4. Security and compliance

└ 4.4 Security and compliance scanning

└ 4.4.2 Microsoft Defender for DevOps Security

---

1. What is Microsoft Defender for DevOps and what problems does it solve?
2. How do you onboard Azure DevOps and GitHub into Microsoft Defender for DevOps?
3. What types of security risks does Defender for DevOps detect?
4. Where are Defender for DevOps findings surfaced and managed?
5. How is remediation workflow handled for vulnerabilities in Defender for DevOps?
6. How does Defender for DevOps integrate with GitHub Advanced Security?
7. What permissions are required to connect GitHub or Azure DevOps to Defender for DevOps?
8. How do you enable continuous monitoring across multiple repositories?
9. How can you enforce policies based on Defender for DevOps findings?
10. What best practices ensure effective implementation of Defender for DevOps?

---

1. What is Microsoft Defender for DevOps and what problems does it solve?

It's a cloud-native security tool that integrates with *Azure DevOps* and *GitHub* to detect

- misconfigurations,
- exposed secrets,
- vulnerable dependencies,
- and infrastructure risks

across the DevOps pipeline.

---

2. How do you onboard Azure DevOps and GitHub into Microsoft Defender for DevOps?

In *Microsoft Defender for Cloud*, go to *DevOps Security → Environment Settings → Add DevOps platform*. Authenticate and connect your *Azure DevOps* org or *GitHub* repo via OAuth or service connection.

---

3. What types of security risks does Defender for DevOps detect?

It identifies

- exposed secrets,
- dependency vulnerabilities,
- IaC misconfigurations,
- weak configurations in workflows,
- and usage of unapproved tools or practices.

---

4. Where are Defender for DevOps findings surfaced and managed?

Findings are centralized in *Microsoft Defender for Cloud* under *DevOps Security* and *Security Recommendations*. Integration with *Azure Security Center* enables correlation with broader cloud risks.

---

5. How is remediation workflow handled for vulnerabilities in Defender for DevOps?

Defender provides actionable remediation steps and enables exporting findings to *Azure Boards*, *GitHub issues*, or third-party ticketing systems. Findings can be prioritized by severity.

---

6. How does Defender for DevOps integrate with GitHub Advanced Security?

Defender imports results from *GitHub Advanced Security* (CodeQL, secret scanning, dependency scanning) to enrich risk context and provide a unified dashboard for security posture.

7. What permissions are required to connect GitHub or Azure DevOps to Defender for DevOps?
You need Owner/admin privileges on the *GitHub* org or *Azure DevOps* project. OAuth permissions must include access to metadata, repos, and workflows. *Defender* must be granted permission to scan code.

---

8. How do you enable continuous monitoring across multiple repositories?
Connect the org level (not just individual repos), enable auto-discovery of new repos, and configure scanning rules and policies to apply by default across all new projects.

---

9. How can you enforce policies based on Defender for DevOps findings?
Integrate with *Azure Policy* or *GitHub* branch protection rules. Use conditional access policies or block deployments based on unresolved critical issues via pipeline gates.

---

10. What best practices ensure effective implementation of Defender for DevOps?
Onboard all repos/orgs, enable integration with *GitHub Advanced Security*, triage alerts regularly, automate issue tracking, review pipeline permissions, and monitor via centralized dashboards.