Applying CodeQL Scanning

A Practical, Exam-Oriented Guide

How to introduce CodeQL analysis to a repository, source and configure queries, fan out multi-language scans with matrix strategies, and run scans via GitHub Actions vs. the CodeQL CLI.

Learning Objectives

- Introduce a CodeQL analysis workflow to a repository.
- List where CodeQL queries can be specified and referenced.
- Configure a language matrix in a CodeQL workflow.
- Reference queries and configuration from public, private, and local sources.
- Execute scans using the CodeQL CLI and contrast with GitHub Actions.

1 Introduce CodeQL Analysis with GitHub Actions

Integrating CodeQL into your repository automates security analysis on `push`, `pull_request`, and on schedules. The standard starter workflow is shown below.

```yaml
# .github/workflows/codeql-analysis.yml
name: "CodeQL"

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]
  schedule:
    - cron: "0 6 * * 1"   # Mondays at 06:00 UTC

jobs:
  analyze:
    runs-on: ubuntu-latest

    strategy:
      fail-fast: false
      matrix:
        language: [ "python" ]  # add more languages as needed

    permissions:
      actions: read
      contents: read
      security-events: write

    steps:
      - name: Checkout
        uses: actions/checkout@v4

      - name: Initialize CodeQL
        uses: github/codeql-action/init@v3
        with:
          languages: ${{ matrix.language }}
          # Optional: query suites or config provided later sections

      - name: Autobuild
        uses: github/codeql-action/autobuild@v3

      - name: Analyze
        uses: github/codeql-action/analyze@v3
        with:
          category: "/language:${{ matrix.language }}"
```

2 Where Can CodeQL Queries Be Specified?

CodeQL queries are `.ql` files (often with a `qlpack.yml` and lock file for dependencies). You can point your workflow at queries from several locations:

1. Inline in the Workflow (least common for large queries)

```yaml
- name: Initialize CodeQL with inline queries
  uses: github/codeql-action/init@v3
  with:
    languages: ${{ matrix.language }}
    queries: |
      query: |
        import javascript
        from Expr e
        where e.toString() = "example"
        select e
```

2. Local Path in the Same Repository

```yaml
- name: Initialize CodeQL with local queries
  uses: github/codeql-action/init@v3
  with:
    languages: ${{ matrix.language }}
    queries: "./codeql/queries"   # directory or specific .ql file
```

3. Another Internal (Private) Repository

```yaml
- name: Initialize CodeQL (internal repo)
  uses: github/codeql-action/init@v3
  with:
    languages: ${{ matrix.language }}
    queries: "org-sec/codeql-queries/python@main:queries/special.ql"
```

4. Public Repository (e.g., GitHub's CodeQL packs)

```yaml
- name: Initialize CodeQL (public repo)
  uses: github/codeql-action/init@v3
  with:
    languages: ${{ matrix.language }}
    queries:
    ↪  "github/codeql:python/ql/src/codeql-suites/python-security-and-quality.qls"
```

Reliability Tip. When referencing remote content, prefer pinning to tags or commits (e.g., `@vX.Y` or a commit SHA) to avoid breakage when the upstream changes.

3 Configure a Language Matrix

Use `strategy.matrix` to parallelize analysis across languages (or other axes like OS).

```yaml
strategy:
  fail-fast: false
  matrix:
    language: [ "python", "javascript" ]

# Use the matrix context downstream
with:
  languages: ${{ matrix.language }}
```

4 Reference a CodeQL Configuration File

Centralize reusable settings (disabling noisy queries, adding custom packs, selecting suites, etc.).

Config in the Same Repository

```yaml
# .github/workflows/codeql-analysis.yml (excerpt)
- name: Initialize CodeQL (with config)
  uses: github/codeql-action/init@v3
  with:
    languages: ${{ matrix.language }}
    config-file: ".github/codeql/codeql-config.yml"
```

```yaml
# .github/codeql/codeql-config.yml
name: "org-defaults"
queries:
  - uses: security-extended
  - uses: security-and-quality
  - uses: ./queries/custom-sql-injection.ql
paths-ignore:
  - "vendor/**"
  - "third_party/**"
```

Config in a Remote Repository or URL

```yaml
- name: Initialize CodeQL (remote config)
  uses: github/codeql-action/init@v3
  with:
    languages: ${{ matrix.language }}
    config-file: "org-sec/codeql-configs@v1:python/config.yml"
    # For non-GitHub hosts, use a full URL if supported.
```

5 Execute Scans with the CodeQL CLI

Great for local research, quick checks, and iterative query development.

```
# 1) Create a database (per language) from source
codeql database create db-python \
  --language=python \
  --source-root=./src

# 2) Run a query or suite
codeql database analyze db-python \
  ./codeql/queries/custom.ql \
  --format=sarifv2.1.0 --output=results-python.sarif

# 3) Use packs / suites
codeql pack download codeql/python-queries
codeql database analyze db-python \
  codeql/python-queries:codeql-suites/python-security-and-quality \
  --format=sarifv2.1.0 --output=results.sarif

# 4) View results (example: VS Code SARIF viewer) or upload to GitHub code scanning
```

6 Actions vs. CLI — When to Use Which

- **GitHub Actions:** Continuous, automated scanning on PRs, pushes, and schedules; central results in code scanning alerts; scalable via matrices and runners.
- **CodeQL CLI:** Local, manual, and exploratory scanning; ideal for developing custom queries and validating hypotheses before baking into CI.

7 Case Studies & Practices

- **Public Queries ("Blue Yonder" example).** Fast access to community expertise. Risk: upstream changes or disappearance. Mitigate by pinning refs.
- **Private/Internal Packs ("Graphite Industries").** Strong control and governance. Ensure permissions and auth are configured for runners.
- **Local Queries in-Repo.** Highest stability and simplest pathing. Versioned with the application code.

Compilation Notes

This document uses `minted` for code highlighting. Compile with `-shell-escape`, for example:

```
latexmk -pdf -shell-escape applying-codeql-scanning.tex
```

This quick-reference was distilled from course-style notes on applying CodeQL scanning in practice.