Outils de versionning

ESGI - 3ème année IW 2020

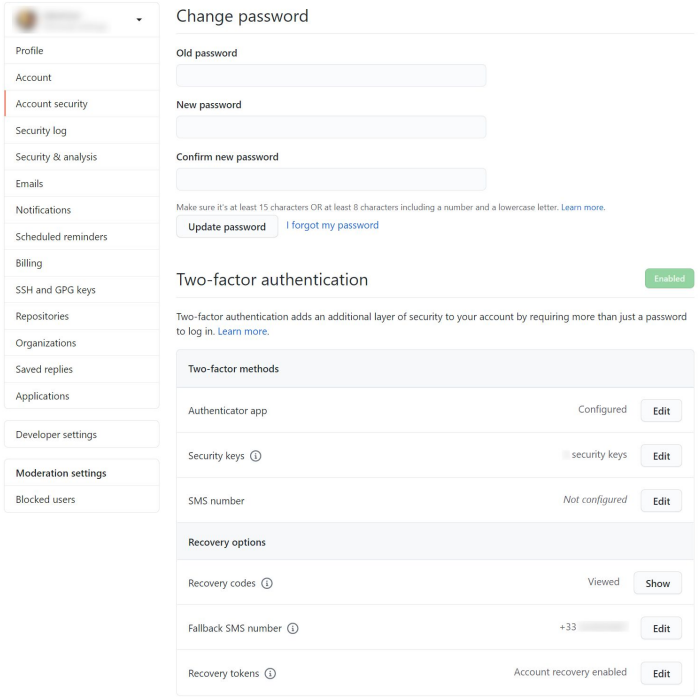
Sécurité, authentification et autorisations

Authentification multi-facteurs

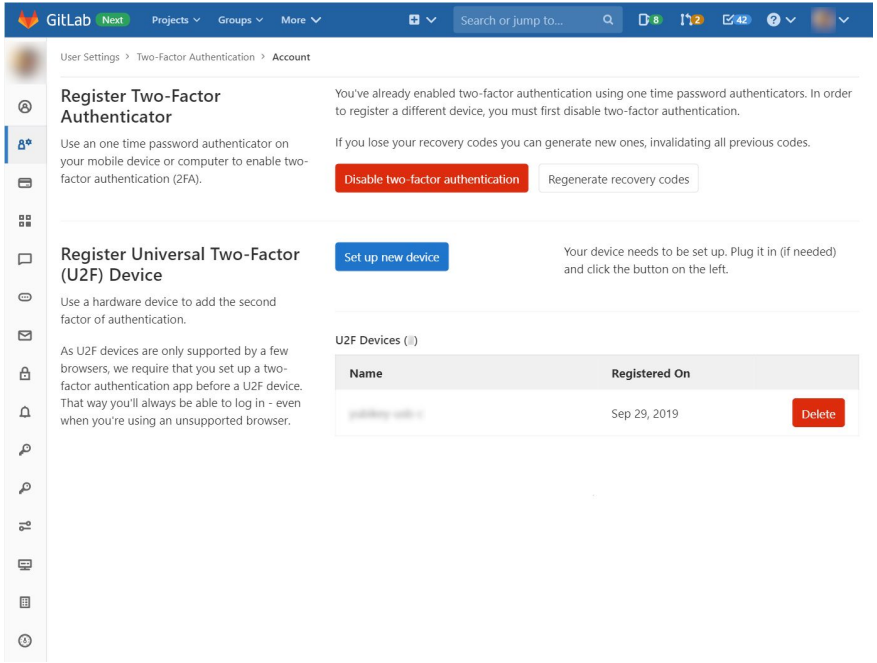
Sécurité, authentification et autorisations

Ajouter des facteurs d’authentification

https://github.com/settings/security

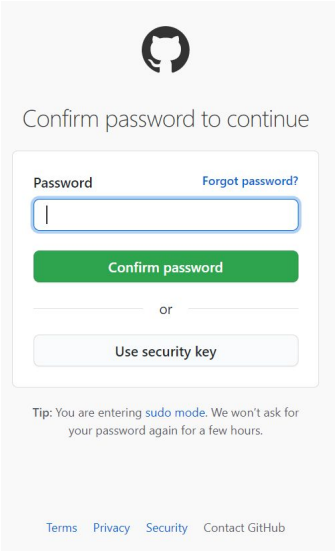


https://gitlab.com/profile/two_factor_auth

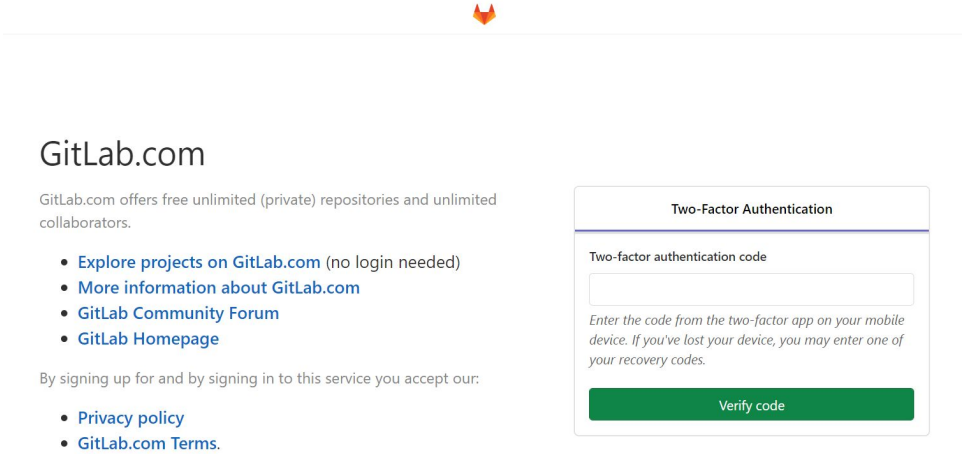


Confirmation de second facteur

Github



Gitlab



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
```

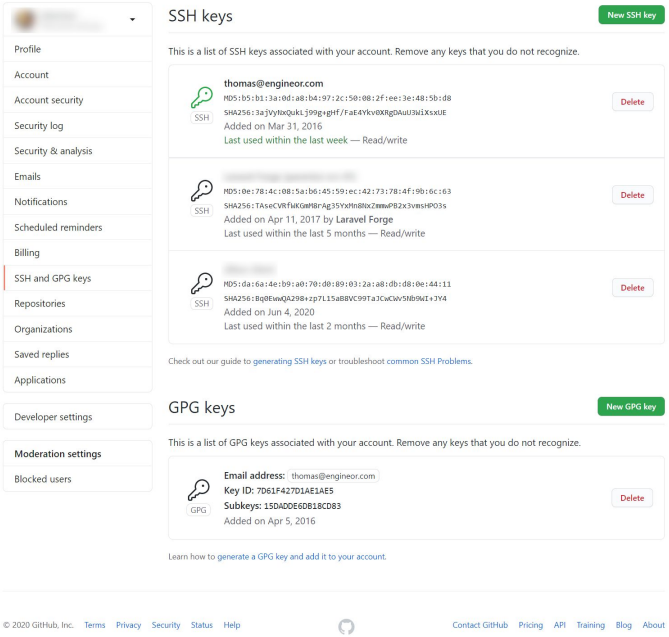
- `.ssh`, chmod 700 : seul le propriétaire peut utiliser ce dossier (x = traverser)
- `.ssh/id_rsa`, chmod 600 : clé privée, seul le propriétaire peut écrire/lire
- `.ssh/id_rsa.pub`, chmod 644 : clé publique, tout le monde peut lire, seul le propriétaire modifie

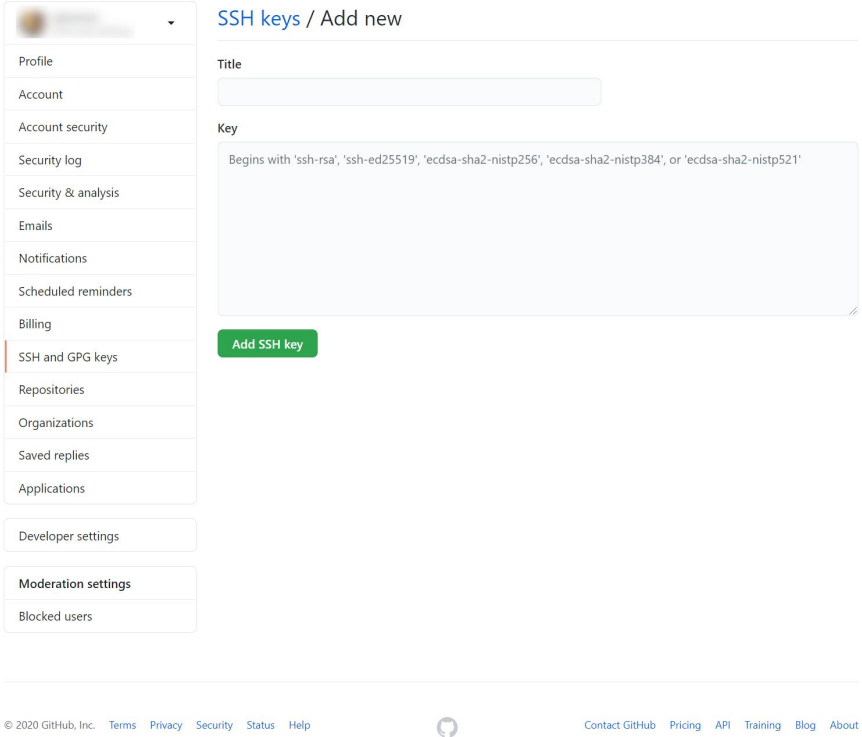
Copier une clé publique

- 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





Vérification de la connection

```
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:nThbg6kXUpJWGl7E1IGOCspRomTxdCARLviKw6E5SY8
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,ec
dsa-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,s
sh-rsa-cert-v01@openssh.com,ssh-dss-cert-v01@openssh.com,ssh-ed25519,ecdsa-sha2-nistp521,ecds
a-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)

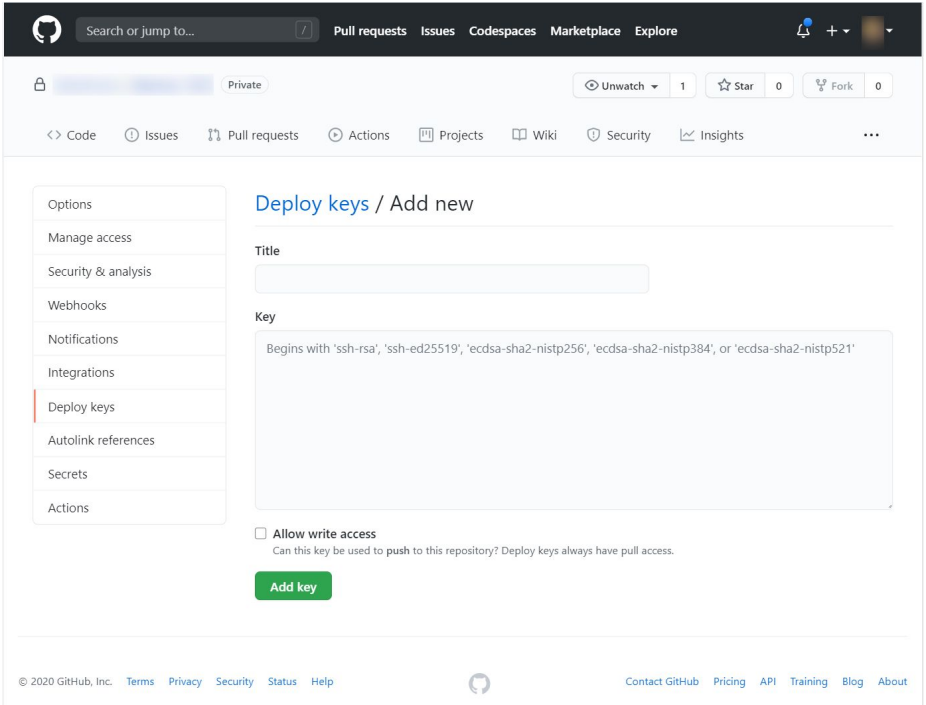
https://www.youtube.com/watch?v=gclyVcmpRaM

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



Vérification git ssh sur Github avec deploy key

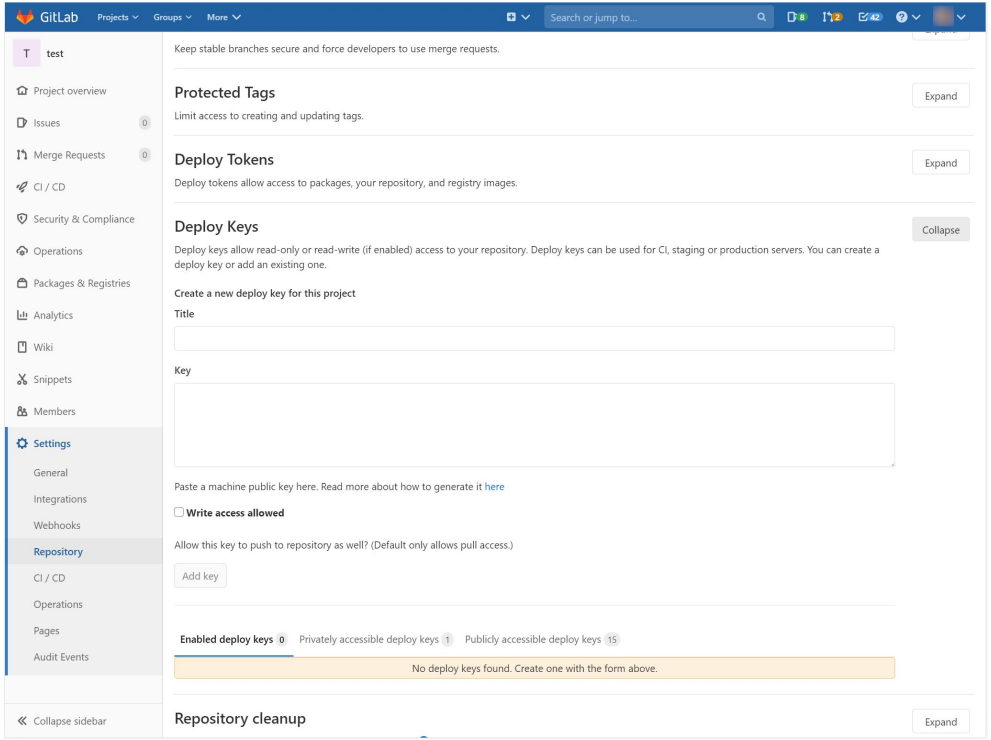
```
utilisateur@machine:~/$ ssh -i ~/.ssh/deploy_key git@github.com
PTY allocation request failed on channel 0
Hi utilisateur/depot! You've successfully authenticated, but GitHub does not provide shell access.

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

GitLab Projects Groups More Search or jump to...

T test

Project overview
Issues 0
Merge Requests 0
CI / CD
Security & Compliance
Operations
Packages & Registries
Analytics
Wiki
Snippets
Members
Settings
General
Integrations
Webhooks
Repository
CI / CD
Operations
Pages
Audit Events
Collapse sidebar

Limit access to creating and updating tags.

Deploy Tokens

Collapse

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

- [x] 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

Expand

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/tdutrion/test/-/settings/repository#js-protected-branches-settings

https://gitlab.com/tdutrion/test/-/settings/repository#js-protected-tags-settings

GitLab Projects Groups More Search or jump to...

Project overview
Issues 0
Merge Requests 0
CI / CD
Security & Compliance
Operations
Packages & Registries
Analytics
Wiki
Snippets
Members
Settings
General
Integrations
Webhooks
Repository
CI / CD
Operations
Pages
Audit Events
Collapse sidebar

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: `Select branch or create wildcard`

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

Allowed to merge: Select

Allowed to push: Select

Only groups that have this project shared can be added here

Protect

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.

Deploy Tokens

Expand

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

Deploy Keys

GitLab Projects Groups More Search or jump to...

Project overview
Issues 0
Merge Requests 0
CI / CD
Security & Compliance
Operations
Packages & Registries
Analytics
Wiki
Snippets
Members
Settings
General
Integrations
Webhooks
Repository
CI / CD
Operations
Pages
Audit Events
Collapse sidebar

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: `Select tag or create wildcard`

Wildcards such as `v*` or `*-release` are supported

Allowed to create: Select

Only groups that have this project shared can be added here

Protect

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

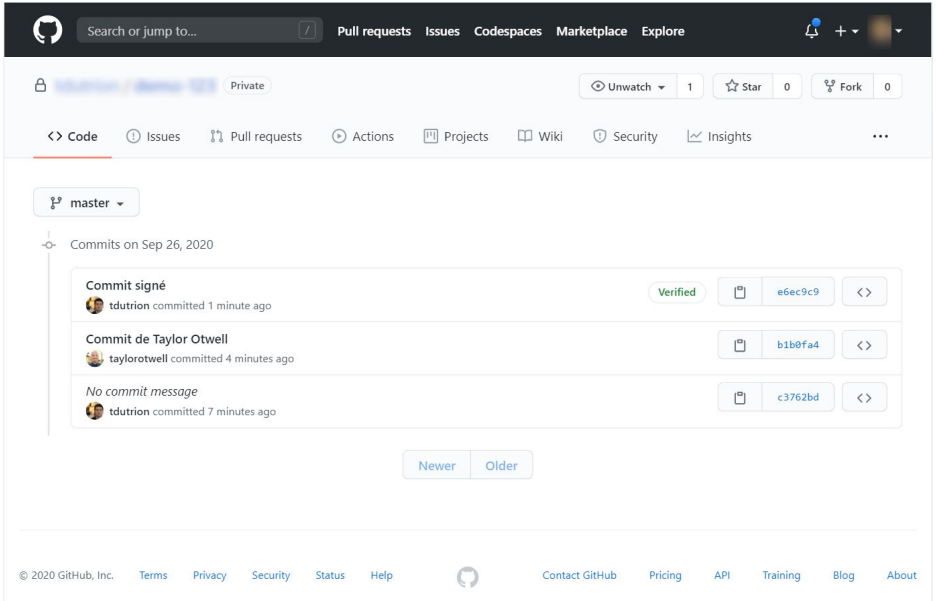
Sécurité, authentification et autorisations

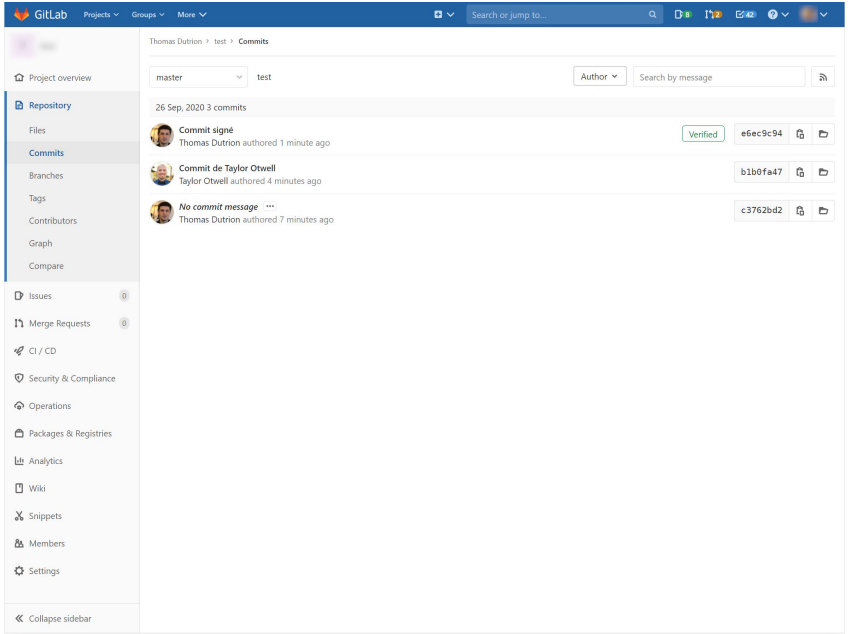
Usurpation d’identité

- 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
```





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 (démo) <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 (démo) <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

Commit signés sur Github

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

tdutrion / 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
tdutrion committed 22 minutes ago
Unverified df63e9c

Email address not mapping
tdutrion committed 26 minutes ago
f80a6

Signed commit
tdutrion committed 30 minutes ago
003a9

Non signed commit
tdutrion committed 32 minutes ago
703bd19

The email in this signature doesn't match the committer email.

GPG key ID: D0E215E16738781B
Learn about signing commits

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

tdutrion / 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
tdutrion committed 23 minutes ago
Unverified df63e9c

Email address not mapping
tdutrion committed 27 minutes ago
Unverified 76f80a6

Signed commit
tdutrion committed 31 minutes ago
003a9

Non signed commit
tdutrion committed 33 minutes ago
3bd19

Upload your public GPG key to verify your signature.

GPG key ID: ED89CC2151C77F73
Learn about signing commits

Newer Older

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

Commit signé sur Github

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

tdutrion / 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
tdutrion committed 24 minutes ago — Unverified — df63e9c

Email address not mapping
tdutrion committed 28 minutes ago — Unverified — 76f80a6

Signed commit
tdutrion committed 32 minutes ago — Verified — 07903a9

Non signed commit
tdutrion committed 34 minutes ago — d19

This commit was signed with a verified signature.

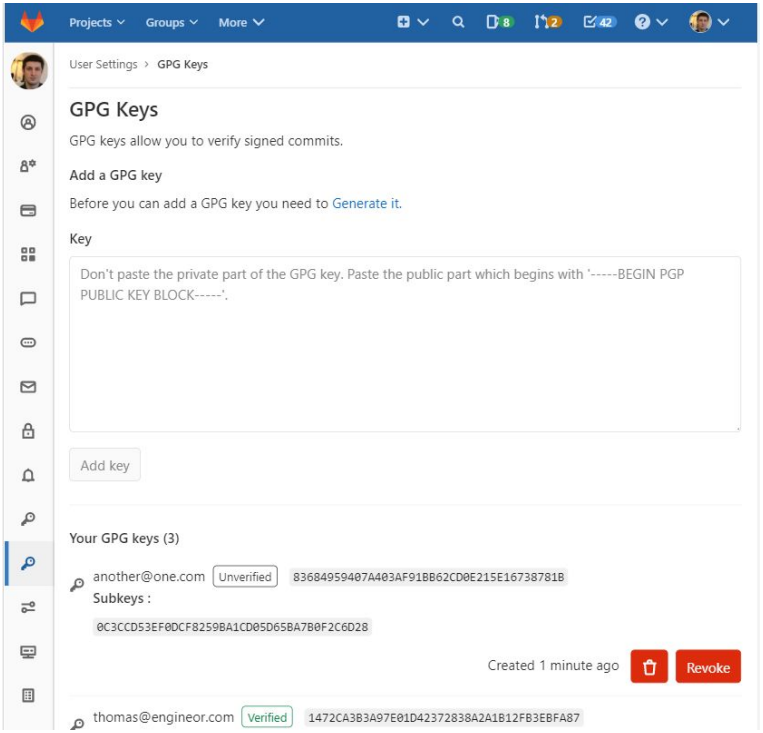
tdutrion
Thomas Dutrion

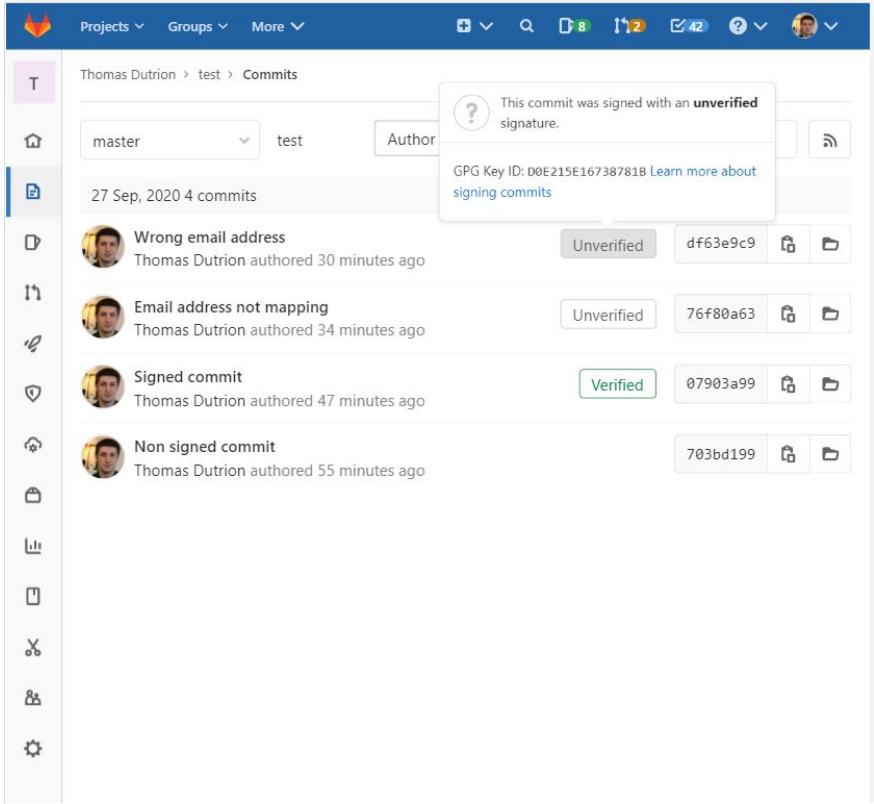
GPG key ID: A2A1B12FB3EBFA87
Learn about signing commits

Newer

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

Commit signé sur Gitlab

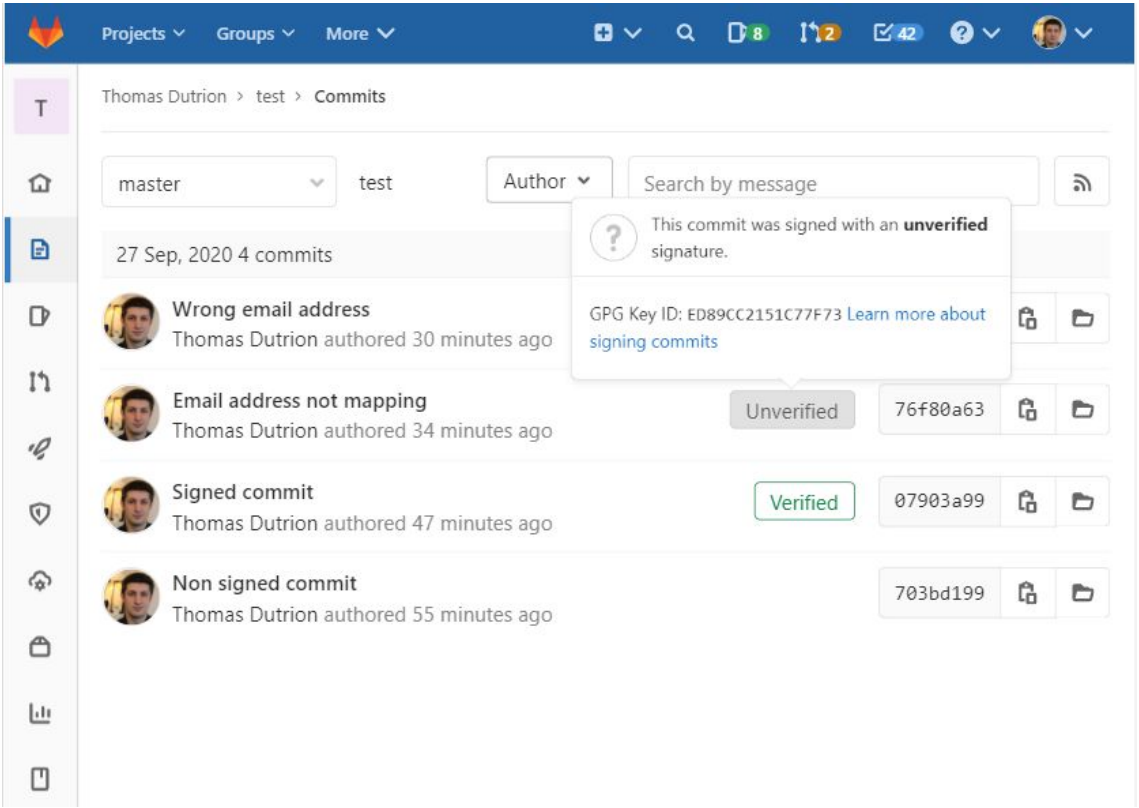




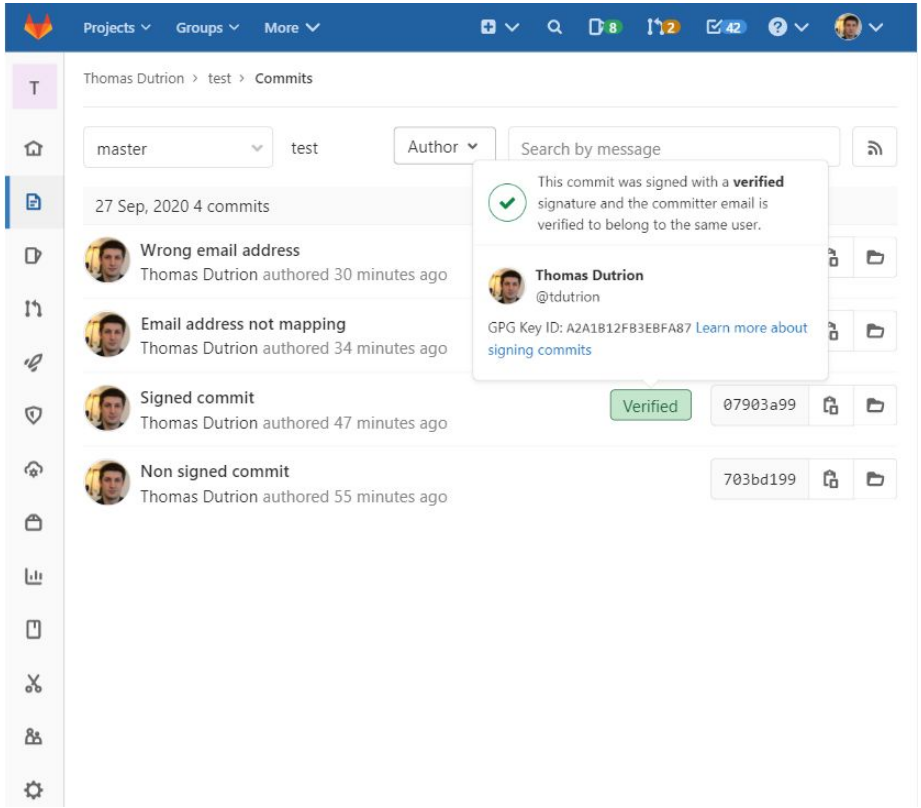
Commit signé sur Gitlab

```
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



Commit signé sur Gitlab



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