Security, privacy and data integrity

A-Level Computer Science

Security, privacy and integrity — three different ideas

These sound alike but mean different things:

- **security** 安全 — protecting data from **unauthorised** 未授权 access, change or destruction.
- **privacy** 隐私 — an individual's right to **control who sees their personal data**, with consent and a clear purpose.
- **integrity** 完整性 — the data being **accurate and complete** — not corrupted or accidentally changed.

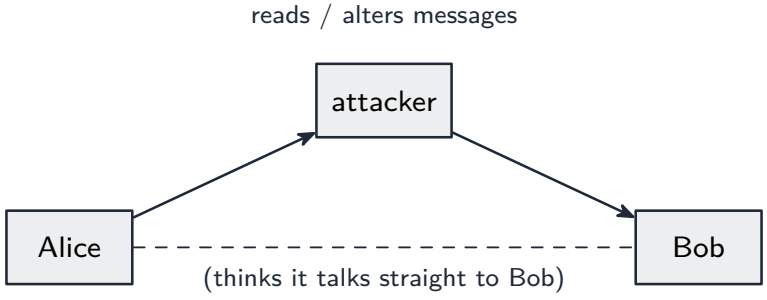
A file can be secure (only the right people can open it) but lack integrity (a typo corrupted it); or accurate but not private (anyone can read it). All three are needed.

Why security matters

Two things to protect: the **data** itself (keep it confidential, intact and available) and the **computer system** (a compromised system can attack others, steal credentials, or be held to ransom).

Threats from networks and the internet

Threats fall into three groups.



A man-in-the-middle attacker sits between the two parties

1. **Malware** 恶意软件 (malicious software) — harmful programs:

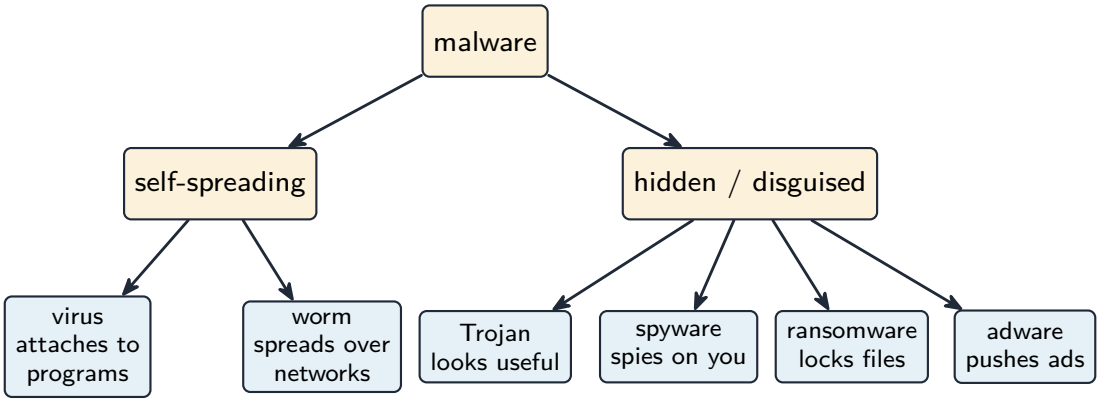
- **virus** 病毒 — self-copying code that attaches to other programs and spreads when they run.
- **worm** 蠕虫 — self-copying code that spreads over **networks** 网络 with no user action.
- **Trojan horse** 木马 — looks useful but hides malicious code.
- **spyware** 间谍软件 — secretly collects information (keystrokes, passwords).
- **ransomware** 勒索软件 — encrypts your files and demands payment.
- **adware** 广告软件 — pushes unwanted adverts.

2. **Tricking people** (social attacks):

- **phishing** 网络钓鱼 — fake emails/sites that trick users into giving credentials.
- **pharming** 域名欺骗 — redirects a user to a fake site even when they type the correct address.
- **social engineering** 社会工程 — tricking people into giving up information.

3. **Attacks on the network:**

- **hacking** 黑客入侵 by **hackers** 黑客 — unauthorised access, often via weak passwords or software flaws.
- **denial of service** 拒绝服务 (DoS/DDoS) — floods a server so real users cannot reach it.
- **eavesdropping** 窃听 — capturing data in transit (a risk on open Wi-Fi).
- **man-in-the-middle** 中间人攻击 — an attacker secretly relays or alters messages between two parties.



Malware by behaviour: self-spreading (virus, worm) versus hidden/disguised (Trojan, spyware, ransomware, adware)

Security measures

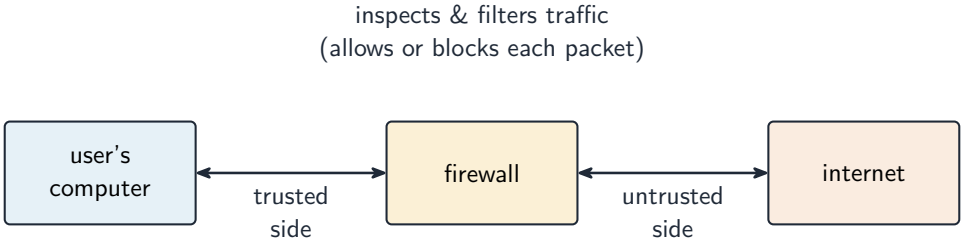
Measures protect both the **security of data** (against loss, theft or corruption) and the **security of the computer system** (its hardware, software and network).

A standalone PC

- a strong password; **antivirus** kept up to date; prompt **software updates**; **backup** 备份 to separate media; full-disk **encryption** 加密; a locked screen.

A networked PC

All the above, plus a **firewall** 防火墙, per-user **permissions** (admin rights only for admins), central management of **user accounts** 用户账户, and **audit logs** 审计日志 (who logged in, what they touched).



A firewall sits between the user's computer and the internet

Across the internet

- **VPN** 虚拟专用网 — encrypts traffic between the user and the corporate gateway.
- **HTTPS / TLS** — encrypt web traffic.
- **digital signatures** 数字签名 — prove who sent a message and that it was not altered in transit.
- intrusion detection — watches traffic for known attack patterns.

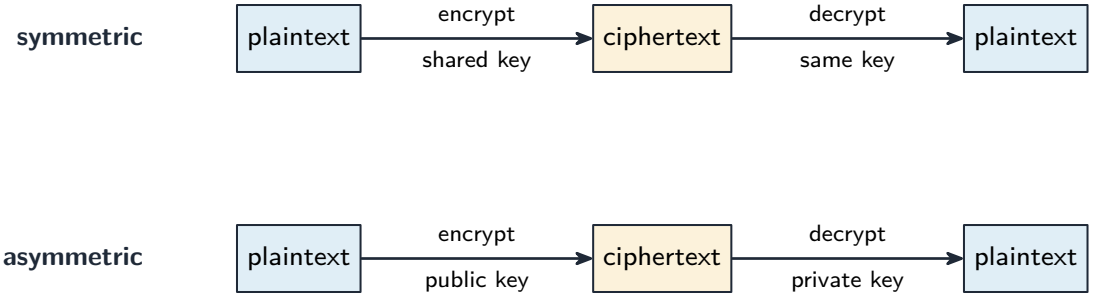
Matching measures to threats

- **interception in transit** → **encrypt** the data (HTTPS, VPN). Intercepted ciphertext is useless without the key.
- **unauthorised access** → strong **authentication** 身份验证 (long passwords; **two-factor authentication** 双因素认证 with a phone code or key); user **authorisation** 授权; lock-out after failed logins.

- **malware** → **anti-virus software** and **anti-spyware** 反间谍软件 with real-time scanning; patching; avoid untrusted downloads.
- **phishing** → user training; email filtering; check the URL before entering credentials.
- **internal threats** → the **least-privilege** 最小权限 principle (give each user only what they need); auditing.
- **DDoS** → rate limiting and traffic filtering.

Protecting the data itself

- **encryption** — turn **plaintext** 明文 into **ciphertext** 密文 with a key. **Symmetric encryption** 对称加密 (AES) uses one shared key; **asymmetric encryption** 非对称加密 (RSA) uses a **public key** 公钥 and a **private key** 私钥. Protects data at rest and in transit.
- **access control** 访问控制 — file permissions (read/write/execute) and **access rights** 访问权限, enforced by the OS.
- **authentication** — **authentication techniques** verify the user: something you know (password), have (token, phone), or are (**biometrics** 生物识别 — fingerprint, face, iris); strongest combined.
- **backups** — keep copies (some **off-site**) so loss or corruption is recoverable.
- **physical security** — locked server rooms, cable locks.



Symmetric uses one shared key; asymmetric uses a public key to encrypt and a private key to decrypt

A security token shows a changing code for two-factor authentication ("something you have")

A fingerprint reader checks "something you are" — a feature of the person, not a password

Data integrity

Data has integrity when it is accurate and complete. Two techniques: **data validation** (catch bad data before storing) and **data verification** (confirm data was entered or transferred correctly).

Validation — does the data make sense?

Validation 验证 checks data against sensible rules, automatically:

- **range check** — within limits (a month is 1–12).
- **limit check** — on the correct side of a single limit (e.g. age $\geq$ 18).
- **existence check** — the referenced item exists (e.g. a product code is in the table).
- **length check** — the right number of characters.
- **type / character check** — the right kind of data (a phone field allows only digits).
- **format check** — matches a pattern (an email must contain `@`).
- **presence check** — required fields are not empty.
- **check digit** 校验位 — an extra digit computed from the others (ISBN, card numbers) that spots transcription errors.
- **lookup check** and **consistency check** (e.g. delivery date $\geq$ order date).

Validation catches data that is wrongly **formatted**, but not data that is the right format yet factually wrong ("Bob" for "Bib").

Verification — was the data entered or transferred correctly?

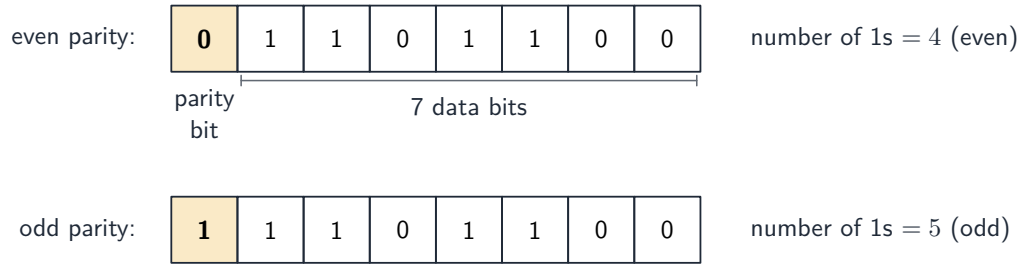
Verification 核对 checks the data was not changed in moving from one place to another.

During entry: **double entry** (type it twice and compare, as for a new password) or **visual** check.

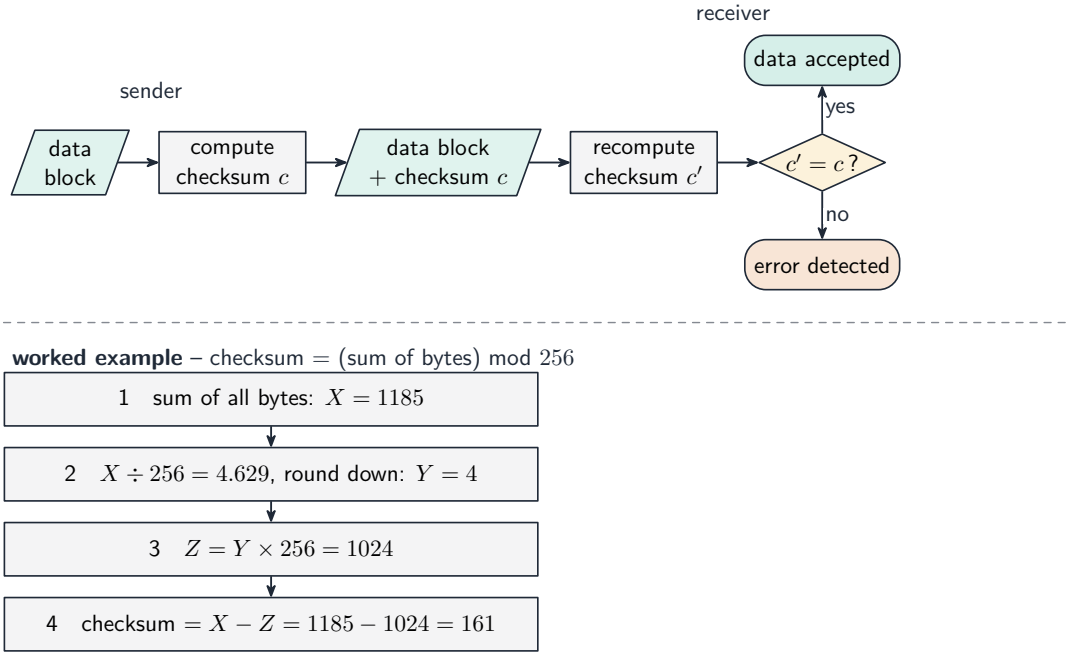
During transfer (bits can flip):

- **parity check** 奇偶校验 — an extra bit makes the number of 1s even (even parity) or odd. The receiver re-counts. Catches single-bit errors.
- **checksum** 校验和 — the sender sends a summary value of the data; the receiver recomputes it and compares.
- **cyclic redundancy check** 循环冗余校验 (CRC) — a stronger checksum using polynomial division, catching many more error types.

A **parity block check** 奇偶块校验 goes further and *locates* the error. Arrange the bytes in a grid: give each byte a **row** parity bit, then compute one extra **parity byte** whose bits are the **column** parity of the bytes above. A single flipped bit now fails **one row and one column** – their intersection pinpoints exactly which bit changed, so it can even be corrected.



The parity bit is set to make the number of 1s even or odd



worked example – checksum = (sum of bytes) mod 256

1 sum of all bytes: $X = 1185$

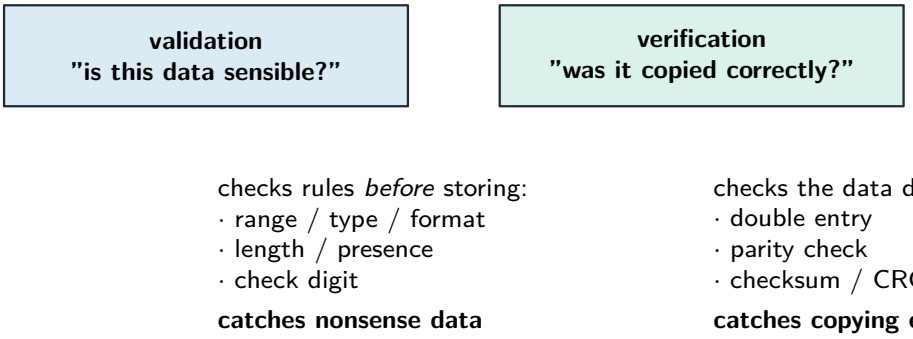
2 $X \div 256 = 4.629$, round down: $Y = 4$

3 $Z = Y \times 256 = 1024$

4 checksum $= X - Z = 1185 - 1024 = 161$

Working out a checksum for a block of data

Verification only proves what arrived matches what was sent — not that the data is correct, and not against deliberate tampering. **Validation** asks "is this sensible?"; **verification** asks "was this copied correctly?" — use both.



Validation checks the data makes sense; verification checks it was copied without change

Worked example. A user types their date of birth as `31/02/2009`, and types their email address twice. Which check catches which error, and what is the difference? **Validation** asks "is this data **sensible**?" - the computer tests it against a rule, and a format or range check rejects `31/02/2009` because February never has 31 days. **Verification** asks "was this data **entered correctly**?" - typing the email twice is **double entry**, and comparing the two copies catches a typing slip. The limit is what makes this a favourite question: validation can never tell you the data is **right**, only that it is **possible** - `01/02/2009` passes every validation rule even if the user was actually born on a different day. Say what each check **can** and **cannot** catch.

Exam tips

- Keep the three ideas separate: **security** (keeping data safe), **privacy** (who may see it), **integrity** (keeping it correct).
- Match each **threat** (malware, hacking, phishing, interception) to a **measure** (firewall, encryption, authentication, access rights).
- Encryption protects **confidentiality, not integrity** — use a checksum, parity or check digit for integrity.
- Distinguish a **virus, worm and Trojan** and how each spreads.