Security

Article • 08/25/2023 • 5 contributors

In this article

Dev tunnels is a security-focused developer tunneling service. In this article, learn about how dev tunnels are secured.

Overview

By default, hosting and connecting to a tunnel requires authentication with the same Microsoft, Microsoft Entra ID, or GitHub account that created the tunnel. Tunneling requires outbound connections to be made to the service hosted in Azure. No inbound connections are required to use the service.

Domains

Access to dev tunnels can be controlled by allowing or denying outbound access to the following domains:

- Authentication
  - `github.com`
  - `login.microsoftonline.com`
- 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`

The list of current `[clusterId]` values is available at https://global.rel.tunnels.api.visualstudio.com/api/v1/clusters .

Web-forwarding

Tunnel ports using the HTTP(S)/WS(S) protocols can be accessed directly via the provided web-forwarding url (for example: `https://tunnelid-3000.devtunnels.ms`).

- Insecure client connections are always automatically upgraded to HTTPS/WSS.
- HTTP Strict Transport Security (HSTS) is enabled with a one year max-age.
- The minimum TLS version the service supports is 1.2, with TLS 1.3 being the preferred version.

- TLS termination is done at service ingress using service certificates, issued by a Microsoft CA.
  - After TLS termination, header rewriting takes place. This is required for many web application development scenarios.

Anti-phishing protection

When connecting to a web-forwarding url for the first time, users are presented with an interstitial anti-phishing page. The page is skipped under the following circumstances:

- The request uses a method other than `GET`
- The request `Accept` header doesn't contain `text/html`
- The request contains the `X-Tunnel-Skip-AntiPhishing-Page` header
- The request contains the `X-Tunnel-Authorization` header
- The user has already visited the page and clicked continue

Tunnel access

By default, tunnels and tunnel ports are private and only accessible to the user who created the tunnel.

If a tunnel or tunnel port does need to be accessed without authentication, an allow-anonymous Access control entry (ACE) can be added (use `--allow-anonymous`).

Tunnel access can also be extended to your current Microsoft Entra tenant (use `--tenant`) or specific GitHub organizations (use `--organization`); for the latter see GitHub Organization Access below.

The CLI can also be used to request access tokens that grant limited access to anyone holding the token (use `devtunnel token`). This is an advanced feature but can be useful in specific situations.

Currently, four types of tunnel access tokens are available:

- A "client access token" allows the bearer to connect to any ports of the tunnel.
- A "host access token" allows the bearer to host the tunnel and accept connections, but not make any other changes to it.
- A "manage ports access token" allows the bearer to add and delete ports on a tunnel.
- A "management access token" allows the bearer to perform any operations on that tunnel, including setting access controls, hosting, connecting, and deleting the tunnel.

All of the tokens are limited to the current tunnel; they don't grant access to any of the current user's *other* tunnels, if any. The tokens expire after some time (currently 24 hours). Tokens can only be refreshed using an actual user identity that has manage-scope access to the tunnel (not just a management access token).

Most CLI commands can accept a `--access-token` argument with an appropriate token as an alternative to logging in.

Web clients can pass a token in a header to authorize requests to a tunnel URI:

HTTP Copy

```
X-Tunnel-Authorization: tunnel <TOKEN>
```

Tip

This is useful for non-interactive clients as it allows them to access tunnels without requiring anonymous access to be enabled. We use the `X-Tunnel-Authorization` header

instead of the standard `Authorization` header to prevent potentially interfering with application-specific authorization.

See the Manage dev tunnel access section to learn more about how to manage tunnel access through the CLI.

GitHub Organization Access

To support tunnels granting access to all members of a GitHub organization, install the Dev Tunnels GitHub app in the organization. That gives the Dev Tunnels service permission to check users' membership status in that organization. (Dev Tunnels does not require repo permissions to the org.) You may need to be an admin in the GitHub organization to perform this operation.

Further questions

If after reviewing this page, you have further questions, see Feedback and support.

Feedback

Was this page helpful? Yes No

Get help at Microsoft Q&A

Additional resources

Training

Module
Introduction to Remote - Tunnels extension - Training

Learn how to enable remote tunnels with Visual Studio Code.

Certification
Microsoft Certified: Azure Security Engineer Associate - Certifications

Demonstrate the skills needed to implement security controls, maintain an organization's security posture, and identify and remediate security vulnerabilities.