PassQueen
a.k.a dumpsqueen.com

WGUSecure Software Design (KEO1) Exam

WGU Secure-Software-Design

Version Demo

Total Demo Questions: 13

Total Premium Questions: 134

Buy Premium PDF

https://passqueen.com

support@passqueen.com

QUESTION NO: 1

What is the protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or distribution to provide confidentiality, integrity, and availability?

A. Availability

B. Integrity

C. Confidentiality

D. Information Security

ANSWER: D

Explanation:

Information Security is correct because it is the discipline concerned with protecting information and the systems that store, process, and transmit it from unauthorized access, use, disclosure, disruption, modification, or destruction. Its central objective is to preserve the confidentiality, integrity, and availability (CIA) of information. Confidentiality ensures information is accessible only to authorized people and processes; integrity ensures that information remains accurate and is changed only in authorized ways; and availability ensures authorized users can access information and systems when needed. The question's definition addresses all three CIA objectives and the broader set of threats to both information and information systems, which is the scope of information security rather than any one CIA objective alone. NIST defines information security in substantially these terms in its glossary and uses the CIA triad as a foundational security model. Secure software design applies information-security principles throughout requirements, architecture, implementation, testing, and operations so that software handles data and system resources safely. See the NIST definition of information security and NIST's FIPS 199 for the confidentiality, integrity, and availability security objectives.

QUESTION NO: 2

Which secure coding best practice says to assume all incoming data should be considered untrusted and should be validated to ensure the system only accepts valid data?

A. General coding practices

B. Input validation

C. Session management

D. System configuration

ANSWER: B

Explanation:

Input validation is the secure coding practice that requires an application to treat data entering a trust boundary as untrusted unless it has been verified. This includes values supplied through web forms, API requests, HTTP headers, cookies, uploaded files, command-line arguments, message queues, and external services. Before the application uses the data in processing, queries, file operations, or output, validation should confirm that it conforms to the expected allowlisted type, length, format, range, and permitted character set. For example, a field intended to contain a numeric account identifier should accept only an appropriately bounded numeric value, rather than arbitrary text.

Input validation is a foundational control because it reduces the likelihood that malformed or malicious data reaches sensitive application components. Validation is most dependable when enforced on the server side, close to the point where the application receives or consumes the data, since client-side checks can be altered or bypassed. OWASP recommends validating all untrusted input as early as possible and favoring allowlists over attempts to block known bad patterns. See the OWASP Input Validation Cheat Sheet and the MITRE CWE-20: Improper Input Validation for implementation guidance.

QUESTION NO: 3

Which type of security analysis is performed by injecting malformed data into open interfaces of an executable or running application and is most commonly executed during the testing or deployment phases of the SDLC?

A. Static Analysis

B. Fuzz Testing

C. Dynamic Analysis

D. Manual Code Review

ANSWER: B

Explanation:

Fuzz Testing is correct because it assesses a running executable or application by automatically supplying unexpected, malformed, random, or semi-random inputs through exposed interfaces such as APIs, file parsers, network services, and form fields. The objective is to observe whether the application crashes, hangs, exposes unintended behavior, or mishandles input in a way that could indicate a security weakness. Since fuzzing requires an executable target and benefits from runtime monitoring, it is typically performed during testing and can continue in deployment-oriented validation, such as pre-release security testing or testing a staging environment. Effective fuzzing records the inputs that trigger failures so developers can reproduce the problem, identify the underlying defect, and remediate it before release. This technique is particularly valuable for discovering input-validation flaws, memory-safety errors, and unhandled edge cases that may not be apparent through normal functional tests. The National Institute of Standards and Technology describes fuzz testing as submitting invalid, unexpected, or random data to an application to identify defects and vulnerabilities. See NIST's definition of fuzz testing and OWASP's guidance on fuzzing.

QUESTION NO: 4

During fuzz testing of the new product, random values were entered into input elements Search requests were sent to the correct API endpoint but many of them failed on execution due to type mismatches.

How should existing security controls be adjusted to prevent this in the future?

A. Ensure all user input data is validated before it is processed by the application

B. Ensure all requests and responses are encrypted

C. Ensure sensitive transactions can be traced through an audit log

D. Ensure the contents of authentication cookies are encrypted

ANSWER: A

Explanation:

Ensure all user input data is validated before it is processed by the application is the appropriate control because the fuzz test demonstrated that the API accepted requests whose values did not conform to the data types expected by the application. Input validation should be enforced at the server-side trust boundary, before the application uses request fields in searches, business logic, database queries, or other processing. The validation rules should explicitly define the expected type, format, length, range, and allowable values for every input field. For example, an identifier expected to be an integer should be parsed and accepted only as an integer within its permitted range; a search term intended to be text should be treated according to its defined schema. Requests that fail these checks should be rejected safely with a controlled error response rather than progressing until a type-related execution failure occurs. This makes application behavior predictable, reduces attack surface, and improves resilience against malformed or unexpected inputs generated by fuzzing. OWASP identifies allowlist-based input validation as a primary defense and recommends validating data as early as possible: OWASP Input Validation Cheat Sheet. NIST also describes validating inputs as part of secure software development practices: NIST SP 800-218.

QUESTION NO: 5

What is an advantage of using the Agile development methodology?

A. Customer satisfaction is improved through rapid and continuous delivery of useful software.

B. Each stage is clearly defined, making it easier to assign clear roles to teams and departments who feed into the project.

C. The overall plan fits very neatly into a Gantt chart so a project manager can easily view the project timeline.

D. There is much less predictability throughout the project regarding deliverables.

ANSWER: A

Explanation:

Customer satisfaction is improved through rapid and continuous delivery of useful software. This is a central advantage of Agile development because Agile teams work in short iterations and deliver small, usable increments of a product frequently. Rather than postponing value until the end of a long project, stakeholders can review working software early and regularly. Their feedback helps the team confirm that the product remains aligned to real customer needs, priorities, and changing business conditions. The Agile Manifesto explicitly values customer collaboration and states that its principles include early and continuous delivery of valuable software to satisfy customers. Frequent delivery also makes progress visible, supports timely refinement of requirements, and enables the organization to realize value sooner. This approach is reflected in the Scrum framework as well: each Sprint produces a usable, potentially releasable Increment that meets the team's Definition of Done. By continually inspecting delivered functionality and adapting the Product Backlog, the team can maintain focus on the capabilities that provide the greatest customer value. See the Agile Manifesto principles and the Scrum Guide for the foundational guidance on iterative delivery and customer value.

QUESTION NO: 6

The software security group is conducting a maturity assessment using the Open Web Application Security Project Software Assurance Maturity Model (OWASP OpenSAMM). They are currently focused on reviewing design artifacts to ensure they comply with organizational security standards.

Which OpenSAMM business function is being assessed?

A. Construction

B. Deployment

C. Verification

D. Governance

ANSWER: C

Explanation:

Verification is correct because reviewing design artifacts to determine whether they satisfy defined organizational security standards is a design-review assurance activity. In the OpenSAMM model version reflected by these business-function choices, Verification addresses the evaluation of security controls and development deliverables through activities such as design review, code review, and security testing. A design review occurs before or independently of implementation testing and provides evidence that the application's planned architecture, component interactions, trust boundaries, and security mechanisms align with the organization's security expectations. This enables the security group to identify and record design-level findings early, when remediation is generally less costly and more effective than correcting security issues after construction or release. The review is therefore assessing whether the design has been adequately verified against security criteria, rather than managing the program itself or performing release-oriented operational work. OWASP's SAMM project describes the model as a framework for measuring and improving software assurance practices across an organization; its model materials document verification-oriented assessment practices, including evaluation of software design and security controls. See the OWASP SAMM project and the OWASP SAMM model repository.

QUESTION NO: 7

The security team is reviewing all noncommercial software libraries used in the new product to ensure they are being used according to the legal specifications defined by the authors.

What activity of the Ship SDL phase is being performed?

A. Policy compliance analysis

B. Open-source licensing review

C. Penetration testing

D. Final security review

ANSWER: B

Explanation:

Open-source licensing review is the Ship-phase activity being performed. Noncommercial libraries commonly include open-source components, and their authors or maintainers publish license terms that define the conditions for use, modification, redistribution, attribution, disclosure of source code, patent rights, and notices. Before a product ships, the team must identify included third-party components, determine the license associated with each component and version, and verify that the product's planned distribution model satisfies those obligations. This review also produces the information needed for required notices, license files, and software-bill-of-materials or third-party-attribution materials delivered with the release.

This activity supports secure software delivery because component governance is part of release readiness: an organization needs to know not only whether dependencies are technically acceptable, but also whether it has the legal right to distribute them in the intended product. Microsoft's Security Development Lifecycle identifies release-stage practices intended to confirm that security and release requirements have been completed before shipment. The Open Source Initiative explains that open-source licenses set the rights and obligations governing use and distribution. See Microsoft Security Development Lifecycle practices and the Open Source Initiative license information.

QUESTION NO: 8

A development team corrects a reflected cross-site scripting vulnerability and reruns the same security tests that originally exposed the issue. The team also performs related regression tests to confirm that the correction did not introduce new weaknesses. Management wants evidence showing which tests were performed and how many times each was executed.

Which design and development deliverable should provide this evidence?

A. Remediation report

B. Security test execution report

C. Threat profile

D. Privacy compliance report

ANSWER: B

Explanation:

Security test execution report is correct because it records the security tests performed, their execution history, and re-evaluation activity after changes are made. The report provides auditable evidence that the original finding was retested and that related security checks were executed as part of regression testing.

A remediation report focuses on corrective actions for findings, while a security testing report commonly communicates findings and recommendations to stakeholders. The specific requirement to document evaluation types and repeated executions is the purpose of the security test execution report.

QUESTION NO: 9

Which mitigation technique can be used to light against a threat where a user may gain access to administrator level functionality?

A. Encryption

B. Quality of service

C. Hashes

D. Run with least privilege

ANSWER: D

Explanation:

Run with least privilege is the appropriate mitigation because it limits each user account, process, and service to only the permissions required for its intended task. Administrative capabilities are granted only when specifically needed, rather than being available through ordinary user execution. This reduces the opportunity for a user or compromised application to access privileged functions and limits the impact if an account is misused or an attacker attempts privilege escalation. Least privilege should be implemented through role-based authorization, separate standard and administrative accounts, restricted access-control lists, and narrowly scoped service identities. Administrative actions can also require explicit elevation and additional verification, ensuring that higher-risk functionality is not routinely available to standard users. This is a core secure-design control: the system should deny access by default and grant elevated permissions only to authorized subjects in defined circumstances. NIST identifies least privilege as requiring that users receive only the authorizations necessary to accomplish assigned tasks, and OWASP recommends minimizing privileges to reduce attack impact. See NIST's definition of least privilege and the OWASP Authorization Cheat Sheet.

QUESTION NO: 10

A company's product security group maintains a catalog of common attack patterns observed across prior projects. Before each new project begins design, the group provides relevant attack patterns and security guidance to the development team so that the team can identify likely threats early.

Which BSIMM domain is most directly demonstrated by this activity?

A. Governance

B. Software security development life cycle (SSDL) touchpoints

C. Intelligence

D. Deployment

ANSWER: C

Explanation:

Intelligence is correct because the product security group is collecting, maintaining, and applying knowledge about attacks across development efforts. Reusable attack patterns help teams recognize likely adversary behavior and incorporate appropriate protections during design.

Governance addresses program management, policy, and measurement. Software security development life cycle touchpoints address direct security activities performed during project work, such as reviews and testing. Deployment focuses on security activities associated with released software and its operational environment.

QUESTION NO: 11

A recent vulnerability scan uncovered an XML external entity (XXE) Haw that could allow attackers to return the contents of a system file by including a specific payload in an XML request.

How should the organization remediate this vulnerability?

A. Ensure audit trails exist for all sensitive transactions

B. Disable resolution of external entities in the parsing library

C. Enforce role-based authorization in all application layers

D. Ensure authentication cookies are encrypted

ANSWER: B

Explanation:

Disable resolution of external entities in the parsing library is the appropriate remediation because XXE occurs when an XML parser processes externally defined entities supplied in untrusted XML. An attacker can define an entity that references a local file or a remote resource, then cause the application to include that resource's contents in its response or send requests to internal systems. Disabling external general entities and external parameter entities prevents the parser from dereferencing attacker-controlled entity declarations, eliminating the behavior that enables file disclosure in this scenario.

The change must be applied to every XML parser, framework, and processing path that accepts untrusted XML, including libraries invoked indirectly by application components. Secure parser configuration should prohibit DTD processing where it is not required and ensure external entity loading is disabled. The organization should then test with representative XXE payloads and retain the setting in application configuration and deployment standards so that upgrades or new services do not re-enable unsafe defaults. OWASP identifies disabling external entity processing as the primary defense for XXE and provides parser-specific guidance. See the OWASP XXE Prevention guidance and CWE-611: Improper Restriction of XML External Entity Reference.

QUESTION NO: 12

While performing functional testing of the new product from a shared machine, a QA analyst closed their browser window but did not logout of the application. A different QA analyst accessed the application an hour later and was not prompted to login. They then noticed the previous analyst was still logged into the application.

How should existing security controls be adjusted to prevent this in the future?

A. Ensure no sensitive information is stored in plain text in cookies

B. Ensure user sessions timeout after short intervals

C. Ensure role-based access control is enforced for access to all resources

D. Ensure strong password policies are enforced

ANSWER: B

Explanation:

Ensure user sessions timeout after short intervals is the appropriate adjustment because the incident is caused by a valid authenticated session remaining usable on a shared device after the original user stops actively using the application. Session-management controls must limit the lifetime of an authenticated session so that an abandoned browser session cannot provide continuing access to the next person using the machine. An inactivity timeout should invalidate the server-side session and require fresh authentication after a defined period without activity. The timeout duration should be selected based on the application's risk level, the sensitivity of accessible data, and the shared-device context; shared QA workstations generally warrant a relatively short inactivity period. The application should enforce expiration on the server, rather than relying only on browser behavior, because closing a window does not necessarily remove persistent authentication cookies or invalidate an active server session. A secure implementation also clears or expires the associated session cookie when the session ends and requires reauthentication before protected functions are available again. OWASP identifies inadequate session expiration as a session-management weakness and recommends idle and absolute session timeouts. See the OWASP Session Management Cheat Sheet and OWASP Session Timeout guidance.

QUESTION NO: 13

While performing functional testing of the ordering feature in the new product, a tester noticed that the order object was transmitted to the POST endpoint of the API as a human-readable JSON object.

How should existing security controls be adjusted to prevent this in the future?

A. Ensure passwords and private information are not logged

B. Ensure sensitive transactions can be traced through an audit log

C. Ensure the contents of authentication cookies are encrypted

D. Ensure all requests and responses are encrypted

ANSWER: D

Explanation:

Ensure all requests and responses are encrypted. JSON is deliberately human-readable at the application layer, so its use in an API request is not inherently a defect. The security requirement is that the HTTP message, including its URL, headers, and JSON body, is protected while it travels between the client and API endpoint. Enforcing HTTPS with current Transport Layer Security (TLS) encrypts data in transit, preventing a network observer or an attacker positioned on the communication path from reading the order object. TLS also supplies integrity protection, helping detect modification of requests or responses in transit, and authenticates the server through its certificate. The control should therefore require encrypted transport for every API route and response, redirect or reject unencrypted HTTP traffic, use valid certificates and modern TLS configurations, and avoid sending sensitive API data over plaintext channels. This is consistent with OWASP guidance to encrypt sensitive data in transit and use TLS for web applications and APIs. See the OWASP Transport Layer Security Cheat Sheet and OWASP API Security Project.