ForeScout Secure Connector Local Privilege Escalation

Jordan Potti · March 30, 2021

red team

Application: ForeScout CounterACT Secure Connector
Operating System tested on: Windows 10 1809 (x64)
Vulnerability: ForeScout CounterACT SecureConnector Local Privilege Escalation through Insecure Folder Permissions

Overview:
This vulnerability exists due to the permissions set on the logs directory used by the ForeScout SecureConnector application. Every several seconds, a new log entry is placed into `c:\ProgramData\ForeScout SecureConnector\Logs\sc.log`. The Logs directory, as well as the file `sc.log`, allows Everyone Full Control.

Due to this, a low privileged user can create a symbolic link in the sc.log location, and point it at a privileged location such as `c:\Windows\System32`. The log entries will be created in a file at the receiving end of the symbolic link. By setting the receiving end of the symbolic link as a valid DLL in that location, the DLL is overwritten with the log file, and the permissions of the "log" file allows Everyone Full Control. At that point, the DLL can be overwritten with a malicious DLL to gain privileged code execution.

Walkthrough:
While searching for file operation vulnerabilities, I came across the ForeScout SecureConnector making a `CreateFile` call as `NT AUTHORITY\SYSTEM`, in the ProgramData directory. This was discovered using Process Monitor.

In order to determine if this was a vulnerability, I used PowerShell's `get-acl` function to determine the file permissions of the parent directory. As shown below, the `Everyone` group has `FullControl`.

In order to exploit this as a low privileged user, we first need to create an NTFS junction pointing to a writable Object Manager directory. `\RPC Control\` is commonly used since it is writable by everyone. In order to create a directory junction, the source directory must be empty. For some reason, I couldn't create the junction even after emptying the Logs directory. By deleting the Logs directory all together, the junction was finally able to be created.

Using `mklink /J`, we are able to create our junction pointing to RPC Control.

Once the junction is created, it's now possible to create a symbolic link pointing to our target directory with the file name of our choosing.

I used CreateSymLink.exe from from James Forshaws Symbolic Link Testing Tools repo. For the target, I used `ualapi.dll` (From https://enigma0x3.net/2019/07/24/cve-2019-13382-privilege-escalation-in-snagit/).

As we can now see, the `ualapi.dll` file located in `C:\Windows\system32` is now being populated with the log entries from ForeScout.

This alone still isn't enough to do anything beside possible a DOS, we need to now replace `ualapi.dll` with a malicious DLL. The file `ualapi.dll` in this case does not inherit the permissions from parent directory (system32), but instead, inherits the permission from the source, in this case, our symlink which was created by our low privileged user.

`ualapi.dll` is used by the Spooler service, and although it typically isn't located in `c:\windows\system32`, due to the DLL search order, `c:\windows\system32` will be checked before finding the actual ualapi.dll, therefore executing our malicious DLL.

Currently, the malicious DLL is simply a log file, we need to replace it with something of our own. Since the permissions allow for that, we can copy over an actual malicious DLL.

We can either reboot, or wait for the spooler service to restart in order to validate the DLL execution. The DLL I placed there echoed the current user to C:\, as shown below, we now have code execution as System.

This vulnerability was fixed by ForeScout on November 2020 in CounterACT version 8.1.4.