Detecting 'Dev Tunnels'

BlueteamOps · Follow
Published in Detect FYI · 5 min read · Oct 23, 2023

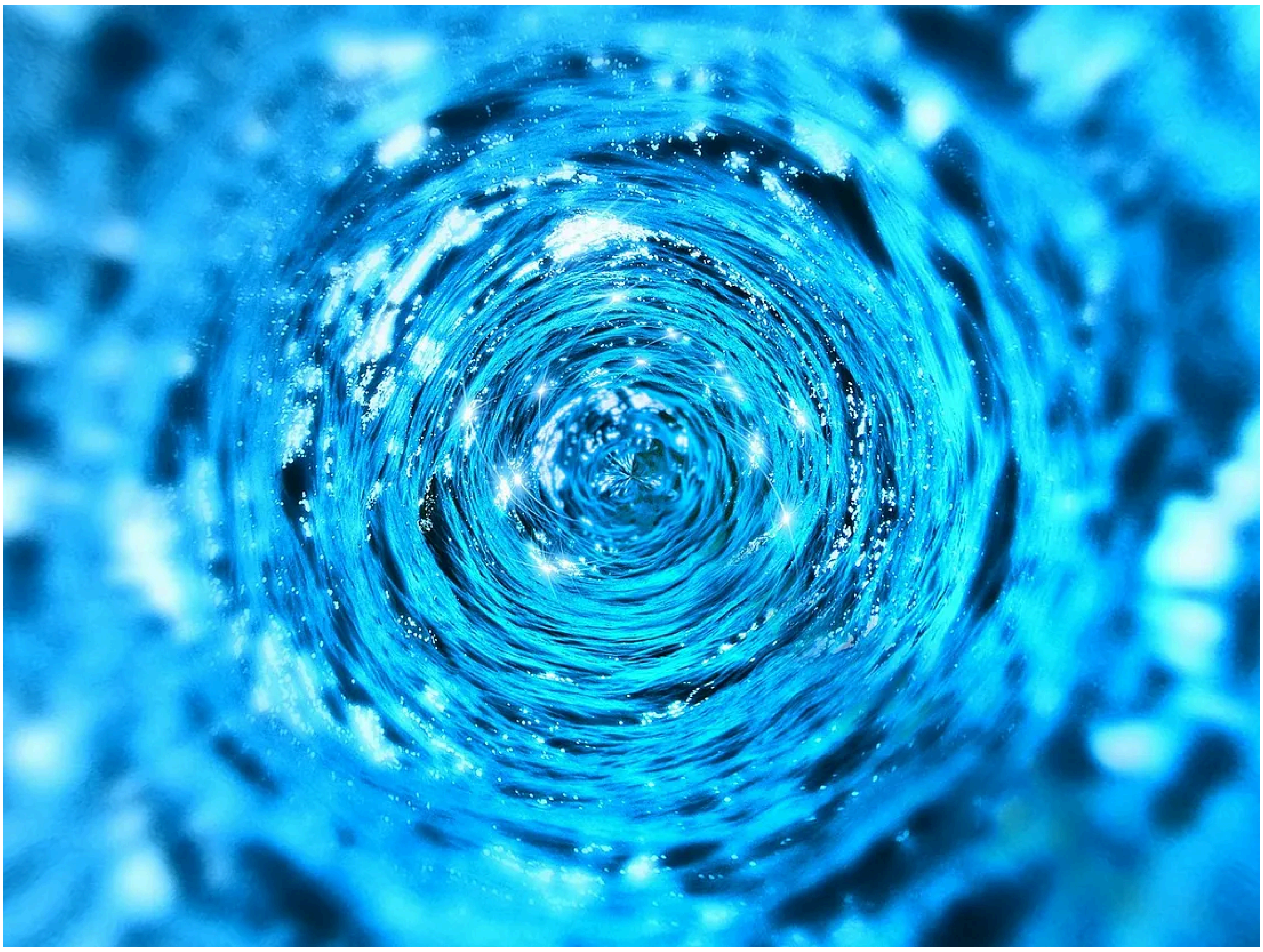
Image by Jiří Rotrekl from Pixabay

DevOps teams leverages tools such as Cloudflared, ngrok to make local services (i.e. an internal web application) accessible from the public internet using secure HTTP connections. This avoids the need to create special firewall rules, as HTTPS traffic is usually allowed on most networks. Microsoft recently launched Dev tunnels for developers. It allows tunnelling of multiple different ports using the same tunnel. It leverages Microsoft's Dev Tunnel relay service (a cloud service) which secure communications between a dev tunnel host and clients..

Adversaries can abuse these tools for command and control, persistence, and to evade defenses. There are already red team writeups available on how Dev Tunnels can be being used for C2. Hence, I thought of spending sometime using this tool to get to know it better so defenders can be ready. In this article I will be going through detection opportunities when Dev Tunnels is being used for tunnelling RDP.

By design, Microsoft has implemented the following security measures:

- **Requires a Microsoft or GitHub account**: This helps to ensure that only authorised users can create and access Dev Tunnels.
- **Anti-phishing protection**: First time a Dev Tunnels user connects to a web-forwarding url, the user is presented with an anti-phishing page.
- **Authentication required to access the tunnel by default:** By default, only the user who created a tunnel can access it. This can be overridden by enabling anonymous access or by sharing a token with other users.
- **Expiration**: Tunnels expire after 30 days by default, but this can be customized. This helps to prevent unauthorised access to tunnels that are no longer needed.
- **Token-based access**: Tokens can be used to grant other users access to a tunnel without enabling anonymous access. Tokens expire after 24 hours, which helps to reduce the risk of unauthorised access.

. . .

Detection Opportunities

Network detections

Following DNS queries were recorded in my test hosts and the queries occurred in the following order.

DNS queries (clusterId’s highlighted)

aue.rel.tunnels.api.visualstudio.com
global.rel.tunnels.api.visualstudio.com
aue-data.rel.tunnels.api.visualstudio.com

Based on Microsoft documentation Dev Tunnels may use any of the following sub-domains/domains during its operation. An up to date list of clusterId’s can be found here.

global.rel.tunnels.api.visualstudio.com
[clusterId].rel.tunnels.api.visualstudio.com
[clusterId]-data.rel.tunnels.api.visualstudio.com

*.[clusterId].devtunnels.ms
*.devtunnels.ms

Host based detection opportunities

Process activity of interest

Before creating a tunnel, a user must log in with either a Microsoft ID or GitHub ID. The command line arguments are as follows:

```
Authenticates using Microsoft

devtunnel user login

Authenticates using Github with either browser based or via device code

devtunnel user login -g
devtunnel user login -d
devtunnel user login -g -d
```

To create a tunnel for RDP with anonymous access, run the following command on the host. By default, only the user who created the tunnel can access it. Alternatively, you can create a token (which expires after 24 hours) to grant access to others. Once the tunnel is setup, the command-line output shows the randomly generated URL and the tunnel ID. This can be used to access host. Output shown on-screen after creation of the tunnel contains a URL to access it. It should be noted that the URL contains the port that will be accessed via the tunnel as shown in Figure 1.

Figure 1 — Creation of a temporary Dev tunnel with anonymous access

Command below will create a temporary tunnel to expose port 3389.

```
devtunnel.exe host -p 3389 - allow-anonymous
```

A persistent tunnel with anonymous access targeting port 3389 can be created by running the following command sequence.

```
devtunnel.exe create -a
devtunnel.exe port create -p 3389
devtunnel.exe host
```

Figure — 2 Logging in via RDP using the newly created tunnel

Dev Tunnels execution results in creation of number of files. In my testing the files were observed to be created within the folder path *C:\Users\dev\AppData\Local\Temp\.net\devtunnel\***1ad4***\*

Additional testing identified that the 4 digit sub-folder (highlighted above) is randomised with each tunnel execution. Following regex can be used to detect file creation activity.

(?i)\w:\\(Users|windows)\\.*\\AppData\\Local\\Temp\\\.net\\\w+\\\w{4}\\

Below list contains the most notable files since Dev tunnels leverages SSH to carryout secure communications and port forwarding activities.

Microsoft.DevTunnels.Ssh.Tcp.dll
Microsoft.DevTunnels.Ssh.dll
Microsoft.DevTunnels.Management.dll
Microsoft.DevTunnels.Contracts.dll
Microsoft.DevTunnels.Connections.dll

Following registry set activity was recorded when the devtunnel.exe process set a Microsoft Authentication Library related DLL.

Figure 3 — Event ID 13 — devtunnel.exe setting MSAL dll in registry

Event ID 25 and Event ID 24 from Microsoft-Windows-TerminalServices-LocalSessionManager/Operational contains the Source Network Address with the value **%16777216** which is a positive indication of tunnelling activity (not just applicable to Dev Tunnels but others as well). Sophos and Logpoint articles also covers this in detail.

Figure 4 — Event ID 25 entry related to the RDP interaction

Figure 5 and Figure 6 shows successful authentication through the tunnel. Initial 4624 event had the Logon type as 3 and also leaks the adversary workstation name. The source address is shown as ::1 and source port as 0. Within seconds we see the Logon type 10 record with the same source address and port. However, in this instance the workstation name has the value of the host that runs the tunnel and not the hostname of adversary's machine.

Figure 5 — Successful authentication via Dev Tunnels — Logon Type 3

Figure 6 — Successful authentication via Dev Tunnels — Logon Type 10

Stay tuned for Sigma rules!

Detection Engineering Threat Detection Threat Hunting Threat Research Dfir

--

Written by BlueteamOps

Follow

81 Followers · Writer for Detect FYI

Janantha Marasinghe’s Research