

## 6 Security, privacy and data integrity

In this chapter, you will learn about

- the terms security, privacy and integrity of data
- the need for security of data and security of computer systems
- security measures to protect computer systems such as user accounts, passwords, digital signatures, firewalls, antivirus and anti-spyware software and encryption
- security threats such as viruses and spyware, hacking, phishing and pharming
- methods used to reduce security risks such as encryption and access rights
- the use of validation to protect data integrity
- the use of verification during data entry and data transfer to reduce or eliminate errors.

## 6.1 Data security

### WHAT YOU SHOULD ALREADY KNOW

Try these five questions before you read the first part of this chapter.

- 1 a) What is meant by *hacking*?  
b) Is hacking always an illegal act? Justify your answer.
- 2 Contactless credit cards and debit cards are regarded by some as a security risk. Discuss the advantages and disadvantages of using contactless cards with particular reference to data security.
- 3 What are the main differences between *cracking* and *hacking*?
- 4 a) What are pop-ups when visiting a website? Are they a security risk?  
b) What are cookies? Do cookies pose a security threat?  
c) Describe:
  - i) session cookies
  - ii) permanent cookies
  - iii) third party cookies.
- 5 Why must the correct procedures be carried out when removing a memory stick from a computer?

### Key terms

**Data privacy** – the privacy of personal information, or other information stored on a computer, that should not be accessed by unauthorised parties.

**Data protection laws** – laws which govern how data should be kept private and secure.

**Data security** – methods taken to prevent unauthorised access to data and to recover data if lost or corrupted.

**User account** – an agreement that allows an individual to use a computer or network server, often requiring a user name and password.

**Authentication** – a way of proving somebody or something is who or what they claim to be.

**Access rights (data security)** – use of access levels to ensure only authorised users can gain access to certain data.

**Malware** – malicious software that seeks to damage or gain unauthorised access to a computer system.

**Firewall** – software or hardware that sits between a computer and external network that monitors and filters all incoming and outgoing activities.

**Anti-spyware software** – software that detects and removes spyware programs installed illegally on a user's computer system.

**Encryption** – the use of encryption keys to make data meaningless without the correct decryption key.

**Biometrics** – use of unique human characteristics to identify a user (such as fingerprints or face recognition).

**Hacking** – illegal access to a computer system without the owner’s permission.

**Malicious hacking** – hacking done with the sole intent of causing harm to a computer system or user (for example, deletion of files or use of private data to the hacker’s advantage).

**Ethical hacking** – hacking used to test the security and vulnerability of a computer system. The hacking is carried out with the permission of the computer system owner, for example, to help a company identify risks associated with malicious hacking of their computer systems.

**Phishing** – legitimate-looking emails designed to trick a recipient into giving their personal data to the sender of the email.

**Pharming** – redirecting a user to a fake website in order to illegally obtain personal data about the user.

**DNS cache poisoning** – altering IP addresses on a DNS server by a ‘pharmer’ or hacker with the intention of redirecting a user to their fake website.

## 6.1.1 Data privacy

Data stored about a person or an organisation must remain private and unauthorised access to the data must be prevented – **data privacy** is required.

This is achieved partly by **data protection laws**. These laws vary from country to country, but all follow the same eight guiding principles.

---

- 1 Data must be fairly and lawfully processed.
  - 2 Data can only be processed for the stated purpose.
  - 3 Data must be adequate, relevant and not excessive.
  - 4 Data must be accurate.
  - 5 Data must not be kept longer than necessary.
  - 6 Data must be processed in accordance with the data subject's rights.
  - 7 Data must be kept secure.
  - 8 Data must not be transferred to another country unless that country also has adequate protection.
- 

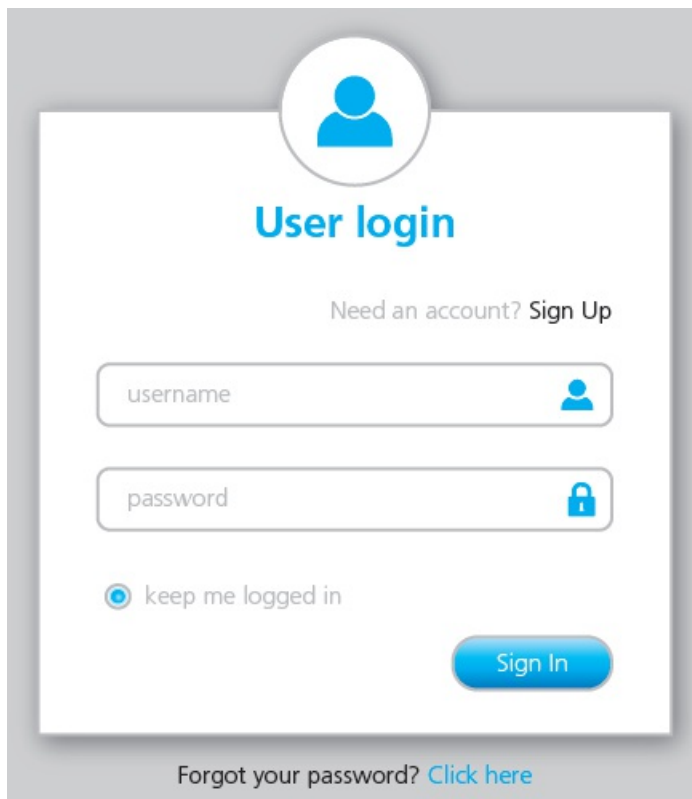
Data protection laws usually cover organisations rather than private individuals. Such laws are no guarantee of privacy, but the legal threat of fines or jail sentences deters most people.

## 6.1.2 Preventing data loss and restricting data access

**Data security** refers to the methods used to prevent unauthorised access to data, as well as to the data recovery methods if it is lost.

### *User accounts*

**User accounts** are used to **authenticate** a user (prove that a user is who they say they are). User accounts are used on both standalone and networked computers in case the computer can be accessed by a number of people. This is often done by a screen prompt asking for a username and password:



**Figure 6.1** A login screen

User accounts control **access rights**. This often involves levels of access. For example, in a hospital it would not be appropriate for a cleaner to have access to data about one of the patients. However, a consultant would need such access. Therefore, most systems have a hierarchy of access levels depending on a person's level of security. This could be achieved by username and password with each username (account) linked to the appropriate level of access.

### **EXTENSION ACTIVITY 6A**

An airport uses a computer system to control security, flight bookings, passenger lists, administration and customer services. Describe how it is possible to ensure the safety of the data on the system so that senior staff can see all data, while customers can only access flight

times (arrivals and departures) and duty free offers.

## *Use of passwords*

Passwords are used to restrict access to data or systems. They should be hard to crack and changed frequently to retain security. Passwords can also take the form of biometrics (such as on a mobile phone, as discussed later). Passwords are also used, for example, when

- accessing email accounts
- carrying out online banking or shopping
- accessing social networking sites.

It is important that passwords are protected. Some ways of doing this are to

- run anti-spyware software to make sure your passwords are not being relayed to whoever put the spyware on your computer
- regularly change passwords in case they have been seen by someone else, illegally or accidentally
- make sure passwords are difficult to crack or guess (for example, do not use your date of birth or pet's name).

Passwords are grouped as either strong (hard to crack or guess) or weak (relatively easy to crack or guess). Strong passwords should contain

- at least one capital letter
- at least one numerical value
- at least one other keyboard character (such as @, \*, &)

Example of a strong password: Sy12@#TT90kj=0

Example of a weak password: GREEN

## **EXTENSION ACTIVITY 6B**

Which of the following are weak passwords and which are strong passwords?

Explain your decision in each case.

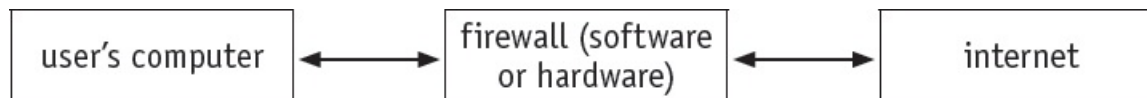
- a)** 25-May-2000
- b)** Pas5word
- c)** ChapTer@06
- d)** N55!
- e)** 12345X

## *Digital signatures*

Digital signatures protect data by providing a way of identifying the sender of, for example, an email. These are covered in more depth in [Chapter 17](#).

## *Use of firewalls*

A **firewall** can be software or hardware. It sits between the user's computer and an external network (such as the internet) and filters information in and out of the computer. This allows the user to decide to allow communication with an external source and warns a user that an external source is trying to access their computer. Firewalls are the primary defence to any computer system to protect from hacking, **malware** (viruses and spyware), phishing and pharming.



**Figure 6.2** Firewall

The tasks carried out by a firewall include

- examining the traffic between the user's computer (or internal network) and a public network (such as the internet)
- checking whether incoming or outgoing data meets a given set of criteria
- blocking the traffic if the data fails to meet the criteria, and giving the user (or network manager) a warning that there may be a security issue
- logging all incoming and outgoing traffic to allow later interrogation by the user (or network manager)
- preventing access to certain undesirable sites – the firewall can keep a list of all undesirable IP addresses
- helping to prevent viruses or hackers entering the user's computer (or internal network)
- warning the user if some software on their system is trying to access an external data source (such as an automatic software upgrade). The user is given the option of allowing it to go ahead or request that such access is denied.

The firewall can be a hardware interface which is located somewhere between the computer (or internal network external link) and the internet connection. In these cases, it is often referred to as a gateway. Alternatively, the firewall can be software installed on a computer, sometimes as part of the operating system.

However, sometimes the firewall cannot prevent potential harmful traffic. It cannot

- prevent individuals, on internal networks, using their own modems to by-pass the firewall
- control employee misconduct or carelessness (for example, control of passwords or user accounts)
- prevent users on stand-alone computers from disabling the firewall.

These issues require management and/or personal control to ensure the firewall can work effectively.

## ***Antivirus software***

Running antivirus software in the background on a computer will constantly check for virus attacks. Although different types of antivirus software work in different ways, they all

- check software or files before they are run or loaded on a computer
- compare possible viruses against a database of known viruses
- carry out heuristic checking (check software for behaviour that could indicate a virus, which is

- useful if software is infected by a virus not yet on the database)
- quarantine files or programs which are possibly infected and
    - allow the virus to be automatically deleted, or
    - allow the user to make the decision about deletion (it is possible that the user knows that the file or program is not infected by a virus – this is known as a false positive and is one of the drawbacks of antivirus software).

Antivirus software needs to be kept up to date since new viruses are constantly being discovered. Full system checks need to be carried out regularly (once a week, for example), since some viruses lie dormant and would only be picked up by this full system scan.

## *Anti-spyware software*

**Anti-spyware software** detects and removes spyware programs installed illegally on a user's computer system. The software is either based on rules (it looks for typical features associated with spyware) or based on known file structures which can identify common spyware programs.

## *Encryption*

If data on a computer has been accessed illegally (by a hacker, for example) it is possible to encrypt the data, making it virtually impossible to understand without **encryption** keys to decode it. This cannot stop a hacker from deleting the files, but it will stop them using the data for themselves. This is covered in more depth in [Chapter 17](#).

## *Biometrics*

In an attempt to stay one step ahead of hackers and malware writers, many modern computer devices use **biometrics** as part of the password system. Biometrics rely on the unique characteristics of human beings. Examples include fingerprint scans, retina scans (pattern of blood capillary structure), face recognition and voice recognition.

### *Fingerprint scans*

Images of fingerprints are compared against previously scanned fingerprints stored in a database; if they match then access is allowed; the system compares patterns of 'ridges' and 'valleys' which are fairly unique (accuracy is about 1 in 500).





**Figure 6.3** Fingerprint

### ***Retina scans***

Retina scans use infra-red to scan the unique pattern of blood vessels in the retina (at the back of the eye). It requires a person to stay still for 10 to 15 seconds while the scan takes place; it is very secure since nobody has yet found a way to duplicate the blood vessels patterns' (accuracy is about 1 in 10 million).



**Figure 6.4** Retina scan

Mobile phones use biometrics to identify if the phone user is the owner.

## 6.1.3 Risks to the security of stored data

### *Hacking*

You will see the term **hacking** used throughout this textbook. There are two types of hacking: malicious and ethical.

**Malicious hacking** is the illegal access to a computer system without the user's permission or knowledge. It is usually employed with the intention of deleting, altering or corrupting files, or to gain personal details such as bank account details. Strong passwords, firewalls and software which can detect illegal activity all guard against hacking.

**Ethical hacking** is authorised by companies to check their security measures and how robust their computer systems are to resist hacking attacks. It is legal, and is done with a company's permission with a fee paid to the ethical hacker.

### *Malware*

Malware is one of the biggest risks to the integrity and security of data on a computer system. Many software applications sold as antivirus are capable of identifying and removing most of the forms of malware described below.

#### *Viruses*

Programs or program code that can replicate and/or copy themselves with the intention of deleting or corrupting files or causing the computer to malfunction. They need an active host program on the target computer or an operating system that has already been infected before they can run.

#### *Worms*

A type of stand-alone virus that can replicate themselves with the intention of spreading to other computers; they often use networks to search out computers with weak security.

#### *Logic bombs*

Code embedded in a program on a computer. When certain conditions are met (such as a specific date) they are activated to carry out tasks such as deleting files or sending data to a hacker.

#### *Trojan horses*

Malicious programs often disguised as legitimate software. They replace all or part of the legitimate software with the intent of carrying out some harm to the user's computer system.

#### *Bots (internet robots)*

Not always harmful and can be used, for example, to search automatically for an item on the internet. However, they can cause harm by taking control over a computer system and launching attacks.

#### *Spyware*

Software that gathers information by monitoring, for example, key presses on the user's

keyboard. The information is then sent back to the person who sent the software – sometimes referred to as key logging software.

## ***Phishing***

**Phishing** is when someone sends legitimate-looking emails to users. They may contain links or attachments which, when clicked, take the user to a fake website, or they may trick the user into responding with personal data such as bank account details or credit card numbers. The email often appears to come from a trusted source such as a bank or service provider. The key is that the recipient has to carry out a task (click a link, for example) before the phishing scam causes harm.

There are numerous ways to help prevent phishing attacks:

- Users need to be aware of new phishing scams. Those people in industry or commerce should undergo frequent security awareness training to become aware of how to identify phishing (and pharming) scams.
- Do not click on links unless certain that it is safe to do so; fake emails can often be identified by greetings such as ‘Dear Customer’ or ‘Dear emailperson@gmail.com’, and so on.
- It is important to run anti-phishing toolbars on web browsers (this includes tablets and mobile phones) since these will alert the user to malicious websites contained in an email.
- Look out for https and/or the green padlock symbol in the address bar (both suggest that traffic to and from the website is encrypted).
- Regularly check online accounts and frequently change passwords.
- Ensure an up-to-date browser, with all of the latest security upgrades, is running, and run a good firewall in the background at all times. A combination of a desktop firewall (usually software) and a network firewall (usually hardware) considerably reduces risk.
- Be wary of pop-ups – use the web browser to block them; if pop-ups get through your defences, do not click on ‘cancel’ since this often leads to phishing or pharming sites – the best option is to select the small X in the top right hand corner of the pop-up window, which closes it down.

## ***Pharming***

**Pharming** is malicious code installed on a user’s computer or on a web server. The code redirects the user to a fake website without their knowledge (the user does not have to take any action, unlike phishing). The creator of the malicious code can gain personal data such as bank details from users. Often, the website appears to belong to a trusted company and can lead to fraud or identity theft.

### ***Why does pharming pose a threat to data security?***

Pharming redirects users to a fake or malicious website set up by, for example, a hacker. Redirection from a legitimate website can be done using **DNS cache poisoning**.

Every time a user types in a URL, their web browser contacts the DNS server. The IP address of the website is then sent back to their web browser. However, DNS cache poisoning changes the real IP address values to those of the fake website consequently, the user’s computer connects to the fake website.

Pharmers can also send malicious programming code to a user's computer. The code is stored on the HDD without their knowledge. Whenever the user types in the website address of the targeted website, the malicious programming code alters the IP address sent back to their browser which redirects it to the fake website.

### ***Protection against pharming***

It is possible to mitigate the risk of pharming by

- using antivirus software, which can detect unauthorised alterations to a website address and warn the user
- using modern web browsers that alert users to pharming and phishing attacks
- checking the spelling of websites
- checking for https and/or the green padlock symbol in the address bar.

It is more difficult to mitigate risk if the DNS server itself has been infected (rather than the user's computer).

---

## **EXTENSION ACTIVITY 6C**

Pharmers alter IP addresses in order to send users to fake websites. However, the internet does not only have one DNS server. Find out how a user's internet service provider (ISP) uses its own DNS servers which cache information from other internet DNS servers.

---

## 6.1.4 Data recovery

This section covers the potential impact on data caused by accidental mal-operation, hardware malfunction and software malfunction.

In each case, the method of data recovery and safeguards to minimise the risk are considered.

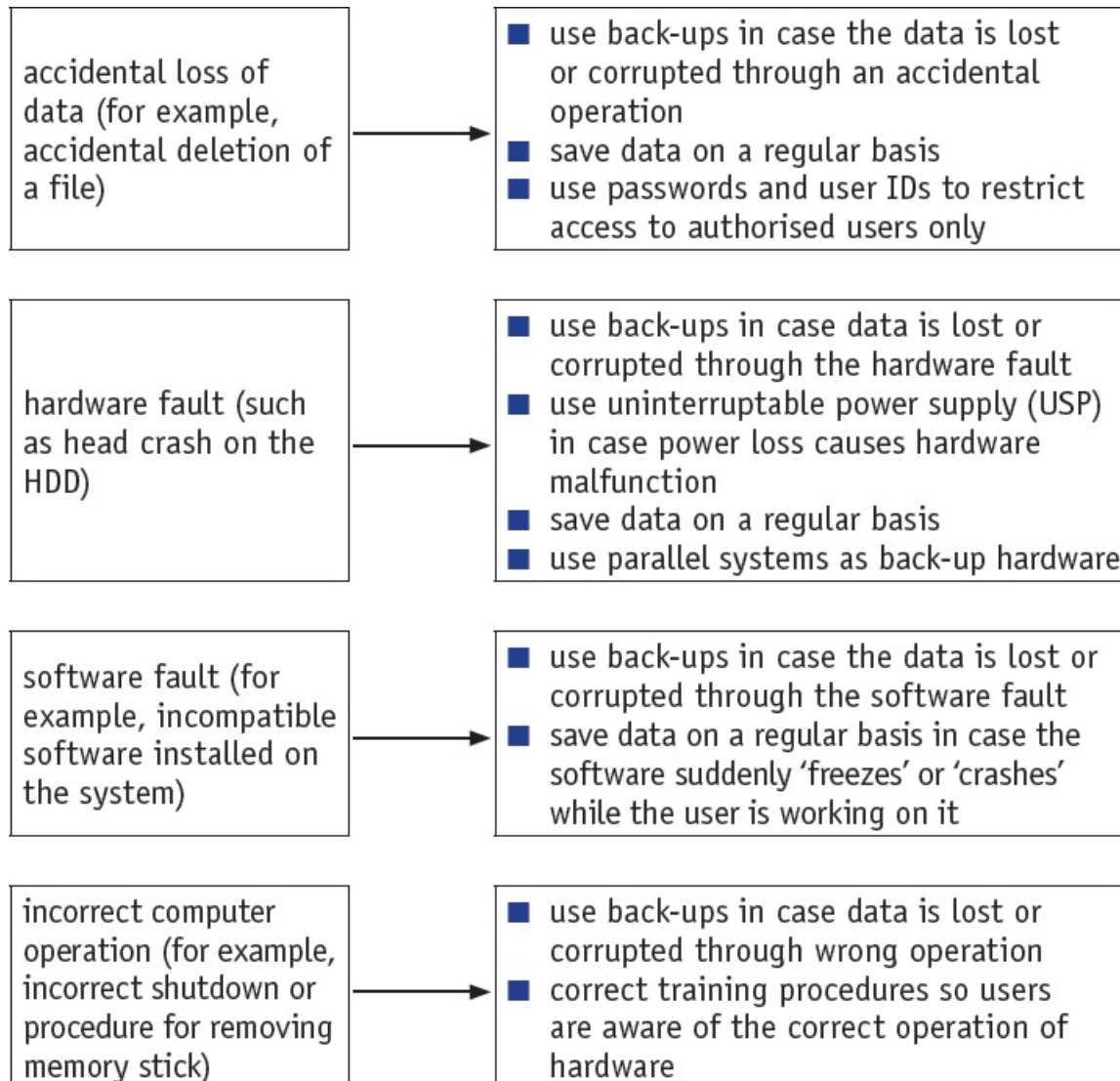


Figure 6.5 Safeguards

In all cases, the backing up of data regularly (automatically and/or manually at the end of the day) onto another medium (such as cloud storage, or removable HDD) is key to data recovery. The back-up should be stored in a separate location in case of, for example, a fire or an office break-in. Somebody should be given the role of carrying out back-ups, to ensure it is always done.

Backing up data may not be a suitable method of recovery in the case of a virus infection, as the backed up data may contain strands of the virus which could re-infect the 'cleaned' computer.

## ACTIVITY 6A

- 1 A company has offices in four different countries. Communication and data sharing between the offices is done via computers connecting over the internet.  
Describe **three** data security issues the company might encounter during their day to day communications and data sharing.  
For each issue described, explain why it could be a threat to the security of the company.  
For each issue described, describe a way to mitigate the threat which has been posed.
- 2 Define these three terms.
  - a) Worm
  - b) Logic bomb
  - c) Trojan horse
- 3 John works for a car company. He maintains the database which contains all the personal data of the people working for the car company. John was born on 28 February 1990 and has two pet cats called Felix and Max.
  - a) John needs to use a password and a username to log onto the database. Why would the following three passwords **not** be a good choice?
    - i) 280290
    - ii) FeLix1234
    - iii) John04
  - b) Describe how John could improve his passwords.  
How should he maintain his passwords to maximise database security?
  - c) When John enters a password on his computer, he is presented with the following question on screen.

Would you like to save the password on this device?

Why is it important that John always says 'no' to this question?

- d) John frequently orders goods from an online company called NILE.com.  
He opens an email which purports to be from NILE.com.

Dear NILE.com user

This is to confirm your recent order for:

01230123 A level Computer Science Workbook, \$15.90

If this is not your order, please click on the following link and update your details:

[myorders@NILE.com](mailto:myorders@NILE.com)

Thank you. Customer services.

Explain why John should be suspicious of the email.

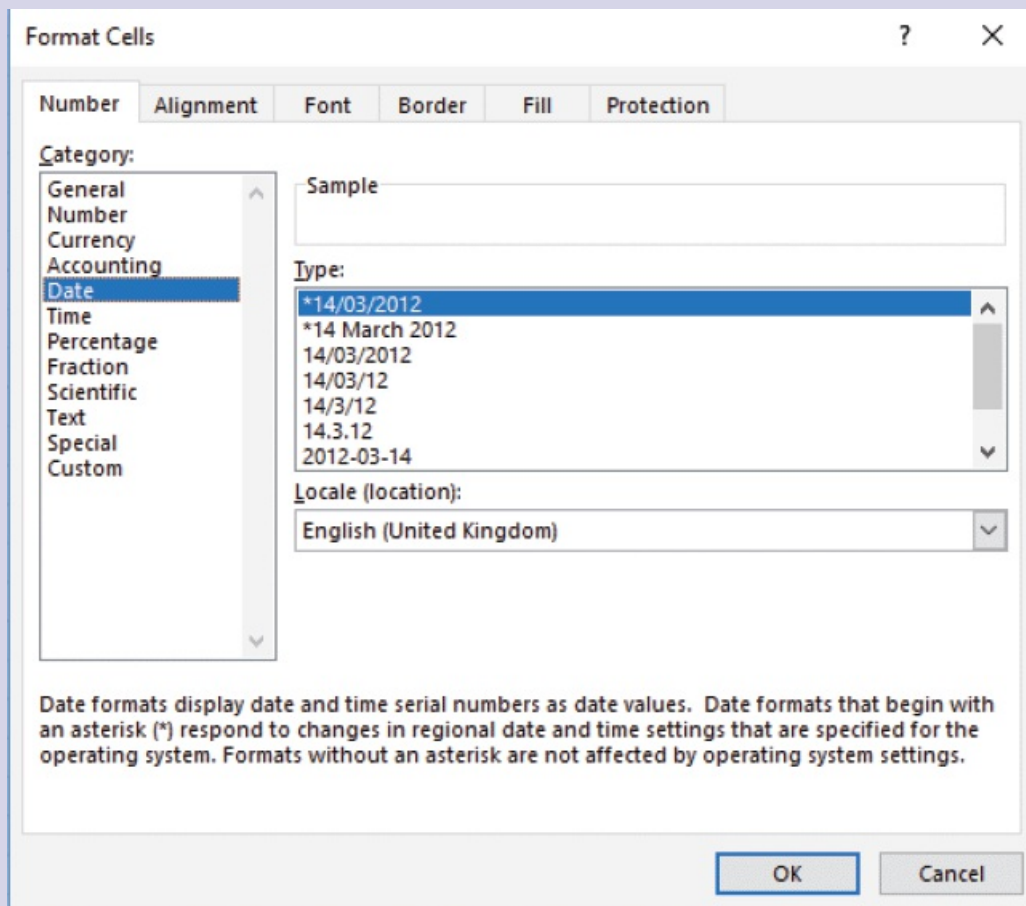
Include, in your explanation, the type of security threat identified by this email.

## 6.2 Data integrity

### WHAT YOU SHOULD ALREADY KNOW

Try these three questions before you read the second part of this chapter.

- 1 Look at the following validation screen from a spreadsheet.  
Why is it important to have validation in applications such as spreadsheets?



- 2 Why is proofreading not the same as verification?
- 3 Discuss **one** way online form designers can ensure that only certain data can be input by a user. Use the date: 12 March 2019 as the example.

### Key terms

**Data integrity** – the accuracy, completeness and consistency of data.

**Validation** – method used to ensure entered data is reasonable and meets certain input criteria.

**Verification** – method used to ensure data is correct by using double entry or visual checks.

**Check digit** – additional digit appended to a number to check if entered data is error free.

**Modulo-11** – method used to calculate a check digit based on modulus division by 11.

**Checksum** – verification method used to check if data transferred has been altered or corrupted, calculated from the block of data to be sent.

**Parity check** – method used to check if data has been transferred correctly that uses even or odd parity.

**Parity bit** – an extra bit found at the end of a byte that is set to 1 if the parity of the byte needs to change to agree with sender/receiver parity protocol.

**Odd parity** – binary number with an odd number of 1-bits.

**Even parity** – binary number with an even number of 1-bits.

**Parity block** – horizontal and vertical parity check on a block of data being transferred.

Data stored on a computer should always be accurate, consistent and up to date. Two of the methods used to ensure data integrity are validation and verification.

The accuracy (**integrity**) of data can be compromised

- during the data entry and data transmission stages
- by malicious attacks on the data, for example caused by malware and hacking
- by accidental data loss caused through hardware issues.

These risks – together with ways of mitigating them – are discussed in the rest of this chapter.



## 6.2.1 Validation

**Validation** is a method of checking if entered data is reasonable (and within a given criteria), but it cannot check if data is correct or accurate. For example, if somebody accidentally enters their age as 62 instead of 26, it is reasonable but not accurate or correct. Validation is carried out by computer software; the most common types are shown in [Table 6.1](#).

Validation test	Description	Example of data failing validation test	Example of data passing validation test
type	checks whether non-numeric data has been input into a numeric-only field	typing sk.34 in a field which should contain the price of an item	typing 34.50 in a field which should contain the price of an item
range	checks whether data entered is between a lower and an upper limit	typing in somebody's age as -120	typing in somebody's age as 48
format	checks whether data has been entered in the agreed format	typing in the date as 12-12-20 where the format is dd/mm/yyyy	typing in the date as 12/12/2020 where the format is dd/mm/yyyy
length	checks whether data has the required number of characters or numbers	typing in a telephone number as 012 345 678 when it should contain 11 digits	typing in a telephone number as 012 345 678 90 when it should contain 11 digits
presence	checks to make sure a field is not left empty when it should contain data	please enter passport number:.....	please enter passport number: AB 1234567 CD
existence	checks if data in a file or a file name actually exists	data look up for car registration plate A123 BCD which does not exist	data look up for a file called books_in_stock which exists in a database
limit check	Checks only one of the limits (such as the upper limit OR the lower limit)	typing in age as -25 where the data entered should not be negative	typing in somebody's age as 72 where the upper limit is 140
consistency check	checks whether data in two or more fields match up correctly	typing in Mr in the title field and then choosing female in the sex field	typing in Ms in the title field and then choosing female in the sex field
uniqueness check	checks that each entered value is unique	choosing the user name MAXIMUS222 in a social networking site but the user name already exists	choosing the website name Aristooo.com which is not already used

**Table 6.1** Common validation

## Key terms

**Parity byte** – additional byte sent with transmitted data to enable vertical parity checking (as

well as horizontal parity checking) to be carried out.

**Automatic repeat request (ARQ)** – a type of verification check.

**Acknowledgement** – message sent to a receiver to indicate that data has been received without error.

**Timeout** – time allowed to elapse before an acknowledgement is received.

---

## 6.2.2 Verification

**Verification** is a way of preventing errors when data is entered manually (using a keyboard, for example) or when data is transferred from one computer to another.

### *Verification during data entry*

When data is manually entered into a computer it needs to undergo verification to ensure there are no errors. There are three ways of doing this: double entry, visual check and check digits.

#### *Double entry*

Data is entered twice, using two different people, and then compared (either after data entry or during the data entry process).

#### *Visual check*

Entered data is compared with the original document (in other words, what is on the screen is compared to the data on the original paper documents).

#### *Check digits*

The **check digit** is an additional digit added to a number (usually in the right-most position). They are often used in barcodes, ISBNs (found on the cover of a book) and VINs (vehicle identification number). The check digit can be used to ensure the barcode, for example, has been correctly inputted. The check digit can catch errors including

- an incorrect digit being entered (such as 8190 instead of 8180)
- a transposition error where two numbers have been swapped (such as 8108 instead of 8180)
- digits being omitted or added (such as 818 or 81180 instead of 8180)
- phonetic errors such as 13 (thirteen) instead of 30 (thirty).

Figure 6.6 shows a barcode with an ISBN-13 code with check digit.



Figure 6.6 Barcode

An example of a check digit calculation is **modulo-11**. The following algorithm is used to generate the check digit for a number with seven digits:

- 1 Each digit in the number is given a weighting of 7, 6, 5, 4, 3, 2 or 1, starting from the left.
- 2 The digit is multiplied by its weighting and then each value is added to make a total.
- 3 The total is divided by 11 and the remainder subtracted from 11.
- 4 The check digit is the value generated; note if the check digit is 10 then X is used.

For example:

Seven digit number:	4 1 5 6 7 1 0
Weighting values:	7 6 5 4 3 2 1
Sum:	$(7 \times 4) + (6 \times 1) + (5 \times 5) + (4 \times 6) + (3 \times 7)$ $+ (2 \times 1) + (1 \times 0)$ $= 28 + 6 + 25 + 24 + 21 + 2 + 0$
Total:	= 106
Divide total by 11:	9 remainder 7
subtract remainder from 11:	$11 - 7 = 4$ (check digit)
final number:	4 1 5 6 7 1 0 4

When this number is entered, the check digit is recalculated and, if the same value is not generated, an error has occurred. For example, if 4 1 5 7 6 1 0 4 was entered, the check digit generated would be 3, indicating an error.

## EXTENSION ACTIVITY 6D

- 1 Find out how the ISBN-13 method works and confirm that the number 978 034 098 382 has a check digit of 9.
- 2 Find the check digits for the following numbers using both modulo-11 and ISBN-13.
  - a) 213 111 000 428
  - b) 909 812 123 544
- 3 Find a common use for the modulo-11 method of generating check digits.

## Verification during data transfer

When data is transferred electronically from one device to another, there is always the possibility of data corruption or even data loss. A number of ways exist to minimise this risk.

### Checksums

A **checksum** is a method to check if data has been changed or corrupted following data transmission. Data is sent in blocks and an additional value, the checksum, is sent at the end of the block of data.

To explain how this works, we will assume the checksum of a block of data is 1 byte in length. This gives a maximum value of  $2^8 - 1 (= 255)$ . The value 0000 0000 is ignored in this calculation. The following explains how a checksum is generated.

If the sum of all the bytes in the transmitted block of data is  $\leq 255$ , then the checksum is this value. However, if the sum of all the bytes in the data block  $> 255$ , then the checksum is found using the following simple algorithm.

In the example we will assume the value of X is 1185.

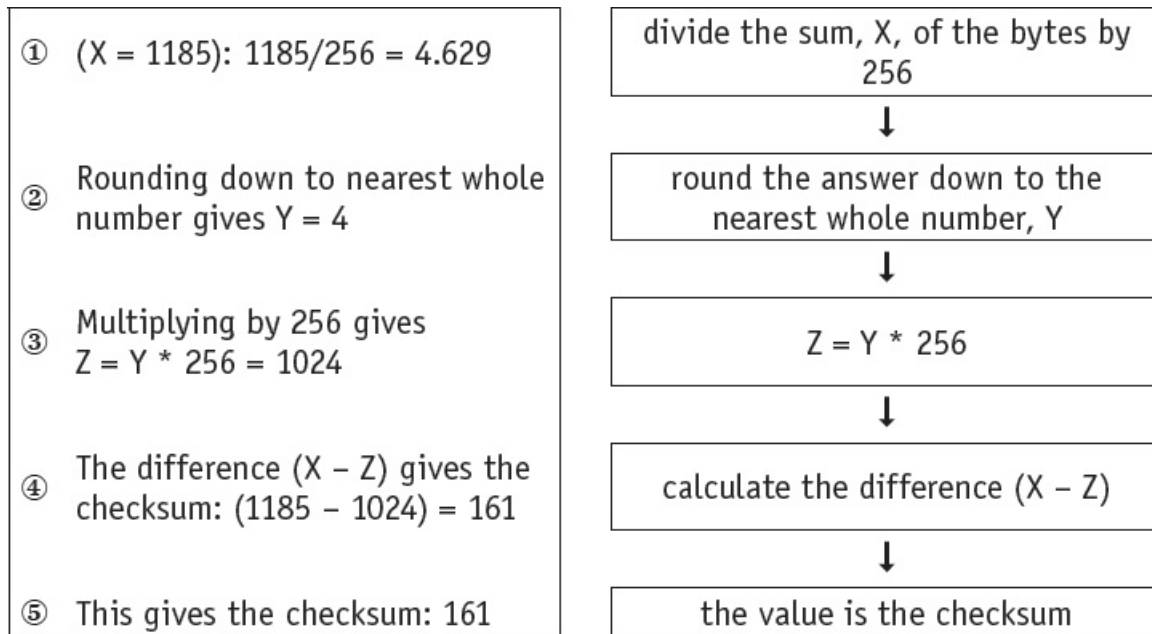


Figure 6.7

When a block of data is about to be transmitted, the checksum for the bytes is first calculated. This value is transmitted with the block of data. At the receiving end, the checksum is re-calculated from the block of data received. This calculated value is compared to the checksum transmitted. If they are the same, then the data was transmitted without any errors; if they are different, then a request is sent for the data to be re-transmitted.

### Parity checks

A **parity check** is another method to check whether data has been changed or corrupted following transmission from one device or medium to another.

A byte of data, for example, is allocated a **parity bit**. This is allocated before transmission. Systems that use **even parity** have an even number of 1-bits; systems that use **odd parity** have an odd number of 1-bits.

Consider the following byte:

	1	1	0	1	1	0	0
--	---	---	---	---	---	---	---

parity bit

Figure 6.8

If this byte is using even parity, then the parity bit needs to be 0 since there is already an even number of 1-bits (in this case, four).

If odd parity is being used, then the parity bit needs to be 1 to make the number of 1-bits odd. Therefore, the byte just before transmission would be:

either (even parity):

0	1	1	0	1	1	0	0
---	---	---	---	---	---	---	---

parity bit

or (odd parity):

1	1	1	0	1	1	0	0
---	---	---	---	---	---	---	---

parity bit

Figure 6.9

Before data is transferred, an agreement is made between sender and receiver regarding which of the two types of parity are used. This is an example of a protocol.

## EXTENSION ACTIVITY 6E

Find the parity bits for each of the following bytes:

- 1 1 1 0 1 1 0 1 even parity used
- 2 0 0 0 1 1 1 1 even parity used
- 3 0 1 1 1 0 0 0 even parity used
- 4 1 1 1 0 1 0 0 odd parity used
- 5 1 0 1 1 0 1 1 odd parity used

If a byte has been transmitted from 'A' to 'B', and even parity is used, an error would be flagged if the byte now had an odd number of 1-bits at the receiver's end.

For example:

For example:

Sender's byte:

0	1	0	1	1	1	0	0
---	---	---	---	---	---	---	---

parity bit

Receiver's byte:

0	1	0	0	1	1	0	0
---	---	---	---	---	---	---	---

parity bit

Figure 6.10

In this case, the receiver's byte has three 1-bits, which means it now has odd parity, while the byte from the sender had even parity (four 1-bits). This means an error has occurred during the transmission of the data.

The error is detected by the computer re-calculating the parity of the byte sent. If even parity has been agreed between sender and receiver, then a change of parity in the received byte indicates that a transmission error has occurred.

## EXTENSION ACTIVITY 6F

1 Which of the following bytes have an error following data transmission?

- a) 1 1 1 0 1 1 0 1 even parity used
- b) 0 1 0 0 1 1 1 1 even parity used

- c) 0 0 1 1 1 0 0 0    even parity used
- d) 1 1 1 1 0 1 0 0    odd parity used
- e) 1 1 0 1 1 0 1 1    odd parity used

2 In each case where an error occurs, can you work out which bit is incorrect?

Naturally, any of the bits in the above example could have been changed leading to a transmission error. Therefore, even though an error has been flagged, it is impossible to know exactly which bit is in error.

One of the ways around this problem is to use **parity blocks**. In this method, a block of data is sent and the number of 1-bits are totalled horizontally and vertically (in other words, a parity check is done in both horizontal and vertical directions). As the following example shows, this method not only identifies that an error has occurred but also indicates *where* the error is.

In this example, nine bytes of data have been transmitted. Agreement has been made that even parity will be used. Another byte, known as the **parity byte**, has also been sent. This byte consists entirely of the parity bits produced by the vertical parity check. The parity byte also indicates the end of the block of data.

Table 6.2 shows how the data arrived at the receiving end:

	parity bit	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8
byte 1	1	1	1	1	0	1	1	0
byte 2	1	0	0	1	0	1	0	1
byte 3	0	1	1	1	1	1	1	0
byte 4	1	0	0	0	0	0	1	0
byte 5	0	1	1	0	1	0	0	1
byte 6	1	0	0	0	1	0	0	0
byte 7	1	0	1	0	1	1	1	1
byte 8	0	0	0	1	1	0	1	0
byte 9	0	0	0	1	0	0	1	0
parity byte	1	1	0	1	0	0	0	1

Table 6.2

A careful study of the table shows that

- byte 8 (row 8) has incorrect parity (there are three 1-bits)
- bit 5 (column 5) also has incorrect parity (there are five 1-bits).

First, the table shows that an error has occurred following data transmission.

Second, at the intersection of row 8 and column 5, the position of the incorrect bit value (which caused the error) can be found. This means that byte 8 should have been:

0	0	0	1	0	0	1	0
---	---	---	---	---	---	---	---



which would also correct column 5 giving an even vertical parity (now has four 1-bits).

This byte could, therefore, be corrected automatically, as shown above, or an error message could be relayed back to the sender asking them to re-transmit the block of data. One final point; if two of the bits change value following data transmission, it may be impossible to locate the error using the above method.

For example, using the above example again:

0	1	0	1	1	1	0	0
---	---	---	---	---	---	---	---

This byte could reach the destination as:

0	1	1	1	1	1	0	1
---	---	---	---	---	---	---	---

All three are clearly incorrect, but they have retained even parity so will not trigger an error message at the receiving end. Clearly, other methods to complement parity when it comes to error checking of transmitted data are required (such as checksum).

### ***Automatic repeat request (ARQ)***

**Automatic repeat request (ARQ)** is another method to check data following data transmission.

This method can be summarised as follows:

- ARQ uses **acknowledgement** (a message sent to the receiver indicating that data has been received correctly) and **timeout** (the time interval allowed to elapse before an acknowledgement is received).
- When the receiving device detects an error following data transmission, it asks for the data packet to be re-sent.
- If no error is detected, a positive acknowledgement is sent to the sender.
- The sending device will re-send the data package if
  - it receives a request to re-send the data, or
  - a timeout has occurred.
- The whole process is continuous until the data packet received is correct or until the ARQ time limit (timeout) is reached.
- ARQ is often used by mobile phone networks to guarantee data integrity.

---

## **ACTIVITY 6B**

- 1** The following block of data was received after transmission from a remote computer; odd parity was being used by both sender and receiver. One of the bits has been changed during the transmission stage. Locate where this error is and suggest a corrected byte value:

	parity bit	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8
byte 1	0	1	1	0	0	0	1	0
byte 2	1	0	1	1	1	1	1	1
byte 3	1	0	0	1	1	0	0	0
byte 4	0	1	1	0	1	0	1	0
byte 5	1	1	1	0	0	1	1	0
byte 6	1	0	0	0	0	1	0	1
byte 7	0	1	1	1	0	0	0	0
byte 8	0	0	0	0	0	0	0	1
byte 9	0	1	1	1	1	0	1	0
parity byte	1	0	1	1	1	1	0	0

- 2 a) A company is collecting data about new customers and is using an online form to collect the data, as shown below.  
Describe a suitable validation check for each of the four groups of fields.

①	Name of person	<input type="text"/>
②	Date of birth	<input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/>
③	Telephone number	<input type="text"/>
④	Title	<input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/>
	Sex	Female: <input type="checkbox"/> Male: <input type="checkbox"/>

- b) Explain the differences between *validation* and *verification*.  
Why are both methods used to maintain the integrity of data?
- 3 A shopkeeper is populating a database containing information about goods for sale in their shop.  
They are entering the data manually, using both validation and verification to ensure the integrity of the entered data.  
Here is an example of a record:

A21516BX	25	205.50	03334445556
code of the item NXXXXNN (N = letter; X = digit)	number in stock (1-100)	unit cost in dollars	telephone number of supplier of item

- a) Describe how verification could be used to ensure the accuracy of the entered data.  
b) Describe suitable validation checks for all four fields and give examples of data which

would fail your chosen validation methods.

---

## End of chapter questions

- 1 A college is using a local area network (LAN) to access data from a database.
- a) Give **two** security measures to protect the data on the college's computer system. [2]
  - b) Data regarding new students joining the college is being entered into the database. Each student has a 7-digit identification number (ID). A check digit is used as a form of checking to ensure errors have not been made when entering the ID numbers. The verification routine uses modulo-11 with the check digit as the eighth (right-most) digit. The weightings used to calculate the check digit are: 7, 6, 5, 4, 3, 2 and 1; the value 7 is the multiplier for the left-most digit. The ID number is: 1 5 6 3 4 1 2 Calculate the check digit. [4]
  - c) Name and describe **two** validation checks that could be carried out on the student ID number. [4]
- 2 a) Explain what antivirus software is and how it can be used to ensure data security. [4]
- b) Explain how a firewall can be used to identify illegal attempts at accessing a computer system and how they can be used to keep data safe. [4]
- 3 The following block of data was received after transmission from a remote computer. Odd parity was being used by both sender and receiver. One of the bits has been changed during the transmission stage. Locate where this error is and suggest a corrected byte value. [5]

	parity bit	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8
byte 1	0	1	0	0	1	0	0	1
byte 2	0	0	1	1	1	1	1	0
byte 3	1	0	0	0	0	1	1	0
byte 4	1	0	1	1	0	0	0	0
byte 5	1	1	1	0	1	1	1	0
byte 6	0	1	0	0	0	1	0	1
byte 7	1	0	0	1	1	0	1	1
byte 8	1	1	0	1	1	0	0	1
byte 9	0	1	0	1	1	1	1	0
parity byte	0	0	0	0	1	1	0	1

---