semgrep-rule-creator

semgrep-rule-creator

> *Catalogue généré le 2026-05-11*

En une phrase

Aide à écrire des règles personnalisées pour Semgrep — un outil qui scanne le code à la recherche de motifs vulnérables — afin de détecter automatiquement un type de bug spécifique à ton projet.

Quand l'utiliser

- Tu as identifié un motif vulnérable récurrent et tu veux qu'un scan automatique le détecte à chaque commit.
- Tu veux imposer une convention de code à ton équipe (par exemple : « interdiction d'utiliser telle fonction dangereuse »).
- Tu veux suivre la trace d'une donnée « salie » par l'utilisateur jusqu'à un point sensible (mode « taint », qui suit le flux des données).
- Tu écris une règle Semgrep et tu veux la tester proprement avec des cas vulnérables ET des cas sûrs.

Comment l'invoquer

- **Slash command** : `/semgrep-rule-creator` (ou `/semgrep-rule`).
- **Phrases déclencheurs (texte)** : "write a Semgrep rule", "build a custom static analysis detection", "detect this pattern".
- **Auto-invocation** : Sur demande explicite.

Description détaillée

Semgrep est un outil qui parcourt ton code et signale tous les endroits où un certain motif apparaît. Par exemple, on peut lui demander de signaler tous les appels à `eval()` (une fonction réputée dangereuse) ou toutes les requêtes SQL construites par concaténation. Le skill `semgrep-rule-creator` t'aide à écrire ces règles correctement, parce qu'une règle mal écrite produit soit trop d'alertes inutiles (faux positifs qui érodent la confiance), soit pas assez (faux négatifs qui laissent passer des vrais bugs).

Le skill insiste sur la méthode : commencer par un motif large pour bien capturer le bug initial, puis l'affiner jusqu'à ce qu'il ne déclenche que sur le vrai cas dangereux. Il rappelle aussi qu'une règle non testée ne sert à rien : il faut toujours fournir des exemples de code vulnérable (qui doivent déclencher l'alerte) et des exemples sûrs (qui ne doivent pas la déclencher). Il décourage les patterns trop larges (qui matchent tout et n'importe quoi) et trop spécifiques (qui ratent les variantes).

Pour les cas où on veut suivre la propagation d'une donnée — par exemple, une entrée utilisateur qui finit dans une commande système — il guide vers le mode « taint » de Semgrep, qui modélise les sources (où entre la donnée) et les sinks (où elle ne devrait jamais arriver sans nettoyage). En sortie : une règle YAML prête à intégrer dans ton pipeline d'intégration continue (CI).

Pour aller plus loin

Pour les détails techniques, exemples et patterns spécifiques, voir le SKILL.md original.

Source

- **Plugin** : `trailofbits/semgrep-rule-creator`
- **Nom interne** : `semgrep-rule-creator`
- **Fichier** : `/home/thymon/.claude/plugins/cache/trailofbits/semgrep-rule-creator/1.1.0/skills/semgrep-rule-creator/SKILL.md`

Revision #2
Created 2026-05-11 21:19:28 UTC by thymon
Updated 2026-05-11 21:37:05 UTC by thymon