UNIVERSITY OF LONDON | INTERNATIONAL PROGRAMMES

Computer security

R. Shipsey

CO3326

2009

Undergraduate study in
Computing and related programmes

This is an extract from a subject guide for an undergraduate course offered as part of the University of London International Programmes in Computing. Materials for these programmes are developed by academics at Goldsmiths.
For more information, see: www.londoninternational.ac.uk

Goldsmiths
UNIVERSITY OF LONDON

This guide was prepared for the University of London International Programmes by:

R. Shipsey

This guide was produced by Sarah Rauchas, Department of Computing, Goldsmiths, University of London.

This is one of a series of subject guides published by the University. We regret that due to pressure of work the author is unable to enter into any correspondence relating to, or arising from, the guide. If you have any comments on this subject guide, favourable or unfavourable, please use the form at the back of this guide.

University of London International Programmes
Publications Office
32 Russell Square
London WC1B 5DN
United Kingdom

www.londoninternational.ac.uk

Published by: University of London

Contents

Preface

This is a level three half-unit subject for the BSc Computing and Information Systems programme. It aims to serve as an introduction to some aspects of computer security. One of the major roles of computers in the 21st century is the generation, storage and communication of data. This data may be sensitive or confidential and unauthorised disclosure – whether accidental or malicious – can cause major problems. There are many examples of the need for security ranging from the private to the global. For example, your medical records and bank details will be stored on a computer somewhere. These details are personal and confidential and you would only want people with the appropriate authority to see this information about you. In many countries there are data protection laws that are supposed to enforce the privacy of any personal data stored for whatever reason by a company or government agency. On a larger scale, it may be of the utmost importance that military secrets are kept confidential and only accessible by those who have the highest security clearance.

This guide will introduce you to some important techniques of computer security including:

- passwords and identification
- encryption
- access controls
- symmetric key cryptosystems
- asymmetric key cryptosystems
- digital signatures
- key management
- hash functions
- internet protocols.

Some characters will appear frequently throughout the subject guide. These are Alice and Bob who are always sending each other messages using the cryptographic protocols and methods described in the subject guide. Alice and Bob just represent any sender and receiver, A and B, but it is often helpful to think of them as people. Another character, the bad-guy, who makes a frequent appearance is Charles who represents a cryptanalyst or hacker.

It is the intention of this subject that you become familiar with the need for security in computing systems, and learn about some particular computer security techniques that are currently in use. You should know how these techniques can be applied and the range of problems where they are applicable. Learning outcomes are given at the start of the chapter. These are a summary of what you should learn from the chapter. However, be aware that the examination questions may cover any of the material in the guide and any additional material covered in the coursework.

Each chapter also includes learning activities and sample examination questions which can be used to test your understanding. Solutions to the examination questions are given at the back of the guide. Some of the activities do not have

solutions either because they are practical exercises, or because there is no 'right' answer.

Examination questions are likely to test material from more than one chapter. Therefore the examination questions at the end of each chapter may be partial rather than complete questions. An entire examination paper with solutions is included at the end of the guide so that you can see the type and level of questions to expect in the examination.

Prerequisites

No particular computing units are required as prerequisites for this subject. However, as this is a third year subject, it is assumed that students have some mathematics and computing experience. Please do not be put off by the mathematics. You need to be able to apply the given mathematics to particular problems in computer security, but you do not need to reproduce any proofs.

The ability to program in java will be useful, particularly when completing the coursework. However, remember that the coursework is designed to test your understanding of computer security and not your programming ability. Examiners are not looking for elegant programming solutions or fancy data input screens.

Method of assessment

There will be one examination lasting 2 hours and 15 minutes at the end of the academic year. This examination consists of five questions; candidates have to answer **any three out of** these **five** questions to achieve full marks. The examination is worth 80 per cent of your final mark for this subject.

You are also expected to complete two coursework assignments. These each count as 10 per cent of your final mark. The coursework assignments typically include some independent research and practical work.

Recommended reading

The subject guide is a complete account of the subject. However, you should be aware that developments in the world of computer security means that the subject guide can never be completely up-to-date. You are therefore advised to access further reading wherever possible to keep abreast of the current state of technology in this field.

Following is a list of books that are recommended. By no means do you need to have copies of all of these books but a selection of your choice would complement the material covered in the subject. Some of the books are available to download free of charge; where appropriate I have given the website addresses.

Anderson, R. ***Security Engineering – The Book.***

Six sample chapters of this book are available to download free of charge from the website – `http://www.cl.cam.ac.uk/$\sim$rja14/book.html`. The first five chapters deal with protocols, passwords, access control and cryptography and are particularly relevant to the subject.

Ferguson, N. and B. Schneier ***Practical Cryptography.*** (New York; Chichester: Wiley, 2003) [ISBN 0471223573](pbk).

As the title suggests this book is a practical guide to choosing and using

cryptographic tools. Some students may prefer this practical approach over the more academic approach of traditional text books.

Gollmann, D. ***Computer Security.*** (Hoboken, NJ: Wiley c2006) second edition [ISBN 0470862939(pbk)].

This book provides useful further reading on identification and authentication and access control, including a chapter on Unix security. There are lots of thought provoking and practical exercises.

Menezes, A., P. Van Oorschot and S. Vanstone ***Handbook of Applied Cryptography.*** (Boca Raton, Fla.; London: CRC, 2001) fifth edition.

Chapters of this book are available to download free of charge from `http://www.cacr.math.uwaterloo.ca/hac/index.html`. This book is intended for professional cryptographers and is the ultimate reference book for cryptography. Thorough details on all aspects of cryptography are given. This book is over 700 pages long and is not recommended for a light read but it is well worth looking at the website.

Pfleeger, C. and S. Pfleeger ***Security in Computing.*** (Upper Saddle River, NJ; London: Prentice Hall 2007) fourth edition [ISBN 9780132390774].

In this book the authors introduce the core concepts of computer security and then identify and assess the threats currently facing programs, operating systems, database systems and networks. Attacks on RSA, SHA and DES are discussed.

Piper, F. and S. Murphy ***Cryptography: A Very Short Introduction.*** (Oxford: Oxford University Press, 2002)[ISBN 9780192803153].

This really is a very short book and a great introduction to cryptography. The ideas behind symmetric key and public key cryptography and their uses are clearly explained.

Schneier, B. ***Secrets and Lies: Digital Security in a Networked World.*** (Wiley, 2004) new edition [ISBN 0471453803].

Computer security from a business world perspective. This book examines the necessity for computer security in the real world. It is written more like a reading book than a textbook and gives an interesting background to the subject of computer security.

Stallings, W. ***Network Security Essentials: Applications and Standards.*** (Upper Saddle River, NJ; Harlow: Pearson Education, 2008) second edition [ISBN 9780132303781(pbk)].

This book covers internet security tools and applications. It includes good descriptions of symmetric cryptosystems including DES, 3DES, AES, IDEA, Blowfish and RC5. There are lots of exercises and test questions.

Chapter 1

Security

1.1 Introduction

In this chapter, we will introduce the notion of computer security, provide some basic definitions and discuss the features that a good security system should provide.

Supplementary reading

Chapter 1 of *Computer security* by Gollmann gives a good introduction to the notion of computer security.
Part 1 of *Secrets and Lies* by Schneier is easy to read and puts computer security into context.

After studying this chapter and the additional reading, you should be able to:

- Recognise the need for computer security and describe how computer security differs from security in the traditional sense.
- Define what is meant by the terms integrity, availability, non-repudiation, authentication, accountability and access control with regards to computer security.
- Discuss the various types of attack that may threaten a security system.
- Discuss the many design considerations to take into account when designing a security system.

1.2 What is security?

In the broadest sense security can be defined as the protection of assets. There are three main aspects to security:

- prevention
- detection
- reaction.

Consider security in the traditional sense – for example, securing your house against burglary. You may take steps to **prevent** a burglary such as locking the doors and windows and installing a burglar alarm. If a burglary did occur, you would be able to **detect** this because items would be missing and the burglar may have caused damage to your house while breaking in. You might **react** to the burglary by reporting it to the police, working out what had been stolen and making an insurance claim.

1.2.1 How is information security different?

Although the definition of security given above still applies when we are talking about information, there are some major differences between traditional security and information security.

- Information can be stolen – but you still have it.

  If a physical item such as a car is stolen then the thief has possession of the car and you no longer have it. If a thief steals a file from your computer, he will probably make a copy of the file for himself and leave the original on your computer. Hence you still have the file but it has also been stolen.
- Confidential information may be copied and sold – but the theft might not be detected.

  If your car has been stolen it is not hard to detect the fact – the car is missing! However as mentioned above, a thief who steals computer files may leave the files on your computer and only copy them for himself. Nothing appears to have changed on your computer so you may not be aware that anything untoward has happened.
- The criminal may be on the other side of the world.

  If a thief steals your car you at least know where he was when he stole the car. However, it is possible to hack into computer systems remotely from anywhere in the world. This makes it very hard to know who is responsible for catching a computer criminal. Is it the police in the country where the computer is, or the police in the country where the criminal is?

Although there is no single definition of computer security, we can say that:

> Computer security deals with the prevention and detection of unauthorised actions by users of a computer system.

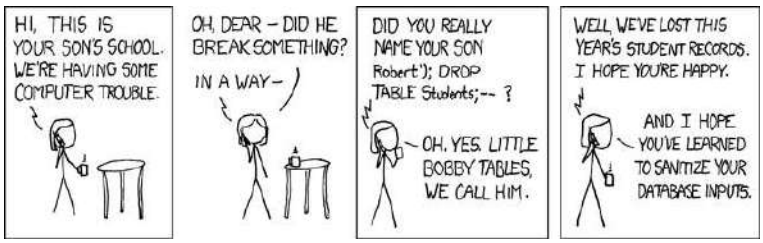
This subject deals with the theory of computer security. You should be aware that unfortunately things that are great in theory do not always work in practice. As Schneier says in *Secrets and Lies*:

> *Theory works best in ideal conditions and laboratory settings. We can design idealised operating systems that are provably secure, but we can't actually build them to work securely in the real world. The real world involves design trade-offs, unseen variables and imperfect implementations.*

Schneier kept a log of 'security events' for the first week of March 2000. He recorded approximately 100 events during this time including hackers launching denial-of-service attacks, leakage of personal data from supposedly secure websites, email worms and viruses, and websites being defaced. Most of these attacks and vulnerabilities were the result of the perpetrator bypassing the security mechanism, or exploiting a weakness in the system such as an overflowing buffer.

Learning activity

The following cartoon by Randall Munroe is taken from xkcd.com.



Do an Internet search on SQL injection vulnerabilities to find out why this is funny!

1.3 Features of a security system

In order to prevent and detect unauthorised actions by its users a good security system should provide (some of) the following features:

- confidentiality
- integrity
- availability
- non-repudiation
- authentication
- access controls
- accountability.

We will look at each of these features in turn. Note that different authors on computer security disagree as to which of these features are the most important. It will depend on the main purpose of the system – is confidentiality paramount or is the prevention of denial of service attacks more important? This will depend on the system in question. For example a computer system which holds personal medical records must certainly provide access controls in order to ensure that personal information does not fall into the wrong hands, and integrity to ensure that the information stored is accurate. Other features such as non-repudiation and availability may not be so important in this case. On the other hand, it is essential for a computer system which transfers money electronically to guarantee non-repudiation and accountability in order to prevent and/or detect dishonest transactions occurring.

In this context, the term *unauthorised* implies not only malicious or criminal, but could also be accidental. For example, a breach of confidentiality arises maliciously if a spy deliberately hacks into a computer and looks at confidential material stored there. It happens accidentally if the material is left out on a desk and is seen by the office cleaner.

1.3.1 Confidentiality

Confidentiality is the prevention of unauthorised disclosure of information.

In other words, confidentiality means keeping information private or safe. Confidentiality may be important for military, business or personal reasons. Confidentiality may also be known as *privacy* or *secrecy*.

1.3.2 Integrity

Integrity is the prevention of unauthorised writing or modification of information.

Integrity in a computer system means that there is an external consistency in the system – everything is as it is expected to be. *Data integrity* means that the data stored on the computer is the same as what is intended.

1.3.3 Availability

Availability is the prevention of unauthorised with-holding of information.

Information should be accessible and usable upon appropriate demand by an authorised user. *Denial of service* attacks are a common form of attack against computer systems whereby authorised users are denied access to the computer system. Such an attack may be orchestrated by the attacker flooding the system with requests until it cannot keep up and crashes. Authorised users are unable to access the system. Consider the damage that such an attack may cause to an electronic commerce site such as an internet shop.

1.3.4 Non-repudiation

Non-repudiation is the prevention of either the sender or the receiver denying a transmitted message.

A computer security system must be able to prove that certain messages were sent and received, who sent the message, who received the message and perhaps what the message said. For example, suppose a dishonest trader sends an electronic message to a stock broker telling him to buy £2,000 worth of shares in CryptoCom. The next day the price of CryptoCom shares soars. The trader now pretends that his original message said to buy £20,000 worth of shares. Conversely if the share price fell he might pretend that the original message said to buy shares in KryptoCom instead. Non-repudiation means that the trader is not able to deny his original message.

Non-repudiation is often implemented by using *digital signatures* (see section 7.3).

1.3.5 Authentication

Authentication is proving a claim – usually that you are who you say you are, where you say you are, at the time that you say it is.

Authentication may be obtained by the provision of a password or by a scan of your retina for example. See Chapter 2 for further methods of authentication.

1.3.6 Access controls

Access controls provide the limitation and control of access to authorised users through identification and authentication.

A system needs to be able to identify and authenticate users for access to data, applications and hardware. In a large system there may be a complex structure determining which users and applications have access to which objects. See Chapter 3 for further details on access control models.

1.3.7 Accountability

Accountability means that the system is able to provide audit trails of all transactions.

The system managers are accountable to scrutiny from outside the system and must be able to provide details of all transactions that have occurred. Audit trails must be selectively kept (and protected to maintain their integrity) so that actions affecting security can be traced back to the responsible party.

Learning activity

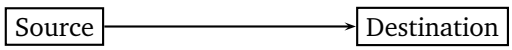
Consider the following scenario and think about the questions at the end.

A student suspects there is a vulnerability on a system in a university public access laboratory. She tests this by trying to exploit the vulnerability. She succeeds, and obtains privileges that she would not normally have. She reports both the hole and her exploiting it to the system staff, who in turn report it to the manager of the laboratory. The manager files charges of breaking into the computing system against the student. The student has to appear before the Student Judicial Authority – she is in trouble!

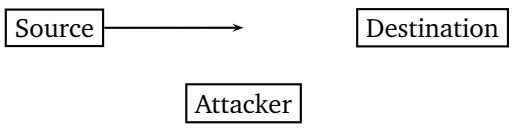
1. Did the student act ethically by testing the system for the security hole before reporting it?
2. Did the manager act ethically by filing charges against the student?
3. The manager told the system staff not to bother fixing the hole, because the action taken by the SJA would deter any further break-ins through the hole. Was the manager's action appropriate?

1.4 Security attacks

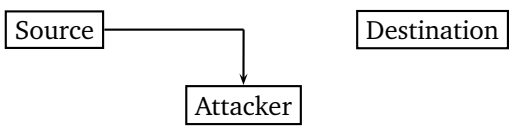
There are a number of ways in which an attacker can disrupt communications. Normally, information goes from the source to the destination.



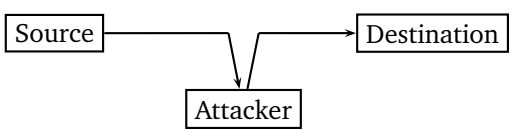
Communication is *interrupted* if the attacker does not allow the information to reach the destination.



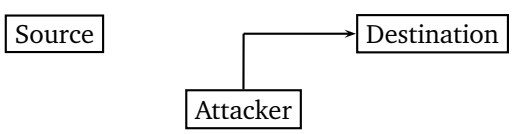
Communication is *intercepted* if the attacker interrupts the communication and receives the source information.



Modification occurs when the attacker intercepts the communication, alters it in some way, and then sends it on to the destination. The attacker intends to deceive the destination into thinking that the modified communication has come directly from the source. This is also known as a *Man-in-the-middle attack*.



An attacker may also make up a communication and send it to the destination pretending that it has come from the source. This is called *fabrication*.



1.5 Security systems

A computer security system is not just a computer package. It also requires security conscious personnel who respect the procedures and their role in the system. For example, an access control system may be rendered worthless by employee Fred Smith who chooses user-name *Fred* and password *Smith* and therefore leaves the system open to abuse by password hackers (see section 2.3.1). Conversely, a good security system should not rely on personnel having security expertise.

1.5.1 Risk analysis

When designing or implementing a computer security system it is very important to bear in mind the level of risk involved and the value of the information that is to be protected. As an illustration, consider that you may be willing to leave £50 in a changing room locker, but you would not be likely to leave £5,000 unattended. You would assess the risk involved before deciding whether to leave the money or not. On the other hand, it would be foolish to pay someone, say £20, to look after your £50, but this might be a good investment in the case of the £5,000 (assuming that you totally trust the person charged with keeping your money safe of course!).

In terms of computer security, the disadvantages of security systems are that they are time consuming, costly, often clumsy, and impede management and the smooth running of the system. *Risk analysis* is the study of the cost of a particular system (in terms of effort and time as well as cost) against the benefits of the system (the level of security offered).

Learning activity

Think about your own circumstances and where you might be affected by computer security. For example, do you use a password or PIN for any purpose? Are there medical or employment records about you? What features of a security system are involved with each example?

Consider a computer system that you are familiar with; for example, perhaps you have a networked system where you work or study, or a PC at home that is used by more than one person. How good is the security of the system? How easy is it to access other people's files or to read their emails? How difficult would it be to add extra security to the system?

1.5.2 Design considerations

There are a number of questions which need to be considered when designing a security system. We will pose five design questions here. See Gollmann, Chapter 1 for further discussion of these questions.

- Does the system focus on the data, operations or users of the system?

  For example, is it more important to have a data focused rule such as: *Only data of type A can be inserted in data box A* or a user focused rule such as: *Only section managers are able to access the information in data box A*?

- What level should the security system operate from?

  The security system may consist of a software package that runs on top of the operating system, such as Norton Internet Security which runs on top of Windows. Alternatively, it may be part of the hardware and have physical control over the data such as where it is stored and how it is manipulated, for example Security Enhanced Linux (SELinux).

- Should the security system be simple or sophisticated?

  As discussed above, there are disadvantages to having a security system, not least in terms of time and cost. The more sophisticated a system the costlier it is likely to be. On the other hand, a system which is too simple may not provide the necessary level of security. It is obviously not a good idea to spend millions

of dollars on a state of the art security system which is to be used to protect data that is not of high importance or value.

- In a distributed system should the security be centralised or spread?

  Should a security manager have ultimate control, for example over access control issues (this will make it easier to achieve a consistent and rigorous approach, but may cause time delays if the security manager has to be applied to for every change of access rights)? Alternatively, should individual users be allowed to choose who has access to their files? See section 3.3.3 for a description of how SELinux implements mandatory access control.

- How do you secure the levels below the level of the security system?

  An attacker may manage to gain access to the operating system and from there make alterations to access control limitations giving themselves access to other parts of the system. The logical access controls of the system may be by-passed by gaining direct access to the physical memory. It is therefore important to ensure that physical security measures are in place as well as the logical computer security mechanisms.

1.6 Security models

Computer security protects the computer system and the data it processes. Success depends on the implementation of security controls designed for the system. A *security model* is a means of formally expressing the rules of the security policy. The model should:

- be easy to comprehend
- be without ambiguity
- be possible to implement
- reflect the policies of the organisation.

Different systems require different models. We will look at the theoretical Bell-LaPadula security model and the practical Unix security model in Chapter 3.

Learning activity

Do an Internet search for some examples of definitions of security concepts. A good starting point is the web site of the UK National Technical Authority for Information Assurance: `http://www.cesg.gov.uk`. Other governments have similar sites, and many major IT companies also have pages discussing security.

1.7 Summary

In this chapter we have introduced some important concepts and definitions regarding computer security. We have discussed features that a computer security system may provide including confidentiality, integrity, availability, non-repudiation, authentication, accountability and access controls. We have also looked at the different ways in which an attacker may threaten a security system including

interrupting, intercepting, modifying and fabricating communications. We have discussed the many design questions that need to be taken into account when designing a security system.

1.8 Learning outcomes

After studying this chapter and the additional reading, you should be able to:

- Recognise the need for computer security and describe how computer security differs from security in the traditional sense.
- Define what is meant by the terms integrity, availability, non-repudiation, authentication, accountability and access control with regards to computer security.
- Discuss the various types of attack that may threaten a security system.
- Discuss the many design considerations to take into account when designing a security system.

1.9 Sample examination questions

Question 1

a) The following are seven features that may be provided by a security system. For each write a sentence describing what is meant by the feature:

   i. confidentiality
   ii. integrity
   iii. availability
   iv. non-repudiation
   v. authentication
   vi. access control
   vii. accountability.

[7]

b) A University department has a file called *exam marks* which contains a list of examination marks indexed by student names in alphabetical order. A student manages to access the exam marks file. The student cannot read the file since it is encrypted. However they can work out the position of their own mark making use of the fact that the students are listed in alphabetical order. The student swaps their mark with that of the student who is always 'top of the class'.

   Write a paragraph discussing which of the security features listed in part a) have been breached.

[5]

Question 2

Three aspects of security are prevention, detection and reaction. Write a paragraph explaining why methods used for the prevention of, detection of and reaction to, theft of physical property may not be appropriate when the crime involves the theft of digital information.

[6]

Chapter 2

Identification and authentication

2.1 Introduction

In this chapter, we will consider why identification is an extremely important aspect of computer security and look at some methods that can be implemented in order to identify computer users. We will also consider the ways in which an identification system can be abused and methods that can be used to minimise the threats to security.

Supplementary reading

Chapter 2 of *Computer security* by Gollmann is a good introduction to identification and authentication.
Do an Internet search on *password crackers*. Analysis of a password cracking program will give you an insight on how fast these programs run and the size of the dictionaries that they use.

After studying this chapter and the recommended reading, you should be able to:

- show familiarity with the concepts of identification and authentication
- describe how a user-name/password system works
- understand that a user may prove their identity using *something they are*, *something they know* or *something they have*
- identify different kinds of threats such as password guessing attacks, password spoofing attacks and attacks on the password file; and give measures that can be used to prevent or detect these attacks
- understand the importance of educating system users so that they do not choose a weak password that may undermine the security of the system
- describe how to protect a password file by using a one-way function to encrypt the passwords
- know that *one-time passwords* can be implemented to reduce the risk of a password being discovered by an attacker
- show familiarity with alternative methods for identification and authentication and know that it is important to assess the security need when choosing which method to apply in a given situation.

2.2 User-names and Passwords

When a computer system has to verify a user's identity, there are two basic questions that have to be asked and answered appropriately. The first is:

Who are you?

The computer system has to establish somehow *who* is trying to gain access to its files. This is usually done by use of a *user-name* which, although probably unique to the user, is not a secret. The user-name is often simply produced using all or part of the user's actual name. For example, the user-name of John Smith might be *JSmith* or *johnsmith*.

When John Smith correctly enters his user-name, the computer can establish, by looking in a database of authorised user-names, that John Smith is an authorised user of the system. However, a second question now has to be asked:

How do I know that you are who you say you are?

The computer must now establish that the person logging into the system as John Smith actually is John Smith. Since the user-name is not a secret, anyone could try to log into the system using the identity of John. The person logging in must somehow prove that they are the genuine John Smith. This is usually done by using a *password*. The password is a secret and is only known to the genuine user John Smith. By entering this secret password, in conjunction with his user-name, John proves to the computer that he is an authorised user and is allowed access to the system.

Thus there are typically two stages in the process of identification.

1. A *user-name* is used to establish identity.
2. A *password* is used to establish authentication of identity.

Your own name is an example of *something you are*. If you apply for a bank loan or a travel visa you will have to prove you are who you say you are by showing, for example, your passport. Your passport is an example of *something you have*. Another example of *something you have* might be a bank card. In order to use the bank card to withdraw money at a cash machine, you do not have to prove who you are, but you do have to prove that you are authorised to use the card. This is done by using *something you know*. In this case the PIN number associated with the bank card. It is possible (although not recommended) for a person to lend their bank card to someone else. However, the second person will only be able to withdraw money using the bank card if they know the correct PIN number. A password is another example of *something you know*. Thus, a person can identify themselves by using *something they are, something they have*, or *something they know*.

2.3 Threats

A basic identification system consists of a database of passwords indexed by user-names. This is called the *password file*. When a user logs into the system, the computer checks that the user-name and password input match an entry in the password file. If a match is found, the process is complete and the user is allowed access to the system. If not, access is denied although the user may be given another chance to enter their user-name and password.

There are various ways in which a user-name/password identification system can be abused. The simplest attacks include the hacker looking over the user's shoulder when they are typing in their password, or finding a written note that the user has

made of their password. In the following sections we will consider some further possible attacks that might be used and defences that can be employed to either prevent, or detect, an attack.

2.3.1 Password guessing

Suppose that a hacker wants to access a system which is protected by a user-name/password identification system. We will assume that the hacker knows the user-name of an authorised user since this information is not generally secret. Therefore if the hacker can guess the user's password he will gain access to the system. There are several ways in which the hacker can find out the user's password. These include:

Guessing using personal knowledge of the user

Many people use passwords which relate to them personally. For example, they may use the name of their spouse or child or pet. They may use their football team or street name or birth date. If the hacker can find out personal information about the user, then they may be able to guess a personal password without too much difficulty.

This attack will fail if the user is careful not to use a password which is personally related to them in any way.

Dictionary searching

Another favourite method of generating easy to remember passwords is for the user to choose a word, usually in their own language. If the hacker cannot directly guess the user's password then he may set up a *dictionary attack*. This means that he will run a computer programme which tries every word in a dictionary as the password of the user until he finds a match.

This attack will fail if the user does not use a word which appears in a dictionary as their password.

Intelligent searching

Some user-name/password systems insist that the user's password contains a mix of letters and numbers. The most common thing for a user (who has not been educated in password security) to do is add a number onto the end of a word. For example, using a password such as banana1. An intelligent dictionary search might try all words with numbers added. Thus if the hacker knows that a particular password system insists that passwords are a minimum of six characters long and must contain at least one number, then the hacker may try all five letter words with each of the digits 0,..,9 attached. Thus apple0, apple1, apple2,....,apple9, apply0, apply1,... and so on would form part of this search.

If this attack does not succeed, the next step might be to capitalise the first letter of each word in the dictionary. Other intelligent dictionary modifications include capitalising each letter of the word in turn, including a number at the front of the

word, including a number in any position in the word or replacing letters which are similar to numbers with that number. For example, replacing the letter l with the number 1 or the letter o with the number 0.

Exhaustive searching

If the user has been clever enough to use a random, meaningless string of characters as their password, then the hacker may have to resort to trying an *exhaustive search attack*. An exhaustive search is similar to a dictionary search, but in the exhaustive case, the computer programme used by the hacker will try every possible combination of permissible characters as the password in order to find a match. Thus if searching for a six character password, the hacker might try aaaaaa, aaaaab, aaaaac, ..., aaaaaz, aaaaa0,...., aaaaa9, aaaaa*, etc. and move systematically through all possible permutations.

This attack will always succeed eventually. Since *every* possible password is tried in turn sooner or later a match will be found. However, there are ways of making an exhaustive search so time consuming for the hacker that it is not successful during the life of the password (i.e. before the exhaustive search is successful the password has been changed). Some password systems insist that the users change their passwords every three months, for example.

Learning activity

A hacker is trying to find a password in order to get access to a computer system. He does not have any personal information about the system users, but he knows that all passwords are at least eight characters long and can contain any upper or lower case letter, digit, or other keyboard character. What kind of attack do you think the hacker should attempt?

2.3.2 Number of passwords

An intelligent attacker will carry out dictionary and intelligent or modified dictionary attacks before attempting an exhaustive search. This is because, although an exhaustive search is bound to succeed eventually and a dictionary search may fail, if it succeeds, the dictionary search is much faster.

Suppose that passwords are six characters long.

If the password is made up only of lower case letters, then there are 26 choices for each character in the password. Hence there are $26^6 = 308,915,776 \approx 3^8$ possible passwords of six lower case letters.

If we include lower and capital letters, there are $52^6 \approx 2^{10}$ possible passwords.

Adding in digits as well, gives a choice out of 62 for each character in the password and there are now $62^6 \approx 5.7^{10}$ possible passwords.

Finally if we allow any keyboard character including ¡ ¿ * & etc. there are approximately 100 different choices for each character in the password and hence there are $100^6 = 10^{12}$ possible passwords.

In general, if a password is n characters long and is made up from an alphabet of A different characters, then there are A^n possible different passwords.

Now suppose that a hacker has written a computer program which can try 10,000 passwords per second.

The hacker has a dictionary file which contains 1,000,000 common six letter words. First he runs a dictionary attack trying every word in his dictionary. This will only take him $1,000,000/10,000 = 100$ seconds to complete.

Making modifications to the dictionary, for example capitalising each word is easy and it only takes the hacker a few more minutes to run modified dictionary searches.

If a dictionary search is not successful then the hacker may try every combination of lowercase letters. This will take $26^6/10,000$ seconds, which is just over 8.5 hours to try every combination of lower case letters.

In comparison, if the hacker attempts an exhaustive search using all 100 possible characters in every combination, it will take $100^6/10,000 = 10^8$ seconds to complete and this is over three years!

Note that the **average time** for a hacker to find a particular kind of password is only **half** the time taken to do a complete search (i.e. if a user has chosen a dictionary word as their password, then the hacker will, on average, only have to search through half of the dictionary in order to find the password). Likewise, on average, a hacker using an exhaustive search will only have to search through half of the possible passwords before finding a match.

Learning activity

1. How many different passwords of lower case letters are there if the password is of length four? How many if the password is of length eight?
2. How many different alphanumeric (any letter or digit) passwords are there if the password is of length four? How many if the password is of length eight?
3. On average, how long will it take a hacker to find a password of length eight which is made up entirely of lowercase letters:
   (a) if the hacker tries only combinations of lowercase letters?
   (b) if the hacker tries all alphanumeric combinations?

   Assume that the hacker can try 10,000 passwords per second.

2.3.3 Password spoofing

A *spoofing attack* is when the user is fooled into giving the hacker their password. Spoofing attacks may be very simple or very sophisticated.

Asking the user

This may sound unlikely, but it is a fact that a lot of people will tell you a password if you can convince them that you need to know it.[1] For example, the hacker may phone the user, and tell them that he is from their office computer staff and that there is a problem with the files. All backed-up information is going to be lost so he needs the user password in order to recover the data. Sometimes an approach as simple as this will work and the user is fooled into giving the hacker their password.

This attack will fail if the user has been educated in computer security and refuses to reveal their password.

Fake log-in screens

A more sophisticated spoofing attack is when the hacker sets up a fake log-in screen which exactly resembles the genuine log-in screen for the system. The user is presented with this log-in screen and unsuspectingly enters their user-name and password. The hacker captures this information and then typically gives the user an error message saying that they have incorrectly typed in their password. The genuine log-in screen is then displayed. The user cannot be sure that they did not make a typing mistake, so they type in their user-name and password again and gain access to the system. The user may have no idea that they have been the victim of a spoofing attack.

This attack will fail if the user notices that there is something wrong with the log-in screen and so does not enter their user-name and password. Some log-in interfaces contain patterns or pictures which are impossible to replicate accurately. The attack can be detected (although not prevented) if the user is informed, at every log-in, of the time of the last failed log-in attempt. After a spoof attack, the user thinks that they had a failed log-in. If when the user successfully logs in, the system does not inform them of this failed log-in then the user is alerted to the fact that they may have been the victim of a spoof attack.

Phishing

Phishing is similar to the above. Communications such as emails or instant messages purporting to be from reliable sites such as eBay, PayPal or online banks direct users to a fake website which looks very like the genuine one. Here the user is asked to input their username, password and perhaps their bank details.

Phishing is a growing problem and attempts to deal with it include legislation, user training, public awareness and technical security measures.

[1] In 2004 an experiment was done by a small group of researchers at a London railway station. They asked the commuters at the station to reveal one of the passwords that they used at work in exchange for a bar of chocolate. Over 70 per cent of the commuters gave a password away! There was no check that the passwords were genuine so wily individuals may have given false passwords. However, it is very likely that many genuine passwords were revealed. You can find more information about this experiment by doing an Internet search on *password for chocolate*.

2.3.4 User and system defences

There are various things that users can do in order to minimise the risk of a hacker getting hold of their password. When a user-name/password system is implemented, it is important that the users are informed of the following measures:

- The user should always set up a password and not leave the password option as blank.
- The user should change the default password.
- The user should change their password frequently.
- The user should not use the same password for all systems.
- When changing a password, the user should not just add a digit onto the end of the old password.
- The user should not choose a password that relates to them personally such as their date of birth or the name of their child.
- The user should not choose a dictionary word as their password.
- The user should not choose a password that is too short.
- The user should choose a password that contains a mix of letters and numbers.
- The user should not write their password down or reveal it to anyone.

Some of these measures can be enforced by the system. Things that the system can do in order to minimise the risk of attack include:

- insist that the user creates a password
- provide the user with a default password
- enforce the user to change the default password at the first log-in
- enforce the user to change their password at frequent intervals (say every three or six months depending on the security need)
- check password choices against a dictionary and reject weak passwords
- insist that passwords contain an alphanumeric mix of characters
- insist that passwords are at least a minimum length (say 6+ characters depending on the security need)
- limit log-in attempts (a maximum of three attempts is usual) after which time the system administrator will have to reset the password for the user
- inform users of each unsuccessful log-in attempt.

Learning activity

Suppose a user has an account on a user-name/password system and that they want to change their password. Write a protocol for a secure procedure that should be followed to enable this.

2.4 Attacking the password file

The *password file*, where the system stores the data for verifying passwords, is very sensitive to attack. In an insecure system, the password file will be a list of passwords indexed by user-name. A hacker with access to this file has potential knowledge of every password. It is therefore essential that the password file is protected.

There are essentially two ways in which the password file can be protected:

- using cryptographic protection
- implementing access control over the password file.

Ideally, the password file should be both encrypted and protected from unauthorised access by the implementation of access controls.

2.4.1 Cryptographic protection

A password file can be encrypted by using a *one-way function*. After encryption, the password file is just a list of garbled characters. Even if a hacker manages to view the file, it will not help him to gain access to the system.

One-way functions

A problem is said to be *one-way* if it is easy to do one way but hard to do in reverse. A non-mathematical example is making a cup of instant coffee. It is easy to put coffee granules, boiling water and milk into a mug and stir them together to make a cup of coffee. However, given a cup of coffee, it is difficult to reverse the operation and retrieve the separate components of milk, coffee granules and water.

In cryptography, the one-way problems used are mathematical functions. A good example of a mathematical one way function is multiplying/factorising.

A one-way function is a function $f : X \rightarrow Y$ which satisfies the following two properties:

- Given x in X it is easy to compute $y = f(x)$ in Y.
- Given y in Y it is very difficult to find an x in X such that $f(x) = y$.

A good example of a mathematical one-way function is multiplying/factorising. It is very easy (especially given a computer or calculator) to multiply together two integers, even if those integers are very large. However, given the resulting number, it is very hard (even with access to a computer) to find the two numbers that were originally multiplied together. In this example, both X and Y are the set of positive integers.

See Chapter 8 for more details on how the factorisation problem can be used as the basis for encryption.

2.4.2 Encrypting the password file

The password file can be protected by using a one-way function $f(x)$ to encrypt the stored passwords as follows:

To create a new user-name/password pair:

- The user inputs their user-name and password x.
- The system computes $f(x)$.
- The password file does not store x but instead stores $f(x)$ indexed by user-name.

To verify a user:

- The system asks for the user-name and password.
- The system computes $f(x')$ where x' is the password entered by the user.
- The system checks to see if there is a match between the $f(x)$ stored for the given user-name and $f(x')$ just computed.
- If $f(x) = f(x')$ then $x = x'$ and the user is verified. If $f(x) \neq f(x')$ then the password entered by the user is incorrect and access to the system is denied.

Attacking an encrypted password file

If a hacker manages to access a password file which has been encrypted using a one-way function, all he will see is the encrypted passwords, indexed by user-names. These encrypted passwords will not enable the hacker to access the system, and the actual passwords are not stored anywhere.

The function used to encrypt the passwords is not usually a secret, so the hacker may try to find an actual password by running a computer program that encrypts a dictionary list or an exhaustive list of passwords and then check to see if the result matches any of the stored encrypted passwords. If a match is found then the hacker has a password and can now gain access to the system.

This type of attack can be thwarted by using a relatively inefficient function to encrypt the passwords. Consider that the hacker may have to encrypt millions of possible passwords before a match is found. If each encryption takes one or two seconds then this will take many days. However, for a genuine individual user a time lapse of a few seconds each time they enter their user-name and password is negligible.

Rainbow tables

If a well known function, such as a secure hashing function, is used to encrypt passwords then pre-computed *rainbow tables* can be used to find passwords very quickly.

A rainbow table is a table that stores the encryption of all possible passwords of a given format. For example, all passwords that are eight characters long and contain lower case letters and digits. These rainbow tables are huge and require a large amount of storage space and initially a lot of time to compile. However, once they

are built they can be searched very quickly to find password matches. These tables are used to retrieve lost user passwords and they are very useful for this purpose. However, in the wrong hands they can obviously be used to find passwords for malicious purposes.

To avoid pre-compiled rainbow tables being used on a security system, the function used to encrypt the passwords should be somehow unique to the system. Pre-compiled tables will therefore not be available. If a user loses or forgets their password it will be irretrievable. An alternative secure method for resetting the lost password to a new value will have to be devised.

2.4.3 Password salting

Password salting is a process used to ensure that all passwords in a system are unique. Most systems insist that all user-names are unique. If a new user tries to create an account with a user-name that is already in use, they will be informed that the user-name is already used and that they should choose another. However, the system cannot inform a new user that the password they have chosen is already in use – that would be a gift for a hacker! Instead, the system adds some *salt* which is another piece of information such as the user-name to all the passwords before encryption. This ensures that every password is unique.

Learning activity

Why is it important that every password should be unique? Suppose a hacker found two encrypted passwords that were the same – how could he use this information?

2.4.4 One-time passwords

Given enough time and attempts, a *static* password (i.e. a password which remains the same) may be accessed by an unauthorised attacker. To counter this, some systems are now making use of *one-time passwords* or OTP. By constantly changing the password, the risk of the password being discovered is greatly reduced. Furthermore, an attacker who does find a password, will only be able to use it to gain access to the system once. The next time the attacker tries to use the password it will be rejected.

One-time passwords typically work in one of three ways.

- A mathematical algorithm is used to generate a new password based on the previous password.
- A time synchronisation protocol is used between the authentication server and the client providing the password.
- A mathematical algorithm is used to create each new password based on a challenge such as a random number chosen by the authentication server and a counter.

To implement a OTP, users generally have a *token* (similar to a small electrical keyring, for example) which generates the passwords either based on a

mathematical algorithm, or if the token contains a clock synchronised with the authentication server, using the current time. Work is currently being done on the use of mobile phones as tokens. This would be practical and cost effective since most Internet users also have a mobile phone.

2.4.5 Alternative methods for authentication

There are many alternative methods used for identification and authentication. Some are used when the risk is low and others where security is of paramount importance. Of course, in general, as the level of security increases so does the cost, so it is sensible to assess the risk before deciding on the level of security required. Alternative methods include:

- Answering a question that only you are likely to know the answer to such as your mother's maiden name or date of birth. This information is not that hard for a hacker to acquire so provides only a low level of security.
- Presentation of something that you have, such as a credit card or passport. These can be forged or stolen but in general are a good means of identification and authentication.
- Use of finger prints, retina patterns or palm prints. This is a high cost solution, but fingerprints, etc. are fairly hard to replicate and are not something that the genuine user can lose or forget! However, a determined attacker with adequate financial resources can replicate these physical attributes leading to a catastrophic failure of a supposedly high security identity system.

2.4.6 Authentication failure

An identification and authentication system can fail in two ways. Firstly, it can accept an unauthorised user. In this case, the security may be too weak. Secondly, it can reject an authorised user. This may be because security is too high. For example, if the system insists that user passwords are 15 characters long and include digits and letters then it is likely that genuine users will forget or mistype their passwords leading to authentication failure.

2.5 Summary

In this chapter, we have discussed how user-name/password systems are used to provide identification and authentication. We have described various attacks that may threaten such systems and measures that can be taken in order to prevent or detect these attacks. We have looked at one-way functions and seen how they can be used to encrypt a password file.

2.6 Learning outcomes

After studying this chapter and the recommended reading, you should be able to:

- show familiarity with the concepts of identification and authentication

- describe how a user-name/password system works
- understand that a user may prove their identity using *something they are*, *something they know* or *something they have*
- identify different kinds of threats such as password guessing attacks, password spoofing attacks and attacks on the password file; and give measures that can be used to prevent or detect these attacks
- understand the importance of educating system users so that they do not choose a weak password that may undermine the security of the system
- describe how to protect a password file by using a one-way function to encrypt the passwords
- know that *one-time passwords* can be implemented to reduce the risk of a password being discovered by an attacker
- show familiarity with alternative methods for identification and authentication and know that it is important to assess the security need when choosing which method to apply in a given situation.

2.7 Sample examination questions

Question 1

After reading a newspaper 'scare story' about password security, Walter has decided to implement strict rules regarding the passwords used by the staff in his company. Walter insists that:

- Staff passwords are of length 15 characters or more.
- Staff change their passwords at least once a week.
- Every password contains a mix of letters and digits.

a) Explain why Walter's password policy is likely to make the password system at his company less rather than more secure.

[3]

b) Write a more suitable password policy explaining the importance of each rule you suggest.

[10]

c) Assuming that a password cracking program can check 10,000 passwords per minute, calculate the average amount of time that it would take to find a password based on the policy that you have written in part b).

[4]

Question 2

a) What two properties are required for a *one-way* function?

[2]

b) Describe how a one-way function can be used to protect password files.

[4]

c) Explain why the one-way function used to protect a password file should not be an efficient function.

[4]

Appendix B

Solutions

B.1 Subject guide activity solutions

B.1.1 Chapter 1

Activity 1.2.1

SQL injection is a technique exploiting security vulnerabilities occurring in the database layer of an application. If user input is either incorrectly filtered or not strongly typed, it may be unexpectedly executed. The key disaster in an injection attack (SQL here but this can also occur in HTML or Javascript) in so-called *cross-site scripting* attacks, is treating untrusted user input directly as program text. There are many real world examples of injection attacks. You may not think the cartoon is funny, but the point is that attacks can come from unexpected sources.

Activity 1.3.7

There is no right or wrong answer to these questions, but the activity demonstrates that security issues are not always straight forward. Note that this is a real life example of a situation that occurs quite frequently.

B.1.2 Chapter 2

Activity 2.3.1

The hacker should start with a dictionary search using words of eight characters. First he uses just lower case letters and next he uses the same words but with the first letter of each word capitalised. If this is not successful, he should next try a modified dictionary search using words of seven characters with a digit appended. He could also try a dictionary search but with letters replaced with similar digits such as the number 0 in place of the letter O and so on. The chances are that at least one user of the system has a weak password. It will be faster to check all the users for weak passwords than to perform an exhaustive search for even a single user. Only if all dictionary modifications fail to bring success should the hacker move onto an exhaustive search.

Activity 2.3.2

1. $26^4 = 456976$ possible passwords of length four. $26^8 \approx 2 * 10^{11}$ possible passwords of length eight using lower case letters only.

2. $62^4 = 14776336$ possible alphanumeric passwords of length four. $62^8 \approx 2.2 * 10^{14}$ possible alphanumeric passwords of length eight.

3. (a) There are 26^8 combinations to try. This will take $26^8/10,000 \approx 242$ days. But on average the hacker will only have to try half of the possible combinations which will take approximately 121 days.

   (b) Now there are 62^8 combinations to try and this will take over 692 years. Even given that on average the hacker will only have to search half of the possible combinations, he will not manage to complete this search!

Activity 2.3.4

Answers will vary but the important point is that the user **must** log into the system using their current password **before** being allowed to change the password.

Activity 2.4.3

If a hacker found that two employees had the same password he could assume that these were not random passwords (since it is unlikely that two employees would happen to have the same random password) and he would be likely to succeed in finding the password using an intelligent search. He could either try to find out what the two employees have in common – perhaps they both have a wife named Alison or perhaps they both support the same football club. This might give him a clue to the password. Or, knowing that the password is probably not a random combination he could try a dictionary search with a high probability of success.

B.1.3 Chapter 3

Activity 3.3.2

i)

	O_1	O_2	O_3	O_4	O_5
S_1	√	√			
S_2	√	√	√	√	
S_3	√		√	√	
S_4		√	√	√	√
S_5			√	√	√
S_6			√	√	

ii) S_1:O_1, O_2
S_2:O_1, O_2, O_3, O_4
S_3:O_1, O_3, O_4
S_4:O_2, O_3, O_4, O_5
S_5:O_3, O_4, O_5
S_6:O_3, O_4