GitHub Docs | Version: Free, Pro, & Team

Authentication / Connect with SSH / Generate new SSH key

Generating a new SSH key and adding it to the ssh-agent

After you've checked for existing SSH keys, you can generate a new SSH key to use for authentication, then add it to the ssh-agent.

Mac **Windows** Linux

In this article

About SSH key passphrases

You can access and write data in repositories on GitHub.com using SSH (Secure Shell Protocol). When you connect via SSH, you authenticate using a private key file on your local machine. For more information, see "About SSH."

When you generate an SSH key, you can add a passphrase to further secure the key. Whenever you use the key, you must enter the passphrase. If your key has a passphrase and you don't want to enter the passphrase every time you use the key, you can add your key to the SSH agent. The SSH agent manages your SSH keys and remembers your passphrase.

If you don't already have an SSH key, you must generate a new SSH key to use for authentication. If you're unsure whether you already have an SSH key, you can check for existing keys. For more information, see "Checking for existing SSH keys."

If you want to use a hardware security key to authenticate to GitHub, you must generate a new SSH key for your hardware security key. You must connect your hardware security key to your computer when you authenticate with the key pair. For more information, see the OpenSSH 8.2 release notes.

Generating a new SSH key

You can generate a new SSH key on your local machine. After you generate the key, you can add the public key to your account on GitHub.com to enable authentication for Git operations over SSH.

> **Note:** GitHub improved security by dropping older, insecure key types on March 15, 2022.
>
> As of that date, DSA keys (`ssh-dss`) are no longer supported. You cannot add new DSA keys to your personal account on GitHub.com.
>
> RSA keys (`ssh-rsa`) with a `valid_after` before November 2, 2021 may continue to use any signature algorithm. RSA keys generated after that date must use a SHA-2 signature algorithm. Some older clients may need to be upgraded in order to use SHA-2 signatures.

1. Open Git Bash.

2. Paste the text below, replacing the email used in the example with your GitHub email address.

   ```
   ssh-keygen -t ed25519 -C "your_email@example.com"
   ```

   > **Note:** If you are using a legacy system that doesn't support the Ed25519 algorithm, use:
   >
   > ```
   > ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
   > ```

   This creates a new SSH key, using the provided email as a label.

   ```
   > Generating public/private ALGORITHM key pair.
   ```

   When you're prompted to "Enter a file in which to save the key", you can press **Enter** to accept the default file location. Please note that if you created SSH keys previously, ssh-keygen may ask you to rewrite another key, in which case we recommend creating a custom-named SSH key. To do so, type the default file location and replace id_ALGORITHM with your custom key name.

```
> Enter file in which to save the key (/c/Users/YOU/.ssh/id_ALGORITHM):[Press enter]
```

3. At the prompt, type a secure passphrase. For more information, see "Working with SSH key passphrases."

   ```
   > Enter passphrase (empty for no passphrase): [Type a passphrase]
   > Enter same passphrase again: [Type passphrase again]
   ```

Adding your SSH key to the ssh-agent

Before adding a new SSH key to the ssh-agent to manage your keys, you should have checked for existing SSH keys and generated a new SSH key.

If you have GitHub Desktop installed, you can use it to clone repositories and not deal with SSH keys.

1. In a new *admin elevated* PowerShell window, ensure the ssh-agent is running. You can use the "Auto-launching the ssh-agent" instructions in "Working with SSH key passphrases", or start it manually:

   ```
   # start the ssh-agent in the background
   Get-Service -Name ssh-agent | Set-Service -StartupType Manual
   Start-Service ssh-agent
   ```

2. In a terminal window without elevated permissions, add your SSH private key to the ssh-agent. If you created your key with a different name, or if you are adding an existing key that has a different name, replace *id_ed25519* in the command with the name of your private key file.

   ```
   ssh-add c:/Users/YOU/.ssh/id_ed25519
   ```

3. Add the SSH public key to your account on GitHub. For more information, see "Adding a new SSH key to your GitHub account."

Generating a new SSH key for a hardware security key

If you are using macOS or Linux, you may need to update your SSH client or install a new SSH client prior to generating a new SSH key. For more information, see "Error: Unknown key type."

1. Insert your hardware security key into your computer.

2. Open Git Bash.

3. Paste the text below, replacing the email address in the example with the email address associated with your account on GitHub.

   ```
   ssh-keygen -t ed25519-sk -C "your_email@example.com"
   ```

   > **Note:** If the command fails and you receive the error `invalid format` or `feature not supported`, you may be using a hardware security key that does not support the Ed25519 algorithm. Enter the following command instead.
   >
   > ```
   > ssh-keygen -t ecdsa-sk -C "your_email@example.com"
   > ```

4. When you are prompted, touch the button on your hardware security key.

5. When you are prompted to "Enter a file in which to save the key," press Enter to accept the default file location.

   ```
   > Enter a file in which to save the key (c:\Users\YOU\.ssh\id_ed25519_sk):[Press enter]
   ```

6. When you are prompted to type a passphrase, press **Enter**.

   ```
   > Enter passphrase (empty for no passphrase): [Type a passphrase]
   > Enter same passphrase again: [Type passphrase again]
   ```

7. Add the SSH public key to your account on GitHub. For more information, see "Adding a new SSH key to your GitHub account."