4. Security and compliance

└ **4.3 Secret and sensitive data management**

└ **4.3.2 Secrets in GitHub Actions/Azure Pipelines**

---

1. How are secrets stored and accessed in GitHub Actions?
2. What are environment secrets in GitHub Actions and how do they differ from repository secrets?
3. How do you securely use secrets in GitHub Actions workflows?
4. How do you define and use secrets in Azure Pipelines?
5. What are secure files in Azure Pipelines and when should you use them?
6. What are common mistakes that expose secrets in CI/CD workflows?
7. How do you use variable groups to manage secrets across pipelines in Azure DevOps?
8. What permissions are required to access secrets in GitHub and Azure Pipelines?
9. How can secrets be injected into runtime containers securely in pipelines?
10. What are the best practices for managing secret lifecycle in DevOps pipelines?

---

1. How are secrets stored and accessed in GitHub Actions?

Secrets are added under *Settings > Secrets and variables > Actions*. Access them in workflows using `${{ secrets.SECRET_NAME }}`.

---

2. What are environment secrets in GitHub Actions and how do they differ from repository secrets?

Environment secrets are tied to specific deployment environments and support protection rules. Repository secrets apply globally to all workflows.

---

3. How do you securely use secrets in GitHub Actions workflows?

Use `${{ secrets.NAME }}` in workflow YAML. Never echo secrets or log them. Use secrets.* only within run blocks or secure inputs.

---

4. How do you define and use secrets in Azure Pipelines?

Define secrets in variable groups or pipeline variables (isSecret: true). Access them in scripts via `$(secretName)` or environment variables.

---

5. What are secure files in Azure Pipelines and when should you use them?

Secure files are encrypted files (e.g., certificates) uploaded via the UI or API. Use them when pipelines need non-text secrets like PFX or SSH keys.

---

6. What are common mistakes that expose secrets in CI/CD workflows?

- Echoing secrets to logs
- Using secrets in non-secure scripts
- Committing secrets to source control
- Disabling scrubbers or secret masking

---

7. How do you use variable groups to manage secrets across pipelines in Azure DevOps?

Create a variable group, mark secrets as secret, and link the group to pipelines. Requires linking authorization for YAML usage.

---

8. What permissions are required to access secrets in GitHub and Azure Pipelines?

- GitHub: Write access to secrets and workflow access permissions
- Azure DevOps: Pipeline or service connection must have access to variable groups/secure files

9. How can secrets be injected into runtime containers securely in pipelines?
Pass them as environment variables or mount secure files.
In GitHub, use env: block; in Azure, use environment: or script variables.

---

10. What are the best practices for managing secret lifecycle in DevOps pipelines?

- Rotate secrets regularly
- Use Key Vault or GitHub Environments for sensitive scopes
- Avoid hardcoding; retrieve secrets dynamically
- Audit and clean unused secrets periodically