# Checkov IaC security scanner

## CONCEPTS

Checkov scans infrastructure-as-code (Terraform, Kubernetes, Dockerfiles, CloudFormation, CI configs) for misconfigurations without deploying anything.

**IaC files** tf, yaml, Dockerfile
↓ `checkov -d`
**Framework runners** auto-detected
↓ `policies`
**Checks** CKV_* IDs
↓ `-o`
**Report** pass / fail / skip

Every check has an ID like CKV_AWS_20; you use these IDs to select, skip and suppress checks.

By default all matching frameworks run; narrow with --framework to speed up large repos.

## SETUP

**`pip install checkov`**
Install with pip

**`checkov --version`**
Show the version

**`checkov --help`**
List all options

**`checkov --list`**
List available checks

## SCANNING

**`checkov -d <dir>`**
Scan a directory recursively

**`checkov -f <file>`**
Scan a single file

**`checkov -f <a.tf> -f <b.tf>`**
Scan several files

**`checkov -f tfplan.json --repo-root-for-plan-enrichment <dir>`**
Scan a Terraform plan

**`checkov -d <dir> --skip-path <regex>`**
Ignore paths matching a regex

**`checkov -d <dir> --download-external-modules true`**
Fetch remote Terraform modules

## FRAMEWORKS

**`checkov -d <dir> --framework <name>`**
Run only these frameworks

**`checkov -d <dir> --skip-framework <name>`**
Leave frameworks out

**`checkov -d <dir> --framework secrets`**
Hunt for hard-coded secrets

**`checkov -d <dir> --framework kubernetes helm`**
Scan manifests and charts

**`checkov -d <dir> --framework dockerfile`**
Lint Dockerfiles

**`checkov -d <dir> --framework github_actions`**
Scan GitHub workflows

Common names: terraform, terraform_plan, kubernetes, helm, kustomize, dockerfile, cloudformation, arm, bicep, github_actions, gitlab_ci, secrets.

## CHOOSING CHECKS

**`checkov -d <dir> -c <id>,<id>`**
Run only these checks

**`checkov -d <dir> --skip-check <id>,<id>`**
Skip these checks

**`checkov -d <dir> --external-checks-dir <dir>`**
Load custom Python or YAML checks

## OUTPUT

**`checkov -d <dir> --compact`**
Hide code snippets

**`checkov -d <dir> --quiet`**
Show only failed checks

**`checkov -d <dir> -o <format>`**
Pick an output format

**`checkov -d <dir> -o json`**
JSON report

**`checkov -d <dir> -o sarif`**
SARIF for code scanning

**`checkov -d <dir> -o junitxml`**
JUnit report for CI

## OUTPUT (cont.)

**`checkov -d <dir> -o cli -o json --output-file-path console,<file>`**
Print and save in one run

**`checkov -d <dir> --output-file-path <dir>`**
Save reports in a folder

## EXIT CODES AND CI

**`checkov -d <dir> --soft-fail`**
Always exit 0

**`checkov -d <dir> --hard-fail-on <id|sev>`**
Fail only on these checks

**`checkov -d <dir> --soft-fail-on <id|sev>`**
Pass when only these fail

**Exit codes**

| Code | Meaning |
|---|---|
| `0` | All checks passed (or soft-fail) |
| `1` | At least one check failed |
| `2` | Crash or platform failure |

## BASELINES

**`checkov -d <dir> --create-baseline`**
Save current findings as a baseline

**`checkov -d <dir> --baseline <file>`**
Report only new failures

Baselines let you adopt Checkov on a legacy repo: block new problems now, fix old ones later.

## SUPPRESSING FINDINGS

**Terraform: inside the resource**

```
resource "aws_s3_bucket" "site" {
  #checkov:skip=CKV_AWS_20:public static site
}
```

**Kubernetes: annotation**

```
metadata:
  annotations:
    checkov.io/skip1: CKV_K8S_20=needs root for init
```

## CONFIG FILE

**`checkov --config-file <file>`**
Use a YAML config file