Global Information Assurance Certification Paper

An in-depth look at W32.Winux and W2K.Stream and the ever growing "Proof of Concept" Virus

GSEC Practical Assignment Version 1.2b

Thomas A. Smit

Two of the most innovative viruses have come out of the group 29A. Both of these viruses were created or co-authored by a programmer that goes simply by the name "Benny".

In a world where the "Virus in a can" approach has dominated the spotlight, these two viruses are a throw-back to the days of "old-school" viruses (e.g. Monkey, Stoned). These viruses aren't spread automatically by spamming your address book. They can't be created by downloading an application, clicking a few radio buttons, and pressing the generate button.

Both viruses did absolutely nothing harmful the system infected. The technology behind the two viruses is so innovative though, that the two provide a springboard to launch other viruses off of.

This paper will attempt to explain both viruses and take an in-depth look into the technology behind them, and the inherent threat in the concept. We'll then take a look at basic "proof of concept" viruses, and discuss some startling historical examples of these type of viruses.

Winux: at a glance

Virus Name: W32.Winux, a.k.a. Linux.Winux

System(s) Affected: x86 Systems with WinX (Win9x, WinNT, 2K) and/or Linux (any variant, e.g. Redhat, Mandrake using the later ELF binaries, a.out binaries seem to not be affected)

Programmed Language: x86 Assembly

Files Affected: Windows Portable Executable (PE) files and Linux ELF files.

Author: Benny/29A

Severity: Low

Portability: High

W32.Winux: Overview

W32.Winux is a proof of concept virus that surfaced on March 27th, 2001. It is the first publicly known virus that can infect both Windows based and Linux based machines. It is written in an x86 based assembly program language, making it only affect x86 processor machines. It has been stipulated that it truthfully only infects Intel based x86 systems, leaving AMD systems uninfected, this is unconfirmed though.

The virus does nothing harmful to the system. Because of this, many people have coined this as a "proof of concept" virus, or a virus written to prove that it could be written.

The virus is broken into two parts, one for each platform that it is capable of affecting.

Stream: at a glance

Virus Name: W2K.Stream

System(s) Affected: x86 Systems with Windows 2000 with NTFS file systems, NT is not affected due to programming in the virus

Programmed Language: C/C++

Files Affected: Any file

Author: Benny & Ratter/29A

Severity: Low

Portability: High

W32.Winux: Win32

According to the readme that came with the virus source code: "Virus infects PE filez by overwritting .reloc section, so it does not enlarge host file size. Filez that don't have .reloc section, big enough for virus code, can't be infected (explorer.exe can be used to test infection capabilities). It can pass thru directory tree by well known "dotdot" method ("cd ..") and there infects all PE and ELF filez - virus does not check extensionz, it analyses victim's internal format and then decidez whata do. When all filez are passed and/or infected virus will execute host code." (Benny, http://benny29a.kgb.cz/)

```
1.      and     dword ptr [ebp + sucElf - gdelta],0
2.      test    [esi.WFD_dwFileAttributes], FILE_ATTRIBUTE_DIRECTORY
3.      jne     end_seh
4.      xor     ecx,ecx
5.      cmp     [esi.WFD_nFileSizeHigh],ecx
6.      jne     end_seh
7.      mov     eax,[esi.WFD_nFileSizeLow]
8.      cmp     eax,4000h
9.      jb      end_seh
10.     mov     [ebp + l_lseek - gdelta],eax
```

Figure 1

Figure 1 shows the first snippet of code that we'll investigate. These ten lines of code show the process of deciding which files to infect and which files to bypass. In line two(2) we see the author compare (test) the directory attribute of the file. If the current file being inspected is a directory, it is passed and not infected. This shows us one very important aspect of the virus. Due to this limitation of the program, the virus is unable to traverse the entire directory structure of your machine. For example, if the virus starts at "C:\test\viruses\bbb", it will infect files in the current directory, the parent directory, and it's subsequent parent directory until it reaches the root of the drive. However, if you have twenty directories under "C:\test" only the directory "\viruses\bbb" would be infected. The same holds true for the "C:\test\viruses" directory.

Lines four through nine of figure 1 filter out files larger than 4 gigabytes (defined by a procedure call to esi.WFD_nFileSizeHigh) and it also filters out files smaller than 16k (4000h).

```
1.      pushad
2.      mov     edi,[edi.SH_PointerToRawData]
3.      add     edi,[ebp + lpFile - gdelta]
4.      lea     esi,[ebp + Start - gdelta]
5.      mov     ecx,virus_end-Start
6.      rep     movsb
7.      popad
```

Figure 2

Figure 2 shows the only malicious activity of the virus. In the first line of figure 2, the program pushes all registers into the stack. Over the next five lines, it copies the entire virus to the .reloc section of the file.

W32.Winux: Linux

According to the readme that came with the virus source code:
"Virus infects ELF filez by overwritting host code by viral code. The original host code is stored at the end of host file. It can infect all filez (both of PE and ELF) in current directory, also without checking file extensionz. When all filez are passed and/or infected virus will restore host code (overwrite itself by original host code) and execute it." (Benny, http://benny29a.kgb.cz/)

```
1.      pushad
2.      mov     ecx,[esi+14h]
3.      mov     esi,[esi+10h]
4.      add     esi,edi
5.      push    esi
6.      xchg    eax,edi
7.      rep     movsb
8.      pop     edi
9.      lea     esi,[ebp + Start - gdelta]
10.     mov     ecx,virtual_end-Start
11.     rep     movsb
12.     popad
13.     add     dword ptr [edi+18h],LinuxStart-Start
14.     mov     [ebp + sucElf - gdelta],edi
15.     jmp     close_file
```

Figure 3

The Linux portion of this virus, not surprisingly, is considerably smaller in size, however, still accomplishes the end results of copying the virus into the entry point of the executable, and then placing the actual instructions for the executable at the end of the binary.

Once executed, the virus spreads and propagates itself, and then passes control to the actual instructions for the binary. In lines 1-11 in figure 3 above the virus is copying the original instructions into a buffer. It then copies the virus into the entry point of the binary, then finally copies the buffer back into the executable. "In a rather twisted mockery of open source spirit, the original virus code is then stored at the end of the ELF executable." (Delio)

W32.Winux: Scalability and Severity

Due to the rather benign nature of this virus, many anti-virus companies and

technology news companies have downplayed the importance and effect of this virus. And rightly so. This virus, as a separate entity, contains no threat to machines, except for possible causing a headache in cleaning.

The problems start to occur when you apply the current virus life cycle to this virus. Imagine five months from now, a tool that will generate a cross-platform virus that propagates itself through Exchange and/or Sendmail and is a lot more malicious than just replicating itself.

This virus is a typical "proof of concept" virus. This idea will be covered later in this paper.

W2K.Stream: Overview

(http://www.sans.org/infosecFAQ/win2000/streaming_virus.htm)

There is already quite a bit of information available on the SANS website dealing with the Stream virus, therefore, this section is going to be a general overview of the virus. At the same time, there has not been much study of this virus as a "proof of concept" virus.

Due to a coding oversight, W2K.Stream only affects Windows 2000, even though Windows NT has Alternate Data Streams (ADS). According to Microsoft, ADS was created to allow compatibility with MAC file systems, which stores data such as icons in the alternate data streams.

The virus copies the original executable into an ADS and then copies itself into the original executable. Then, when executed, it propagates itself, and then executes the alternate data stream.

W2K.Stream: Scalability and Severity

Again, due the non-malicious affect that this virus had, quite a few anti-virus companies and technology news companies downplayed the virus when it was first discovered. Just as with W32.Winux, the virus is not the threat, the concept behind it is.

As the paper by Ruth Parish (SANS) states, many anti-virus companies do not consider it a threat, simply because malicious code cannot be executed from the alternate data stream without being called from the primary application. However, according to the spring 2001 issue of Windows 2000 Magazine, it is possible to execute a VB Script or executable stored in an ADS by modifying the registry "Run" key under HKLM. Using a very basic WSH script to write to the registry key (which by default is read/write to everyone) and copying malicious code into an ADS, we have an instant virus that can't be detected by conventional anti-virus programs (as seen below).

```
1. Wsh.regWrite (HKLM\Software\Microsoft\Windows\CurrentVersion\Run\virus,
c:\winnt\notepad.exe:virus.vbs)
2. Wsh.Run (copy_virus_into_ADS.bat, 1)
{Run mail propagate scheme}
```

Figure 3

Proof of Concept Viruses: Overview

In 1960 when President Kennedy challenged the American people to make it to the moon by the end of the century, the first question on a lot of engineers' minds was: "How do we do that?". Instead of shooting straight for the moon, they performed a series of tests, or proof of concepts, to ensure a safe and successful mission to the moon. One such proof of concept was Senator John Glenn's historic mission into space. The main purpose of this mission was to prove a number of things, such as the ability to launch a man into space and retrieve him.

A proof of concept (PoC) virus is similar to this example, in that it is written specifically to prove something. PoC viruses in the past have been written to prove a number of things including Outlook propagation, alternate data stream capability, master boot record infection, and even cross platform compatibility. A PoC virus adheres to two very specific attributes which can be seen by studying historical PoC virus examples:

- Non-destructive - PoC viruses are, by design, usually not code with malicious intent. The virus writer is simply trying to infect the machine, and in the process, concentrates his energy on the actual infection and propagation subroutines of the virus. The only effect to the end user is usually cleansing headaches and anti-viral software upgrades.
- Innovative - PoC viruses have to be innovative. The virus has to take advantage of an exploit that no other virus has ever taken advantage of, or in such a way that no one ever thought before.

Proof of Concept: A brief History

Since most of these viruses are covered in depth in multiple areas including magazine literature and the SANS Institute website, the discussion of these viruses will be contained to the model spoken of above.

- Melissa - most anti-virus companies didn't feel that Melissa.VBS was a proof of concept virus. However, if we analyze it closely, it does fit into the model of a PoC virus. It was not originally coded maliciously, it's only real intent was to propagate itself through the VB capabilities of Outlook and Windows. It was deemed malicious only after it clogged mail servers over the entire net. It was also innovative. It was the first large-scale virus of it's kind to propagate through Exchange by using VB script. However, most present day viruses are modeled after Melissa, adding a destructive element to an email-attachment propagation scheme.
- BubbleBoy - named for an episode of Seinfeld, this virus was a non-destructive PoC virus that propagated itself through email without user intervention. It takes advantage of the HTML rendering in Outlook/Express clients and propagates itself. Like Melissa, BubbleBoy was not malicious, but it also had the ability to bring mail servers to their knees. This virus was innovative due to the fact that it could propagate itself without user intervention, just by having been viewed, this email virus could infect your system and propagate itself on using the global

address book.

- Stream - discussed above, this virus fits into the model of a PoC virus perfectly. It wasn't coded with malicious intent and was the first virus seen to employ alternate data stream technology.
- Winux - discussed above, this virus fits into the model of a PoC virus also. W32.Winux doesn't cause damage to the infected machine and it is the first virus seen that made it possible to infect cross platform operating systems.

Bibliography

"Benny" "Benny's Website", URL
http://benny29a.kgb.cz/

Central Command "Discovery of the first Cross Platform (Linux/Windows) Virus", 27, MAR, 2001 URL
http://www.avx.com/winux.html

CNN "Virus that infects Windows and Linux identified" 28, Mar, 2001, URL
http://www.cnn.com/2001/TECH/internet/03/28/virus.winux.02/

Delio, Michelle "A Virus that Leaps Platforms" 27, MAR, 2001, URL
http://www.wired.com/news/technology/0,1282,42672,00.html

Parish, Ruth "Windows 2000 Streaming Virus" 20, Nov, 2000 URL
http://www.sans.org/infosecFAQ/win2000/streaming_virus.htm

Pearson, Daniel "New Windows/Linux virus no big threat to Linux users" 28, MAR, 2001 URL
http://www.newsforge.com/article.pl?sid=01/03/28/0532200&mode=nocomment

Reuters "Emergent virus can infect Windows, Linux" 27, MAR, 2001 URL
http://news.cnet.com/news/0-1003-200-5329436.html

Sullivan, Bob "BubbleBoy virus found on Net" 12, Nov, 99 URL
http://www.zdnet.com/zdnn/stories/news/0,4586,2392757,00.html

Zenkin, Dennis & Kaspersky, Eugene. "NTFS Alternate Data Streams." Windows 2000 Magazine Spring 2001 (2001): 45 – 48.