



malcomvetter / CSExec Public

Notifications Fork 62 Star 316

<> Code Issues 1 Pull requests Actions Projects Security Insights

master

Go to file

<> Code

File list

csexec

csexecsvc

.gitignore

README.md

csexec.sln

screen.png

README

CSExec (a C Sharp psexec implementation)

This is an example for how to implement `psexec` (from SysInternals Suite) functionality, but in open source C#. This does not implement all of the psexec functionality, but it does implement the equivalent functionality to running: `psexec -s \\target-host cmd.exe`

About

An implementation of PSExec in C#

Readme

Activity

316 stars

8 watching

62 forks

Report repository

Releases

No releases published

Packages

No packages published

Contributors 2

Languages

```
C:\Users\admin\Desktop>ping dc0

Pinging DC0 [fe80::b84e:5267:14de:3786%2] with 32 bytes of data:
Reply from fe80::b84e:5267:14de:3786%2: time=1ms
Reply from fe80::b84e:5267:14de:3786%2: time<1ms

Ping statistics for fe80::b84e:5267:14de:3786%2:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms
Control-C
^C
C:\Users\admin\Desktop>csexec.exe \\dc0
[*] hostname: \\dc0
[*] Found .NET version: v4.0
[*] Choosing net40
[*] Installed net40 Service on \\dc0
[*] Service Started on \\dc0
csexec> whoami
nt authority\system

csexec> echo %computername%
DC0

csexec>
```

● C# 100.0%

`psexec` works by doing the following steps:

- copy a windows service executable (`psexecsvc.exe`) that is embedded within the `psexec.exe` binary to `\\target-host\admin$\system32`
- remotely connect to the service control manager on `\\target-host` to install and start the `psexecsvc.exe` service
- connect to the named pipe on the target host: `\\target-host\pipe\psexecsvc`
- send commands to the `psexecsvc` via the named pipe
- receive output via the `psexecsvc` named pipe
- upon exit, uninstall service, delete service executable

This project `csexec` mimicks those steps in native C# with only a minimal amount of `pinvoke` for the remote service installation. It's actually surprisingly simple and takes a very minimal amount of code to implement.

The primary difference between this and `psexec` is that it must determine the .NET runtime on the remote host in order to install the correctly compiled service executable.

Build in Visual Studio to create .NET 3.5, 4.0, and 4.5
executables for your client preference (Win 7 - Win 10+)

[Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

