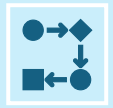
Investigations dans GitHub Entreprise

Emilien Lassalle

Introduction
Objectifs

Introduire l’utilisation de GitHub Enterprise et son implication dans les procédés DevOps.

Présenter quelques menaces sur la plate-forme.



Mettre en avant les possibilités d’investigation.

Disclaimer : Cette présentation n’est pas une publicité pour la solution et n’a pas vocation à être à charge.

Introduction

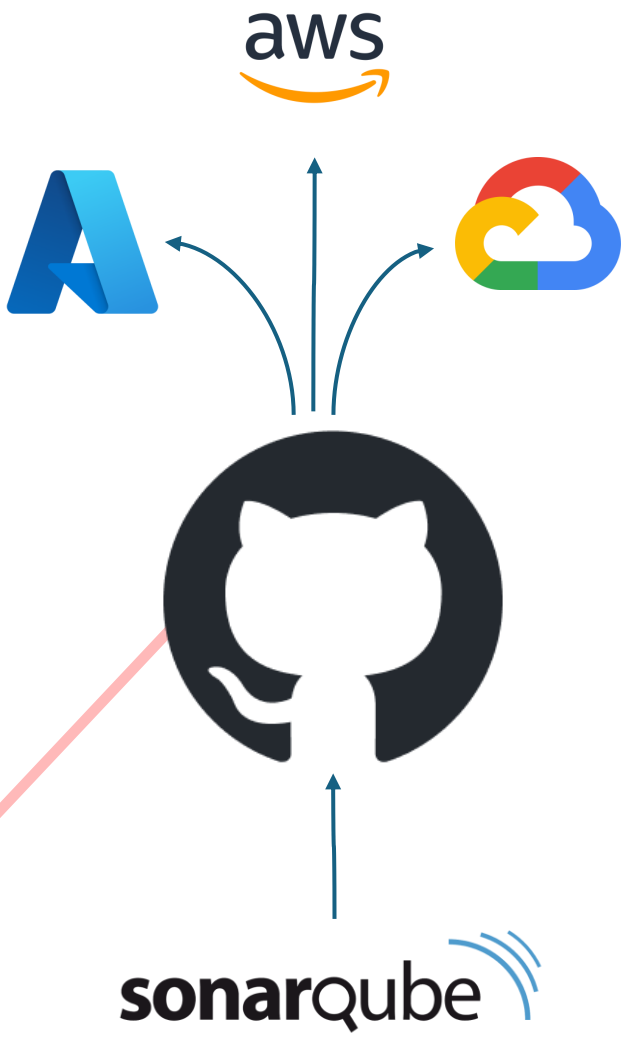
Les ressources Cloud en entreprise

Move to Cloud	➢Migration d'application On-Prem
Applications Cloud-native	➢Construites à même le fournisseur
Environnement multi-cloud	➢Pour l'hébergement des ressources
Multitude d'outils de développements	➢GitLab, GitHub, SonarQube…

Introduction

Contraintes et rationalisations

Production	• Uniformiser les coûts • Fournir une base de connaissance
Résilience	• Centraliser le code des applications
Contrôle	• Empêcher les mauvaises configurations • Détecter des vulnérabilités dans le code



Infrastructure as Code, CICD

Introduction

GitHub Entreprise (Cloud)

IAM

- Fédération des utilisateurs
- Administration

Repository

- Hébergement de fichier
- Développement collaboratif

Actions

- Automatisation des activités de développement

Introduction
Les Logs

Envoi des logs vers :

- AWS, Azure, GCP
- Datadog, Splunk

Push → Détection

- Par API ou export manuel

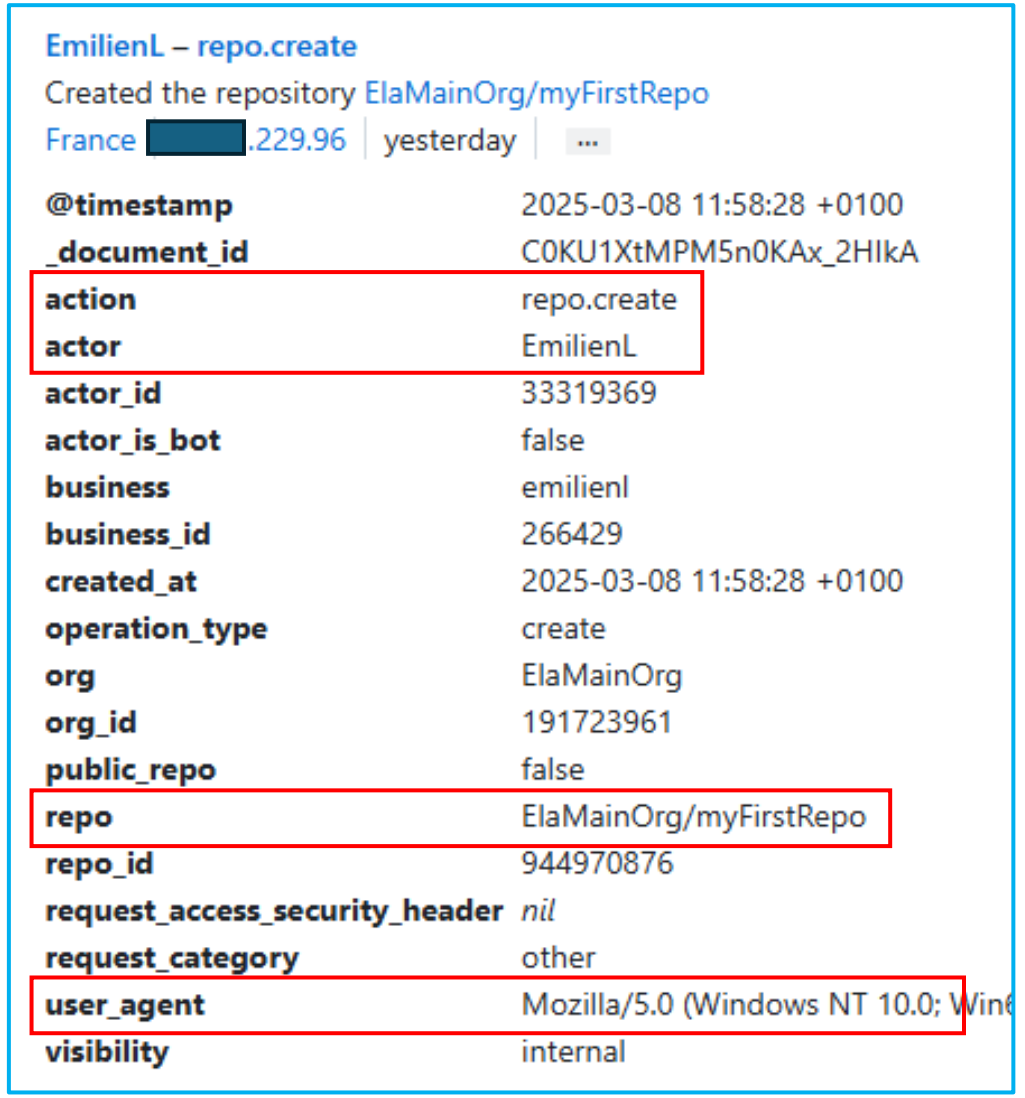
Pull → Investigation

Deux types de journaux :

- Audit – 180 jours de rétention
- Git – 7 jours

GitHub ne trace pas :

- Les IPs sources (activable)
- Les activités Git effectuées par navigateur



Extrait du portail github.com/enterprises/<enterprise_name>/settings/audit-log

GitHub

Dépôt de code (Repository)

Hébergement de fichier :

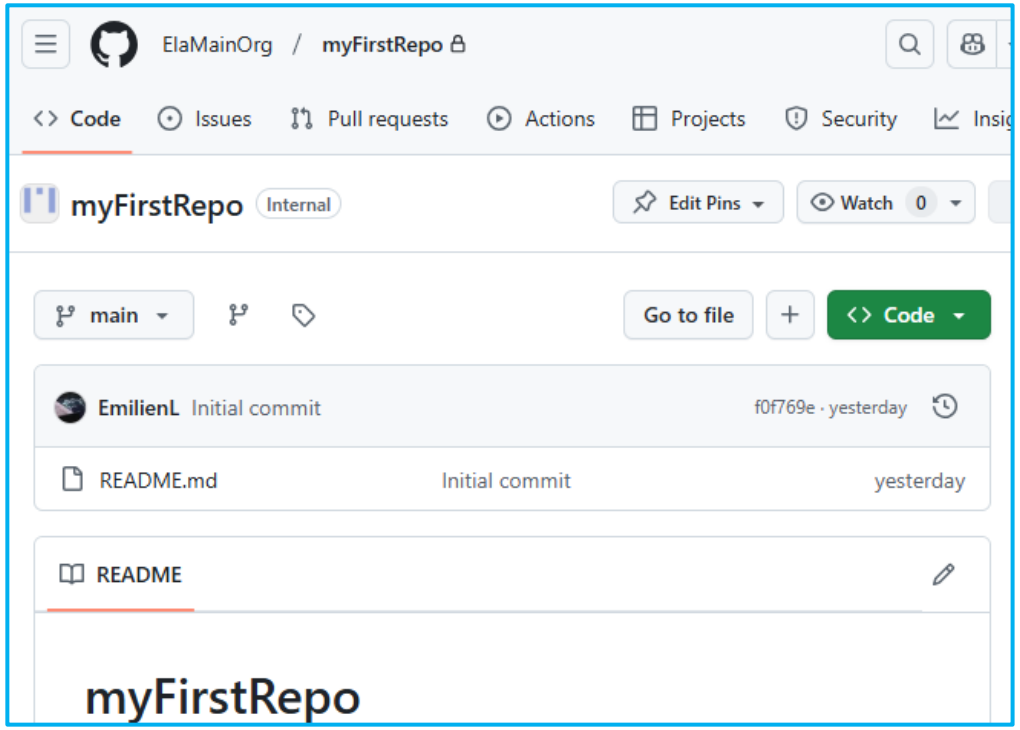
- Majoritairement du code

Exposition :

- Publique
- Interne – limité aux utilisateurs de l'organisation
- Privée – limité à certains utilisateurs

Menaces :

- Propriété intellectuelle
- Supply-chain
- Accès aux vulnérabilités du code



GitHub Repository
Superviser l’accès

Simple consultation :

- Aucune trace

Archivage à l’aide de la commande *git archive* :

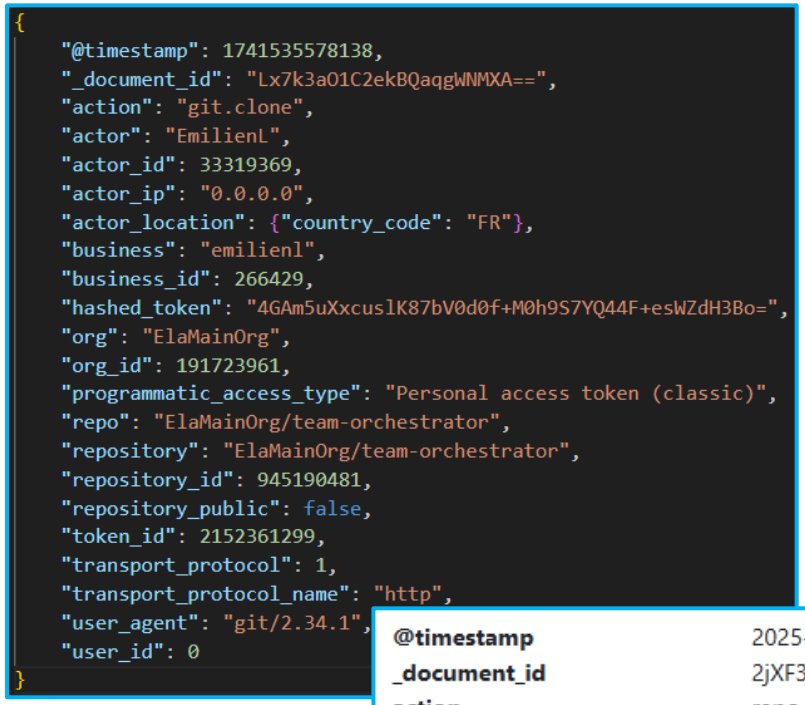
- Aucune trace
- Cependant un évènement *repo.archived* existe si l’archivage est fait depuis le portail

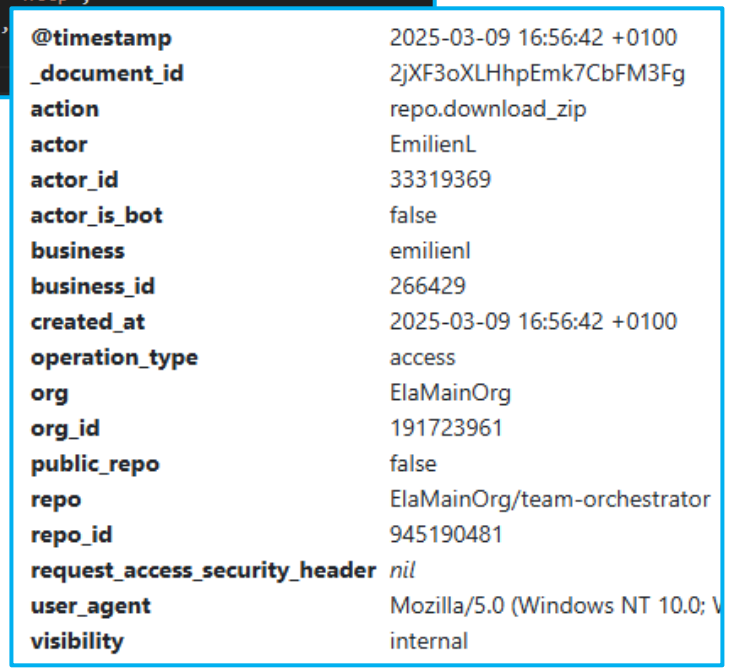
Clone :

- Un évènement *git.clone* est généré dans les journaux Git

Téléchargement par le portail :

- Un évènement *repo.download_zip* est généré dans les journaux d’audit





GitHub

Identity Access Management

Federation (ou pas) des utilisateurs :

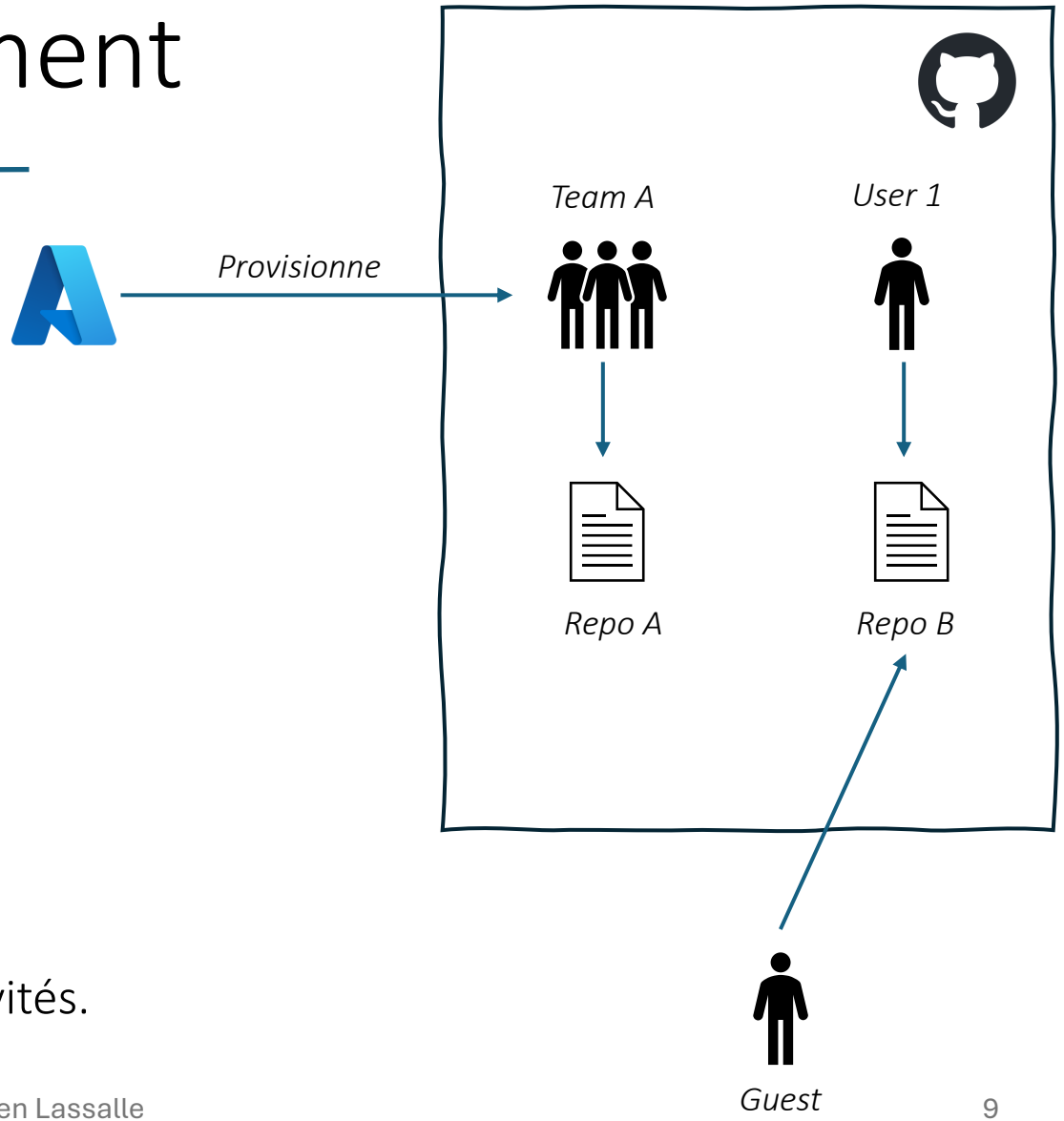
- Enterprise Managed Users (*EMU*)
- Exemple de fournisseur partenaire : *EntraID*

Le type d'utilisateur **va orienter les investigations** :

- *EMU* → au niveau de l'Identity Provider
- *Personal* → au sein de GitHub

Les utilisateurs peuvent être **organisés en équipe.**

Des collaborateurs externes peuvent également être invités.

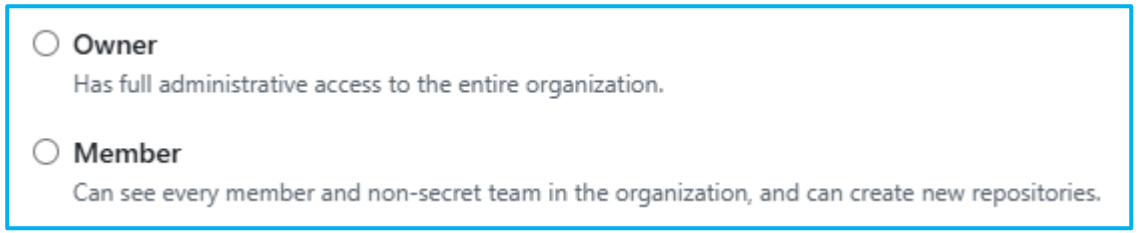


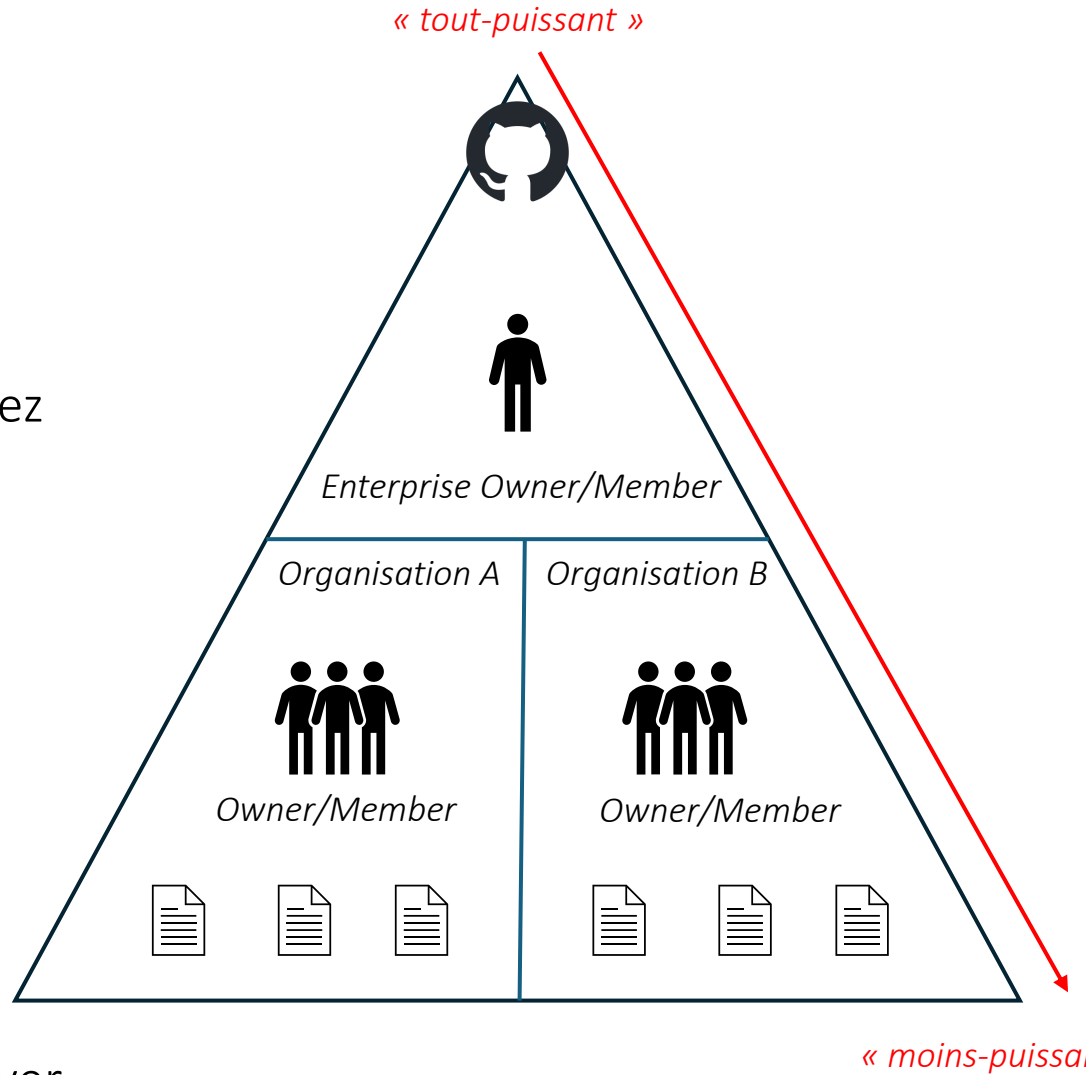
GitHub IAM
Hiérarchie

GitHub vous fournit **une entreprise** dans laquelle vous pouvez créer **plusieurs organisations**.

➢ Ce sont les organisations qui portent les dépôts

Deux types de permissions (Entreprise ou Organisation) :





Note : Un simple *Member* au niveau *Entreprise* peut s'octroyer le droit *Owner* sur une Organisation.

GitHub IAM

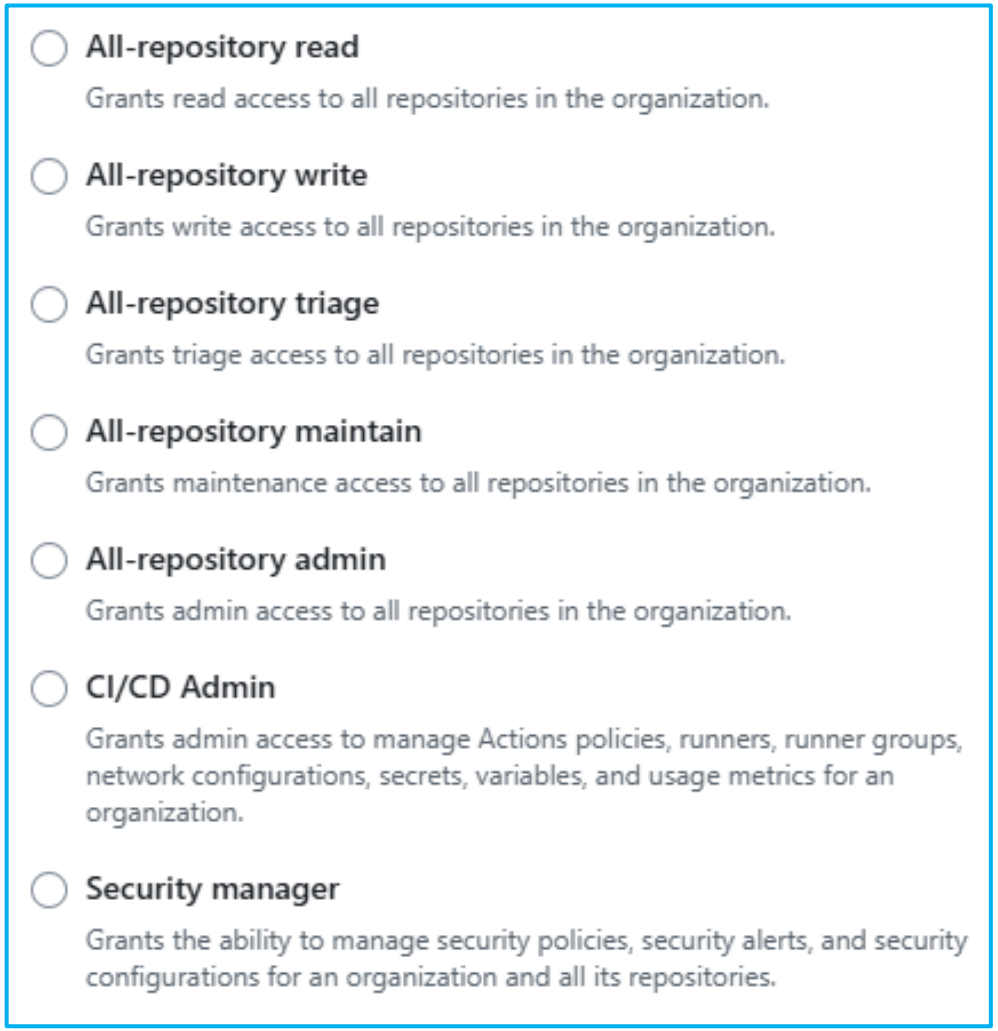
Rôles d'organisation

Au sein d'une même organisation, n'importe quel membre ou équipe peut se voir attribuer :

- Un **accès à un ou des dépôts**
- Un **rôle prédéfini** (Personnalisé ou natif)

Le rôle Security Manager :

- Particulièrement intéressant pour les **investigations**
- Ne permet **aucune action de réponse**



Extrait du portail github.com/organizations/<organisation_name>/settings/org_rôle_assignments

GitHub IAM
Accès programmatique

Personnal Access Token (PAT) :

- Classic
- Fine-grained (*preview*)

Pour tracer l'utilisation d'un PAT :

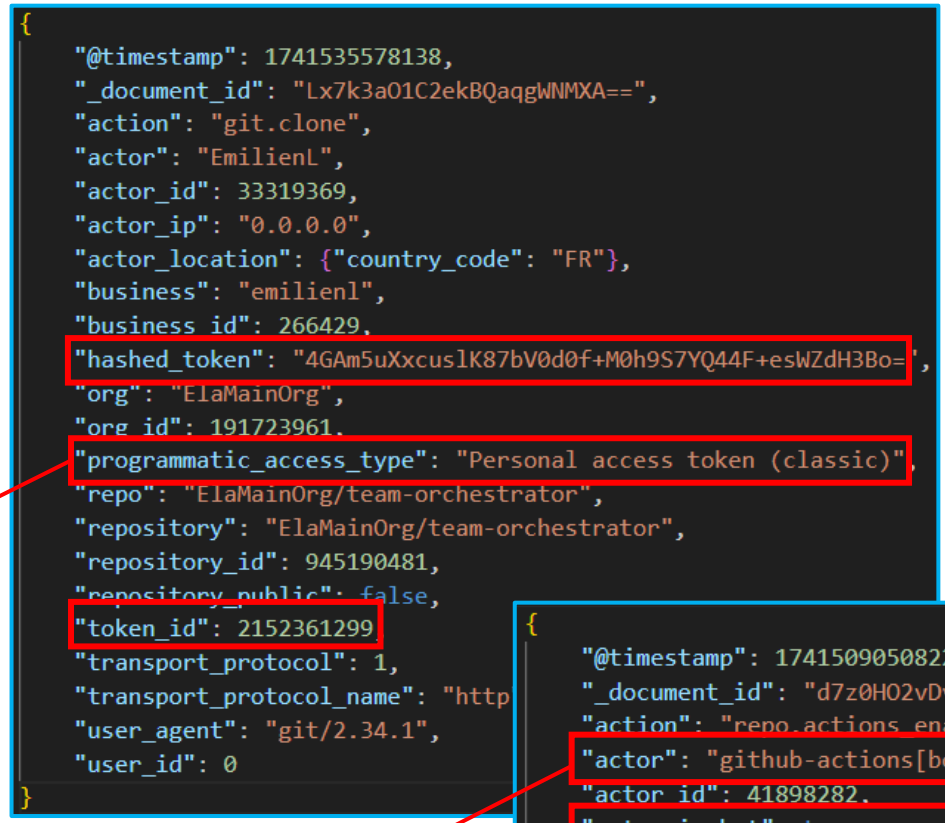
- *Programmatic_access_type, hashed_token, token_scopes*

GitHub Applications :

- Elles agissent au niveau de l'organisation
- Peuvent hériter de droits privilégiés

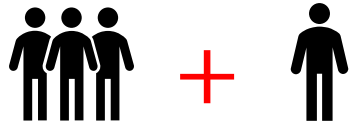
Pour tracer l'utilisation d'une GitHub app :

- *actor : *[bot], actor_is_bot*



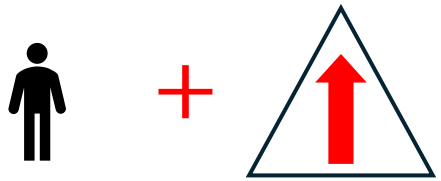
GitHub IAM

Persistance et élévation de privilège



Création d'une identité

- Org.add_member
- Org.add_outside_collaborator
- Business.invite_admin
- Team.create



Attribution de droits

- Organization_role.Assign
  - \+ Organization_rôle_name == <rôle>
- Integration_installation.create
  - \+ repository_selection: « all »
- Business.add_admin
- Org.update_member
  - \+ permission == « admin »

Attributions de droits sur dépôt

- Team.add_member
- Team.add_repository
- Team.create
- Repo.add_member
- Repo.update_member

https://docs.github.com/fr/enterprise-cloud@latest/admin/monitoring-activity-in-your-enterprise/reviewing-audit-logs-for-your-enterprise/audit-log-events-for-your-enterprise

GitHub Actions
CICD

GitHub Actions implémente les concepts **CICD** :

https://www.redhat.com/fr/topics/devops/what-is-ci-cd

Ces étapes misent bout à bout constituent une/des pipelines.

La partie **CD est particulièrement sensible** car elle nécessite des **secrets**, **clefs d'API**, **délégations** pour effectuer des actions en production.

GitHub Actions
Workflows

GitHub Actions utilise des workflows au format .yml et les stockent dans le dossier .github à la racine du dépôt.

Un workflow défini d'abord un **élément déclencheur** puis va être des *jobs* et des *steps*.

Il est possible de leur partager des **variables** et notamment des **secrets** tels que des clefs d'API ou des identifiants d'application GitHub.

Ce workflow sera exécuté sur un ***Runner***, il en existe deux types :

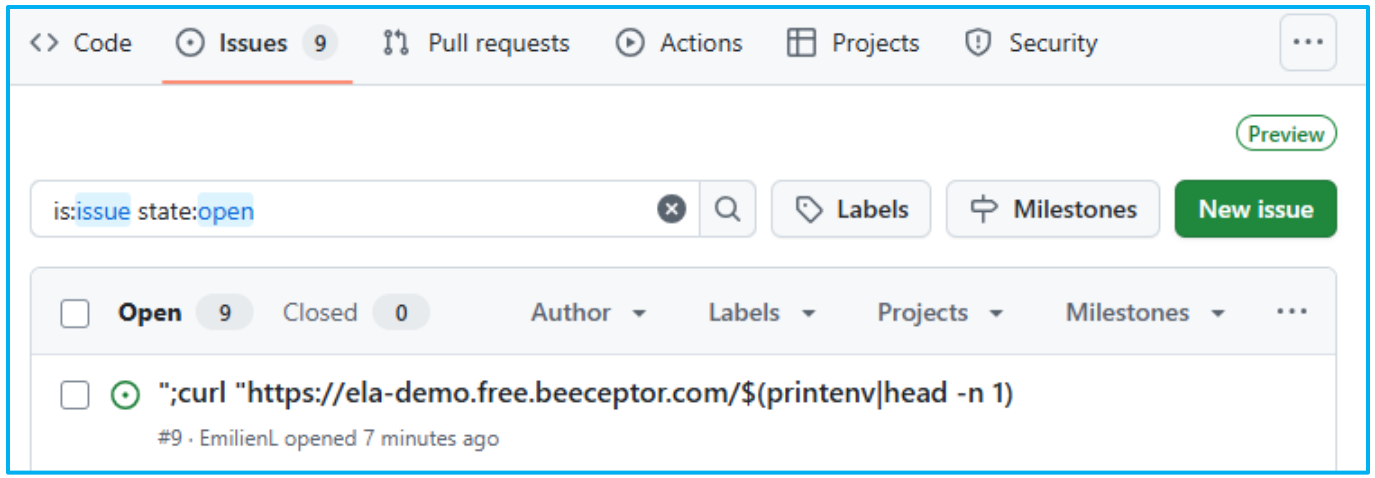
- *GitHub-hosted* – « **public** » → déployés à la volé pour chaque job
- *Self-hosted* – « **privé** » → sur votre infrastructure

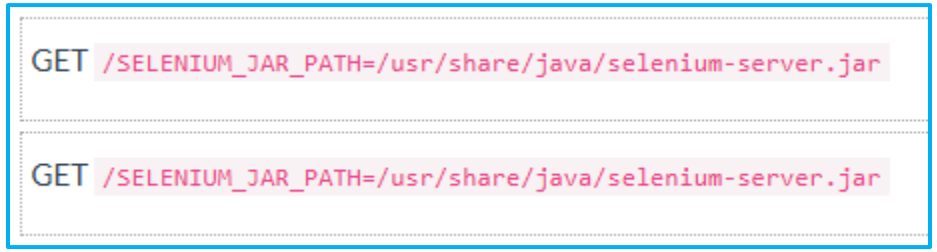
```yaml
name: CI

# Controls when the workflow will run
on:
  pull_request:
    branches: [ "main" ]

# A workflow run is made up of one or more jobs
# They can run sequentially or in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest

    # Steps represent a sequence of tasks
    # They will be executed as part of the job
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE
      # Needed by the job to access the repo
      - uses: actions/checkout@v4

      # Runs a single command using the runners shell
      - name: Run a one-line script
        run: echo Hello, world!
```

GitHub Actions

Injection de script

```
- name: injection démo
  run: |
    myIssue="${{github.event.issue.title}}"
    echo my issue is "${{github.event.issue.title}}"
```





En fonction du déclencheur, les workflows peuvent récupérer des informations telles que :

- **Le titre** d'une pull-request
- **Le contenue** d'une issue
- …

Certaines de ces informations sont à la main de l'utilisateur. Si le workflow est mal configuré, il devient possible d'injecter des commandes.

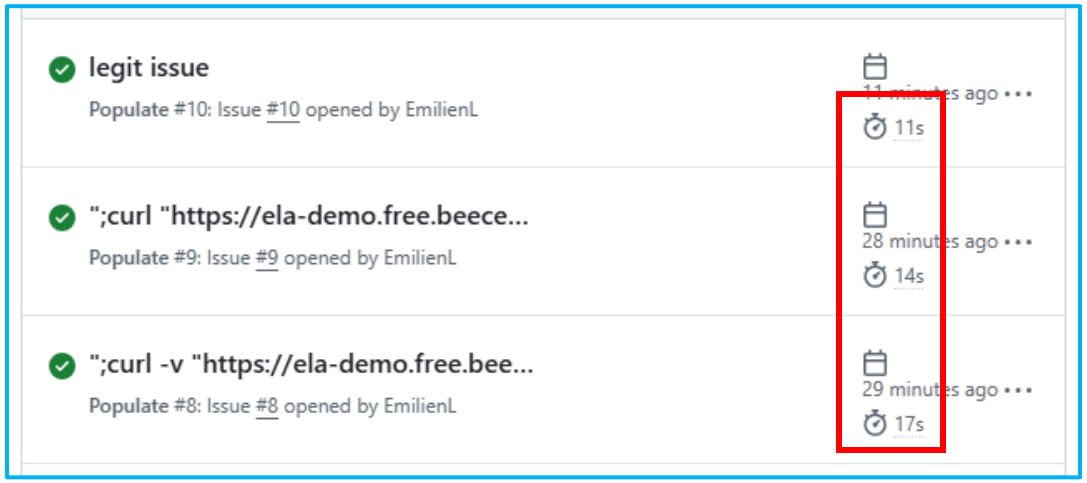
Un attaquant pourrait vouloir **exfiltrer des secrets ou exécuter du code** non prévu.

https://docs.github.com/en/actions/security-for-github-actions/security-guides/security-hardening-for-github-actions

GitHub Actions
Injection de script

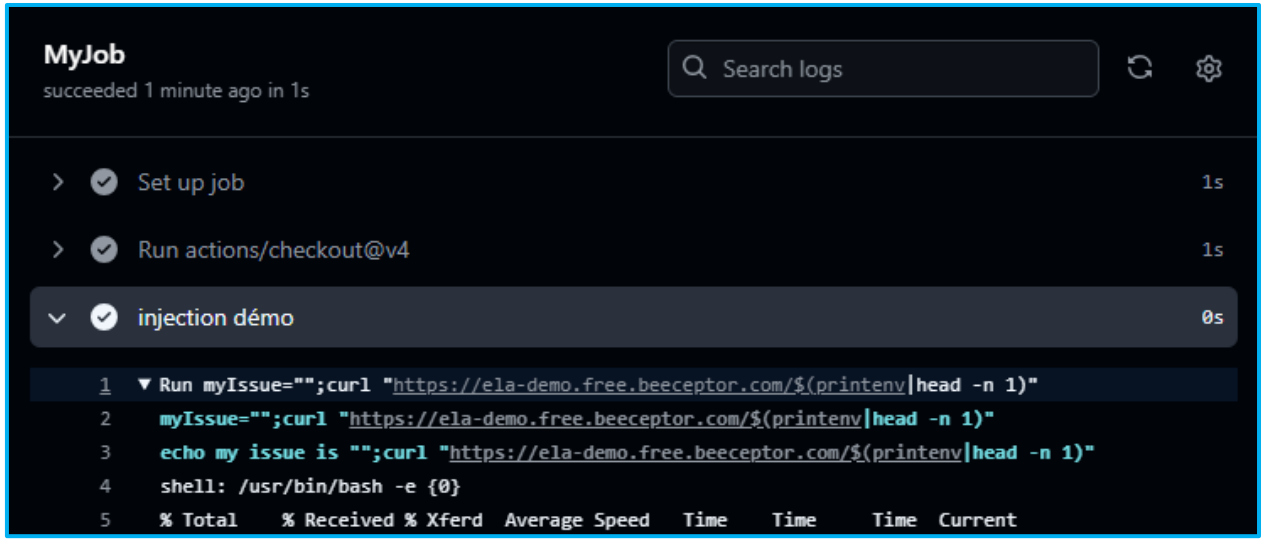
Il faudra chercher des déviations d'exécutions de ces workflows :

- Exécution plus lente
- Si le runner est self-hosted
  - Tracer les exécutions, EDR ?
  - Contrôler les flux sortants
- Si l'exécution utilise une application GitHub
  - Vérifier qu'elle exécute les actions prévues



Investiguer :

- Une opération Workflows.created_workflow_run
  - workflows.delete_workflow_run → suppression des traces
- Directement sur le portail



GitHub Actions
Latéralisation

Un attaquant peut **réutiliser les secrets** qu'il a volé pour **pivoter**.

Exploitation d'*Open ID Connect* (**OIDC**) :

- ➢ Si GitHub est enregistré comme *Identity Provider* auprès du CSP
- ➢ Permet d'utiliser un workflow si le dépôt est déclaré

EntraID - Enterprise application registration

Certificates (0) Client secrets (1) **Federated credentials (1)**

Allow other identities to impersonate this application by establishing a trust with an external OpenID Connec
tokens to access Microsoft Entra ID protected resources that this application has access to like Azure and Mic

+ Add credential

Name	Description	Subject identifier or claims matching expression
OIDC-demo		repo:ElaMainOrg/MyFirstRepo:ref:refs/heads/main

AWS – Trust policies

```
"Condition": {
    "StringEquals": {
        "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
        "token.actions.githubusercontent.com:sub": "repo:elamainorg/myfirstrepo:environment:prod"
    }
}
```

https://docs.github.com/en/actions/security-for-github-actions/security-hardening-your-deployments/about-security-hardening-with-openid-connect

Conclusion

- Les plateformes de DevOps sont centrales et critiques.
- Il existe peu d'outils défensif adapté.
- Les logs ne font pas tout dans une investigation de compromission CICD, il faut alterner avec le portail.
- La sensibilité d'un dépôt de code va dépendre de votre connaissance de l'infrastructure.

Potentiel schéma d'attaque