mTLS

- Interface

import { TabItem, Tabs, WranglerConfig } from "~/components";

When using HTTPS, a server presents a certificate for the client to authenticate in order to prove their identity. For even tighter security, some services require that the client also present a certificate.

This process - known as mTLS - moves authentication to the protocol of TLS, rather than managing it in application code. Connections from unauthorized clients are rejected during the TLS handshake instead.

To present a client certificate when communicating with a service, create a mTLS certificate binding in your Worker project's Wrangler file. This will allow your Worker to present a client certificate to a service on your behalf.

:::caution

Currently, mTLS for Workers cannot be used for requests made to a service that is a proxied zone on Cloudflare. If your Worker presents a client certificate to a service proxied by Cloudflare, Cloudflare will return a `520` error.

:::

First, upload a certificate and its private key to your account using the `wrangler mtls-certificate` command:

:::caution

The `wrangler mtls-certificate upload` command requires the SSL and Certificates Edit API token scope. If you are using the OAuth flow triggered by `wrangler login`, the correct scope is set automatically. If you are using API tokens, refer to Create an API token to set the right scope for your API token. :::

```
npx wrangler mtls-certificate upload --cert cert.pem --key key.pem --name my-client-cert
```

Then, update your Worker project's Wrangler file to create an mTLS certificate binding:

```
{
"mtls_certificates": [
{
"binding": "MY_CERT",
"certificate_id": "<CERTIFICATE_ID>"
}
]
}
```

:::note

Certificate IDs are displayed after uploading, and can also be viewed with the command `wrangler mtls-certificate list`. :::

Adding an mTLS certificate binding includes a variable in the Worker's environment on which the `fetch()` method is available. This `fetch()` method uses the standard Fetch API and has the exact same signature as the global `fetch`, but always presents the client certificate when establishing the TLS connection.

:::note

mTLS certificate bindings present an API similar to service bindings. :::

Interface

```
export default {
async fetch(request, environment) {
return await environment.MY_CERT.fetch("https://a-secured-origin.com");
},
};
```

```
interface Env {
  MY_CERT: Fetcher;
}
```

```
export default {
    async fetch(request, environment): Promise<Response> {
        return await environment.MY_CERT.fetch("https://a-secured-origin.com")
    }
} satisfies ExportedHandler<Env>;
```

□□□ #2
□ □□□ □□□ 2026-11-01 05:48:50 UTC
□ □□□ □□□ 2026-11-01 05:48:51 UTC