Blaming the operating system for allowing people to create files with unusual characters in their names

devblogs.microsoft.com/oldnewthing/20200617-00

June 17, 2020

Raymond Chen

A security vulnerability report came in that claimed that Windows had a remote code execution vulnerability because it allows malicious characters in file names.

Specifically, they noted that if you created a Web server with an image upload page, and if an uploaded file contained a special character like an ampersand, then you could obtain remote code execution.

They included a complicated server-side script, but here's the gist:

```
void OnFileUploaded(string path)
{
 system("dosomething.exe " + path);
}
```

A user could upload files with odd names like

```
x & logoff &.jpg
x & certutil -installcert -f badcert.jpg
x & sc stop server &.jpg
```

The resulting commands passed to the `system` function are

```
dosomething.exe x & logoff &.jpg
dosomething.exe x & certutil -installcert -f badcert.jpg
dosomething.exe x & sc stop server &.jpg
```

The ampersand is a `CMD.EXE` metacharacter for combining multiple commands into a single line, so each of the above lines is treated as multiple commands. The first is `dosomething.exe x`, which presumably fails because there is no file named `x`. The second is something dangerous. And sometimes there's a third command `.jpg` which is probably going to fail with `'.jpg' is not recognized as an internal or external command, operable program or batch file.`

The finders claimed that this proves that ampersands in file names are a remote code execution vulnerability in Windows.

What we have here is a case of creating an insecure system and then being surprised that the system is insecure. The `OnFileUploaded` function passes untrusted data directly to the `system` function without any attempt to sanitize it first. This is a classic injection attack (obXKCD), and it is the responsibility of code that builds commands out of strings to be alert to issues like this.

You'd be best advised to avoid things that do their own level of nontrivial parsing over the generated command line. SQL commands and command lines are examples of this. In the above example, it would be better to run the `dosomething.exe` program directly and pass the file name (suitably quoted), to avoid any funny business with `CMD.EXE` command line parsing.[1]

When we informed the finders that there was no Windows vulnerability here, they insisted that the issue they found is a critical remote code execution vulnerability, shared a paper they planned on presenting, and warned that publication could likely result in widespread attacks within hours of disclosure, resulting in compromised Windows systems around the globe.

I reviewed the paper and verified that it didn't do anything beyond what they had described in their original report. We advised them that the vulnerability was in their Web server, not in Windows, and gave them permission to disclose.

As far as I can tell, they never did publish that paper anywhere.

Bonus chatter: They claimed that the issue could be fixed by simply adding the ampersand to the list of illegal file name characters. They forgot about the percent sign (for injecting environment variables), the caret (for escaping), and possibly even the apostrophe. They may also want to file vulnerabilty reports against unix since it is vulnerable to the exact same problem: If somebody takes an untrusted file name and injects into into a command passed to the `system` function, bad things can happen. It's even worse on unix, because unix has almost no forbidden file name characters. Even a newline is a legal filename character! So go ahead, put a bitcoin miner in your file name, let the server do some real work for you.

[1] If you don't trust yourself to quote potentially-malicious file names safely (and I don't blame you), you could pass the file name some other way entirely, like via stdin or by putting the dangerous file name in a file, and passing that file as an indirect parameter: `something.exe @indirect`.

Raymond Chen

Follow