

Security, privacy and data integrity AS

Read online at pastopic.com/cambridge/computer-science-9618/notes/security-privacy-and-data-integrity

Last checked 30 September 2026

Read first: [Information representation](#)

© 2026 Pastopic. Free for personal study. Please share the link, not the file.

What you need to know

By the end of this topic you should be able to:

1. **Explain the difference between security, privacy and integrity of data.**
2. **Explain why both the data and the computer system need to be kept secure**, not just one of them.
3. **Describe the security measures that protect a computer system**, from a single stand-alone PC up to a whole network: user accounts, passwords, authentication (including digital signatures and biometrics), firewalls, anti-virus software, anti-spyware and encryption.
4. **Explain the threats that networks and the internet bring**: malware (viruses and spyware), hackers, phishing and pharming.
5. **Describe ways to reduce the risk from each threat.**
6. **Describe the methods that protect the data itself**, including encryption and access rights.
7. **Describe how validation and verification help protect the integrity of data.**
8. **Describe and use validation checks**: range, format, length, presence, existence and limit checks, and check digits.
9. **Describe and use verification methods** during data entry (visual check, double entry) and during data transfer (parity check by byte and by block, checksum).

How public and private keys, digital certificates and SSL/TLS work in detail comes later, in the A Level topic on security. Here you need only the outline of a digital signature.

Understand it

Security, privacy and integrity

- **Data security** is protecting data from being **lost, corrupted or damaged**, and from unauthorised change or deletion. A back-up keeps data secure even when nobody is attacking it: a failed disk loses the data, a back-up brings it back.

- **Data privacy** is making sure data can be seen or used **only by the people authorised to see it**. Privacy is about **who** can access the data.
- **Data integrity** is making sure data is **accurate, complete, consistent and up to date**, both when it is entered and after it is processed or sent. Integrity is about the data being **right**, and it is protected by validation and verification.

Security measures (passwords, firewalls, encryption) keep data private and safe; validation and verification keep it accurate. A hospital record that a hacker **reads** has lost its privacy; one that a virus **deletes** has lost its security; one where a nurse typed the wrong blood group has lost its integrity.

Why protect the data and the system

- **The data** must be protected because people could **read, change, delete or steal** it. Leaked, changed or lost data harms the people it is about and the organisation.
- **The computer system** (hardware, operating system, software, network) must be protected because someone who gets into it can **install malware, change settings, damage it or use it to reach the data**. If the system is down, nobody can use the data at all.

Threats from networks and the internet

A stand-alone PC can only be attacked by someone sitting at it; on a network or the internet, attackers anywhere can try to reach it.

Malware (malicious software) is any program written to harm a computer or its user. Two kinds are named in the syllabus:

- A **virus** is malicious code that **copies itself** (replicates) into other files or computers and **deletes, corrupts or changes** data and files, or makes the computer malfunction. It usually arrives hidden in a download or an email attachment.
- **Spyware** is installed **without the user knowing** and **secretly records** what they do, for example their **keystrokes** (a key logger) or the websites they visit, and **sends this to a third party**, who may use it to find passwords or bank details. Spyware does not usually damage files and does not replicate.

Both are malicious, both are installed without the user's knowledge, and both run in the background; the differences are that a virus replicates and damages data, while spyware copies data and sends it out.

A **hacker** is a person who gains **unauthorised access** to a computer or network, for example by guessing weak passwords or exploiting a flaw. Once in, they can read, change or delete data, or install malware.

Phishing is sending emails (or messages) that **pretend to be from a trusted organisation**, such as a bank, to trick the reader into **giving away personal details** or clicking a link to a fake website. Phishing needs the **user to act**: open the email, click, type.

Pharming is **malicious code** installed on a computer (or on a server that looks up web addresses) that **redirects the user to a fake website** even when they type the correct address. The fake site looks real and collects the details the user enters. Pharming is **automatic**: the user does nothing unusual.

Phishing and pharming both aim to get personal or financial details, and both use fake websites made to look like a real organisation's; the difference is how the user gets there (tricked by a message, or silently redirected).

Security measures for the system

Learn what each measure **does**; "it stops hackers" earns nothing on its own.

- **User accounts** give each person their own login, so the system knows who is using it and can apply the right access rights to them.
- **Passwords** mean only someone who knows the secret can log in. They work only if they are **strong** (long, a mix of character types, not guessable), secret and changed if they may have leaked.
- **Authentication** is proving you are who you claim to be. Passwords are one way. **Biometrics** use something you **are**: a fingerprint, face, iris or voice is scanned and compared with the stored template for that user, and it can't be forgotten or easily passed to someone else. **Two-factor authentication** asks for two different things, such as a password and a code sent to a phone.
- A **firewall** (hardware, software or both) sits between a computer or network and the outside. It **monitors all incoming and outgoing traffic**, **compares** each packet with a set of **rules** (allowed and blocked IP addresses, ports, types of traffic, an allow list or deny list), and **blocks** anything that fails the rules. It can also stop unknown programs inside sending data out.
- A **proxy server** receives users' requests, forwards them using **its own IP address** so the inside computers are hidden, and screens what comes back.
- **Anti-virus software scans** files and memory, **compares** them with a **database of known viruses** (their signatures) and watches for suspicious behaviour, then **deletes or quarantines** anything it finds. It must be **kept up to date**, because new viruses appear every day, and it can scan downloads before they are opened.
- **Anti-spyware** works the same way for spyware: it scans for known spyware, removes it, and can block programs that try to record keystrokes or send data out.
- **Physical measures**, such as locking the room or the computer, stop someone simply walking up to it.

To **restrict the risks** from malware and hackers more generally: install software only from reputable sources, keep the operating system and anti-malware updated, don't give everyday users administrator rights (so they can't install software), control removable media, don't open attachments or links in unexpected emails, use a spam filter, and keep **back-ups** so data lost to an attack can be restored.

Protecting the data: encryption and access rights

Encryption uses an algorithm and a **key** to turn readable data (**plaintext**) into **ciphertext**, which looks like meaningless characters. Only someone with the right **key** can **decrypt** it back to plaintext.

Encryption does **not** stop data being intercepted, copied or deleted. It means that **if** the data is intercepted or stolen, it **cannot be understood** without the key. It is the standard answer for protecting data **during transmission** (emails, files over the internet, a VPN), and it also protects stored files on a stolen laptop.

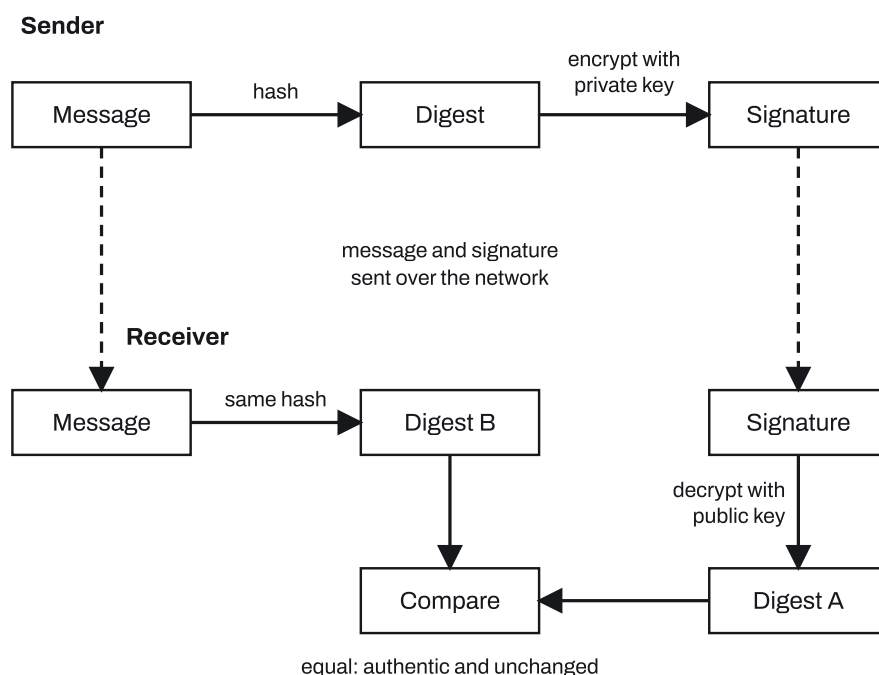
Access rights (permissions) decide what each user can do with each file or folder: **no access**, **read-only**, or **read and write** (and sometimes delete or run). A school might let teachers read and write marks but let students only read their own. Access rights give privacy and stop accidental or deliberate changes by people who shouldn't make them.

Digital signatures

A **digital signature** lets the receiver check two things about a message or document: that it really came from the **claimed sender** (it is authentic), and that it has **not been changed** on the way. It is not a picture of a handwritten signature.

It uses two ideas:

- A **hashing algorithm** turns any message into a short, fixed-length value called a **digest** (or hash). Change even one character of the message and the digest changes completely, and you cannot work backwards from the digest to the message.
- Each sender has a pair of keys: a **private key** only they have, and a matching **public key** anyone can have. Data encrypted with the private key can only be decrypted with the matching public key.



The steps:

1. The sender **hashes** the message to produce a **digest**.
2. The sender **encrypts the digest with their private key**. The result is the **digital signature**.
3. The message and the signature are **sent together** to the receiver.
4. The receiver **decrypts the signature with the sender's public key**, which gives back the sender's digest.
5. The receiver **hashes the received message** with the **same** hashing algorithm, making a second digest.

6. The receiver **compares** the two digests. If they are **the same**, the message came from the owner of the private key (authentic) and was not changed in transit. If they differ, it was altered or did not come from that sender.

Validation and verification

- **Validation** is done **by the computer**, and checks that data is **reasonable and follows set rules** (sensible, in range, in the right format). It cannot check that data is **correct**: an age of 34 passes a range check of 0 to 120 even if the person is really 43.
- **Verification** checks that data **matches the original**: that what was typed matches the source document, or that what arrived matches what was sent. It cannot check that the original was right: if the form itself says 34, verification happily confirms 34.

So data can pass both and still be wrong, because the **original data** was wrong, or because a wrong value happened to fit the rules.

Validation checks

Each check tests one rule. Using the example of a student's test mark out of 60, and a product code like KT4071:

Check	What it tests	Example
Range check	the value is between a lower and an upper limit	mark is from 0 to 60 inclusive
Limit check	the value is on the right side of one limit only	mark is not more than 60
Presence check	something has been entered (not left blank)	a mark must be entered
Length check	the number of characters is right (exact, minimum or maximum)	product code has exactly 6 characters
Format check	the characters follow a pattern	two letters followed by four digits
Existence check	the value already exists in a list or in the system	the product code is one in the product file
Check digit	an extra digit calculated from the others matches	see below

A **type check** (the data is the right data type, such as a whole number) is also accepted by mark schemes.

A **check digit** is an extra digit added to the end of a code, **calculated from the other digits**. When the code is entered, the computer recalculates the check digit and compares it with the one typed. It catches most typing errors, including two digits swapped. One common method is **modulo 11**:

1. Multiply each digit by a weight, counting down to 2 at the right: for the code 47032, the weights are 6, 5, 4, 3, 2.
2. Add the products: $4 \times 6 + 7 \times 5 + 0 \times 4 + 3 \times 3 + 2 \times 2 = 24 + 35 + 0 + 9 + 4 = 72$.

3. Divide by 11 and find the remainder: $72 = 6 \times 11 + 6$, remainder 6.
4. Check digit = $11 - 6 = 5$, so the full code is 470325. (A remainder of 0 gives check digit 0; a result of 10 is written as X.)

To check an entered code, multiply the check digit by 1 and the rest by their weights, and add: $72 + 5 = 77 = 7 \times 11$, remainder 0, so it is accepted. If someone types 407325 by mistake, the total is 70, remainder 4, so the code is rejected.

Verification during data entry

- **Visual check:** the person entering the data (or a second person) **reads what is on screen and compares it by eye** with the source document, then corrects any difference.
- **Double entry:** the data is **entered twice**, often by two different people, and **the computer compares** the two versions. If they differ, it is flagged to be entered again.

The difference to keep clear: a visual check is done **by a person**; with double entry, **the computer** does the comparing.

Verification during data transfer

Data sent over a network can be changed by interference, so the receiver checks that what arrived matches what was sent.

Parity check (byte). Sender and receiver agree on **odd** or **even** parity. Each byte holds 7 data bits and 1 **parity bit**. The sender counts the 1s in the data bits and sets the parity bit so the **total number of 1s in the byte** is even (for even parity) or odd (for odd parity). The receiver counts the 1s in each byte; if the count has the wrong parity, the byte has an error and is sent again.

For the data bits 1010110 (four 1s): with even parity the parity bit is 0 (total stays four); with odd parity it is 1 (total five). The question tells you whether the parity bit goes at the left (most significant) or right (least significant) end.

A parity byte check has two weaknesses: it cannot tell **which** bit is wrong, and if **two** bits in one byte change, the count's parity is unchanged, so the error is **not detected**.

Parity block check. The bytes are arranged as rows of a grid. Each row has its parity bit as before, and the sender also works out a parity bit for **every column** (every bit position), making an extra **parity byte** that is sent after the block. The receiver checks every row and every column. A single wrong bit makes **one row and one column** fail, and the bit where they cross is the wrong one, so it can even be corrected.

	parity bit							1s	
Byte 1	1	0	0	1	1	0	1	0	4
Byte 2	0	1	1	0	0	1	1	0	4
Byte 3	1	1	1	1	0	0	0	1	5 odd $\times$
Byte 4	0	1	0	1	0	1	1	0	4
Parity byte	0	1	0	0	1	0	1	1	
column 1s	2	4	2	3	2	2	4	2	
				odd $\times$					

In the diagram the sender's third byte was 11100001. The fourth bit flipped to 1 on the way, so row 3 now has five 1s and column 4 has three 1s (counting the parity byte too). Both are odd under even parity, so the error is at row 3, column 4.

Checksum. The sender runs an **algorithm on the whole block of data** to calculate a value, the **checksum**, and sends it with the data. The receiver runs the **same algorithm** on the data it received and **compares** its result with the checksum it was sent. If they match, the data is accepted; if not, there was an error and the block is sent again.

A simple example: add the bytes' values and keep the remainder after dividing by 256 (so it fits in one byte). For the bytes 200, 117, 64, 90, the sum is 471 and $471 - 256 = 215$, so the checksum 215 is sent. If the third byte arrives as 66, the receiver's sum is 473, giving $217 \neq 215$, so it asks for the data again. Real systems use more complex algorithms, but the exam wants the process, not a particular formula.

Key facts and formulas

This topic has no formulas, and the exam gives no formula list for it, so every fact below is **Learn it**.

Fact	Status
Security: protect from loss, corruption, damage · Privacy: only authorised people can access · Integrity: accurate, complete, consistent, up to date	Learn it
System measures: user accounts, passwords, biometrics and other authentication, digital signatures, firewall, anti-virus, anti-spyware, encryption	Learn it
Data measures: encryption, access rights (none, read-only, read and write)	Learn it
Threats: virus, spyware, hacker, phishing, pharming	Learn it
Digital signature: hash → digest → encrypt with sender's private key → send both → decrypt with sender's public key → hash again → compare	Learn it

Fact	Status
Validation (reasonable, by the computer) vs verification (matches the original)	Learn it
Validation checks: range, limit, presence, length, format, existence, check digit (type also accepted)	Learn it
Verification on entry: visual check, double entry · on transfer: parity byte, parity block, checksum	Learn it
Parity: even parity makes the total number of 1s even; odd parity makes it odd	Learn it

How to do it

In the Paper 1 questions we have tagged for this topic (29 questions, 2021–2025), the types came up this often. 20 of the 29 questions mix more than one type, so the counts add up to more than 29.

Question type	Questions
Security measures and how they protect	10
Threats and how to prevent them	9
Security, privacy and integrity	8
Encryption	7
Validation checks in context	7
Verification during data transfer	6
Verification during data entry	6
Naming the check an algorithm performs	3
Digital signatures	3
Validation versus verification	3
Parity calculations	1

Marks come from **technical words** and **applying them to the scenario**.

1. Security measures and how they protect (10 questions)

- 1. Read the exclusions.** Questions often say "other than passwords" or "authentication is one method; identify one other". Don't give the excluded one, and if two methods are asked for, make them different.
- 2. Name the measure** (usually 1 mark).
- 3. Describe how it works** in two or three linked steps, as in "Understand it" (firewall: monitor → compare with rules → block; anti-virus: scan → compare with updated database → quarantine).

4. **Link it to the scenario** where you can: "so a hacker can't reach the exam papers on the server".

Variations you will meet:

- **"State how ... protects"** a list of measures: one sentence each, saying what it achieves.
- **"Explain how the risks of malware can be restricted"**: give several measures from "Understand it", each with its reason (no administrator rights → users can't install it; back-ups → data can be restored).
- **Matching** each measure to a description: encryption converts data to another form; a firewall accepts or rejects packets by criteria; authenticity belongs to a digital signature, not encryption.
- **Storing vs sending**: "one method on the computer and one during transmission". Encryption fits transmission; a firewall, anti-malware, access rights or a locked room fit storage.
- **A proxy server** is also accepted for protecting a server.

2. Threats and how to prevent them (9 questions)

1. **Pick clearly different threats** (see Common mistakes).
2. **Describe each with its key words** from "Understand it": replicates, keystrokes, third party, unauthorised access, pretends to be, redirects.
3. **Give a prevention that matches that threat**, different for each row: anti-virus for a virus; anti-spyware (or an onscreen keyboard) for spyware; a firewall or strong passwords for hacking; a spam filter or not clicking links in unknown emails for phishing; anti-malware and checking the website address for pharming.

Variations you will meet:

- **Compare two threats**. A similarity must be true of **both** (phishing and pharming both try to get personal data; both use fake websites). A difference needs a statement about **each** side: "pharming redirects automatically, **but** phishing needs the user to click a link in an email". "A virus damages data, spyware doesn't" is too thin; add what spyware does instead.
- **Identify two threats**: one word each is enough.
- **In a context**: say what the threat would do to **that** data.

3. Security, privacy and integrity (8 questions)

1. Use the key words of the definitions in "Understand it": **loss or corruption** (security), **authorised access** (privacy), **accurate, consistent, up to date** (integrity).
2. For "the difference between", write one sentence for each term.

Variations you will meet:

- **"Why does the data need to be kept secure?"**: to stop it being lost, corrupted, changed or read by unauthorised people, and so it can be recovered.
- **"Why do both the data and the system need protecting?"**: data, so nobody can change, delete or take it; the system, so nobody can install malware, damage it or use it to reach the data. Don't just repeat "from hackers" if the question already says unauthorised access.

- **Sorting measures** into "keeps data secure" and "protects integrity": firewall, password and access rights are security; double entry and presence check are integrity.

4. Encryption (7 questions)

1. **Name encryption** (VPN is accepted as an example).
2. **Describe it**: the data is converted into **ciphertext** using an algorithm and a **key**.
3. **Say what it achieves**: if the data is **intercepted**, it **cannot be understood** without the decryption key.

5. Validation checks in context (7 questions)

1. **Find the rules in the question**: a format ("1 letter, 3 digits, 2 letters"), a range ("0 to the test's maximum"), whether it's a date, text or a number.
2. **Name each check and apply it to those values**: "length check: the registration must be exactly 6 characters", not "checks the length is right". "Range check: the month must be from 1 to 12."
3. **Give different checks** if asked for two, and skip any the question has already used (often the presence check).

Variations you will meet:

- **Non-numeric data**: format, length, presence, existence and type checks work; a range check doesn't suit text.
- **Dates**: range (the year between 1900 and this year; the month from 1 to 12; the day from 1 to the month's last day), presence, length or format (for example DD/MM/YYYY, 10 characters).
- **"Describe one other method of protecting integrity"** (other than verification): validation, described as checking that data is reasonable and within set bounds.

6. Verification during data transfer (6 questions)

1. **Name the method**: parity byte check, parity block check or checksum.
2. **Describe the sender's step, then the receiver's step, then what happens on a mismatch**, as in "Understand it".

Variations you will meet:

- **Fill in the gaps** on parity: odd or even; groups of 7 bits; parity **block**; parity **byte**.
- **Matching methods to entry or transfer**: parity byte, parity block and checksum are transfer; visual check is entry.

7. Verification during data entry (6 questions)

1. Name **visual check** and **double entry**.
2. Describe each so that it could not apply to the other: say **who compares** (a person by eye, or the computer).

3. Use the question's data: "the administrator types the tutor ID twice and the system checks both match".

8. Naming the check an algorithm performs (3 questions)

1. Read the IF condition that leads to OUTPUT "Invalid".
2. Match its shape:
 - $x < a$ OR $x > b \rightarrow$ **range check**; only $x > b$ (or only $x < a$) $\rightarrow$ **limit check**;
 - $x = \text{" "}$ $\rightarrow$ **presence check**;
 - $\text{LENGTH}(x) <> n \rightarrow$ **length check**;
 - x not equal to any value in a fixed list $\rightarrow$ **existence check**;
 - a loop looking at each character (with MID) for a pattern or a required symbol $\rightarrow$ **format check**.
3. Write just the name, and check you have not used one twice.

9. Digital signatures (3 questions)

Write the six steps from "Understand it" in order, one mark each up to the total. Name the keys (**sender's private key** to encrypt, **sender's public key** to decrypt): "encrypted with a key" loses the mark. A 2-mark "how does it ensure the email is authentic" needs the hash-to-digest step and the private-key/public-key pairing.

10. Validation versus verification (3 questions)

1. **Difference**: one sentence for each, using "same as the original" and "reasonable" from "Understand it".
2. **"Why might the data still be wrong?"**: give both reasons from "Understand it" (fits the rules but isn't the true value; the **source itself may be wrong**), tied to the question's data, for example "a registration number in the right format, but not this car's".

11. Parity calculations (1 question)

1. **Parity bit**: count the 1s in the data bits; for even parity add 0 if the count is even, 1 if odd; for odd parity the other way round. Put it where the question says (left or right end).
2. **Locate an error in a block**: count the 1s in every row, including its parity bit, and mark the rows with the wrong parity (not the parity byte itself, which is only part of the columns); do the same for every column, including the parity byte's bit. The wrong bit is where the failing row and failing column cross. Circle it, or state its row and bit position.

Worked examples

The bold codes show where marks are earned, as in Cambridge mark schemes. The number is how many marks.

- **M1** method mark: for a correct method, even if the numbers slip.
 - **A1** accuracy mark: for a correct answer; it needs the M mark before it.
 - **B1** mark for a correct result on its own, with no method needed.
-

Example 1

A public library records members on a computer system. Each membership code is two capital letters followed by four digits, for example **LB4071**.

- (a) State the difference between data validation and data verification. **[2]**
- (b) A presence check makes sure a membership code has been entered. Describe **two** other validation checks for the membership code. **[2]**
- (c) A new member fills in a paper form, and a librarian types the member's date of birth into the system. Describe **two** methods of verification the librarian can use. **[4]**
- (d) Explain why the date of birth might still be wrong after it has been validated and verified. **[2]**
- (a)** Validation checks that the data entered is reasonable and follows set rules **B1**; verification checks that the data entered matches the original source **B1**.
- (b)** Length check: the code must be exactly 6 characters long **B1**. Format check: the first two characters must be letters A–Z and the last four must be digits 0–9 **B1**.
- (c)** Visual check **B1**: the librarian reads the date on screen and compares it by eye with the date on the paper form **B1**. Double entry **B1**: the date is typed twice and the computer compares the two entries, rejecting them if they differ **B1**.
- (d)** Validation only checks the date is reasonable (for example a real date in a sensible range), not that it is this member's true date of birth **B1**. Verification only checks it matches the form, so if the member wrote the wrong date on the form, the wrong date is accepted **B1**.

Example 2

A computer sends data using **odd** parity. The parity bit is the **most significant (leftmost)** bit of each byte.

- (a) Write the complete byte for each group of 7 data bits: (i) 1101001 (ii) 0110100. **[2]**
- (b) A block of four bytes is sent, followed by a parity byte, using odd parity. The bytes received are:

	Parity bit							
Byte 1	0	0	1	1	0	0	1	0
Byte 2	1	1	0	1	1	0	0	0
Byte 3	0	0	0	0	1	1	0	1
Byte 4	0	1	1	0	0	1	1	1
Parity byte	0	1	1	1	1	0	1	1

One bit was changed during transmission. Identify it. **[2]**

(c) Explain why a parity byte check might not detect an error in a byte. **[2]**

(d) The computer also sends a checksum with each block. Describe how the checksum is used to detect an error. **[3]**

(a) (i) 1101001 has four 1s, so the parity bit is 1 to make five: 1101001 **B1**. (ii) 0110100 has three 1s, already odd, so the parity bit is 0: 00110100 **B1**.

(b) Rows: byte 2 has four 1s (even), so byte 2 fails; the others have 3, 3 and 5. Columns: column 6 has $0 + 0 + 1 + 1 + 0 = 2$ ones (even), so column 6 fails; every other column is odd. **M1** for finding the failing row and column.

The wrong bit is **byte 2, bit 6 from the left**: it arrived as 0 and should be 1. **A1** (Don't test the parity byte as a row: with four odd-parity bytes, the parity byte always has an even number of 1s.)

(c) If two bits in the same byte change (for example two 0s become 1s), the number of 1s changes by an even number **B1**, so the byte still has odd parity and passes the check even though it is wrong **B1**.

(d) Before sending, a checksum is calculated from the block of data using an algorithm **B1**, and it is sent with the data **B1**. The receiver calculates the checksum again from the data it received and compares it with the one sent; if they differ, an error has occurred and the block is sent again **B1**.

Example 3

A travel agency's staff use laptops connected to the agency's network and the internet. The laptops hold customers' passport and payment details.

(a) Complete the table by identifying **two** threats to the customers' data. Describe each threat and give a **different** prevention method for each. **[6]**

Threat	Description	Prevention
--------	-------------	------------

(b) The agency's network has a firewall. Explain how the firewall helps to protect the customers' data. **[3]**

(a) One mark for each threat, matching description and prevention.

Threat	Description	Prevention
Spyware B1	Installed on a laptop without the user knowing; secretly records keystrokes, such as a customer's card number, and sends them to a third party B1	Install and run up-to-date anti-spyware B1
Pharming B1	Malicious code redirects a member of staff from the real payment site to a fake one that collects the details they enter B1	Anti-malware to detect and remove the redirecting code B1

Hacking with strong passwords, or a virus with anti-virus, would also score.

(b) The firewall monitors all traffic coming into and going out of the network **B1**. It compares each packet with set rules, such as a list of allowed and blocked IP addresses and ports **B1**. It blocks any traffic that doesn't meet the rules, so a hacker's connection or spyware sending data out is stopped **B1**.

Example 4

An architect emails building plans to a planning office over the internet.

(a) Explain how encryption protects the plans while they are being sent. [3]

(b) The email includes a digital signature. Explain how the planning office can use it to check that the plans came from the architect and have not been changed. [5]

(a) The plans are converted into ciphertext using an encryption algorithm and a key **B1**. If the email is intercepted, the plans cannot be understood **B1** without the matching key to decrypt them **B1**.

(b) Any five of:

- The architect's software hashes the plans to produce a digest **B1**.
- The digest is encrypted with the architect's private key to make the digital signature **B1**.
- The plans and the signature are sent together.
- The planning office decrypts the signature with the architect's public key to recover the digest **B1**.
- It hashes the received plans with the same hashing algorithm to produce a second digest **B1**.
- It compares the two digests; if they match, the plans came from the architect and were not changed **B1**.

Common mistakes

- **Vague definitions.** "Security is how safe the data is and privacy is how private it is" earns nothing. Security is about loss and corruption; privacy about authorised access; integrity about accuracy and consistency. Some students also swap security and integrity round.
- **Saying validation or verification makes data correct.** Validation only checks it is reasonable; verification only checks it matches the original. Examiners see this misconception every year.

- **Generic answers to context questions.** When the question gives a date of birth, a car registration or a test mark, "a length check checks the length" scores nothing. Give the actual rule: "exactly 6 characters", "month from 1 to 12".
- **Naming instead of describing.** "Describe" needs how the check or measure works; one word such as "anti-virus" is only the name. Say that it scans, compares with a database of known viruses that is kept up to date, and quarantines.
- **"A firewall is a wall that stops viruses."** Describe it technically: monitors traffic in and out, checks it against rules, blocks what fails.
- **"Encryption stops the data being intercepted."** It doesn't. It stops intercepted data being understood without the key. Use the word ciphertext.
- **Mixing up a digital signature with a scanned or touchscreen signature,** or with a digital certificate. Describe the hash, the digest and which key (sender's private, sender's public) is used where.
- **Two threats that are the same thing,** such as "malware" and "virus". Choose clearly different threats and different preventions.
- **Answering data transfer when the question is about data entry.** Copying from a paper form needs a visual check or double entry, not parity.
- **Making visual check and double entry sound the same.** Say who compares: a person by eye, or the computer.
- **Explaining the pseudocode instead of naming the check.** "It checks x is between 0 and 10" is not the answer; "range check" is.
- **Mixing up check digits, checksums and parity blocks.** A check digit is validation on a typed code; a checksum and a parity block are verification of transmitted data.

How to get the marks

- **Identify, state, give:** a name or a short fact is enough. **Describe:** say what it is and how it works, in two or three linked points. **Explain:** give reasons, including why or how it helps in this situation.
- **Name + description tables** usually give 1 mark for the name and 1 or 2 for a matching description. The description must match the name you wrote; a correct description of a different method scores nothing.
- **Use the key technical words:** ciphertext, key, unauthorised access, replicates, keystrokes, third party, redirects, digest, private key, public key, reasonable, matches the original. Examiners want technical terms at this level.
- **Apply to the scenario.** If the question is about a bank, a school or a car registration, mention it. Generic textbook descriptions often lose marks.
- **"Each must be different"** means exactly that: two threats, two preventions, two checks, all different.
- **Don't repeat the question.** If it says "to prevent unauthorised access", writing "to stop hackers" adds nothing.
- **Comparisons:** a similarity is true of both; a difference needs something about each one.

Quick check

1. State the validation check each algorithm performs. Each check is different.

(i) IF $x > 250$ THEN OUTPUT "Invalid" ENDIF

(ii) IF LENGTH(x) $\neq$ 8 THEN OUTPUT "Invalid" ENDIF

(iii) IF $x = ""$ THEN OUTPUT "Invalid" ENDIF

(iv) IF $x < 1$ OR $x > 31$ THEN OUTPUT "Invalid" ENDIF

2. A system uses even parity, with the parity bit as the least significant (rightmost) bit. Write the complete bytes for the data bits 0111010 and 1000001, and say whether the received byte 10110110 is accepted.

3. A checksum is the sum of the byte values, keeping the remainder after dividing by 256. Find the checksum for the bytes 150, 98, 33, 7, and say what the receiver does if the bytes arrive as 150, 98, 35, 7.

4. A code uses a modulo 11 check digit with weights 5, 4, 3, 2 on its four digits. Find the check digit for 3814.

5. Give one similarity and one difference between a virus and spyware.

Answers

1. (i) Limit check (only an upper limit). (ii) Length check. (iii) Presence check. (iv) Range check.
2. 0111010 has four 1s, so the parity bit is 0: 01110100. 1000001 has two 1s, so the parity bit is 0: 10000010. 10110110 has five 1s, which is odd, so it is **rejected** as an error.
3. $150 + 98 + 33 + 7 = 288$, and $288 - 256 = 32$, so the checksum is 32. The receiver gets $150 + 98 + 35 + 7 = 290$, giving $34 \neq 32$, so it detects an error and asks for the block to be sent again.
4. $3 \times 5 + 8 \times 4 + 1 \times 3 + 4 \times 2 = 15 + 32 + 3 + 8 = 58$. $58 = 5 \times 11 + 3$, remainder 3, so the check digit is $11 - 3 = 8$ and the code is 38148.
5. Similarity: both are malicious software installed without the user's knowledge (both run in the background). Difference: a virus replicates itself and corrupts or deletes data, while spyware does not replicate but records the user's actions and sends them to a third party.

Past questions to try

- **9618/13/M/J/22 Q5**: why data must be secure, two ways to keep it secure, a verification method for copying from paper, range, presence and length checks on a date of birth, and why checked data can still be wrong.
- **9618/12/M/J/22 Q4**: why both data and the computer system need protecting, a table of two threats with descriptions and preventions, and what encryption is and why it is used.
- **9618/13/M/J/21 Q6**: naming the validation check in three short algorithms.
- **9618/12/O/N/25 Q6**: part (c) only: a parity bit, then finding the altered bit in a parity block (the other parts are on compression, reading binary codes and file size).