

GitLab Quickstart

Jens Lechtenböcker

Summer Term 2022

Contents

1	Note	1
2	Install Git	1
3	Create SSH Key Pair	1
4	Register Public Key on GitLab Server	3
5	Test	3
6	Perform First-Time Git Setup	3

1 Note

This quickstart just provides pointers

- to install Git and
- to setup SSH keys for use with GitLab.

See [this presentation](#) for an introduction to Git.

2 Install Git

First, [install Git](#). Or, maybe, install GNU/Linux first, then Git.

3 Create SSH Key Pair

Secure Shell (SSH) is based on [asymmetric cryptography](#) and is used by Git in the background to setup secure communication channels with strong [authentication](#).

Briefly, with asymmetric cryptography, keys come in pairs, a private and a public key. While you share your public key, you need to protect your private key so that nobody else can access it. For authentication of Git commands, you register your public key on the server (as [instructed below](#)). Whenever you perform a Git operation that requires authentication, the server initiates

a challenge-response protocol, in which your client needs to prove that it can access the private key that belongs to the public key registered on the server.

Thus, you need an SSH key pair (consisting of private and public key).

Along the lines of [instructions on GitLab](#), you may want to create a key pair using the algorithm Ed25519 with the following command (in your command line, e.g., [Bash](#), potentially coming with [Git for Windows](#)):

```
ssh-keygen -t ed25519 -C "Test key pair for gitlab"
```

Ed25519 may not be available in older SSH implementations. You can leave out option `-t ed25519` to go with the program's default settings, or use `-t rsa -b 4096` for RSA keys with a length of 4096 bits.

This produces a response and asks you where to save your private key:

```
Generating public/private ed25519 key pair.
```

```
Enter file in which to save the key (/home/user/.ssh/id_ed25519):
```

Press Enter/Return to **accept the default** (if you change this, Git may not find your key later on).

Then, you are asked for a passphrase.

```
Enter passphrase (empty for no passphrase):
```

Your private key is protected with that passphrase. Thus, if you leave this empty, everyone with access to your machine (or the file with the private key, e.g., a backup) can use that private key and authenticate as you. If you specify a non-empty passphrase, you need to enter it when Git attempts to use your private key (the passphrase can be cached with `ssh-agent` or `gpg-agent`, which is beyond the scope of this quickstart). The command asks you to confirm your passphrase:

```
Enter same passphrase again:
```

Afterwards, more output tells you where the public key is stored (and more, which is not important for our purposes).

```
Your identification has been saved in /home/user/.ssh/id_ed25519.
```

```
Your public key has been saved in /home/user/.ssh/id_ed25519.pub.
```

```
The key fingerprint is:
```

```
SHA256:0mT81aPfvIYiab64hrwSvIZ0yYlFvh3E90LcmqXFYcE Test key pair for gitlab
```

```
The key's randomart image is:
```

```
+--[ED25519 256]--+
|          ..          |
|         . E. .       |
|          + o . o     |
|         . = = o . .  |
|         . + S * .    |
|o oo . + B   . o    |
|. = . = o = o   o o |
|.o * + o.= . . . . |
|..o +o+o+oo . . .  |
+-----[SHA256]-----+
```

4 Register Public Key on GitLab Server

You need to store the **public** key on the server.

You could open the public key (`/home/user/.ssh/id_ed25519.pub` in my case above) in an editor and copy it to your clipboard or use one of commands suggested in [instructions on GitLab](#) to copy it to the clipboard. In any case, follow the section “Add an SSH key to your GitLab account” to register your public key with your GitLab account—on our server `wiwi-gitlab.uni-muenster.de`, not on `gitlab.com` (short story of that section: sign in, go to your “Preferences” via avatar icon on top right, select “SSH Keys”, paste public key into “Key” box, enter a title). After that section, return here and continue in this document to test your SSH setup.

5 Test

If you are not sure whether SSH setup was successful, execute this:

```
ssh -T git@wiwi-gitlab.uni-muenster.de
```

When you connect for the first time, your SSH client does not know the server’s public key yet. To prevent man-in-the-middle attacks (someone else between you and the server might pretend to be the server), it displays the fingerprint of that key and asks you whether to continue. Type **yes**, if you see this fingerprint among the output:

```
SHA256:yyvXkpVi0q1Gg/HgLOjkUG4hjva0sFPxKZ5PgsHAvE
```

If you see a success message from the previous command, you are good to go. Otherwise, add option `-v` for verbose output:

```
ssh -v -T git@wiwi-gitlab.uni-muenster.de
```

Pay attention to lines about `identity file`. Those ending in `-1` imply that `ssh` tried to access a key that does not exist. At least one must exist. Ask.

6 Perform First-Time Git Setup

Perform the [First-time Git setup](#) if you did not do so already.

License Information

This document is part of an OER collection to teach basics of distributed systems. Source code and source files are available on [GitLab](#) under free licenses.

Except where otherwise noted, the work “GitLab Quickstart”, © 2020-2022 Jens Lechtenbörger, is published under the Creative Commons license CC BY-SA 4.0.

Note: This PDF document is an inferior version of an OER [HTML page](#); [free/libre Org mode source repository](#).