



About the authors:

Dan Haagman (BSc., CSTP, CFIA) & Byrne Ghavalas (CSTP, CFIA, GCFA) instruct and practice in computer forensics for 7Safe – an independent Information Security practice delivering an innovative portfolio of services including; Forensic Investigation, BS7799 Consulting, Penetration Testing & Information Security Training. 14/01/2005

Trojan Defence: A Forensic View

The Trojan defence; “I didn’t do it, someone else did”– myth or reality? This two part article investigates the fascinating area of Trojan & network forensics and puts forward a set of processes to aid forensic practitioners in this complex and difficult area. Part I examines the Trojan defence, how Trojan horses are constructed and considers the collection of volatile data. Part II takes this further by investigating some of the forensic artefacts and evidence that may be found by a forensic practitioner and considers how to piece together the evidence to either accept or refute a Trojan defence.

A Background to the Trojan defence

This two-part article examines some of the issues surrounding the Trojan defence from the perspective of the forensic practitioner. However, before we start here are some comments worth considering:

“A landmark trial recently found that illegal pornography had been placed on an innocent man's computer by a Trojan program ...”

¹ BEWARE TROJANS BEARING GIFS BY NEIL BARRETT, ITWEEK 03 JUN 2003

“Julian Green, 45, endured nine months of being branded a paedophile before it was proved that the 172 images were caused by a computer virus.”

² 'CHILD PORN VIRUS WRECKED MY LIFE' BY RICHARD ALLEN, EVENING STANDARD 31 JULY 2003

“The acquittal of a teenager accused of carrying out a high-profile hack attack has cast doubts over future computer crime prosecutions, say experts.”

³ QUESTIONS CLOUD CYBER CRIME CASES BBC NEWS UK EDITION 17 OCTOBER 2003

A forensic analysis of Caffrey’s computer revealed no trace of a Trojan. Graham Cluley, senior technology consultant at the security firm Sophos, said “The Caffrey case suggests that even if no evidence of a computer break-in is unearthed on a suspect’s PC, they might still be able to successfully claim that they were not responsible for what their computer does, or what is found on its hard drive.”⁴ The Trojan defence places a lot of pressure on the prosecution which in turn places pressure on the forensic investigators to prove, beyond all reasonable doubt, that the accused is responsible for the evidence located on the computer. Mark Rasch of SecurityFocus, comments in his article, “The Giant Wooden Horse Did It!”⁵ that this defence is all the more frightening because it could be true. He asks “...if you were a hacker, would you want to store your contraband files on your own machine, or, like the cuckoo, would you keep your eggs in another bird's nest?”

1 <http://www.itweek.co.uk/comment/1141339>
2 <http://www.thisislondon.co.uk/news/articles/6026981?source=evening%20standard>
3 <http://news.bbc.co.uk/1/hi/technology/3202116.stm>
4 <http://news.bbc.co.uk/1/hi/technology/3202116.stm>
5 http://www.theregister.co.uk/2004/01/20/the_giant_wooden_horse_did/

Storing files on other systems is a common tactic for attackers. Warez Dudes store their contraband on high-speed servers, hackers store their 'rootkits' or other tools on compromised systems or other publicly accessible servers. No doubt many forensic practitioners have seen examples of this; however, the HoneyNet Project⁶ has several challenges which show evidence of this practice.

Mark further points out, "In late December 2003, companies around the world began to report a new kind of cyber-attack that had been apparently going on for about a year. Cyber extortionists (reportedly from Eastern Europe) threatened to "plant" child pornography on their computers and then call the cops if they didn't agree to pay a small fee. Unless the recipient pays a nominal amount (\$30), the hacker claims he will either wipe the hard drive or plant child porn. The possibility of Trojans and the relative ease with which they could be used to promulgate just such an attack made the threats credible."

It is clear that the Trojan defence needs to be carefully considered. As forensic practitioners, it is important that whenever an examination is conducted, we should keep the Trojan defence possibility at the forefront of our minds. The methodologies used to conduct an investigation differ from practitioner to practitioner, however this two-part article aims to show some steps that should be considered which may substantiate or refute the Trojan defence.

Definitions

First, it is worth looking at the definition of a "Trojan" and how it relates to backdoors. According to Wikipedia⁷ "a Trojan horse or Trojan is a malicious program that is disguised as legitimate software ... Trojan horse programs **cannot replicate themselves**, in contrast to some other types of malware, like viruses or worms. A Trojan horse can be deliberately attached to otherwise useful software by a programmer, or it can be spread by tricking users into believing that it is a useful program." (*Emphasis added by author*).

A Trojan is simply a delivery mechanism. It contains a payload to be delivered elsewhere. The payload may consist of almost anything such as a piece of spyware, adware, a backdoor, implanted data or simply a routine contained within a batch file. Additional tools such as keyloggers, packet generation tools (for denial-of-service attacks) and sniffers may form part of the payload. It is beyond the scope of this article to discuss each of these in turn as we would simply not have enough space so we will instead concentrate on backdoors themselves as part of the overall Trojan debate.

The above properties are important to an analyst. Finding the original infection vector or artefacts relating to the Trojan could influence the timeline and validity of evidence. Locating the actual Trojan and understanding its payload and capabilities is exceptionally useful when building (or defending) a case.

Wikipedia explains, "A backdoor in a computer system (or a cryptosystem, or even in an algorithm) is a method of bypassing normal authentication or obtaining remote access to a computer, while intended to remain hidden to casual inspection. The backdoor may take the form of an installed program (e.g., Back Orifice) or could be a modification to a legitimate program. ... Many computer worms, such as Sobig and Mydoom, install a backdoor on the affected computer (generally a PC on broadband running insecure versions of Microsoft Windows and Microsoft Outlook). Such backdoors appear to be installed so that spammers can send junk email from the machines in question."⁸

⁶ <http://project.honeynet.org/scans/index.html>
⁷ http://en.wikipedia.org/wiki/Trojan_horse_%28computing%29 ⁸ <http://en.wikipedia.org/wiki/Backdoor>



Again, the properties of the backdoor can influence the case. For example, many investigators use an Anti Virus (AV) tool to process the forensic image. The AV tool will highlight files containing malware, including backdoors. However, the mere existence of the files does not necessarily mean that the backdoor was ever active. Establishing this fact can be crucial and will be revisited in part II of this article.

Trojan Making: Binders, Wrappers & Joiners

Remember that Trojans are delivery vehicles for some form of payload. But how are they made? What tools are available to do this? Many Trojans are created by Trojan-making kits which are often referred to as wrappers because they “wrap” the functionality of malicious software into other carrier software. The final innocent looking package is then distributed through whatever means the malware author deems appropriate be it to a mass audience, to targeted groups or direct to individuals. Typical distribution mechanisms are:

- P2P
- email
- file sharing & removable media
- direct implant through hacking etc.

The process for Trojan making has been around for a very long time and the kits are widely distributed and vary in quality and complexity. Traditionally, we saw tools that would allow an attacker to take their own preconfigured backdoor and then wrap it with an executable of their choice as per the following diagram:

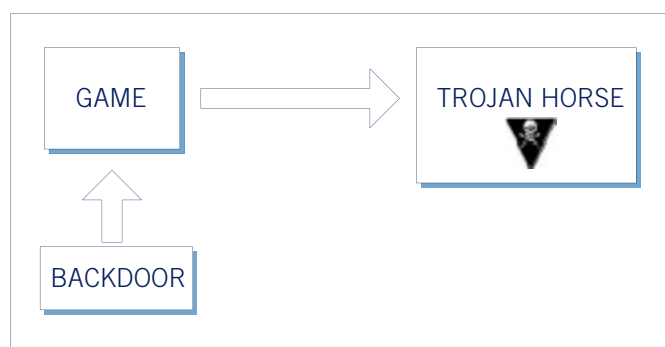


Figure 1: The process of binding a backdoor to a game

The kit (be it GUI or command-line driven) will usually give options of how to unpack each piece of software. Take for example, EliteWrap⁹. In the following example, the malware author can take a game called *new-game.exe* and some malicious payload called *malware.exe* and bind them together:

⁹<http://www.packetstormsecurity.org/trojans/elitewrap.zip> (originally from: <http://www.holodeck.f9.co.uk/elitewrap>)



```
Select C:\WINNT\System32\cmd.exe
C:\hinders>elitewrap
eLiteWrap 1.03 - (C) Tom "eLite" McIntyre
tom@dundeecake.demon.co.uk
http://www.dundeecake.demon.co.uk/elitewrap
Stub size: 7712 bytes
Enter name of output file: new-game.exe
Operations: 1 - Pack only
           2 - Pack and execute, visible, asynchronously
           3 - Pack and execute, hidden, asynchronously
           4 - Pack and execute, visible, synchronously
           5 - Pack and execute, hidden, synchronously
           6 - Execute only, visible, asynchronously
           7 - Execute only, hidden, asynchronously
           8 - Execute only, visible, synchronously
           9 - Execute only, hidden, synchronously
Enter package file #1: game.exe
Enter operation: 2
Enter command line: game.exe
Enter package file #2: malware.exe
Enter operation: 3
Enter command line: malware.exe
Enter package file #3:
All done :)
C:\hinders>
```

Figure 2: EliteWrap used to wrap two pieces of software together; the original game (game.exe) will unpack in the foreground when executed whilst malware.exe unpacks itself in a stealthy manner.

Over time this process has become very simple and wide-spread with graphical tools making the process extremely simple.

Changing Shape

The terms “packer” and “compressor” are often used interchangeably to describe utilities that essentially change the binary structure of a file by drawing out or compressing unnecessary space within a file. Simple examples of this are archive compression utilities such as WinZip or a UNIX equivalent of GZip.

Taking a well-known backdoor which is readily detectable by AV and compressing it using normal archive compression such as WinZip would still result in AV flagging the backdoor as found (this is because the vendors will hold a signature for that level of compression thus revealing a simple match). However, there are many other types of compression algorithm available which the attacker has at his/her disposal which when run on the normally detectable backdoor, will create a file with a new binary signature. The result is that it therefore becomes undetectable to AV until it is decompressed.

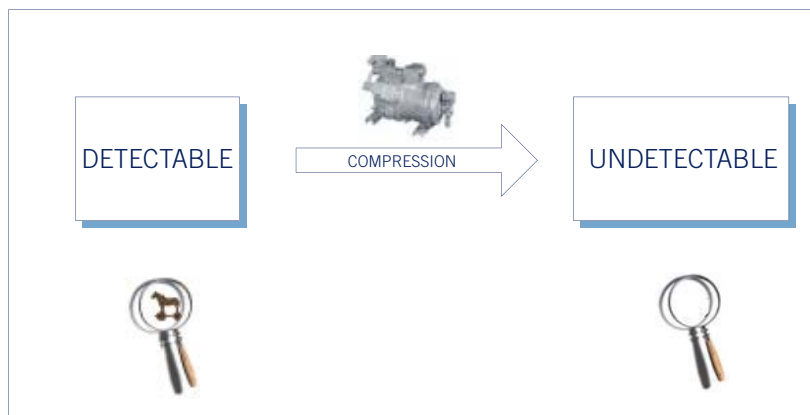


Figure 3: Compression used to make a backdoor unrecognisable to AV



Naturally there are some people who do not have the knowledge of how to run compression utilities directly. Instead they download kits (packers)¹⁰ many of which are driven by a simple GUI. They can then test to see whether their payloads will trigger typical AV engines and the pattern files within. Decompression would expose the malware to the AV engine, but this can be overcome by deploying an AV killer.

Anti-Virus & Personal-Firewall Killers

As the names of these tools suggest, they are designed to shutdown or disable the protection afforded by traditional AV and personal firewall software on the client/victim machine. They exist in several forms including standalone AV killers, standalone Firewall killers or combination tools that address both; for example “KILLer” by “illwill”.¹¹

So how do these tools influence our work as practitioners? The technology is always moving and the vendors are continually developing their AV and personal firewall software. Unfortunately, the hackers are undertaking field tests on how their “killers” work against the latest AV engine/pattern file etc.

If a victim does receive and inadvertently execute some malicious software which deploys an AV killer before launching the main payload, we are unlikely to see any events or logs alerting us to this incident. Furthermore, if a clean-up operation is subsequently performed and AV reinstated, we may not have enough recoverable evidence to refute some claims. So is AV good enough for us nowadays? We will address this issue in part II of this article.

Piecing Evidence Together

When considering the Trojan Defence and the issue of Trojans and backdoors in their entirety, it is important to understand what artefacts may be found in an investigation. Generally, we classify the classic backdoor/Trojan kit into three components:

- 1.Server** – the backdoor itself, often wrapped up into the overall Trojan; configured with specific options & may also include other helper modules termed plugins
- 2.Client** – used to control the backdoor from a remote location
- 3.Creation tool / kit** – used to configure the behaviour of the backdoor before it is released to the intended victim/s. Of course if we were to find anything other than the server part of the overall kit on a suspect’s machine then questions would need to be raised as to why a creation or remote control GUI was also present.



Trojan Scenarios

So what can the overall Trojan package do? What evidence would be left behind? Let us now take two scenarios which we will build upon in part II:

Scenario 1:

In this scenario, the victim has up-to-date AV present on their machine and downloads a game from a Peer-To-Peer network such as KaZaa. The game is in this case a Trojan horse designed to deliver a number of payloads including a backdoor as follows:

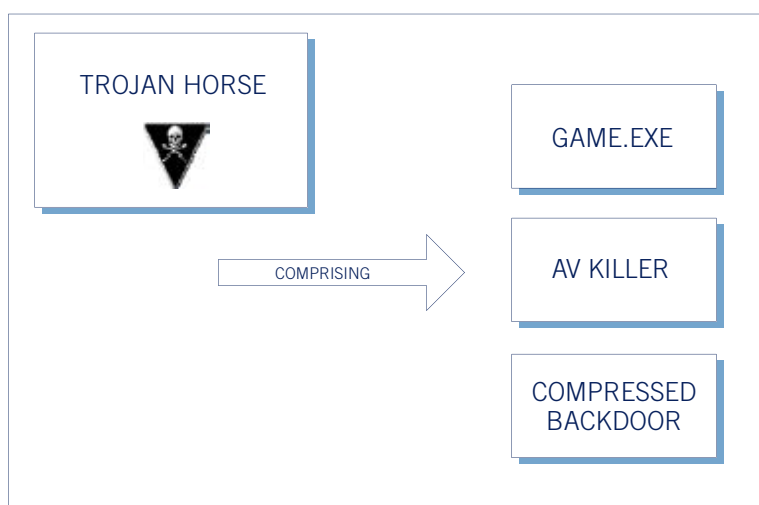


Figure 4: Trojan horse game containing an AV killing agent together with a compressed backdoor

You will note that the backdoor has been compressed so any AV engines (including that of the investigator) will not necessarily detect the backdoor payload. The problem for the attacker is however, that as soon as the backdoor is released and decompresses, this could possibly trigger an AV response. To combat this, the Trojan first delivers its AV killer designed to disable the AV engine. Once complete, the backdoor is then deployed and installs itself in stealth mode allowing the attacker access to the victim machine remotely. At the same time, the backdoor notifies its "owner" (the attacker) of its presence via email, establishing an outbound connection over a port which is likely to be open in the user's personal firewall settings (SMTP TCP /25). If we were to place a network sniffer between the victim's machine and the Internet, it would be likely that the notification output would be captured and could be analysed, (subject to no encryption being used by the backdoor).

Scenario 2:

If the above scenario was not bad enough then consider the same type of Trojan deployment, but also with a FW killer to disable personal firewall software and a routine that could implant false registry keys into the victim's system. Such keys could ensure stealthy start-up of rogue processes or could even add falsified histories relating to Internet surfing activity. The possibilities are numerous:



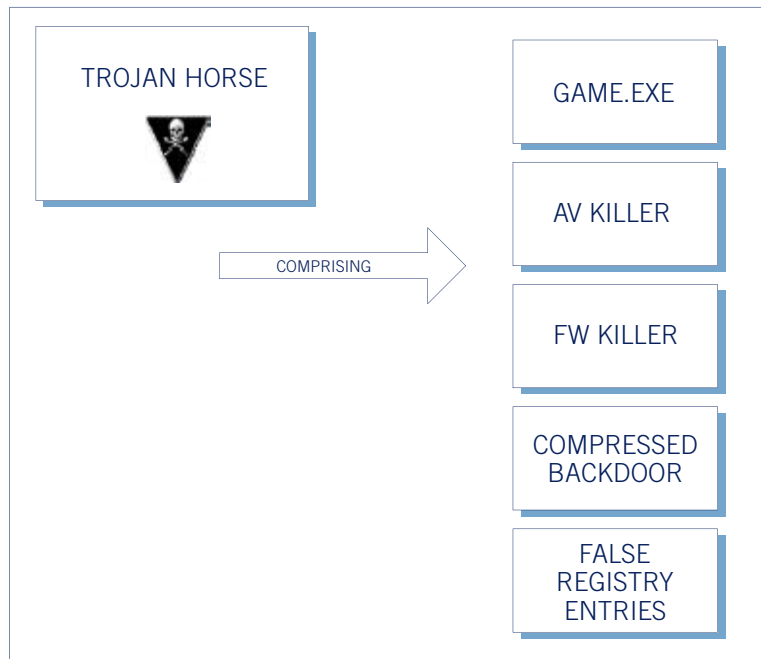


Figure 5: A more complex example of the potential payloads a forensic investigator may have to contend with when analysing a Trojan horse (or the traces of) found on a system

Whilst all this may seem rather complex and possibly too difficult to achieve, remember that tools have emerged that automate much of the above. We now see all-in-one kits such as OptixPro which makes the overall Trojan, configures an integral backdoor and has features such as AV killing:



Figure 6: Part of the configuration options within OptixPro

Considering Volatile Evidence

To date forensic practitioners have developed various methodologies for dealing with a computer crime scene which comply with various rules and best practices. One of the primary rules for processing a computer crime scene is “[to] Acquire the evidence without altering the original”¹². To this end, many forensic practitioners take the approach of ‘pulling the plug’ on a suspect computer. The rationale is that whilst volatile information such as running processes, network connections and data stored in memory is lost, the evidence on the hard disk should remain intact. Naturally, there are pro’s and con’s to every option – as we know, simply doing nothing still changes data and therefore the evidence.

12 Kruse, Warren G. and Jay G. Heiser. Computer Forensics: incident response essentials. Indianapolis: Addison-Wesley, 2002.



It is our belief that, whenever possible and especially considering a potential Trojan defence, volatile information should be gathered. Very often, this volatile data can be used to help an investigator during the offline investigation. A list of open network ports can help support or refute the presence of an active backdoor, memory often contains useful information such as decrypted applications or passwords, sometimes malicious code that has not been saved to disk and only runs from memory can be obtained (as in the case of the Code Red worm).

Network Evidence

Having a well-rehearsed plan for acquiring live evidence is critical. Using trusted and forensically sound tools is a must. Before gathering the evidence from the suspect system, it could be worth considering a network forensic approach by sniffing the communication flows to and from the suspect system. Unfortunately, this tends to be easier said than done – both from a legal and a technical perspective.

In some situations, such as a corporate environment or a home networked environment, it may be possible to intercept communication through the use of the 'port spanning' function of a switch. Plugging in to an existing hub or placing a hub between the suspect system and the network may also be an option. The investigator's machine would then be configured to capture all traffic too and from the suspect machine. It may be preferable to capture the raw packets using Linux and tcpdump, but various options exist, both free and commercial, for Windows and Linux.

In other situations, such as a home user with ADSL and a USB modem, it may be necessary to use a proprietary device such as the DSL Phantom™ by TraceSpan™¹³. This device is able to extract traffic and dump it via USB to the analysis machine.

Some of these techniques require that the connection be disrupted; in such cases, we usually obtain the volatile information from the computer before obtaining the network communications. The Regulation of Investigatory Powers Act 2000¹⁴ and The Telecommunications (Lawful Business Practice) (Interception of Communications) Regulations 2000¹⁵ govern the interception of communications.

It is recommended that you obtain legal advice regarding the interception of communications. Legally obtained information from a packet capture could significantly influence the investigation. The capture may provide evidence of a backdoor, an active compromise or it may show ongoing activities that enhance the case.

A Next Step: Volatile Information From a Live System

After obtaining the network captures, the next step involves gathering volatile information from the system. One tool that should be part of every responder's toolkit is the Windows Forensic Toolchest (WFT)¹⁶. This tool, written by Monty McDougal as part of his SANS GCFA¹⁷ certification, is designed to provide an automated incident response on a Windows system and to collect security-relevant information from the system in a forensically sound manner. Alas, this tool is only for Windows – unfortunately a similar tool for Linux systems doesn't appear to exist. WFT is an excellent incident response tool in that it provides a simplified way of scripting these responses using a sound methodology for data collection. While running individual tools is an option – using a scripted technique helps ensure consistency and eliminate mistakes.

13 http://www.tracespan.com/2_2LI%20Monitoring.html

14 <http://www.hmso.gov.uk/acts/acts2000/20000023.htm>

15 <http://www.hmso.gov.uk/si/si2000/20002699.htm>

16 <http://www.foolmoon.net/security/wft/>

17 http://www.giac.org/practical/GCFA/Monty_McDougal_GCFA.pdf



