Win32/Lethic Botnet Analysis

I 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.

	ASCII dump
00404000	VAmazinsdud010201_108_59_2_221_ws2_32_d11_advani32_d11kerne132_
00404040	.111LoadLibraryHCloseHandle.ExitThreadCreateMutexA
00404080	WaitForSingleObject.SleepLocalAllocLocalFreeDeleteFileA.
	SetFileAttributesAGetTickCountCreateThreadCreateFileA.
00404100	WriteFileGetTempPathACreateProcessAlstrlenAlstrcatA
00404140	user32.dllMessageBoxA.ws2_32.dllWSAStartupsocketsend
00404180	recvclosesocket.ioctlsocket.connect.inet_addrgethostb
004041C0	ynamehtonsselectsetsockoptVSAGetLastErfor
00404200	pdate\MSupdate.exe
00404240	ft\Windows_NT\CurrentVersion\WinlogonShellexplorer.exe
00404280	Software\Microsoft\Windows\CurrentVersion\RunWindows Update M
	anagerIsWoW64Processkerne132.d11k.e.r.n.e.1.b.a.s.ed.
	1.1IsWoW64ProcessGetSystemWow64DirectoryWkerne132.dl1
	text\c.a.l.ce.x.eexplorer.exe
00404380	
004043C0	
00404400	
00404440 00404480	
00404480 004044C0	
00404460	
00404500	

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:

0040136E	. 52	PUSH EDX	hModule = 00154FE0
0040136F	. FF15 38304000	CALL DWORD PTR DS: [<&KERNEL32.GetP	
00401375	. 8B4D 08	MOU ECX, LARG.1] MOU DWORD PTR DS:[ECX+6C],EAX	ntd11.7C910228
00401378	. 8941 6C	HOU DWORD PTR DS:[ECX+6C],EAX	ws2_32.WSAGetLastError
0040137B	. 68 C8414000	PUSH bin.004041C8	<pre>ProcNameOrOrdinal = "htons"</pre>
00401380	. 8B55 FC		
00401383	. 52	HOU EDX, LLOCAL, 1] PUSH EDX	hModule = 00154FE0
00401384	DD4E 20204000	CALL DWORD PTR DS: [<&KERNEL32.GetP	Cat Page Oddage
00401384 0040138A	. FF15 38304000	TOU FOR TARC 41	at 111 2001 0020
		MOU ECX, [ARG.1]	ntdl1.7C910228
0040138D	. 8941 70	HOU DWORD PTR DS:[ECX+70],EAX	ws2_32.WSAGetLastError
00401390	. 68 D0414000	PUSH bin.004041D0	<pre>ProcNameOrOrdinal = "select"</pre>
00401395	. 8B55 FC	HOU EDX, LLOCAL.1 I	
00401398	. 52	PUSH EDX	hModule = 00154FE0
00401399	. FF15 38304000	CALL DWORD PTR DS: [<&KERNEL32.GetP	GetProcAddress
0040139F	. 8B4D 08	HOU ECX, [ARG.1] HOU DWORD PTR DS:[ECX+74],EAX	ntd11.7C910228
00401362	. 8941 74	HOU DWORD PTR DS:[ECX+74],EAX	ws2_32.WSAGetLastError
004013A5	. 68 D8414000	PICH bio 004041 DS	<pre>ProcNameOrOrdinal = "setsockopt"</pre>
004013AA	. 8B55 FC	HOU EDX. (LOCAL.1)	a beauting a second be
004013AD	50	PUSH EDX	hModule = 00154FE0
004013AE	TT+F 20204000	TUON LUOD DTD DC+1/2/CDUEL22 C++D	
	. FF15 38304000	CALL DWORD PTR DS: [<&KERNEL32.GetP	
004013B4	- 8B4D 08	HOU ECX, [ARG.1] HOU DWORD PTR DS:[ECX+78],EAX	ntd11.7C910228
004013B7	. 8941 78	HOU DWORD PIR DS: LECX+781, ERX	ws2_32.WSAGetLastError
004013BA	. 68 E4414000	PUSH bin.004041E4	<pre>ProcNameOrOrdinal = "WSAGetLastError"</pre>
004013BF	. 8B55 FC	MOU EDX, LLOCAL.1]	
004013C2	. 52	PUSH EDX	hModule = 00154FE0
004013C3	. FF15 38304000	CALL DWORD PTR DS: [<&KERNEL32.GetP	
004013C9	. 8B4D 08	HOV ECX, LARG.1]	ntd11.7C910228
004013CC	. 8941 7C	HOU DWORD PTR DS:[ECX+7C],EAX	ws2_32.WSAGetLastError
004013CF	. 8BE5	MOU ESP, EBP	haz _22 . Houge ends erriot.
004013D1	. 5D		
004013D2	L. C2 0400 CC	RETN 4	
00401305	66	INT3	
004013D5	00		
<	00	1NI 3 1NT 2	
✓ Stack [0012]	FF44]=00000000	1113	
<	FF44]=00000000		
✓ Stack [0012]	FF44]=00000000	1113	
✓ Stack [0012]	FF44]=00000000		
Stack (0012 EBP=0012FFC	FF44]=00000000 0	1122	
Address Ua	FF44]=988899999 8 1ue ASCI Comment	1122	
Address Ua	FF44]=988899999 8 1ue ASCI Comment	1122	
Address Ua	FF44]=988899999 8 1ue ASCI Comment	1172	
Address Ua	FF44]=988899999 8 1ue ASCI Comment	2.LoadLibraryA	
Address Ua	FF44]=988899999 8 1ue ASCI Comment	2.LoadLibraryA	
Address Ua Ø01550EC 00 001550FF 00 001550F8 00 001550F8 20 001550F8 20 001550F8 20 001550F8 20 001550F8 20	CC FF44]=000000000 0 1ue ASCI Connent 000000 000000 80107B <<*C: kernel3	2.LoadLibraryA 2.CloseHandle 2.ExitThread	
Stack (0012 EBP=0012FFC Address Ua 001550FC 00 001550F6 00 001550F8 7C 001550F7 7C 001550F6 7C 001550F7 7C 001550F6 7C 001550F7 7C 001550F6 7C 001550F6 7C 001550F7 7C 001550F6 7C 001550F7 7C 001550F8 7C 001550F9 7C	CC FF44]=00000000 0 1ue ASCI Comment 000000 000000 000000 801D7B C+C: 809BE7 tC: x802BF 0'C: 8050F -0'C: 80E9DF -0'C:	2.LoadLibraryA 2.CloseHandle 2.ExitIhread 2.CreateMutexA	
Address Ua 001550EC 00 001550F4 00 001550F4 00 001550F7 00 001550F8 00 001550F7 00 001550F8 00 001550F9 00 001550F9 00 001550F9 00 001550F0 00 001550F8 00 00155100 70 00155104 70	CC FF44]=0000000 0 1ue 000000 000000 00107B 2002 801D7B 4*C: kernel3 80C0F8 *C: *C: kernel3 80C0F8 *C: *C: kernel3 802530 80254 80254	2.LoadLibraryA 2.CloseHandle 2.ExitThread 2.CreateMutexA 2.WaitForSingleObject	
Address Ua 801550EC 00 001550FC 00 001550F4 00 001550F5 00 001550F4 00 001550F5 00 001550F4 00 001550F4 00 001550F4 00 001550F4 00 00155108 7C 00155108 7C	CC FF44]=00000000 0 1ue ASCI Connent 000000 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExitInread 2.CreateMutexA 2.WaitForSingleObject 2.Sleep	
Address Ua 801550EC 00 001550FC 00 001550F4 00 001550F5 00 001550F4 00 001550F5 00 001550F4 00 001550F4 00 001550F4 00 001550F4 00 00155108 7C 00155108 7C	CC FF44]=00000000 0 1ue ASCI Comment 000000 000000 801D7B <	2.LoadLibraryA 2.CloseHandle 2.ExitInread 2.CreateMutexA 2.WaitForSingleObject 2.Sleep	
Stack (0012 EBP=0012FFC Address Ua 001550EC 00 001550F0 00 001550F4 7C 001550F8 7C 00155108 7C	CC FF44]=00000000 0 1ue ASCI Comment 000000 000000 801D7B <	2.LoadLibraryA 2.CloseHandle 2.ExitIhread 2.CreateMutexA 2.WaitForSingleObject 2.Sleep 2.LocalAlloc	
Address Ua 001550EC 00 001550FC 00 001550F7 00 001550F7 00 001550F8 00 001550F0 00 001550F7 00 001550F8 00 001550F9 00 001550F0 00 00155108 00 00155108 00 00155108 00 00155108 00 00155108 00 00155108 00	CC FF44]=00000000 0 1ue ASCI Connent 000000 000000 000000 801D7B <mc: kernel3<="" td=""> 809BE7 %C: kernel3 8060F8 %C: kernel3 8062530 %C: kernel3 802540 %C: kernel3 802540 %C: kernel3 802540 %C: kernel3 802446 F\$C: kernel3 80992D -UC: kernel3 80942D -UC: kernel3</mc:>	2.LoadLibraryA 2.CloseHandle 2.CroseHandle 2.ExitThread 2.VaitForSingleObject 2.VaitForSingleObject 2.Sleep 2.LocalAlloc 2.LocalFree	
Address Ua 8tack (0012 EBP=0012FFC 001550EC 00 001550F0 00 001550F4 7C 001550F4 7C 001550F5 000 001550F2 7C 00155100 7C 00155104 7C 00155104 7C 00155108 7C 00155107 7C 00155108 7C 00155108 7C 00155110 7C 001551107 7C 00155110 7C	CC FF44]=00000000 0 1ue ASCI Connent 000000 000000 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.CroseHandle 2.ExitThread 2.VaitForSingleObject 2.VaitForSingleObject 2.Sleep 2.LocalAlloc 2.LocalFree	
Stack [0012 EBP=0012FFC 001550EC 000 001550FC 001 001550F7 02 001550F7 02 001550F8 02 00155100 02 00155100 02 00155100 02 00155100 02 00155100 02 00155100 02 00155100 02 00155100 02 00155100 02 00155100 02 00155110 02 00155110 7C 00155110 7C 00155111 7C 00155111 7C	CC FF44]=00000000 0 1ue ASCI Comment 000000 000000 801D7B C:: 809087 xCI: 809087 xCI: 809087 xCI: 809087 xCI: 809087 xCI: 809088 xCI: 8052530 BZC: 8025446 FSC: 80920 -UC: 80921 -UC: 80925 HIX: 80926 -UC: 80927 Acrnel 3 80928 -UC: 80928 -UC: 80927 HIX: 831F4D HIX: 8312CFA -, üi:	2.LoadLibraryA 2.CloseHandle 2.ExitIhread 2.CreateMutexA 2.WaitForSingleObject 2.Sleep 2.LocalAlloc 2.LocalFree 2.DeleteFileA 2.SetFileAttributesA	
Address Ua Ø01550EC 00 001550FC 00 001550F7 00 001550F8 00 001550F8 00 001550F4 00 001550F6 00 001550F7 00 001550F8 00 00155104 00 00155104 00 00155104 00 001551104 00 001551104 00 001551104 00 001551104 00 001551104 00 001551104 00 001551104 00 001551104 00 001551104 00 001551114 00 00155114 00 00155114 00	CC FF44]=00000000 0 1ue ASCI Comment 000000 000000 809000 809000 809000 809000 809000 809000 809000 809000 8002530 0%C: kerne13 8092530 0%C: kerne13 8092530 0%C: kerne13 80925446 F\$C: kerne13 80920 -ÜC: kerne13 80921<-ÜC: kerne13	2.LoadLibraryA 2.CloseHandle 2.ExitThread 2.CreateMutexA 2.WaitForSingleObject 2.Steep 2.LocalAlloc 2.LocalFree 2.DeleteFileA 2.SetFileAttributesA 2.GetFileAttributesA	
Stack (0012 EBP=0012FFC 001550EC 000 001550F0 000 001550F8 000 001550F8 000 001550F8 000 00155100 7C 00155104 7C 00155104 7C 00155107 7C 00155108 7C 00155108 7C 00155110 7C 00155110 7C 001551120 7C 00155114 7C 001551120 7C 001551120 7C	CC FF44]=00000000 0 1ue ASCI Connent 000000 000000 000000 000000 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExiThread 2.CreateMutexA 2.WaitForSingleObject 2.LocalAlloc 2.LocalFree 2.DeletFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread	
Stack [0012 EBP=0012FFC 001550EC 000 001550FC 001 001550F7 00 001550F8 00 001550F8 00 00155100 00 00155108 00 00155108 00 00155108 00 00155108 00 00155108 00 00155108 00 00155108 00 00155108 00 00155110 00 00155	CC FF44]=00000000 0 1ue ASCI Connent 000000 000000 000000 000000 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExitIhread 2.CreateMutexA 2.WaitForSingleObject 2.SlocalAlloc 2.LocalFree 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateFileA	
Stack [0012 EBP=0012FFC 001550EC 00 001550F4 7C 001550F7 000 001550F8 7C 00155100 7C 00155100 7C 00155104 7C 00155104 7C 00155104 7C 00155104 7C 00155104 7C 00155104 7C 00155110 7C 00155110 7C 00155111 7C 00155120 7C	CC FF44]=00000000 0 1ue ASCI Connent 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.CloseHandle 2.ExitThread 2.CreateMutexA 2.WaitForSingleObject 2.WaitForSingleObject 2.JocalAlloc 2.LocalFree 2.LocalFree 2.LocalFree 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateFileA 2.WriteFile	
Stack (0012 EBP=0012FFC 001550EC 00 001550F4 7C 001550F4 7C 001550F4 7C 001550F4 7C 001550F4 7C 001550F4 7C 001550F6 00 00155100 7C 00155100 7C 00155100 7C 00155110 7C 00155110 7C 00155112 7C 00155112 7C 00155120 7C	CC FF44]=00000000 0 1ue ASCI Connent 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExiThread 2.CreateMutexA 2.WaitForSingleObject 2.LocalAlloc 2.DeleteFileA 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateTheA 2.WriteFile 2.WriteFile 2.GetTempPathA	
Address Ua Address Ua 001550EC 00 001550F8 CC 001550F7 CC 001550F8 CC 001550F8 CC 001550F8 CC 001550F8 CC 00155104 CC 00155106 CC 00155114 CC 00155116 CC 00155116 CC 00155117 CC 00155118 CC 00155124 CC	CC FF44]=00000000 0 1ue ASCI Connent 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExiThread 2.CreateMutexA 2.WaitForSingleObject 2.LocalAlloc 2.DeleteFileA 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateTheA 2.WriteFile 2.WriteFile 2.GetTempPathA	
Address Ua Address Ua 001550EC 00 001550F8 CC 001550F7 CC 001550F8 CC 001550F8 CC 001550F8 CC 001550F8 CC 00155104 CC 00155106 CC 00155114 CC 00155116 CC 00155116 CC 00155117 CC 00155118 CC 00155124 CC	CC FF44]=00000000 0 1ue ASCI Connent 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExiThread 2.CreateMutexA 2.WaitForSingleObject 2.LocalAlloc 2.DeleteFileA 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateTheA 2.WriteFile 2.WriteFile 2.GetTempPathA	
Stack (0012 EBP=0012FFC 001550EC 00 001550F4 0C 001550F4 7C 001550F4 7C 001550F4 7C 001550F4 7C 001550F4 7C 001550F4 7C 00155100 7C 00155104 7C 00155108 7C 00155110 7C 00155112 7C 00155112 7C 00155124 7C 00155128 7C	CC FF44]=00000000 0 1ue ASCI Connent 000000 801D7B <+C:	2.LoadLibraryA 2.CloseHandle 2.ExitInread 2.CreateMutexA 2.Sleep 2.LocalAlloc 2.LocalFree 2.DocalFree 2.DocalFree 2.BetFileAtributesA 2.GetTickCount 2.CreateFileA 2.CreateFileA 2.WriteFile 2.WriteFile 2.GetTempPathA 2.CreateFreeA 2.IstrleeA	
Address Ua Ød1550EC 00 Ød1550FC 00 Ød1550F7 00 Ød1550F8 00 Ød1550F4 00 Ød1550F4 00 Ød1550F4 00 Ød1550F4 00 Ød155104 00 Ød155104 00 Ød1551108 00 Ød1551120 00 Ød1551120 00 Ød1551224 00 Ød155124 00 Ød155124 00 Ød155124 00 Ød155124 00 Ød155124 00 Ød155124 00 Ød155134 00 Ød155134 00 Ød155134 00 Ød155134 00 Ød155138 00	CC FF44]=00000000 00 1ue ASCI Connent 000000 801D7B \$000000 801D7B \$000000 800107B \$000000 800107B \$000000 800107B \$0000000 \$0000000 \$00000000 \$0000000	2.LoadLibraryA 2.CloseHandle 2.ExiThread 2.CreateMutexA 2.WaitForSingleObject 2.LocalFree 2.LocalFree 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateThread 2.CreateFileA 2.GetTempPathA 2.GetTempPathA 2.GetTenpA 2.IstrlenA 2.IstrlenA 2.IstrcatA	
Address Ua Stack [0012 EBP=0012FFC Address Ua 001550EC 00 001550FC 00 001550F7 CC 001550F8 CC 00155104 CC 00155104 CC 001551104 CC 00155112 CC 00155124 CC 00155130 CC 00155130 CC 00155134 CC 00155138 CC 00155138 CC 00155138 CC 00155138 CC 00155138 CC <tr< td=""><td>CC FF44]=000000000 0000000 0000000 0000000 801D7B %C: kernel3 809BE7 %C: kernel3 8000F8 %C: kernel3 8029DF %C: kernel3 802446 FSC: kernel3 809241 -UC: kernel3 809244 FSC: kernel3 809244 FSC: kernel3 809244 SUCFA %C: kernel3 809265 %C: kernel3 801A28 %C: %C: %C: %S02568 %C: %AC: %AC: %AC: %AC: %AC: %AC:</td><td>2.LoadLibraryA 2.CloseHandle 2.CloseHandle 2.ExitThread 2.WaitForSingleObject 2.WaitForSingleObject 2.JocalAlloc 2.LocalFree 2.LocalFree 2.DeleteFileA 2.SetFileAtributesA 2.GetTickCount 2.CreateThread 2.CreateThread 2.CreateFileA 2.WriteFile 2.GetTempPathA 2.CreateProcessA 2.IstrlenA 2.IstrlenA 2.IstrcatA</td><td></td></tr<>	CC FF44]=000000000 0000000 0000000 0000000 801D7B %C: kernel3 809BE7 %C: kernel3 8000F8 %C: kernel3 8029DF %C: kernel3 802446 FSC: kernel3 809241 -UC: kernel3 809244 FSC: kernel3 809244 FSC: kernel3 809244 SUCFA %C: kernel3 809265 %C: kernel3 801A28 %C: %C: %C: %S02568 %C: %AC: %AC: %AC: %AC: %AC: %AC:	2.LoadLibraryA 2.CloseHandle 2.CloseHandle 2.ExitThread 2.WaitForSingleObject 2.WaitForSingleObject 2.JocalAlloc 2.LocalFree 2.LocalFree 2.DeleteFileA 2.SetFileAtributesA 2.GetTickCount 2.CreateThread 2.CreateThread 2.CreateFileA 2.WriteFile 2.GetTempPathA 2.CreateProcessA 2.IstrlenA 2.IstrlenA 2.IstrcatA	
Address Ua Stack [0012 EBP=0012FFC Bd01550EC 000 001550F2 000 001550F4 7C 001550F7 7C 001550F6 7C 00155100 7C 00155100 7C 00155104 7C 00155104 7C 00155104 7C 00155104 7C 00155110 7C 00155110 7C 00155112 7C 00155120 7C 00155120 7C 00155130 7C 00155134 7C 00155134 7C 00155134 7C 00155134 7C 7E 00155134 7C 7E	CC FF44]=000000000 0000000 0000000 0000000 801D7B %C: kernel3 809BE7 %C: kernel3 8000F8 %C: kernel3 8029DF %C: kernel3 802446 FSC: kernel3 809241 -UC: kernel3 809244 FSC: kernel3 809244 FSC: kernel3 809244 SUCFA %C: kernel3 809265 %C: kernel3 801A28 %C: %C: %C: %S02568 %C: %AC: %AC: %AC: %AC: %AC: %AC:	2.LoadLibraryA 2.CloseHandle 2.CloseHandle 2.ExitThread 2.WaitForSingleObject 2.WaitForSingleObject 2.JocalAlloc 2.LocalFree 2.LocalFree 2.DeleteFileA 2.SetFileAtributesA 2.GetTickCount 2.CreateThread 2.CreateThread 2.CreateFileA 2.WriteFile 2.GetTempPathA 2.CreateProcessA 2.IstrlenA 2.IstrlenA 2.IstrcatA	
Address Ua Ød1550EC 0012FFC Ød1550EC 000 Ød1550FF 00 Ød1550FF 00 Ød1550FF 00 Ød1550FF 00 Ød1550FF 00 Ød155108 00 Ød155108 00 Ød155108 00 Ød155108 00 Ød155118 00 Ød155120 00 Ød155120 00 Ød155128 00 Ød155128 00 Ød155128 00 Ød155128 00 Ød155128 00 Ød155138 00 Ød155138 00 Ød155138 00 Ød155138 10 Ød155138 10 Ød155138 10 Ød155138 10 Ød155140 10 Ød155144 10	CC FF44]=00000000 0 1ue ASCI Comment 000000 801D7B C 801D7B C 800008 C 800107B C 801D7B C 80206F O.C.: kernel3 802530 B.C.: kernel3 802446 F.S.: kernel3 809A2D -U.C.: kernel3 809A2D -U.C.: kernel3 801A28 <-til kernel3 801A28 <-til kernel3 801A28 <-til kernel3 8012FA j'ali kernel3 802366 k#C: kernel3 802366 k#C: kernel3 804261 BMA3: <thkernel3< th=""> 834DE1<td>2.LoadLibraryA 2.CloseHandle 2.ExitThread 2.ExitThread 2.WaitForSingleObject 2.JocalFree 2.LocalAlloc 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateThread 2.CreateFileA 2.GetTempPathA 2.GetTempPathA 2.GetTenpPathA 2.IstrcatA MessageBoxA WSAStartup Socket</td><td></td></thkernel3<>	2.LoadLibraryA 2.CloseHandle 2.ExitThread 2.ExitThread 2.WaitForSingleObject 2.JocalFree 2.LocalAlloc 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateThread 2.CreateThread 2.CreateFileA 2.GetTempPathA 2.GetTempPathA 2.GetTenpPathA 2.IstrcatA MessageBoxA WSAStartup Socket	
Address Ua Stack [0012] EBP=0012FFC 001550EC 001550EC 000 001550F4 7C 001550F5 7C 001550F6 7C 001550F7 7C 001550F8 7C 00155104 7C 001551104 7C 001551104 7C 00155112 7C 00155124 7C 00155124 7C 00155138 7C 00155134 7C 00155124 7C 00155134 7C 00155144 71	CC FF44]=0000000 0 1ue ASCI Connent 000000 801D7B **C: kernel3 809BE7 *C: **C: kernel3 8059BE7 *C: **C: kernel3 8029DF *C: **C: kernel3 802446 FSC: \$80246 FSC: \$809925 4000: \$809942 -U00: \$809942 -U00: \$809944 JGC: \$80994 •U00: \$801428 <-C:	2.LoadLibraryA 2.CloseHandle 2.Exithread 2.Exithread 2.CreateMutexA 2.WaitForSingleObject 2.Sleep 2.LocalAlloc 2.LocalFree 2.DeleteFileA 2.SetFileAttributesA 2.GetTickCount 2.CreateFileA 2.CreateFileA 2.WriteFile 2.GetTempPathA 2.CreateFrocessA 2.lstrlenA 2.lstrcatA MessageBoxA WSAStartup socket send	

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 ZXZ1LmZsZXVyeQ== 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.