Administration of GitHub Advanced Security (GHAS)

Notes, SOP, and Example Markdown Templates

Learning Objectives

After completing this guide, you should be able to:

- Explain how GHAS features are enabled across enterprise, org, and repo scopes.
- Set org-level defaults and repo overrides with least-privilege RBAC.
- Use APIs and the `gh` CLI for automation and reporting.
- Standardize community health files and workflow templates via `.github`.

1 .github Org Repository

Suggested Structure

```
.github/
  CODE_OF_CONDUCT.md
  CONTRIBUTING.md
  SECURITY.md
  FUNDING.yml
  ISSUE_TEMPLATE/
    bug_report.md
    feature_request.md
  workflow-templates/
    codeql.yml
    codeql.properties.json   (template metadata)
```

2 Permissions, APIs, and CodeQL (Quick Reference)

List Code Scanning Alerts (repo scope)

```
gh api repos/OWNER/REPO/code-scanning/alerts --paginate --jq '.[].rule.id'
```

List Dependabot Alerts (org scope)

```
curl -s -H "Authorization: Bearer $GITHUB_TOKEN" \
  -H "Accept: application/vnd.github+json" \
  "https://api.github.com/orgs/ORG/dependabot/alerts?per_page=100"
```

List Secret Scanning Alerts (repo scope)

```
gh api repos/OWNER/REPO/secret-scanning/alerts --paginate
```

Baseline CodeQL Workflow (Template)

```yaml
name: CodeQL
on:
  push: { branches: ["main"] }
  pull_request: { branches: ["main"] }
  schedule: [ { cron: "23 3 * * 1" } ]

jobs:
  analyze:
    runs-on: ubuntu-latest
    permissions:
      security-events: write
      contents: read
      actions: read
    steps:
      - uses: actions/checkout@v4
      - uses: github/codeql-action/init@v3
        with:
          languages: auto
          # queries: security-extended,security-and-quality
      - uses: github/codeql-action/autobuild@v3
      - uses: github/codeql-action/analyze@v3
```

3 Markdown File Examples

This section provides copy-pasteable starting points for each Markdown file in the structure above. Replace placeholders like `ORG`, `CONTACT_EMAIL`, and `SECURITY_TEAM_URL`.

3.1 CODE_OF_CONDUCT.md

```
# Code of Conduct

## Our Pledge
We as members, contributors, and leaders pledge to make participation in our
community a harassment-free experience for everyone.

## Our Standards
Examples of behavior that contributes to a positive environment include:
- Demonstrating empathy and kindness
- Being respectful of differing viewpoints and experiences
- Accepting constructive feedback gracefully

Examples of unacceptable behavior include:
- Harassment, trolling, or derogatory comments
- Publishing others' private information without permission
- Other conduct which could reasonably be considered inappropriate

## Enforcement Responsibilities
Project maintainers are responsible for clarifying and enforcing our standards.
They will take appropriate and fair corrective action in response to any behavior
they deem inappropriate, threatening, offensive, or harmful.

## Scope
This Code applies within project spaces and in public spaces when an individual
is representing the project or its community.

## Enforcement
Instances of abusive, harassing, or otherwise unacceptable behavior may be
reported to the maintainers at CONTACT_EMAIL. All complaints will be reviewed
and investigated promptly and fairly.

## Attribution
This Code of Conduct is adapted from common open-source community guidelines.
```

3.2 CONTRIBUTING.md

```
# Contributing to ORG

Thanks for taking the time to contribute! The following is a set of guidelines
for contributing to projects under the ORG organization.

## Ground Rules
- Follow our [Code of Conduct](../CODE_OF_CONDUCT.md).
- Discuss significant changes via issues before opening a PR.
- Keep PRs focused and small when possible.

## Development Workflow
1. Fork the repo and create a feature branch:
   - `git checkout -b feat/short-description`
2. Run tests and linters locally.
3. Commit using conventional commits where possible.
4. Open a PR against `main` and fill out the PR template.

## Commit Message Guidelines (Conventional Commits)
- `feat: add new user profile endpoint`
- `fix: correct null pointer in auth middleware`
- `docs: update README for setup`
- `ci: bump CodeQL version to v3`

## Pull Request Checklist
- [ ] Tests added or updated
- [ ] Docs updated (README/SECURITY/CHANGELOG as applicable)
- [ ] CI is green
- [ ] No secrets or credentials included

## Issue Labels
- `bug`, `feature`, `documentation`, `security`, `good first issue`

## Licensing
By contributing, you agree that your contributions will be licensed under the
project's license.
```

3.3 SECURITY.md

```
# Security Policy

## Supported Versions
We generally support security fixes for the latest release and the most recent
minor version line. Older releases may receive fixes at our discretion.

## Reporting a Vulnerability
Please email CONTACT_EMAIL or use our private intake form:
SECURITY_TEAM_URL

**Do not** open public issues for suspected vulnerabilities.

Provide as much detail as possible:
- Affected version(s) and component(s)
- Reproduction steps and proof of concept
- Impact assessment (confidentiality, integrity, availability)
- Any suggested mitigations

We will acknowledge receipt within 2 business days and provide regular updates
until resolution.

## Disclosure Policy
We follow coordinated disclosure. After a fix is available and users have had
a reasonable window to update, we may publish details in release notes.

## Security Hardening Guidance
- Enable branch protection and required status checks
- Use GHAS features (secret scanning, Dependabot, CodeQL)
- Rotate credentials regularly and prefer OIDC for CI
```

3.4 ISSUE_TEMPLATE/bug_report.md

```
---
name: "Bug report"
about: "Create a report to help us improve"
title: "[Bug]: short summary"
labels: ["bug"]
assignees: []
---

## Describe the bug
A clear and concise description of the problem.

## To Reproduce
Steps to reproduce the behavior:
1. Go to '...'
2. Run '...'
3. Observe '...'

## Expected behavior
What you expected to happen.

## Screenshots or logs
If applicable, add screenshots or logs.

## Environment
- OS: e.g., Ubuntu 22.04
- Runtime/SDK: e.g., Node.js 20.x
- Version/Commit: e.g., v1.2.3 / abc123

## Additional context
Add any other context about the problem here.
```

3.5 ISSUE_TEMPLATE/feature_request.md

```
---
name: "Feature request"
about: "Suggest an idea for this project"
title: "[Feature]: short summary"
labels: ["feature"]
assignees: []
---

## Motivation / Problem Statement
Is your request related to a problem? Please describe it clearly.

## Proposed Solution
Describe the solution you'd like, including API or UX changes if relevant.

## Alternatives
Describe any alternative solutions or features you've considered.

## Acceptance Criteria
- [ ] Clear, testable criteria 1
- [ ] Clear, testable criteria 2
- [ ] Docs updated

## Additional context
Add any other context or mockups here.
```

4 Policy and Inheritance Cheatsheet

Level	What to Prefer Centralizing	Typical Overrides
Enterprise	Global minimums/guardrails, audit, org defaults	Org-specific constraints
Organization	Default GHAS enablement, templates, required checks	Repo languages, schedules, build steps
Repository	Local exceptions, custom queries/build	N/A (justify exceptions)

5 Operational Playbook

1. Establish governance: `.github` repo, community health files, templates.
2. Enable GHAS org-wide: secret scanning, Dependabot, CodeQL.
3. RBAC: Security teams with least-privilege permissions.
4. Triage and SLAs: severity, dismissal reasons, escalation paths.
5. Automate: APIs and `gh` for alert export, dashboards, tickets.
6. Measure: MTTR, reopened rates, coverage, query packs.