Semgrep Static code scanning

Rules YAML / registry `--config` → **Targets** files, ignores applied `scan` → **Findings** severity, location `--json` / `--sarif` → **Report** exit code, files

CONCEPTS

Semgrep matches code structure, not text: a pattern like foo($X) finds the call however it is spaced or nested.

Without --error, scan exits 0 even when it finds issues. Add it (or use semgrep ci) to fail a pipeline.

Rules from the registry need the network, and by default metrics are sent; use --metrics off to disable, --config <file> to stay local.

SCANNING

`semgrep scan`
Scan the current directory

`semgrep scan --config auto`
Registry rules chosen for your code

`semgrep scan --config <file|dir>`
Run local rule files

`semgrep scan --config p/<ruleset>`
Run a registry ruleset

`semgrep scan <path>`
Scan only a file or folder

SCANNING (cont.)

`semgrep scan --severity <level>`
Filter by INFO, WARNING, ERROR

`semgrep scan --exclude-rule <id>`
Skip a rule

`semgrep scan --include <glob>`
Only paths matching glob

`semgrep scan --exclude <glob>`
Skip paths matching glob

`semgrep scan -j <n>`
Number of parallel jobs

`semgrep scan --timeout <sec>`
Per-rule per-file time limit

`semgrep scan --max-target-bytes <n>`
Raise the file size limit

`semgrep scan --metrics off`
Disable metrics

AD-HOC PATTERNS

`semgrep -e '<pattern>' -l <lang> <path>`
Search with a pattern

`semgrep -e 'eval(...)' -l python .`
Find any eval call

AD-HOC PATTERNS (cont.)

`semgrep -e '$X == $X' -l <lang> .`
Metavariable, same value twice

`semgrep -e '<pat>' --replacement '<new>' -l <lang> --autofix`
Rewrite matches

Pattern syntax

Syntax	Matches
`...`	any arguments or statements
`$X`	any expression (metavariable)
`<... p ...>`	p nested anywhere inside

OUTPUT

`semgrep scan --json`
JSON to stdout

`semgrep scan --sarif -o <file>`
SARIF report to a file

`semgrep scan --json-output <file>`
JSON file, text on screen

`semgrep scan --junit-xml`
JUnit XML output

`semgrep scan --gitlab-sast`
GitLab SAST format

OUTPUT (cont.)

`semgrep scan --dataflow-traces`
Show taint paths in text

`semgrep scan -q`
Quiet: findings only

CI & BASELINE

`semgrep ci`
Scan in CI, fail on blocking findings

`semgrep ci --baseline-commit <ref>`
Only findings new since ref

`semgrep scan --baseline-commit <ref>`
Same for plain scan

`semgrep scan --error`
Exit 1 when findings exist

`semgrep login`
Authenticate with semgrep.dev

IGNORING

`# nosemgrep`
Ignore this line or the next

`# nosemgrep: <rule-id>`
Ignore one rule only

`.semgrepignore`
Paths to skip, gitignore syntax

RULES

`semgrep scan --validate --config <file>`
Check rule files are valid

`semgrep scan --test <dir>`
Run rule tests

Exit codes

Code	Meaning
0	ok, or findings without --error
1	findings with --error
2	fatal error
4	invalid pattern in rule
7	invalid rule or missing config

RECIPES

Fail CI on serious findings

```
semgrep scan --config auto \
  --severity ERROR --error
```

Only new findings in a pull request

```
semgrep scan --config p/ci \
  --baseline-commit origin/
main --error
```

Quick custom rule check

```
semgrep -e
'requests.get(..,
verify=False)' \
  -l python src/
```

Report for code scanning

```
semgrep scan --config auto \
  --sarif -o semgrep.sarif
```