



Home

About



# Blocking Visual Studio Code embedded reverse shell before it's too late

📅 Sep 17, 2023 · 11 min read

• [BLUETEAM](#)

[DEFENSE](#)

[WINDOWS](#)

[SPLUNK](#)

[SOC](#)

[POWERSHELL](#)



## Overview

- [Introduction](#)
- [Mitigation](#)
  - [Domains blacklist](#)
  - [Applocker](#)
  - [GPO](#)
- [Detection](#)
  - [Process](#)

Search...

ipfyx

Yet another  
cybersecurity blog

[READ MORE](#)

## Featured Posts

- [Blocking Visual Studio Code embedded reverse shell before it's too late](#)

## Recent Posts

- [D LINK DNS323 Fan Speed Control](#)
- [Bluekeep, why](#)

We would like to use [third party code](#) to improve the functionality of this website.

[Manage preferences](#)

[Deny](#)

[Allow](#)



Home

About



# Visual studio code tunnel

## Introduction

Since July 2023, Microsoft is offering the perfect reverse shell, embedded inside Visual Studio Code, a widely used development tool. With just a few clicks, any user with a github account can share their visual studio desktop on the web. VS code tunnel is almost considered a [lolbin \(Living Of the Land Binary\)](#).

I am so glad that my users now have the ability to expose their computer with highly sensitive data right on the web, through an authentication I nor control, nor supervise. My internal network is now accessible from anywhere !

The worse part is that this tunnel can be triggered [from the cmdline](#) with the [portable version of code.exe](#). An attacker just has to upload the binary, which won't be detected by any anti-virus since it is legitimate and signed windows binary.

It is therefore something to watch out for.

Execute it, open the link mentioned, log in to your github account and there is your reverse shell.

DEFENSE 2

ADMIN 1

CYBERSECURITY 1

## Tags

POWERSHELL 2

WINDOWS 2

BASH 1

BLUEKEEP 1

BLUETEAM 1

CVE 1

DEFENSE 1

LINUX 1

RDP 1

SOC 1

SPLUNK 1

WSUS 1

POWERSHELL

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



the Microsoft Privacy Statement (<https://privacy.microsoft.com>)

\*

The binary is definitely signed :

```
POWERSHELL
Get-AppLockerFileInformation 'C:\Users\ipfyx\AppData\Local\Programs\Microsoft VS Code\Code.exe'
PublisherName      : O=MICROSOFT CORPORATION, L=REDMOND, S=WASHINGTON, C=USA
ProductName        :
BinaryName         :
BinaryVersion      : 0.0.0.0
HasPublisherName   : True
HasProductName     : False
HasBinaryName      : False
```

Notice how there are nor ProductName nor BinaryName value, we will use them later.

[pfiatde blog post](#) already did a great job at describing what an attacker can do so I won't say much more. Go check his awesome blog post.

## Mitigation

But now, as a defender, how do we block or detect its usage ?

We would like to use [third party code](#) to improve the functionality of this website.



Home About



If you're part of an organization who wants to control access to Remote Tunnels, you can do so by allowing or denying access to the domain `global.rel.tunnels.api.visualstudio.com`.

Yeah it's not that simple. It will indeed block most users but, from my testing, it won't block an attacker that has already established a tunnel once. Meaning the tunnel can be kept active or restarted at will despite this domain being blacklisted.

From my understanding, VScode contacts

`global.rel.tunnels.api.visualstudio.com` to get its "clusters" :

<https://global.rel.tunnels.api.visualstudio.com/api/v1/clusters>

Here is a sample of the resulting json :

```
JSON
[
  {
    "clusterId": "auc1",
    "uri": "https://auc1.rel.tunnels.api.visualstudio.com",
    "azureLocation": "AustraliaCentral"
  },
  {
    "clusterId": "aue",
    "uri": "https://aue.rel.tunnels.api.visualstudio.com",
    "azureLocation": "AustraliaEast"
  },
  ...
]
```

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



```
- Dev Tunnels
- global.rel.tunnels.api.visualstudio.com
- [clusterId].rel.tunnels.api.visualstudio.com
- [clusterId]-data.rel.tunnels.api.visualstudio.com
- *.[clusterId].devtunnels.ms
- *.devtunnels.ms
```

Blocking those domains will block and cut off any VS code tunnel :

```
*.tunnels.api.visualstudio.com
*.devtunnels.ms
```

## Applocker

Applocker is Microsoft application whitelisting technology. When enabled and configured, with default rules for example, everything is blocked by default except for the executables and scripts defined in the preceding rules. Let's pretend we use Microsoft default rules for demo purpose.

To generate them, in *Group Policy Management Editor*, go to *Computer Configuration -> Windows Settings -> Security Settings -> Application Control Policies -> AppLocker*. Right Click on *Create Default Rules*.

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



Here is the resulting xml :

```
XML
<AppLockerPolicy Version="1">
  <RuleCollection Type="Appx" EnforcementMode="Enabled" />
  <RuleCollection Type="Dll" EnforcementMode="NotConfigured" />
  <RuleCollection Type="Exe" EnforcementMode="Enabled">
    <FilePathRule Id="921cc481-6e17-4653-8f75-050b80acca20" Name="
      <Conditions>
        <FilePathCondition Path="%PROGRAMFILES%*" />
      </Conditions>
    </FilePathRule>
    <FilePathRule Id="a61c8b2c-a319-4cd0-9690-d2177cad7b51" Name="
    ...
```

When using those rules, everything that is in Program Files and Windows is allowed, everything else is blocked. Administrators are allowed to execute anything, VSCode for instance. But if you were to define a rule that blocks VSCode, this rule would apply first, since "Deny rules" are applied before "Allowed rules".

Let's build a rule to block VScode altogether. The rule applies to Everyone (SID S-1-1-0), therefore to Administrators (SID S-1-5-32-544).

```
POWERSHELL
Get-AppLockerFileInformation 'C:\Users\ipfyx\AppData\Local\Progr
```

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



BinaryVersionRange to match any vscode version.

Warning : Do not import the following xml in Applocker.  
Imported alone, this rule will completely block your system.  
Add a Allow \* rule beforehand.

```
XML
<AppLockerPolicy Version="1">
  <RuleCollection Type="Exe" EnforcementMode="NotConfigured">
    <FilePublisherRule Id="1dd70b30-eb06-4220-b808-1"
      <Conditions>
        <FilePublisherCondition Publish
          <BinaryVersionRange Low
        </FilePublisherCondition>
      </Conditions>
    </FilePublisherRule>
  </RuleCollection>
  ...
```

Included in MS default rules, here is the resulting xml (called  
vscode.xml from now on) :

```
XML
<AppLockerPolicy Version="1">
  <RuleCollection Type="Appx" EnforcementMode="Enabled" />
  <RuleCollection Type="Dll" EnforcementMode="NotConfigured" />
  <RuleCollection Type="Exe" EnforcementMode="Enabled">
    <FilePathRule Id="921cc481-6e17-4653-8f75-050b8036c320" Name
```

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



Unfortunately, this rule won't apply to code tunnel because this binary doesn't have a product name (thanks Microsoft !):

```
POWERSHELL
Get-AppLockerFileInformation 'C:\Users\ipfyx\AppData\Local\Programs\Microsoft VS Code\Code.exe'
New-AppLockerPolicy : Les règles ne peuvent pas être créées. Le chemin d'accès spécifié est introuvable.
%OSDRIVE%\USERS\ipfyx\DOWNLOADS\VSCODE_CLI_WIN32_X64_CLI\CODE.EXE
```

To be sure, let's test the previously generated vscode rule.

```
POWERSHELL
# Portable code tunnel is blocked but not by the rule
Test-AppLockerPolicy -XmlPolicy .\vscode.xml -Path .\code.exe -FilePath C:\Users\ipfyx\Downloads\vscode_cli_win32_x64_cli.exe
PolicyDecision : DeniedByDefault
MatchingRule :

# Code tunnel is blocked but not by the rule
Test-AppLockerPolicy -XmlPolicy .\vscode.xml -Path 'C:\Users\ipfyx\AppData\Local\Programs\Microsoft VS Code\Code.exe'
FilePath : C:\Users\ipfyx\AppData\Local\Programs\Microsoft VS Code\Code.exe
PolicyDecision : DeniedByDefault
...
```

The *PolicyDecision* is *DeniedByDefault*, which means the rule we just built from the vscode binary does not apply to code-tunnel. If no rules matches, the default behaviour for applocker is to deny any execution. Code-tunnel is denied there, but it would not be if it was

We would like to use [third party code](#) to improve the functionality of this website.





Home

About



POWERSHELL

```
# Export
Get-AppLockerFileInformation .\code.exe | New-AppLockerPolicy -I

# Code tunnel is blocked
Test-AppLockerPolicy -XmlPolicy .\vscode-tunnel-hash.xml -Path
FilePath           : C:\Users\ipfyr\AppData\Local\Programs\Microsof
PolicyDecision      : Denied
MatchingRule        : code.exe
```

Warning : Do not import the following xml in Applocker.  
Imported alone, this rule will completely block your system.  
Add a Allow \* rule beforehand.

XML

```
<AppLockerPolicy Version="1">
  <RuleCollection Type="Exe" EnforcementMode="NotConfigur
    <FileHashRule Id="15b4ef38-5f18-484b-bc83-e03d2
      <Conditions>
        <FileHashCondition>
          <FileHash Type="SHA256"
        </FileHashCondition>
      </Conditions>
    </FileHashRule>
  </RuleCollection>
```

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



Visual Studio has the so wanted features :

- Dev Tunnels - controls test functionality

But as of today (1.8.82), VScode doesn't. You can force automatic updates though ! UpdateMode\_default would be my goto.

POWERSHELL

```
gc 'C:\Users\ipfyr\AppData\Local\Programs\Microsoft VS Code\pol:
```

XML

```
<?xml version="1.0" encoding="utf-8"?>
<policyDefinitionResources revision="1.0" schemaVersion="1.0">
  <displayName />
  <description />
  <resources>
    <stringTable>
      <string id="Application">Visual Studio (
      <string id="Supported_1_67">Visual Stud:
      <string id="Category_updateConfiguration
      <string id="UpdateMode">UpdateMode</str:
```

...

## Detection

Privacy Policy

We would like to use third party code to improve the functionality of this website.



```
index=win sourcetype="XmlWinEventLog" EventCode=4688 code tunnel  
| stats min(_time) as time_min max(_time) as time_max count as c
```

| host        | SubjectUser | cmdline  |
|-------------|-------------|--|
| sauteTomate | ipfyx       | "c:\Users\ipfyx\AppData\Local\Microsoft\VS Code\bin\code-tunnel" tunnel license-terms --name totallyNo |

But the binary *code.exe* could be renamed. However, what an attacker can't change are the *tunnel* and *--accept-server-license-terms* options :

```
index=win sourcetype="XmlWinEventLog" EventCode=4688 tunnel accept  
| stats min(_time) as time_min max(_time) as time_max count as c
```

| host        | SubjectUser | cmdline  |
|-------------|-------------|--|
| sauteTomate | ipfyx       | "c:\Users\ipfyx\AppData\Local\Microsoft\VS Code\bin\c0de.exe" tunnel license-terms --name totallyNo    |
| sauteTomate | ipfyx       | "c:\Users\ipfyx\AppData\Local\Microsoft\VS Code\bin\code-tunnel" tunnel license-terms --name totallyNo |

We would like to use third party code to improve the functionality of this website.



Home

About



## Looking for suspicious child process

Great idea from [lobas](#) :

```
IOC: Process tree: code.exe -> cmd.exe -> node.exe -> winpty-ag
```

A straightforward search would be :

```
index=win sourcetype="XmlWinEventLog" EventCode=4688 code (cmd |  
| table _time, host, NewProcess, ParentProcess
```

But once again, *code.exe* could be renamed

We could use the search above to look for suspicious child process  
from our process with *accept-server-license-terms* option :

```
index=win sourcetype="XmlWinEventLog" EventCode=4688 (cmd OR poi  
[search index=win sourcetype="XmlWinEventLog" EventCode=4688  
| table host NewProcessId  
| rename NewProcessId as ParentProcessId  
| format]  
| table _time, host, NewProcess, ParentProcess
```

We would like to use [third party code](#) to improve the functionality of this website.



|                   |             |                               |
|-------------------|-------------|-------------------------------|
| 09-18<br>11:36:21 | sauteTomate | C:\Windows\System32\WindowsPo |
|-------------------|-------------|-------------------------------|

EDIT 20230924 : license terms agreement is not mandatory.  
And even if it were, this could be bypass by creating the file `license_content.json` with the right content (see below). All we have left is detecting the string `"*.exe*tunnel"` in a commandline, which is subject to false positive.

## Applocker

EDIT 20230924

Since we defined some Applocker rules, we got some log from it !  
We could just search for that weird publisher value :

```
index=win sourcetype="XmlWinEventLog" EventCode=8004 event_prov:  
| table _time, host, FilePath, PolicyName, RuleName Fqbn
```

| time | host | FilePath |
|------|------|----------|
|------|------|----------|

We would like to use [third party code](#) to improve the functionality of this website.



Home About



09-18  
11:35:54

sauteTomate

%OSDRIVE%\USERS\IPFYX\APP  
VS CODE\BIN\CODE.EXE

Got you ! There should not be that many binary signed without ProductName right ? Let's count them quickly in System32 for example :

```
POWERSHELL  
gci -Recurse C:\Windows\System32 -include *.exe |Get-AppLockerF:  
  
Count 20
```

Sigh. 20 binary names to masquerade as...

## File creation

Another great idea from [lobas](#) :

IOC: File write of code\_tunnel.json which is parametrizable, but defaults to: %UserProfile%\vscode-cli\code\_tunnel.json

license\_consent.json file could also be watched.

```
POWERSHELL  
PS > gc C:\Users\ipfyr\.vscode\cli\code_tunnel.json  
{  
  "name": "sauteTomate",  
  "id": "9s43zAc9",  
  "cluster": "uks1"  
}  
PS > gc C:\Users\ipfyr\.vscode\cli\license_consent.json
```

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



inside the user home directory is not enough since it can be changed with the `--cli-data-dir` option.

POWERSHELL

```
.\code.exe tunnel help
...
GLOBAL OPTIONS:
  --cli-data-dir <CLI_DATA_DIR>  Directory where CLI metadata
```

## Web traffic monitoring

If you haven't blocked the domains I mentioned earlier, watch out for any HTTP traffic toward those domains.

## Conclusion

A GPO parameter would be awesome but it is yet to be seen. An Applocker hash rule is not sustainable. Moreover, your non-domain-joined computers would not be concerned by neither of them. Therefore, for now, we are only left to blocking those two bad boys :

```
*.tunnels.api.visualstudio.com
*.devtunnels.ms
```

We would like to use [third party code](#) to improve the functionality of this website.



Home

About



We would like to use [third party code](#) to improve the functionality of this website.