**4. Security and compliance**

**└ 4.4 Security and compliance scanning**

**└ 4.4.1 Dependency/code/secret/license scanning**

---

1. What tools are commonly used for dependency scanning in Azure DevOps and GitHub?
2. How does GitHub Dependabot work and what vulnerabilities can it detect?
3. What is CodeQL and how is it integrated into GitHub Actions?
4. How do you configure secret scanning in GitHub repositories?
5. What license compliance features does GitHub provide?
6. How do you enable and enforce security scanning policies in GitHub Advanced Security?
7. How can Microsoft Defender for DevOps be integrated for scanning in Azure DevOps?
8. What are the differences between static code analysis and secret scanning?
9. How are vulnerabilities tracked and managed after detection in GitHub?
10. What best practices ensure effective and compliant security/compliance scanning pipelines?

---

**1. What tools are commonly used for dependency scanning in Azure DevOps and GitHub?**

- *GitHub* uses *Dependabot* and *GitHub Advanced Security*;
- *Azure DevOps* relies on integrations like *Microsoft Defender for DevOps* and third-party tools like *WhiteSource* or *Snyk*.

---

**2. How does GitHub Dependabot work and what vulnerabilities can it detect?**

*Dependabot* scans dependency files (e.g., package.json, requirements.txt) for known CVEs and suggests automatic PRs with updated versions. It uses the *GitHub Advisory Database*.

---

**3. What is CodeQL and how is it integrated into GitHub Actions?**

*CodeQL* performs static code analysis to find security vulnerabilities. It runs as a *GitHub Action* and supports scheduled or PR-triggered scans across multiple languages.

---

**4. How do you configure secret scanning in GitHub repositories?**

Enable it under *repository settings → Security → Code security and analysis*. *GitHub* scans commits and PRs for credential patterns and alerts repo admins.

---

**5. What license compliance features does GitHub provide?**

*GitHub* scans dependencies to detect license types and highlight potential violations. License metadata is shown in dependency graphs; violations can trigger *Dependabot* alerts.

---

**6. How do you enable and enforce security scanning policies in GitHub Advanced Security?**

Enable Advanced Security per repository. Configure required status checks (e.g., *CodeQL*, secret scanning) in branch protection rules to block merging if checks fail.

---

**7. How can Microsoft Defender for DevOps be integrated for scanning in Azure DevOps?**

Connect via *Microsoft Defender for Cloud → DevOps Security → Connect Azure DevOps*. It enables dependency, IaC, and container scanning and centralizes findings in Defender.

---

**8. What are the differences between static code analysis and secret scanning?**

Static analysis (e.g., *CodeQL*) finds logic flaws in code structure; secret scanning searches for exposed credentials and tokens in code and commits.

**9. How are vulnerabilities tracked and managed after detection in GitHub?**
Vulnerabilities appear in the *Security tab* → *Dependabot alerts*. You can view affected versions, upgrade recommendations, and dismiss alerts with a justification.

---

**10. What best practices ensure effective and compliant security/compliance scanning pipelines?**
Use CI-integrated scanning (e.g., *CodeQL*, secret scan), auto-fix with *Dependabot*, enforce branch protections, monitor security dashboards, and integrate alerting with workflows.