G Data

White Paper 03/2014

Exploit Protection

System Security Research

G Data. Security Made in Germany.

Motivation

The biggest entry point for malware these days is attacks via the web. According to one study, a total of 68% of all malware originates from there. With malware that is not currently covered by signatures, it is no less than 90% (Palo Alto Networks, 2013). According to research by Google, 98% of malicious websites distribute malware via so-called drive-by downloads (Rajab, et al., 2011). These involve malware being downloaded and executed without the user even realising. "Exploits" are used to do this – tools that exploit vulnerabilities in the compromised user's system.

Unfortunately, supposedly secure surfing behaviour, in which no shady websites are visited, will not help prevent this. Formerly, malware was mainly distributed via sex and gambling sites. Today 85% of attacks stem from compromised legitimate websites (Websense, 2013).

The German Federal Office for Information Security (BSI) has recently reported multiple such attacks. Those affected were "popular sites for news, politics, lifestyle and specialist magazines, daily newspapers, job boards and town portals" (BSI, 2013). Affected sites included IT site pcwelt.de[1] and the weather site wetter.com[2].

Besides use on infected websites, exploits are also used in other areas. For example, primed PDFs are frequently sent to companies as part of targeted attacks: after being opened, vulnerabilities in Adobe Reader are exploited.

It is often claimed that keeping the software on your computer fully up to date at all times is sufficient protection against exploits. Certainly the security of computers is increased by using the latest software patches. But practically it is very difficult to achieve the goal of having a really up-to-date computer. To do this, 150 patches for 50 applications from 14 different providers would have to be installed every year on an average computer (Frei, 2011). Besides this almost incomprehensible amount of patches, it should also be noted that many software packages are pre-installed by the manufacturer of the computer. Therefore the user is often unaware that there even is vulnerable software on the computer. Because of compatibility issues or an expired service it may also be impossible to install updates.

In fact 39% of all computers are susceptible to just the exploits on the web that we know about[3]. In practice, computers with no security holes do not exist.
Every piece of software contains vulnerabilities that the provider is often not even aware of, or has not yet provided a patch for. Exploitation of such vulnerabilities is referred to as a zero-day attack. It normally takes about ten months for such attacks to be identified by the provider of the software. After public disclosure, 42% of such attacks are carried out on a large-scale by cyber criminals within the first 30 days. However, this period is usually not long enough for the provider of the affected software to remove the vulnerability (Bilge & Dumitras, 2012).

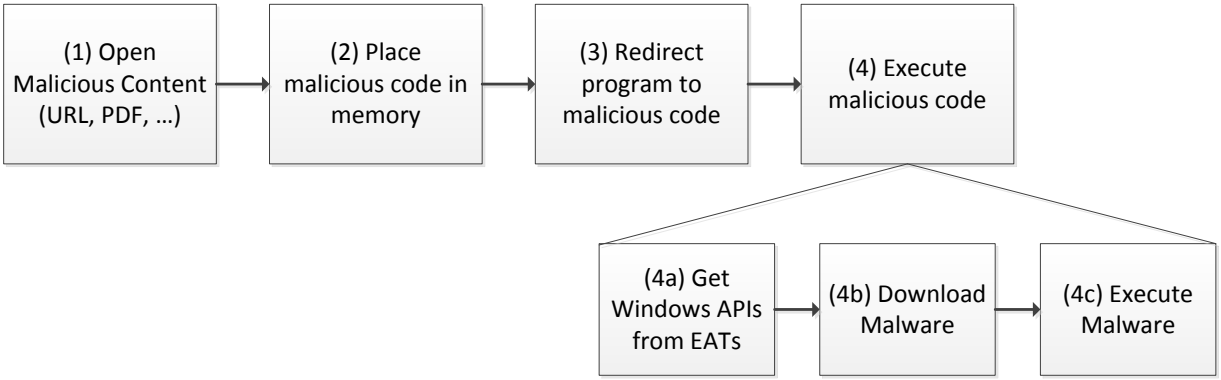
In summary, it can be said that security holes and the exploits associated with them look like they are the biggest security problem currently in existence.

[1] http://www.pcwelt.de/news/pcwelt.de_ist_wieder_Malware-frei-In_eigener_Sache-7383231.html
[2] http://www.heise.de/security/meldung/Virenalarm-auf-Wetter-com-1575304.html
[3] https://community.qualys.com/blogs/laws-of-vulnerabilities/2013/11/27/secure-your-browser-before-shopping-online

How does an exploit work?

Every exploit begins with malicious content being loaded (1). As stated, this might for example involve a URL being accessed in a browser, or a PDF being opened in a PDF reader.

The exploit itself functions as follows in the simplest case: malicious code is stored somewhere in the memory (2). In the next step, a vulnerability enables the program under attack – for example the web browser – to be redirected to this malware (also called shellcode) (3).



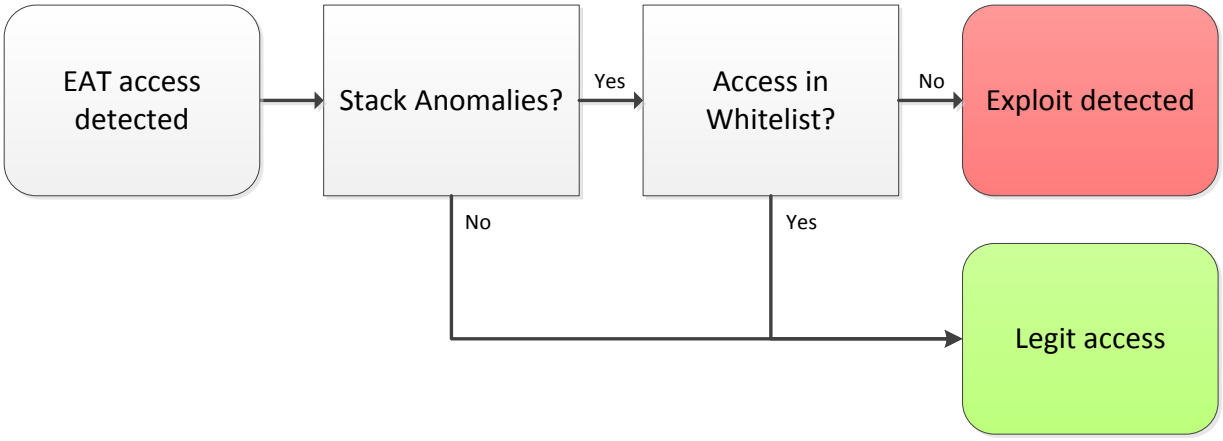
In order to be able to carry out malicious activities, the shellcode first determines which Windows interfaces (Windows APIs) are required for executing the malicious functions. Normally this involves functions for accessing the hard disk or registry, or functions for accessing the Internet.
Every operating system has a different memory layout. So, regardless of the precise version, the shellcode does this by searching the tables in the system libraries (export address tables or EATs). This is where the memory addresses for the functions it is looking for are stored (4a).

In practice, shellcode is mostly very minimalist code whose only function is to download the actual malware from the Internet (4b) and run it (4c). This then is the drive-by download.

How does G Data Exploit Protection prevent this?

G Data Exploit Protection is based on an access filter on the tables that contain the Windows APIs (Export Address Table Access Filtering or EAF).

If access is determined, the process is checked for anomalies that indicate an exploit. Such anomalies in the memory "stack" for the protected program occur when the program flow is redirected to the shellcode.



Basically Exploit Protection is a whitelisting approach. This means that every access that is not expressly permitted is considered malicious. In this way Exploit Protection also provides proactive protection against previously unknown attacks, such as the zero-day attacks mentioned above. In this regard Exploit Protection differs from traditional blacklisting processes, as are used to detect malware via signatures, for example.

If an exploit is detected, a message is displayed and the compromised program is terminated.

Except Java

In recent times, Java has been the biggest attack vector for malware. According to research by Cisco, Java was an entry point on 91% of all computers compromised by exploits in 2013 (Cisco, 2014).

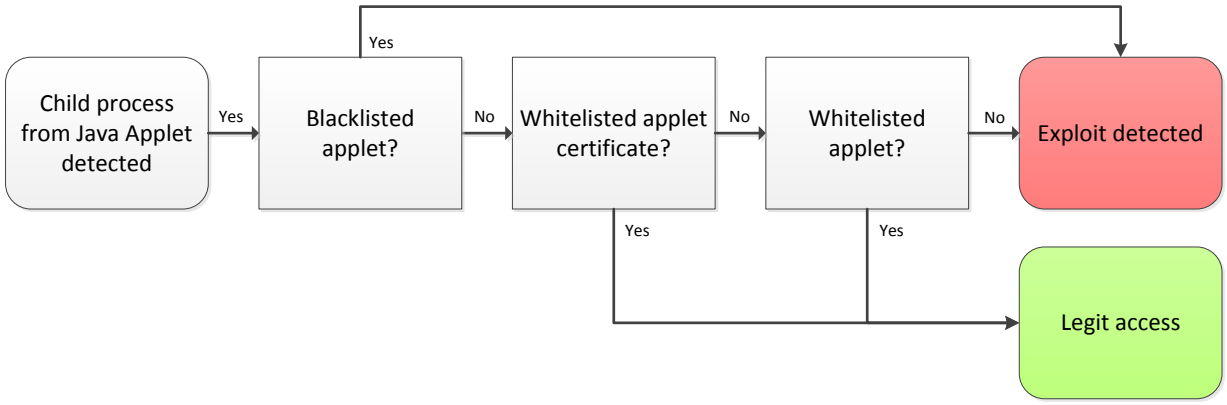
Java exploits are so popular with attackers because Java is so widely used (on 89% of all computers in the USA, according to Cisco). This also means that vulnerable versions are often installed (76% according to Cisco), and so many security holes have been detected that even the most recent version is often still vulnerable.

In addition, Java exploits are often more stable than traditional exploits. Traditional exploits attack via shellcode, as described above, and so often depend on parameters that differ from system to system. Java exploits, on the other hand, function within the logic of Java, and so are just as system-independent as any Java code.

The Export Address Table Access Filter in Exploit Protection cannot deal with Java exploits working within the Java logic. Therefore another type of protection has been developed.

In practice Java exploits are implanted as applets that are called up via a browser. These applets then use vulnerabilities to switch off the Java Security Manager, which normally sets restricted rights for applets. This normally stops Java applets from launching any system processes of their own. However, in this case, once the Java Security Manager has been overridden, drive-by downloads can be carried out again – which is when the actual malware is downloaded and run.

G Data Exploit Protection utilizes the fact that applets that launch system processes are only permitted under exceptional conditions, and in practice are as good as non-existent.



Therefore a mechanism developed in-house is used to ensure that applets are only permitted to launch system processes if the applet itself is on a whitelist, or is certified by a known provider.

List of protected processes

The list of protected processes can be adapted via signature updates.

Current status (2 February 2014):

Browser (incl. plug-ins such as Adobe Flash):

- Internet Explorer
- Mozilla Firefox
- Google Chrome
- Opera

Microsoft Office:

- Excel
- Outlook
- PowerPoint
- Word

Office (other):

- Adobe Reader
- Foxit Reader
- Mozilla Thunderbird

Media players:

- Windows Media Player
- Apple Quicktime Player
- Radionomy Winamp
- RealNetworks Real Player
- VideoLAN VLC

(Un)packers:

- 7-Zip
- WinZip
- WinRAR

Other:

- Oracle Java
- Pidgin Instant Messenger
- IrfanView

Bibliography

Bilge, L., & Dumitras, T. (2012). *Before we knew it - An Empircal Study of Zero-Day Attacks In The Real World.* Downloaded from http://users.ece.cmu.edu/~tdumitra/public_documents/bilge12_zero_day.pdf

BSI. (2013). *PR-Mitteilung: BSI weist erneut auf breitflächige Verteilung von Schadprogrammen über Werbebanner hin.* Downloaded from https://www.bsi.bund.de/DE/Presse/Pressemitteilungen/Presse2013/Verteilung_von_Schadprogrammen_ueber_Werbebanne_05042013.html

Palo Alto Networks. (2013). *The Modern Malware Review.* Downloaded from http://www.ioactive.com/pdfs/ZeusSpyEyeBankingTrojanAnalysis.pdf

Rajab, M. A., Ballard, L., Jagpal, N., Mavrommatis, P., Nojiri, D., Provos, N., et al. (2011). *Trends in Circumventing Web-Malware Detection.* Downloaded from http://research.google.com/archive/papers/rajab-2011a.pdf

Websense. (2013). *Threat Report.* Downloaded from http://www.websense.com/assets/reports/websense-2013-threat-report.pdf