GH-500 Dumps Download, Practical GH-500 Information

P.S. Free & New GH-500 dumps are available on Google Drive shared by TrainingDumps: https://drive.google.com/open?id=17cLiqr3VbFvtAVJ6MrOtAM47ADVeIQyy

Generally speaking, preparing for the GH-500 exam is a very hard and even some suffering process. Because time is limited, sometimes we have to spare time to do other things to review the exam content, which makes the preparation process full of pressure and anxiety. But from the point of view of customers, our GH-500 Study Materials will not let you suffer from this. As mentioned above, our GH-500 study materials have been carefully written, each topic is the essence of the content. Only should you spend about 20 - 30 hours to study GH-500 study materials carefully can you take the exam.

Our GH-500 test braindumps are in the leading position in the editorial market, and our advanced operating system for GH-500 latest exam torrent has won wide recognition. As long as you choose our GH-500 exam questions and pay successfully, you do not have to worry about receiving our learning materials for a long time. We assure you that you only need to wait 5-10 minutes and you will receive our GH-500 Exam Questions which are sent by our system. When you start learning, you will find a lot of small buttons, which are designed carefully. You can choose different ways of operation according to your learning habits to help you learn effectively.

>> GH-500 Dumps Download <<

Clear the Microsoft GH-500 Exam with TrainingDumps

Our GH-500 exam prep has already become a famous brand all over the world in this field since we have engaged in compiling the GH-500 practice materials for more than ten years and have got a fruitful outcome. You are welcome to download the free demos to have a general idea about our GH-500study questions. Since different people have different preferences, we have prepared three kinds of different versions of our GH-500 training guide: PDF, Online App and software.

Microsoft GH-500 Exam Syllabus Topics:

Topic	Details
Topic 1	• Configure and use secret scanning: This domain targets DevOps Engineers and Security Analysts with the skills to configure and manage secret scanning. It includes understanding what secret scanning is and its push protection capability to prevent secret leaks. Candidates differentiate secret scanning availability in public versus private repositories, enable scanning in private repos, and learn how to respond appropriately to alerts. The domain covers alert generation criteria for secrets, user role-based alert visibility and notification, customizing default scanning behavior, assigning alert recipients beyond admins, excluding files from scans, and enabling custom secret scanning within repositories.

Topic 2	• Describe GitHub Advanced Security best practices, results, and how to take corrective measures: This section evaluates skills of Security Managers and Development Team Leads in effectively handling GHAS results and applying best practices. It includes using Common Vulnerabilities and Exposures (CVE) and Common Weakness Enumeration (CWE) identifiers to describe alerts and suggest remediation, decision-making processes for closing or dismissing alerts including documentation and data-based decisions, understanding default CodeQL query suites, how CodeQL analyzes compiled versus interpreted languages, the roles and responsibilities of development and security teams in workflows, adjusting severity thresholds for code scanning pull request status checks, prioritizing secret scanning remediation with filters, enforcing CodeQL and Dependency Review workflows via repository rulesets, and configuring code scanning, secret scanning, and dependency analysis to detect and remediate vulnerabilities earlier in the development lifecycle, such as during pull requests or by enabling push protection.
Topic 3	• Configure and use Code Scanning with CodeQL: This domain measures skills of Application Security Analysts and DevSecOps Engineers in code scanning using both CodeQL and third-party tools. It covers enabling code scanning, the role of code scanning in the development lifecycle, differences between enabling CodeQL versus third-party analysis, implementing CodeQL in GitHub Actions workflows versus other CI tools, uploading SARIF results, configuring workflow frequency and triggering events, editing workflow templates for active repositories, viewing CodeQL scan results, troubleshooting workflow failures and customizing configurations, analyzing data flows through code, interpreting code scanning alerts with linked documentation, deciding when to dismiss alerts, understanding CodeQL limitations related to compilation and language support, and defining SARIF categories.
Topic 4	• Describe the GHAS security features and functionality: This section of the exam measures skills of Security Engineers and Software Developers and covers understanding the role of GitHub Advanced Security (GHAS) features within the overall security ecosystem. Candidates learn to differentiate security features available automatically for open source projects versus those unlocked when GHAS is paired with GitHub Enterprise Cloud (GHEC) or GitHub Enterprise Server (GHES). The domain includes knowledge of Security Overview dashboards, the distinctions between secret scanning and code scanning, and how secret scanning, code scanning, and Dependabot work together to secure the software development lifecycle. It also covers scenarios contrasting isolated security reviews with integrated security throughout the development lifecycle, how vulnerable dependencies are detected using manifests and vulnerability databases, appropriate responses to alerts, the risks of ignoring alerts, developer responsibilities for alerts, access management for viewing alerts, and the placement of Dependabot alerts in the development process.
Topic 5	• Configure and use Dependabot and Dependency Review: Focused on Software Engineers and Vulnerability Management Specialists, this section describes tools for managing vulnerabilities in dependencies. Candidates learn about the dependency graph and how it is generated, the concept and format of the Software Bill of Materials (SBOM), definitions of dependency vulnerabilities, Dependabot alerts and security updates, and Dependency Review functionality. It covers how alerts are generated based on the dependency graph and GitHub Advisory Database, differences between Dependabot and Dependency Review, enabling and configuring these tools in private repositories and organizations, default alert settings, required permissions, creating Dependabot configuration files and rules to auto-dismiss alerts, setting up Dependency Review workflows including license checks and severity thresholds, configuring notifications, identifying vulnerabilities from alerts and pull requests, enabling security updates, and taking remediation actions including testing and merging pull requests.

Microsoft GitHub Advanced Security Sample Questions (Q31-Q36):

NEW QUESTION # 31
Which details do you have to provide to create a custom pattern for secret scanning? (Each answer presents part of the solution. Choose two.)

- A. The name of the pattern
- B. Additional match requirements for the secret format
- C. The secret format
- D. A list of repositories to scan

Answer: A,C

Explanation:
When defining a custom pattern for secret scanning, two key fields are required:
Name of the pattern: A unique label to identify the pattern
Secret format: A regular expression that defines what the secret looks like (e.g., token format) You can optionally specify additional match requirements (like required context keywords), but they're not mandatory. Listing repositories is also not part of the required fields during pattern creation.

NEW QUESTION # 32
What is the purpose of the SECURITY.md file in a GitHub repository?

- A. support.md
- B. contributing.md
- C. security.md
- D. readme.md

Answer: C

Explanation:
The correct place to look is the SECURITY.md file. This file provides contributors and security researchers with instructions on how to responsibly report vulnerabilities. It may include contact methods, preferred communication channels (e.g., security team email), and disclosure guidelines.
This file is considered a GitHub best practice and, when present, activates a "Report a vulnerability" button in the repository's Security tab.

NEW QUESTION # 33
Which syntax in a query suite tells CodeQL to look for one or more specified .ql files?

- A. query
- B. qlpack
- C. qls

Answer: A

Explanation:
In a query suite (a .qls file), the **query** key is used to specify the paths to one or more .ql files that should be included in the suite.
Example:
- query: path/to/query.ql
qls is the file format.
qlpack is used for packaging queries, not in suite syntax.

NEW QUESTION # 34
Assuming there is no custom Dependabot behavior configured, where possible, what does Dependabot do after sending an alert about a vulnerable dependency in a repository?

- A. Scans any push to all branches and generates an alert for each vulnerable repository
- B. Constructs a graph of all the repository's dependencies and public dependents for the default branch
- C. Creates a pull request to upgrade the vulnerable dependency to the minimum possible secure version
- D. Scans repositories for vulnerable dependencies on a schedule and adds those files to a manifest

Answer: C

Explanation:
After generating an alert for a vulnerable dependency, Dependabot automatically attempts to create a pull request to upgrade that dependency to the minimum required secure version-if a fix is available and compatible with your project.
This automated PR helps teams fix vulnerabilities quickly with minimal manual intervention. You can also configure update behaviors using dependabot.yml, but in the default state, PR creation is automatic.

NEW QUESTION # 35
Which of the following options would close a Dependabot alert?

- A. Viewing the Dependabot alert on the Dependabot alerts tab of your repository
- B. Leaving the repository in its current state
- C. Viewing the dependency graph
- D. Creating a pull request to resolve the vulnerability that will be approved and merged

Answer: D

Explanation:
A Dependabot alert is only marked as resolved when the related vulnerability is no longer present in your code - specifically after you merge a pull request that updates the vulnerable dependency.
Simply viewing alerts or graphs does not affect their status. Ignoring the alert by leaving the repo unchanged keeps the vulnerability active and unresolved.

NEW QUESTION # 36
.....