PowerShell Threats Grow Further and Operate in Plain Sight

To get a better understanding of today's landscape, we analyzed morethan 115,000 randomly selected malicious PowerShell command lines that we saw in 2018. Many of them came from Microsoft Office documents or self-propagating worms.

The first thing that caught our eye was that, even though there are many obfuscation tricks available for PowerShell and even automated tools that can obfuscate scripts for users, these are rarely used in the wild. Only four percent of the PowerShell command lines we analyzed tried toobfuscate themselves by using a mixture of lower- and uppercase letters.

Many attackers probably are not too concerned about obfuscation because they know that not everybody is monitoring PowerShell activity anyway. Even if the security personnel do, then a non-obfuscated command line might be more likely to slip unnoticed through the cracks. Too much obfuscation can be a red flag.

Of course, another option for the attacker is to use a BASE64-encoded blob to hide their commands, resulting in an extra step of decoding required, before the payload can be seen. This is often done by all kinds of scripts, including benign ones.

From all analyzed scripts, 5.7 percent used a BASE64-encoded string to hide their commands. With 80 percent of this subset of scripts, most simply used the built-in "-EncodedCommand" command line argument or a variation thereof to decode and run their commands.

The most common command line parameters we observed are still "-NoP -NonI -W Hidden". Although the arguments can be written in different ways, most attackers stick to the short version.

Command line argument	Percentage of use
NoProfile/NoP	77.9%
Window hidden/W hidden	78.9%
Noninteractive/NonI	76.6%
ExecutionPolicy bypass	10.7%

Table. Percentage of use of specific command line arguments

One of the main goals of most malicious PowerShell scripts is still to download and execute some remote payload. From all samples analyzed, 17 percent downloaded something through HTTP or HTTPS. The scripts are getting more robust and often try multiple URLs, use the local proxy settings, or set a specific user agent in order to succeed.