



Introduction : importance de GitHub et dangers

Si vous êtes développeur, membre d'un projet open-source ou d'une entreprise qui développe des produits informatiques, vous avez nécessairement entendu parler de GitHub. Vous leur confiez déjà probablement votre code et tout le travail de vos développeurs sans vous poser de questions. Et c'est normal, GitHub est le plus important hébergeur de code dans le monde. Cependant, il y a toujours un risque, en avril 2022, un certain nombre d'organisations ont vu leurs dépôts GitHub être volés à cause d'une faille de sécurité présente dans une application externe qu'elles utilisaient.

Mais rassurez-vous cet article représente le meilleur moyen de sécuriser étape par étape vos projets hébergés sur GitHub.

1. Configurer un espace de travail sécurisé

Établir une connexion sécurisée avec GitHub :

Il existe plusieurs manières de sécuriser sa connexion avec GitHub :

SSH :

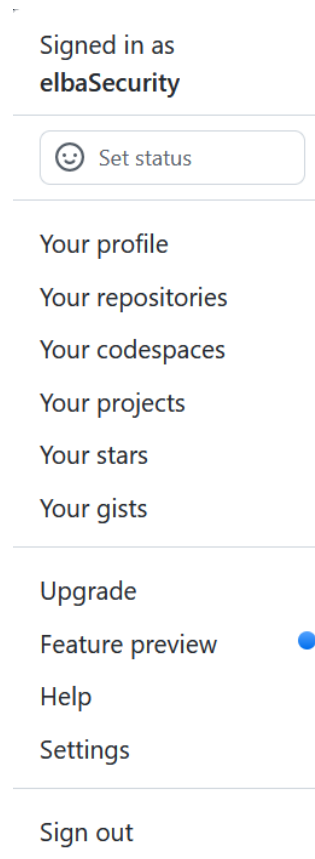
Le protocole [SSH](#) est un protocole de chiffrement asymétrique qui permet d'établir un canal de communication sécurisé avec GitHub. Ce protocole permet donc de s'authentifier à GitHub sans avoir à renseigner votre nom d'utilisateur et votre mot de passe. En effet, après avoir créé votre paire de clés, la clé privée est stockée de manière sécurisée sur votre ordinateur et vous pouvez envoyer votre clé publique au dépôt GitHub distant pour pouvoir vous y authentifier. Ce protocole représente donc un gain de temps à la connexion, mais surtout une manière de sécuriser votre compte puisqu'à chaque requête vers le dépôt GitHub vous êtes authentifié en arrière-plan. Vous pouvez donc choisir un mot de passe très robuste sans avoir à le réécrire à chaque fois.

Pour ajouter une clé SSH à votre compte :

-Il vous faut une paire de clés SSH.

Si vous n'en avez pas, regardez [Comment générer une nouvelle paire de clés SSH](#)

-Puis rendez-vous dans “Settings”



Signed in as
elbaSecurity

😊 Set status

Your profile
Your repositories
Your codespaces
Your projects
Your stars
Your gists

Upgrade
Feature preview ●
Help
Settings

Sign out

-Dans la section “Access” cliquez sur “SSH and GPG keys”

-Cliquez ensuite sur “New SSH key”

SSH keys

New SSH key

There are no SSH keys associated with your account.

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

-Donnez-lui un nom dans le champ “Title” et copiez la clé dans le champ “Key”

SSH keys / Add new

Title

Key

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'

Add SSH key

-Puis cliquez sur “Add SSH key”

-Enfin rentrez votre mot de passe GitHub Enterprise Cloud

GPG :

Le protocole [GPG](#) est également un protocole de chiffrement asymétrique. Il vous permet de chiffrer et de signer les données envoyées sur Internet. L'utilisation du protocole GPG vous permettra donc de signer vos commits vers votre dépôt GitHub. Ainsi, chaque requête signée par un développeur connu sera authentifiée. Vous pourrez donc être sûr de l'origine des commits.


Pour ajouter une clé GPG à votre compte :

-Il vous faut une paire de clés GPG.

Si vous n'en avez pas, regardez [Comment générer une nouvelle paire de clés GPG](#)

-Rendez-vous dans “Settings”

Signed in as
elbaSecurity

 Set status

Your profile
Your repositories
Your codespaces
Your projects
Your stars
Your gists

Upgrade
Feature preview ●
Help
Settings

Sign out

-Dans la section *“Access”* cliquez sur *“SSH and GPG Keys”*

-Cliquez sur *“New GPG key”*

GPG keys New GPG key

There are no GPG keys associated with your account.

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

-Donnez-lui un nom dans le champ *“Title”* et copiez la clé dans le champ *“Key”*

GPG keys / Add new

Title

Key

Begins with '-----BEGIN PGP PUBLIC KEY BLOCK-----'

Add GPG key

-Puis cliquez sur “Add GPG key”

-Enfin rentrez votre mot de passe GitHub Enterprise Cloud

SAML single sign-on :

[SAML SSO](#) est un protocole d'authentification qui permet de faire le lien entre deux parties : un fournisseur d'identité (IdP) et un service. Cela permet à une organisation de sécuriser et de contrôler ses flux de données comme les dépôts ou les requêtes. SAML SSO permet aussi d'authentifier des comptes personnels qui pourraient contribuer à votre projet.

Pour plus d'informations sur l'authentification [SAML SSO pour les entreprises](#)

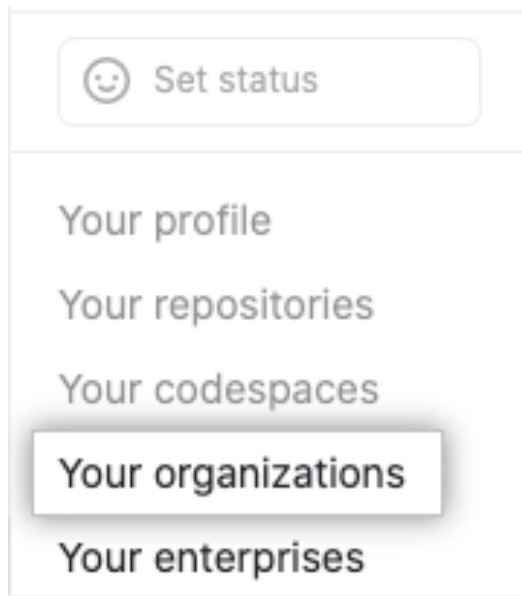
Liste d'adresses IP autorisées :

GitHub permet de restreindre l'accès aux projets d'organisations privées via une [Liste d'adresses IP](#). Ce paramètre permet par exemple à une entreprise de rendre ses dépôts GitHub uniquement accessibles depuis le réseau de ses bureaux ou de permettre aux freelances de se connecter aux dépôts depuis une unique adresse IP, considérée sécurisée.

Pour activer la liste d'adresses IP, rendez vous sur [GitHub](#) :

-Dans le coin en haut à droite, cliquez sur votre photo de profil

-Cliquez sur **“Your organizations”**



-Puis **“Settings”**



-Puis **“Security”**

-Enfin cliquez sur **“Authentication security”**

-Dans le paragraphe **“IP allow list”**, cochez **“Enable IP allow list”**

IP allow list

An IP allow list lets your organization limit access based on the IP address a person is accessing from. [Learn more.](#)

Enable IP allow list
Enabling will allow you to restrict access by IP address to resources owned by this organization.

Save

-Pour sauvegarder, cliquez sur **“Save”**

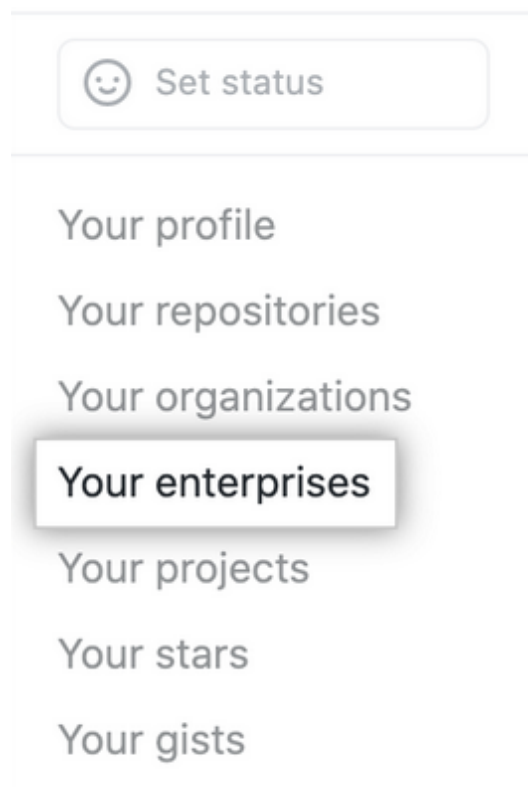
Rendre obligatoire la double authentification :

Avec GitHub il est possible de rendre obligatoire la double authentification dans votre organisation. Par défaut, le propriétaire de l'organisation peut régler les paramètres de sécurité et donc activer la double authentification pour tous, à condition qu'il l'ait lui-même activée sur son compte.

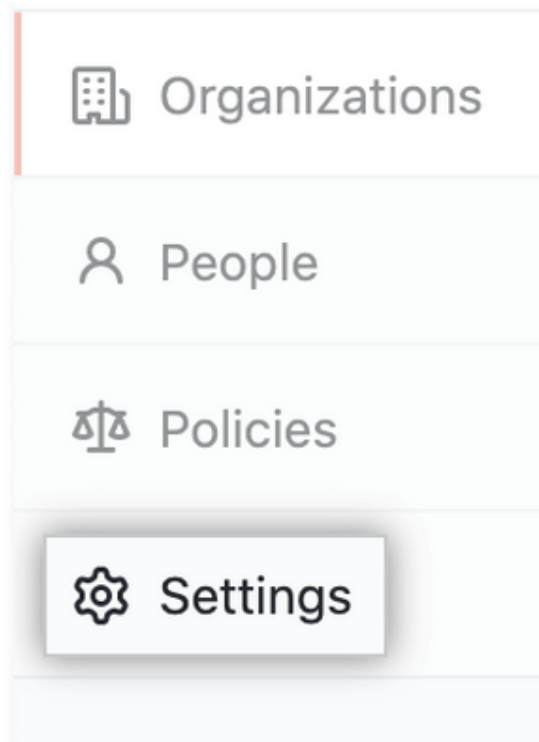
Pour [activer la double authentification pour toute l'organisation](#) :

-Dans le coin en haut à droite de [GitHub.com](https://github.com), cliquez sur votre photo de profil

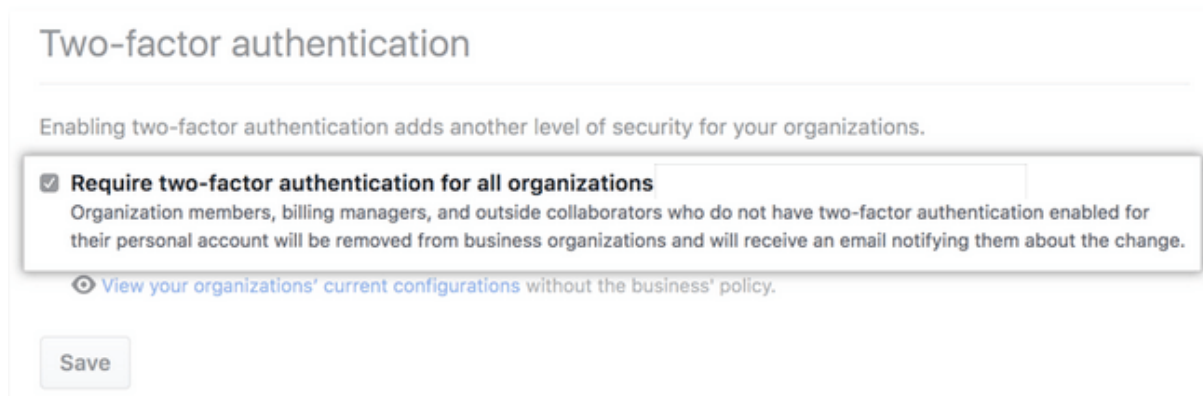
-Puis cliquez sur "Your enterprises", sélectionnez la bonne entreprise



-Cliquez sur "Settings", puis sur "Security"



-En-dessous de “Two-factor authentication”, cochez “Require two-factor authentication for all organizations in your business”



-Enfin, cliquez sur “Save”

Tous les membres de votre organisation qui n'ont pas activé la double authentification seront alors exclus. Si on vous demande une confirmation car certains de vos collaborateurs n'ont pas activé la double authentification, tapez le nom de votre entreprise. Vous pouvez bien sûr envoyer un message à ceux qui ont été exclus pour les prévenir et les ré-inviter une fois qu'ils ont activé la double authentification.

Mais avant de prendre une décision aussi brutale pour vos collaborateurs qui n'auraient pas activé la double authentification vous pouvez [vérifier si vos collaborateurs ont activé la 2FA](#)

GitHub permet la double authentification ([2FA](#)) par sms ou avec une application de double authentification comme :

- 1Password ([1Password](#))
- Authy ([Authy](#))
- LastPass Authenticator ([LastPass Authenticator](#))
- Microsoft Authenticator ([Microsoft Authenticator](#))

Gérer les accès au code :

- *Permissions des utilisateurs*

Il est très important d'accorder les bons accès aux bonnes personnes. En effet, vos collaborateurs doivent avoir les accès suffisants pour pouvoir travailler, mais ceux-ci doivent rester les plus limités possible pour éviter que le code puisse être compromis dans une trop large mesure en cas de faille de sécurité ou si un collaborateur voulait nuire à l'entreprise.

- Rôles :

GitHub permet de définir des rôles [Rôles dans une organisation](#). Il est possible de définir ces rôles pour une équipe ou un individu, à chaque rôle correspond à un certain niveau de permissions pour agir sur l'organisation, ce qui limite le risque qu'un collaborateur ne nuise au code consciemment ou non.

GitHub permet également de définir des [Rôles dans dans les dépôts](#), avec plus ou moins de permissions.

From least access to most access, the roles for an organization repository are:

- **Read:** Recommended for non-code contributors who want to view or discuss your project
- **Triage:** Recommended for contributors who need to proactively manage issues and pull requests without write access
- **Write:** Recommended for contributors who actively push to your project
- **Maintain:** Recommended for project managers who need to manage the repository without access to sensitive or destructive actions
- **Admin:** Recommended for people who need full access to the project, including sensitive and destructive actions like managing security or deleting a repository

- *Permission des applications tierces*

Il est également important de surveiller les accès qu'on accorde aux applications tierces qu'on intègre au projet GitHub. Comme pour vos collaborateurs, il faut leur fournir les accès suffisants pour qu'elles puissent réaliser leurs tâches sans pour autant pouvoir compromettre la totalité de votre projet. Ces applications tierces représentent autant de nouvelles failles potentielles pour votre projet. Il faut donc bien s'assurer qu'elles aient des accès adaptés à leurs besoins.

Au début de cet article je vous parlais d'une cyber-attaque qui avait permis à des pirates de voler des dépôts GitHub, cette attaque a été rendue possible par la compromission d'un jeton OAuth donnant accès aux dépôts GitHub de la plateforme Heroku. Les pirates ont ensuite pu récupérer d'autres jetons pour se connecter à d'autres organisations et poursuivre leurs opérations de la même manière à chaque fois qu'ils trouvaient des informations de connexion.

Mais pour faire face à ce genre de risque, GitHub propose deux outils très utiles qui surveillent les dépendances que vous utilisez.

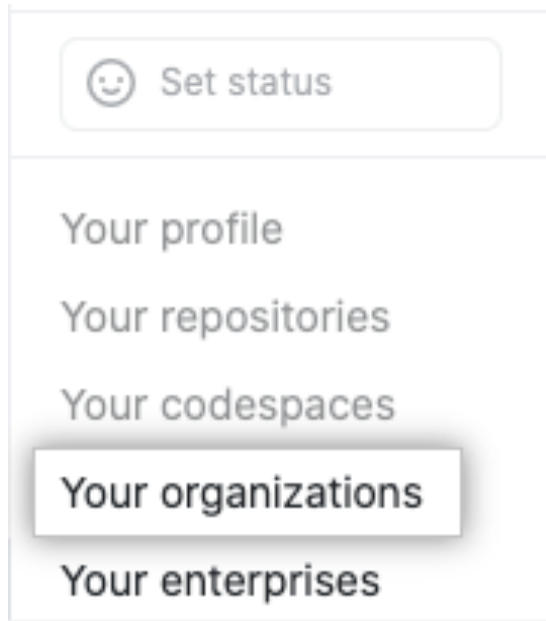
- Dependabot :

GitHub est capable de détecter les vulnérabilités présentes dans les dépôts publics que vous utilisez. [Dependabot](#) est donc un outil qui vous alerte lorsque vous construisez votre code à partir de dépendances publiques qui ne sont pas sécurisées. Utiliser des dépendances non sécurisées peut causer des problèmes importants, il est donc judicieux de mettre en place Dependabot.

Pour [Configurer les alertes Dependabot](#) :

-Cliquez sur votre photo de profil dans le coin en haut à droite de n'importe quelle page GitHub

-Puis cliquez sur "Your organizations"

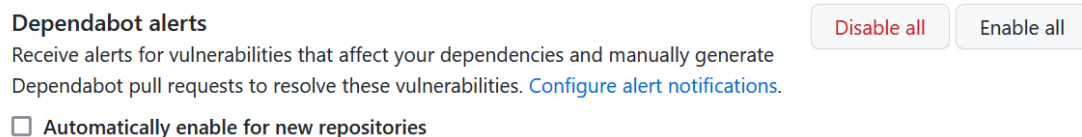


-A droite de l'organisation cliquez sur "Settings"



-Dans la section "Security", cliquez sur "Code security and analysis"

-À droite de "Dependabot alerts" cliquez sur "Enable all"



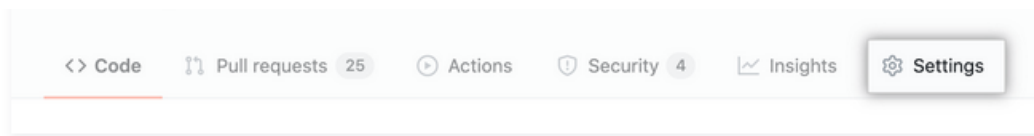
-Enfin si vous le souhaitez activez Dependabot pour tous les nouveaux dépôts et tous ceux que vous possédez déjà

- Graphe de dépendances :

Le [Graphe de dépendance](#) est quant à lui un outil permettant de visualiser facilement l'ensemble des dépendances que vous utilisez dans un projet. La combinaison de Dependabot et du graphe de dépendances permet donc d'identifier très rapidement les dépendances non sécurisées et de réagir en conséquence.

Pour [Configurer le graphe de dépendances](#)

-Cliquez sur "Settings", sous le nom de votre dépôt



-Dans la section “Security”, cliquez sur “Code security and analysis”

-À droite de “Dependency graph” cliquez sur “Enable”

Dependabot alerts

Receive alerts for vulnerabilities that affect your dependencies and manually generate Dependabot pull requests to resolve these vulnerabilities. [Configure alert notifications.](#)

Automatically enable for new repositories

Disable all

Enable all

Analyser votre code à la recherche d’erreurs et de vulnérabilités

L’outil Code scanning de GitHub est un outil qui scanne les dépôts de votre organisation à la recherche d’erreurs ou de vulnérabilités dans le code. Cet outil vous permet donc de sécuriser facilement votre code.

Pour configurer Code scanning : [Mise en place de Code scanning](#)

Surveiller les informations sensibles :

Nous disions précédemment qu’il était possible que des informations confidentielles comme des mots de passe, des clés API ou des jetons OAuth se glissent dans les dépôts GitHub. En plus d’encourager vos développeurs à faire attention, il existe des outils pour scanner le dépôt à la recherche d’informations sensibles.

GitHub dispose de ses propres outils de scan comme [Secret scanning](#). Secret scanning est un outil qui scanne tout le code d’un dépôt à la recherche de chaînes de caractères qui ressembleraient aux formats de clés privées ou de jetons de connexions, en lien avec les services partenaires de GitHub. Secret Scanning parcourt également les archives des dépôts à la recherche d’informations secrètes.

Mais il existe aussi des solutions privées comme [GitGuardian](#) qui surveillent aussi bien les dépôts internes que publiques. GitGuardian a analysé près d'un milliard de commits publiques en 2021 et a découvert 6 millions de "secrets".



Securing your systems starts with securing your software development process. GitGuardian understands this, and they have built a pragmatic solution to an acute security problem. Their credentials monitoring system is a must-have for any serious organization.



Solomon Hykes, Co-founder

Actualiser régulièrement les clés et tokens de connexion

Actualiser régulièrement les clés et tokens de connexion permet de s'assurer que seuls les personnes légitimes à avoir accès aux dépôts y aient accès. En effet, si une clé SSH est compromise et permet à une personne extérieure d'accéder à vos dépôts, le simple fait de changer les clés tous les trois mois par exemple permettra de purger votre organisation à intervalles réguliers.

Pour générer une nouvelle clé SSH cliquez ici :

[Générer une nouvelle paire de clés SSH.](#)

Pour générer une nouvelle clé GPG cliquez ici :

[Générer une nouvelle paire de clés GPG](#)

Les clés inutilisées représentent également un risque, il est donc important de s'assurer régulièrement qu'aucune clé n'est abandonnée.

[-Chercher les clés SSH existantes](#)

[-Chercher les clés GPG existantes](#)

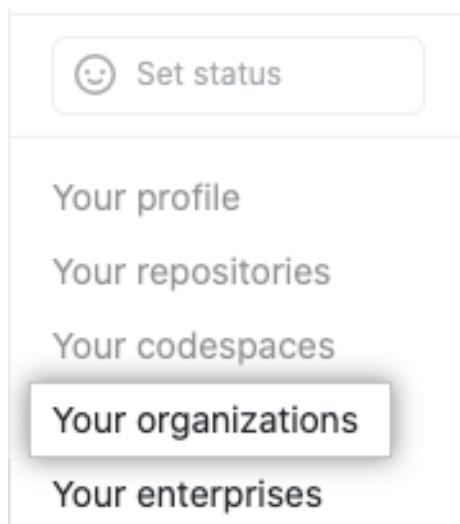
Pour des raisons de sécurité, GitHub supprime toutes les clés inutilisées après un an, mais nous vous recommandons de le faire plus souvent.

Mettre en place un offboarding sécurisé :

Quand un collaborateur quitte votre organisation, il est crucial de révoquer rapidement tous ses accès aux systèmes de l'entreprise et particulièrement à GitHub, car les comptes sont généralement personnels. Vous ne pouvez pas changer le mot de passe du compte de votre ancien collaborateur, il faut donc bien veiller à ce qu'il soit exclu du projet très rapidement pour qu'il ne puisse pas le compromettre.

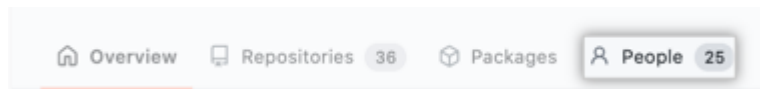
Pour exclure un membre de votre organisation sur GitHub cliquez [ici](#) ou suivez les étapes :

-Cliquez sur votre photo de profil, puis rendez vous dans "Your organizations"



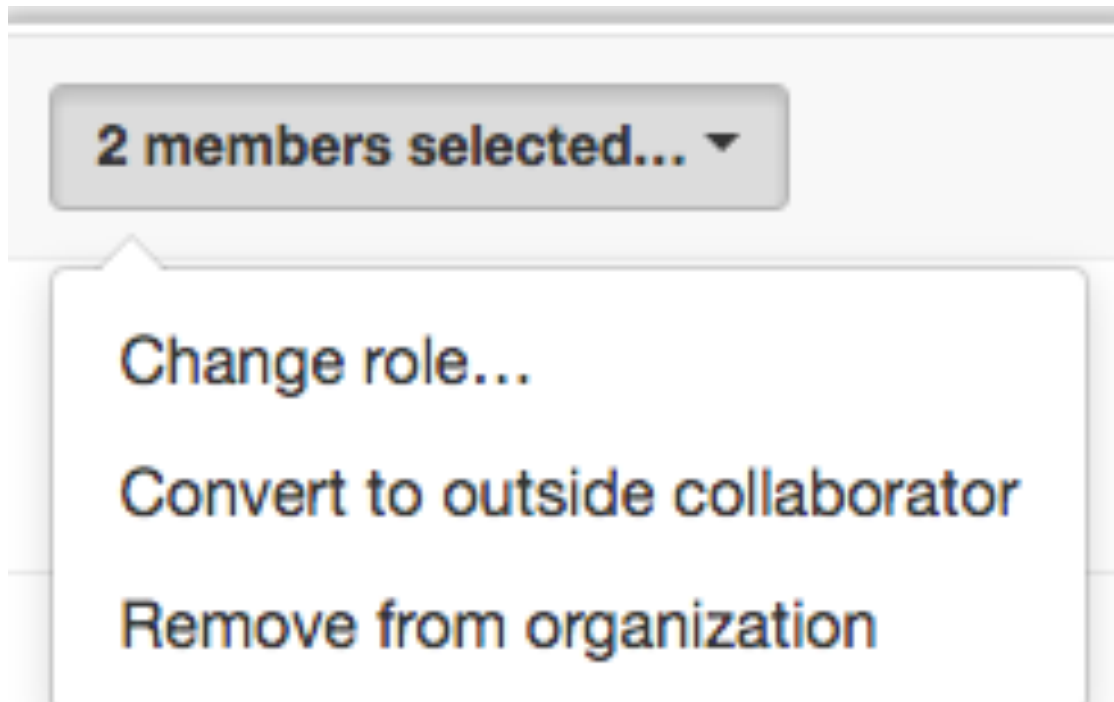
-Sélectionnez la bonne organisation

-Sous le nom de votre organisation cliquez sur "People"



-Sélectionnez le profil dont vous souhaitez révoquer les accès

-Au-dessus de la liste des membres cliquez sur le menu déroulant et choisissez "Remove from organization"



-Enfin cliquez sur "Remove member"

Conclusion

Si vous êtes arrivés jusqu'ici, vous vous rendez bien compte qu'il existe beaucoup de moyens pour sécuriser vos projets GitHub, mais même si vous les activez tous cela ne sera jamais suffisant. Ces configurations sont les fondations sur lesquelles vous bâtissez votre politique de sécurité, vous ne la rendrez véritablement efficace que par la formation de vos collaborateurs.