

Installation from binary



- [System requirements](#)
 - [Downloading and installing the binary](#)
 - [Setting up the runner user](#)
 - [Setting up the container environment](#)
 - [Docker](#)
 - [Podman](#)
 - [LXC](#)
 - [Host](#)
 - [Configuration](#)
 - [Podman socket configuration](#)
 - [Starting the runner](#)
 - [Running as a systemd service](#)
-

System requirements

Forgejo Runner requires that Git is installed, and has been tested with a minimum version of Git 2.24.3.

Downloading and installing the binary

Download the latest [binary release](#) and verify its signature:

```
$ export ARCH=$(uname -m | sed 's/x86_64/amd64;/s/aarch64/arm64/')
$ export RUNNER_VERSION=$(curl -X 'GET'
```

```
https://data.forgejo.org/api/v1/repos/forgejo/runner/releases/latest | jq .name -r | cut -c 2-
)
$ export
FORGEJO_URL="https://code.forgejo.org/forgejo/runner/releases/download/v${RUNNER_VERSION}/forgejo-runner-${RUNNER_VERSION}-linux-${ARCH}"
$ wget -O forgejo-runner ${FORGEJO_URL} || curl -o forgejo-runner ${FORGEJO_URL}
$ chmod +x forgejo-runner
$ wget -O forgejo-runner.asc ${FORGEJO_URL}.asc || curl -o forgejo-runner.asc
${FORGEJO_URL}.asc
$ gpg --keyserver hks://keys.openpgp.org --recv EB114F5E6C0DC2BCDD183550A4B61A2DC5923710
$ gpg --verify forgejo-runner.asc forgejo-runner && echo "✓ Verified" || echo "✗ Failed"
Good signature from "Forgejo <contact@forgejo.org>"
      aka "Forgejo Releases <release@forgejo.org>"
✓ Verified
```

Next, copy the downloaded binary to `/usr/local/bin` and make it executable:

```
$ cp forgejo-runner /usr/local/bin/forgejo-runner
```

You should now be able to test the runner by running `forgejo-runner -v`:

```
$ forgejo-runner -v
forgejo-runner version v13.0.0
```

Setting up the runner user

Set up the user to run the daemon:

```
$ useradd --create-home runner
```

If the runner will be using Docker, ensure the `runner` user has access to the docker socket. Run the following command to grant access to the docker socket:

```
$ usermod -aG docker runner
```

If the runner will be using Podman, no particular permissions are needed, but you will need to configure your system to run the Podman service. See the [Podman](#) section below for more information.

Setting up the container environment

The Forgejo Runner relies on application containers (Docker, Podman, etc.) or system containers (LXC) to execute a workflow in an isolated environment. They need to be installed and configured independently.

It is common for workflows to also require interaction with a container environment, for example to execute `docker build` commands. This is distinct from how Forgejo Runner itself executes jobs, and is an optional configuration that is described in detail in [Utilizing Docker within Actions](#).

Docker

See the [Docker installation](#) documentation for more information.

Podman

On most recent Linux distributions, the `podman` package alone should give you a working version of Podman which can run as the unprivileged `runner` user. You *do not* need to install any supplemental packages for Docker compatibility or "rootless" operation. However, some configuration is necessary to ensure that the runner is started at boot time and can access the Podman service.

These instructions are for Debian 13 (trixie) but should also apply to Ubuntu, and likely to other `systemd` based distributions. They assume you have downloaded and installed the `forgejo-runner` and created the `runner` user as described above.

First, install `podman`:

```
sudo apt install podman
```

Next, you will configure `systemd` to create a socket which will launch Podman as the `runner` user when accessed by `forgejo-runner`. As your normal user (it is *not* necessary to enable logins for `runner`), run:

```
sudo systemctl --user -M runner@ enable --now podman.socket
```

Verify that the socket is enabled and find its location:

```
$ sudo systemctl --user -M runner@ status podman.socket
● podman.socket - Podman API Socket
    Loaded: loaded (/usr/lib/systemd/user/podman.socket; enabled; preset: enabled)
    Active: active (listening) since Wed 2026-04-29 16:47:03 EDT; 78ms ago
  Invocation: 18779cd4131144e197422ade1c100a0e
    Triggers: ● podman.service
      Docs: man:podman-system-service(1)
     Listen: /run/user/1004/podman/podman.sock (Stream)
```

Use the path to this socket shown above in the `docker_host` parameter when [configuring the runner](#).

Finally, you must set the `runner` user to "linger" so that the socket will be created at boot:

```
sudo loginctl enable-linger runner
```

On non-systemd distributions, you are free to choose any path you like for the podman socket (for instance, `/home/runner/podman.sock`). It can then be provided by running, as the `runner` user:

```
podman system service -t 0 unix:///home/runner/podman.sock
```

See the documentation for your distribution's init system for how to ensure this is started at boot time.

LXC

For jobs to run in LXC containers, the Forgejo Runner needs passwordless sudo access for all `lxc-*` commands on a Debian GNU/Linux `bookworm` system where [LXC](#) is installed. The [LXC helpers](#) can be used as follows to create a suitable container:

```
$ git clone https://code.forgejo.org/forgejo/lxc-helpers
$ sudo cp -a lxc-helpers/lxc-helpers{,-lib}.sh /usr/local/bin
$ lxc-helpers.sh lxc_container_create myrunner
$ lxc-helpers.sh lxc_container_start myrunner
$ lxc-helpers.sh lxc_container_user_install myrunner 1000 debian
```

“ [!note] Note Multiarch [Go](#) builds and [binfmt](#) need `bookworm` to produce and test binaries on a single machine for people who do not have access to dedicated

hardware.

The Forgejo Runner can then be installed and run within the `myrunner` container.

```
$ lxc-helpers.sh lxc_container_run forgejo-runners -- sudo --user debian bash
$ sudo apt-get install docker.io wget gnupg2
$ wget -O forgejo-runner
https://code.forgejo.org/forgejo/runner/releases/download/v13.0.0/forgejo-runner-13.0.0-linux-
amd64
...
```

“ [!warning] Warning LXC containers do not provide a level of security that makes them safe for potentially malicious users to run jobs. They provide an excellent isolation for jobs that may accidentally damage the system they run on.

Host

There are no requirements for jobs that run directly on the host.

“ [!warning] Warning There is no isolation at all and a single job can permanently destroy the host.

“ [!warning] Warning Processes forked out of a job may linger after the job is complete, possibly forever, if the job fails to wait for them to complete.

Configuration

As the `runner` user, generate the default configuration file for `Forgejo Runner`, and store it in a file accessible to the user (`/home/runner/runner-config.yml` for instance):

```
forgejo-runner generate-config > /home/runner/runner-config.yml
```

Forgejo Runner needs to be configured and registered with Forgejo before it can be started successfully. [Configure Forgejo Runner](#), editing `/home/runner/runner-config.yml` file as you proceed.

“ [!note] Note Forgejo Runner requires the configuration file to be explicitly specified with the `-c` command-line option. There is no default configuration file location.

Podman socket configuration

When using Podman, you must configure the `docker_host` parameter in the `container` section of the configuration file with the path to the Podman socket created [above](#). For example, if the output of `systemctl` shows the socket as `/run/user/1004/podman/podman.sock`, then the configuration would be:

```
container:
  docker_host: unix:///run/user/1004/podman/podman.sock
```

You can also set this with the `DOCKER_HOST` environment variable.

As a reminder, to find the path to the podman socket, as the `runner` user, you can run:

```
XDG_RUNTIME_DIR=/run/user/$(id -u) systemctl --user status podman.socket
```

Note that the `XDGRUNTIME_DIR` environment variable required by `systemctl` is *not* set in non-login shells.

Starting the runner

After the runner has been registered, it can be started by running `forgejo-runner daemon` as the `runner` user, in the home directory:

```
$ whoami
runner
$ pwd
/home/runner
$ forgejo-runner daemon -c runner-config.yml
INFO[2024-09-14T19:19:14+02:00] Starting runner daemon
```

Running as a systemd service

To automatically start the runner when the system starts, copy [forgejo-runner.service](#) to `/etc/systemd/system/forgejo-runner.service`.

Then run `systemctl daemon-reload` to reload the unit files. Run `systemctl start forgejo-runner.service` to test the new service. If everything works, run `systemctl enable forgejo-runner.service` to enable auto-starting the service on boot.

Use `journalctl -u forgejo-runner.service` to read the runner logs.

#3

❑ cocobook-sync ## 2025-07-17 07:42:12 UTC

❑ cocobook-sync ## 2026-05-27 14:50:59 UTC