Skip to content

[![logo](../../assets/logo-letter.svg)](../.. "uv")

uv

Authentication

Initializing search

[uv](https://github.com/astral-sh/uv "Go to repository")

[![logo](../../assets/logo-letter.svg)](../.. "uv") uv

[uv](https://github.com/astral-sh/uv "Go to repository")

  * [Introduction](../..)
  * [Getting started](../../getting-started/)

Getting started

    * [Installation](../../getting-started/installation/)
    * [First steps](../../getting-started/first-steps/)
    * [Features](../../getting-started/features/)
    * [Getting help](../../getting-started/help/)
  * [Guides](../../guides/)

Guides

* [Installing Python](../../guides/install-python/)

* [Running scripts](../../guides/scripts/)

* [Using tools](../../guides/tools/)

* [Working on projects](../../guides/projects/)

* [Publishing packages](../../guides/package/)

* [Integrations](../../guides/integration/)

Integrations

* [Docker](../../guides/integration/docker/)

* [Jupyter](../../guides/integration/jupyter/)

* [GitHub Actions](../../guides/integration/github/)

* [GitLab CI/CD](../../guides/integration/gitlab/)

* [Pre-commit](../../guides/integration/pre-commit/)

* [PyTorch](../../guides/integration/pytorch/)

* [FastAPI](../../guides/integration/fastapi/)

* [Alternative indexes](../../guides/integration/alternative-indexes/)

* [Dependency bots](../../guides/integration/dependency-bots/)

* [AWS Lambda](../../guides/integration/aws-lambda/)

* [Concepts](../../concepts/)

Concepts

* [Projects](../../concepts/projects/)

Projects

Configuration

* [The pip interface](../../pip/)

The pip interface

  * [Using environments](../../pip/environments/)
  * [Managing packages](../../pip/packages/)
  * [Inspecting packages](../../pip/inspection/)
  * [Declaring dependencies](../../pip/dependencies/)
  * [Locking environments](../../pip/compile/)
  * [Compatibility with pip](../../pip/compatibility/)
* [Reference](../../reference/)

Reference

  * [Commands](../../reference/cli/)
  * [Settings](../../reference/settings/)
  * [Troubleshooting](../../reference/troubleshooting/)

Troubleshooting

    * [Build failures](../../reference/troubleshooting/build-failures/)
    * [Reproducible examples](../../reference/troubleshooting/reproducible-examples/)
  * [Resolver](../../reference/resolver-internals/)
  * [Benchmarks](../../reference/benchmarks/)
  * [Policies](../../reference/policies/)

Policies

Table of contents

1. [Introduction](../..)

2. [Configuration](../)

# Authentication

## Git authentication

uv allows packages to be installed from Git and supports the following schemes for authenticating with private repositories.

Using SSH:

* `git+ssh://git@<hostname>/...` (e.g. `git+ssh://[[email protected]](/cdn-cgi/l/email-protection)/astral-sh/uv`)

* `git+ssh://git@<host>/...` (e.g. `git+ssh://[](/cdn-cgi/l/email-protection)/astral-sh/uv`)

See the [GitHub SSH documentation](https://docs.github.com/en/authentication/connecting-to-github-with-ssh/about-ssh) for more details on how to configure SSH.

Using a password or token:

* `git+https://<user>:<token>@<hostname>/...` (e.g. `git+https://git:[[email protected]](/cdn-cgi/l/email-protection)/astral-sh/uv`)
* `git+https://<token>@<hostname>/...` (e.g. `git+https://[[email protected]](/cdn-cgi/l/email-protection)/astral-sh/uv`)
* `git+https://<user>@<hostname>/...` (e.g. `git+https://[[email protected]](/cdn-cgi/l/email-protection)/astral-sh/uv`)

When using a GitHub personal access token, the username is arbitrary. GitHub does not support logging in with password directly, although other hosts may. If a username is provided without credentials, you will be prompted to enter them.

If there are no credentials present in the URL and authentication is needed, the [Git credential helper](https://git-scm.com/doc/credential-helpers) will be queried.

## HTTP authentication

uv supports credentials over HTTP when querying package registries.

Authentication can come from the following sources, in order of precedence:

* The URL, e.g., `https://<user>:<password>@<hostname>/...`
* A [`.netrc`](https://everything.curl.dev/usingcurl/netrc) configuration file
* A [keyring](https://github.com/jaraco/keyring) provider (requires opt-in)

If authentication is found for a single net location (scheme, host, and port), it will be cached for the duration of the command and used for other queries to that net location. Authentication is not cached across invocations of uv.

`.netrc` authentication is enabled by default, and will respect the `NETRC` environment variable if defined, falling back to `~/.netrc` if not.

To enable keyring-based authentication, pass the `--keyring-provider subprocess` command-line argument to uv, or set `UV_KEYRING_PROVIDER=subprocess`.

Authentication may be used for hosts specified in the following contexts:

* `index-url`
* `extra-index-url`
* `find-links`
* `package @ https://...`

See the [`pip` compatibility guide](../../pip/compatibility/#registry-authentication) for details on differences from `pip`.

## Custom CA certificates

By default, uv loads certificates from the bundled `webpki-roots` crate. The `webpki-roots` are a reliable set of trust roots from Mozilla, and including them in uv improves portability and performance (especially on macOS, where reading the system trust store incurs a significant delay).

However, in some cases, you may want to use the platform's native certificate store, especially if you're relying on a corporate trust root (e.g., for a mandatory proxy) that's included in your system's certificate store. To instruct uv to use the system's trust store, run uv with the `--native-tls` command-line flag, or set the `UV_NATIVE_TLS` environment variable to `true`.

If a direct path to the certificate is required (e.g., in CI), set the `SSL_CERT_FILE` environment variable to the path of the certificate bundle, to instruct uv to use that file instead of the system's trust store.

If client certificate authentication (mTLS) is desired, set the `SSL_CLIENT_CERT` environment variable to the path of the PEM formatted file containing the certificate followed by the private key.

Finally, if you're using a setup in which you want to trust a self-signed certificate or otherwise disable certificate verification, you can instruct uv to allow insecure connections to dedicated hosts via the `allow-insecure-host` configuration option. For example, adding the following to `pyproject.toml` will allow insecure connections to `example.com`:

```
[tool.uv]
allow-insecure-host = ["example.com"]
```

`allow-insecure-host` expects to receive a hostname (e.g., `localhost`) or hostname-port pair (e.g., `localhost:8080`), and is only applicable to HTTPS connections, as HTTP connections are inherently insecure.

Use `allow-insecure-host` with caution and only in trusted environments, as it can expose you to security risks due to the lack of certificate verification.

## Authentication with alternative package indexes

See the [alternative indexes integration guide](../../guides/integration/alternative-indexes/) for details on authentication with popular alternative Python package indexes.

October 24, 2024

Back to top [Previous Environment variables](../environment/) [Next Package indexes](../indexes/)

Made with [Material for MkDocs Insiders](https://squidfunk.github.io/mkdocs-material/)

[ ](https://github.com/astral-sh/uv "github.com") [ ](https://discord.com/invite/astral-sh "discord.com") [ ](https://pypi.org/project/uv/ "pypi.org") [ ](https://x.com/astral_sh "x.com")