Post-build obfuscation is something that can be done on tools to remove known AV signatures. ConfuserEx (and its various forks) is a popular tool for obfuscating .NET assemblies. Download the release package (ConfuserEx_bin.zip) to your TeamCity machine and extract it to `C:\ConfuserEx`.

ConfuserEx templates are based on XML. For simplicity, let's use the "aggressive" pre-set.

```
<project baseDir="<path>" outputDir="<path>" xmlns="http://confuser.codeplex.com">
    <rule pattern="true" preset="aggressive" inherit="false" />
    <packer id="compressor" />
    <module path="<target>.exe" />
</project>
```

There are several variables we need to fill in - the base directory, output directory and module path.

The base directory should be the final `bin\Release` directory where the original compiled `Rubeus.exe` will be. If we're happy for the confused assembly to replace the original, the output directory can be the same. The module path needs to be the filename of the assembly to confuse, in this case, `Rubeus.exe`.

We could hardcode these values or we can make it more flexible by leveraging TeamCity's build parameters.

The two that we can use here are `%teamcity.build.checkoutDir%` and `%system.teamcity.projectName%`.

The former will give us the temporary checkout directory of the code, something like `C:\TeamCity\buildAgent\work\9ea9d73f7fc269a0` (where the last portion is random) and the later will give us `Rubeus` since this is the project name.

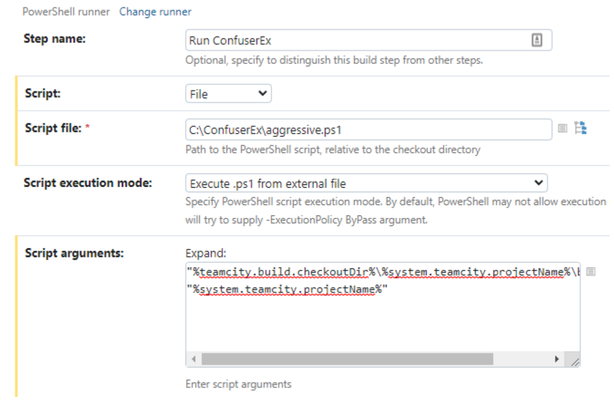
Open PowerShell ISE (or any text editor) on the TeamCity VM and paste the following:

```
param
(
[Parameter(Mandatory=$true)] [String]$path,
[Parameter(Mandatory=$true)] [String]$projectName
)

$template = @"
<project baseDir="$path" outputDir="$path" xmlns="http://confuser.codeplex.com">
  <rule pattern="true" preset="aggressive" inherit="false" />
  <packer id="compressor" />
  <module path="$projectName.exe" />
</project>
"@

$template | Out-File -FilePath "$path\$projectName.crproj"
C:\ConfuserEx\Confuser.CLI.exe -n "$path\$projectName.crproj"
```

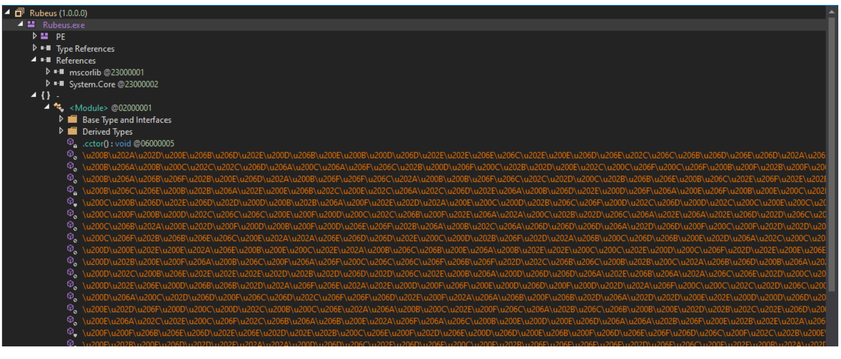
We define `$path` and `$projectName` as parameters that will be passed in by TeamCity. We set those parameters inside the XML template and then write it to a `.crproj` file. Then finally we call `Confuser.CLI.exe`. Save this to `C:\ConfuserEx\aggressive.ps1`.

Back in the TeamCity UI, add a new build step to the project and select the PowerShell runner again. Ensure **File** is selected next to the **Script** dropdown and enter the **Script file** path. Then under Script arguments, add:

```
"%teamcity.build.checkoutDir%\%system.teamcity.projectName%\bin\Release"
"%system.teamcity.projectName%"
```



Run the build and inspect the artifact with a .NET decompiler like dnSpy or dotPeek to see the result.



Having this work as an external PowerShell script makes it very easy to re-use across multiple projects. You can also have multiple template definitions and just call whichever one you want to use in your build.

You can also scan the final assembly with a tool like ThreatCheck to see that AV detections are now gone.

```
PS C:\Tools\ThreatCheck\ThreatCheck\ThreatCheck\bin\Debug> .\ThreatCheck.exe -f C:\Users\Daniel\Downloads\Rubeus.exe -
e Defender
[+] No threat found!
[*] Run time: 1.66s

PS C:\Tools\ThreatCheck\ThreatCheck\ThreatCheck\bin\Debug> .\ThreatCheck.exe -f C:\Users\Daniel\Downloads\Rubeus.exe -
e AMSI
[+] No threat found!
[*] Run time: 3.95s
```