GitHub Advanced Security (GHAS) — Quick Reference

Version: October 31, 2025 | Scope: Enterprise → Org → Repo

What GHAS Gives You

- **Code scanning (CodeQL)** — SAST with a code-as-data model.
- **Secret scanning** (+ push protection) — detect and prevent credential leaks.
- **Dependency insights** — advisories, alerts, automated PRs (Dependabot).
- **Security Overview dashboards** at Enterprise/Org/Repo.
- **Policy inheritance** to auto-enable for new repositories.
- **Shift-left integration** in PR checks and GitHub Actions.
- **Prioritized fix guidance** and audit trail via checks & reviews.
- **Container/image scanning** surfaced in Security Overview.
- **Compliance support**: enforce status checks before merge.

1 Key Concepts

Term	Meaning / Notes
GHAS	Paid add-on; unlocks code scanning, secret scanning (incl. push protection), and advanced dashboards.
GHEC	GitHub Enterprise Cloud (github.com).
GHES	GitHub Enterprise Server (self-hosted).
Security Overview	Aggregated dashboards: view risk & coverage, drill into alerts, track remediation.

2 Enablement Path (Top-down)

1. **Enterprise Settings** → `Policies` → `Code security and analysis`: prefer *inheritance* and default-on for new orgs/repos.
2. **Organization Settings** → `Security & analysis`: enable *Code scanning*, *Secret scanning* (with *Push protection*), and *Dependency alerts*.
3. **Repository Settings** → `Security & analysis`: confirm features & configure branch protection rules to *require* passing checks.

Minimum Safe Defaults

- Secret scanning **ON** & **Push protection ON** for all repos (including private).
- Code scanning **ON** with default CodeQL queries for primary languages.
- Dependency alerts **ON** & Dependabot security updates **ON**.
- Protected branches require: *All code scanning checks* and *No secret scanning violations* before merge.

3 CI/CD Integration

Code scanning (CodeQL) — GitHub Actions

Create `.github/workflows/codeql.yml`:

CodeQL code scanning for a multi-language repo

```yaml
name: CodeQL
on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]
  schedule:
    - cron:  '0 3 * * 0'   # weekly run (Sun @ 03:00 UTC)

jobs:
  analyze:
    runs-on: ubuntu-latest
    permissions:
      actions: read
      contents: read
      security-events: write
    strategy:
      fail-fast: false
      matrix:
        language: [ 'javascript', 'typescript', 'python', 'java' ]
    steps:
      - uses: actions/checkout@v4
      - uses: github/codeql-action/init@v3
        with:
          languages: ${{ matrix.language }}
      - uses: github/codeql-action/autobuild@v3
      - uses: github/codeql-action/analyze@v3
        with:
          category: "/language:${{ matrix.language }}"
```

Dependabot Security Updates

Create `.github/dependabot.yml`:

Dependabot security updates (weekly)

```yaml
version: 2
updates:
  - package-ecosystem: "npm"
    directory: "/"
    schedule: { interval: "weekly" }
  - package-ecosystem: "maven"
    directory: "/"
    schedule: { interval: "weekly" }
  - package-ecosystem: "pip"
    directory: "/"
    schedule: { interval: "weekly" }
```

Secret Scanning & Push Protection

- Enabled via `Settings` → `Security & analysis`. Configure org-wide defaults and *require fixes before merge* using branch protection.
- **Triage tip**: Verify, revoke, rotate, then remediate. Add dismissals only with justification (false positive, test credential, or mitigated).

4 Triage & Policy

Alert Handling Flow

1. **Open alert** → assess severity, exploitability, and reachability.
2. **Decide**: Fix now (*patch/PR*), mitigate (feature flag, config), or temporarily suppress (with justification & expiry).
3. **Track**: Link to issue/PR, assign owner, set SLA (e.g., Critical 24–48h, High 5d, Medium 10d, Low 30d).
4. **Verify**: Ensure status checks pass; close alert with reference to commit or PR.

Dismissal Reasons (Use Sparingly)

Reason	When Appropriate
False positive	Tool finding demonstrably incorrect for this code path.
Used in tests	Credential/pattern is non-production & gated to test fixtures.
Mitigated	Compensating control prevents exploitation (documented).
Won't fix (temporary)	Accepted risk with deadline, owner, and business justification.

5 AppSec Core Processes ↔ GHAS Features

AppSec Process	GHAS Tie-in
Secure SDLC Governance	Enforce branch protections; require checks from CodeQL & secret scanning before merge.
Threat Detection/Prevention	Secret scanning with push protection; Dependabot advisories.
Static Analysis (SAST)	CodeQL workflows on push, PR, and schedule.
Dependency/Container Risk	Dependabot PRs; surfaced in Security Overview.
Triage & Remediation	Use alert pages, issues/PR links, and dashboards to track closure & SLA.

6 Operational Tips

- **Auto-on for new repos**: set org policy to inherit security features by default.
- **Language coverage**: verify CodeQL supports your primary languages; supplement with third-party linters where needed.
- **Monorepos**: scope CodeQL with `paths`/`paths-ignore` to focus on critical dirs.
- **CI reliability**: pin actions to major versions and use weekly scheduled runs to catch drift.
- **Reporting**: use Security Overview exports and Issues/Projects for tracking KPIs (MTTR, % fixed by severity, coverage).

Merge Gates Checklist

- Code scanning checks ✓
- Secret scanning (no blocking secrets) ✓
- Dependency alerts addressed ✓
- Required reviewers approved ✓
- Status checks passing ✓
- Protected branch rules enforced ✓
- PR links to issue/ticket ✓
- CI passed (build, test) ✓
- Release notes updated ✓
- Ownership & SLA assigned ✓

This one-pager is a concise reference for enabling and operating GHAS across Enterprise → Org → Repo scopes. Adapt the examples to your repositories and compliance policies.