A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

17.1 Encryption, Encryption Protocols and Digital Certificates

Name: _____ Class: _____ Date: _____ **Total: 35 marks**

KEY WORDS digital certificate 数字证书 • encryption 加密 • digital signature 数字签名 • symmetric encryption 对称加密 • asymmetric encryption 非对称加密

Objective

Build the skills to answer exam questions on **encryption** 加密 — symmetric and asymmetric keys, TLS, and digital certificates.

You must be able to:

- explain how **encryption** works and compare **symmetric** and **asymmetric**
- describe the **hybrid** approach and outline **TLS**
- explain a **digital certificate** and how it is checked

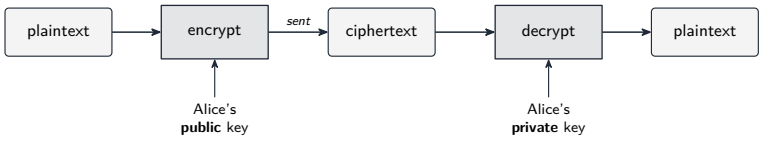
1 Worked examples

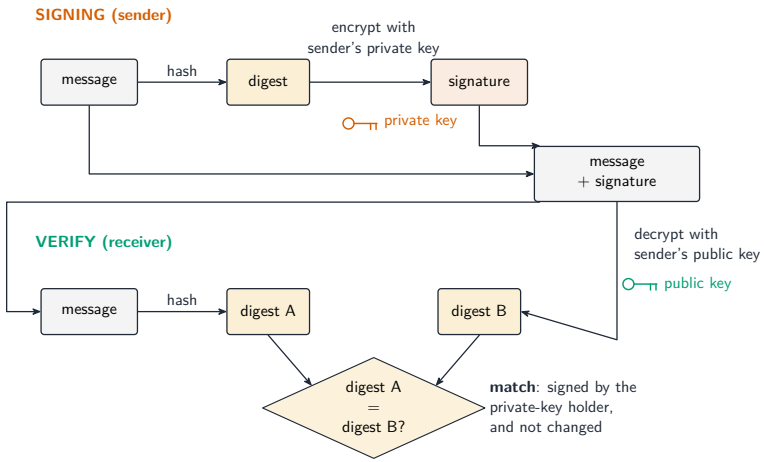
Study these first. Each one shows the method for a question type used later — follow the steps and you can do the Practice and Exam-style questions yourself.

Encryption basics

Encryption turns readable **plaintext** into unreadable **ciphertext** using a **key**; only the right key reverses it (**decryption**).

- **Symmetric** — the **same** key encrypts and decrypts. Fast (good for bulk data), but how do you **share the key** safely?
- **Asymmetric (public-key)** — a **pair** of keys. Encrypt with the recipient's **public** key; only their **private** key can decrypt:





A digital signature authenticates a message

Hybrid (how HTTPS really works)

Asymmetric is **slow**, so it is used only to exchange a fresh **session key**, then fast **symmetric** encryption carries the data. A **TLS** handshake: the server sends its **digital certificate** (with its public key); the client checks it; both agree a **session key**; all later traffic is symmetric.

Digital certificate

A **digital certificate** binds an **identity** to a **public key**, signed by a trusted **Certificate Authority (CA)**. The browser checks the **expiry**, that the **name matches**, and that it is **signed by a trusted CA**.

Digital signature

A **digital signature** 数字签名 proves a message came from the real sender (**authenticity**) and was not changed (**integrity**):

1. the sender **hashes** the message to a fixed-length **digest**;
2. they **encrypt the digest with their *private* key** — this is the signature, sent with the message;
3. the receiver decrypts the signature with the sender's **public key** to recover the digest, and **re-hashes** the received message;
4. if the two digests **match**, the message is genuine and unchanged; if not, it was altered.

2 Practice

Now apply the methods above.

2.1 Define **plaintext** and **ciphertext**. [2]

2.2 State the key difference between **symmetric** and **asymmetric** encryption. [1]

2.3 State the main **problem** with symmetric encryption. [1]

2.4 State what a **digital certificate** binds together. [1]

3 Exam-style questions

3.1 Bob wants to send Alice a secret message using asymmetric encryption. Which of Alice's keys does Bob use to **encrypt**, and which does Alice use to **decrypt**? Explain why this is secure. [3]

3.2 Explain why real systems (like HTTPS) use a **hybrid** approach combining asymmetric and symmetric encryption. [3]

3.3 State **two** things that **TLS** provides. [2]

3.4 State **one** check a browser makes on a **digital certificate**. [1]

3.5 Explain how a **digital signature** lets the receiver check that a message has **not been changed** during transmission. [3]

3.6 An online shop uses **asymmetric** encryption and digital certificates to protect its customers.

(a) **Describe** how a pair of asymmetric keys is used so that a customer can send card details that only the shop can read. [3]

(b) **Explain** why the shop must never publish its **private** key, and what it does publish instead. [2]

(c) The shop obtains a **digital certificate** from a certificate authority. **Describe** what the certificate contains and how it proves the shop's identity. [4]

(d) **Describe** the purpose of the **SSL/TLS** protocol, and **state** where it sits relative to the other protocols in use. [3]

(e) **Outline** the steps of the TLS handshake that take place before any card details are sent. [4]

(f) **Explain** why the session then switches to **symmetric** encryption. [2]

Solutions

Each answer shows the steps, then the **result**.

2.1

- **plaintext** — the original readable data
- **ciphertext** — the scrambled, unreadable form after encryption

2.2

- **symmetric** uses the **same** key for both; **asymmetric** uses a **pair** (public to encrypt, private to decrypt)

2.3

- **key distribution** — sharing the secret key safely with the other party

2.4

- an **identity** (domain/organisation) and its **public key** (signed by a CA)

3.1

- Bob encrypts with **Alice's public key**
- only **Alice's private key** can decrypt it
- since Alice keeps her private key secret, an interceptor with only the public key and ciphertext cannot read the message

3.2

- asymmetric is **secure for key exchange** but **slow**
- symmetric is **fast** but needs a shared key
- so asymmetric is used once to share a **session key**, then fast symmetric encryption carries the bulk data

3.3

- any two: **encryption** of data in transit; **authentication** of the server (via certificate); **integrity** (detecting tampering)

3.4

- any one: the certificate is **in date**; the **subject name matches** the site's URL; it is **signed by a trusted CA** / chains to a trusted root

3.5

- the sender hashes the message and encrypts that hash with their **private key** to form the signature
- the receiver decrypts the signature with the sender's **public key** to get the original hash, and **re-hashes** the received message
- if the two hashes **match** the message is unchanged, and if they differ it was altered in transit

3.6

(a) the shop publishes its **public** key and keeps its **private** key secret

- the customer's browser encrypts the card details with the shop's **public** key
- only the matching **private** key can decrypt them, so only the shop can read the message

(b) the private key is the only thing that can decrypt messages sent to the shop and the only thing that can create its signature

- publishing it would let anyone read customers' data and impersonate the shop; the shop publishes the **public** key instead, inside its certificate

(c) the certificate contains the shop's **public key**, its domain name and organisation details, the CA's name, and the validity dates

- the whole lot is **signed** by the CA using the CA's own private key
- a browser that trusts that CA can verify the signature with the CA's public key, so it knows the public key really belongs to that shop

(d) SSL/TLS provides a **secure, encrypted channel** between the browser and the server

- giving confidentiality, integrity and authentication of the server
- it sits **between** the application layer protocol, such as HTTP, and the transport layer — which is why the secure version is called HTTPS

(e) the browser sends a "hello" listing the TLS versions and cipher suites it supports

- the server replies with its chosen cipher suite and its **digital certificate**
- the browser verifies the certificate against a trusted CA
- the two then agree a **session key**, which the browser sends encrypted with the server's public key

(f) symmetric encryption is far **faster** and less demanding on the processor for large amounts of data

- asymmetric encryption is used only to exchange the session key safely, after which the bulk of the traffic uses that shared key