

Outils de versionning

ESGI - 3ème année IW 2020

Sécurité, authentification et autorisations

Authentication multi-facteurs

Sécurité, authentification et autorisations

Ajouter des facteurs d'authentification

<https://github.com/settings/security>

Profile

Account

Account security

Security log

Security & analysis

Emails

Notifications

Scheduled reminders

Billing

SSH and GPG keys

Repositories

Organizations

Saved replies

Applications

Developer settings

Moderation settings

Blocked users

Change password

Old password

New password

Confirm new password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Update password

I forgot my password

Two-factor authentication

Enabled

Two-factor authentication adds an additional layer of security to your account by requiring more than just a password to log in. [Learn more.](#)

Two-factor methods	
Authenticator app	Configured <div>Edit</div>
Security keys ¹	security keys <div>Edit</div>
SMS number	Not configured <div>Edit</div>
Recovery options	
Recovery codes ¹	Viewed <div>Show</div>
Fallback SMS number ¹	+33 <div>Edit</div>
Recovery tokens ¹	Account recovery enabled <div>Edit</div>

https://gitlab.com/profile/two_factor_auth

GitLab Next

Projects

Groups

More

Search or jump to...

8

42

User Settings > Two-Factor Authentication > Account

Register Two-Factor Authenticator

Use an one time password authenticator on your mobile device or computer to enable two-factor authentication (2FA).

Disable two-factor authentication

Regenerate recovery codes

Register Universal Two-Factor (U2F) Device

Set up new device

Your device needs to be set up. Plug it in (if needed) and click the button on the left.

Use a hardware device to add the second factor of authentication.


As U2F devices are only supported by a few browsers, we require that you set up a two-factor authentication app before a U2F device. That way you'll always be able to log in - even when you're using an unsupported browser.

U2F Devices (1)

Name	Registered On	
Authenticator app	Sep 29, 2019	Delete

Confirmation de second facteur

Github



Confirm password to continue

Password [Forgot password?](#)

Confirm password


or

Use security key

Tip: You are entering `sudo mode`. We won't ask for your password again for a few hours.

[Terms](#) [Privacy](#) [Security](#) [Contact GitHub](#)

Gitlab



GitLab.com

GitLab.com offers free unlimited (private) repositories and unlimited collaborators.

- [Explore projects on GitLab.com](#) (no login needed)
- [More information about GitLab.com](#)
- [GitLab Community Forum](#)
- [GitLab Homepage](#)

By signing up for and by signing in to this service you accept our:

- [Privacy policy](#)
- [GitLab.com Terms](#).

Two-Factor Authentication

Two-factor authentication code

Enter the code from the two-factor app on your mobile device. If you've lost your device, you may enter one of your recovery codes.

Verify code

Git clone et 2FA

```
utilisateur@machine:~/$ git clone https://github.com/utilisateur/depot.git
Cloning into 'depot'...
Username for 'https://github.com': utilisateur
Password for 'https://utilisateur@github.com':
remote: Invalid username or password.
fatal: Authentication failed for https://github.com/utilisateur/depot.git/
```

```
utilisateur@machine:~/$ git clone https://gitlab.com/utilisateur/depot.git
Cloning into 'depot'...
Username for 'https://gitlab.com': utilisateur
Password for 'https://utilisateur@gitlab.com':
remote: HTTP Basic: Access denied
remote: You must use a personal access token with 'read_repository' or 'write_repository' scope for Git over HTTP.
remote: You can generate one at https://gitlab.com/profile/personal_access_tokens
fatal: Authentication failed for 'https://gitlab.com/utilisateur/depot.git/'
```

git + ssh = ♥

Sécurité, authentification et autorisations

Avantages

- Possibilité d'avoir plusieurs clés
 - un compte, un mot de passe, mais plusieurs clés, donc une par ordinateur possible pour invalider certains clients facilement.
- Compatible avec la double authentification
- Facile à partager temporairement avec quelqu'un
- Agis uniquement sur les dépôts et pas sur le compte, donc plus sécurisé en cas de vol.

Références :

- <https://gist.github.com/grawity/4392747>
- <http://jr0cket.co.uk/2016/05/ssh-or-https-that-is-the-github-question.html>

Générer une paire de clés

SSH fonctionne avec un échange de clés (<https://www.youtube.com/watch?v=y2SWzw9D4RA>).

```
ssh-keygen -t rsa -b 4096 -C "utilisateur@machine" -f /root/.ssh/id_rsa -N  
demo
```

- **-t** : algorithme de chiffrement à utiliser
- **-b** : taille de la clé en bits
- **-C** : commentaire, souvent l'adresse email ou le `utilisateur@machine`, mais peut être n'importe quelle chaîne de caractères
- **-f** : fichiers de sortie, ici `/root/.ssh/id_rsa` et `/root/.ssh/id_rsa.pub`
- **-N** : passphrase

Les permissions sont importantes

```
utilisateur@machine:~/$ ls -la ~/
total 12
drwxr-xr-x   3 root    root      4096 Sep 26 07:54.
drwxr-xr-x   1 root    root      4096 Sep 26 08:01..
drwx-----  2 root    root      4096 Sep 26 07:54.ssh
utilisateur@machine:~/$ ls -la ~/.ssh
total 16
drwx-----  2 root    root      4096 Sep 26 07:54.
drwxr-xr-x   3 root    root      4096 Sep 26 07:54..
-rw-----  1 root    root      3434 Sep 26 07:54 id_rsa
-rw-r--r--  1 root    root       745 Sep 26 07:54 id_rsa.pub
```

- `.ssh` , `chmod 700` : seul le propriétaire peut utiliser ce dossier (x = traverser)
- `.ssh/id_rsa` , `chmod 600` : clé privée, seul le propriétaire peut écrire/lire
- `.ssh/id_rsa.pub` , `chmod 644` : clé publique, tout le monde peut lire, seul le propriétaire modifie

Copier une clé publique

- MacOS :

- `cat ~/.ssh/id_rsa.pub | clip.exe`
- `cat ~/.ssh/id_rsa.pub | pbcopy`

- Linux (installer les logiciels) :

- `cat ~/.ssh/id_rsa.pub | xclip -selection clipboard`
- `cat ~/.ssh/id_rsa.pub | xsel --clipboard --input`

Ajouter la clé publique copiée sur Github

<https://github.com/settings/keys>

Profile

Account

Account security

Security log

Security & analysis

Emails

Notifications

Scheduled reminders

Billing

SSH and GPG keys

Repositories

Organizations

Saved replies

Applications

Developer settings

Moderation settings

Blocked users

SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

thomas@engineer.com

MD5:105:81:3a:8d:1a:14:97:12:150:08:1f:ee:3e:48:15b:db

SHA256:1a3jyhoQkL599gYHf/at6vkeXngp0u3a2kxue

Added on Mar 31, 2016

Last used within the last week — Read/write

Delete

MD5:0e1:78:4c:08:1a:16:45:159:ec:42:73:78:4f:0b:6c:63

SHA256:TAucVrfuKqBrAg35VtrB0z7mwR2k3vmsHP03s

Added on Apr 11, 2017 by Laravel Forge

Last used within the last 5 months — Read/write

Delete

MD5:da:6a:4e:b9:a0:70:d8:b9:03:2a:a8:db:0e:144:11

SHA256:BqptwQ298+zp7L15aBVC99Ta3Cw3v569C+7Y4

Added on Jun 4, 2020

Last used within the last 2 months — Read/write

Delete

Check out our guide to [generating SSH keys](#) or [troubleshooting common SSH Problems](#).

GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.

Email address: thomas@engineer.com

Key ID: 7061F427D1AE3AE5

Subkeys: 15DAD0E6D8BCD83

Added on Apr 5, 2016

Delete

Learn how to [generate a GPG key](#) and add it to your account.



Profile

Account

Account security

Security log

Security & analysis

Emails

Notifications

Scheduled reminders

Billing

SSH and GPG keys

Repositories

Organizations

Saved replies

Applications

Developer settings

Moderation settings

Blocked users

SSH keys / Add new

Title

Key

Begins with 'ssh-rsa', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

Add SSH key



Vérification de la connexion

```
utilisateur@machine:~/$ ssh git@github.com
Enter passphrase for key '/home/utilisateur/.ssh/id_rsa':
PTY allocation request failed on channel 0
Hi utilisateur! You've successfully authenticated, but GitHub does not provide shell access.
Connection to github.com closed.
```

```
utilisateur@machine:~/$ ssh git@gitlab.com
Enter passphrase for key '/home/utilisateur/.ssh/id_rsa':
PTY allocation request failed on channel 0
Welcome to GitLab, @utilisateur!
Connection to gitlab.com closed.
```

Connection en détail

```
utilisateur@machine :~/ $ ssh -v git@github.com
OpenSSH_7.9p1 Debian-10+deb10u2, OpenSSL 1.1.1d 10 Sep 2019
debug1: Reading configuration data /home/utilisateur/.ssh/config
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: Applying options for *
debug1: Connecting to github.com [140.82.121.4] port 22.
debug1: Connection established.
debug1: identity file /home/utilisateur/.ssh/id_rsa type 0
debug1: identity file /home/utilisateur/.ssh/id_rsa-cert type -1
debug1: identity file /home/utilisateur/.ssh/id_dsa type -1
debug1: identity file /home/utilisateur/.ssh/id_dsa-cert type -1
debug1: identity file /home/utilisateur/.ssh/id_ecdsa type -1
debug1: identity file /home/utilisateur/.ssh/id_ecdsa-cert type -1
debug1: identity file /home/utilisateur/.ssh/id_ed25519 type -1
debug1: identity file /home/utilisateur/.ssh/id_ed25519-cert type -1
debug1: identity file /home/utilisateur/.ssh/id_xmss type -1
debug1: identity file /home/utilisateur/.ssh/id_xmss-cert type -1
debug1: Local version string SSH-2.0-OpenSSH_7.9p1 Debian-10+deb10u2
debug1: Remote protocol version 2.0, remote software version babeld-2103f6d3
debug1: no match: babeld-2103f6d3
debug1: Authenticating to github.com:22 as 'git'
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: curve25519-sha256
debug1: kex: host key algorithm: rsa-sha2-512
debug1: kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit>
compression: none
debug1: kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit>
compression: none
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: ssh-rsa SHA256:nThbg6kXupJWG17E1IGOCspRomTxdCARLviKw6E5SY8
debug1: Host 'github.com' is known and matches the RSA host key.
debug1: Found key in /home/utilisateur/.ssh/known_hosts:4
debug1: rekey after 134217728 blocks
```

```
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: SSH2_MSG_NEWKEYS received
debug1: rekey after 134217728 blocks
debug1: Will attempt key: /home/utilisateur/.ssh/id_rsa RSA
SHA256:3ajVyNxQuKlj99g+gHf/FaE4Ykv0XRgDAuU3WiXsxUE
debug1: Will attempt key: /home/utilisateur/.ssh/id_dsa
debug1: Will attempt key: /home/utilisateur/.ssh/id_ecdsa
debug1: Will attempt key: /home/utilisateur/.ssh/id_ed25519
debug1: Will attempt key: /home/utilisateur/.ssh/id_xmss
debug1: SSH2_MSG_EXT_INFO received
debug1: kex_input_ext_info:
server-sig-algs=<ssh-ed25519-cert-v01@openssh.com,ecdsa-sha2-nistp521-cert-v01@openssh.com,ecdsa-sha2-nistp384-cert-v01@openssh.com,ecdsa-sha2-nistp256-cert-v01@openssh.com,rsa-sha2-512-cert-v01@openssh.com,rsa-sha2-256-cert-v01@openssh.com,ssh-rsa-cert-v01@openssh.com,ssh-ed25519,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-nistp256,rsa-sha2-512,rsa-sha2-256,ssh-rsa,ssh-dss>
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey
debug1: Next authentication method: publickey
debug1: Offering public key: /home/utilisateur/.ssh/id_rsa RSA
SHA256:3ajVyNxQuKlj99g+gHf/FaE4Ykv0XRgDAuU3WiXsxUE
debug1: Server accepts key: /home/utilisateur/.ssh/id_rsa RSA
SHA256:3ajVyNxQuKlj99g+gHf/FaE4Ykv0XRgDAuU3WiXsxUE
Enter passphrase for key '/home/utilisateur/.ssh/id_rsa':
debug1: Authentication succeeded (publickey).
Authenticated to github.com ([140.82.121.4]:22).
debug1: channel 0: new [client-session]
debug1: Entering interactive session.
debug1: pledge: network
debug1: Sending environment.
debug1: Sending env LANG = en_US.UTF-8
PTY allocation request failed on channel 0
debug1: client_input_channel_req: channel 0 rtype exit-status reply 0
Hi utilisateur! You've successfully authenticated, but GitHub does not provide shell access.
debug1: channel 0: free: client-session, nchannels 1
Connection to github.com closed.
Transferred: sent 3784, received 2744 bytes, in 0.2 seconds
Bytes per second: sent 16162.4, received 11720.3
debug1: Exit status 1
```

Accès communs : keys et token

Sécurité, authentification et autorisations

Expression du besoin

- Déléguer un droit de lecture ou écriture à un individu ou une machine, pour un dépôt spécifique
- Donner accès en https
- Donner accès en ssh

SSH : le principe

- Échange de clés de chiffrement symétrique à l'aide d'un chiffrement asymétrique (coûteux, opération appelée poignée de main)
- Une connection (transport) est alors ouverte, chiffré avec le chiffrement symétrique issu de la première opération (et donc moins coûteux)

<https://www.youtube.com/watch?v=gclyVcmpRaM>

Utilisation d'une clé ssh spécifique

```
utilisateur@machine:~/$ ssh-keygen -t rsa -b 4096 -C "deploy key" -f ~/.ssh/deploy_key -N ""
```

```
utilisateur@machine:~/$ ssh -i ~/.ssh/deploy_key github.com
```

Warning: Permanently added the RSA host key for IP address '140.82.121.3' to the list of known hosts.

root@github.com: Permission denied (publickey).

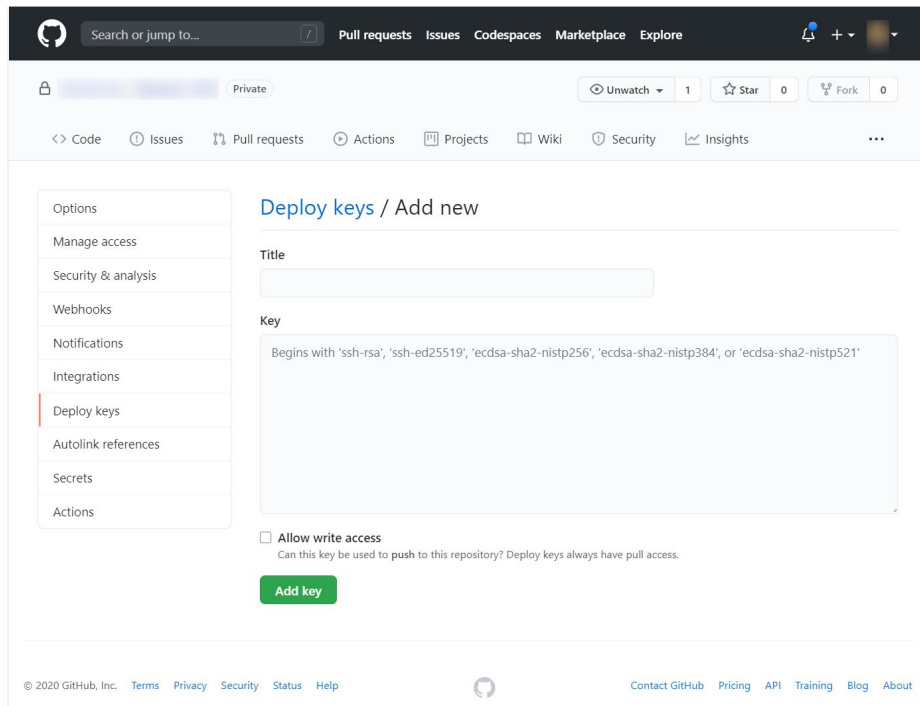
```
utilisateur@machine:~/$ ssh -i ~/.ssh/deploy_key gitlab.com
```

Warning: Permanently added the ECDSA host key for IP address '172.65.251.78' to the list of known hosts.

root@gitlab.com: Permission denied (publickey).

Accès lecture (et écriture) en ssh sur Github

<https://github.com/utilisateur/depot/settings/keys/new>



The screenshot shows the GitHub interface for adding a new deploy key to a repository. The top navigation bar includes the GitHub logo, a search bar, and links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below this, the repository name is shown as 'Private', along with 'Unwatch' (1), 'Star' (0), and 'Fork' (0) buttons. A secondary navigation bar contains links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, and Insights. On the left, a sidebar lists various settings: Options, Manage access, Security & analysis, Webhooks, Notifications, Integrations, Deploy keys (highlighted with a red bar), Autolink references, Secrets, and Actions. The main content area is titled 'Deploy keys / Add new' and contains a form with a 'Title' field, a 'Key' field with a placeholder text 'Begins with 'ssh-rsa', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'', and a checkbox for 'Allow write access' with a note 'Can this key be used to push to this repository? Deploy keys always have pull access.' A green 'Add key' button is at the bottom of the form. The footer shows copyright information for 2020 GitHub, Inc. and links for Terms, Privacy, Security, Status, Help, Contact GitHub, Pricing, API, Training, Blog, and About.

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

Private Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

Options
Manage access
Security & analysis
Webhooks
Notifications
Integrations
Deploy keys
Autolink references
Secrets
Actions

Deploy keys / Add new

Title

Key

Begins with 'ssh-rsa', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

☐ Allow write access
Can this key be used to push to this repository? Deploy keys always have pull access.

Add key

© 2020 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

Vérification git ssh sur Github avec deploy key

```
utilisateur@machine:~/$ ssh -i ~/.ssh/deploy_key git@github.com
```

```
PTY allocation request failed on channel 0
```

```
Hi utilisateur/depot! You've successfully authenticated, but GitHub does not provide shell access.
```

```
utilisateur@machine:~/$ git clone git@github.com:utilisateur/depot.git
```

```
Cloning into 'depot'...
```

```
git@github.com: Permission denied (publickey).
```

```
fatal: Could not read from remote repository.
```

Please make sure you have the correct access rights

and the repository exists.

```
utilisateur@machine:~/$ GIT_SSH_COMMAND='ssh -i /root/.ssh/deploy_key -o IdentitiesOnly=yes' git clone
```

```
git@github.com:utilisateur/depot.git
```

```
Cloning into 'depot'...
```

```
warning: You appear to have cloned an empty repository.
```

Utiliser un ssh config

```
~/.ssh/config
```

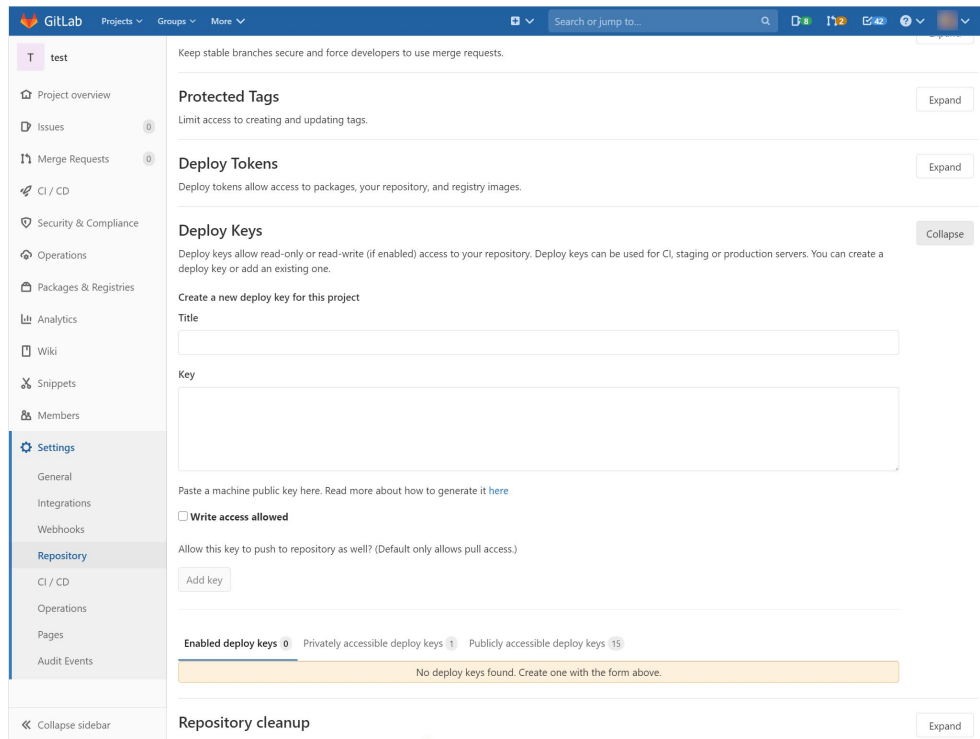
```
Host github.com
HostName github.com
User git
IdentityFile ~/.ssh/deploy_key_github
IdentitiesOnly=yes
```

```
Host gitlab.com
HostName gitlab.com
User git
IdentityFile ~/.ssh/deploy_key_gitlab
IdentitiesOnly=yes
```

```
Host unautredepot
HostName gitlab.com
User git
IdentityFile ~/.ssh/deploy_key_gitlab_2
IdentitiesOnly=yes
# git remote add unautredepot unautredepot:utilisateur/depot.git
```

Accès lecture (et écriture) en ssh sur Gitlab

<https://gitlab.com/utilisateur/depot/-/settings/repository#js-deploy-keys-settings>



Vérification git ssh sur Gitlab avec deploy key

```
utilisateur@machine:~/$ ssh -i ~/.ssh/deploy_key git@gitlab.com
PTY allocation request failed on channel 0
Welcome to GitLab, @utilisateur!
Connection to gitlab.com closed.
```

```
utilisateur@machine:~/$ git clone git@gitlab.com:utilisateur/depot.git
Cloning into 'depot'...
git@gitlab.com: Permission denied (publickey).
fatal: Could not read from remote repository.
```

Please make sure you have the correct access rights
and the repository exists.

```
utilisateur@machine:~/$ GIT_SSH_COMMAND='ssh -i /root/.ssh/deploy_key -o IdentitiesOnly=yes' git clone
git@gitlab.com:utilisateur/depot.git
Cloning into 'depot'...
warning: You appear to have cloned an empty repository.
```

Accès lecture (et écriture) en https sur Gitlab

<https://gitlab.com/{utilisateur}/{depot}/-/settings/repository#js-deploy-tokens>

The screenshot shows the GitLab web interface for the 'Deploy Tokens' settings of a project. The left sidebar contains navigation links: Project overview, Issues, Merge Requests, CI / CD, Security & Compliance, Operations, Packages & Registries, Analytics, Wiki, Snippets, and Members. The 'Settings' section is expanded, showing sub-sections: General, Integrations, Webhooks, Repository (selected), CI / CD, Operations, Pages, and Audit Events. The main content area is titled 'Deploy Tokens' and includes a description: 'Deploy tokens allow access to packages, your repository, and registry images.' Below this is a section 'Add a deploy token' with a prompt to 'Pick a name for the application, and we'll give you a unique deploy token.' There are three input fields: 'Name', 'Expires at', and 'Username'. The 'Scopes' section is checked for 'read_repository' (Allows read-only access to the repository) and has checkboxes for 'read_registry', 'write_registry', 'read_package_registry', and 'write_package_registry'. A green 'Create deploy token' button is present. At the bottom, it shows 'Active Deploy Tokens (0)' and a message: 'This project has no active Deploy Tokens.' The 'Deploy Keys' section is partially visible at the bottom.

GitLab Projects Groups More

test

Project overview

Issues

Merge Requests

CI / CD

Security & Compliance

Operations

Packages & Registries

Analytics

Wiki

Snippets

Members

Settings

General

Integrations

Webhooks

Repository

CI / CD

Operations

Pages

Audit Events

Limit access to creating and updating tags.

Deploy Tokens

Deploy tokens allow access to packages, your repository, and registry images.

Add a deploy token

Pick a name for the application, and we'll give you a unique deploy token.

Name

Expires at

Username

Default format is "gitlab+deploy-token-{n}". Enter custom username if you want to change it.

Scopes

☒ read_repository
Allows read-only access to the repository

☐ read_registry
Allows read-only access to the registry images

☐ write_registry
Allows write access to the registry images

☐ read_package_registry
Allows read access to the package registry

☐ write_package_registry
Allows write access to the package registry

Create deploy token

Active Deploy Tokens (0)

This project has no active Deploy Tokens.

Deploy Keys

Vérification git https sur Gitlab avec deploy token

```
utilisateur@machine:~/$ git clone https://gitlab.com/utilisateur/depot.git
Cloning into 'depot'...
Username for 'https://gitlab.com': token-utilisateur
Password for 'https://token-utilisateur@gitlab.com':
warning: You appear to have cloned an empty repository.
```

```
utilisateur@machine:~/$ git clone https://token-utilisateur:WW6euauQwCwqtiDwhJas@gitlab.com/tdutrion/test.git
Cloning into 'test'...
warning: You appear to have cloned an empty repository.
```

!! Attention : utiliser la syntaxe utilisateur:mot-de-passe@hôte va laisser des traces dans votre sh/bash/zsh/... history, et laisser apparaître le mot de passe (token) en clair sur votre écran.

Cette manipulation est donc fortement déconseillée !

Protéger branches et tags


Sécurité, authentification et autorisations

Pour quoi faire ?

- Limiter les personnes pouvant envoyer du code sur une branche
- Limiter les personnes pouvant créer un tag
- Forcer le respect du workflow choisi
- Utiliser des patterns pour cibler certains groupes de branches ou tags

Protection des branches

<https://github.com/utilisateur/depot/settings/branches>

 Search or jump to... Pull requests Issues Codespaces Marketplace Explore

Private

Unwatch 1 Star 0 Fork 0

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

Options

Manage access

Security & analysis

Branches

Webhooks

Notifications

Integrations

Deploy keys

Autolink references

Secrets

Actions

Default branch

The default branch is considered the “base” branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

The default branch is set to `master`. To change this setting, [add another branch](#).

Branch protection rules

Add rule

Define branch protection rules to disable force pushing, prevent branches from being deleted, and optionally require status checks before merging. New to branch protection rules? [Learn more](#).

No branch protection rules defined yet.

© 2020 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub Pricing API Training Blog About

Options
Manage access
Security & analysis
Branches
Webhooks
Notifications
Integrations
Deploy keys
Autolink references
Secrets
Actions

Branch protection rule

Branch name pattern

Protect matching branches

☐ Require pull request reviews before merging

When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

☐ Require status checks to pass before merging

Choose which [status checks](#) must pass before branches can be merged into a branch that matches this rule. When enabled, commits must first be pushed to another branch, then merged or pushed directly to a branch that matches this rule after status checks have passed.

☐ Require signed commits

Commits pushed to matching branches must have verified signatures.

☐ Require linear history

Prevent merge commits from being pushed to matching branches.

☐ Include administrators

Enforce all configured restrictions above for administrators.

Rules applied to everyone including administrators

☐ Allow force pushes

Permit force pushes for all users with push access.

☐ Allow deletions

Allow users with push access to delete matching branches.

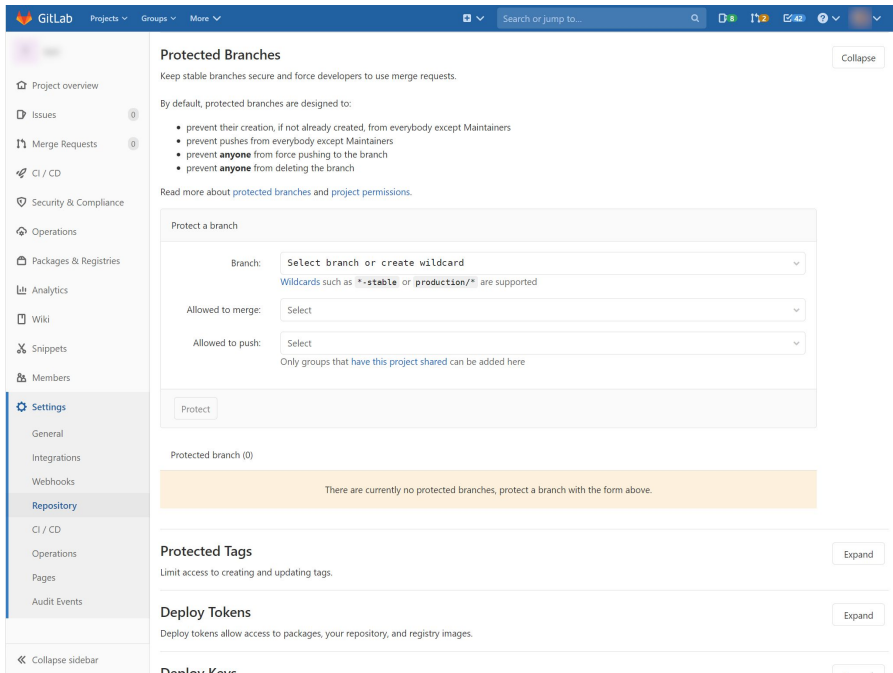
Create

Push sur une branche protégée Github

```
utilisateur@machine:~/$ GIT_SSH_COMMAND='ssh -i ~/.ssh/deploy_key -o IdentitiesOnly=yes' git push github master
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote: error: GH006: Protected branch update failed for refs/heads/master.
remote: error: Cannot force-push to this protected branch
To github.com:utilisateur/depot.git
! [remote rejected] master -> master (protected branch hook declined)
error: failed to push some refs to 'git@github.com:utilisateur/depot.git'
```

Protection des branches et tags

<https://gitlab.com/tdutrien/test/-/settings/repository#js-protected-branches-settings>



The screenshot shows the 'Protected Branches' settings in GitLab. The left sidebar contains navigation links: Project overview, Issues, Merge Requests, CI / CD, Security & Compliance, Operations, Packages & Registries, Analytics, Wiki, Snippets, Members, Settings (selected), General, Integrations, Webhooks, Repository (selected), CI / CD, Operations, Pages, and Audit Events. The main content area is titled 'Protected Branches' and includes a 'Collapse' button. It contains a description, default rules, a link to read more, a 'Protect a branch' form, and a section for 'Protected Tags' with an 'Expand' button. The 'Protect a branch' form has fields for 'Branch' (with a dropdown and wildcard examples), 'Allowed to merge' (dropdown), and 'Allowed to push' (dropdown). Below the form is a 'Protect' button. The 'Protected Tags' section is currently empty, showing a message: 'There are currently no protected tags, protect a tag with the form above.'

Protected Branches Collapse

Keep stable branches secure and force developers to use merge requests.

By default, protected branches are designed to:

- prevent their creation, if not already created, from everybody except Maintainers
- prevent pushes from everybody except Maintainers
- prevent **anyone** from force pushing to the branch
- prevent **anyone** from deleting the branch

Read more about [protected branches](#) and [project permissions](#).

Protect a branch

Branch:
Wildcards such as `*-stable` or `production/*` are supported

Allowed to merge:

Allowed to push:
Only groups that [have this project shared](#) can be added here

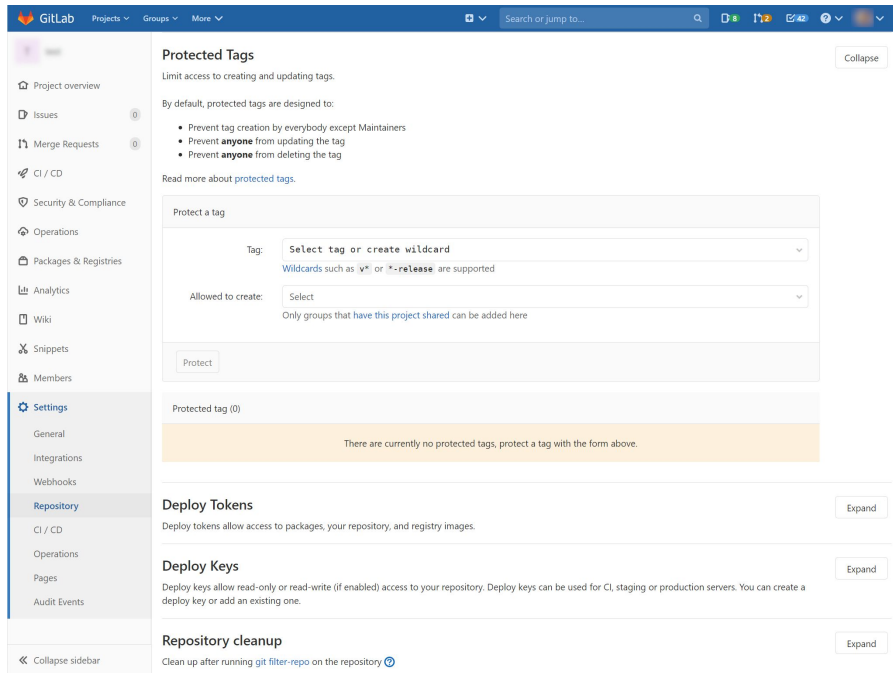
Protected branch (0)

There are currently no protected branches, protect a branch with the form above.

Protected Tags Expand

Limit access to creating and updating tags.

<https://gitlab.com/tdutrien/test/-/settings/repository#js-protected-tags-settings>



The screenshot shows the 'Protected Tags' settings in GitLab. The left sidebar is identical to the previous screenshot. The main content area is titled 'Protected Tags' and includes a 'Collapse' button. It contains a description, default rules, a link to read more, a 'Protect a tag' form, and a section for 'Protected tags' with an 'Expand' button. The 'Protect a tag' form has fields for 'Tag' (with a dropdown and wildcard examples) and 'Allowed to create' (dropdown). Below the form is a 'Protect' button. The 'Protected tags' section is currently empty, showing a message: 'There are currently no protected tags, protect a tag with the form above.'

Protected Tags Collapse

Limit access to creating and updating tags.

By default, protected tags are designed to:

- Prevent tag creation by everybody except Maintainers
- Prevent **anyone** from updating the tag
- Prevent **anyone** from deleting the tag

Read more about [protected tags](#).

Protect a tag

Tag:
Wildcards such as `*-stable` or `production/*` are supported

Allowed to create:
Only groups that [have this project shared](#) can be added here

Protected tag (0)

There are currently no protected tags, protect a tag with the form above.

Deploy Tokens Expand

Deploy tokens allow access to packages, your repository, and registry images.

Deploy Keys Expand

Deploy keys allow read-only or read-write (if enabled) access to your repository. Deploy keys can be used for CI, staging or production servers. You can create a deploy key or add an existing one.

Repository cleanup Expand

Clean up after running [git filter-repo](#) on the repository

Push sur une branche protégée Gitlab

```
utilisateur@machine:~/$ GIT_SSH_COMMAND='ssh -i ~/.ssh/deploy_key -o IdentitiesOnly=yes' git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 251 bytes | 251.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: GitLab: You are not allowed to push code to protected branches on this project.
To gitlab.com:utilisateur/depot.git
! [remote rejected] master -> master (pre-receive hook declined)
error: failed to push some refs to 'git@gitlab.com:utilisateur/depot.git'
```

Commits signés

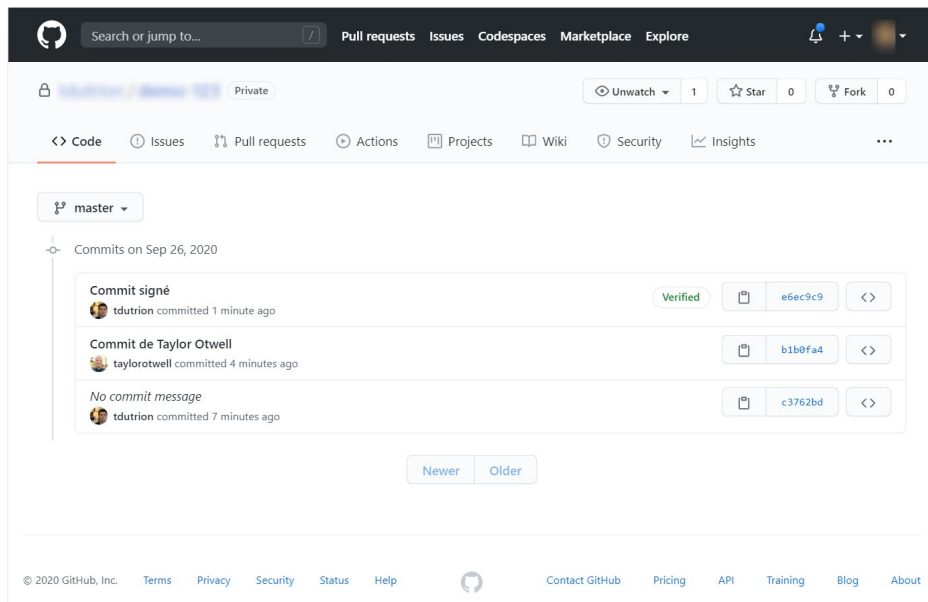
Sécurité, authentification et autorisations

Usurpation d'identité

- Git ne gère pas les utilisateurs
- Github et Gitlab gèrent l'authentification et autorisation
 - sur le couple nom d'utilisateur et mot de passe
 - sur la clé ssh
 - sur un couple token id et token

Usurpation d'identité

```
git init
git config user.name "Taylor Otwell"
git config user.email taylor@laravel.com
```



Search or jump to... Pull requests Issues Codespaces Marketplace Explore

Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

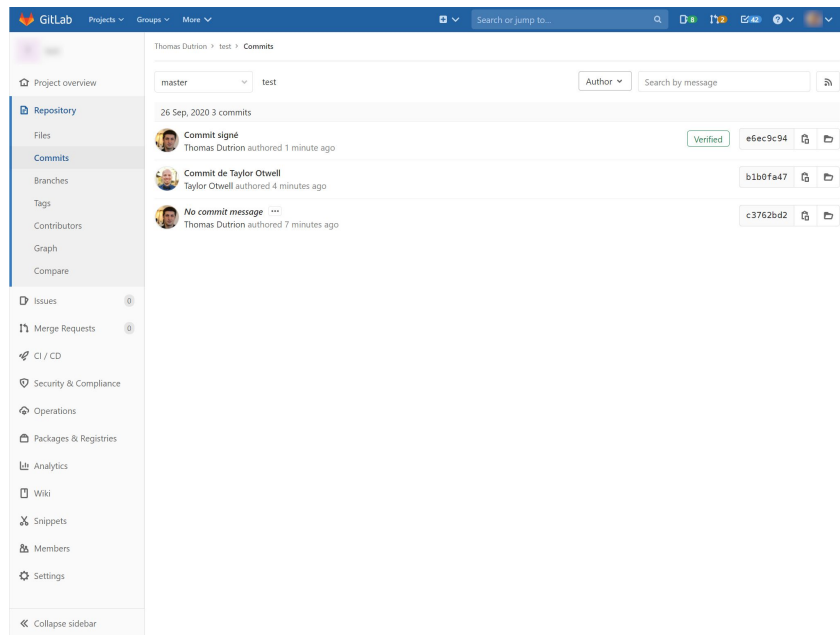
master

Commits on Sep 26, 2020

- Commit signé** Verified e6ec9c9 <>
tdutrition committed 1 minute ago
- Commit de Taylor Otwell** b1b0fa4 <>
taylorotwell committed 4 minutes ago
- No commit message** c3762bd <>
tdutrition committed 7 minutes ago

Newer Older

© 2020 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About



GitLab Projects Groups More Search or jump to...

Project overview

Repository

- Files
- Commits
- Branches
- Tags
- Contributors
- Graph
- Compare

Issues Merge Requests CI / CD Security & Compliance Operations Packages & Registries Analytics Wiki Snippets Members Settings

Thomas Dutrion test Commits

master test Author Search by message

26 Sep, 2020 3 commits

- Commit signé** Verified e6ec9c9 <>
Thomas Dutrion authored 1 minute ago
- Commit de Taylor Otwell** b1b0fa4 <>
Taylor Otwell authored 4 minutes ago
- No commit message** c3762bd <>
Thomas Dutrion authored 7 minutes ago

« Collapse sidebar

La solution : PGP

Micode : Booba utilise PGP : <https://www.youtube.com/watch?v=EoxV52jtB7c>

- Chiffrement (exemple mails, messageries instantannées, <https://www.facebook.com/notes/protect-the-graph/securing-email-communications-from-facebook/1611941762379302/>)
- Signature
 - <https://docs.github.com/en/free-pro-team@latest/github/authenticating-to-github/signing-commits>
 - https://docs.gitlab.com/ee/user/project/repository/gpg_signed_commits/

Générer une clé PGP (1/4)

```
utilisateur@machine:~/$ gpg --full-gen-key  
gpg (GnuPG) 2.2.23; Copyright (C) 2020 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.
```

```
gpg: directory '/home/utilisateur/.gnupg' created  
gpg: keybox '/home/utilisateur/.gnupg/pubring.kbx' created  
Please select what kind of key you want:  
  (1) RSA and RSA (default)  
  (2) DSA and Elgamal  
  (3) DSA (sign only)  
  (4) RSA (sign only)  
  (14) Existing key from card  
Your selection? 1
```

Générer une clé PGP (2/4)

RSA keys may be between 1024 and 4096 bits long.

What keysize do you want? (3072) **4096**

Requested keysize is 4096 bits

Please specify how long the key should be valid.

0 = key does not expire

<n> = key expires in n days

<n>w = key expires in n weeks

<n>m = key expires in n months

<n>y = key expires in n years

Key is valid for? (0) **0**

Key does not expire at all

Is this correct? (y/N) **y**

GnuPG needs to construct a user ID to identify your key.

Real name: **Utilisateur De La Machine**

Email address: **utilisateur@machine.fr**

Comment: **démo**

Générer une clé PGP (3/4)

You selected this USER-ID:

```
"Utilisateur De La Machine (d mo) <utilisateur@machine.fr>"
```

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? **O**

We need to generate a lot of random bytes. It is a good idea to perform some other action (type on the keyboard, move the mouse, utilize the disks) during the prime generation; this gives the random number generator a better chance to gain enough entropy.

We need to generate a lot of random bytes. It is a good idea to perform some other action (type on the keyboard, move the mouse, utilize the disks) during the prime generation; this gives the random number generator a better chance to gain enough entropy.

```
gpg: /home/utilisateur/.gnupg/trustdb.gpg: trustdb created
```

```
gpg: key A2A1B12FB3EBFA87 marked as ultimately trusted
```

```
gpg: directory '/home/utilisateur/.gnupg/openpgp-revocs.d' created
```

```
gpg: revocation certificate stored as
```

```
'/home/utilisateur/.gnupg/openpgp-revocs.d/1472CA3B3A97E01D42372838A2A1B12FB3EBFA87.rev'
```

```
public and secret key created and signed.
```

Générer une clé PGP (4/4)

```
pub   rsa4096 2020-09-27 [SC]
      1472CA3B3A97E01D42372838A2A1B12FB3EBFA87
uid           Utilisateur De La Machine (d  mo) <utilisateur@machine.fr>
sub    rsa4096 2020-09-27 [E]
```

Générer une clé PGP (4/4)

```
utilisateur@machine:~/$ gpg --list-secret-keys --keyid-format LONG utilisateur@machine.fr
gpg: checking the trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
sec   rsa4096/A2A1B12FB3EBFA87 2020-09-27 [SC]
      1472CA3B3A97E01D42372838A2A1B12FB3EBFA87
uid                               [ultimate] Thomas Dutrion (tdutrion) <thomas@engineor.com>
ssb   rsa4096/A54ED7277CBE4BB5 2020-09-27 [E]

utilisateur@machine:~/$ gpg --armor --export A2A1B12FB3EBFA87
```

Copier la sortie sur

- <https://github.com/settings/gpg/new>
- https://gitlab.com/profile/gpg_keys

Signer un commit

```
utilisateur@machine:~/$ git config --global user.signingkey A2A1B12FB3EBFA87
```

```
utilisateur@machine:~/$ # git config --global gpg.program gpg2 # facultatif
```

```
utilisateur@machine:~/$ # export GPG_TTY=$(tty) # facultatif
```

```
utilisateur@machine:~/$ git commit -S -m "Signed commit"
```

```
utilisateur@machine:~/$ git config --global commit.gpgsign true
```

```
utilisateur@machine:~/$ git commit -m "Signed commit"
```

Troubleshooting

```
utilisateur@machine:~/$ echo "test" | gpg --clearsign
```

```
utilisateur@machine:~/$ echo "test" | gpg2 --clearsign
```

[https://www.gnupg.org/\(it\)/documentation/manuals/gnupg/Common-Problems.html](https://www.gnupg.org/(it)/documentation/manuals/gnupg/Common-Problems.html)

Commit signés sur Github

Search or jump to... / Pulls Issues Codespaces Marketplace Explore

tdtutrition / demo-123 Private Unwatch 1 Star 0 Fork 0

<> Code ⓘ Issues 🔗 Pull requests ▶ Actions 📁 Projects 📖 Wiki ...

master

Commits on Sep 27, 2020

Wrong email address	Unverified	df63e9c	<>
tdtutrition committed 22 minutes ago			
Email address not mapping		f80a6	<>
tdtutrition committed 26 minutes ago			
Signed commit		803a9	<>
tdtutrition committed 30 minutes ago			
Non signed commit		703bd19	<>
tdtutrition committed 32 minutes ago			

Newer Older

© 2020 GitHub, Inc. Terms Privacy Security Status Help
Contact GitHub Pricing API Training Blog About

Search or jump to... / Pulls Issues Codespaces Marketplace Explore

tdtutrition / demo-123 Private Unwatch 1 Star 0 Fork 0

<> Code ⓘ Issues 🔗 Pull requests ▶ Actions 📁 Projects 📖 Wiki ...

master

Commits on Sep 27, 2020

Wrong email address	Unverified	df63e9c	<>
tdtutrition committed 23 minutes ago			
Email address not mapping	Unverified	76f80a6	<>
tdtutrition committed 27 minutes ago			
Signed commit		803a9	<>
tdtutrition committed 31 minutes ago			
Non signed commit		8bd19	<>
tdtutrition committed 33 minutes ago			

Newer Older

© 2020 GitHub, Inc. Terms Privacy Security Status Help
Contact GitHub Pricing API Training Blog About

Commit signé sur Github

The screenshot shows the GitHub interface for the repository `tdutrion / demo-123`. The page displays a list of commits on the `master` branch. The commits are:

- Wrong email address**: tdutrion committed 24 minutes ago. Status: Unverified. SHA: `df63e9c`.
- Email address not mapping**: tdutrion committed 28 minutes ago. Status: Unverified. SHA: `76f80a6`.
- Signed commit**: tdutrion committed 32 minutes ago. Status: **Verified**. SHA: `07903a9`.
- Non signed commit**: tdutrion committed 34 minutes ago. Status: Unverified. SHA: `...`.

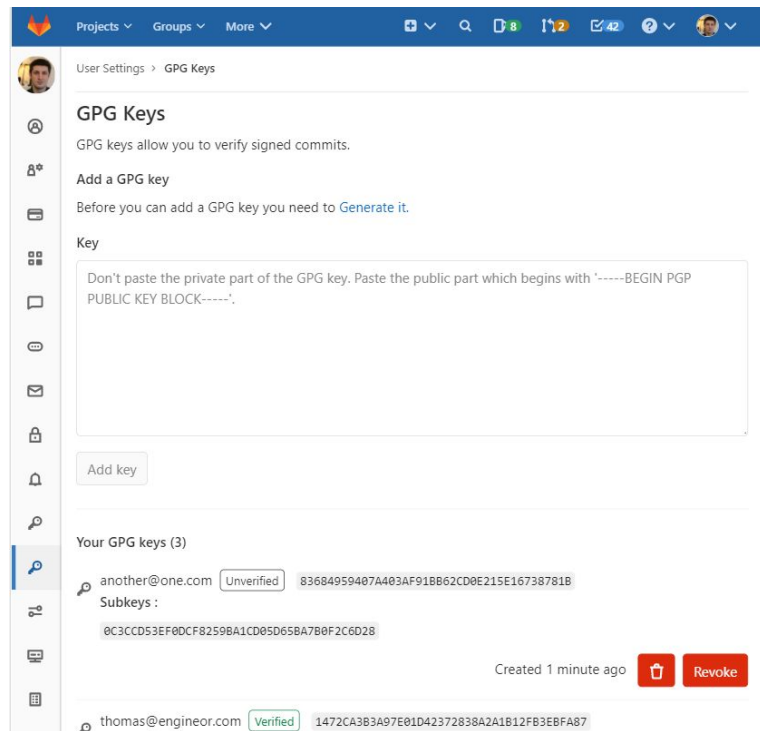
A tooltip for the 'Signed commit' shows:

- A green checkmark icon.
- Text: "This commit was signed with a **verified signature**."
- Profile picture and name: **tdutrion** Thomas Dutrion.
- GPG key ID: `A2A1B12FB3EBFA87`.
- Link: [Learn about signing commits](#).

At the bottom of the page, the footer contains:

© 2020 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#)
[Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

Commit signé sur Gitlab



The screenshot shows the 'GPG Keys' section of a user's settings in GitLab. It includes instructions on how to add a GPG key, a text area for pasting the public key, and a list of existing keys. One key is listed as 'Unverified' and another as 'Verified'.

User Settings > GPG Keys

GPG Keys

GPG keys allow you to verify signed commits.

Add a GPG key

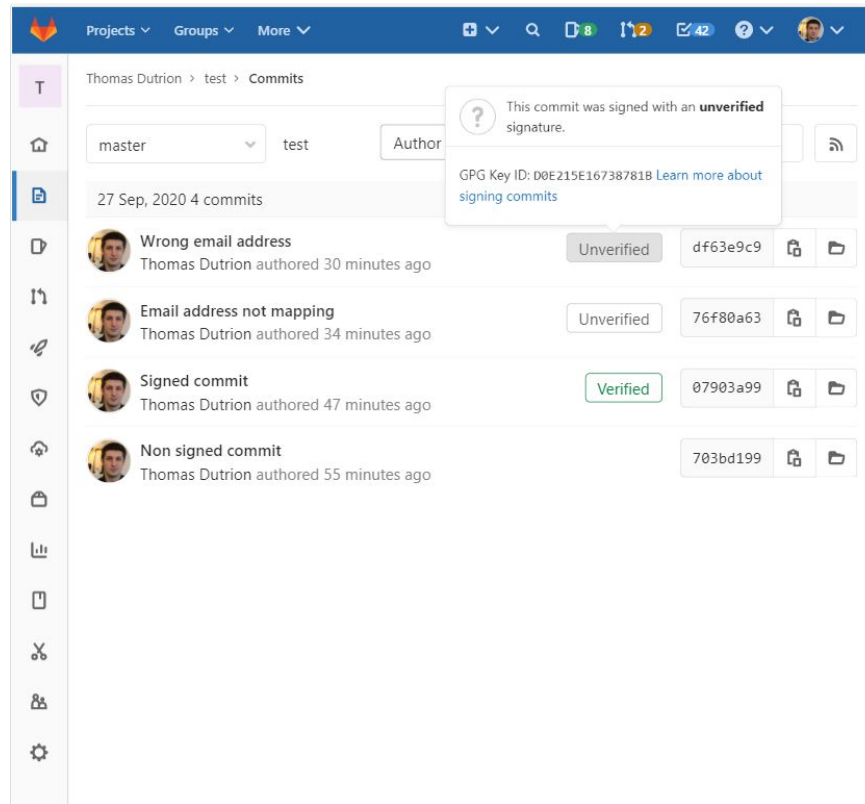
Before you can add a GPG key you need to [Generate it](#).

Key

Don't paste the private part of the GPG key. Paste the public part which begins with '-----BEGIN PGP PUBLIC KEY BLOCK-----',

Your GPG keys (3)

Key	Status	Created
another@one.com Subkeys : 0C3CCD53EF0DCF82598A1CD05D658A7B0F2C6D28	Unverified	Created 1 minute ago
thomas@engineor.com 1472CA3B3A97E01D42372838A2A1B12FB3E8FA87	Verified	



The screenshot shows the 'Commits' page for a user named Thomas Dutrion. It displays a list of commits with their status (Unverified or Verified) and a tooltip explaining that a commit was signed with an unverified signature.

Thomas Dutrion > test > Commits

master test Author

27 Sep, 2020 4 commits

Commit	Status	Key ID
Wrong email address Thomas Dutrion authored 30 minutes ago	Unverified	df63e9c9
Email address not mapping Thomas Dutrion authored 34 minutes ago	Unverified	76f80a63
Signed commit Thomas Dutrion authored 47 minutes ago	Verified	07903a99
Non signed commit Thomas Dutrion authored 55 minutes ago		703bd199

Tooltip: This commit was signed with an **unverified** signature.
GPG Key ID: D0E215E16738781B [Learn more about signing commits](#)

Commit signé sur Gitlab

```
utilisateur@machine:/# gpg --list-secret-keys --keyid-format LONG another@one.com
sec   rsa4096/D0E215E16738781B 2020-09-27 [SC]
      83684959407A403AF91BB62CD0E215E16738781B
uid           [ultimate] Another One <another@one.com>
ssb   rsa4096/5D65BA7B0F2C6D28 2020-09-27 [E]
```

Commit signé sur Gitlab

The screenshot displays the GitLab web interface. The top navigation bar is blue with the GitLab logo, 'Projects', 'Groups', and 'More' dropdowns. On the right of the bar are icons for adding content, search, repository status (8), merge requests (2), issues (42), help, and a user profile. The left sidebar contains a vertical menu with icons for home, code, merge requests, issues, and other project features. The main content area shows the 'Commits' page for the 'test' branch. It includes filters for 'master' and 'test' branches, an 'Author' dropdown, and a 'Search by message' input. Below these, a commit history table lists four commits by 'Thomas Dutrion'. The first commit, 'Wrong email address', is highlighted. A tooltip is visible over this commit, stating: 'This commit was signed with an **unverified** signature. GPG Key ID: ED89CC2151C77F73 [Learn more about signing commits](#)'. The commit details table shows the commit message, author, time, and a 'Verified' status (green box) next to the commit ID '76f80a63'. The other commits are 'Email address not mapping' (Unverified), 'Signed commit' (Verified), and 'Non signed commit'.

Commit Message	Author	Time	Signature Status	Commit ID
Wrong email address	Thomas Dutrion	30 minutes ago	Unverified	76f80a63
Email address not mapping	Thomas Dutrion	34 minutes ago	Unverified	76f80a63
Signed commit	Thomas Dutrion	47 minutes ago	Verified	07903a99
Non signed commit	Thomas Dutrion	55 minutes ago		703bd199

Commit signé sur Gitlab

The screenshot displays the GitLab web interface. At the top, a blue navigation bar contains the GitLab logo, dropdown menus for 'Projects', 'Groups', and 'More', and icons for repository management, search, and user profile. Below the navigation bar, the breadcrumb path 'Thomas Dutrion > test > Commits' is visible. The left sidebar shows a list of icons for navigation. The main content area shows a list of commits for the 'test' branch. The commits are: 'Wrong email address' (30 minutes ago), 'Email address not mapping' (34 minutes ago), 'Signed commit' (47 minutes ago), and 'Non signed commit' (55 minutes ago). A tooltip is displayed over the 'Signed commit' entry, indicating that the commit was signed with a verified signature and the committer email is verified to belong to the same user. The tooltip also shows the user's profile information: Thomas Dutrion (@tdutrion) and the GPG Key ID: A2A1812FB3EBFA87. The commit hash for the signed commit is 07903a99, and for the non-signed commit, it is 703bd199.

Thomas Dutrion > test > Commits

master test Author Search by message

27 Sep, 2020 4 commits

Wrong email address
Thomas Dutrion authored 30 minutes ago

Email address not mapping
Thomas Dutrion authored 34 minutes ago

Signed commit
Thomas Dutrion authored 47 minutes ago

Non signed commit
Thomas Dutrion authored 55 minutes ago

This commit was signed with a **verified** signature and the committer email is verified to belong to the same user.

Thomas Dutrion
@tdutrion
GPG Key ID: A2A1812FB3EBFA87 [Learn more about signing commits](#)

Verified 07903a99

703bd199

Mise en application

Sécurité, authentification et autorisations

Mise en application

- Pouvoir montrer un commit signé
- Pouvoir envoyer du code avec une clé de déploiement ssh
- Pouvoir lire du code avec une clé de déploiement ssh
- Sécuriser des branches et montrer que les commits ne passent pas