

IT Security Conference for the Next Generation

DRIVE-BY DOWNLOAD ATTACKS: EFFECTS AND DETECTION METHODS

Aikaterinaki Niki

(nik_aik@yahoo.gr)

MSc Information Security, 2008-2009
Royal Holloway University of London

Supervisor: Dr Jason Crampton

(jason.crampton@rhul.ac.uk)

Author:

Aikaterinaki Niki

Date of birth: 25/07/1985

51 Tudor Court

Princes Riverside

London

SE16 5RH

+44(0)7532069516

1 INTRODUCTION	4
2 BACKGROUND.....	5
3 ANATOMY OF A DRIVE-BY DOWNLOAD.....	7
3.1 PHASES OF A DRIVE-BY DOWNLOAD ATTACK	7
3.2 INJECTING MALICIOUS CODE INTO A WEB PAGE.....	8
3.2.1 INJECTION MECHANISMS	8
3.2.1.1 WEB SERVER SECURITY	8
3.2.1.2 USER CONTRIBUTED CONTENT.....	9
3.2.1.3 ADVERTISING.....	9
3.2.1.4 THIRD-PARTY WIDGETS	10
3.2.2 AUTOMATED TOOLS	10
3.3 EXPLOITATION STRATEGIES.....	10
3.3.1 EXPLOIT SOFTWARE	11
3.3.1.1 BROWSER VULNERABILITIES	11
3.3.1.2 PLUG-IN VULNERABILITIES	12
3.3.1.3 FILE FORMAT VULNERABILITIES	13
3.3.2 TRICK THE USER.....	14
4 IMPACT OF DRIVE-BY DOWNLOADS ON THE USER'S SYSTEM.....	16
4.1 SYSTEM CHANGES	16
4.1.1 REGISTRY CHANGES.....	16
4.1.2 RUNNING PROCESSES AND FILE ACTIVITY	16
4.1.3 NETWORK TRAFFIC	16
4.2 IMPLEMENTING DRIVE-BY DOWNLOADS.....	17
4.2.1 FIRST DRIVE-BY DOWNLOAD ATTEMPT.....	17
4.2.1.1 ANALYSIS OF RUNNING PROCESSES AND FILE SYSTEM ACTIVITY	19
4.2.1.2 ANALYSIS OF REGISTRY CHANGES	23
4.2.1.3 ANALYSIS OF NETWORK TRAFFIC	24
4.2.1.4 VISIBLE EFFECTS OF THE DRIVE-BY DOWNLOAD	25
4.2.2 SECOND DRIVE-BY DOWNLOAD ATTEMPT.....	26
4.2.2.1 ANALYSIS OF RUNNING PROCESSES AND FILE SYSTEM ACTIVITY	27
4.2.2.2 ANALYSIS OF REGISTRY CHANGES	29
4.2.2.3 ANALYSIS OF NETWORK TRAFFIC	30
4.2.2.4 VISIBLE EFFECTS OF THE DRIVE-BY DOWNLOAD	31
4.3 CONCLUSION	31
5 DETECTION OF MALICIOUS WEB PAGES.....	32
5.1 ANTI-DETECTION MECHANISMS USED BY ATTACKERS	32
5.2 MALICIOUS URL DETECTION MECHANISMS.....	33
5.2.1 COMBINING STATIC HEURISTICS AND CLIENT HONEYPOTS.....	33
5.2.1.1 FIRST STAGE – USING STATIC HEURISTICS.....	33
5.2.1.2 SECOND STAGE – USING CLIENT HONEYPOTS.....	35
5.2.1.3 EXAMPLE OF THE DETECTION MECHANISM	35
5.2.2 ANALYZING DNS AND WEB SERVER RELATIONSHIPS	36
5.3 CONCLUSION	37
6 CONCLUSION.....	38

EXECUTIVE SUMMARY

The aim of this study is to present and analyze the problem of drive-by download attacks. Drive-by downloads have become one of the most common ways to infect a large group of unsuspected users. Attackers take full advantage of the functionality of the Internet and its dominance in various transactions of everyday life and spread malware by exploiting vulnerable systems for financial gain.

These client-side attacks launched when visiting malicious web sites have become the centre of attention for security researchers and anti-virus companies. The main objective of this paper, therefore, is to understand and describe the conditions under which these attacks happen, the weaknesses that create the problem and the effects they cause on the victims' machines, in order to get deeper to the changes that occur to the system after the infection. An actual infection from malware caused by a drive-by download intends to support the last task. The study, furthermore, spreads on the main techniques known today as a detection mechanism for malicious web pages and the challenges associated with them.

1 INTRODUCTION

A few days ago about 55,000 web sites were compromised by cybercriminals with malicious iFrames targeting to infect with exploit scripts every user that visits them [1]. The installation of a number of malware in each user's machine comes to be the result of these attacks, the so called *drive-by download attacks*. A similar large scale attack became known last May, when more than half a million web pages distributed a variant of the Zlob Trojan to unsuspected users through malicious JavaScripts [2].

Drive-by downloads are caused when a user visits a website that exploits browser vulnerabilities and launches the automatic download and installation of malware without the knowledge or permission of the user. The variety of client applications installed in web servers and personal computers give enough space to attackers to find vulnerable systems to attack. According to [3], almost 80% of web users are using unpatched versions of Adobe Flash and Acrobat Reader, popular web browser plug-ins.

An eruption of drive-by download attacks is observed lately. Sophos, in the Security threat report of 2008, claims that a new infected page is discovered every 14 seconds [4]. At a level this increase is resulting from the change of the web. Internet has become an essential part of our everyday life. Banking transactions, shopping, news, social networking are just a few of the possibilities and web browsers are the media to perform these activities. On the other hand, the great importance of the web has lead cybercriminals towards it, using it as the tool to perform their illegal actions. Over the last two years, a continuously increasing number of legitimate sites are being attacked, making the death of trusted sites a fact [5]. UK government web sites, companies in the malware sector like F-Secure and Kaspersky are among the victims of these attacks [6, 7, 8].

Along with the Internet evolution, the design and use of malware has changed dramatically. Today malware use more stealth mechanisms and polymorphism to avoid being detected. Crashing the system is not their main target anymore. Most of the web malware are intended to either steal the user's personal information, such as credit card details and passwords, or cause the victim machine to join a botnet [9]. Compromised computers belonging to a botnet are under the control of the attacker. It is considered a major security problem as spam and Denial of Service attacks are initiated by them.

Malware drive-by downloads are a new challenge, as their prevalence seems to be increasing more and more in malware distribution attacks. They are a serious threat for the safety of the Internet, so understanding the details of these attacks is of major importance.

The remainder of this paper is structured as follows. Section 2 provides a background based on some statistics on drive-by download attacks. Section 3 gives a more detailed analysis on the steps followed when a drive-by download occurs and the techniques used in both the server and the client to inject malicious code. The changes caused in the user's system by implementing drive-by download infections are analysed in Section 4. Section 5 includes a discussion on some of the detection methods used to identify malicious URLs and, finally, Section 6 concludes.

2 BACKGROUND

A study carried out by Google during the year 2007 [8, Ch.4] reveals that approximately 1.3% of the incoming search queries to Google's search engine returns at least one malicious URL, which shows that a significant portion of users might be exposed in drive-by download attacks. The graph below, taken from that study, shows that during the period of January 2007 to October 2007 the percentage of queries returning a harmful URL increased more and more every month.

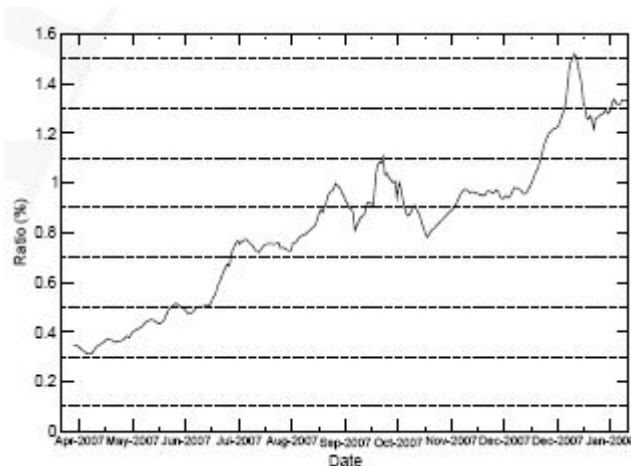


Figure 1: Search results containing a harmful URL

The same study sorted out in categories a number of malicious URLs according to their function and subject. Figure 2 shows this categorization. The outcome of this is that web sites from all categories are possible candidates for launching a drive-by download attack. Even if users are careful in visiting legitimate sites and avoid the ones that theoretically are more “dangerous”, adult sites for example, they are always at risk in getting infected.

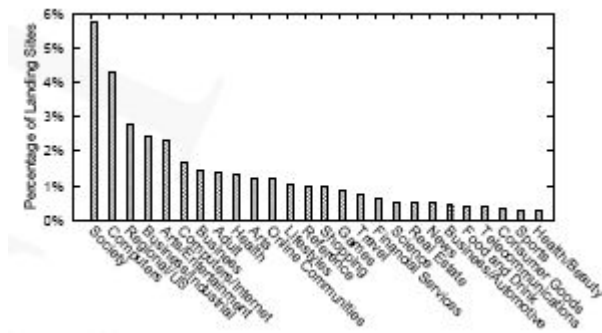


Figure 2: Malicious URLs per category

An indication on the volume of drive-by download attacks launched by compromised web sites due to iframes and exploit code is provided by ScanSafe’s results on web-based malware blocks during 2008. According to the Annual Threat Report “in 2008, an overall average of 57% of all ScanSafe web malware blocks were due to iframes and exploit code resulting from website compromise, compared to 21% of blocks in 2007” [10, Ch.4].

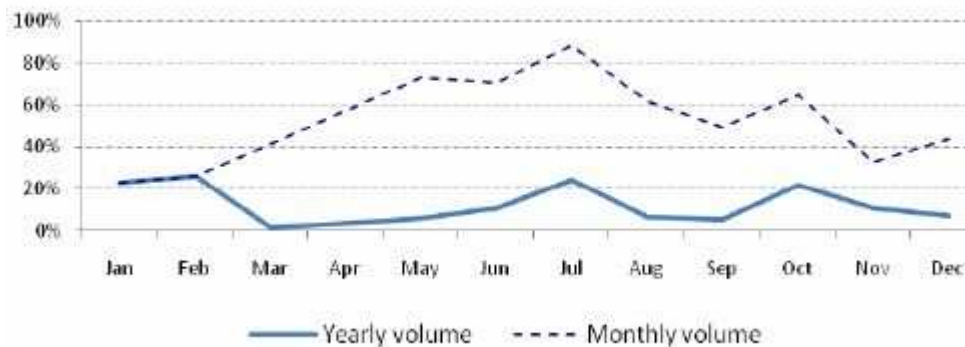


Figure 3: ScanSafe blocks indicative of website compromise (per month and per year volume)

3 ANATOMY OF A DRIVE-BY DOWNLOAD

3.1 PHASES OF A DRIVE-BY DOWNLOAD ATTACK

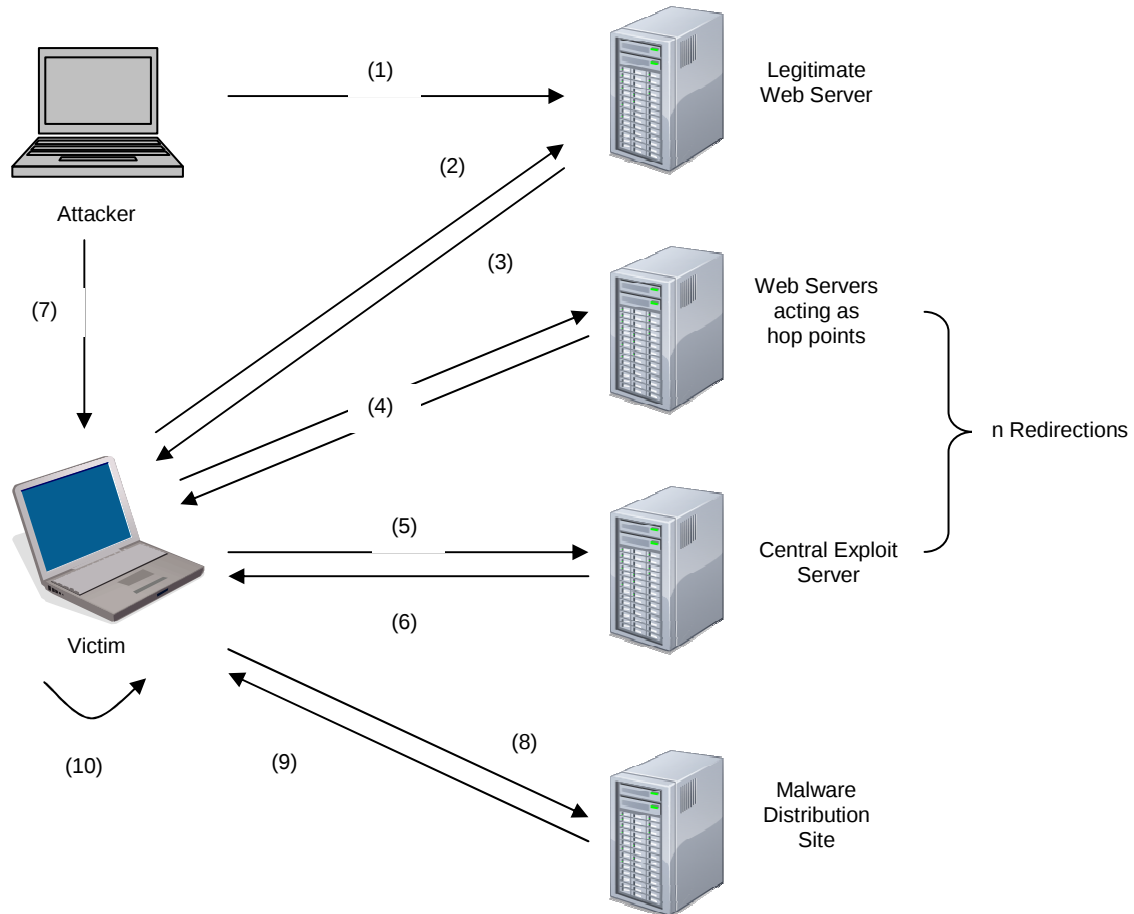


Figure 4: The typical steps of a drive-by download attack.

When a drive-by download occurs, the following steps usually take place as shown in Figure 4.

- 1) The attacker compromises a legitimate web server and inserts a script in a web application.
- 2) The victim visits the web site that was compromised.
- 3) The web server sends along with the requested page the script the attacker injected. This script executed is either the exploit script or a script that imports it from a central exploit server. This import is either a direct inclusion of the

resources from the remote server or a number of redirections the browser is instructed to follow.

- 4) A redirection starts from one web server to the other that actually play the part of hop points.
- 5) After following a number n of redirections the victim reaches the central exploit server.
- 6) The server sends the exploit script.
- 7) The attacker gains control over the victim's system, after exploiting the vulnerability that was targeted.
- 8) The exploit instructs the browser to visit the malware distribution site. This is, actually, when the drive-by download starts.
- 9) Malware executables are downloaded.
- 10) The victim's computer automatically installs and executes the malicious code.

3.2 INJECTING MALICIOUS CODE INTO A WEB PAGE

The starting point for the adversary is to control a legitimate web server or a web application to be able to insert a script that will execute malicious code when run. The ease of setting up and deploying web sites and the number of different applications used to operate these sites make their task even easier. A variety of automated tools that do the whole job instead of them come to scale up the attacks. The things attackers most commonly take advantage of, according to a research from Google on the current state of web-based malware [11, Ch. 4], are a non-secure web server, user contributed content, advertisements and third-party widgets.

3.2.1 INJECTION MECHANISMS

3.2.1.1 WEB SERVER SECURITY

Keeping the web server secure and ensuring the applications used for the web site's existence are updated with the most recent security patches is an important security practice that administrators tend to pass by. Software running on the web server, if outdated, gives the space to the attacker to exploit well known vulnerabilities and thus, gain control over the server and modify the web site's content.

For years the Internet Information Services Web Server was the major target for the attackers, because it was prone to multiple vulnerabilities that allowed them to have access to the database and other sensitive data or configuration files [12, 13].

Malicious content can be inserted as an entry in databases using SQL injection techniques [14] or by taking advantage of vulnerabilities in scripting applications like PHP, ASP, Perl, Python etc. These vulnerabilities are caused by programming code errors and if exploited by an attacker, he would be able to have direct access to the

operating system and make calls to functions associated to system processes and resources.

A known vulnerability in PHP is an SQL injection vulnerability in Invision Power Board that allows an attacker to manipulate database records [15]. A second one, also in PHP, has to do with the XML RPC Remote Commands Execution. The lack of input validation gives the possibility to adversaries to remotely insert arbitrary code that will be executed by the `xmlrpc.php` script [16].

3.2.1.2 USER CONTRIBUTED CONTENT

Many web sites as for example forums, blogs, bulletin boards are web applications that allow their users to contribute their own content in the form of comments, reviews etc. The lack of input validation and checking allows every user to enter arbitrary HTML anywhere in the page user input is requested. Attackers can insert code including “`iframe`” or “`script`” tags and expose every user seeing the post to the exploit script containing malicious code.

The cross site scripting attack (XSS) is relevant to this weakness [17]. The last years, it is thought to be one of the most dangerous and preferred method for attackers, because it is easy to implement. Many security incidents have been recorded, in which an XSS vulnerability causes malware to spread. In 2006, an XSS worm exploiting a high level application layer appeared in MySpace [18], while in 2007, another worm affected more than 650.000 users of the social networking site Orkut [19].

3.2.1.3 ADVERTISING

Advertising is usually achieved by large advertising companies that provide a fixed piece of code to be inserted in web pages. This piece of code, therefore, is not directly controlled by the administrator of the web site that displays the advertisement, which means that the company should be trusted for not providing malicious content. This condition by itself is a security weakness that becomes worse by the increasing use of ad syndication. Ad syndication “allows an advertiser to sell advertising space to other advertising companies that in turn can yet again syndicate their content to other parties” [20, Ch. 5.2]. Trust is difficult to exist under these conditions and absolute control of the content cannot be achieved, as “a single visit to a single web page can result in content being delivered from multiple domains across the globe” [21].

Adversaries find this as an attractive way to insert malicious content to popular web sites that display advertisements without making any effort to compromise the web server and search for vulnerabilities. It is a technique in which malware can be spread widely if the target are advertisements in popular web sites. News and media advertising networks and online gaming websites quite often become the target of these attacks. Malicious advertisements were displayed in eWeek home page by

Google's DoubleClick, a platform for displaying advertising, in February [22] and another incident was noticed in May, in the Digital Spy, a British web site with entertainment and media news [23].

3.2.1.4 THIRD-PARTY WIDGETS

Third-party widgets are scripts provided by third parties that are commonly used to provide extra functionality to a web site. Most of them are accessed through a link contained in an external javascript or iframe. The link leads to the web site that hosts the widget. What can go wrong in this case is the possibility an adversary changes the code of the widget without the knowledge of the web master and serves malicious content. This is the reason a third party providing widgets should be trusted, which, similarly to advertising, is not feasible.

3.2.2 AUTOMATED TOOLS

A variety of exploit toolkits that perform vulnerability scanning and automate the attacks is widely available in the Internet. Many of them have a graphical user interface and simple design and don't require deep technical knowledge to run attacks. According to an article in the BBC News [24], in 2007, when the article was written, there were over 68000 downloadable hacking tools. Among the most popular are:

- MPack: It is a tool using cross-site scripting to hide iframes in compromised sites that redirect the user who visits it to the attacker's malicious web site. The tool based on the browser type and the operating system installs the most suitable exploit script to the user's system and if it succeeds it sends the malicious file [25].
- El Fiesta: This is a tool targeting exclusively Adobe PDF formats. PDF exploits have risen sharply during the last year causing a number of security patches to be released.
- Neosploit: With functionality similar to the MPack, it is said to be responsible for injecting malicious code to more than 80000 legitimate web sites. In October 2008, it came into sight after it was revealed that more than 200000 login credentials from organizations in 86 countries were stored in a server hosting the toolkit [26].

3.3 EXPLOITATION STRATEGIES

The exploit of vulnerabilities in the user's system is the next important step for the adversary in order to gain control over his computer. Two are the main ways used by the attacker to pass the exploit script into the user's system, exploiting software or tricking the user [11, Ch. 5].

Another method that was widely used in the past was by remotely exploiting vulnerable network services, via worms for example. This is not successful lately, because of the use of protection mechanisms to a user's personal computer. Firewalls and Network Address Translators, by filtering the incoming connections, prevent the attacker to connect remotely and exploit services running on the user's system.

3.3.1 EXPLOIT SOFTWARE

Exploiting software on a user's computer may target either the browser or external programs launched automatically including browser plug-ins or common file formats. Users' ignorance or negligence leaves their computers running insecure applications. According to some statistics gathered from the Secunia PSI [27], about 42% of computers have more than 11 unpatched applications. These unpatched applications carry vulnerabilities that allow the adversary to successfully exploit them and start a drive-by download attack. SANS Institute has listed the top 20 most critical vulnerabilities that are massively exploited in the wild [28].

3.3.1.1 BROWSER VULNERABILITIES

Web browsers are maybe the main target at the adversary's effort to exploit software on the user's computer. And the most serious threat is the remote execution of arbitrary code by exploiting in most of the cases a buffer overflow vulnerability [29, Page 9]. Cross-site scripting comes as the second of the most dangerous threats. Figure 5 shows the total vulnerabilities of the most used web browsers observed from 2004 until today [30]. The graph includes vulnerabilities listed from Secunia.

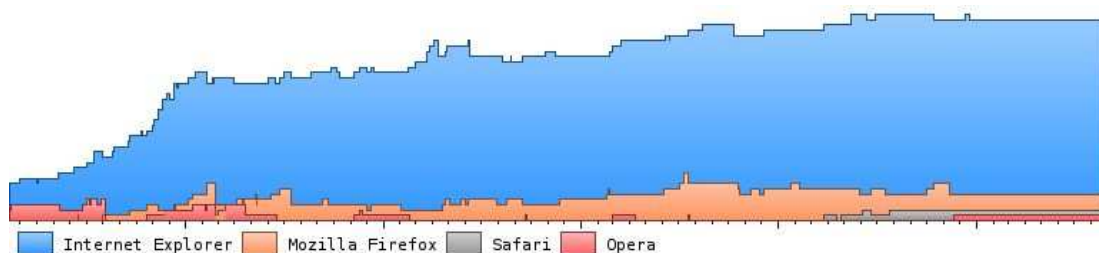


Figure 5: Web browser vulnerabilities, 2004-2009

JavaScript, VBScript and Ajax are technologies integrated with web browsers, as they support dynamic client-side content. The disadvantage of these technologies, that attackers take full advantage of, is that they share the same address space with the browser and they are placed on the browser's heap. The attacker makes use of that and with the use of a piece of scripting code (the exploit script) he loads the shellcode into memory and executes it. Heap spraying is a technique used by

attackers to make browser exploitation based on JavaScript more reliable [31, 32]. The browser heap is filled with multiple copies of the shellcode and when one of these copies is placed at a desired, by the attacker, location in memory he redirects the program control flow to the shellcode.

ActiveX is another technology used by web browsers and misused by attackers. CNET Glossary defines it as “a broad set of technologies and services based on Microsoft’s Component Object Model (COM)” [33]. ActiveX controls are small programs implementing plug-in support and allowing for interactive content, such as spreadsheets. This means that a user can view the spreadsheet in his browser without having to open it with an external application. An ActiveX control’s code can easily be changed by attackers with malicious code, in a way similar to the scripts written in JavaScript. An example is the MPEG2TuneRequest ActiveX Control. Attackers found a remote code execution vulnerability in this control and exploited it to inject malicious code to the user’s system [34].

3.3.1.2 PLUG-IN VULNERABILITIES

Plug-ins are programs developed from third parties to extend the browser’s functionality. Most of them are written in C language which is not considered a safe language and, thus, the plug-ins are susceptible to vulnerabilities like buffer overflows, memory corruption issues and pointer overwrites [35, Ch.1].

Many of them are executed in the context of the browser, meaning that they get high privileges on the browser and the operating system and they, furthermore, share the address space with the browser. All these combined give a nice target for the attacker to exploit [36, Ch.2].

Every browser supports many different plug-ins to enable media playback, software updates, Java applets or Microsoft Office and PDF documents view. Common targets appear to be the Adobe Flash Player, Adobe Reader, Apple’s QuickTime, Real Player, Microsoft’s Media Player, plug-ins a large number of users install in their browsers. Table 1, taken from [37], shows the most widely used plug-ins, according to the percent of users that have installed it in their computer, and the browsers that support it.

PLUG-IN	VENDOR	USERS(%)	BROWSER
Flash Player	Adobe	98.8	All
Java	Sun	84.0	All
Media Player	Microsoft	82.2	IE only
QuickTime Player	Apple	66.8	All
Shockwave Player	Adobe	55.6	All

Real Player	Real Networks	47.1	All
-------------	---------------	------	-----

Table 1: Usage percentage of popular browser plug-ins

CVE	VULNERABILITY
CVE-2009-0658	Adobe Reader 9
CVE-2009-2011	DX Studio Player Firefox Plug-in Command Injection
CVE-2007-6244	Adobe Flash Player ActiveX Control Universal Cross-Site Scripting Application
CVE-2006-3311	Adobe Flash Player Code Execution (Action Script)
CVE-2005-2340	Apple QuickTime Malformed GIF Heap Overflow
CAN-2004-1029	Sun Java Plug-in Arbitrary Package Access Vulnerability

Table 2: Examples of known vulnerabilities in browser plug-ins

Table 2 displays a small sample of well-known vulnerabilities found in browser plug-ins. The CVE (Common vulnerabilities and Exposures) [38] identifier of the vulnerability is presented and a short description of the vulnerability.

3.3.1.3 FILE FORMAT VULNERABILITIES

Word, Excel or PDF documents are, also, used as a way to inject malicious code to the victim's computer. Vulnerabilities found in file formats make it easy for the attacker to attach malware to these documents. The malicious files are distributed either as email attachments or by visiting compromised web sites. When the associated editing program opens the file the exploit is carried out that will finally lead to the drive-by download.

CVE	VULNERABILITY
CVE-2006-2492	Word Malformed Object Pointer Vulnerability
CVE-2006-0022	PowerPoint Remote Code Execution Using a Malformed Record Vulnerability
CVE-2006-6456	Word Malformed Data Structure Vulnerability
CVE-2007-0671	Excel Malformed Record Vulnerability

CVE-2007-1747	Drawing Object Vulnerability
CVE-2008-0081	Macro Validation Vulnerability
CVE-2008-2244	Word Record Parsing Vulnerability

Table 3: Vulnerabilities in Microsoft Office suites file formats [39, p.53]

PDF exploits were one of the main highlights of web security in 2008 [40]. Attacks targeting PDF file formats rise in exponential rhythms. Figure 6 shows the attacks carried in Adobe Reader each month in 2008 [39, p.57].

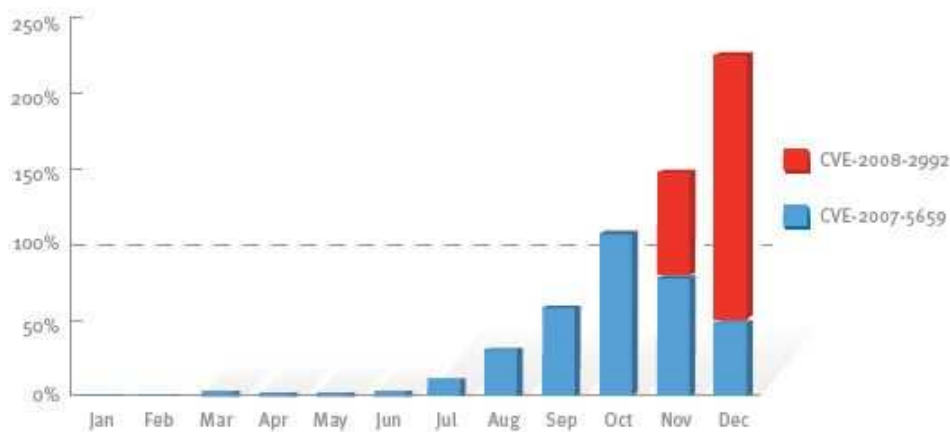


Figure 6: Adobe Reader exploits by month, 2008

3.3.2 TRICK THE USER

The method of tricking the user is used when the attacker finds no exploitable vulnerability. It is a form of social engineering when trying to lure the user to follow links that either lead to a malicious site or to a page that asks the user to download and run certain programs that are, actually, malware executables that end up installed in the user's computer. These programs appear to be a special codec or plug-in needed to display the content or even an anti-virus program that performs a fake scan on the user's machine and displays virus alerts.



Figure 7: Example of a web page requesting an ActiveX plug-in to play a video file.

A massive threat lately is the rogue anti-virus programs. Advertised in well-designed web sites that appear to be professional and legitimate, they target on the computer user's fear. Continuous virus alerts and warnings about the system having been infected with serious threats make the user install the malicious executable. Usually, he will be asked to buy the fake anti-virus to remove the threats and that way the attacker will be able to gain personal information and credit card details [39, p.92-99], [41].



Figure 8: Windows Protection Suite
Rogue Anti-virus Website

4 IMPACT OF DRIVE-BY DOWNLOADS ON THE USER'S SYSTEM

As discussed before, a drive-by download usually initiates a number of downloads and installations, after the successful exploitation of a vulnerability in the browser or one of its plug-ins. The executables are malware used for different purposes that cause changes to the system state and affect the user's machine depending on their type. The main changes are observed in the registry, the system's processes and network's activity.

4.1 SYSTEM CHANGES

4.1.1 REGISTRY CHANGES

The registry according to The Microsoft Computer Dictionary is “a central hierarchical database used in Microsoft Windows operating systems to store information that is necessary to configure the system for one or more users, applications and hardware devices” [42]. Whenever a malicious program is installed on a computer, it modifies some of the registry keys in order to gain some privileges on the system. These malicious modified and deleted registry entries can affect the computer's operations and its performance and can, therefore, cause serious damage.

4.1.2 RUNNING PROCESSES AND FILE ACTIVITY

The automatic execution of binaries increases at once the number of running processes. In some cases it is increased in such a length that the system's processor cannot handle the overhead and “crashes”. The processes started by the execution of the malware are often associated with multiple file creations, modifications and deletions. A common strategy malware follow is the replacement of critical files of the operating system, like Dynamic Link Library files (.dll files). Malicious files masquerade as the original files aiming to inject themselves in running processes and change or manipulate their behavior and consequently the behavior of the linked programs.

4.1.3 NETWORK TRAFFIC

An increased network activity is the result of visiting a malicious web page. HTTP connections are the most prevalent due to the large number of downloads that are initiated automatically when the exploitation succeeds. It is also a common tactic for malware to scan for other vulnerable systems in the LAN of the infected host and start sending numerous TCP and UDP packets in order to listen to open ports.

4.2 IMPLEMENTING DRIVE-BY DOWNLOADS

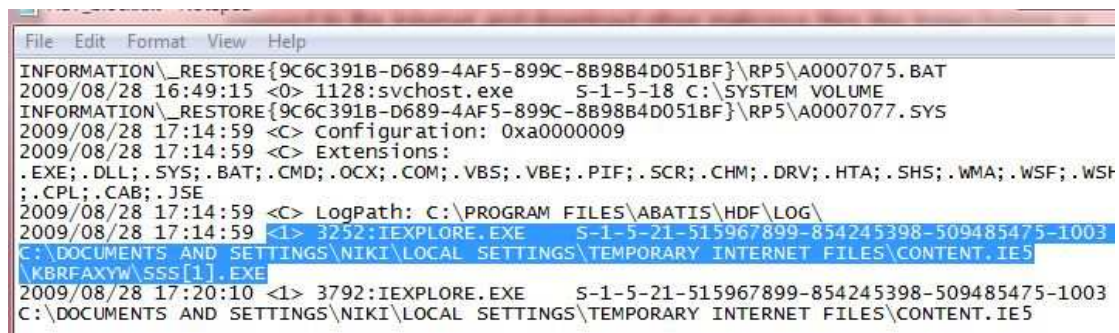
In order to test the effects of drive-by downloads on a system, a virtual machine was set up using the VMware Workstation v6.5 to isolate the host system from the damaging -- that malware may cause. The virtual machine runs under Windows XP Professional. A number of programs were installed to the guest operating system, necessary for the observation of the changes.

- *HDF* is a hard disk firewall that prevents malware infection. When activated it blocks unwanted software from storing to the computer [43].
- *Process Monitor* is a monitoring tool that shows real-time file system, registry and process/thread activity [44].
- *Wireshark* is a network protocol analyzer that enables live capture of network traffic and offline analysis [45].

Malicious URLs, already detected for starting drive-by downloads, were chosen from the “Malware Domain List” [47] and the “Malware URL” [48] websites that record any domain or IP address responsible for spreading malicious executables.

4.2.1 FIRST DRIVE-BY DOWNLOAD ATTEMPT

The first attempt to start a drive-by download was by visiting the URL <http://219.148.34.10/sss.exe>. This domain is registered at the site as known to spread a Downloader. A Downloader is a malware whose behaviour is to connect to the Internet and download other malicious files like trojan horses or adware. The first try was with the HDF firewall activated to check if the link is active and there are blocked entries in the log file. The result can be seen in Figure 9 below, where it is clear that the file sss[1].exe was blocked as indicated by the value <1> in the third field.

A screenshot of a text-based log window titled 'HDF Logfile'. The window has a menu bar with 'File', 'Edit', 'Format', 'View', and 'Help'. The log content shows several entries. The entry of interest is dated '2009/08/28 17:14:59' and shows a process '3252:IEXPLORE.EXE' with a configuration value '<1>' and a file path 'C:\DOCUMENTS AND SETTINGS\NIKI\LOCAL SETTINGS\TEMPORARY INTERNET FILES\CONTENT.IE5\KBRFAXYW\SSS[1].EXE'. The file path is highlighted in blue. Other entries show system information and file extensions.

```
File Edit Format View Help
INFORMATION\_RESTORE{9C6C391B-D689-4AF5-899C-8B98B4D051BF}\RP5\A0007075.BAT
2009/08/28 16:49:15 <0> 1128:svchost.exe S-1-5-18 C:\SYSTEM VOLUME
INFORMATION\_RESTORE{9C6C391B-D689-4AF5-899C-8B98B4D051BF}\RP5\A0007077.SYS
2009/08/28 17:14:59 <C> Configuration: 0xa0000009
2009/08/28 17:14:59 <C> Extensions:
.EXE;.DLL;.SYS;.BAT;.CMD;.OCX;.COM;.VBS;.VBE;.PIF;.SCR;.CHM;.DRV;.HTA;.SHS;.WMA;.WSF;.WSH
;.CPL;.CAB;.JSE
2009/08/28 17:14:59 <C> LogPath: C:\PROGRAM FILES\ABATIS\HDF\LOG\
2009/08/28 17:14:59 <1> 3252:IEXPLORE.EXE S-1-5-21-515967899-854245398-509485475-1003
C:\DOCUMENTS AND SETTINGS\NIKI\LOCAL SETTINGS\TEMPORARY INTERNET FILES\CONTENT.IE5\
KBRFAXYW\SSS[1].EXE
2009/08/28 17:20:10 <1> 3792:IEXPLORE.EXE S-1-5-21-515967899-854245398-509485475-1003
C:\DOCUMENTS AND SETTINGS\NIKI\LOCAL SETTINGS\TEMPORARY INTERNET FILES\CONTENT.IE5
```

Figure 9: Part of the HDF Logfile showing the blocked executable

Making sure the malicious URL works, I visited the page again but this time with the HDF de-activated. After allowing the executable to run, I opened the HDF log file to check for the entries. As expected the sss[1].exe written by the iexplore.exe process was allowed to get installed in the system.

Moreover, we notice that some seconds only later the executable file triggers the automatic download and installation of a number of other executable files and libraries. At the screenshot in Figure 10 we see that these files are named `bho.dll`, `miniup.exe`, `play.dll`, `ser.exe`. The download continues the same way by the executable `miniup.exe` this time which downloads the `minidll.dll`, `up.dll`, `kvon.dll` and `fvy.dll`. It is, also, noticeable that each file is being installed twice.

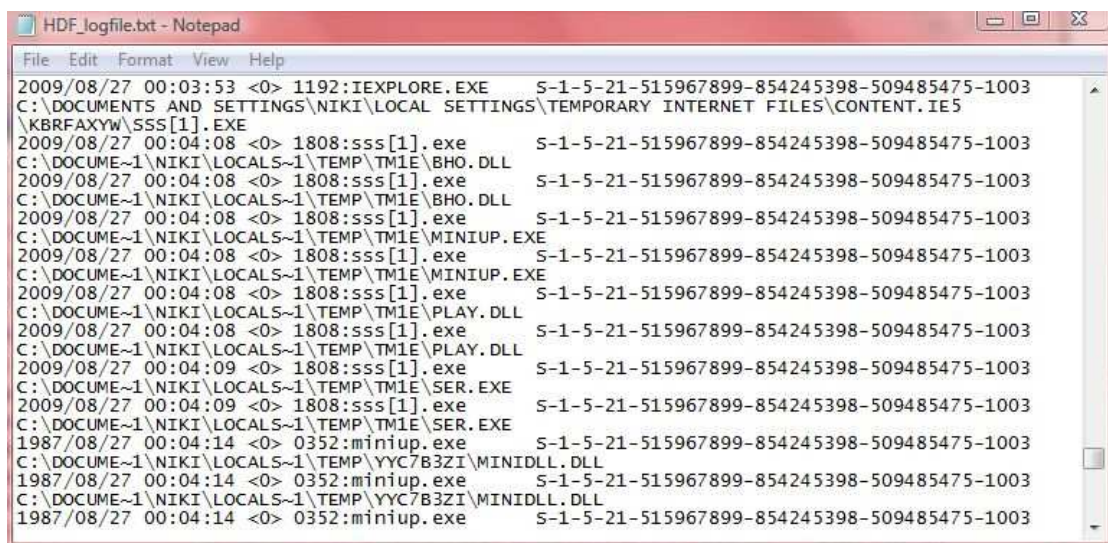
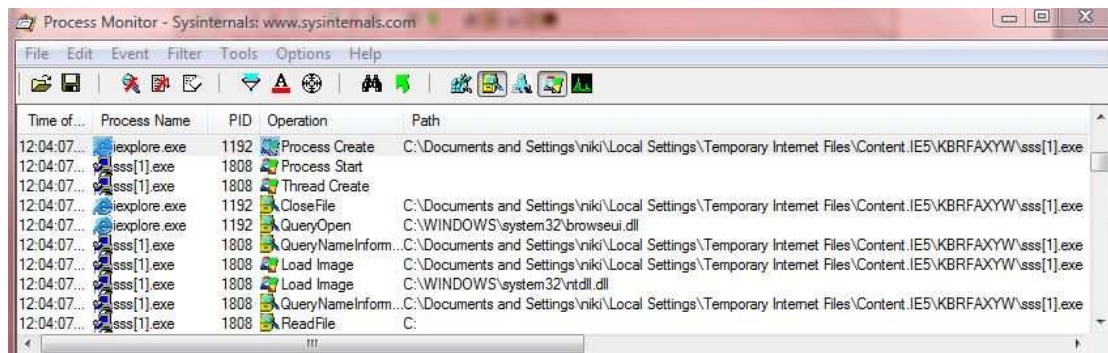


Figure 10: Part of the HDF Logfile showing a number of downloaded files

Two seconds later the sss[1].exe starts installing different files at the WINDOWS and *WINDOWS/System32* directories of the hard disk. These directories are of major importance in every version of Microsoft Windows, because they contain the core operating system files. The malicious files try to get themselves installed. After a while a file named sss[2].exe is downloaded in the Temporary Internet Files folder and a number of other downloads, triggered from the cml10.tmp process, follow.

Turning off the HDF protection about 5 minutes later blocked the download of further files. An entry in the log file shows that a file swflash[1].cab was tried to get downloaded after that point. This could contain files needed to install a Shockwave Flash ActiveX control or display Shockwave Flash media file.

4.2.1.1 ANALYSIS OF RUNNING PROCESSES AND FILE SYSTEM ACTIVITY

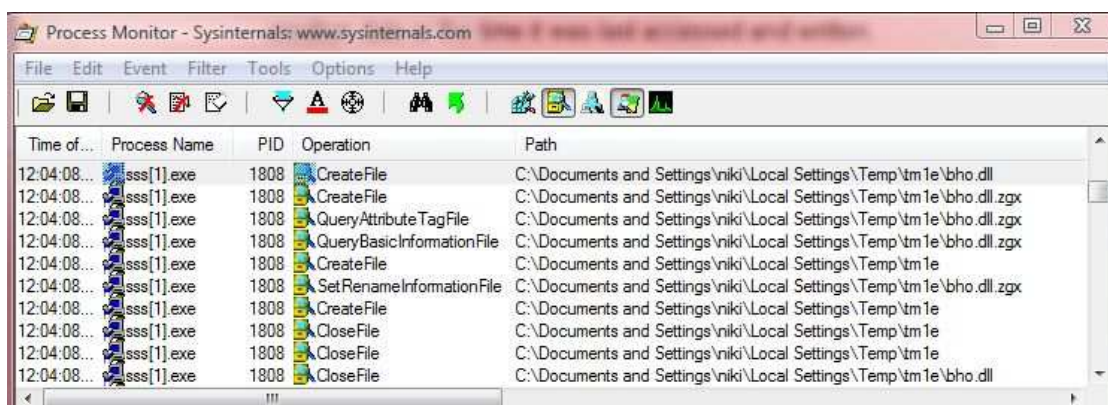


Time of...	Process Name	PID	Operation	Path
12:04:07...	iexplore.exe	1192	Process Create	C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW\sss[1].exe
12:04:07...	sss[1].exe	1808	Process Start	
12:04:07...	sss[1].exe	1808	Thread Create	
12:04:07...	iexplore.exe	1192	CloseFile	C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW\sss[1].exe
12:04:07...	iexplore.exe	1192	QueryOpen	C:\WINDOWS\system32\browseui.dll
12:04:07...	sss[1].exe	1808	QueryNameInform...	C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW\sss[1].exe
12:04:07...	sss[1].exe	1808	Load Image	C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW\sss[1].exe
12:04:07...	sss[1].exe	1808	Load Image	C:\WINDOWS\system32\ntdll.dll
12:04:07...	sss[1].exe	1808	QueryNameInform...	C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW\sss[1].exe
12:04:07...	sss[1].exe	1808	ReadFile	C:

Figure 11: Processes and file system activity sample

After opening the malicious web page, sss[1].exe is being downloaded and cached in the Temporary Internet Files folder. Below it is given a basic description of the processes that run from that point and on and the file system activity, after analyzing the results displayed by Process Monitor.

- The iexplore.exe process initiated by Internet Explorer executes the file sss[1].exe from the Temporary Internet Files folder, as seen in Figure 11.
- The sss[1].exe process starts and creates a file in the Prefetch folder of the Windows directory in order to ensure a quick start-up when the computer is turned on.
- During its execution it loads images of .dll files from the System32 directory and accesses operating system's files to get basic information about each file like the file's name and its length, file attributes, its creation date or the time it was last accessed and written.
- At the same time it downloads a number of other malicious files that installs in the newly created folder called tm1e in the directory *C:\Documents and Settings\'user\'Local Settings\Temp*. The files, as mentioned before, are the miniup.exe, bho.dll, play.dll and ser.exe. In Figure 12 a number of file operations concerning the installation of bho.dll is displayed.



Time of...	Process Name	PID	Operation	Path
12:04:08...	sss[1].exe	1808	CreateFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e\bho.dll
12:04:08...	sss[1].exe	1808	CreateFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e\bho.dll.zgx
12:04:08...	sss[1].exe	1808	QueryAttributeTagFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e\bho.dll.zgx
12:04:08...	sss[1].exe	1808	QueryBasicInformationFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e\bho.dll.zgx
12:04:08...	sss[1].exe	1808	CreateFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e
12:04:08...	sss[1].exe	1808	SetRenameInformationFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e\bho.dll.zgx
12:04:08...	sss[1].exe	1808	CreateFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e
12:04:08...	sss[1].exe	1808	CloseFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e
12:04:08...	sss[1].exe	1808	CloseFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e
12:04:08...	sss[1].exe	1808	CloseFile	C:\Documents and Settings\niki\Local Settings\Temp\tm1e\bho.dll

Figure 12: File operations during the installation of bho.dll

- Then it starts a regsvr32.exe process, a process of the Register Server. As shown in the “Details” section the command `C:\WINDOWS\system32\regsvr32.exe /u /s "C:\WINDOWS\system32\c671.dll"` was executed in order to unregister a server silently, without displaying any message boxes.
- Following that, the process miniup.exe was started by the sss[1].exe process which, also, makes sure to create a file in the Prefetch directory. The creation of the miniup.exe process is shown in Figure 13.
- Miniup.exe in its turn downloads the files miniDll.dll and up.dll that installs in a folder named yyc7b3zi in the *C:\Documents and Settings\user\Local Settings\Temp* directory and the files kvon.dll and fvy.dll that installs in the directory *C:\WINDOWS\Downloaded Program Files*.

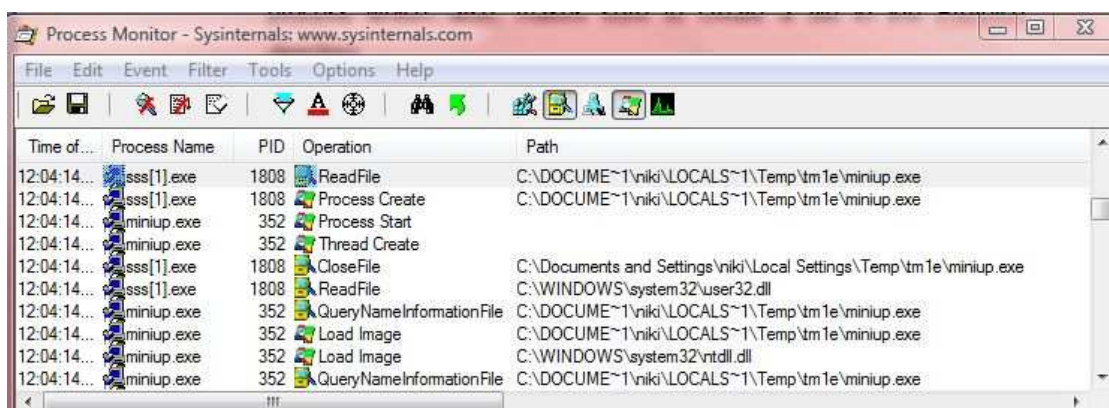


Figure 13: Creation of miniup.exe process

- Next, it sets the offset of the end of the file software.Log, contained in *system32\config* directory. This offset's value is changed a few times again. software.Log is a log file that keeps the changes made whenever new software is installed in the computer.
- Two separate rundll32.exe processes are started from the miniup.dll process to run the kvon.dll and fvy.dll files.
- The sss[1].exe process after using information from the files it downloaded before, it continues with the installation of malicious files whose names are random characters. Among them there are the files 5c1.dll, 67751.exe and c671.dll installed in the system32 directory. The last one according to the PrevX malware specification documents [49] is responsible for creating and registering a Browser Helper Object in Internet Explorer.
- A rundll32.exe process is created after that to run the c671.dll file.
- The rundll32.exe process started from the miniup.exe process creates a folder in the directory *C:\Documents and Settings\All Users\Application Data* where .dat files are installed. It, moreover, downloads files in the Temporary Internet Files directory.

- sss[1].exe having installed earlier the file 67751.exe starts the corresponding process. This file is known, according to PrevX [50], to modify Windows initialization and system settings, gain partial control of the system and of input and output, create, delete or execute other processes.
- Another 67751.exe process is started after a few seconds by services.exe that loads a number of process images and then exits.
- 5c1.dll is run with the creation of another rundll32.exe process by sss[1].exe.
- A sss[2].exe file is downloaded and stored in the Temporary Internet Files folder and a cml10.tmp file stored in *C:\Documents and Settings\user\Local Settings\Temp*.
- cml10.tmp process is created and started by the running rundll32.exe process that continues with the download of the file taobao0821[1].lz in Temporary Internet Files.
- More files are being created in the new directory t/ad in *C:\Documents and Settings\All Users\Application Data*. Among them .swf, .gif, .js file types are sorted out.
- A rundll32.exe process is started again and after a while creates and initiates a RunDll32.exe process instructed to execute a cml12.tpm file. RunDll32.exe is a malicious file that appears to be the same with the Windows executable, though the upper-case letters in its name state that it is not. It masquerades to the rundll32.exe, but in fact it is an executable designed to play advertisements and thus considered to belong in the group of adware.

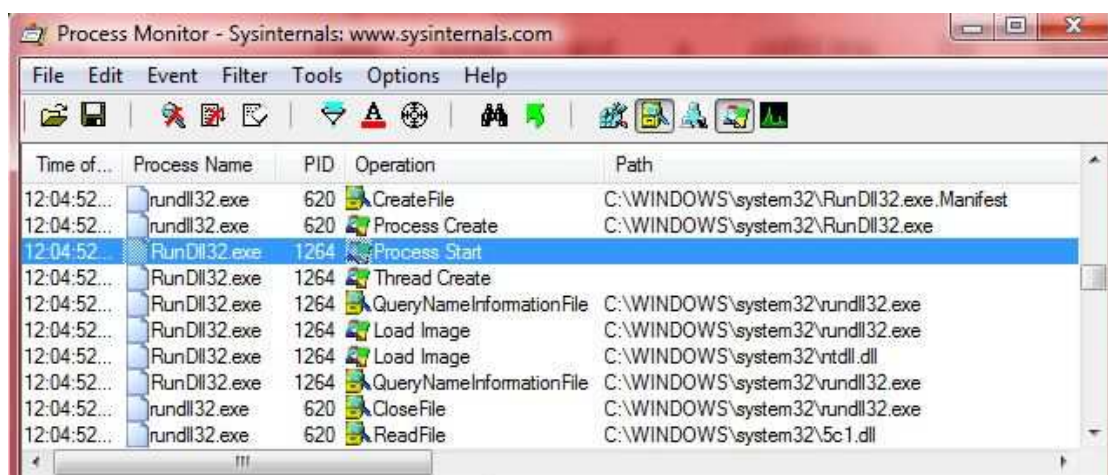


Figure 14: Creation or RunDll32.exe process

- After a while a process 22bd.exe is started by the cml10.tmp process.

A more detailed and organized view of the running processes is provided at Figures 15 and 16. The screenshots display parts of the process tree, which demonstrates all the processes in a hierarchy that reflects their parent-child relationship.

Process	Command
services.exe (736)	C:\WINDOWS\system32\services.exe
vmacthlp.exe (904)	"C:\Program Files\VMware\VMware Tools\vmacthlp.exe"
svchost.exe (948)	C:\WINDOWS\system32\svchost -k DcomLaunch
svchost.exe (1036)	C:\WINDOWS\system32\svchost -k rpcss
svchost.exe (1132)	C:\WINDOWS\System32\svchost.exe -k netsvcs
svchost.exe (1268)	C:\WINDOWS\system32\svchost.exe -k NetworkService
svchost.exe (1500)	C:\WINDOWS\system32\svchost.exe -k LocalService
spoolsv.exe (1624)	C:\WINDOWS\system32\spoolsv.exe
VMwareService.exe (176)	"C:\Program Files\VMware\VMware Tools\VMwareService.exe"
alg.exe (1300)	C:\WINDOWS\System32\alg.exe
67751.exe (1216)	C:\WINDOWS\system32\67751.exe
22bd.exe (1656)	C:\WINDOWS\system32\22bd.exe
rundll32.exe (524)	C:\WINDOWS\system32\rundll32 C:\WINDOWS\system32\dlldc.dll, Always
RunDll32.exe (1752)	RunDll32 C:\WINDOWS\TEMP\cml14.tmp,playAdh

Figure 15: Part of the process tree

Process	Command
ieexplore.exe (1192)	"C:\Program Files\Internet Explorer\ieexplore.exe"
ssss[1].exe (1808)	"C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW\ssss[1].exe"
regsvr32.exe (300)	C:\WINDOWS\system32\regsvr32.exe /u /s "C:\WINDOWS\system32\c671.dll"
miniup.exe (352)	C:\DOCUME~1\niki\LOCALS~1\Temp\tm1e\miniup.exe
rundll32.exe (1784)	rundll32 "C:\WINDOWS\Downlo~1\kvon.dll",start
rundll32.exe (1208)	rundll32 "C:\WINDOWS\Downlo~1\fyv.dll",Run
regsvr32.exe (800)	C:\WINDOWS\system32\regsvr32.exe /s "C:\WINDOWS\system32\c671.dll"
67751.exe (836)	C:\WINDOWS\system32\67751.exe -i
67751.exe (1328)	C:\WINDOWS\system32\67751.exe -s
rundll32.exe (620)	C:\WINDOWS\system32\rundll32 C:\WINDOWS\system32\5c1.dll,Always
cml10.tmp (1676)	C:\DOCUME~1\niki\LOCALS~1\Temp\cml10.tmp /S
RunDll32.exe (1264)	RunDll32 C:\DOCUME~1\niki\LOCALS~1\Temp\cml12.tmp,playAds

Figure 16: Part of the process tree

After a restart of the virtual machine, a check in the running processes was made to ascertain if any changes exist or not. The results in the display pane of Process Monitor, which appear in Figure 17, showed 67751.exe process to be running immediately after the system start-up and loading the c671.dll.

Process Monitor - Sysinternals: www.sysinternals.com					
Time of...	Process Name	PID	Operation	Path	Result
12:22:48...	svchost.exe	1104	Load Image	C:\WINDOWS\system32\wbem\wbemsvc.dll	SUCCESS
12:22:48...	67751.exe	1888	Load Image	C:\WINDOWS\system32\shell32.dll	SUCCESS
12:22:49...	67751.exe	1888	Load Image	C:\WINDOWS\system32\shell32.dll	SUCCESS
12:22:49...	67751.exe	1888	Load Image	C:\WINDOWS\system32\c671.dll	SUCCESS
12:22:49...	67751.exe	1888	Load Image	C:\WINDOWS\system32\mf42.dll	SUCCESS
12:22:49...	67751.exe	1888	Load Image	C:\WINDOWS\system32\msvc60.dll	SUCCESS
12:22:49...	svchost.exe	1104	Thread Create		SUCCESS
12:22:49...	svchost.exe	1104	Thread Create		SUCCESS
12:22:50...	67751.exe	1888	Load Image	C:\WINDOWS\system32\shell32.dll	SUCCESS
12:22:50...	67751.exe	1888	Load Image	C:\WINDOWS\system32\shell32.dll	SUCCESS

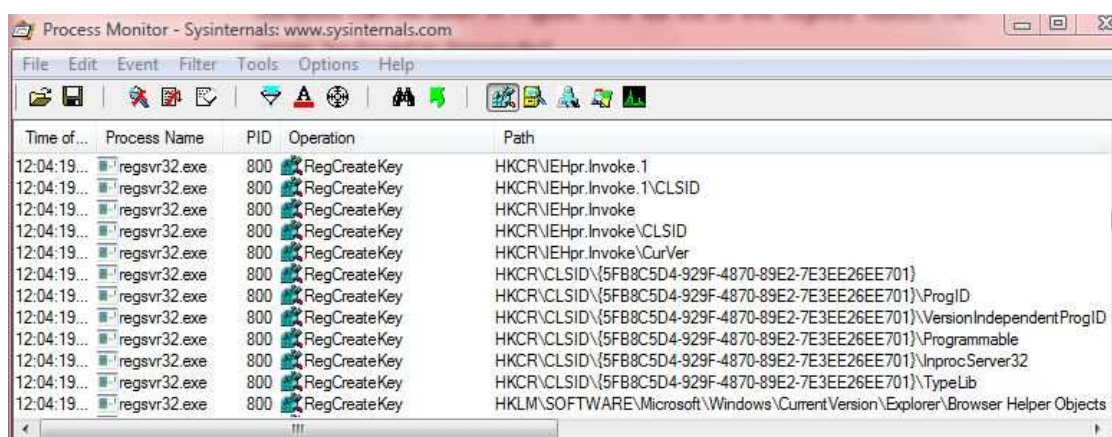
Figure 17: Running processes after the system's start-up

4.2.1.2 ANALYSIS OF REGISTRY CHANGES

Process Monitor is the tool used again to observe the changes in the system's registry. What is important here is the registry keys that were created or changed after visiting the malicious web site, since the registry is normally accessed at least a few hundred times when a program is running. This is why the results are filtered so that the display panel shows only these operations.

The results showed the following:

- The sss[1].exe process opens the HKEY_LOCAL_MACHINE and creates some entries in the *CurrentControlSet \ Control \ SessionManager \ PendingFileRenameOperations* subkey. This subkey stores the names of two files, the first of which is programmed to be renamed to the second one when the system restarts [51]. The entries include the renaming of the files 67751.exe, c671.dll, 5c1.dll and 775c1.dll.
- Next, the process miniup.exe gets a write access on *Software\Microsoft\Windows\CurrentVersion\policies\Explorer\Run* a subkey of HKEY_LOCAL_MACHINE, where it creates the two subkeys kvon and fvy. Each of them contains a command for the rundll32.exe process to run the kvon.dll and fvy.dll files every time the computer starts up.
- cml10.tmp process adds over 20 more entries in the PendingFileRenameOperations subkey.
- In key HKEY_CLASSES_ROOT a number of new subkeys are created. Some of them can be seen in Figure 18 below. Among them a Browser Helper Object is created while the process c671.dll is running, which loads every time the browser starts and affects its functionality [52] and classes belonging to the CLSID key. Objects created of these classes usually represent folders or dialog boxes.



The screenshot shows the Process Monitor application window with the 'Filter' menu open. The main pane displays a list of registry creation events. The columns are 'Time of...', 'Process Name', 'PID', 'Operation', and 'Path'. The events are all 'RegCreateKey' operations performed by 'regsvr32.exe' with PID 800. The paths are various registry keys under HKCR\CLSID and HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects.

Time of...	Process Name	PID	Operation	Path
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\IEHpr.Invoke.1
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\IEHpr.Invoke.1\CLSID
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\IEHpr.Invoke
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\IEHpr.Invoke\CLSID
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\IEHpr.Invoke\CurVer
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\ProgID
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\VersionIndependentProgID
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\Programmable
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\InprocServer32
12:04:19...	regsvr32.exe	800	RegCreateKey	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\TypeLib
12:04:19...	regsvr32.exe	800	RegCreateKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects

Figure 18: Newly created registry keys

- In each one of the newly created keys a value was assigned and some samples can be seen in Figure 19.

The screenshot shows the Process Monitor application window with the following data:

Time of...	Process Name	PID	Operation	Path
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\IEHpr.Invoke.1\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\IEHpr.Invoke.1\CLSID\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\IEHpr.Invoke\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\IEHpr.Invoke\CLSID\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\IEHpr.Invoke\CurVer\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\ProgID\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\VersionIndependentProgID
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\InprocServer32\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\InprocServer32\Threading
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\CLSID\{5FB8C5D4-929F-4870-89E2-7E3EE26EE701}\TypeLib\Default
12:04:19...	regsvr32.exe	800	Reg SetValue	HKCR\TypeLib\{ABBF3E09-6453-43CC-BC46-879C5DC5CB07}\1.0\Default

Figure 19: Newly created registry values

4.1.1.3 ANALYSIS OF NETWORK TRAFFIC

Wireshark is used to capture the traffic of the network during the attempt to get infected by a drive-by download attack. The moment the malicious executable was downloaded Wireshark recorded the HTTP request of the file using the GET request method. Figure 20 shows the traffic recorded at this point of the attack.

The screenshot shows the Wireshark interface with the following data in the packet list:

Source	Destination	Protocol	Info
192.168.198.128	219.148.34.10	HTTP	GET /sss.exe HTTP/1.1
219.148.34.10	192.168.198.128	TCP	http > evm [ACK] Seq=1 Ack=293 win=64240 Len=0
219.148.34.10	192.168.198.128	TCP	[TCP segment of a reassembled PDU]
219.148.34.10	192.168.198.128	TCP	[TCP segment of a reassembled PDU]
192.168.198.128	192.168.198.2	DNS	Standard query PTR 10.34.148.219.in-addr.arpa
219.148.34.10	192.168.198.128	TCP	[TCP segment of a reassembled PDU]
219.148.34.10	192.168.198.128	TCP	[TCP segment of a reassembled PDU]
192.168.198.128	219.148.34.10	TCP	evm > http [ACK] Seq=293 Ack=4648 win=64240 Len=0
219.148.34.10	192.168.198.128	TCP	[TCP segment of a reassembled PDU]
192.168.198.2	192.168.198.128	DNS	Standard query response, No such name

Figure 20: Showing the HTTP Request of the malicious executable

Source	Destination	Protocol	Info
192.168.198.128	60.217.234.138	HTTP	GET /list/2009-08-27/ALL.y HTTP/1.1
60.217.234.138	192.168.198.128	HTTP	HTTP/1.1 200 OK (text/plain)
192.168.198.128	60.217.234.138	HTTP	GET /list/bl.y HTTP/1.1
60.217.234.138	192.168.198.128	HTTP	HTTP/1.1 200 OK (text/plain)
192.168.198.128	60.217.234.138	HTTP	GET /list/2009-08-27/ut_ALL.y HTTP/1.1
60.217.234.138	192.168.198.128	HTTP	HTTP/1.1 200 OK (text/plain)
192.168.198.128	219.148.34.10	HTTP	GET /dmupdate/sss.exe HTTP/1.1
192.168.198.128	219.148.34.7	HTTP	GET /105/bmw.q?uid=316779065212832643084524 HTTP/1.1
219.148.34.7	192.168.198.128	HTTP	HTTP/1.1 200 OK (text/plain)
219.148.34.10	192.168.198.128	HTTP	HTTP/1.1 200 OK (application/octet-stream)
192.168.198.128	60.217.234.138	HTTP	GET /data/5084/taobao0821.lz HTTP/1.1

Figure 21: Displaying a stream of HTTP Requests

Checking out all the HTTP Requests made we are able to see the servers the virtual machine was connected after being infected by the sss[1].exe executable to download the other malicious files. As we see in Figure 21, requests for the lists ALL.y, bl.y and ut_ALL.y are made and when expanding the details of the Hypertext Transfer Protocol we find out that the server hosting the lists is *343.boolans.com*. It is, also, learned that the taobao0821.lz, already known from the processes analysis as a malicious binary, is downloaded from *down.rggzs.com*.

4.1.1.4 VISIBLE EFFECTS OF THE DRIVE-BY DOWNLOAD

As seen in the running processes analysis, the intended purpose of this drive-by download attack is the installation of adware in the user's system. RunDll32.exe is the process responsible for playing advertisements and only some minutes after the initial download it started delivering its payload. An Internet Explorer window opened and displayed the Bank of China web page with URL *www.boc.cn/bocinfo/bi3/200908/t20090824_812459.html*, shown in Figure 22.



Figure 22: Bank of China web site

At the time it opened, a Language pack installation dialog box came up asking for the installation of the Chinese Simplified language pack in order to display correctly the web page. Figure 23 shows the dialog box.



Figure 23: Language pack installation Dialog Box

4.2.2 SECOND DRIVE-BY DOWNLOAD ATTEMPT

The URL *http://x-pager.com/url.exe* picked up for the second drive-by download is recorded in the Malware Domain List web site to install a Banker Trojan. This type of Trojan is known to get installed in the victim's system either by exploiting a web browser vulnerability or by a phishing message. Its main function is stealing personal and financial data, such as usernames and passwords, bank accounts or credit card details.

Before leaving the Trojan to get installed in the virtual machine, I tried to download it with the HDF firewall activated in the normal protection mode. In Figure 24, it is obvious that the file was blocked from entering in the system, as indicated by the value <1>. After de-activating the firewall, the file was finally allowed to get installed. Approximately 3 minutes after I let the malicious file to enter, I set again the protection to on.

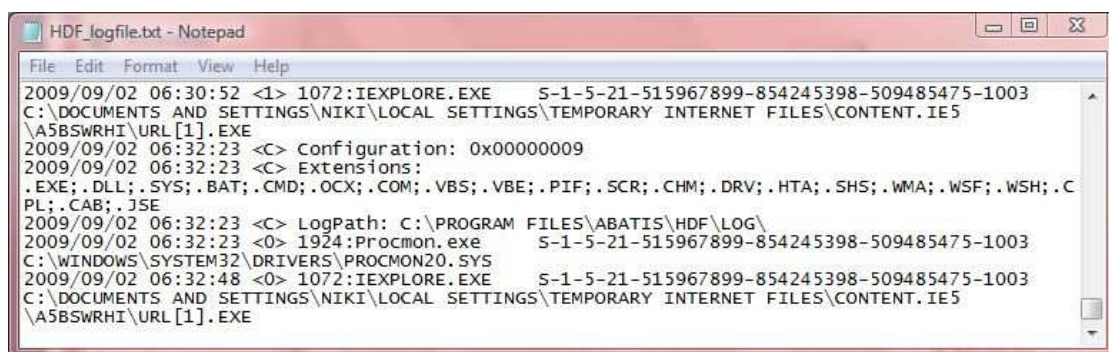


Figure 24: Part of the HDF Logfile

4.2.2.1 ANALYSIS OF RUNNING PROCESSES AND FILE SYSTEM ACTIVITY

Once the compromised web site is visited, the file url[1].exe is downloaded in the Temporary Internet Files. Process Monitor gave a view of the processes that were created and file operations that took place after its installation.

- About 15 seconds after url.exe was downloaded in the system, a number of temporary files were created automatically in the Temp directory of the user's Local Settings. The files were named h2r4.tmp, h2r5.tmp, tst6.tmp, and ms4227.tmp. A sample of the creation of these files can be seen in Figure 25.
- A url[1].exe process was created and started by Internet Explorer that got read access rights on files of the operating system and changed the log that keeps software records.
- The file svchosts.exe is created in the Windows/system32 directory by url[1].exe process.

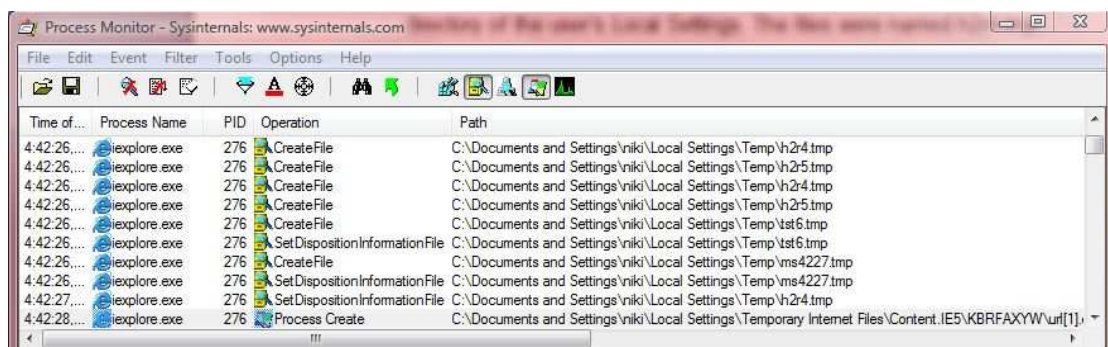


Figure 25: Multiple file creation

- ev[1].htm and inst[1].exe in the Temporary Internet Files folder and after a while it installs the temporary files VRR7.tmp, VRR8.tmp and VRRB.tmp in the WINDOWS Temp directory and creates the corresponding processes.
- VRR7.tmp process modifies the svchost.exe file, responsible for loading services run from dynamic-link libraries and starts a svchost.exe process that loads the services of the malicious file VRR7.tmp.
- The running process ntvdm.exe creates more temporary file in the WINDOWS Temp directory named scs9.tmp and scsA.tmp and makes some modifications.
- A VRRB.tmp process is created by the Winlogon.exe process, which installs the files 1.ico, 2.ico and 3.ico in the WINDOWS Temp directory and creates shortcuts on Desktop opening web pages with adult content. Figure 26 illustrates these changes.
- The same process modifies the iexplore.exe and netsh.exe basic file information and then creates a netsh.exe process that changes the firewall configuration to allow certain incoming connections.
- The sc.exe and sc.ins files are, furthermore, created in the Windows directory.

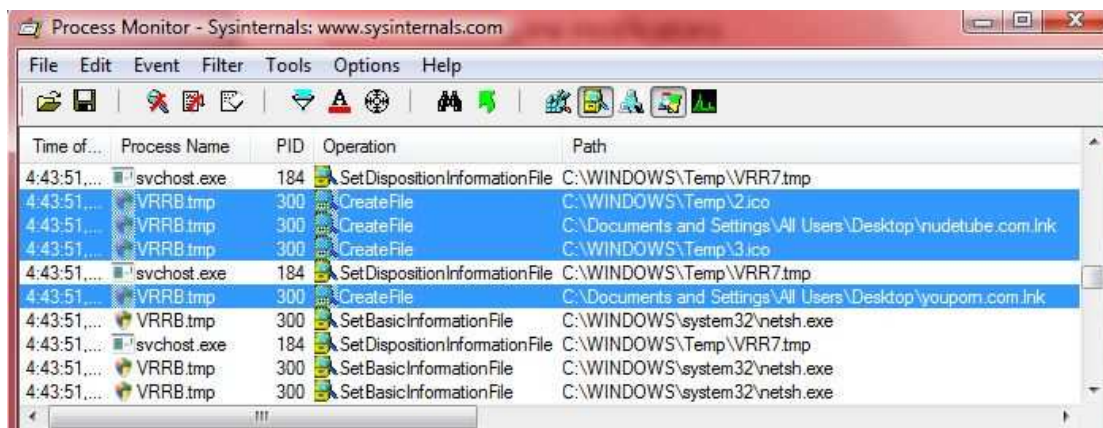


Figure 26: Creation of shortcuts leading to adult pages

- The scvhost.exe process running from before installs the C.tmp, D.tmp and E.tmp files in the system32 folder and lgate[1].htm, fout[1].htm and 41[1].txt in the Temporary Internet Files directory.

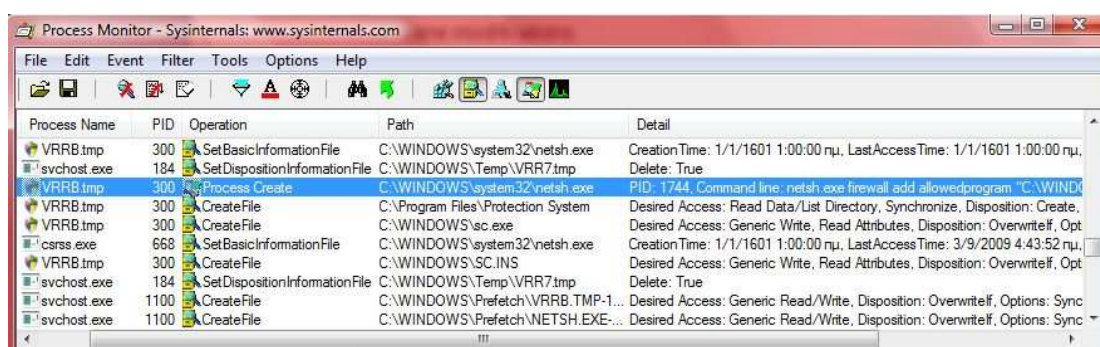


Figure 27: Creation of netsh.exe process

- The E.tmp process is then created that, additionally, installs the oligad32.dll file in system32 directory, makes some modifications in the cmd.exe file and creates a corresponding process that deletes the first one.

Parts of the process tree displaying the relationships between the processes are shown in the Figures 28 and 29 below.

Process	Command
VRR7.tmp (1340)	
svchost.exe (184)	svchost.exe C:\WINDOWS\TEMP\VRR7.tmp
E.tmp (252)	
cmd.exe (1192)	"C:\WINDOWS\system32\cmd.exe" /c del C:\WINDOWS\system32\E.tmp >> NUL
ntvdm.exe (1384)	"C:\WINDOWS\system32\ntvdm.exe" -f +1
VRRB.tmp (300)	"C:\WINDOWS\TEMP\VRRB.tmp"
netsh.exe (1744)	netsh.exe firewall add allowedprogram "C:\WINDOWS\TEMP\VRRB.tmp" "installer" enable
explorer.exe (1516)	

Figure 28: Part of the process tree

Process	Command
VMwareUser.exe (1872)	"C:\Program Files\VMware\VMware Tools\VMwareUser.exe"
ieexplore.exe (276)	"C:\Program Files\Internet Explorer\iexplore.exe"
url[1].exe (1392)	"C:\Documents and Settings\niki\Local Settings\Temporary Internet Files\Content.IE5\KBRFAXYW?url[1].exe"
wireshark.exe (508)	"C:\Program Files\Wireshark\wireshark.exe"
Procmon.exe (1456)	"C:\Program Files\ProcessMonitor\Procmon.exe"
VRRB.tmp (1852)	

Figure 29: Part of the process tree

After the system in the virtual machine was restarted the malicious process svchosts.exe tried to run, as it was configured in the registry, but it was blocked by the HDF Firewall.

4.2.2.2 ANALYSIS OF REGISTRY CHANGES

Process Monitor has been used once again to record the changes that took place in the system's registry. The most important events observed in the registry after visiting the malicious URL are explained below.

- url[1].exe process created the subkey svchosts under the *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run*, as shown in Figure 30, so that the process svchosts.exe will run every time the system starts up.
- The svchost.exe process has set some registry key values concerning TCP/IP parameters, such as the Dhcp Name Server, Default gateway, Domain and Subnet mask.
- A number of Internet Settings registry keys have been modified by the winlogon.exe process, like the default connection settings.
- Epoch, a registry subkey of *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess* is modified by svchost.exe process. The service Shared Access controls the Windows Firewall / Internet Connection Sharing service.

Time of...	Process Name	PID	Operation	Path
4:42:50,...	url[1].exe	1392	RegSetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed
4:42:51,...	url[1].exe	1392	RegCreateKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Run
4:42:51,...	url[1].exe	1392	RegCreateKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Run
4:42:51,...	url[1].exe	1392	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\svchosts
4:42:53,...	url[1].exe	1392	RegCreateKey	HKCU\svchosts
4:42:53,...	url[1].exe	1392	RegCreateKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
4:42:53,...	url[1].exe	1392	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\{Default}
4:42:54,...	winlogon.exe	692	RegCreateKey	HKU\DEFAULT\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Network\Location Awareness
4:42:54,...	winlogon.exe	692	RegCreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\Parameters
4:42:54,...	winlogon.exe	692	RegCreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\Parameters

Figure 30: Newly created registry keys

- The netsh.exe process has set a value to registry subkeys contained in the *HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows NT \ CurrentVersion \ Tracing* and the *HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NapAgent*.
- Finally, a Browser Helper Object is registered under the *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer* and creates a number of new keys in the *HKEY_CURRENT_USER\CLSID* key.

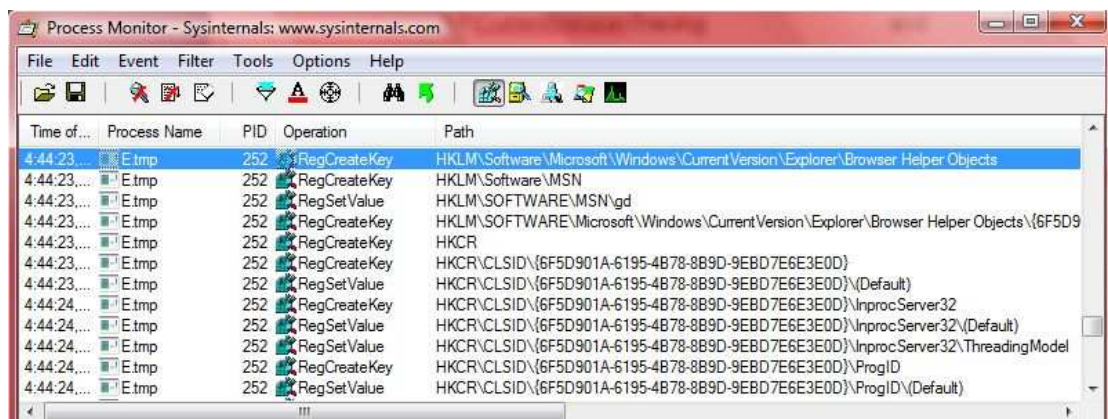


Figure 31: Newly created registry keys and values

4.2.2.3 ANALYSIS OF NETWORK TRAFFIC

Looking at Wireshark's capture file, a flow of the network traffic taking place during the drive-by download is easily detectable. The initial HTTP Request of the malicious URL, is illustrated in Figure 32. The URL *http://x-pager.com* is hosted in the server 213.186.33.2.

The screenshot shows a Wireshark packet capture. The 'Filter' bar is set to 'Expression... Clear Apply'. The packet list shows several packets, with the following details visible for the selected packet (No. 10):

Source	Destination	Protocol	Info
192.168.198.128	192.168.198.2	DNS	Standard query response A x-pager.com
192.168.198.2	192.168.198.128	DNS	Standard query response A 213.186.33.2
192.168.198.128	213.186.33.2	TCP	dab-sti-c > http [SYN] Seq=0 win=64240 Len=0
213.186.33.2	192.168.198.128	TCP	http > dab-sti-c [SYN, ACK] Seq=0 Ack=1 win=64240
192.168.198.128	213.186.33.2	TCP	dab-sti-c > http [ACK] Seq=1 Ack=1 win=64240
192.168.198.128	213.186.33.2	HTTP	GET /url.exe HTTP/1.1
213.186.33.2	192.168.198.128	TCP	http > dab-sti-c [ACK] Seq=1 Ack=291 win=64240
213.186.33.2	192.168.198.128	TCP	[TCP segment of a reassembled PDU]
213.186.33.2	192.168.198.128	TCP	[TCP segment of a reassembled PDU]

Figure 32: Network traffic sample

4.2.2.4 VISIBLE EFFECTS OF THE DRIVE-BY DOWNLOAD

Just a few minutes after the installation of the first malicious files took place, some shortcuts appeared on the desktop that revealed links to adult web sites. An Internet Explorer window was, also, automatically opened that tried to have access to *http://www_getwindowinfo/*, but could not locate the server. It attempted opening the same URI again and again, even after closing the browser. The screenshot in Figure 33 displays these changes.

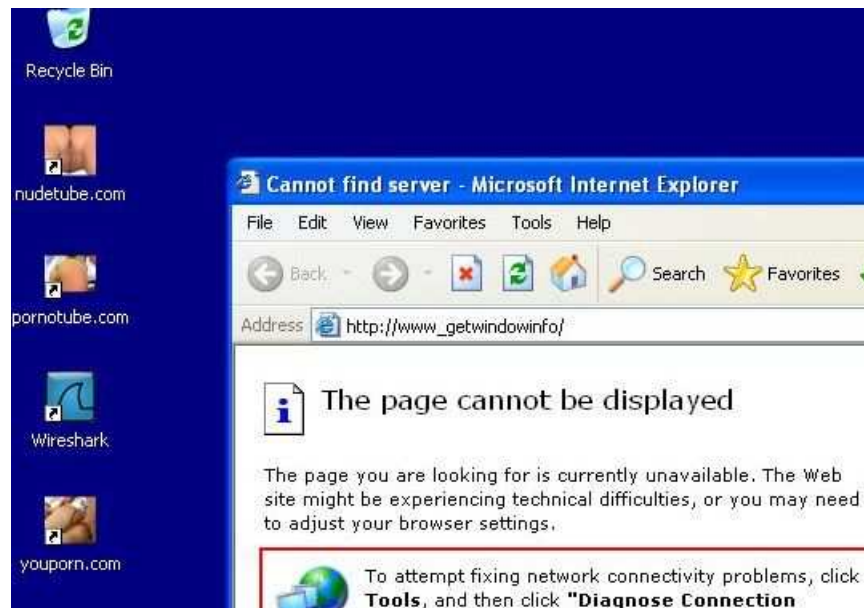


Figure 33: Drive-by download effects

4.3 CONCLUSION

The actual effects that a drive-by download might cause to a computer system are relevant to the installed malware design and function. New processes are being created, others, already existing, are modified or even replaced by a malicious file attempting to disguise like them to trick the user and the system. Important changes occur in the registry, since it is considered to be like a pool keeping the settings of the system, the network and the hardware and software installed. Network communications are being directed to certain servers that either continue the installation of malware in the user's computer or serve advertising purposes.

Special techniques are implemented by web malware to hide their existence from anti-virus programs and enable themselves through firewalls. Stealthy mechanisms become more advanced, so that a user can never know he is infected. A drive-by download attack may lead to major changes taking place in the background, but acts insidiously to infect the more users it can target.

5 DETECTION OF MALICIOUS WEB PAGES

5.1 ANTI-DETECTION MECHANISMS USED BY ATTACKERS

Attackers, nowadays, have a variety of different tools and techniques in their hands that assist them in carrying out their malicious actions. In order to avoid the detection of the malicious scripts they embed in legitimate web pages, they use an obfuscation [53] method to make their scripts undecipherable. Automated tools, like JavaScript obfuscators, have come to support this try to hide the function of the exploit script or the redirection link that leads to it.

Exploit code obfuscation is changing the code in a way that makes it non-understandable but produces the same result when run. JavaScript obfuscation is usually applied, as it is the most common script language processed by web browsers. The script may be transformed multiple times before being injected in the web page to make the code even more complicated and prevent in a high degree any attempt for decompilation and reverse engineering. The example taken from [11, Ch.6] illustrates a three-layer obfuscation. The Visual Basic script below is an exploit script that instructs the browser to go to the URL <http://foto02122006.xxx.ru/foto.scr> and download a malicious binary when it is executed.

```
<script language="VBScript">
    on error resume next
    dl = "http://foto02122006.xxx.ru/foto.scr"
    Set df = document.createElement("object")
    df.setAttribute "classid",
        "clsid:BD96C556-65A3-11D0-983A-00C04FC29E36"
    str="Microsoft.XMLHTTP"
    Set x = df.CreateObject(str,"")
...
    S.close
    set Q = df.createObject("Shell.Application","")
    Q.ShellExecute fname1,"","","open",0
</script>
```

The VBScript exploit was wrapped inside a JavaScript escaped code:

```
<SCRIPT LANGUAGE="Javascript">
<!--
/* criptografado pelo Fal - [...]
document.write(unescape("%0D%0A%3Cscript%20language%3D
%22VBScript%22%3E%0D%0A%0D%0A%20%20%20%20on%20error%20
resume%20next%0D%0A%0D%0A%20%20%20%20%0D%0A%0D%0A%20%20
...

```

```
D%0A%0D%0A%20%20%20%20%3C/script%3E%0D%0A%3C/html%3E"));
//-->
</SCRIPT>
```

And the JavaScript was contained in another JavaScript escaped code, taking the form below that was finally injected in the web page:

```
document.write(unescape("%3CHEAD%3E%0D%0A%3CSCRIPT%20
LANGUAGE%3D%22Javascript%22%3E%0D%0A%3C%21--%0D%0A
/*%20criptografado%20pelo%20Fal%20-%20Deboa%E7%E3o
...
%3C/BODY%3E%0D%0A%3C/HTML%3E%0D%0A"));
//-->
</SCRIPT>
```

The use of this technique has become a trend in the design of drive-by download attacks. Code obfuscation is at an important degree effective against signature and anomaly-based intrusion detection systems, making the detection from anti-virus programs and web analysis tools a difficult task and is certainly a reason for choosing these web attacks as the preferred method in malware distribution.

5.2 MALICIOUS URL DETECTION MECHANISMS

5.2.1 COMBINING STATIC HEURISTICS AND CLIENT HONEYPOTS

This technique, presented in [54], makes use of a static heuristic approach to classify at a starting level URLs given as input. The set of URLs identified as malicious is then examined by in a client honeypot for a final and more reliable classification.

5.2.1.1 FIRST STAGE – USING STATIC HEURISTICS

The first stage of this detection mechanism uses static heuristics, a technique that determines if a URL is malicious or not by examining the HTTP responses and the structure of the HTML page contained. These characteristics are compared with the corresponding ones taken from benign web pages to decide on the maliciousness of the web page.

A set of URLs is retrieved and tested for features that appear commonly in compromised web sites. Script tags, iframes and java obfuscation are some well-known characteristics featured in malicious sites. Table 4, taken from [54, Ch.IV], summarizes the basic attributes extracted from the contained HTML page.

CATEGORY	ATTRIBUTES	DESCRIPTION
Exploit	Plug-ins	Count of the number of applet and object tags.
	Script Tags	Count of script tags.
	XML Processing Instructions	Count of XML processing instructions. Includes special XML processing instructions, such as VML.
Exploit Delivery Mechanism	Frames	Count of frames and iFrames including information about the source.
	Redirects	Indications of redirects. Includes response code, meta-refresh tags, and JavaScript code.
	Script Tags	Count of script tags including information about the source.
Hiding	Script Obfuscation	Functions and elements that indicate script obfuscation, such as encoded string values, decoding functions, etc.
	Frames	Information about the visibility and size of iFrames.

Table 4: Extracted attributes

A machine learning algorithm is, then, used to decide about a URL being malicious or not. The extracted features from the candidate URLs are compared with the already existing features of both benign and malicious web pages and the decision is made with the use of a decision tree that is generated. A sample decision tree is shown in Figure 34, also taken from [54, Ch.IV].

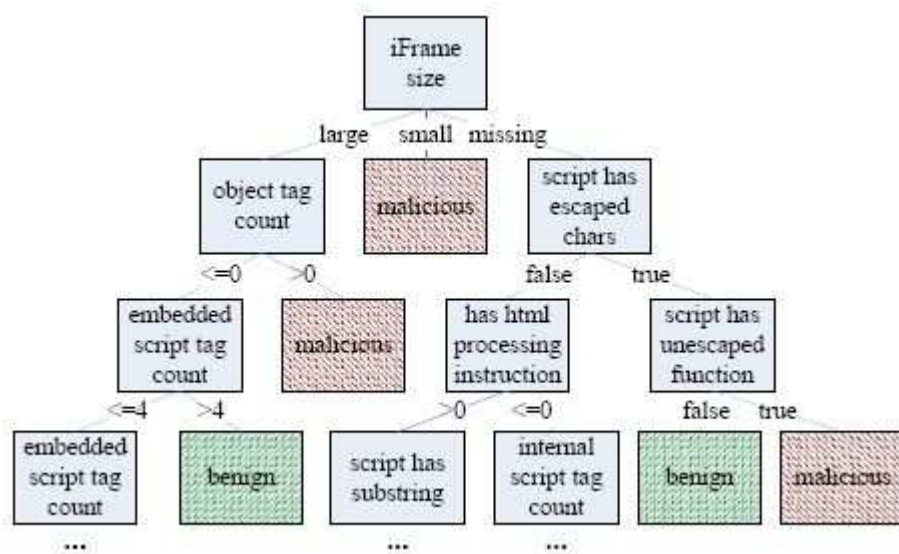


Figure 34: Decision tree

5.2.1.2 SECOND STAGE – USING CLIENT HONEYPOTS

Client honeypots [55] are security devices used widely to detect malicious web pages. They are vulnerable computer systems in order to attract malicious activities. Commercial client honeypot frameworks are available such as, the high interaction Capture-HPC [56] or the low interaction HoneyC framework [57].

For the task of detecting malicious URLs, client honeypots run virtual machines with unpatched versions of web browsers and applications and interact with potentially malicious web servers. During these interactions, the system is monitored for unauthorized changes in its state, by observing registry changes, running processes and file modifications that will identify if malicious activities have taken place.

Client honeypots are used, therefore, at the second stage of the detection mechanism to examine the URLs proved to be malicious at the first stage. This first classification might include a number of false positives (URLs that appear to be malicious but are not actually), because the attributes that usually distinguish a compromised web page may, also, be used in a legitimate way. Honeypots are characterized by a false positive rate close to zero, thus, all the false positives will be excluded. The URLs are visited by the virtual machines and examined having as a basis the changes that took place in the system state and the final classification is made.

5.2.1.3 EXAMPLE OF THE DETECTION MECHANISM

A year ago, Google created a relevant mechanism to detect dangerous web pages that included static heuristics and the development of a large scale honeypot network [20, Ch.3]. Taking advantage of the web repository it maintains, Google tried to identify candidate URLs using the mapreduce [58] framework. Each web page was scoured for features indicative of exploits, like iframes or obfuscated JavaScript. With a machine-learning algorithm they translated these features to end up with a number of potentially malicious web pages. At the next stage, these web pages were retrieved once again by the virtual machines running on the honeynet they had developed. The final set of malicious URLs was decided by combining the observations from the system's state changes and anti-virus scanners results on the incoming HTTP responses. Figure 35 is a diagram of the detection architecture.

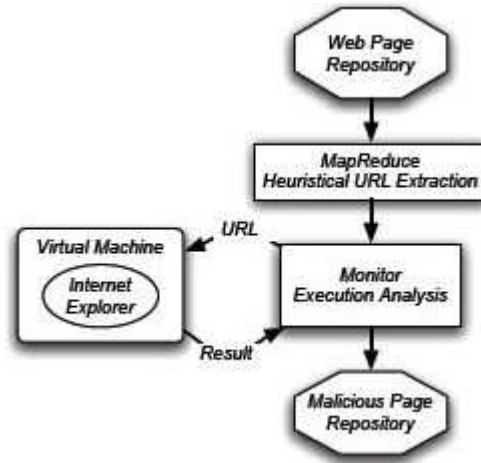


Figure 35: Overview of Google's detection architecture

5.2.2 ANALYZING DNS AND WEB SERVER RELATIONSHIPS

This malicious URL detection mechanism records and analyzes the network traffic during visiting web sites. Web pages that display malicious behavior are following particular patterns that could help with their recognition. The mechanism consists of a machine learning algorithm that compares information collected from the web servers that host the candidate web sites and the DNS servers that are involved in the resolution of the host names to information already acquired from both benign and malicious web pages [59].

ATTRIBUTE	DESCRIPTION
Number of Unique HTTP Servers	The number of unique HTTP servers. Obtained through counting the IP addresses of packets originating on ports 80, 8080, 8088, 3128 and 443.
Number of Redirects	The number of 301, 302 and 303 redirects. Obtained by inspecting the response code of web pages returned by any HTTP server.
Number of Redirects to Different Country	The number of 301, 302 and 303 redirects in which the server that issues the redirect response is located in a different country than the server to which the browser is being forwarded to.
Number of Redirects to Same Country	The number of 301, 302 and 303 redirects in which the server that issues the redirect response is located in the same country than the server to which the browser is being forwarded to.
Number of Domain Name Extensions	The number of domain name extensions of all host names that operate a web server.
Number of Unique	The number of DNS servers involved in making a DNS lookup.

DNS Servers	The DIG tool is used to count the number of responsible DNS servers for each host name encountered.
-------------	---

Table 5: Extracted attributes

Features that characterize a malicious web site may be the number of redirections it follows, the different Web and DNS servers it visits and the countries where these are located. The main attributes that participate in the decision process are presented in Table 5 [59, Ch.IV]. Figure 36, moreover, illustrated a sample of the decision tree created by the machine learning algorithm [59, Ch.IV].

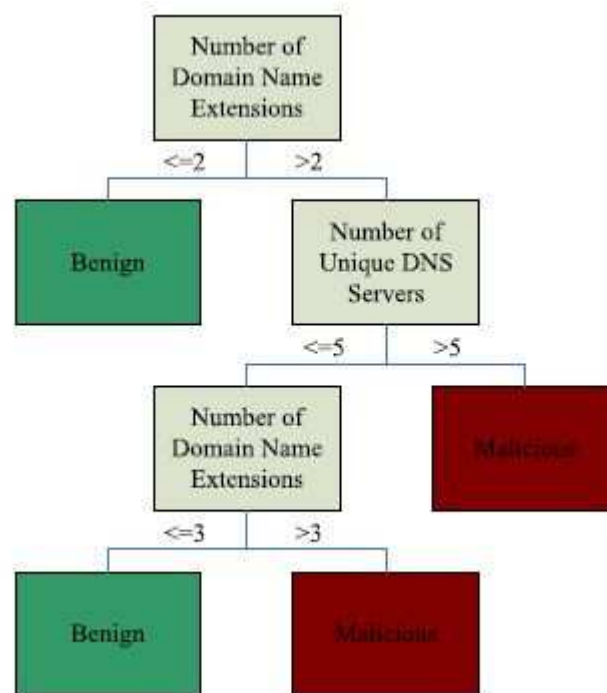


Figure 36: Decision tree

5.3 CONCLUSION

The mechanisms presented here have both advantages and disadvantages. Using a combination of them could be proved a more reliable and successful way to detect malicious URLs.

The first method that uses a static heuristics approach combined with a high interaction client honeypot is, in general, an efficient method. It is able to detect zero-day attacks (attacks that exploit a vulnerability for which no patch has been released) and recognize malicious web sites with a false positive rate of zero, things. The most

important drawback of this detection mechanism is its inability to detect certain malicious web sites. The problem derives from the fact that the attack targets very specific vulnerabilities. If the web browser used by the client honeypot is not compatible with the vulnerability the exploit script targets, the attack will fail and the malicious behavior of the web site will not be detected.

The second method that observes the underlying DNS and web server relationships to detect malicious web pages has the privilege of being implemented in any client capable of rendering a web page. It does not have the need of a client honeypot that requires considerable resources that may affect, consequently, the system's performance. However, it also has the disadvantage of not identifying certain attacks. As discussed in Section 3, some malicious scripts are directly injected in the web server by either using a cross-site scripting attack or by compromising a database. In this case, the web page will appear to be coming from one source, thus the machine learning algorithm will classify it as non-malicious.

6 CONCLUSION

Drive-by download attacks are considered to be among the most threatening malware distribution techniques nowadays, concerning the security world more and more. Targeting vulnerabilities and taking advantage of the wide spread of Internet users, adversaries intend to gain access in the victim's computer to fulfill their malicious purposes silently via infecting "innocent" web sites.

The effects drive-by download attacks cause and the traces they leave on the victim's computer vary according to the type and the function of the malware installed in the computer. The number of processes running after the attack is increased due to the execution of the malicious files that spread rapidly without the user realizing it. Registry keys are created to ensure the existence of malware in the system and the continuation of their malicious activities. Network traffic faces a rise too, due to the number of connections established to different servers for the download of malicious software and the number of TCP packets sent.

Although, the client-based attacks are an increasing threat in the web no big steps have been made for their successful detection and prevention. Anti-virus programs, firewalls and intrusion detection systems have proved to be an effective method to encounter different types of malware, but they are not that efficient when it comes to browser exploitation and drive-by downloads. A number of mechanisms have been developed to detect malicious URLs, but they have not been tested under actual conditions.

A solution to this problem is urgent to be found as I believe that the drive-by download threat will continue to be on the top of the future's challenges and maybe even worse than that. Cybercriminals will always find holes to continue their organized actions for committing fraud and stealing sensitive data towards commercial purposes and financial gain.

BIBLIOGRAPHY

REFERENCES

- [1] R. Naraine, 5000 Web sites hacked to serve up malware cocktail, August 2009, <http://blogs.zdnet.com/security/?p=4091>.
- [2] J. Leyden, Drive-by download attack compromises 500K websites, May 2008, http://www.channelregister.co.uk/2008/05/13/zlob_trojan_forum_compromise_attack/.
- [3] D. Danchev, Research: 80% of Web users running unpatched versions of Flash/Acrobat, August 2009, <http://blogs.zdnet.com/security/?p=4097>.
- [4] Sophos, Security threat report 2008, https://secure.sophos.com/sophos/docs/eng/marketing_material/sophos-security-report-08.pdf.
- [5] ZDNet Australia, Superguide: The death of trusted websites, March 2008, <http://www.zdnet.com.au/insight/security/soa/Superguide-the-death-of-trusted-Web-sites-/0,139023764,339286648,00.htm>.
- [6] J. Leyden, Drive-by download attacks menace UK.gov, July 2008, http://www.theregister.co.uk/2008/07/23/drive_by_download_asprox_menace/.
- [7] R. Westervelt, F-Secure latest security vendor hacked, February 2009, http://searchsecurity.techtarget.com/loginMembersOnly/1,289498,sid14_gci1347639,00.html.
- [8] R. Westervelt, Kaspersky website hacked, Customer activation codes exposed, February 2009, http://searchsecurity.techtarget.com/loginMembersOnly/1,289498,sid14_gci1347341,00.html.
- [9] P. Barford and V. Yagneswaran, *An inside look at Botnets*, Advances in Information Security, Springer, 2007.
- [10] ScanSafe, Annual Global Threat Report, Trends for January 2008 – December 2008, March 2009, http://www.scansafe.com/resources/global_threat_reports2/resources_-_global_threat_reports/gtr_request.
- [11] N. Provos, D. McNamee, P. Mavrommatis, K. Wang and N. Modadugu, The ghost in the browser, Analysis of Web-based Malware, *First Workshop on Hot Topics in Understanding Botnets (HotBots 07)*, 2007.
- [12] SANS Institute Infosec Reading Room, Understanding IIS Vulnerabilities, Fix them, 2001.
- [13] Security Focus, Multiple Microsoft IIS Vulnerabilities, <http://www.securityfocus.com/bid/6068>.
- [14] C. Anley, Advanced SQL Injection in SQL Server Applications, NGSSoftware Insight Security Research (NISR) Publication, 2002.
- [15] SecuriTeam, Invision Power Board SQL Injection Vulnerability, May 2005, <http://www.securiteam.com/exploits/5GP0E2KFQQ.html>.

- [16] SecuriTeam, XMLRPC Remote Commands Execution (Multiple Exploits), July 2005, <http://www.securiteam.com/exploits/5BP0O20GBS.html>.
- [17] Web Application Security Consortium, Threat Classification, Cross-Site Scripting, http://www.webappsec.org/projects/threat/classes/cross-site_scripting.shtml.
- [18] Web Application Security Consortium, MySpace hack spreading, July 2006, http://www.webappsec.org/projects/whid/byid_id_2006-37.shtml.
- [19] Sounds of the Dungeon, Orkut XSS, http://antrix.net/journal/techtalk/orkut_xss.html.
- [20] N. Provos, P. Mavrommatis, M.A. Rajab and F. Monroe, All your iFRAMEs point to us, Google Technical report, *USENIX Security Symposium*, February 2008.
- [21] ScanSafe, Web 2.Owned, A history of malware on the web, White Paper, 2009.
- [22] D. Goodin, Google's DoubleClick spreads malicious ads (again), February 2009, http://www.theregister.co.uk/2009/02/24/doubleclick_distributes_malware/.
- [23] The Wall Street Journal, Sneaky New Web Ads Contain Hidden Viruses, June 2009, <http://www.foxnews.com/story/0,2933,526605,00.html?sPage=fnc/scitech/cybersecurity>.
- [24] BBC News, Cyber crime tool kits go on sale, September 2009, <http://news.bbc.co.uk/1/hi/technology/6976308.stm>.
- [25] McAfee, MPack Tool, http://vil.nai.com/vil/content/v_142501.htm.
- [26] J. Leyden, Hackers exploit Neosploit to booby trap BBC, US Postal Service, October 2008, http://www.theregister.co.uk/2008/10/03/neosploit_powered_mass_hack_attack/.
- [27] Interesting statistics from the Secunia PSI, January 2008, <http://secunia.com/blog/18/>.
- [28] SANS Institute, SANS Top 20, <http://www.sans.org/top20/>.
- [29] C. Alme, Web Browsers: An Emerging Platform Under Attack, McAfee White Paper, June 2009.
- [30] Web Browser Security Summary, <http://www.webdevout.net/browser-security>.
- [31] M. Daniel, J. Honoroff and C. Miller, Engineering heap Overflow Exploits with JavaScript, 2nd *USENIX Workshop on Offensive Technologies*, 2008.
- [32] A. Sotirov, Heap Feng Shui in JavaScript, *BlackHat Europe*, 2007.
- [33] ActiveX Definition, http://reviews.cnet.com/4520-6029_7-5748090-1.html?tag=forums06;posts.
- [34] Microsoft Security Research and Defense, Vulnerability in MPEG2TuneRequest ActiveX Control Object in msvidctl.dll, <http://blogs.technet.com/srd/archive/2009/07/06/new-vulnerability-in-mpeg2tunerequest-activex-control-object-in-msvidctl-dll.aspx>.
- [35] M. Egele, P. Wurzinger, C. Kruegel and Engin Kirda, Defending Browsers against Drive-by Downloads: Mitigating Heap-spraying Code Injection Attacks,

In Detection of Intrusions and Malware, and Vulnerability Assessment, 6th International Conference , DIMVA, 2009.

- [36] M. Egele, E. Kirda and C. Kruegel, Mitigating Drive-by Download Attacks: Challenges and open problems.
- [37] S. Frei, T. Duebendorfer, G. Ollmann and M. May, Understanding the web browser threat: Examination of vulnerable online web browser populations and the “insecurity iceberg”, ETH Zurich Tech Report Nr. 288.
- [38] Common Vulnerabilities and Exposures, <http://cve.mitre.org/>.
- [39] Microsoft Malware Protection Center, Microsoft Security Intelligence Report volume 6 (July through December 2008),
<http://www.microsoft.com/security/portal/Threat/SIR.aspx>.
- [40] D. Goodin, Nasty PDF exploit runs wild, October 2007,
http://www.theregister.co.uk/2007/10/24/pdf_exploit_in_the_wild/.
- [41] Norton from Symantec, Misleading Applications – Rogue Antispyware – Smithfraud Anti-spyware removal,
<http://www.symantec.com/norton/theme.jsp?themeid=mislead>.
- [42] Microsoft Help and Support, Windows registry information for advanced users, February 2008, <http://support.microsoft.com/?scid=kb%3Ben-us%3B256986&x=8&y=8>.
- [43] Abatis HDF. <http://www.abatis-hdf.com/product.html>.
- [44] Process Monitor. <http://technet.microsoft.com/en-us/sysinternals/bb896645.aspx> .
- [45] Wireshark. <http://www.wireshark.org/about.html>.
- [46] Spyware Doctor. <http://www.pctools.com/spyware-doctor/>.
- [47] Malware Domain List. <http://www.malwaredomainlist.com/mdl.php>.
- [48] MalwareURL.com. <http://www.malwareurl.com/listing-urls.php?urls=off>.
- [49] PrevX, C671.DLL,
<http://spywarefiles.prevx.com/RRHEFJ44357287/C671.DLL.html>.
- [50] PrevX, 67751.EXE, <http://www.prevx.com/filenames/2354258062450287753-X1/67751.EXE.html>.
- [51] Microsoft TechNet, PendingFileRenameOperations,
<http://technet.microsoft.com/en-us/library/cc960241.aspx>.
- [52] Browser Helper Objects: The browser the way you want it,
[http://msdn.microsoft.com/en-us/library/bb250436\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/bb250436(VS.85).aspx).
- [53] M. Mateas and N. Montfort, A Box Darkly: Obfuscation, Weird Languages and Code Aesthetics, *Proceedings of the 6th Digital Arts and Culture Conference*, IT University of Copenhagen, 2005.
- [54] C. Seifert, I. Welch and P. Komisarczuk, Identification of Malicious Web Pages with Static Heuristics, 2008.
- [55] C. Seifert, I. Welch and P. Komisarczuk, Taxonomy of honeypots, 2006.
- [56] Capture-HPC Client Honeypot Framework.
<https://projects.honeynet.org/capture-hpc>

- [57] C. Seifert, I. Welch and P. Komisarczuk, HoneyC – The Low-Interaction Client Honeypot, 2006.
- [58] J. Dean and S. Ghemawat, MapReduce: Simplified data processing on large clusters, *Proceedings of the 6th Symposium on Operating System Design and Implementation*, December 2004.
- [59] C. Seifert, I. Welch and P. Komisarczuk, Identification of Malicious Web Pages Through Analysis of Underlying DNS and Web Server Relationships, *33rd Annual IEEE Conference on Local Computer Networks*, 2008.

ADDITIONAL SOURCES

Dr. I.G. Muttik, Manipulating the Internet, McAfee AVERT, *Virus Bulletin Conference*, October 2005.

E. Skoudis, L. Zeltser, *Malware: Fighting malicious code*, Prentice Hall PTR, 2003.

J. Aycock, *Computer Viruses and Malware*, Advances in Information Security, Springer, 2006.

A. Orebaugh, G. Ramirez, J. Burke, G. Morris, L. Pesce and J. Wright, *Wireshark & Ethereal. Network Protocol Analyser Toolkit*. Syngress, 2006.

J. Honeycutt, *Microsoft Windows Registry Guide*, Second Edition, 2005.