It rather involved being on the other side of this airtight hatchway: Executable corruption

devblogs.microsoft.com/oldnewthing/20070807-00

August 7, 2007

Raymond Chen

In the category of dubious vulnerability, I submit the following (paraphrased) report:

> I discovered that if I take an EXE file and corrupt its header, then when I try to run the EXE file, the process starts up and then crashes. I used the information in the crash dialog to direct further investigations, noting that the specific crash location could be controlled by modifying particular bytes in the EXE. Finally, I was able to put all the details together to form an exploit: I modified a block of bytes in the EXE file to consist of code which opens a network socket and connects it to a command shell, then modified the header to point to those bytes. When I run the EXE, the exploit code runs, and I can connect to the network socket from another computer and control the command shell.

Yeah, that's great, but what's the vulnerability? What you did was take a program that you have write permission to and change the code in it to run your exploit. If you can modify an EXE file, then you may as well just replace the entire contents of the file with the bytes of `PWNZ0RD.EXE`. In other words, modifying bytes here and there is just a very slow, inefficient, and unnecessarily complicated way of doing this:

```
copy pwnz0rd.exe victim.exe
```

Then when the user runs the infected program, they're really running the `PWNZ0RD.EXE` program, and your so-called exploit can do whatever it wants. That's a lot easier than trying to modify a dozen bytes here, a dozen bytes there.

In order to trigger the vulnerability, the user has to run the compromised program, but a program is already arbitrary code. No need to be so sneaky about it. It's sort of a tautology: "Here's my clever way to get the user to run my code. Step 1: Write some code. Step 2: Get the user to run it."

Of course, if this corrupted EXE file created other types of problems, such as crashing Explorer or triggering a buffer overflow when the user tried to view its properties, then you'd be onto something. Or if you could somehow avoid detection by not altering the digital

signature, then that'd be interesting as well. But if the only way to trigger code injection is to run the injected code, then that's not really all that interesting. You just found a roundabout way of creating a Trojan horse.

Raymond Chen

Follow