CYBERQUIZZER CHEAT SHEET

Cryptography Algorithms

Three families, three jobs. **Symmetric** encrypts the bulk data, **asymmetric** exchanges the key and signs, **hashing** proves nothing changed. ● marks what you must recognise as broken.

Scan to drill this sheet

Symmetric

One shared key · fast · bulk data · key distribution is the problem

AES	**128 / 192 / 256-bit** · block · the standard. TLS, disk, Wi-Fi (WPA2/3)
ChaCha20	**256-bit** · stream · AES's peer on mobile; TLS 1.3, WireGuard
3DES	**168-bit (112 effective)** · block · deprecated 2023; legacy payments
Blowfish	**32–448-bit** · block · 64-bit block is its weakness; bcrypt is built on it
Twofish	**128–256-bit** · block · AES finalist; VeraCrypt option
DES ●	**56-bit** · block · brute-forced in hours. Never acceptable
RC4 ●	**40–2048-bit** · stream · biased keystream; why WEP fell

Asymmetric

Public + private pair · slow · key exchange, signatures, identity

RSA	**2048–4096-bit** · factoring · encryption *and* signatures; certificates, SSH
ECC	**256–521-bit** · elliptic curves · same strength as RSA at ~1/10 the key size; mobile, IoT
ECDH	**256-bit** (DH: 2048+) · key *agreement* only — no encryption, no signing. Ephemeral (DHE) gives forward secrecy
ECDSA	**256-bit** (DSA: 2048) · signatures only. Signs Bitcoin and most TLS certs
EdDSA	**Ed25519** · signatures · modern default for SSH keys
ElGamal	Discrete log · used inside PGP/GPG
ML-KEM	**Post-quantum** (Kyber) · NIST FIPS 203, 2024 · replacing DH for key exchange

Hashing

No key · one-way · fixed length · integrity, not confidentiality

SHA-2	**224 / 256 / 384 / 512-bit** digest · the workhorse. SHA-256 in TLS, code signing, Bitcoin
SHA-3	**224–512-bit** · Keccak sponge · different design, not a SHA-2 fix
HMAC	Hash *plus* a secret key → integrity **and** authenticity. HMAC-SHA256 in JWTs, APIs
bcrypt	Also Argon2, PBKDF2. Deliberately **slow**, salted · for passwords only. Argon2id is current best
SHA-1 ●	**160-bit** · collision found 2017 (SHAttered). Certs rejected
MD5 ●	**128-bit** · collisions in seconds. Checksums only, never security

One TLS connection, all three families

No real system uses one family alone — the exam wants you to know which does which job.

1 · IDENTITY
Server proves who it is with a certificate signed by **RSA or ECDSA**
Asymmetric — signature

2 · KEY EXCHANGE
Both sides derive a shared secret with **ECDHE**, never sending it
Asymmetric — agreement

3 · BULK DATA
Every byte of the page is encrypted with **AES-GCM or ChaCha20**
Symmetric — fast

4 · INTEGRITY
Each record is tagged with **HMAC-SHA256** or the GCM auth tag
Hashing — tamper-evident

Where people lose the mark

Three distinctions that decide most crypto questions.

Encrypt with which key?

For **confidentiality**: encrypt with the recipient's *public* key. For a **signature**: sign with your own *private* key. Swap them and the answer is wrong.

Hashing is not encryption

A hash has **no key and cannot be reversed**. If the question needs the data back, hashing is never the answer. Stored passwords are hashed, not encrypted.

Key size ≠ strength across families

AES-128 ≈ RSA-3072 ≈ ECC-256. A 256-bit ECC key is **not weaker** than 2048-bit RSA — it is stronger. Compare within a family only.