



Sécurité Web: Analyses et défenses

**Session 4
HandsOn**

| Mai 2025

Didier BERNAUDEAU

didier.bernaudeau@owasp.org

Agenda

1. Signing Commit
2. Daily Security Pipeline
3. Security Check on Pull Request

Each step must be validated!

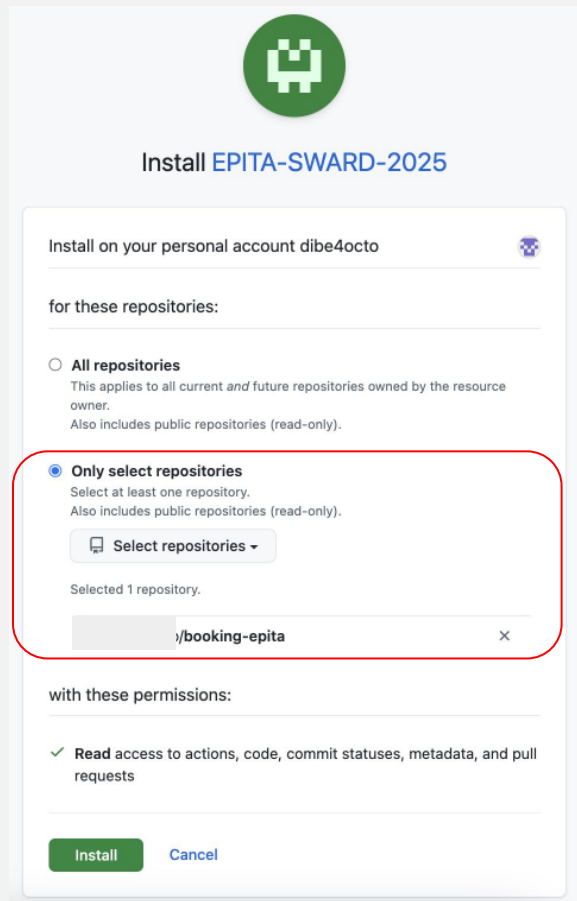
Setup GitHub Repository

- Create Github repository
 - Name: booking-epita
 - Private access
 - Empty (no readme / no licence)
- Get Booking App
 - Clone repository <https://github.com/Oxdbe/booking-epita.git>
 - Change remote "git remote set-url origin..." by your repository
 - Push
- Add "author" file with:
 - First name
 - Last name

Setup GitHub App

This github app is used to clone repository and perform test.

- Installation Page:
<https://github.com/apps/epita-sword-2025>
- Install only on "booking-epita" repository



Install **EPITA-SWORD-2025**

Install on your personal account dibe4octo

for these repositories:

☐ **All repositories**
This applies to all current and future repositories owned by the resource owner.
Also includes public repositories (read-only).

☒ **Only select repositories**
Select at least one repository.
Also includes public repositories (read-only).

Select repositories ▾

Selected 1 repository.

y/booking-epita ×

with these permissions:

✓ **Read access to actions, code, commit statuses, metadata, and pull requests**

Install Cancel



Chapter 1

Signing Commit

Disclaimer

This GPG setup is for beginner

*Advanced user uses subkey and
hardware security module (yubikey or smartcard)*

GPG

Ubuntu

Installed by default on ubuntu 22/24

The following command must be available:

```
gpg, pinentry, gpg-connect-agent
```

gpg-agent should run via systemd:

```
sudo systemctl --user status gpg-agent
```

GPG

MacOS

- Installed GPG and pinentry

```
brew install gnupg pinentry-mac
```

- Configure gpg-agent to use pinentry-mac by editing `~/.gnupg/gpg-agent.conf`

```
pinentry-program /opt/homebrew/bin/pinentry-mac
```


Shell Integration

ZSH

enable GPG Agent in ~/.zshrc

```
plugins=(gpg-agent)
```

Commit with Private Email Address

- Enable this feature:
 - Visit: <https://github.com/settings/emails>
 - enable "Keep my email addresses private"

- Configure Git to use this email

```
git config --global user.email <ID>@users.noreply.github.com
```

Generate signing key

- Create a key using gpg: `gpg --expert --full-generate-key`
- This command start a dialog to configure your key:
 - Please select what kind of key you want: (10) ECC (sign only)
 - Please select which elliptic curve you want: (1) Curve 25519
 - Please specify how long the key should be valid: 0 = key does not expire
 - Real name: <GITHUB_USERNAME>
 - Email address:<PRIVATE_COMMIT_EMAIL>@users.noreply.github.com
- List key to find key ID: `gpg --list-secret-keys --keyid-format LONG`
- Print key: `gpg --armor --export <ID>`

Configure GitHub

- Open <https://github.com/settings/keys>
- Add GPG key
- Enable “Flag unsigned commits as unverified”

Configure Git for signing commit

- Enable signature feature

```
git config --global commit.gpgsign true
```

- Set your GPG signing key in Git

```
git config --global user.signingkey <KEYID>
```



Chapter 2

Daily Security Pipeline

Daily Security Pipeline

- Create a Security workflow
 - Schedule every day
 - On demand (workflow_dispatch) for testing purpose

Daily Security Pipeline

Job 1: DependenciesAnalysis

- Add step to run OSV Scanner
 - Action: google/osv-scanner-action/osv-scanner-action
- Add step to send report to security dashboard
 - Action: github/codeql-action/upload-sarif

Daily Security Pipeline

Job 2: SourceCodeAnalysis

- Setup job to run semgrep

```
runs-on: ubuntu-latest
container:
  image: returntocorp/semgrep
```
- Add step to run analysis with semgrep
 - Apply ruleset for java: "p/java"
- Add step to send report to security dashboard
 - Action: github/codeql-action/upload-sarif

Daily Dependency Scan

Container

- Setup a Daily Security workflow
- Add step to run OSV Scanner
- Send report: to issue or security dashboard



Chapter 3

Security Check on Pull Request

Security Check on Pull Request

- Create a PullRequest workflow to include all security check
- Security Check must be only for new code

Daily Security Pipeline

Job 1: DependenciesAnalysis

- Add step to run OSV Scanner: after PR and before PR
 - Action: google/osv-scanner-action/osv-scanner-action
- Compare analysis
 - Action: google/osv-scanner-action/osv-reporter-action
- Save differential report as artifact
 - actions/upload-artifact

*For better developer experience, this report could be insert as comment in Pull Request
(this need a custom development to call Github API)*

Daily Security Pipeline

Job 2: SourceCodeAnalysis

- Setup job to run semgrep

```
runs-on: ubuntu-latest
container:
  image: returntocorp/semgrep
```
- Add step to run differential analysis with semgrep
 - Apply ruleset for java: "p/java"
 - baseline commit option
- Save differential report as artifact
 - actions/upload-artifact

*For better developer experience, this report could be insert as comment in Pull Request
(this need a custom development to call Github API)*

Security Check on Pull Request

Enable branch protection on default branch using Rules:

- Require status checks to pass:
 - add dependencies
 - Add source code analysis