CERTLABZ CERTIFICATION BOARD

Certified Cryptography Specialist (CCYS)

Ten Sample Questions

These ten questions are written at the difficulty of the live exam. Six carry a figure, a data table, a configuration or a code listing; four are plain scenario questions, as on the live paper. Every one is a scenario a working cryptography engineer would meet, and none of them appears in the live question bank.

The live exam is multistage adaptive: how you score on one stage selects the difficulty of the next, so no two candidates sit the same paper. It is a performance-based exam: each form can carry up to 5 to 12 performance-based questions alongside the multiple-choice questions, which means you build, sequence or complete something rather than only choosing from a list. Answers and full explanations follow each question here; the live exam does not disclose item-level answers.

Question 1

Domain 2: Symmetric Cryptography and Modes

CODE

Emma Wilson reviews the routine a service uses to encrypt records before storing them.

```
KEY   = sha1(config.password)          # 20 bytes, truncated to 16
NONCE = b"\x00" * 12                   # same for every record

def encrypt(record):
    return aes_gcm_encrypt(KEY, NONCE, record)
```

Listing 1.1 The encryption routine as written.

Which finding is the most severe?

A. The key is derived with SHA-1, so the routine is broken by SHA-1 collisions and must switch to SHA-256 to be safe.

B. The fixed nonce is reused for every record under one key, which lets an attacker recover plaintexts and forge records.

C. The key is truncated to 16 bytes, so the routine only gives 128-bit security when 256-bit AES should always be used.

D. The routine returns the ciphertext without Base64 encoding, so the stored records cannot be reliably read back later.

Correct answer: B
AES-GCM is a stream construction: reusing a nonce under the same key leaks the XOR of plaintexts and allows recovery of the authentication key, so forgeries become possible. That is catastrophic. Deriving a key from a password with a single fast hash is also wrong (use a slow, salted KDF), but SHA-1 collisions are not the issue, 128-bit AES is adequate, and encoding is irrelevant to security.

Question 2 — Domain 1: Cryptographic Foundations — DATA

James Carter is asked which primitive to use for four requirements in a new service.

Requirement	Proposed primitive
Prove to a regulator that a report came from the company	HMAC-SHA256 with a key shared with the regulator
Detect accidental corruption of a download	SHA-256 digest published alongside the file
Hide the content of a stored document	AES-256-GCM with a random nonce
Store user passwords	Argon2id with a per-user salt

Table 2.1 Requirements and the primitive proposed for each.

Which proposal does not meet its requirement?

A. The report proposal, because a shared-key MAC cannot prove origin to a third party; a digital signature is required.

B. The download proposal, because a published SHA-256 digest is far too weak and a keyed MAC must replace it there.

C. The document proposal, because AES-256-GCM provides only integrity and a separate cipher is needed for confidentiality.

D. The password proposal, because Argon2id is a slow function and a fast hash such as SHA-256 is the right tool for logins.

Correct answer: A
Non-repudiation needs a digital signature: only the company holds the private key, and the regulator verifies with the public key. With an HMAC, the regulator holds the same key and could have produced the tag, so it proves nothing to a third party. The other three proposals are correct uses of a hash, an AEAD cipher and a slow password hash.

Question 3 — Domain 5: Protocols and Applied Cryptography — CONFIG

Olivia Davis reviews the TLS configuration of a public web server.

```
ssl_protocols       TLSv1 TLSv1.1 TLSv1.2 TLSv1.3;
ssl_ciphers         "AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:RC4-SHA";
ssl_certificate     /etc/ssl/site.crt;      # RSA 2048, SHA-256, expires in 30 days
ssl_stapling        off;
```

Listing 3.1 The server's TLS settings.

Which change matters most for the security of connections?

A. Turn OCSP stapling on, because without it every client must query the CA and revoked certificates go undetected.

B. Replace the RSA certificate with an ECDSA one, because a 2048-bit RSA key no longer meets the minimum strength.

C. Remove TLS 1.0, TLS 1.1 and the RC4 and static AES128-SHA suites, keeping only forward-secret AEAD suites.

D. Renew the certificate now, because a certificate that expires within thirty days is treated as invalid by browsers.

Correct answer: C
Offering deprecated protocol versions and a broken cipher (RC4) plus a static RSA suite without forward secrecy means a downgrade or a future key leak exposes traffic. Removing them so only TLS 1.2 and 1.3 with ECDHE and AEAD suites remain is the change that matters most. Stapling is worthwhile, RSA-2048 is still acceptable, and a certificate is valid until it expires.

Question 4

Domain 4: PKI, Certificates and Trust

DATA

Liam Anderson debugs a client that rejects a server certificate. The chain the server presents is below.

Certificate	Subject	Issuer	Notes
Leaf	CN=api.example.com, SAN: api.example.com	Example Issuing CA 2	Valid dates, not revoked
Intermediate	Example Issuing CA 2	Example Root CA	Valid dates
Root	Example Root CA	Example Root CA	Not in the client trust store

Table 4.1 The presented chain.

The client connects to api.example.com. Why does validation fail, and what is the correct fix?

A. The hostname does not match, because the Common Name is used for matching and the SAN is ignored; add a wildcard.

B. The chain has no trust anchor on the client, because the root is not trusted; install the root in the client store.

C. The intermediate is missing a SAN entry, so the chain cannot be built; reissue the intermediate with the hostname.

D. The root is self-signed, which modern clients refuse; have the root signed by a public CA and reissue the chain.

Correct answer: B
Chain building stops at a trust anchor the client already trusts. Everything else here is fine: the SAN matches the hostname, dates are valid and nothing is revoked. Because the root is absent from the client's trust store, validation fails; installing (or distributing) the root as a trusted anchor fixes it. Roots are always self-signed, and intermediates do not carry hostnames.

Question 5 — Domain 6: Key Management, Governance and Crypto Agility — DATA

Ava Thompson audits how four keys are held.

Key	Current custody
Root CA private key	Offline HSM, three of five custodians, witnessed ceremony
Code-signing key	Exported to a CI server as a plain environment variable
Database data keys	Envelope-encrypted under a KMS master key, rotated yearly
TLS server key	Generated on the host, permissions restricted, renewed by ACME

Table 5.1 Keys and how they are held.

Which finding needs immediate action?

A. The root CA key, because an offline HSM with several custodians makes routine issuance far too slow to operate.

B. The database data keys, because envelope encryption under a KMS master key means the data keys are never rotated.

C. The TLS server key, because a key generated on the host and renewed by ACME is never seen by the security team.

D. The code-signing key, because a plain environment variable on a CI server exposes it to anyone who can run a job.

Correct answer: D
A code-signing key in a plain environment variable can be read by any job, plugin or compromised runner, and a stolen signing key lets an attacker ship malware as the company. It belongs in an HSM or KMS with an approval workflow and audit log. The root custody is exemplary, envelope encryption with rotation is correct, and on-host generation with automated renewal is the right pattern for TLS keys.

Question 6 — Domain 5: Protocols and Applied Cryptography — CODE

A payments partner signs webhooks. The receiver's verification code is shown.

```
def verify(request):
    expected = hmac_sha256(SECRET, request.body)
    if request.headers["X-Signature"] == expected:   # plain string compare
        return True
    return False
# no timestamp is checked
```

Listing 6.1 Webhook verification as implemented.

Which two weaknesses are present?

A. A timing side channel from the ordinary comparison, and replay of a captured valid webhook because no timestamp is checked.

B Use of HMAC instead of a digital signature, and use of SHA-256 where the partner should have chosen SHA-512 for the tag.

C The secret is shared with the partner, which is never acceptable, and the body should be encrypted before it is signed.

D The header name is not standard, and the function should return the computed tag so the caller can compare it itself.

Correct answer: A
An early-exit string comparison leaks how many leading bytes matched, which can be exploited to forge a tag byte by byte, so a constant-time comparison is required. Without a timestamp (or nonce) inside the signed data, a captured valid request can be replayed indefinitely. HMAC with a shared secret is the standard webhook design, SHA-256 is adequate, and encryption is a separate concern.

Question 7

Domain 3: Asymmetric Cryptography and Key Exchange

A firmware update system signs images with ECDSA on the P-256 curve. A reviewer finds that the per-signature nonce is generated from the build timestamp. What is the consequence?

A None, because the nonce in ECDSA only needs to be unique and a build timestamp is unique for every image that is built.

B Signatures become invalid, because ECDSA nonces must be exactly 256 random bits or the verification equation fails.

C The private key can be recovered, because a predictable or repeated nonce reveals it; use RFC 6979 or Ed25519 instead.

D The images become larger, because a timestamp-based nonce adds a second field to every signature that is produced.

Correct answer: C
ECDSA leaks the private key if the per-signature nonce is predictable or repeated: two signatures with the same nonce give the key by simple algebra, and a guessable nonce lets an attacker solve for it directly. This is how the PlayStation 3 signing key was extracted. Deterministic nonces (RFC 6979) or a scheme such as Ed25519 remove the dependence on nonce randomness.

Question 8

Domain 3: Asymmetric Cryptography and Key Exchange

A government contractor must protect archived communications that will stay sensitive for twenty years. Today the VPN uses ECDHE with X25519. Which change is appropriate now?

A No change, because X25519 offers 128-bit security and no attacker can break it within the next twenty years at all.

B Adopt a hybrid key exchange that combines X25519 with ML-KEM, so recorded traffic stays safe if a quantum computer arrives.

C Replace X25519 with RSA-4096, because a larger RSA key resists quantum attack for far longer than any elliptic curve.

D Replace the VPN with AES-256 pre-shared keys alone, because symmetric keys need no exchange and cannot be intercepted.

Correct answer: B
Traffic recorded today can be decrypted later once a cryptographically relevant quantum computer exists, and both X25519 and RSA of any size fall to Shor's algorithm. A hybrid exchange keeps classical security while adding the post-quantum ML-KEM (FIPS 203). Pre-shared keys alone do not scale and lose forward secrecy.

Question 9 Domain 4: PKI, Certificates and Trust

An internal CA's root certificate is installed as trusted on every employee laptop so that internal sites work. The CA has no name constraints. What is the risk, and what reduces it?

A. There is no risk, because an internal CA can only sign internal names and public browsers ignore any other certificate.

B. The risk is slow browsing, because each laptop must contact the internal CA for every site; caching the root fixes it.

C. If the CA is compromised it can issue trusted certificates for any public site and intercept traffic; constrain it to internal names.

D. The risk is certificate expiry, because internal roots are short-lived; renewing the root every ninety days removes it.

Correct answer: C
A trusted root with no name constraints can issue a valid certificate for any domain, so a compromised internal CA becomes a tool for intercepting public sites from every laptop that trusts it. Adding name constraints to the internal namespace and protecting the CA keys in an HSM reduce the blast radius. Browsers trust whatever the root signs; they do not know which names are internal.

Question 10 Domain 6: Key Management, Governance and Crypto Agility

An audit finds 3DES still used by a payment gateway, SHA-1 in an internal signing tool, and RSA-1024 in a legacy VPN. Which is the correct programme response?

A. Replace all three this week regardless of impact, because any deprecated algorithm must be removed before the next audit.

B. Leave all three in place, because each is internal or legacy and deprecated algorithms are only a concern on public systems.

C. Inventory every use, prioritise by exposure and data sensitivity, set deprecation dates and migrate each to a current standard.

D. Increase the key size of each algorithm in place, so that 3DES, SHA-1 and RSA remain acceptable for several more years.

Correct answer: C
Crypto agility is a governed process: build the inventory, rank by exposure and consequence, set dates and replacement standards, migrate, then verify nothing still uses the weak algorithm. Blind immediate replacement causes outages; leaving weak crypto in place accepts real risk; and neither 3DES nor SHA-1 can be rescued by longer keys.

Published by the CertLabz Certification Board, a division of CertLabz LLC. Companion to objectives document CCYS-001. https://certlabz.com/certifications/certified-cryptography-specialist.html