not talk about the unpacking process because there is no relevant information regarding it. The original Lethic build file is about 17 kb. Not the smallest botnet ever seen, Tinba or Gamarue have lower size but it is a low profile botnet with many infections that may be the secret of its longevity.

Let's go, fire up OllyDBG. OK, you are at the entry point of the malware. Looking at the data section, you could see some interesting strings like APIs names to be resolved, modules that are going to be loaded, also CnC IP address, etc… Just by looking at this dump, you could basically get a whole image about the inner working of this bot.



The first thing, it will try to resolve API dynamically using LoadLibrary and GetProcAddress. Put a BP at the RET instruction to see the whole import table filled with the functions pointers:



1. Following it creates a directory in the APPDATA directory under the name of:

“C:Documents and SettingsAdministratorApplication DataWindowsUpdate”

Then, it makes a copy of the file under the name of “MSupdate.exe” in this directory and deletes the original file.

1. Following, to autostart with windows, it creates two registry keys at:

“HKEY_LOCAL_MACHINESOFTWAREMicrosoftWindows NTCurrentVersionWinlogon” under the name “**Taskman**” which points to the executable file specified above.

1. And, another key under the name of “**Windows Update Manager**” in:

“HKEY_LOCAL_USERSoftwareMicrosoftWindowsCurrentVersionRun”

1. Additionally, it modifies the “**shell**” registry entry located at:

“HKEY_LOCAL_MACHINESOFTWAREMicrosoftWindows NTCurrentVersionWinlogon” with the value:

explorer.exe,C:Documents and SettingsAdministratorApplication DataWindowsUpdateMSupdate.exe

Next, it checks whether the operating system is a 32-bit or 64-bit environment by calling IsWoW64Process. For my case, I am running a Windows XP 32bits; this is generally done before performing code injection.

Process Injection

The code injection method used in Lethic is VirtualAllocEx/WriteProcessMemory /CreateRemoteThread in explorer.exe process. To follow debugging at explorer.exe process, you could either replace the first byte of the buffer with 0xCC and make OllyDBG a JIT debugger, and when you step through CreateRemoteThread, a new instance of OllyDBG will fire up and will attach to your process, then you can restore back the first byte to its original state. On the other hand, you could look at the start address parameter in CreateRemoteThread, start a new instance of OllyDBG, and put a BP on it.

Once you are done from this step, Lethic creates a new mutex name “VAmazinsdvd010201” to prevent duplicate process from running in the same machine. Next, it loads some libraries in explorer memory space and right after that; it starts preparing communication with the C&C using Winsock APIs.

C&C Communication

After it successfully receives data from the CnC, depending on the value of the command.

Let’s list and explain what the different commands sent by the C&C can be, and what they do:

- Add Server (0x01): the data includes a public mail server IP address and a port. Lethic then creates a socket and connects to the said mail server. Upon success, it initializes a new MailServerRecord with ID, the given IP and port, socket, and then inserts the record in cc_hdr->Chain. Now that the record has been added to the chain, should any of the following operations on the record fail, Lethic will inform the C&C server; and the latter will then send a “Remove Server” command to remove the faulty record.
- Remove Server (0x02): removes record corresponding to ID from the mail server record chain.

- Send Mail (0x03): sends the buffer data to a mail server, via the MailServerRecord pointed to by ID. The result is sent back to the C&C server, which in turns decides whether it should set the RecvFeedBack flag or not.
- Clean (0x04, 0x05): do some cleaning work, such as freeing the whole MailServerRecords chain, deleting the malware installer, and exit.
- Reserved (0x06), no further operation except for sending the received data back.
- Add Server By Name (0x11), the same as Add Server, the only difference is that a hostname is given instead of an IP address.
- Receive FeedBack (0x13), a flag that is used to set MailServerRecord.bfbFlag.

If the received buffer doesn't include any of the aforementioned commands, Lethic checks each record of the chain. If the bfbFlag is set to TRUE, that means the Send Mail operation was successful, and feedback was received from the mail server. It thus initializes FEEDBACK with the record ID, command, feedback data, and size, and sends it all to the C&C server.

In all aspects, the Zombie acts as a slightly improved mail relay; it forwards received data from the C&C server to mail servers chosen in a managed list. That is not very efficient, in terms of bandwidth consumption, as compared to the traditional template-based approach: Indeed here, the C&C server has to send almost as much data as is sent by each Zombie. Logically, it seems to indicate that Lethic botnets all need to be small to operate smoothly.

Figure 4 shows an example of the spam content.

```
220 mwinf5d14 ME ESMTP server ready
EHLO wanadoo.fr
250-mwinf5d14 hello [78.42.40.76], pleased to meet you
250-HELP
250-AUTH LOGIN PLAIN
250-SIZE 44000000
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 OK
AUTH LOGIN
334 VXNlcm5hbWU6
ZXZlLmZsZXVyeQ==
334 UGFzc3dvcmQ6
MTIwNDgx
235 2.7.0 ... authentication succeeded
MAIL FROM:<eve.fleury@wanadoo.fr>
250 2.1.0 <eve.fleury@wanadoo.fr> sender ok
RCPT TO:<leslieenergy203@gmail.com>
250 2.1.5 <leslieenergy203@gmail.com> recipient ok
DATA
354 enter mail, end with "." on a line by itself
Subject:
From: "CHARLES FISHER" <chase_fisher@hotmail.com>
Content-Type: text/plain;
.charset="us-ascii"
X-Mailer: iPhone Mail (8L1)
Date: Mon, 30 Jun 2014 15:34:27 +0200
To: "leslieenergy203" <leslieenergy203@gmail.com>
Content-Transfer-Encoding: quoted-printable
Mime-Version: 1.0 (1.0)

Hi leslieenergy203


http://getfreehomeloanadvice.com.au/cgi-bin/moon.php?gwce2383dt
```

Conclusions

Lethic is yet another spambot to join the fray. It is unclear what its future holds, and we do not know when it emerged. However this shows how “full” the “ecosystem” for spambots is. Lethic’s complexity is minimal when compared to other spam botnets (no rootkit seen, etc) but it appears effective enough at this time.