**✅ What is Checkov?**

Checkov is an open-source static code analysis tool developed by Bridgecrew (acquired by Prisma Cloud) that detects security and compliance misconfigurations in Infrastructure as Code (IaC) files before they are deployed.

**It supports scanning of:**

* **🧱 Terraform (HCL & JSON)**
* **🛡 CloudFormation (YAML & JSON)**
* **☸️ Kubernetes manifests**
* **🧩 Helm Charts**
* **🐳 Dockerfiles**
* **⚙️ ARM templates, Serverless, and more**

**🔐 Why Use Checkov?**

* **Catch vulnerabilities early: Detect risks like open security groups, unencrypted S3 buckets, hardcoded secrets, and more—before deploying to cloud.**
* **Supports enterprise policies: Use custom policies to enforce security and compliance standards.**
* **Lightweight and fast: CLI-based tool that can be integrated into any CI/CD pipeline.**
* **Cloud-agnostic: Works with AWS, Azure, GCP, Kubernetes, and custom platforms.**

**🧠 Key Features**

* **1000+ built-in policies (covering CIS Benchmarks, NIST, PCI-DSS, SOC2, HIPAA)**
* **Severity filtering (LOW, MEDIUM, HIGH, CRITICAL)**
* **Policy skipping and selective scanning**
* **JSON, JUnit, SARIF output formats**
* **Offline scanning supported**
* **Easy integration with CLI, Docker, or Python Virtual Environments**
* **Supports custom policies using Python**

**⚡ Real-World Use Cases**

* ✅ Detecting insecure S3 buckets (CKV\_AWS\_18)
* ✅ Catching IAM policies with wildcard actions (CKV\_AWS\_40)
* ✅ Preventing public-facing Kubernetes services (CKV\_K8S\_22)
* ✅ Scanning Dockerfiles for latest tag usage or missing USER

**✅ Step 1: Setup (Virtual Environment + Installation)**

sudo apt update

sudo apt install python3 python3-pip -y

# Setup virtual environment (optional but clean)

python3 -m venv venv

source venv/bin/activate

# Install Checkov

pip install checkov

**✅ Step 2: Common Checkov Scanning Commands**

# Scan current directory (auto-detect IaC types)

checkov -d .

# Scan a specific Terraform file

checkov -f main.tf

# Scan specific directory for Kubernetes YAMLs, Terraform, etc.

checkov -d /home/ubuntu/project-directory

# Scan a Dockerfile

checkov -f Dockerfile

# Scan a single Kubernetes manifest

checkov -f deployment.yaml

**✅ Step 3: Advanced Scan Control**

# Skip specific checks by ID

checkov -d . --skip-check CKV\_AWS\_130,CKV\_AWS\_23

# Run only specific checks

checkov -d . --check CKV\_AWS\_20,CKV\_K8S\_21

# Set severity filter: Only show HIGH and CRITICAL issues

checkov -d . --severity HIGH,CRITICAL

**✅ Step 4: Output Formatting and Reporting**

# Output as JSON and save to file

checkov -d . -o json > checkov-report.json

# Output as JUnit XML for CI tools

checkov -d . -o junitxml > junit-report.xml

# Output as SARIF format (for GitHub-compatible static analysis)

checkov -d . -o sarif > report.sarif

**✅ Step 5: Custom Policy Checks**

# Use your own custom checks (Python classes)

checkov -d . --external-checks-dir /path/to/my/custom/checks

**✅ Step 6: Archive/ZIP Scan**

# Scan a ZIP file containing IaC code

checkov -z iac-code.zip

**✅ Step 7: Helm Chart Scan**

# Render a Helm chart and scan

helm template mychart ./mychart > rendered.yaml

checkov -f rendered.yaml

**✅ Step 8: Performance and Miscellaneous Flags**

# Do not fetch remediation docs (useful for offline)

checkov -d . --no-guide

# Set quiet mode (less output)

checkov -d . -q

# Set soft-fail (exit 0 even on failures - useful in non-blocking scans)

checkov -d . --soft-fail

**✅ Step 9: Version and Help**

checkov --version

checkov --help

**✅ Example Combined Full Scan**

checkov -d . \

--check CKV\_AWS\_20,CKV\_K8S\_21 \

--skip-check CKV\_AWS\_130 \

--severity HIGH,CRITICAL \

-o json > secure-scan.json