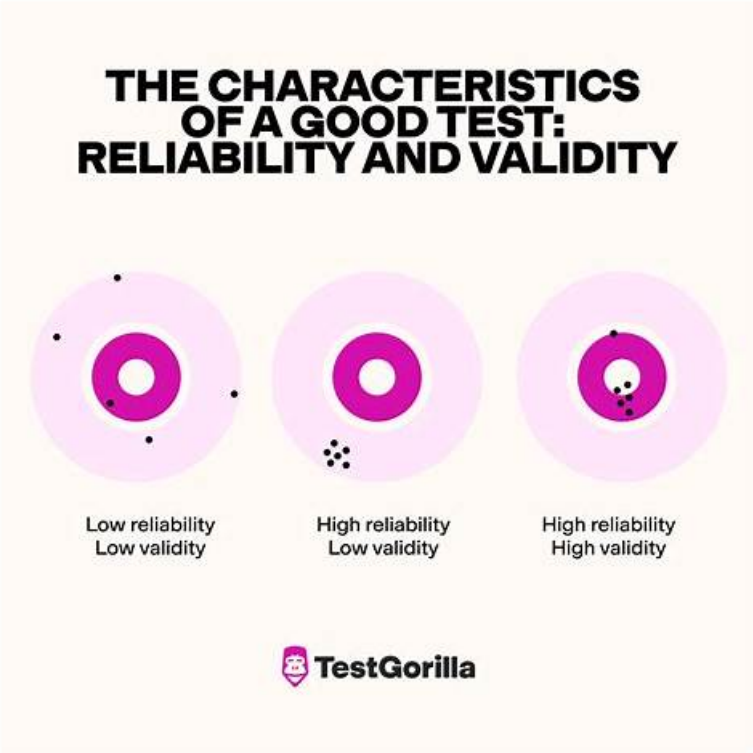
Valid 312-97 Test Sample | 312-97 Valid Test Tips



P.S. Free & New 312-97 dumps are available on Google Drive shared by ExamsReviews: https://drive.google.com/open?id=1PFbo9FeK-VTxGpoxtaLAySyPjuA1NGkl

Our EC-Council Certified DevSecOps Engineer (ECDE) (312-97) PDF file is portable which means customers can carry this real questions document to any place. You just need smartphones, or laptops, to access this EC-Council Certified DevSecOps Engineer (ECDE) (312-97) PDF format. These EC-Council Certified DevSecOps Engineer (ECDE) (312-97) questions PDFs are also printable. So candidates who prefer to study in the old way which is paper study can print EC-Council Certified DevSecOps Engineer (ECDE) (312-97) questions PDF as well.

ECCouncil 312-97 Exam Syllabus Topics:

Topic	Details
Topic 1	• Understanding DevOps Culture: This module introduces DevOps principles, covering cultural and technical foundations that emphasize collaboration between development and operations teams. It addresses automation, CI • CD practices, continuous improvement, and the essential communication patterns needed for faster, reliable software delivery.
Topic 2	• DevSecOps Pipeline - Build and Test Stage: This module explores integrating automated security testing into build and testing processes through CI pipelines. It covers SAST and DAST approaches to identify and address vulnerabilities early in development.
Topic 3	• DevSecOps Pipeline - Operate and Monitor Stage: This module focuses on securing operational environments and implementing continuous monitoring for security incidents. It covers logging, monitoring, incident response, and SIEM tools for maintaining security visibility and threat identification.

>> Valid 312-97 Test Sample <<

Unparalleled Valid 312-97 Test Sample & Leader in Qualification Exams & Perfect 312-97: EC-Council Certified DevSecOps Engineer (ECDE)

The passing rate of our 312-97 training quiz is 99% and the hit rate is also high. Our professional expert team seizes the focus of the exam and chooses the most important questions and answers which has simplified the important 312-97 information and follow the latest trend to make the client learn easily and efficiently. We update the 312-97 Study Materials frequently to let the client practice more. We provide the function to stimulate the 312-97 exam and the timing function of our 312-97 study materials to adjust your speed to answer the questions. You will pass the 312-97 exam easily.

ECCouncil EC-Council Certified DevSecOps Engineer (ECDE) Sample Questions (Q37-Q42):

NEW QUESTION # 37
(Patrick Fisher is a DevSecOps engineer in an IT company that develops software products and web applications. He is using IAST to analyze code for security vulnerabilities and to view real-time reports of the security issues. Patrick is using IAST in development, QA, and production stages to detect the vulnerabilities from the early stage of development, reduce the remediation cost, and keep the application secure. How can IAST perform SAST on every line of code and DAST on every request and response?.)

- A. Because IAST has access to server and local machine.
- B. Because IAST has access to internal and external agents.
- C. Because IAST has access to the code and HTTP traffic.
- D. Because IAST has access to offline and runtime environment.

Answer: C

Explanation:
Interactive Application Security Testing (IAST) works by instrumenting the application at runtime, allowing it to observe both thesource code execution pathsand theHTTP requests and responsesflowing through the application. Because of this dual visibility, IAST can analyze every executed line of code (similar to SAST) while also monitoring real-time application behavior (similar to DAST). This unique capability enables highly accurate vulnerability detection with fewer false positives. The other options do not correctly explain how IAST achieves this hybrid analysis. Access to both code and HTTP traffic is what allows IAST to bridge static and dynamic testing techniques, making it highly effective across development, QA, and production environments.
==========

NEW QUESTION # 38
(Judi Dench has recently joined an IT company as a DevSecOps engineer. Her organization develops software products and web applications related to electrical engineering. Judi would like to use Anchore tool for container vulnerability scanning and Software Bill of Materials (SBOM) generation. Using Anchore grype, she would like to scan the container images and file systems for known vulnerabilities, and would like to find vulnerabilities in major operating system packages such as Alpine, CentOS, Ubuntu, etc. as well as language specific packages such as Ruby, Java, etc. Which of the following commands should Judi run to scan for vulnerabilities in the image using grype?)

- A. grype < image >.
- B. grype < image > --scope all-layers.
- C. grype packages < image >.
- D. grype packages < image > --scope all-layers.

Answer: B

Explanation:
Grype is a vulnerability scanning tool used to analyze container images and file systems for known vulnerabilities across operating system and application dependencies. The most effective way to perform a comprehensive scan is by running the grype <image> --scope all-layers command. This ensures that vulnerabilities are detected acrossall layersof the container image, not just the final runtime layer. Containers often inherit vulnerabilities from base images or intermediate layers, making full-layer scanning essential. The packages subcommand is used for listing detected packages rather than performing vulnerability analysis.
Running Grype during the Build and Test stage allows DevSecOps teams to identify vulnerable base images and dependencies early, reducing the risk of deploying insecure containers into production and supporting secure container lifecycle management.
==========

NEW QUESTION # 39
(Charles Drew has been working as a DevSecOps team leader in an IT company located in Nashville, Tennessee. He would like to look at the applications from an attacker's perspective and make security a part of the organizations' culture. Imagine, you are working under Charles as a DevSecOps engineer. Charles has asked you to install ThreatPlaybook, which is a unified DevSecOps Framework that allows you to go from iterative, collaborative threat modeling to application security testing orchestration. After installation, you must configure ThreatPlaybook CLI; therefore, you have created a directory for the project and then you go to the current directory where you would like to configure ThreatPlaybook. Which of the following commands will you use to configure ThreatPlaybook? (Here, < your-email > represents your email id; < host info > represents IP address; and < port > represents the nginx port.))

- A. playbook configure -e < your-email > -u < host-info > -p < port >.
- B. playbook configure -e < your-email > -h < host-info > -p < port >.
- C. ThreatPlaybook configure -e < your-email > -u < host-info > -p < port >.
- D. ThreatPlaybook configure -e < your-email > -h < host-info > -p < port >.

Answer: D

Explanation:
ThreatPlaybook CLI is configured using the ThreatPlaybook configure command, which initializes the CLI with the required connection and user details. The -e option is used to specify the user's email address, the -h option defines the host information such as IP address or hostname, and the -p option specifies the port number. This configuration enables the CLI to securely communicate with the ThreatPlaybook service for orchestrating threat modeling and application security testing workflows. Options that use playbook configure are incorrect because the executable name is explicitly ThreatPlaybook. Options using -u instead of -h do not correctly specify host information. Configuring ThreatPlaybook during the Plan stage helps teams adopt an attacker's mindset early, embedding security into the organization's culture and ensuring threats are identified and addressed before development and deployment activities begin.
==========

NEW QUESTION # 40
(Craig Kelly has been working as a software development team leader in an IT company over the past 8 years.
His team is working on the development of an Android application product. Sandra Oliver, a DevSecOps engineer, used DAST tools and fuzz testing to perform advanced checks on the Android application product and detected critical and high severity issues. She provided the information about the security issues and the recommendations to mitigate them to Craig's team. Which type of security checks performed by Sandra involve detection of critical and high severity issues using DAST tools and fuzz testing?)

- A. Deploy-time checks.
- B. Build-time checks.
- C. Test-time checks.
- D. Commit-time checks.

Answer: C

Explanation:
Dynamic Application Security Testing (DAST) and fuzz testing require a running application in order to actively probe for vulnerabilities such as injection flaws, authentication bypasses, and improper input handling. These techniques are therefore performed after the application has been built and deployed to a testing environment, categorizing them astest-time checks. Commit-time and build-time checks rely primarily on static analysis and dependency scanning and do not exercise application behavior at runtime.
Deploy-time checks focus on configuration validation rather than aggressive attack simulation. Test-time checks are specifically designed to uncover critical and high-severity vulnerabilities by mimicking real-world attack scenarios. Performing DAST and fuzz testing during this stage allows teams to detect exploitable flaws before production release, significantly strengthening application security.
==========

NEW QUESTION # 41
(Rahul Mehta is working as a DevSecOps engineer in an IT company that develops cloud-native web applications. His organization follows a strict DevSecOps practice and wants to ensure that third-party open- source dependencies used in the application do not

introduce known security vulnerabilities. Rahul decided to integrate a Software Composition Analysis (SCA) tool into the CI pipeline so that every build is automatically scanned. During one of the builds, the SCA tool detects a critical vulnerability in a transitive dependency.
What should ideally happen in a mature DevSecOps pipeline when such a critical vulnerability is detected at build time?.)

- A. The pipeline should fail the build and prevent the artifact from progressing further.
- B. The pipeline should log the vulnerability details and continue the build to avoid delivery delays.
- C. The pipeline should ignore transitive dependencies and only scan direct dependencies.
- D. The pipeline should notify the security team and continue with deploy-time checks.

Answer: A

Explanation:
In a mature DevSecOps pipeline, security controls are enforced asgates, not merely as informational checks.
When an SCA tool detects acritical vulnerabilityin a dependency-whether direct or transitive-the correct response at the Build and Test stage is tofail the build. This prevents vulnerable artifacts from moving forward into later stages such as deployment or production, where remediation would be more expensive and risky. Allowing the build to continue, even with notifications, contradicts the shift-left security principle.
Ignoring transitive dependencies is also dangerous, as many real-world vulnerabilities originate from indirect libraries. Failing the build forces developers to remediate the issue immediately by upgrading, replacing, or mitigating the vulnerable dependency. This approach reduces attack surface, enforces accountability, and ensures that only secure artifacts are released. Therefore, stopping the pipeline upon detection of critical vulnerabilities reflects a strong DevSecOps maturity model and effective security governance.

NEW QUESTION # 42
.....

BTW, DOWNLOAD part of ExamsReviews 312-97 dumps from Cloud Storage: https://drive.google.com/open?id=1PFbo9FeK-VTxGpoxtaLAySyPjuA1NGkl