FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA
NIGER STATE, NIGERIA

CENTRE FOR OPEN DISTANCE AND e-LEARNING(CODeL)

B.TECH. COMPUTER SCIENCE PROGRAMME

COURSE TITLE
COMPUTER AND NETWORK SECURITY

COURSE CODE
CPT 326

COURSE CODE
CPT 326

COURSE UNIT
2

Course Coordinator
Bashir MOHAMMAD (Ph.D.)
Department of Computer Science
Federal University of Technology, (FUT) Minna
Minna, Niger State, Nigeria.

Course Development Team

CPT 326: COMPUTER AND NETWORK SECURITY

Subject Matter Experts	Sapun AKSANA (Mrs.) Popapenko NATALYA Computer Science Department FUT Minna, Nigeria.
Course Coordinator	Bashir Mohammad (Ph.D.) Computer Science Department FUT Minna, Nigeria.
ODL Experts	Amosa Isiaka GAMBARI (Ph.D.) Nicholas Ehikioya ESEZOBOR
Instructional System Designers	Oluwole Caleb FALODE (Ph.D.) Bushrah Temitope OJOYE (Mrs.)
Language Editors	Chinenye Priscilla UZOCHUKWU (Mrs.) Mubarak Jamiu ALABEDE
Centre Director	Abiodun Musa AIBINU (Ph.D.) Centre for Open Distance & e-Learning FUT Minna, Nigeria.

CPT 326 Study Guide

Introduction

CPT 326: Computer and Network Security is a 2-credit unit course for students studying towards acquiring a Bachelor of Science in Computer Science and other related disciplines. The course is divided into 5 modules and 15 study units. It will first take a brief review of the concepts The objective of Computer and Network security includes protection of information and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification while allowing the information and property to remain accessible and productive to its intended.

Course Guide

The course guide therefore gives you an overview of what the course; CPT 211 is all about, the textbooks and other materials to be referenced, what you expect to know in each unit, and how to work through the course material. The course guide introduces to you what you will learn in this course and how to make the best use of the material. It brings to your notice the general guidelines on how to navigate through the course and on the expected actions you have to take for you to complete this course successfully. Also, the guide will hint you on how to respond to your Self-Assessment Question(s) and Tutor-Marked Assignments. The course guide therefore gives you an overview of what the course; CPT 327 is all about, the textbooks and other materials to be referenced, what you expect to know in each unit, and how to work through the course material.

What You Will Learn in This Course

The overall aim of this course, CPT 327 is to introduce you to basic concepts of Computer and Network Security in order to enable you to understand the basic elements of use in day to day activities especially, businesses, Sciences and Technology industries.

Course Aim

This course aims to introduce students to the basics, concepts and features of Computer and Network Security. It is believed the knowledge will enable the reader understand and appreciate decision processes especially for critical and cost involving (life, money, etc.) endeavors. It will help the reader to understand the level of disparity between outcome and reality and why we require a decision analysis and support tool to enable us to evaluate, compare and optimize alternatives.

What You Will Learn In This Course

The course, Computer and Network Security, consists of 5 modules. Specifically, the course discusses the following:

1. introduction to Computer and Network Security and Concepts;
2. Security Evaluation, Policies/ Administration;

3. Threats, Risks and Vulnerabilities;
4. Data Security, Security Models, Database Security;
5. Cryptography;
6. Information System Security, Distributed System Security.

Working through This Course

To complete this course, you are required to study all the units, the recommended text books, and other relevant materials. Each unit contains some self-assessment exercises and tutor marked assignments, and at some point, in this course, you are required to submit the tutor marked assignments. There is also a final examination at the end of this course. Stated below are the components of this course and what you have to do.

Course Materials

The major components of the course are:

1. Course Guide
2. Study Units
3. Text Books
4. Assignment File
5. Presentation Schedule

Study Units

There are 15 study units and 5 modules in this course. They are:

Module One	Unit 1: Basics of Computer and Network Security Unit 2: Computer and Network Security Concepts Unit 3: Security Evaluation
Module Two	Unit 1: Vulnerabilities Risk and Threats Unit 2: Data Security Unit 3: Policies/Administration
Module Three	Unit 1: Information System Security Unit 2: Simulation in Security Planning and Management Unit 3: Firewall
Module Four	Unit 1: Isolation Unit 2: Secure Software Unit 3: Cryptography
Module Five	Unit 1: Principles for Secure Systems Unit 2: Cryptography Unit 3: Operating System Based Security

Assignment File

The assignment file will be given to you in due course. In this file, you will find all the details of the work you must submit to your tutor for marking. The marks you obtain for these assignments will count towards the final mark for the course. Altogether, there are tutor marked assignments for this course.

Presentation Schedule

The presentation schedule included in this course guide provides you with important dates for completion of each tutor marked assignment. You should therefore endeavor to meet the deadlines.

Assessment

There are two aspects to the assessment of this course. First, there are tutor marked assignments; and second, the written examination. Therefore, you are expected to take note of the facts, information and problem solving gathered during the course. The tutor marked assignments must be submitted to your tutor for formal assessment, in accordance to the deadline given. The work submitted will count for 40% of your total course mark.

At the end of the course, you will need to sit for a final written examination. This examination will account for 60% of your total score.

Tutor Marked Assignments (TMAs)

There are TMAs in this course. You need to submit all the TMAs. The best 10 will therefore be counted. When you have completed each assignment, send them to your tutor as soon as possible and make certain that it gets to your tutor on or before the stipulated deadline. If for any reason you cannot complete your assignment on time, contact your tutor before the assignment is due to discuss the possibility of extension. Extension will not be granted after the deadline, unless on extraordinary cases.

Final Examination and Grading

The final examination for CPT 326 will be of last for a period of 2 hours and have a value of 60% of the total course grade. The examination will consist of questions which reflect the self assessment exercise and tutor marked assignments that you have previously encountered. Furthermore, all areas of the course will be examined. It would be better to use the time between finishing the last unit and sitting for the examination, to revise the entire course. You might find it useful to review your TMAs and comment on them before the examination. The final examination covers information from all parts of the course.

The Following Are Practical Strategies for Working through This Course

1. Read the course guide thoroughly.
2. Organize a study schedule. Refer to the course overview for more details. Note the time you are expected to spend on each unit and how the assignment relates

to the units. Important details, e.g. details of your tutorials and the date of the first day of the semester are available. You need to gather together all this information in one place such as a diary, a wall chart calendar or an organizer. Whatever method you choose, you should decide on and write in your own dates for working on each unit.

3. Once you have created your own study schedule, do everything you can to stick to it. The major reason that students fail is that they get behind with their course works. If you get into difficulties with your schedule, please let your tutor know before it is too late for help.
4. Turn to Unit 1 and read the introduction and the objectives for the unit.
5. Assemble the study materials. Information about what you need for a unit is given in the table of content at the beginning of each unit. You will almost always need both the study unit you are working on and one of the materials recommended for further readings, on your desk at the same time.
6. Work through the unit, the content of the unit itself has been arranged to provide a sequence for you to follow. As you work through the unit, you will be encouraged to read from your set books.
7. Keep in mind that you will learn a lot by doing all your assignments carefully. They have been designed to help you meet the objectives of the course and will help you pass the examination.
8. Review the objectives of each study unit to confirm that you have achieved them.

a. If you are not certain about any of the objectives, review the study material and consult your tutor.

9. When you are confident that you have achieved a unit's objectives, you can start on the next unit. Proceed unit by unit through the course and try to pace your study so that you can keep yourself on schedule.
10. When you have submitted an assignment to your tutor for marking, do not wait for its return before starting on the next unit. Keep to your schedule. When the assignment is returned, pay particular attention to your tutor's comments, both on the tutor marked assignment form and also written on the assignment. Consult you tutor as soon as possible if you have any questions or problems.
11. After completing the last unit, review the course and prepare yourself for the final examination. Check that you have achieved the unit objectives (listed at the beginning of each unit) and the course objectives (listed in this course guide).

Tutors and Tutorials

There are 8 hours of tutorial provided in support of this course. You will be notified of the dates, time and location together with the name and phone number of your tutor as soon as you are allocated a tutorial group. Your tutor will mark and comment on your assignments, keep a close watch on your progress and on any difficulties, you might encounter and provide assistance to you during the course. You must mail your tutor marked assignment to your tutor well before the due date. At least two working

days are required for this purpose. They will be marked by your tutor and returned to you as soon as possible.

Do not hesitate to contact your tutor by telephone, e-mail or discussion board if you need help. The following might be circumstances in which you would find help necessary: contact your tutor if:

i. you do not understand any part of the study units or the assigned readings;

ii. you have difficulty with the self test or exercise; and

iii. you have questions or problems with an assignment, with your tutor's comments on an assignment or with the grading of an assignment.

You should endeavor to attend the tutorials. This is the only opportunity to have face to face contact with your tutor and ask questions which are answered instantly. You can raise any problem encountered in the course of your study. To gain the maximum benefit from the course tutorials, have some questions handy before attending them. You will learn a lot from participating actively in discussions.

GOODLUCK!

Table of Content

Module 1

Introduction to Computer and Network Security

Unit 1: Basics of Computer and Network Security

Unit 2: Computer and Network Security Concepts

Unit 3: Security Evaluation

Unit 1

Introduction to Computer and Network Security

Content

1.0 Introduction

Security is a subject that has attracted a rapidly growing interest and concern among scholars in the social sciences thereby generating a wide spectrum of issues on the subject. It has nevertheless been attracting new studies, which have brought-out new breakthroughs and findings in security approaches and methodologies. Across the globe, growing numbers of attacks on computer and networks are increasing due the lack of knowledge on how to protect computers and networks. Interest in Computer and Network Security has never been higher than it has been at the beginning of the twenty-first century.

This study unit brings you into the arena of Computer and Network Security by defining the term of the Computer and Network Security and differences between Computer security and Network Security and discussing Importance of Computer and Network Security

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. Define what is meant by the terms Computer and Network Security
ii. The differences between Computer security and Network Security
iii. Describe the Importance of Computer and Network Security

3.0 Learning Contents

3.1 What are Computer and Network Security?

Computer and Network security includes protection of information and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification while allowing the information and property to remain accessible and productive to its intended.

Computer security can be described as an aspect of information security which basically involves putting some measures in place to secure your computers and networks, or simply protect them against infiltration, illegitimate access or corruption of data. In recent time, computers have replaced normal traditional paper system where information is stored in physical files. You go into government ministries; you often see files on the tables or shelves or storage cabinets marked with ‘confidential’.

In Nigeria, almost every file is marked ‘confidential’ and the irony of it, is that any stranger can have access to any of these files because of the carefree attitude of many public servants, official corruption and absence of security consciousness that characterize the nation’s bureaucracy.

It is most disheartening the way important and valuable files get missing, with no trace of recovery. Several pensioners are losing their pension entitlements simply because their files cannot be traced. And such a situation may have security implication on the

State, For instance, a pensioner who has a number of children in tertiary institutions, and is unable to have his entitlements because his file cannot be traced.

If the retiree does not have any other means of survival, to take care of his family, the children will need to fend for themselves. And in the face of job drought, there is the tendency for (some of such) children to be tempted to engage in anti-social activities like 'yahoo business' (online scam), street begging, stealing, to mention a few, thereby constituting a threat to the security of the larger community.

Emotionally, the children of the deprived retiree will tend to develop hatred towards a system that denies their father of his entitlements. This problem may have also denied the poor retiree an opportunity to carry-out his financial obligations to the family. It is only when these children have creative thinking and positive perception that they might not develop negative emotions, which can sometimes lure them into social vices. I could remember a colleague of mine at the university who always complained of hunger and financial incapacitation due to late and irregular payment of the peanuts his father was receiving as pension.

The abominable verification exercise, which pensioners are often subject to, appears to be a source of worry. Coming to the story of the retiree's son, consequently, the guy had to fend for himself, and in the process due to his vulnerability, some of his peers in the neighbourhood introduced him into armed robbery. He was later arrested but many of his university colleagues were astonished and sympathetic too because he was not only homely but also academically brilliant.

The argument here is that if someone could engage in crime due to the inability of his father, to oblige him financially, resulting from late and irregular payment of his father's pension by the government, then what would be the fate of a dependent whose father was not paid at all for the inability of the relevant authorities to trace the file that contains his employment records? The foregoing painted the danger inherent in ineffective storing and misadministration of information. There is no doubt that absence of proper management of information can provoke a security threat to any State.

However, through the use of computer, the long queue and frustration that adorn pensioners' verification exercise would have fizzled out, and every genuine pensioner can collect his/her pension promptly and happily without stress. The traditional means of data management are becoming obsolete.

The files that fill up a whole building can be saved in a small and compact storage device like computer hard disk or removable disks whereby one can store or/and retrieve or/and amend any file timely and easily. How much space do you think will be acquired, if we physically have to open files for ten million people? Here, it may involve occupying a very big building, which may cost several millions of Naira to acquire but with less than five hundred thousand Naira, we can get computers of high storage capacity that can accommodate several hundreds of millions of such files without taking any space beyond that where you mount your desk(s) that supports the

computer(s). Even, one may not need a desk at all, with the use of computing systems like laptops.

If information is vital to the continued existence of any organization, it is pertinent to put in place necessary structures and applications to protect your computer(s) against any infiltration or damage. The emergent revolution in Information and Communication Technology has rendered the traditional means of storing information like paper files moribund whereby computers have gradually replaced them. It is no surprise therefore, that the managements of many organizations in Nigeria have begun to mandate their staff to undergo various computer trainings in order for such to remain relevant in their various work places.

Many public workers can now use computers effectively, as many government job functions are carried out electronically. Many state governments especially Lagos state have computerized their public service, as most services are now being rendered through electronic means. Taxes are currently paid by individuals and corporate bodies through electronic medium.

The Immigration Service in Nigeria has also gone computerized. Processing and issuance of passport is now done electronically, and this appears to be faster and more convenient. However, let us consider a scenario where the details of all those who applied for Nigerian passports in the last two years get erased through malicious attack from intruders or hackers.

Another case is a situation whereby the data system of a commercial bank gets corrupted through virus attack. How do you think the bank will manage to get out of such crisis without any back-up? Considering these two scenarios, you may agree with me that it is important to provide adequate security for our computer system(s).

Essential measures and applications must be put in place to secure our computer system especially as security experts. The nature of the security profession demands for adequate computer security, and we should make enough efforts to protect our computer systems from malicious attack like corruption of data, theft, intrusion, illegal access to data, and damage emanating from natural disaster. In the subsequent part of this segment, we shall be discussing various ways to secure our computers but before we do that, let us quickly explain key concepts of computer and network security in order to stimulate a better appreciation of the subject.

Self-Assessment Exercises (SAE)

1. What do you understand by Computer and Network Security?

Self-Assessment Answers (SAA)

Computer security can be described as an aspect of information security which basically involves putting some measures in place to secure your computers and networks, or simply protect them against infiltration, illegitimate access or corruption of data.

3.2 Differences between Computer Security and Network Security

A. Computer security is a branch of computer technology known as information security as applied to computers and computer networks. The objective of computer security includes protection of information and property from theft, corruption, or natural disaster, while allowing the information and property to remain accessible and productive to its intended users.

   The term computer system security means the collective processes and mechanisms by which sensitive and valuable information and services are protected from publication, tampering or collapse by unauthorized activities or untrustworthy individuals and unplanned events respectively. The strategies and methodologies of computer security often differ from most other computer technologies because of its somewhat elusive objective of preventing unwanted computer behavior instead of enabling wanted computer behavior.

B. Network security consists of the provisions and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. Network security involves the authorization of access to data in a network, which is controlled by the network administrator.

   Users choose or are assigned an ID and password or other authenticating information that allows them access to information and programs within their authority. Network security covers a variety of computer networks, both public and private, that are used in everyday jobs conducting transactions and communications among businesses, government agencies and individuals. Networks can be private, such as within a company, and others which might be open to public access.

   Network security is involved in organizations, enterprises, and other types of institutions. It does as its title explains: It secures the network, as well as protecting and overseeing operations being done. The most common and simple way of protecting a network resource is by assigning it a unique name and a corresponding password.

Self-Assessment Exercises (SAE)

1. What is the differences between Computer Security and Network Security

Self-Assessment Answers (SAA)

Computer security is a branch of computer technology known as information security as applied to computers and computer networks.

While **Network security** consists of the provisions and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources.

3.3 Importance of Computer and Network Security

In today's technologically advanced world, computers play a dominant role. No matter you are at work, in studies at college or school, or just enjoying a leisurely time in your home, it is certain that you may either switch on your computer or any other related state of the art devices. The importance of computer is further enhanced by increased usage of the internet

However, it is important to ensure that while you use the internet you also maintain the good health of the device that has been used for connecting to the web. It is estimated that when you connect your computer network to the internet, you are physically linking your computer to over 50,000 unfamiliar networks.

Despite the fact that it can open avenues to a number of useful applications and provide options for information sharing, many of the private networks consist of certain information that should not be shared with outside users on the web, which may sometimes result in application layer attacks, IP spoofing, DNS cache poisoning, password attacks, and man in the middle attacks.

Even though the internet has made security system of computer networks a top priority in many of the mainstream technology periodicals, most of the people across the globe do not know the importance of network security system. Network security system is an essential component of the configuration as well as network management. Implementation of effective network security provides both physical and information security to paths, links, and databases

Self-Assessment Exercises (SAE)

1. What are the two commonly ways used in authenticating a user on a computer before accessing information:
   i. Username and Password
   ii. Eye scanner
   iii. Thumbprint Scanner
   iv. Voice Reader

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning of computer security and network security. Computer and Network security includes protection of information and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification. You have also learnt the various Importance of Computer and Network Security.

5.0 Summary

i. What you have learnt in this study unit concerns the definition of computer security and network security.

ii. You also have learnt differences between Computer security and Network Security.

iii. The study unit that follows builds upon what you have learnt in this unit by discussing the Importance of Computer and Network Security.

6.0 Tutor-Marked Assignment (TMA)

1. Briefly define what is computer security and network security?
2. Give a brief description of the differences between Computer security and Network Security.

7.0 References/Further Reading

http://en.wikipedia.org/w/index.php?title=Network_security&oldid=512284848

Cisco. (2011). What is network security? Retrieved from cisco.com

http://en.wikipedia.org/w/index.php?title=Computer_security&oldid=513211914

E. Stewart Lee: Essays about Computer Security Cambridge, 1999

Kinkus, J.F. (2002). Science and Technology Resources on the Internet: Computer Security. Issues in Science and Technology Librarianship, No. 36. Available on ://www.istl.org/02-fall/index.html. Retrieved on 30 August, 2009.

Kinkus, J.F. (2002). Science and Technology Resources on the Internet: Computer Security. Issues in Science and Technology Librarianship, No. 36. Available on ://www.istl.org/02-fall/index.html. Retrieved on 30 August, 2009.

Wells, Joe (1996). Virus Timeline. Available on Retrieved on 31 July 2009. ://en.wikipedia.org/ wiki/Microkernel.

://en.wikipedia.org/wiki/Computer security. Retrieved on 1 August 2009.

://en.wikipedia.org/wiki/decryption. Retrieved on 2 August, 2009.

://neohumanism.org/s/se/secure_cryptoprocessor.html. Retrieved on 31 July 2009.

Wells, Joe (1996). Virus Timeline. Available on ://en.wikipedia.org/wiki/Microkernel. Retrieved on 31 July 2009.

://en.wikipedia.org/wiki/Computer security. Retrieved on 1 August 2009.

://en.wikipedia.org/wiki/decryption. Retrieved on 2 August, 2009.

://neohumanism.org/s/se/secure_cryptoprocessor.html. Retrieved on 31 July 2009.

Unit 2

Computer and Network Security Concepts

Content

1.0 Introduction

Security has different aspects and concepts. In the unit we will be discussing the meaning and key concepts of computer security as well as its various approaches. the basic concepts of computer and network security are very important in knowing how to protect your computers and network with the different forms of attacks your computer or network faces. The concepts of computer and network security concepts are absolutely crucial for computer security and network security in the market place and help solve people's daily problems

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. Explain what computer and network security concepts is
ii. Define what is meant by the terms Security by design

3.0 Learning Contents

3.1 Computer network Security Concepts

Anti-Virus Software: There is a conflict among scholars on the originator of anti-virus software but history has it that the first public virus removal task was performed by Bernt Fix in 1987 (Wells, 1996). Anti-virus software is used to detect, prevent and destroy any malware like computer viruses, worms as well as Trojan horses. Apart from protecting the computer against malicious attacks, anti-virus also helps to detect spyware or any other programmes or websites that can constitute security threats to the computer system like virus attack, intrusion and hacking.

It also assists the computer user to identify sites that are not secure, or those designed to perpetrate online scam, through prompt alert and warning of the imminent danger such sites pose to the user and/or system and will advise that the user should not give the details of his/her vital information or better still to close the suspected sites and avoid copying anything from such sites.

Anti-virus software is actually a set of computers programmes designed purposely to identify, block or destroy computer viruses and malicious agents with the aim of protecting the computer from information theft, corruption, hardware damage, to mention a few. There are various types of anti-virus software in the market today, including Norton, AVG, McAfee among others. Due to the way new viruses are generated almost on a daily basis for commercial, strategic or other reasons, it is very pertinent to upgrade the anti-virus software on one's computer from time to time, so that your computing system will be immune from any attack. Apart from virus attack, hackers may try to break into your system to steal, modify or delete some of the files in your system or the whole information contained therein.

Before the advent of the internet, computer viruses were usually spread through floppy disks (diskettes) but now computing systems get infected with viruses and other forms

of malware through the internet. Before now, it was rare for computer to be infected with viruses through the use of recordable or rewritable discs but now the story is different. That is why it is advisable to restrict access of people into your computer and avoid the use of storage facilities like MP3, Flash disk, diskette etc., that have used somewhere else especially at commercial cyber centers without being scanned properly. It is also important to note that it is most appropriate to delete any virus infected files that cannot be repaired by the anti-virus package on your computer system.

Authentication: This involves a technique in which we create password to restrict access to one's computer. In this case, it is only those who can provide the correct password that can be allowed by the computer to gain entry into it. Authentication can also be defined as: the process of identifying an individual, usually based on a username and password. In security systems, authentication is distinct from authorization which is the process of giving individuals access to system object based on their identity.

Sometimes, within the same system, there may be several users, each of whom will create his/her username and password before he/she can access his/her information on the system but the computer will prevent every user from gaining access to another user's information, if he/she fails to provide the correct username and password. There are several ways compute authentication is initiated by the system, and these may identify the users through username or/and password, identification cards, smart cards, as well as biometric systems.

Automated Theorem Proving: This is a verification tool in secure computing system that allows vital algorithms as well as code to be proven mathematically through which the specification of the computer can be met.

Backups: These are simple techniques that help us to secure information in our computing systems by copying and keeping our important files in another storage location like a more secure section in the computer hard drive (less reliable because it goes with the computer in case of theft), MP3 storage device, i-pods, recordable and/or rewritable discs, tapes, flash disk, external hard drive and file hosting on the web.

It is noteworthy to know that there are inherent dangers in keeping files on the web, if adequate security cannot be guaranteed. Highly secure backups are supposed to be very safe and secure storage locations that are not easily susceptible to theft, loss, or destruction resulting from fire, heat, water, or even natural disasters. A good example is a university that has been existing for more than forty years, and experiences fire outbreak that destroys all its academic records. Without any backups, how do you think it will be able to supply the academic records of those who have graduated from the school? You answer may be the same as mine.

Capability and Access Control List: These techniques are usually used to guarantee privilege separation and compulsory access control.

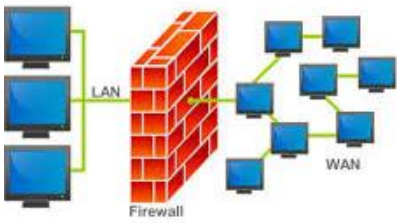
Chain of Trust: This enables us to verify the authenticity of any software loaded on the system, through which we can identify the software certified authentic by the system's designer.

Cryptographic Techniques: These techniques are applied basically to reduce the risk of interception or modification of data whenever data are being exchanged between two or more systems. These techniques involve changing information in such a way that it will remain unreadable to any intruder when data is transmitted from one system to another. In this case, it is only the genuine recipient of the information that can unravel the content of the message while anybody who gains access to such message will not understand the content of the message unless it he/she can break the code to unscramble it, which may be very difficult if the encryption is done very securely.

Encryption: This tool is used to prevent any strange or unintended person from comprehending the content of a message. It involves scrambling of the information in a way that it will be unreadable by anybody other than the real recipient(s) whom the information is meant for. It is the recipient who has the code to unlock the information that can decipher a message. This approach can be used to send secret or very confidential information to several people irrespective of their number in as much they have the cryptographic key, which will enable them to decrypt it.

Decryption: Decryption can be defined as the tool used with the aim of "…reversing an encryption, i.e. the process which converts encrypted data into its original form.

Firewall: This technique helps to protect your system against any malicious attack or illegal access by hackers and intruders whenever you are online. It alerts you whenever it senses any intrusion, so that your computer will not be vulnerable to bugs.



Honey pots: These are computing systems made vulnerable to intrusion and attacks by hackers most times deliberately, to identify areas of defect or vulnerability to effect fixing it.

Mandatory Access Control (MAC): MAC is used to "protect the network and file systems, block users from accessing certain ports and sockets, and more. It is however advisable for optimum use of policy modules, to load many security policy modules at the same time with the aim of providing a multi-layered security setting, and thus "…. a multi-layered security environment, multiple policy modules are in effect to keep security in check" (ibid). The MAC application does not allow the users to change their access codes indiscriminately because all security features are usually

controlled by the access rules presented by the selected security policy modules. Here, it is the system administrator that (absolutely) controls the MAC access rules.

Secure Cryptoprocessor: A secure cryptoprocessor can be said to be “a dedicated Computer for carrying out cryptographic operations, embedded in a packaging with Microkernels: Microkernel can be described as a computer kernel that enables relevant mechanisms, which help to initiate an operating system like low-level address space Management, thread management, and inter-process communication. In a Space Management, thread management, and inter-process communication. In a multiple physical security measures, which give it a degree of resistance”. The essence of a secure cryptoprocessor is to serve as the foundation of securing the system. It is a security sub-system that ensures protection of the system against any intrusion or malware. Some of the examples of secure cryptoprocessor include smart cards and ATM cards. The ways through which secure cryptoprocessor works include:

- Tamper-detecting and tamper-evident containment;
- Automatic zeroization of secrets in the event of tampering;
- Internal battery backup;
- Chain of trust boot-loader which authenticates the operating system before loading it;
- Chain of trust operating system which authenticates application software before loading it; and
- Hardware-based capability registers, implementing a one-way privilege separation model situation whereby multiple privilege levels are offered by the hardware, “the microkernel is the only software executing at the most privileged level (generally referred to as supervisor or kernel mode). Actual operating system services, such as device drivers, protocol stacks, file systems and user interface code are contained in user space” In securing the computing system, microkernels are often used for systems designed for use in high security applications like KeyKOS, EROS and strategic security systems.

Self-Assessment Exercise (SAE)

1. What are the approaches of computer security?

Self-Assessment Answers (SAA)

3.2 Security by Design

The technologies of computer security are based on logic. As security is not necessarily the primary goal of most computer applications, designing a program with security in mind often imposes restrictions on that program's behaviour.

There are 4 approaches to security in computing, sometimes a combination of approaches is valid:

i. Trust all the software to abide by a security policy but the software is not trustworthy (this is computer insecurity).
ii. Trust all the software to abide by a security policy and the software is validated as trustworthy (by tedious branch and path analysis for example).
iii. Trust no software but enforce a security policy with mechanisms that are not trustworthy (again this is computer insecurity).

Trust no software but enforce a security policy with trustworthy hardware mechanisms.

Many systems have unintentionally resulted in the first possibility. Since approach two is expensive and non-deterministic, its use is very limited. Approaches one and three lead to failure. Because approach number four is often based on hardware mechanisms and avoids abstractions and a multiplicity of degrees of freedom, it is more practical. Combinations of approaches two and four are often used in a layered architecture with thin layers of two and thick layers of four.

There are various strategies and techniques used to design security systems. However, there are few, if any, effective strategies to enhance security after design. One technique enforces the principle of least privilege to great extent, where an entity has only the privileges that are needed for its function. That way even if an attacker gains access to one part of the system, fine-grained security ensures that it is just as difficult for them to access the rest.

Furthermore, by breaking the system up into smaller components, the complexity of individual components is reduced, opening up the possibility of using techniques such as automated theorem proving to prove the correctness of crucial software subsystems. This enables a closed form solution to security that works well when only a single well-characterized property can be isolated as critical, and that property is also assessable to math. Not surprisingly, it is impractical for generalized correctness, which probably cannot even be defined, much less proven. Where formal correctness proofs are not possible, rigorous use of code review and unit testing represent a best-effort approach to make modules secure.

The design should use "defence", where more than one subsystem needs to be violated to compromise the integrity of the system and the information it holds. Defence in depth works when the breaching of one security measure does not provide a platform to facilitate subverting another. Also, the cascading principle acknowledges that several low hurdles do not make a high hurdle. So, cascading several weak mechanisms does not provide the safety of a single stronger mechanism.

Subsystems should default to secure settings, and wherever possible should be designed to "fail secure" rather than "fail insecure" (see fail-safe for the equivalent in safety engineering). Ideally, a secure system should require a deliberate, conscious, knowledgeable and free decision on the part of legitimate authorities in order to make it insecure.

In addition, security should not be an all or nothing issue. The designers and operators of systems should assume that security breaches are inevitable. Full audit trails should be kept of system activity, so that when a security breach occurs, the mechanism and extent of the breach can be determined. Storing audit trails remotely, where they can only be appended to, can keep intruders from covering their tracks. Finally, full disclosure helps to ensure that when bugs are found the "window of vulnerability" is kept as short as possible.

3.3 Approaches to Computer Security

a) Security Design: There are several ways through, which security systems are designed. It is paramount to mount effective security strategies that can ensure adequate safety for computing systems. One of such ways is to initiate the principle of least privilege that “where an entity has only the privileges that are needed for its functions” In this case, if an intruder gains access (illegally) into a part of the system, it will be difficult for him/her to access the whole system due to the fine-grained security.

It is therefore advisable to mount a security design that breaks the system into several smaller units with each of the units designed in a less complicated way and may involve the application of automated theorem proving to verify the exactness of key software subsystems. In a situation where formal correctness is missing, careful application of code review and unit testing will be a best- effort approach in securing the modules. Enough efforts should also be made to discourage or eliminate security breaches by the system users, and it is therefore, important to create full audit trails that will assist us in detecting and determining the nature of breach, its degree and the time it occurs. Audit trials should be stored very discretely in such a way that it will be difficult for the intruder to track it to cover up every trace of the illegal entry;

b) Security Architecture: Security Architecture is also a very viable approach to computer and information security. It simply means the design artifacts that explain the state of existing security controls or security countermeasures, showing how they relay with the general information technology architecture. The security controls basically focus on providing platform to enhance the capacity of the system to sustain quality attributes including confidentiality, integrity, availability, accountability and assurance. Security Architecture assists us to identify the areas that demand much security measure, and thus, “if the plan describes a specific solution then, prior to building such a plan, one would make a risk analysis”, but in a situation where “the plan describes a generic high-level design (reference architecture) then the plan should be based on a threat analysis” and

c) Secure Hardware: Computer security can be enhanced through hardware-based security because of the capacity of hardware-based security solutions to present strong resistance against bugs and intrusion. It can deny any intruder or hacker the avenue to read and write access to data.

Self-Assessment Exercise (SAE)

i. What are the approaches of computer security?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning of Computer and Network security concepts. You also have learnt about Security by design. There are 4 approaches to security in computing, sometimes a combination of approaches are very important.

5.0 Summary

1. What you have learnt in this study unit is Computer and Network security concepts.
2. You also have learnt the Security by design.

6.0 Tutor-Marked Assignment (TMAs)

1. What is computer security?
2. List any four concepts of computer security;
3. Discuss any two approaches of computer security

7.0 References/Further Reading

http://en.wikipedia.org/w/index.php?title=Network_security&oldid=512284848
Cisco. (2011). What is network security? Retrieved from cisco.com
http://en.wikipedia.org/w/index.php?title=Computer_security&oldid=513211914
E. Stewart Lee: Essays about Computer Security Cambridge, 1999
://neohumanism.org/s/se/secure_cryptoprocessor.html. Retrieved on 31 July 2009.
://www.freebsd.org/doc/en/books/handbook/mac-understandlabel.html. Retrieved on 31 July 2009.
://www.webopedia.com/TERM/a/authentication.html. Retrieved 1 August 2009.
.infosat.tamu.edu/students/glry/htmossa. Retrieved on 31 August, 2009.
.opensecurityarchitecture.com. Quoted on ://en.wikipedia.org/wiki/ Computer _ security. Retrieved on 31 July 2009.
.securiguard.com/glossary.html. Retrieved on 31 August, 2009.

Unit 3

Security Evaluation

Content

1.0 Introduction

Securing your network is important with the amount of personal data we keep on our computers and the internet is so vast. So, knowing how to evaluate the security treat of is important.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. Processes of Security Evaluation

3.0 Learning Contents

3.1 A Process for Security Evaluation

How do we think about the ways that an adversary might use to penetrate system security or otherwise cause mischief? In this unit, we're going to develop a framework to help you think through these issues. The first place to start, when analyzing a system, is its security goals. What properties do we want the system to have, even when it is under attack? What are we trying to protect from the attacker? Or, to look at it the other way around, what are we trying to prevent? Some common security goals:

1. Confidentiality. Often there is some private information that we want to keep secret from the adversary. Maybe it is a password, a bank account balance, or a diary entry that we don't want anyone else to be able to read. It could be anything. We want to prevent the adversary from learning our secrets.
2. Integrity. If the system stores some information, we might want to prevent the adversary from tampering with or modifying that information.
3. Availability. If the system performs some function, it should be operational when we need it. Consequently, we may need to prevent the adversary from taking the system out of service at an inconvenient time.

For example, consider the database of grade information that we use in this class. One obvious goal is to protect its integrity, so that you can't just give yourself an A+ merely by tampering with the grade database. University rules require us to protect its confidentiality, so that no one else can learn what grade you are getting. We probably also want some level of availability, so that when the end of the semester comes we can calculate the grades everyone will receive. Security goals can be simple, or they can be detailed. Figuring out the set of security goals that must be preserved is an exercise in requirements analysis. They are the specification of it means for a system to be secure.

The security goals are the goals we want to be met even when an adversary is trying to violate them. You can recognize which goals are security goals by asking yourself: if someone were to figure out how to violate this goal, would it be considered a security breach? If the answer is yes, you've found yourself a security goal.

Security goals are highly application-dependent, so it's hard to say much more. Instead, I'll leave you with a famous quote from Young, Boebert, and Kain: A program that has not been specified cannot be incorrect; it can only be surprising. A system without security goals has not been specified and cannot be wrong; it can only be surprising. After you have a set of security goals, the next step is to perform a threat assessment, which asks several questions.

What kind of threats might we face? What kind of capabilities might we expect adversaries to have? What are the limits on what the adversary might be able to do to us? The result is a threat model; a characterization of the threats the system must deal with. When performing a threat assessment, we have to decide how much we can predict about what kind of adversaries we will be facing. Sometimes, we know very well who the adversary is, and we may even know. Their capabilities, motivations, and limitations.

For instance, in the Cold War, the US military was oriented towards its main enemy, the Soviets, and a lot of effort was put into understanding the military capabilities of the USSR (how many battalions of infantry do they have? how effective are their tanks? how quickly can their navy respond to such-and-such threat?). When we know what adversary we will be facing, we can craft a threat model using that knowledge; so that our threat model reflects what that particular adversary is likely to do to us and nothing more.

However, all too often the adversary is not known. In this case, we need to reason more generically about unavoidable limitations that will be placed upon the adversary. As a light-hearted example, physics tells us that the adversary can't go faster than the speed of light. I don't care who they are, they can't violate the rules of physics.

That might be useful to know. More usefully, we can usually look at the design of the system and identify what things an adversary might be in a position to do. For instance, if the system is designed so that secret information is never sent over a wireless network, then we don't need to worry about the threat of eavesdropping upon the wireless communications.

If our system design is such that people might discuss our secrets by phone, we had better include in our threat model the possibility that an insider at the phone company might be able to eavesdrop on our phone calls, or re-route them to the wrong place, or fool people into thinking they are talking with someone legitimate when actually they are speaking with the attacker. A good threat model also specifies what threats we do not care to defend against.

For instance, if I want to analyze the security of my home against burglary, I am not going to worry about the threat that a team of burglars might _y a helicopter over my house and rappel down my chimney to get into the house, Mission Impossible style. There are far easier ways to break into my house, without going to all that trouble. One can often classify adversaries according to their motivation. For instance, consider adversaries who are motivated by financial gain. It's a pretty safe bet that a financially-

motivated adversary is not going to spend more money on the attack than they stand to gain from it. For instance, no burglar is likely to spend thousands of dollars to steal my car radio; my car radio is simply not worth that much. In general, motives are as varied as human nature, and it is a good idea to be prepared for all eventualities.

It's often very helpful to look at the incentives of the various parties. This is probably a familiar principle. Does the local fast food joint make more profit on soft drinks than on the food? Then one might expect some fast food places to take steps to boost sales of soft drinks, perhaps salting its French fries heavily. Do customer service representatives make a bonus if they handle more than a certain number of calls per hour? Then one might expect some representatives to be tempted to cut lengthy service calls short, or to transfer trouble customers to other departments when possible.

Self-Assessment Exercise (SAE)

1. If the system stores some information, we might want to prevent the adversary from tampering with or modifying that information Internal is a definition of a common security goals. What is the goal?

   a. Confidentiality
   b. Availability
   c. Integrity
   d. Incentives

Self-Assessment Answers (SAA)

Do spammers make money from everyone who responds to the spam, while losing nothing from those who didn't wish to receive the spam? Then one can expect that some spammers might be inclined to send their emails as widely as possible, no matter how unpopular it makes them. As a rule of thumb, organizations tend not to act against their own self-interest, at least not too often. Incentives impudence behaviour not always, of course, but frequently enough to help illuminate the motivations of potential adversaries. Incentives are particularly relevant when two parties have opposing interests. When incentives clash, confects often follows.

In this case it is worth looking deeply at the potential for attacks by one such party against the other. If threat assessment sounds difficult, just remember the three W's: Who? How? Why? In other words: Who are the adversaries we might face? How might they try to attack us, and what are their capabilities? Why might they be motivated to attack us, and what are their incentives? Finally, once we have the security goals and a threat model, the last step is to perform a security analysis. The goal of the security

analysis is to see whether there is any attack encompassed within the threat model that can successfully violate the security goals. Security analysis is often highly technical and depends on the details of the particular system being analyzed. We will show you many methods for security analysis in the rest of the course, without saying more now.

An Example

Enough slogans. Let's do an example. Let's analyze the security of my home against intruders. What are my security goals? I'd like to protect the assets in my home from theft or from being tampered with by unauthorized parties (integrity). I'd like my personal safety to be protected; for instance, if someone does break in to steal money, I'd much prefer to know, so that I don't surprise them and get shot. I'd like my house and its contents to remain in full working order whenever I want them (availability). My home is my castle, and I sometimes want a certain measure of privacy when I'm home (confidentiality).

We could probably identify other security goals, but that's more than enough to get us started. Time for the threat assessment. Who am I trying to protect against, and how might they be motivated? A burglar might be motivated by financial gain. A peeping Tom might be motivated by curiosity. Someone who holds a grudge against me might want to secretly get revenge. And so on. What capabilities (tools, skills, knowledge, access, etc.) might an adversary have? What threats might I face? A burglar might have lock picks and the know-how to use them, or a crowbar to smash in the door, or the capability to cut my phone lines. A repairman might have unaccompanied access to the house for a time. Peepers might have binoculars or a telescope.

One of my neighbours might have line-of-sight to my living room window, while another might be blocked by trees. But there are probably some kinds of threats I'm willing to ignore. For instance, I'm not worried that the Navy ships here for Fleet Day are going to start shelling my house. My house has no effective way to defend against such an attack, but I doubt very much that any officer dislikes me enough to throw away his career over it. We could go on for pages, assessing which threats are realistic given the possible motivations of the adversary, and ending up with a detailed threat model.

Finally, the security analysis. What kind of attacks are possible? One can envision all sorts of crazy scenarios. An everyday burglar might smash a window while I'm gone, sneak in and grab some valuables, and run off before they're caught. If I secure the windows, a determined burglar might take a chainsaw to the walls and break in that way (believe it or not, it's happened). A slightly smarter burglar might look under the flowerpot, find the spare house key, and waltz right in. A really sneaky burglar might throw a pebble against my window at 3am each morning, setting off the burglar alarm and bringing the police running, for days at a row, until the police decide to stop paying attention to my obviously unreliable burglar alarm and then the burglar can strike without threat of police response. The neighbour with line-of-sight to my house might use his telescope to peer in through my living window. Someone intent on revenge

might be able to leave unpleasant items on my lawn or depending on my home's security system might be able to smash my windows without my noticing.

An unscrupulous competitor who knows I have an important business meeting with a potential client early tomorrow morning might cut the power to my house in the middle of the night, in hopes that my alarm clock will fail to ring, I will fail to awaken, I will miss my meeting, and I will lose the client. One can come up with some truly outlandish attack scenarios, but you probably get the idea by now. I hope this gives you some feeling for how a security evaluation might go. If the example seems rather trivial, it is probably because home security is already at least somewhat familiar to you. However, when dealing with a complex computer system, it helps to have a framework to structure the security evaluation and that's what the process outlined above is intended to help with.

4.0 Conclusion

In this study unit, you have learnt about the different processes of security evaluation. You should able to know and understand how to analysis and evaluate different threats to you system or network.

5.0 Summary

i. What you have learnt in this study unit concerns the processes of security evaluation.

6.0 Tutor-Marked Assignment (TMA)

1. What do you understand by the term security Evaluation?
2. List and define the 3 common security goals

7.0 References/Further Reading

http://en.wikipedia.org/w/index.php?title=Network_security&oldid=512284848

Cisco. (2011). What is network security? Retrieved from cisco.com

http://en.wikipedia.org/w/index.php?title=Computer_security&oldid=513211914

E. Stewart Lee: Essays about Computer Security Cambridge, 1999

Module 2

Vulnerabilities Risks and Threats

Unit 1: Vulnerabilities Risks and Threats

Unit 2: Data Security

Unit 3: Policies/Administration

Unit 1

Vulnerabilities Risks and Threats

Content

1.0 Introduction

Understanding the kind and threats and risk your network could encounter is so important. From understand how vulnerability of your system with guide you to how to protect your system better.

This study unit introduces you on vulnerability, threats and risks. This unit also will show u how to identify Vulnerabilities and Risks on Your Network.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. What is Vulnerability, Threats, Risks?

ii. Identifying Vulnerabilities and Risks on Your Network

3.0 Learning Contents

3.1 Vulnerability

A computer security Vulnerability is a **'hole'in** any **software**, **operating system** or **service**, that can be exploited by web criminals for their own benefits. There is a difference between bugs and vulnerabilities, though both are result of programming flaws. A bug may or may not be dangerous for the product. A vulnerability, however, has to patch as soon as possible, as web criminals can take advantage using the vulnerability.

A bug fix can wait as if it does not help web criminals in compromising the product. But a vulnerability, which is a bug that is open to people, can use it to gain unauthorized access to the product and via the product, to different parts of a computer network, including the database.

Thus, a vulnerability has to be addressed urgently, to prevent exploitation of the software or these services. Some of the recent example of Vulnerabilities are Shellshock or BASH vulnerability, Heartbleed and the POODLE vulnerability.

Microsoft defines a Vulnerability as follows:

A security vulnerability is a weakness in a product that could allow an attacker to compromise the integrity, availability, or confidentiality of that product.

It then breaks down the definitions to make it easier to understand it – and lays down 4 conditions for anything to be classified as a vulnerability:

1. **A weakness in a product** refers to any type of weakness and we can term it overall as a bug. As explained above, a vulnerability is definitely a bug but a bug need not be a vulnerability all the times. A lower cipher strength can be weakness of the product. An unwarranted additional code may be a weakness that makes the product longer to respond. There can be many examples.

2. **Integrity of Product means trustworthiness.** If the above weakness is bad enough that it allows exploiters to misuse it, the product is not integrated enough. There is a question mark as to how safe the product is.
3. **Availability of the Product** again refers to the weakness whereby an exploiter can take over the product and deny access to it for authorized users.
4. **Confidentiality of the Product** is keeping the data secure. If the bug in the system allows for unauthorized people to collect others' data, it is termed vulnerability.

Thus, according to Microsoft, a bug has to meet the above four criteria before it can be termed as a vulnerability. A normal bug fix can be created at ease and may be released with service packs. But if the bug meets the above definition, it is a vulnerability. In such a case, a security bulletin is issued and a patch is made available as soon as possible.

3.1.1 What is a Zero-Day Vulnerability?

A zero-day vulnerability is previously unknown vulnerability in a software, which gets exploited or attacked. It is called zero-day, since the developer has had no time to fix it, and no patch has been released for it yet. Using the Enhanced Mitigation Experience Toolkit on Windows is a great way to protect your system against zero-day attacks.

3.1.2 Secure & Protect yourself against Vulnerabilities

The best way to protect yourself against vulnerabilities is to ensure that you install updates and security patches for your operating system as soon as they are released, as well as ensure that you have the latest version of any software installed on your Windows computer. If you have Adobe Flash and Java installed on your computer, you will have to take particular care to ensure that you install their updates as soon as possible, as they are among the most vulnerable software and are a commonly used vector – and vulnerabilities in them are being discovered every other day. Also ensure that you install a good Internet security software. Most such software includes a Vulnerability Scan feature that scans your operating system and software and helps you fix them in a click.

There are several other software that can scan your computer for vulnerabilities in your operating system and installed software. Secunia Personal Software Inspector, SecPod Saner Free, Microsoft Baseline Security Analyzer, Protector Plus Windows Vulnerability Scanner, Malwarebytes Anti-Exploit Tool and Exploit Shield are some of the better known free tools available for Windows. These tools will scan your computer for operating system vulnerabilities & unprotected fragments of program code, and typically detect vulnerable and outdated software and plug-ins which expose your otherwise updated & secure Windows computer to malicious attacks.

3.2 Threats

A threat is an action that is taken by a threat agent. A threat agent will exercise and exploit against vulnerability. It's kind of how the terms are laid out and it’s kind of important that you understand what these terms are because we'll see them again when we talk about risk. Now, some information that we need to know about threats: threats don't have to be just the hacker that attacks your system.

A threat can be man-made for sure or it can be natural. A threat could also be a fire or a tornado for example. Threats can be intentional or accidental. Intentional might mean a disgruntled employee that erases a folder of privileged data or data that would be hard to replace.

Accidental might be a threat that would be classified as an employee that simply because they don't know what they're doing, they accidentally erase something that's important. So threats can be humans but they also can be technology.

Here’s a quick explanation of some of the common security threats you may come across:

Malware: Malware is short for “malicious software.” Wikipedia describes malware as a term used to mean a “variety of forms of hostile, intrusive, or annoying software or program code.” Malware could be computer viruses, worms, Trojan horses, dishonest spyware, and malicious rootkits—all of which are defined below.

Computer virus: A computer virus is a small piece of software that can spread from one infected computer to another. The virus could corrupt, steal, or delete data on your computer—even erasing everything on your hard drive. A virus could also use other programs like your email program to spread itself to other computers.

Rogue Security Software: Have you ever seen a pop-up window that advertises a security update or alert? It appears legitimate and asks you to click on a link to install the “update” or “remove” unwanted malicious software that it has apparently detected. This could be rogue security software designed to lure people into clicking and downloading malicious software. Microsoft has a useful webpage that describes rogue security software and how you can protect yourself.

Trojan Horse: Users can infect their computers with Trojan horse software simply by downloading an application they thought was legitimate but was in fact malicious. Once inside your computer, a Trojan horse can do anything from record your

passwords by logging keystrokes (known as a keystroke logger) to hijacking your webcam to watch and record your every move.

In February 2010, a Guardian Analytics and Ponemon Institute study of 500 small businesses in the U.S. found that 55 percent of respondents experienced a fraud attack in the last 12 months. The study reports that “well-funded cyber criminals executed a full-scale assault on authentication, leveraging widespread infection of end-user computers with banking Trojans to sneak into online banking accounts completely undetected.”

Malicious spyware: Malicious spyware is used to describe the Trojan application that was created by cybercriminals to spy on their victims. An example would be key logger software that records a victim’s every keystroke on his or her keyboard. The recorded information is periodically sent back to the originating cybercriminal over the Internet. Keylogging software is widely available and is marketed to parents or businesses that want to monitor their kids’ or employees’ Internet usage.

Computer Worm: A computer worm is a software program that can copy itself from one computer to another, without human interaction. Worms can replicate in great volume and with great speed. For example, a worm can send copies of itself to every contact in your email address book and then send itself to all the contacts in your contacts’ address books.

Because of their speed of infection, worms often gain notoriety overnight infecting computers across the globe as quickly as victims around the world switch them on and open their email. This happened with the Conficker worm (also known as Downadup), which, in just four days, had more than tripled the number of computers it infected to 8.9 million.

Botnet: A botnet is a group of computers connected to the Internet that have been compromised by a hacker using a computer virus or Trojan horse. An individual computer in the group is known as a “zombie” computer.

The botnet is under the command of a “bot herder” or a “bot master,” usually to perform nefarious activities. This could include distributing spam to the email contact addresses on each zombie computer, for example. If the botnet is sufficiently big in number, it could be used to access a targeted website simultaneously in what’s known as a denial-of-service (DoS) attack. The goal of a DoS attack is to bring down a web server by overloading it with access requests. Popular websites such as Google and Twitter have been victims of DoS attacks.

Spam: Spam in the security context is primarily used to describe email spam — unwanted messages in your email inbox. Spam, or electronic junk mail, is a nuisance as it can clutter your mailbox as well as potentially take up space on your mail server. Unwanted junk mail advertising items you don’t care for is harmless, relatively speaking. However, spam messages can contain links that when clicked on could go to a website that installs malicious software onto your computer.

Phishing: Phishing scams are fraudulent attempts by cybercriminals to obtain private information. Phishing scams often appear in the guise of email messages designed to appear as though they are from legitimate sources. For example, the message would try to lure you into giving your personal information by pretending that your bank or email service provider is updating its website and that you must click on the link in the email to verify your account information and password details.

Rootkit: According to TechTarget, a rootkit is a collection of tools that are used to obtain administrator-level access to a computer or a network of computers. A rootkit could be installed on your computer by a cybercriminal exploiting a vulnerability or security hole in a legitimate application on your PC and may contain spyware that monitors and records keystrokes.

Rootkits gained notoriety when, in 2005, a security blogger discovered that a copy-protection tool inside music CDs from Sony BMG Music Entertainment was secretly installing a rootkit when users copied the CD onto their computers. At the time, security expert Bruce Schneier warned that the rootkit could allow a hacker to "gain and maintain access to your system and you wouldn't know it.

3.3 Risks

Risk is something we do every day to determine what we need, what protections we need to put in place if a certain threat happened and exposed a or exploited a vulnerability. Risk is essentially the likelihood that vulnerability will be exploited. Now, risk can be reduced or we say mitigated and most of the time mitigated means to reduce the risk, but it also can mean to transfer the risk or eliminate the risk. Mitigation basically reduces the likelihood that a threat agent is able to carry out an exploit against any particular vulnerability. We can mitigate vulnerability in a system for example.

3.3.1 Types of attack

Classes of attack might include passive monitoring of communications, active network attacks, close-in attacks, exploitation by insiders, and attacks through the service provider. Information systems and networks offer attractive targets and should be resistant to attack from the full range of threat agents, from hackers to nation-states. A system must be able to limit damage and recover rapidly when attacks occur.

There are five types of attack:

Passive Attack

A passive attack monitors unencrypted traffic and looks for clear-text passwords and sensitive information that can be used in other types of attacks. Passive attacks include traffic analysis, monitoring of unprotected communications, decrypting weakly encrypted traffic, and capturing authentication information such as passwords. Passive interception of network operations enables adversaries to see upcoming actions. Passive attacks result in the disclosure of information or data files to an attacker without the consent or knowledge of the user.

Active Attack

In an active attack, the attacker tries to bypass or break into secured systems. This can be done through stealth, viruses, worms, or Trojan horses. Active attacks include attempts to circumvent or break protection features, to introduce malicious code, and to steal or modify information. These attacks are mounted against a network backbone, exploit information in transit, electronically penetrate an enclave, or attack an authorized remote user during an attempt to connect to an enclave. Active attacks result in the disclosure or dissemination of data files, DoS, or modification of data.

Distributed Attack

A distributed attack requires that the adversary introduce code, such as a Trojan horse or back-door program, to a “trusted” component or software that will later be distributed to many other companies and users Distribution attacks focus on the malicious modification of hardware or software at the factory or during distribution. These attacks introduce malicious code such as a back door to a product to gain unauthorized access to information or to a system function at a later date.

Insider Attack

An insider attack involves someone from the inside, such as a disgruntled employee, attacking the network Insider attacks can be malicious or no malicious. Malicious insiders intentionally eavesdrop, steal, or damage information; use information in a fraudulent manner; or deny access to other authorized users. No malicious attacks typically result from carelessness, lack of knowledge, or intentional circumvention of security for such reasons as performing a task

Close-in Attack

A close-in attack involves someone attempting to get physically close to network components, data, and systems in order to learn more about a network Close-in attacks consist of regular individuals attaining close physical proximity to networks, systems, or facilities for the purpose of modifying, gathering, or denying access to information. Close physical proximity is achieved through surreptitious entry into the network, open access, or both.

One popular form of close in attack is social engineering in a social engineering attack, the attacker compromises the network or system through social interaction with a person, through an e-mail message or phone. Various tricks can be used by the individual to revealing information about the security of company. The information that the victim reveals to the hacker would most likely be used in a subsequent attack to gain unauthorized access to a system or network.

Phishing Attack

In phishing attack, the hacker creates a fake web site that looks exactly like a popular site such as the SBI bank or PayPal. The phishing part of the attack is that the hacker then sends an e-mail message trying to trick the user into clicking a link that leads to the fake site. When the user attempts to log on with their account information, the

hacker records the username and password and then tries that information on the real site.

Hijack Attack

In a hijack attack, a hacker takes over a session between you and another individual and disconnects the other individual from the communication. You still believe that you are talking to the original party and may send private information to the hacker by accident.

Spoof Attack

In a spoof attack, the hacker modifies the source address of the packets he or she is sending so that they appear to be coming from someone else. This may be an attempt to bypass your firewall rules.

Buffer Overflow

A buffer overflow attack is when the attacker sends more data to an application than is expected. A buffer overflow attack usually results in the attacker gaining administrative access to the system in a command prompt or shell.

Exploit Attack

In this type of attack, the attacker knows of a security problem within an operating system or a piece of software and leverages that knowledge by exploiting the vulnerability.

Password Attack

An attacker tries to crack the passwords stored in a network account database or a password-protected file. There are three major types of password attacks: a dictionary attack, a brute-force attack, and a hybrid attack. A dictionary attack uses a word list file, which is a list of potential passwords. A brute-force attack is when the attacker tries every possible combination of characters.

3.4 Identifying Vulnerabilities and Risks on Your Network

Understand common attacks. Attacks on and within your network come in many different varieties. Many times, the attackers do not even know who they are attacking, but there are instances of networks or organizations that are specifically targeted. Learning the different methods used to compromise computers and networks will give you the necessary perspective to proceed.

Inventory your vulnerabilities. Establish a full list of potential vulnerabilities. Take special care to identify anything unknown about your network. For example, a library new to network security might think they have a “firewall” while they might just have a router provided by their ISP. For more on this topic, read 10 Steps to Creating Your Own IT Security Audit.

Use vulnerability scanning tools. Many tools exist to check the existing security state of your network. These tools check for open ports, unpatched software and other

weaknesses. Some of these programs focus on a specific machine, while others can scan your entire network. Microsoft offers one such tool, the Microsoft Baseline Security Analyzer. This tool checks for updates and common configuration errors for Microsoft products. Nmap is another popular, free scanning program. For more about Nmap and other vulnerability scanning tools, see Further Resources.

Assess the risks. The various vulnerabilities on your network represent potential costs — time, money and assets — to your library. These costs, along with the chance someone will exploit these vulnerabilities, help determine the level of risk involved. Risk assessment is a combination of both quantifying (the cost of the threat) and qualifying (the odds of the attack). Each library will have to determine its own tolerance for risk depending on the situation. Some examples are provided here.

i. **Patron information:** Having your patron data compromised is unacceptable for any library. You would need to design your network and implement security to minimize this risk. While you can almost never remove risk completely, you can reduce risk to very low levels.

ii. **Slow Internet connection:** A library shares an Internet connection between public networks and staff networks. Since the cost of adding another Internet connection, increasing the speed of the current connection or purchasing complex network monitoring equipment might be too prohibitive, the library has a higher tolerance for a periodically slow Internet connection. Another library hosts its own Web site, online catalogue and email server, which require a more stable Internet connection, so a much lower tolerance for this risk exists.

Self-Assessment Exercise (SAE)

1. How do you identify vulnerabilities and risk on your network?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meanings of Vulnerability, Threats, and Risks. You have also learnt how to Identifying Vulnerabilities and Risks on Your Network.

5.0 Summary

i. What you have learnt in this study unit meanings of Vulnerability, Threats, Risks.

ii. You also have learnt how to Identifying Vulnerabilities and Risks on Your Network.

6.0 Tutor-Marked Assignment (TMAs)

1. What are the meanings of Vulnerability, Threats, Risks?
2. List and define 3 ways of Identifying Vulnerabilities and Risks on Your Network.

7.0 References/Further Reading

http://en.wikipedia.org/w/index.php?title=Network_security&oldid=512284848

Cisco. (2011). What is network security? Retrieved from cisco.com

http://en.wikipedia.org/w/index.php?title=Computer_security&oldid=513211914

E. Stewart Lee: Essays about Computer Security Cambridge, 1999

Unit 2

Data Security

Content

1.0 Introduction

Understanding the Broad range of information security controls to protect databases in very important in computer and network security. Know the different layers in a database with help to know the most important parts with in a database.

This study unit brings you into the Introduction to Database Security. This unit will also introduce you to Database Layers.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. Explain why this is an interesting time to be studying entrepreneurship

ii. Define what is meant by the terms entrepreneur and entrepreneurship

iii. Describe the characteristics of an entrepreneur

3.0 Learning Contents

3.1 Introduction to Database Security

Database security concerns the use of a broad range of information security controls to protect databases (potentially including the data, the database applications or stored functions, the database systems, the database servers and the associated network links) against compromises of their confidentiality, integrity and availability. It involves various types or categories of controls, such as technical, procedural/administrative and physical. Database security is a specialist topic within the broader realms of computer security, information security and risk management.

Security risks to database systems include, for example:

1. Unauthorized or unintended activity or misuse by authorized database users, database administrators, or network/systems managers, or by unauthorized users or hackers (e.g. inappropriate access to sensitive data, metadata or functions within databases, or inappropriate changes to the database programs, structures or security configurations);
2. Malware infections causing incidents such as unauthorized access, leakage or disclosure of personal or proprietary data, deletion of or damage to the data or programs, interruption or denial of authorized access to the database, attacks on other systems and the unanticipated failure of database services;
3. Overloads, performance constraints and capacity issues resulting in the inability of authorized users to use databases as intended;
4. Physical damage to database servers caused by computer room fires or floods, overheating, lightning, accidental liquid spills, static discharge, electronic breakdowns/equipment failures and obsolescence;

5. Design flaws and programming bugs in databases and the associated programs and systems, creating various security vulnerabilities (e.g. unauthorized privilege escalation), data loss/corruption, performance degradation etc.; and
6. Data corruption and/or loss caused by the entry of invalid data or commands, mistakes in database or system administration processes, sabotage/criminal damage etc.

3.2 Database Layers

A database layer is an application programming interface which unifies the communication between a computer application and databases such as SQL server, DB2, MySQL, PostgreSQL, Oracle or SQLite. Traditionally, all database vendors provide their own interface tailored to their products, which leaves it to the application programmer to implement code for all database interfaces he or she would like to support.

Database abstraction layers reduce the amount of work by providing a consistent API to the developer and hide the database specifics behind this interface as much as possible. There exist many abstraction layers with different interfaces in numerous programming languages. If an application has such a layer built in, it is called database-agnostic.

Many layers and types of information security control are appropriate to databases, including:

i. Access control
ii. Auditing
iii. Authentication
iv. Encryption
v. Integrity controls
vi. Backups
vii. Application security

Traditionally databases have been largely secured against hackers through network security measures such as firewalls, and network-based intrusion detection systems. While network security controls remain valuable in this regard, securing the database systems themselves, and the programs/functions and data within them, has arguably become more critical as networks are increasingly opened to wider access, in particular access from the Internet.

Furthermore, system, program, function and data access controls, along with the associated user identification, authentication and rights management functions, have always been important to limit and in some cases log the activities of authorized users and administrators. In other words, these are complementary approaches to database security, working from both the outside-in and the inside-out as it were.

Many organizations develop their own "baseline" security standards and designs detailing basic security control measures for their database systems. These may

reflect general information security requirements or obligations imposed by corporate information security policies and applicable laws and regulations (e.g. concerning privacy, financial management and reporting systems), along with generally-accepted good database security practices (such as appropriate hardening of the underlying systems) and perhaps security recommendations from the relevant database system and software vendors.

The security designs for specific database systems typically specify further security administration and management functions (such as administration and reporting of user access rights, log management and analysis, database replication/synchronization and backups) along with various business-driven information security controls within the database programs and functions (e.g. data entry validation and audit trails). Furthermore, various security-related activities (manual controls) are normally incorporated into the procedures, guidelines etc. relating to the design, development, configuration, use, management and maintenance of databases.

Self-Assessment Exercise (SAE)

1. What is a Database Layer?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning and understanding of database security. The important of know how to protecting your database. This unit will also introduce you to Database Layers.

5.0 Summary

i. You have learnt about the meaning and understanding of database security.

ii. You also have learnt about Database Layers.

6.0 Tutor-Marked Assignment (TMA)

1. List and explain 4 Security risks to database system?
2. Explain the term Database Layers and list 5 different layers you know.

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html
http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf
Dark Reading - Tech Insight: Database Activity Monitoring

Unit 3

Policies/Administration

Content

1.0 Introduction

For systems, the security policy addresses constraints on functions and flow among them, constraints on access by external systems and adversaries including programs and access to data by people.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. the definition of Security policy;
ii. how to Create Security Policies; and
iii. understanding the Nigerian Policies.

3.0 Learning Contents

3.1 Security Policy

Security policy is a definition of what it means to be secure for a system, organization or other entity. For an organization, it addresses the constraints on behavior of its members as well as constraints imposed on adversaries by mechanisms such as doors, locks, keys and walls. For systems, the security policy addresses constraints on functions and flow among them, constraints on access by external systems and adversaries including programs and access to data by people

In business, a security policy is a document that states in writing how a company plans to protect the company's physical and information technology (IT) assets. A security policy is often considered to be a "living document", meaning that the document is never finished, but is continuously updated as technology and employee requirements change. A company's security policy may include an acceptable use policy, a description of how the company plans to educate its employees about protecting the company's assets, an explanation of how security measurements will be carried out and enforced, and a procedure for evaluating the effectiveness of the security policy to ensure that necessary corrections will be made.

3.2 Creating Security Policies

Security policies provide the road map for how to protect your network. These guidelines include the acceptable use of technical resources, the security requirements and why a particular policy exists. Without the clear guidelines from a security policy, your library runs the risk of inconsistent implementation of security. The process of creating a security policy provides a unique opportunity to understand the details of your organization’s network. Why Create Security Policies?

1. **Linking network security to your library’s mission.** Closing every security loophole and blocking every vector of attack would likely render your network unusable for patrons. Finding the right balance between accessibility and security requires a conversation about the library’s mission and audience.

2. **Consistent implementation**. A security policy minimizes the risks created by an inconsistent application of security principles.
3. **Buy wisely.** A security policy can guide your technology purchases.
4. **React and recover.** A security policy will outline the steps needed to respond to a breach in security or critical system failure.

Key Actions

1. **Assemble the appropriate team.** Security policies impact every aspect of an organization. Network security involves rather technical subject matter. However, policy-makers, budgeting, end-users and technical experts all must be included. By involving the appropriate cross-section of the organization, the proper security team increases the transparency and understanding of the decision-making process.
2. **Incorporate the overall mission and goals of the library.** The security policy must abide by existing policies, rules and regulations, as well as promote the overall mission. Any existing computer use policies should be included.
3. **Utilize a policy template.** Find an existing security policy for your library, or search for a library with similar needs and ask to see a copy of their security policy. There are also many security policy templates available online.

3.3 Nigerian Policies

18 May 2010 - Nigeria's Information and Communications Technology for Development, ICT4D plan document was launched in Abuja, Nigeria on May 18, 2010 at the eNigeria 2010 summit organized by the National Information Technology Development Agency (NITDA). The National Information Technology Development Agency (NITDA) in collaboration with the United Nations Economic Commission for Africa (UNECA) coordinated the development of the National ICT4D Strategic Action Plan. The Strategic Action Plan provides concrete implementation strategies for a period of 5 years for the key sectors “health, education, infrastructure, human resource development, Agriculture, Legal/Regulations, private sector/industry, media/community, amongst others - as part of an integrated approach to achieving national development within the context of the Federal Government of Nigeria Seven Point Agenda, the National Economic Empowerment Development Programme (NEEDS) and various socio-economic development programmes and initiatives.

The seven-point agenda includes: Power and energy, Food and security, Wealth creation, Transport sector, Land reforms, Security. Education. The broad goals of NEEDS, NEEDSII and the Seven-Point Agenda are poverty reduction, wealth creation and employment generation through the development of an enabling environment for growth.

General Objectives of the Nigerian IT Policies.

1. To ensure that Information Technology resources are readily available to promote efficient national development.

2. To guarantee that the country benefits maximally and contributes meaningfully by providing the global solutions to the challenges of the Information Age.
3. To empower Nigerians to participate in software and IT development.
4. To encourage local production and manufacture of IT components in a competitive manner.
5. To establish and develop IT infrastructure and maximize its use nationwide.
6. To promote tourism and Nigerian arts and culture.
7. To enhance planning mechanisms and forecasting for the development of local infrastructure.
8. To enhance the effectiveness of environmental monitoring and control systems.
9. To re-engineer and improve urban and rural development schemes.
10. To empower the youth with IT skills and prepare them for global competitiveness.
11. To integrate IT into the mainstream of education and training.
12. To create IT awareness and ensure universal access in order to promote IT diffusion in all sectors of our national life.
13. To create an enabling environment and facilitate private sector (national and multinational) investment in the IT sector.
14. To stimulate the private sector to become the driving force for IT creativity and enhanced productivity and competitiveness.
15. To encourage government and private sector joint venture collaboration.
16. To develop human capital with emphasis on creating and supporting a knowledge-based society.
17. To create Special Incentive Programs (SIPs) to induce investment in the IT sector.
18. To generate additional foreign exchange earnings through expanded indigenous IT products and services.
19. To strengthen National identity and unity.
20. To build a mass pool of IT literate manpower using the NYSC, NDE and other platforms as "train the trainer" Scheme (TTT) for capacity building.
21. To set up Advisory standards for education, working practices and industry.
22. To establish appropriate institutional framework to achieve the goals stated above.

Self-Assessment Exercise (SAE)

1. Mention Five (5) of the General Objectives of the Nigeria IT Policies?
2. Why are security policies created?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning of Security policy. The Import ants of having a security policy to protect how people use networks and to deter wrong doers. You have also learnt on how the Creating Security Policies. Understanding the different process, you will have to follow before a standard policy is created. You will also have learnt the Nigerian policy and it important.

5.0 Summary

i. What you have learnt in this study unit concerns the definition of Security policy.
ii. You also have learnt how to Creating Security Policies.
iii. The study unit also talks about the Nigerian policies.

6.0 Tutor-Marked Assignment (TMA)

1. List and explain the key actions to be taken when creating a security policy?
2. Give five objectives of the Nigerian IT policies.

7.0 References/Further Reading

I-vission International, “The Impact of Cyber Criminality on the Economy of a Developing Nation”, 2009 International Colloquium theme, at: http://www.ivission.net, (accessed on 08 Jan 2011).
Stephanie Perrin, “Offenses against the confidentiality, integrity and availability of computer data and systems”, at: http://www.anu.edu.au/people, (accessed on 10 Jan. 2009). See also, Andrew Conry-Murray, “Strategies and Issues: Deciphering the cost of a Computer Crime”, Network Magazine, 5 April 2000 at: http://www.networkingmagazine.com /articles/NMG200204015003, (accessed on 20 April, 2009).
International Development Research Centre, “Cyber-criminality”, at: http://www.idrc.ca/ en/ev-4?441-201-1-DO_TOPIC.html, (accessed on 13 June, 2011); Emma Okonji,” Nigeria: Controlling Online Fraud Through Cyber Security – New Horizon, EC-Council Take Lead”, in Daily Independent (Lagos, Nigeria), 27 June 2011, at: http://www.allAfrica.com, (accessed on 28 June, 2011) ,

Module 3

Information System Security

Unit 1: Information System Security

Unit 2: Simulation in Security Planning and Management

Unit 3: Firewall

Unit 1

Information System Security

Content

1.0 Introduction

Protecting information and information systems from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction is one of the most important things in computer and network securities.

This study unit brings you the Basic principles on how to protect your information on a computer or network.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. explain the term and important information system security;
ii. list of different basic principles of information system.

3.0 Learning Contents

3.1 Information System Security

Information System Security means protecting information and information systems from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

The terms information security, computer security and information assurance are frequently used interchangeably. These fields are interrelated often and share the common goals of protecting the confidentiality, integrity and availability of information; however, there are some subtle differences between them. Information security can be defined as a means of protecting information systems from any illegitimate access and use, theft, amendment, or malicious attacks or penetration. Information security can also be described as “the process by which an organization protects and secures its systems, media, and facilities that process and maintain information vital to its operations”

Conceptually, it is important to clarify the differences in the meaning between the terms information security and computer security, which are often mistaken for each other. There is no doubt that the two concepts are inter-related because professionally speaking, they aim to advance the protection of information through the principles of confidentiality, integrity and availability. Despite the similar goals they both articulate or simply pursue, the two terms still have some differences. These differences are fundamentally found in their approaches, methodologies and variability in their areas of focus. Here, information security concentrates on ways to provide adequate confidentiality, integrity and availability of information but is less concerned with the data form be it electronic or print or any other forms of data. So, information security goes beyond the use of computer to create, modify, delete or store information. Therefore, its boundary goes beyond mere electronic medium.

On the other hand, in computer security, the central focus dwells on the techniques that enhance the availability and correct operation of a computer system with little attention on information stored and processed by the computer. One thing to note from the foregoing is the limited boundary that computer security acquires in information discourse. The point is that computer is entirely electronic, and there are other means like print through which we generate, amend, store or discard information. It is against this background that we conclude that information security is wider in scope than computer security, but each of them complements the functions and operations of the other.

Away from conceptual differences between information security and computer security, the experience of state and actors in the contemporary world has shown the growing relevance of information security. The increasing complexity of the modern state and sophisticated nature of contemporary business environment and corporations underscore the importance in mounting relevant mechanisms towards the protection of information and information system. For instance, let us look at the commercial banks in Nigeria in the 20th century, by the close of that century, virtually none of the banks could boast of having a capital base of US$ 1 billion. Again, during that period, online financial transactions like the use of ATMs, online shopping etc., were either non- existent or very limited.

Similarly, the period saw majority of the banks not having services that could enable their customers to save and withdraw their money from any of their branches without any geographic limitation, while most banks did not have up to one hundred thousand customers. But due to the opportunities provided by information technology revolution and recapitalization agenda of the Nigerian government, initiating the financial sector to be proactive in increasing their capital base and improving on their information system. The emergent information technology breakthrough especially the development of ground breaking computer software and electronic machines like ATM have really helped in gradually fizzling-out long queues and the use of tally number as well as manual counting money by cashiers.

Nostalgically, I could remember those days when customers would wake up as early as 4am in order to reach their respective banks latest by 6am in order to be attended to early enough due to heavy traffic of customers. One could even reach the bank and meet some people already on the queue outside the bank premises. Then one would be asking himself if those met on the queue slept at home. But now, with the introduction of ATM, people can withdraw their money electronically anytime and anywhere.

Considering the opportunities provided by innovations in information technology, you may agree with me that financial transactions, trading, information exchange, mailing, communication etc. have been made so easy. You should however note that there are always two sides to a coin. Though, there is much convenience and fun derivable from modern information system but it can be awful considering the risk and danger one can fall into. For instance, if one wants to purchase items online and logs on a wrong

web platform hosted to perpetrate scam, and you ignorantly provide your credit card details, and before you know, you have been duped. For this reason, people are being alerted to be very careful when doing online transactions.

One of the ways to reduce such risk is to install very strong security software that can easily detect and inform you if you are on a malicious site. Many organizations have gone into comatose or collapsed as a result of stealing, modifying, corrupting or deleting of vital information. It is therefore very important to put in place viable structures and programmes to protect your information. Now, let us quickly explain some of the methods we can use to safeguard our information systems and protect our information.

3.1.1 Approaches to Information Security

a) Confidentiality: In securing information systems, it is very germane to mount necessary machineries to advance the confidentiality of information. It is very paramount for the management of any organization to enlighten its staff on the need to take the issue of information very seriously. They need to know that it is very essential to prevent the organization's vital information from disclosure to unauthorized person(s) or system(s). Stiff penalty should be applied against any erring staff to deter others from doing the same. By commission or omission, if confidential information gets into the wrong hands of unauthorized person(s) or system(s), it will amount to a breach of confidentiality.

In procuring an international passport, you can log-on the website of the Nigerian Immigration Service to begin the process for your e-passport application. After filling the necessary forms online, you are requested to proceed with your payment, and here, you are given two options: offline or online payment. You may decide to do the payment online with use of your ATM card. In the process, the system will demand for your ATM details through ETRANSACT platform, and you provide it correctly. The gateway will debit your account where the ATM card domiciles, and consequently you will be allowed into the next stage of the application process after the confirmation of your payment, which will be electronically receipted.

Here, you need to be careful by not allowing anybody to peep into your financial transaction to avoid an unauthorized person to have access to your secret (pin) code, and if you allow such to happen before, during or after the period of transaction, you have committed a breach of confidentiality. You should know that it is incumbent on the organization(s) you transact with online to uphold the principle of confidentiality. Expectedly, when you are making your transaction, your credit card or ATM card details including pin-code will be transmitted from you to the organization/party with which you transact, and the details will also be transmitted from the said organization to a transaction processing network.

The system will ensure that confidentiality is enforced by encrypting (refer to 3.1.1 of last unit for the meaning of encryption) your ATM details especially the pin-code during the transmission, which are stored in a very secure location with highly restricted

access. In a situation whereby, you confirm that unauthorized person(s) has access into your financial details while the fault or criminal intent does not emanate from you, you have the right to institute a legal action against such erring organization for a breach of confidentiality.

Even if your spouse is allowed to access your bank statement account without authorisation is a breach of confidentiality, and the affected person can institute a legal action or warning to the bank management for the breach or may close down his/her account for lack of security necessitated by the confidentiality breach. If one has the right to sue for a breach of confidentiality that involves his/her own spouse let alone a stranger or a distanced person, it is evident that information security is very important. It will be baseless, if the management of an organization argues that it will not be liable if an offence of breach of confidentiality is committed by any of its staff without its involvement against any customer especially if it involves loss of money.

Recently, due to importance of information, the approach of due care and due diligence has been adapted in information security. Due care involves measures and actions that are taken by a company not only to protect its corporate image but also show liability for all activities that take place within it and establish regulations that will help to protect the company, its resources and employees. Due diligence means "continual activities that make sure the protection mechanisms are continually maintained and operational" (Harris, 2003). Therefore, if an organization fails in its responsibilities to check the activities of its staff, it will definitely be liable for any misdeed perpetrated by any of its staff such as disclosure of a customer's information to an unauthorized person, leading to a breach of confidentiality;

b) Integrity: Integrity can be described as a way of protecting information by restricting access to modification of data without authorization. Here, no amendment can be made on the data without authorization. In information security, efforts should be made to restrict activities of users of the systems from any data modification without being authorized. There are several ways through which the integrity of an information system can be violated. One of such ways is accidental or deliberate exposing of the system to malicious attacks.

Undoubtedly, this kind of violation occurs in many organizations where employees exhibit nonchalant attitude or criminal intent to compromise the integrity of the system. For instance, in some organizations with large network of information systems, family members, siblings or friends and even neighbours visit some employees, and many of these visitors may be allowed to use the organization's computers especially where they are connected to the internet. These use all sorts of storage facilities to down-load information from the internet. This attitude is likely to violate the integrity of the information system because apart from exposing the system to malware or virus attack, it will also allow strangers or visitors to have access to information, which ordinarily they are not supposed to;

c) Availability: Information system cannot be complete if there is no availability of information. It is the availability of information that makes information system what it

is. Therefore, it is imperative to have a network of actions functioning well. These actions include the computer system that is tasked with the storing and processing of the information while the security controls do the protection of the system, and the communication channels enable the users to access that information.

If we consider this network of functions, we may agree that it is difficult to have 'high availability systems' in countries like Nigeria where there is frequent power outage that can easily disrupt the operation of the system. High availability systems are those systems that are always available, and

demand necessary mechanisms to be brought to bear in order to prevent disruptions that may result from hardware failures, power outages, physical destruction of the information systems, to mention a few;

d) Authenticity: Information security demands that we can just collect information whether electronic or print for the sake of it, but we should endeavour to clarify the authenticity or genuineness of such information. It is by so doing we can have reliable and quality information;

e) Risk Management: Everything about life is a risk. There is risk, even in the relationship between two or more people. How do we describe risk management? According to CISA Review Manual (2006), risk management can be described as:

…. the process of identifying vulnerabilities and threats to the information resources used by an organization in achieving business objectives, and deciding what countermeasures, if any, to take in reducing risk to an acceptable level, based on the value of the information resource to the organization.

Risk is the everyday business of each man. Sometimes, we decide out of the blues to see a loved one in his/her place work, even after calling him/her of our coming, the subconscious still has a doubt about meeting him/her at the office because every second is clouded with eventuality. It is possible for the receptionist to inform us that he/she had an emergency from the headquarters and he/she tried to get you on phone to inform you about the new development (urgent call to report at the headquarters) but he/she could not reach on phone.

Considering the above scenario, you may agree with me that risk is second nature to man. The decision of a man and a lady to get married is a risk: the marriage may succeed or fail. It is against this background that many people adopt various approaches or measures to manage risk in their relationships. In information security, risk management is very essential because it determines the preparedness of an organization against any threat as it relates its information system.

Furthermore, in as much that information from its collection, modification to erasure involves risk, it is pertinent to develop countermeasures or controls to manage the risks but it is more important to "strike a balance between productivity, cost, effectiveness of the countermeasures, and the value of the informational asset being protected" (Error! Hyperlink reference not valid.), meaning that it is not appropriate to spend too much money or employing highly productive and effective in securing an

information asset that has little importance to the overall interest of an organisation; and

f) Information Classification: There are different types of information in information system, and there is the need to classify these according to their level of importance and confidentiality. By doing so, we will be able identify the amount of protection to be accorded to each of this available information. For the purpose of engendering culture of information security, it is imperative for an organization to adopt a classification policy, so that it will be able to determine the required security controls of every information in accordance with its classification.

For instance, the head of an organization is having an extra-marital affair with a former female staff who he communicates on regular basis. May be because he is not internet-literate and always asks his male secretary to help him send mails to his 'lady-friend' while mandating the secretary to classify the mails as top secret, giving more importance to such mails than the very important information of the organization. This reflects a clear case of misuse of office and misclassification because it abnormal to place private issues above those that concern the organization and the work environment.

Within an organizational setting, we can classify information based on the value of it to the organization. In private organizations, classification models are usually labelled as: public, sensitive, private, confidential. But in big security outfits and government organizations the classification labels used include: unclassified, sensitive, but unclassified, restricted, confidential, secret, top secret. These classification labels are listed according to the security controls needed in protecting them, and the classification exercise should be continually reviewed.

Self-Assessment Exercise (SAE)

1. What are the approaches to information security?

Self-Assessment Answers (SAA)

3.2 Basic Principles

Confidentiality

Confidentiality is the term used to prevent the disclosure of information to unauthorized individuals or systems. For example, a credit card transaction on the Internet requires the credit card number to be transmitted from the buyer to the merchant and from the merchant to a transaction processing network. The system attempts to enforce confidentiality by encrypting the card number during transmission, by limiting the places where it might appear (in databases, log files, backups, printed receipts, and

so on), and by restricting access to the places where it is stored. If an unauthorized party obtains the card number in any way, a breach of confidentiality has occurred.

Confidentiality is necessary (but not sufficient) for maintaining the privacy of the people whose personal information a system holds.

Integrity

In information security, integrity means that data cannot be modified undetectably. This is not the same thing as referential integrity in databases, although it can be viewed as a special case of Consistency as understood in the classic ACID model of transaction processing. Integrity is violated when a message is actively modified in transit. Information security systems typically provide message integrity in addition to data confidentiality.

Accessibility

For any information system to serve its purpose, the information must be available when it is needed. This means that the computing systems used to store and process the information, the security controls used to protect it, and the communication channels used to access it must be functioning correctly. High availability systems aim to remain available at all times, preventing service disruptions due to power outages, hardware failures, and system upgrades. Ensuring availability also involves preventing denial-of-service attacks.

Authenticity

In computing, e-Business, and information security, it is necessary to ensure that the data, transactions, communications or documents (electronic or physical) are genuine. It is also important for authenticity to validate that both parties involved are who they claim they are.

Non-repudiation

In law, non-repudiation implies one's intention to fulfil their obligations to a contract. It also implies that one party of a transaction cannot deny having received a transaction nor can the other party deny having sent a transaction.

Electronic commerce uses technology such as digital signatures and public key encryption to establish authenticity and non-repudiation.

Self-Assessment Exercise (SAE)

1. What are the basic principles to information security?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning of information system security. How important it is to know how to protect your information on a computer or network. You have also learnt the basic principles of information system security.

5.0 Summary

i. Explain the term and importance information system security

ii. List of different basic principles of information system security

6.0 Tutor-Marked Assignment (TMAs)

1. List and explain 4 Basic principles of information system security?
2. Explain the term information system security.

7.0 References/Further Reading

I-vission International, “The Impact of Cyber Criminality on the Economy of a Developing Nation”, 2009 International Colloquium theme, at: http://www.ivission.net, (accessed on 08 Jan 2011).

Stephanie Perrin, “Offenses against the confidentiality, integrity and availability of computer data and systems”, at: http://www.anu.edu.au/people, (accessed on 10 Jan. 2009). See also, Andrew Conry-Murray, “Strategies and Issues: Deciphering the cost of a Computer Crime”, Network Magazine, 5 April 2000 at: http://www.networkingmagazine.com/ articles/NMG200204015003, (accessed on 20 April, 2009).

International Development Research Centre, “Cyber-criminality”, at: http://www.idrc.ca/en/ev-4?441-201-1-DO_TOPIC.html, (accessed on 13 June, 2011); Emma Okonji,” Nigeria: Controlling Online Fraud Through Cyber Security – New Horizon, EC-Council Take Lead”, in Daily Independent (Lagos, Nigeria), 27 June 2011, at: http://www.allAfrica.com, (accessed on 28 June, 2011) ,

Unit 2

Simulation in Security Planning and Management

Content

1.0 Introduction

Simulation can be applied in different ways, and these may include modelling of natural systems aimed at having an idea of possible vulnerability of the system to specific threats or attacks. It can also be applied to insight on the performance of (security) technology for optimization and effectiveness, safety engineering and training. Simply, simulation assists us to identify potential shortcomings or failures in operation that we may later encounter in the discharge of our duties as security experts or professional.

It is important to note that we must be very careful in our choice of information source to be selected for the simulation process. Selection of relevant information is very strategic in arriving at decision on the crucial characteristics and behaviour to select for the process. It is essential to use simplifying approximations and assumptions in simulation activities. In this unit, the reader/student shall be exposed to the meaning and various types of simulation and show how they can be applied as well as their relevance to security planning and management.

2.0 Learning Outcomes

At the end of this unit, you should be able to:

1. Describe the concept of simulation;
2. Identify types of simulation; and
3. explain the identified types of simulation.

3.0 Learning Contents

3.1 Meaning of Simulation

Simulation is not a new concept and practice in security. In traditional African society, simulation was often applied by the guards and warriors. For instance, in the traditional Ila-Orangun in present Osun state, Nigeria, after a series of simulation exercise by the community guards, they discovered that offensive attacks may come from enemies without prior notice especially in the middle of the night. Therefore, they decided to dig a trench to surround the entire community, as a booby trap against the enemy(ies) who may wish to launch an offensive attack and avoid being caught unaware. Other examples may include the old Oyo Empire wall, Kano wall etc. At this juncture, let us draw our attention to the task of this segment of the unit, which is to expose ourselves to some of the existing definitions of the term simulation. Simulation can be defined as the imitation of some real thing, state of affairs, or process. The act of stimulating something generally entails representing certain key characteristics or behaviour of a selected physical or abstract system"

.... the process of creating a model (i.e., an abstract representation or facsimile) of an existing or proposed system (e.g., a project, a business, a mine, a watershed, a forest, the organs in your body) in order to identify and understand those factors which control

the system and/or to predict (forecast) the future behavior of the system. Almost any system which can be quantitatively described using equations and/or rules can be simulated. The underlying purpose of simulation is to shed light on the underlying mechanisms that control the behavior of a system. More practically, simulation can be used to predict (forecast) the future behavior of a system and determine what you can do to influence that future behavior. That is, simulation can be used to predict the way in which the system will evolve and respond to its surroundings, so that you can identify any necessary changes that will help make the system perform the way that you want it to a broad collection of methods used to study and analyze the behavior and performance of actual or theoretical systems. Simulation studies are performed, not on the real-world system, but on a (usually computer-based) model of the system created for the purpose of studying certain system dynamics and characteristics. The purpose of any model is to enable its users to draw conclusions about the real system by studying and analyzing the model. The major reasons for developing a model, as opposed to analyzing the real system, include economics, unavailability of a “real” system, and the goal of achieving a deeper understanding of the relationships between the elements of the system (://www.answers.com/topic/simulation).

If we subject the foregoing definitions to operational dissection, you may agree with me that simulation can be used differently by various professions or for different purposes. But, at the beginning of the 20th century, introduction of computer to the world population and the emerging appreciation of systems theory and cybernetic studies unified to a large extent the processes of simulation in various fields. For instance, relevant officials in an Examination body like the West African Examination Council (WAEC) can conduct simulation to test the reliability of various measures put in place by the examination body to curb or reduce examination practices. Emergency workers can also engage in simulation to examine the effectiveness or efficiency of their emergency systems and level of preparedness to responding to emergency situations.

In addition, the police can also conduct simulation exercise to put their preparedness to test on how timely and effectively they can respond to any security threats. They act the simulating scripts as if the situation is real. Frankly, it is no exaggeration that Nigeria police lacks the culture of security simulation. Otherwise, the way the men and officers of the police are being killed on a regular basis from attacks on police stations, armed robbers' bullets or any other threats/hazards would have been very minimal. It is no surprise that police personnel always fail to respond very appropriately and effectively to emergencies. This shows that their level of preparedness is very far below the average. This is one of factors responsible for the call being made by concerned citizens, for the introduction of joint patrol that would be composed of members of police and the armed forces especially the Army.

Simulation can be facilitated with computer in health-care, military, or education simulation or any other sector. A lot of simulation software have been invented,

having the user to decide on which one will fit into the purpose for which the simulation is to be carried out.

Self-Assessment Exercise (SAE)

1. What is simulation?

Self-Assessment Answer (SAA)

3.2 Types of Simulation

The importance of simulation in strategic security planning and management cannot be over-emphasized, especially as it concerns identifying any weakness in our systems and operation through which we can develop alternative ideas and policies to address such weakness and vulnerability to mitigate the risk of system or/and operational failure. As we have rightly pointed out earlier, simulation can be used for numerous purposes depending on the problem we want to unravel and apply solution to.

The complexity of the problem we are working on may require a simulation package or exercise that will demand from us, a very sound knowledge on how to apply the tool(s). In this case, the services of experts may be required to guide the simulation process if the application is a (very) technical one. Let us go back to the basis of this segment of the unit, which involves explaining various types of simulation. Basically, types of simulation may include the following:

(a) Education and Training Simulation: The security profession requires sufficient mental alertness and physical strength, and that is why it is not everybody that can hold a security job. The nature of the security profession underscores the need by professionals to engage themselves in periodic training. For instance, it is very sad to hear such situations where police pursue armed robbers and fail to pin down the bandits despite the numerical strength and strategic advantage the police ought to have over them. Sometimes, the bandits appear to be more equipped than the police as a result of operational failure on the part of the police. Situations like these undermine the relevance of the public security personnel's capacity to maintain law and order as well as check any acts of Criminality (Onyeozili, 2005: 40).

Also, the problem of police team being overpowered by the sophistication of the weapons that robbers carry can be easily addressed if the men and officers of the police undergo simulation exercise from time to time. Even members of patrol team should simulate before going out to discharge their duties, so that they can weigh their vulnerability against their capacity for optimal performance. If they can identify areas of vulnerability in their operations, they can then map out strategies to build their

capacity towards preventing the threats from happening or mitigating the effects, that such threats can have on them or their operation when and where they occur.

For instance, a security patrol team that receives a signal that a bank robbery is going on in a place will be expected to act immediately to foil the robbery. But, most times, police patrol (rescue) teams are in the habit of announcing their (the police) coming to the robbers through the blowing of sirens. Unfortunately, before they reach the location of the robbery, the robbers would have laid ambush for them, a situation which often forces the lucky policemen to retreat, as they always find it difficult to recover promptly due to lack preparedness in hazard mitigation and strategic planning.

Where security men simulate, their preparedness level against any attack will be very high because this would have projected into the future and identified potential threats or challenges that may be encountered in the course of discharging duties as a security professional. And in doing this, you will prepare yourself beforehand, and in the event that such hazard or attack occurs the losses that may be recorded will be minimal. I feel it is more appropriate for the police not to alert the robbers of their coming through blowing of siren. This is because the daring nature of most armed robbers in the country has made blowing of siren by law enforcement agents obsolete. They are die-hard and always willing to challenge the law enforcement agents in gun battle. Strategically, for the fact that the rescue team do not know the identities of the bandits, it is better that they send an intelligence team to do some collation of vital information about the happening in the affected area for situation analysis.

After conducting situation analysis, we need to conduct risk analysis and assessment to know if it is most appropriate to move straight into where robbery is taking place or it is most reasonable to block all roads that lead to the affected location and wait for the robbers to some out. The reason for this process is to avoid fatality or loss of lives especially civilians that may be hit by strong bullets in the course of engaging the bandits.

After conducting a risk assessment, it may be agreed that a team should be sent to the scene, all members of whom should be in plain-cloths, so that they cannot be easily identified by the robbers. But, if it is considered that the robbers will do more harm by being allowed to complete their operation before attacking them, a team of experienced officers and men may be sent in.

Caution should be exercised here as there is the need to equip the policemen and officers with bullet-proof vests because people are not applying for security profession with the ultimate desire to suffer avoidable death. The use of tear-gas may be required to make the robbers lose their balance and destabilize them. I feel, here is the most convenient place to stop our discussion on ways to foil or liquidate a robbery incident. As you may know, we cannot exhaust all aspects of security discourse in a forum. Security requires continued research and intellectual probing.

By and large, education simulation helps us to know the most appropriate training (academic or fitness) we need to undertake to optimise our performance in the

discharge of our duties as security experts and professionals. There is no doubt, security professionalism is very tasking considering the hazards and work overload that characterize it. Security personnel work long hours that sometimes they hardly have time to attend to personal issues. It is pertinent for policy makers to see education simulation as important element of security sector reform (SSR).

There are three types of education and training simulation. The first one is live simulation. In live simulation, it is expected that trainees use stimulated or mannequin equipment in the real world. As you may be aware, it is not all security trainings that can be undertaken with the real equipment. For instance, if a training is going to be conducted in knowing how effectively, each of your security officers can act in the face of hazard or in shoot-outs with criminals (terrorists, militants, armed robbers, etc.), one cannot expect that those trainees should be equipped with live ammunitions because of the risk involved in such action.

It might not be wise to allow the use of real weapons for training due to the possibility of recording avoidable deaths among the trainees. It is, therefore, advisable to use live simulation through which we can still know the level of competence of each trainee without putting them to unnecessary risk of killing themselves. In the process, the trainees will identify their individual and collective areas of vulnerability and subsequently develop ways through which they can improve on their capacity for optimal performance.

It is unfortunate that in Nigeria, security personnel, most times, are posted to specific positions without considering their level of competence for the job or assignment. For instance, officers of the armed forces lobby to be included in peacekeeping contingent without training on simulation, which can help in making them aware of the inherent risks involved in undertaking peacekeeping mission in troubled zone(s), and be well-prepared psychologically, physically, emotionally and strategically. Many soldiers lose their lives after being seduced with ladies by the enemies. Live simulation is also known as “high fidelity”. This demonstrates the samples of the (possible) real performance of the trainees compared to the “low fidelity” simulation that is based on the use of pencil and paper that can only show “signs of performance” rather than the practical performance of the trainee(s).

The second type of simulation is virtual simulation. Virtual simulation actually involves the use of real people using simulated tools in a simulated world. In virtual simulation, the process often involves the use of computer simulation by the trainees in undergoing training. Virtual simulation usually involves security employees training in an artificial environment (generated through computer software) and take it as if it is real. The exercise is undertaken through the manipulation of the computer keys or mouse. Virtual simulation is often facilitated through sight and sound.

The third type is Constructive simulation. Constructive simulation usually involves a process whereby simulated people use simulated tools in a simulated world. This simulation type is also known as war-gaming. It can be described as a form of simulation whereby “players command armies of soldiers and equipment that move

around on around on a board" (Error! Hyperlink reference not valid.). We shall discuss this type of simulation extensively subsequently (see military simulation);

(b) Health Care Simulation: This form of simulation is also in security and safety. It affords health care providers an opportunity to examine their capacity to respond to emergency situations. Simulation helps to reduce the situation of crisis in patients because, as stressed by Eder-Van Hook (2004):

A health care provider's ability to react to prudently in an unexpected situation is one of the most critical factors in creating positive outcome in medical emergency, regardless of whether it occurs on the battlefield, freeway, or hospital emergency room

Eder-Van Hook stressed further, saying that medical errors or lack of adequate medical attention to patients by health care providers have led to almost ninety- eight thousand (98,000) deaths, with a lot of financial implications, which amounts to between $US37 and $US50 million on an annual basis. It is very unfortunate the way health-care providers in Nigeria respond to emergency cases. For instance, somebody who is shot by a group of robbers will hardly survive the bullet wounds he/she sustains from the bandits. The issue is not basically because he/she has been shot in part(s) of the body and can hardly survive but the poor handling of the situation always results in high rate of avoidable deaths in our hospitals.

It is a very abnormal situation for some patients in teaching hospitals to be given referral to private clinics for treatment, and the reason always given is lack of competent medical hands that can manage the medical needs of these poor patients. What a situation like that denotes is that our teaching hospitals can no longer be considered as tertiary health-care institutions as they have relegated their responsibilities (in the name of making money) and transfer such to private clinics many of which can hardly boast of having enough qualified medical staff. In this kind of messy situation, health-care simulation will make no sense to the health practitioners because many of them have found it more important to make brisk business than saving lives. Sadly, the decision makers that are supposed to caution them are less bothered because they can afford treatment abroad most times.

Bringing our attention back to the subject under discourse, health-care simulation has been demonstrating growing relevance in the modern medical world. Several medical simulations usually involve connecting computer to a plastic simulation of related anatomy e.g. the use of dummy that reacts to injected drugs and can also be automated to generate simulations of life threatening emergency situations or cases. In some instances, the simulation procedures are captioned and reproduced by computer graphics tools. Health- care simulation has continued to be found useful in training medical practitioners on the ways to build their individual and collective capacities in responding effectively and timely to emergency situations especially as regards the issue of saving lives and management of health crisis; and

(c) Military simulation is also known as war-gaming. It involves an act of simulating with the intention of putting various theories of war into test and if need be, refine those

theories without undergoing the real hostility of warfare. The rationale behind military simulation is to facilitate a process through which one can arrive at tactical, strategic and doctrinal answers to problems that bother on defence and warfare. War-gaming can be described as a form of game or hobby that showcases various activities of military operation in a simulated environment.

War-gaming can be used for relaxation or game-playing, and this is known as conflict simulations or consims. But, if we desire to engage in war-gaming for the purpose of warfare, the process is usually known as war-game or military exercise. Meanwhile, those that engage in war-gaming as a hobby don't usually draw any distinction between the two aspects of warming earlier mentioned because they always contend that whether for war-making or relaxation, any war-game should demonstrate to a large extent characteristic of human behaviour as it would be have been in the real world when war is being conducted.

However, war-game can be classified as historical, hypothetical, fantasy, or science fiction. Historical war-games are often modelled after real events and putting into simulation reasonable approximations of the actual forces, terrain as well important factors that represent the experience of the real players or participants. For instance, those who usually have experience of Play Station (game) especially the soccer, will notice that each of the players demonstrates the real capacity of the model player(s).

In Play Station 2, the scoring ability of Christian Ronaldo (formerly of Manchester United FC of England, but now of Real Madrid FC of Spain) cannot be compared with his scoring ability in Play Station 3, there being tremendous improvement in his scoring ability. Play Station 3 reflects the improvement in his scoring ability, showing that the simulated Christian Ronaldo demonstrates the actual ability of the real Ronaldo as at the time the programme or application was being written. Here, the history of the performance and experience of players is considered in determining the capacity of the simulated players.

Hypothetical war-games are actually those games that involve drawing simulating materials about wars that did not happen. Fantasy and science fiction war-games usually involve developing games from wars generated from works of fiction or (creative) imagination. There are more types of simulation but the three we have discussed in this unit can be considered as the basic types of simulation for security planning and management. The reason is that they cover virtually all aspects of securitisation namely disaster management, law enforcement, defence, warfare, to mention a few.

Self-Assessment Exercise (SAE)

1. Explain types of simulation

Self-Assessment Answers (SAA)

4.0 Conclusion

In the final summation of our discourse on the concept of simulation, the concept can be described as an essential instrument of decision analysis. It affords us a great opportunity as security experts or practitioners to know the various areas of deficiency in our service delivery, operations as well as potential threats. The relevance of simulation in modern security planning and management is enormous owing to the nature of uncertainty that characterises the general affairs including security issues in recent times. For instance, since the end of the Cold War, state actors have constituted less threat to national and international security compared to the destructive attitude portrayed by some non-state actors, which absolutely undermine the potentials of the security sector to maintain law and order.

The event of September 11 2001 terrorist attack in the US has shown the very destructive dimension that threat emanating from non-state actors has assumed. Prior to that ugly incident, we could hardly conceive the idea that commercial planes could be used as weapons of mass destruction. There is no doubt, simulation assists us to appraise and compare among alternative designs, plans and policies, so that we can choose those that will enable us achieve optimal performance for best results. Simulation is very vital in security planning due to its capacity to unravel secrets of uncertainties and guide us in deriving solutions to the problem of uncertainty that can affect our operations and policy-actions as security professionals in a quantifiable way. Above all, simulation provides us an avenue to explore ways through which we can improve on our preparedness for risk and hazard mitigation and high recovery capacity in the event of any hazards or threats.

5.0 Summary

In this unit we explored some of the existing definitions of the concept ‘simulation’. One fact that appeared from the various definitions presented, shows that simulation can be applied in various fields or disciplines and for different purposes. We have therefore discussed some of the basic types of simulation that can be employed in security planning and management in order to safeguard the security of people as well as their property. I hope that you have found this unit as interesting as you expected it to be. If you have any question on any aspect of this unit or the course in general, please feel free to get in touch with the instructional and tutorial facilitator. Good luck.

6.0 Tutor Marked Assignment (TMAs)

1. What is simulation?
2. Write short note on three types of simulation.

7.0 References and Further Reading

://en.wikipedia.org/wiki/simulation. Retrieved on 25 August 2014.

://en.wiktionary.org/wiki/simulation. Retrieved on 25 August 2014.

Error! Hyperlink reference not valid. Retrieved on 25 August 2014.

://www.alanemrich.com/PGD/Week_03/PGD_what_is_a_Wargame.htm. Retrieved on 14 April, 2014.

://www.answers.com/topic/simulation. Retrieved on 24 August 2014.

://www.goldsim.com/ Content.asp? PageID=91. Retrieved on 24 August 2014,

Unit 3

Firewall

Content

1.0 Introduction

Firewalls are usually used to enforce a particularly simple kind of access control policy. Inside users are permitted to connect to any network service desired. External users are restricted: there are some services that are intended to be externally visible, and external users are permitted to connect to these services.

2.0 Learning Outcome

After studying the material in this unit, you should be able to:

1. Explain the motivation for firewalls.

3.0 Learning Contents

3.1 The Motivation for Firewalls

Suppose you are given a machine and asked to harden it against external attack. How do you do it? One starting point is to look at the network services that this machine is providing to the outside world. If any of its network services are buggy or have security holes, a hacker may be able to penetrate your machine by interacting with that application. As we know, bugs are inevitable, and bugs in security-critical applications often lead to security holes. Thus, the more network services your machine runs, the greater the risk. This suggests one simple way to reduce the risk of external attack. Turn off every unnecessary network service. Disable every network-accessible application that isn't absolutely needed. Build a stripped-down box that is running the least amount of code necessary; after all, any code that you don't run, can't harm you. And for any network service that you do have to run, double-check that is has been implemented and configured securely, and take every precaution you can to render its use safe.

This is an intuitive and effective approach, and it can work well when you only have one or two machines to secure, but now let's consider what happens when we scale things up. Suppose you are in charge of security for all of FUT Minna. Your job is to protect the computer systems, networks, and computing infrastructure of the entire company from external attack. How are you going to do it? If the company has thousands of computers, it won't be easy to harden every single machine individually. There may be many different operating systems and hardware platforms. Different users may have vastly different users, and a service that can be disabled for one user might be necessary to another user's job. Moreover, new machines are bought all the time, machines come and go every day, and users upgrade their machines.

At this scale, it is often hard even to get an accurate list of all machines inside the company and if you miss even one machine, it is then a vulnerable point that can be broken into and might serve as a jumping-off point for attackers to use to attack the rest of your network. The sheer complexity of managing all of this might make it infeasible to harden each machine individually.

A little bit of background. In its simplest form, we can visualize the topology of the internal network as shown in Figure 1. We have an internal network, which hosts all the company's machines, and the external world (e.g., the rest of the Internet), and a communications link between the two. How do we decide what is inside, and what is outside? We might decide that we trust all company employees, but we don't trust anyone else (a very simple threat model). Then we'll define the internal network to contain machines owned by trusted employees, and the external world to include everything else. The link to our ISP might be the link between these two worlds.

The very simplest security policy is an outbound-only policy. Let's distinguish between inbound and outbound connections. Inbound connections are initiated by external users and attempt to connect to services running on internal machines, while outbound connections are attempts by internal users to initiate contact with external services. An outbound-only policy would permit all outbound connections (reasoning: internal users are trusted; if they want to open a connection, we'll let them), but all inbound connections would be strictly denied. The effect is that none of our network services are visible to the outside world, though of course they can still be accessed by internal users.

Unfortunately, this policy is probably too restrictive for any large organization, since it means that the company cannot run a web server, a mail server, a FTP server, and so on. Therefore, we will need a little more flexibility in how we define the security policy. In general, the security policy is going to be a particular kind of access control policy. We will have two subjects: an anonymous external user, and a generic inside user1. The objects are the set of network services that are run on all inside machines; if there are 1000 machines, and each machine runs 5 network services, we end up with 5000 objects. The access control policy should then specify, for each subject and each object, whether that subject has permission to access that object.

Firewalls are usually used to enforce a particularly simple kind of access control policy. Inside users are permitted to connect to any network service desired. External users are restricted: there are some services that are intended to be externally visible, and external users are permitted to connect to these services, but there are also other services that are not intended to be accessible from the outside world, and those services are blocked by the access policy. The first thing the security administrator needs to do is identify a security policy, or in other words, which services external users should and shouldn't be given access to. How should we do it? Broadly speaking, there are two philosophies we might use to determine which services we allow external users to connect to:

Default-allow: By default, every network service is permitted, unless it has been specially listed as denied. Under this approach, one might start off by allowing outside users access to all internal services, and then mark a few that are known to be unsafe and should be blocked. For instance, if tomorrow we hear about a new Slammer II worm, which spreads by exploiting vulnerability in SQL servers, then we might revise our security policy by denying outsiders’ access to all our SQL servers.

Default-deny: By default, every network service is denied, unless it has been specially listed as allowed. We might start off with a list of a few known servers that need to be visible to the outside world and that have been adjudged to be reasonably safe; external users will then be implicitly denied access to any service not the list. If our users complain that, say, their department's FTP server is not accessible to the outside world, we can check whether they are running a reasonably safe and properly configured implementation of the FTP service and add them to the .allow. List if so.

A default-allow policy sounds a lot more convenient, because from a functionality point of view, everything stays working. However, from a security perspective, default-allow is seriously _awed. The problem is that default-allow fails open: if you make any mistake (i.e., there is some service that is vulnerable, but you forget to add it to the .deny. list), then the result is likely to be a security failure. In comparison, default-deny fails closed: if you make a mistake (i.e., some service that is safe has been mistakenly omitted from the .allow. list), then the result is merely a loss of functionality or availability. When operating at large scales, such errors of omission are likely to be common2. Because errors of omission are a lot more dangerous in a default-allow policy than in a default-deny policy, and because the cost of a security failure is often a lot more than the cost of a loss of functionality, default-deny is usually a much safer bet.

Default-deny has another advantage. When the system fails open, you may never notice the failure. Attackers who penetrate your system security are unlikely to tell you that they have done so, and so security Breaches may go unnoticed for a long time. This gets you into an arms race, where you have to keep up with all the attacks hackers discover and even stay ahead of them. Arms races are generally a losing proposition, because there are a lot more of the hackers than there are of the defenders, and the hacker only has to win once to make you really miserable. In contrast, when the system fails closed, someone will probably notice (they'll complain: why isn't the FTP service working?), and the omission will be immediately evident and easily correctable. This makes failures in default-allow systems that much costlier than failures in default-deny systems.

For these reasons, almost all well-implemented _firewalls use a default-deny policy. The security policy species a list of .allowed services, which external users are permitted to connect to, and all other services are forbidden. In many cases, some kind of risk assessment and cost-benefit analysis is applied to every network service on the allowed list; if some service is too risky compared its benefits, and then it is removed from the allowed list.

Self-Assessment Exercise (SAE)

1. What do you understand by Motivation for Firewalls?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning of Motivation for Firewalls The important of firewalls in security our networks and computers. The firewall protects viruses and other threats from entering your network and computers.

5.0 Summary

Explain the Motivation for Firewalls

6.0 Tutor-Marked Assignment (TMAs)

i. Give a brief explanation of The Motivation for Firewalls.

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html

http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

Dark Reading - Tech Insight: Database Activity Monitoring

Module 4

Isolation

Unit 1: Isolation

Unit 2: Secure Software

Unit 3: Cryptography

Unit 1

Isolation

Content

1.0 Introduction

A program is isolated if it cannot affect other programs on the system. Thus, isolation refers to an inability to causally influence other programs on the system. Isolation is related to some topics we have seen before, such as access control. You will also learn the Decomposing Software for Security. This how software are used to protect networks

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

1. explain What is Isolation;
2. define what is meant by the terms entrepreneur and entrepreneurship.

3.0 Learning Contents

3.1 Isolation

The topic for today is isolation. A program is isolated if it cannot affect other programs on the system. Thus, isolation refers to an inability to causally influence other programs on the system. Isolation is related to some topics we have seen before, such as access control. One difference is that access control is a mechanism for enforcing some security policy (a means to an end), whereas isolation is a security goal (the end itself). Memory protection does not prevent other kinds of influence, such as opening an IPC connection from one process to another. When we want to isolate a process, we want to isolate again all influences, so memory protection alone is not enough.

What are the applications of isolation? Here are a few I run across a cool piece of software that will draw dancing pigs on the screen, and I want to download it and try it, but I don't know whether I can trust the developer. It would be great to be able to run it in an isolated environment, where it cannot harm the rest of my machine even if it contains malicious code or bugs.

This is often known as the sandboxing problem. We want to give the downloaded software its own little sandbox where it can do whatever it wants, as long as it doesn't escape the sandbox. If it tries to do anything disruptive, the effect will be limited to its sandbox, so the worst that can happen is it will disrupt itself. I want to display a MS Word _le that someone emailed me, but I don't want it to be able to infect my machine with a macro virus. It would be great if I could run a sandboxed instance of MS Word to view just this document, with no fear that it will trash my other Word documents. I'm designing a complicated software application. Following the principle of least privilege, I want to decompose the application into multiple pieces. Each piece should be isolated from the others, so that if one piece is penetrated, the integrity of the others will be preserved.

Soon I will start to show you a number of different ways to try to enforce a policy of isolation, but first let's explore the software decomposition issue a bit more to understand some of the requirements.

In database systems, isolation determines how transaction integrity is visible to other users and systems. For example, when a user is creating a Purchase Order and has created the header, but not the PO lines, is the header available for other systems/users, carrying out concurrent operations (such as a report on Purchase Orders), to see?

A lower isolation level increases the ability of many users to access data at the same time but increases the number of concurrency effects (such as dirty reads or lost updates) users might encounter. Conversely, a higher isolation level reduces the types of concurrency effects that users may encounter but requires more system resources and increases the chances that one transaction will block another.

It is typically defined at database level as a property that defines how/when the changes made by one operation become visible to other, but on older systems may be implemented systemically, for example through the use of temporary tables. In two-tier systems, a TP (Transaction Processing) manager is required to maintain isolation. In n-tier systems (such as multiple websites attempting to book the last seat on a flight) a combination of stored procedures and transaction management is required to commit the booking and confirm to the customer.

Isolation levels

Of the four ACID properties in a DBMS (Database Management System), the isolation property is the one most often relaxed. When attempting to maintain the highest level of isolation, a DBMS usually acquires locks on data or implements multiversion concurrency control, which may result in a loss of concurrency. This requires adding logic for the application to function correctly.

Most DBMSs offer a number of transaction isolation levels, which control the degree of locking that occurs when selecting data. For many database applications, the majority of database transactions can be constructed to avoid requiring high isolation levels (e.g. SERIALIZABLE level), thus reducing the locking overhead for the system. The programmer must carefully analyze database access code to ensure that any relaxation of isolation does not cause software bugs that are difficult to find. Conversely, if higher isolation levels are used, the possibility of deadlock is increased, which also requires careful analysis and programming techniques to avoid.

The isolation levels defined by the ANSI/ISO SQL standard are listed as follows:

Serializable

This is the highest isolation level. With a lock-based concurrency control DBMS implementation, serializability requires read and write locks (acquired on selected data) to be released at the end of the transaction. Also range-locks must be acquired

when a SELECT query uses a ranged WHERE clause, especially to avoid the phantom reads phenomenon (see below).

When using non-lock-based concurrency control, no locks are acquired; however, if the system detects a write collision among several concurrent transactions, only one of them is allowed to commit. See snapshot isolation for more details on this topic.

Repeatable reads

In this isolation level, a lock-based concurrency control DBMS implementation keeps read and write locks (acquired on selected data) until the end of the transaction. However, range-locks are not managed, so phantom reads can occur.

Read committed

In this isolation level, a lock-based concurrency control DBMS implementation keeps write locks (acquired on selected data) until the end of the transaction, but read locks are released as soon as the SELECT operation is performed (so the non-repeatable reads phenomenon can occur in this isolation level, as discussed below). As in the previous level, range-locks are not managed.

Putting it in simpler words, read committed is an isolation level that guarantees that any data read is committed at the moment it is read. It simply restricts the reader from seeing any intermediate, uncommitted, 'dirty' read. It makes no promise whatsoever that if the transaction re-issues the read, it will find the same data; data is free to change after it is read.

Read uncommitted

This is the lowest isolation level. In this level, dirty reads are allowed, so one transaction may see not-yet-committed changes made by other transactions.

Since each isolation level is stronger than those below, in that no higher isolation level allows an action forbidden by a lower one, the standard permits a DBMS to run a transaction at an isolation level stronger than that requested

3.2 Decomposing Software for Security

At one point, one of the most popular mail daemons was sendmail, an application written by Eric Allman while he was a staff member here in Berkeley EECS. Sendmail is a large, monolithic application, consisting over 100K lines of C code. It runs as root. Unfortunately, it has been plagued by security problems and because it runs with root privilege, each one of those security holes exposes the entire machine to mischief by the intruder. qmail is a secure mailer written by Dan Bernstein, partly in response to these problems. qmail is now the second most popular mail daemon on the Internet. Interestingly, in 1997 Bernstein offered a $500 prize to the first person to find a serious security hole in his system. The $500 still remains unclaimed. Let's see why qmail has fared so much better than sendmail.

In short, qmail uses many little programs because this is the easiest way to get isolation in Unix. Each program is (at least partially) isolated from all the others. And isolation is what lets us meaningfully decompose the application into watertight compartments. That minimizes the spread of damage. In effect, it provides a _firewall between each module of the application.

These examples also make clear a more fundamental point about sandboxing and isolation: pure isolation is usually too strict. Much more commonly, we need to combine isolation with controlled sharing. Isolation is a starting point that is analogous to the deny-all starting point of a default-deny policy. Controlled sharing is about allowing limited escape routes out of the sandbox, so that one can interact usefully with the sandboxed application (hopefully without exposing ourselves to attack). In the case of qmail, the controlled sharing between the qmail programs occurs primarily through explicit communication channels between pairs of programs.

Let's try another example to help see how to decompose applications for security. We'll design software architecture for a simple web service that converts _les from one format to another. say, from MP3 to OGG.

This web service needs to accept incoming connections to port 80, receive the contents of a _le, perform some complicated computations to translate the _le into the new format, and then send the result back out over the network. Thus, we might break this down into two pieces: the master, a process that receives _les on port 80, invokes the slave with the contents of this _le, and sends the slave's output over the network; and the slave, which takes as input a byte array, interprets its input as an MP3, transforms the MP3 into OGG format, and produces as output another byte array containing the OGG data.

Notice that the slave is computing a deterministic function of its input and does not need any permissions at all (not to read _les, or to access the network, or to do anything else at all except compute). Thus, we can arrange that all the complicated code is sandboxed in a process that cannot do anything at all but pass bytes back and forth to the master. Consequently, if that complicated MP3-to-OGG code is buggy, the worst that can happen is that people will receive incorrect or bogus OGG _les, but no harm can happen to our machine.

Here's another example for you to ponder. You're writing a web browser. You want to decompress a _le that was received from across the network, but the decompression program looks fairly complex and you're not 100% sure whether you can trust it. How do you structure your application to minimize trust in the decompressor? (This is an example with some historical merit. In 2002, it was discovered that any program that used versions of the zlib decompression library dating from Feb 1998.Mar 2002 were vulnerable to a code injection exploit.)

Today, this kind of software design is not especially common, perhaps because common operating systems (including Unix and Windows) don't make it very easy to

get the kind of isolation needed. Nonetheless, as we've seen, there are many applications where sandboxing and isolation would be very useful.

Self-Assessment Exercise (SAE)

1. Mention some decomposition software for security?
2. What are the isolation levels defined by the ANSI/ISO SQL standard?

Self-Assessment Answers (SAA)

4.0 Conclusion

In this study unit, you have learnt about the meaning of entrepreneurship and the entrepreneur. The entrepreneur does not just have good business ideas, but has the courage to make them into a reality by harnessing scarce resources in order to take advantage of opportunities. You have also learnt the various characteristics necessary for anyone with ambitions of becoming a successful entrepreneur.

5.0 Summary

1. Explain what Isolation is
2. Define what is meant by the terms entrepreneur and entrepreneurship

6.0 Tutor-Marked Assignment (TMA)

1. What is Isolation?

2. Explain how Isolation helps in protecting your information.

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html

http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

Dark Reading - Tech Insight: Database Activity Monitoring

Unit 2

Secure Software

Content

1.0 Introduction

The notion of a trusted computing base (TCB); then, we will describe thirteen useful principles. The principles are neither necessary nor sufficient to ensure the design of a secure system, but they are often very helpful nonetheless trusted component is a part of the system that we rely upon to operate correctly, if the whole system is to be secure; to turn it around, a trusted component is one that is able to violate our security goals.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. explain principles of secure software;

ii. define what is meant by the terms the Trusted Computing Base (TCB).

3.0 Learning Contents

3.1 Principles of Secure Software

First, we will show one powerful concept, the notion of a trusted computing base (TCB); then, we will describe thirteen useful principles. The principles are neither necessary nor sufficient to ensure the design of a secure system, but they are often very helpful nonetheless. This lecture focuses primarily on what you can do at design time to improve security. How can you choose an architecture that will help reduce the likelihood of _aws in your system, or increase the likelihood that you will be able to survive such _aws? That's the focus of this lecture.

3.2 The Trusted Computing Base (TCB)

Earlier in this class we introduced the notion of trusted and trustworthy components. A trusted component is a part of the system that we rely upon to operate correctly, if the whole system is to be secure; to turn it around, a trusted component is one that is able to violate our security goals. A trustworthy component is a part of the system that we would be justified in trusting, i.e., where we'd be justified in expecting it to operate correctly. For instance, on Unix systems the super-user (root) is trusted; hopefully she is also trustworthy, or else we are in trouble. In any system, the trusted computing base (TCB) is that portion of the system that must operate correctly, for the security goals of the system to be assured. We have to rely on every component in the TCB to work correctly. However, anything that is outside the TCB isn't relied upon in any way: even if it misbehaves or operates maliciously, it cannot defeat the system's security goals. Indeed, we can take the latter statement as our definition of the TCB: the TCB must be large enough so that nothing outside the TCB can violate security.

Example: Suppose the security goal is that only authorized users are allowed to log into my system using SSH. What is the TCB? Well, the TCB includes the SSH daemon, since it is the one that makes the authentication and authorization decisions. if it has

a bug (say, a buffer overrun), or if it was programmed to behave maliciously (say, the SSH implementer has included a backdoor in it), then it will be able to violate my security goal (e.g., by allowing access to unauthorized users). That's not all. The TCB also includes the operating system, since the operating system has the power to tamper with the operation of the SSH daemon (e.g., by modifying its address space). Likewise, the CPU is in the TCB, since we are relying upon the CPU to execute the SSH daemon's machine instructions correctly. Suppose a web browser application is installed on the same machine; is the web browser in the TCB? Hopefully not! If we've built the system in a way that is at all reasonable, the SSH daemon is supposed to be protected (by the operating system's memory protection) from interference by unprivileged applications, like a web browser. Another example: Suppose that we deploy a firewall at the network perimeter to enforce the security goal that only authorized connections should be permitted into our internal network. Then, in this case, the firewall is the TCB for this security goal.

A third example: When we build access control into a system, there is always some mechanism that is responsible for enforcing the access control policy. As you may remember from the lecture notes on firewalls, this mechanism is known as a reference monitor. The reference monitor is the TCB for security goal of ensuring that the access control policy is followed. Basically, the notion of a reference monitor is just the idea of a TCB, specialized to the case of access control.

In fact, the three guiding principles for a reference monitor also apply to TCB. We repeat them here. A TCB should be:

i. Unbypassable: There must be no way to breach system security by bypassing the TCB.

ii. Tamper-resistant: The TCB should be protected from tampering by anyone else. For instance, other parts of the system outside the TCB should not be able to modify the TCB's code or state. The integrity of the TCB must be maintained.

iii. Verifiable: It should be possible to verify the correctness of the TCB. This usually means that the TCB should be as simple as possible, as generally it is beyond the state of the art to verify the correctness of subsystems with any appreciable degree of complexity.

Keeping the TCB simple and small is good practice. The less code you have to write, the less chances you have to make a mistake or introduce some kind of implementation _aw. Industry standard error rates are 1.5 defects per thousand lines of code. Thus, a TCB containing 1000 lines of code might have 1.5 defects, while a TCB containing 100,000 lines of code might have 100.500 defects. I know which I'd pick. (Windows XP consists of about 40 million lines of code, all of which is in the TCB. Yikes!) The lesson is to shed code: design your system so that as much of the code can be moved outside the TCB.

2 TCBs: What are they good for? Who cares about all this esoteric stuff about TCBs? Actually, the notion of a TCB is a very powerful and pragmatic one. The concept of a TCB allows a primitive yet effective form of modularity. It lets us separate the system into two parts: the part that is security-critical (the TCB), and everything else. This separation is a big win for security. Security is hard. It is really hard to build systems that are secure and correct. The more pieces the system contains, the harder it is to assure its security. If we are able to identify a clear TCB, then we will know that only the parts in the TCB must be correct for the system to be security. Thus, when thinking about security, we can focus our effort where it really matters. And, if the TCB is only a small fraction of the system, we have much better odds at ending up with a secure system: the less of the system we have to rely upon, the less likely that it will disappoint. Let's do a concrete example. You've been hired by the National Archives to help with their email retention system. They're chartered with saving a copy of every email ever sent by government officials. They want to ensure that, once a record is saved, it cannot be subsequently deleted or destroyed. For instance, if someone is investigated, they are worried about the threat that someone might try to destroy embarrassing or incriminating documents previously stored in the archives. The security goal is to prevent this kind of after the- fact document destruction.1 So, you need to build a document storage system which is .append-only. once a document is added to the collection, it cannot be removed. How are you going to do it? One possible approach: You could augment the email program sitting on every government official's desktop computer to save a copy of all emails to some special directory on that computer. What's the TCB for this approach? Well, the TCB includes every copy of the email application on every government machine, as well as the operating systems, other privileged software, and system administrators with root/Administrator level privilege on those machines. That's an awfully large TCB. The chances that everything in the TCB works correctly, and that no part of the TCB can be subverted, don't sound too good. After all, any system administrator could just delete _les from a special directory after the fact. We'd better find a better solution. A different idea: We might set up a high-speed networked line-printer. An email will be considered added to the collection when it has been printed. We might have a giant roll of blank paper, which feeds through the printer. Once the paper is printed, the paper might spool out into some giant canister. We'll lock up the room to make sure no one can tamper with the printouts.

3.3 What's The Trusted Computing Base in This System?

The TCB includes the physical security of the room. Also, it includes the printer. More specifically, suppose we arrange to add a ratchet in the paper spool so that the spool can only rotate in one direction (once printed, the paper feed cannot be reversed). Then the rest of the printer doesn't need to be trusted. Wow! The TCB includes only this one little ratchet gizmo, and the physical security for the room, but nothing else. Neat! That sounds like something we could secure. One problem with this one-way ratcheted printer business is that it involves paper. A lot of paper. (Government

bureaucrats can generate an awful lot of email.) Also, paper isn't keyword-searchable. Instead, let's try to _nd an electronic solution. An all-electronic approach: We set up a separate computer that is networked and runs a special server. The computer accepts connections to that server; when an email is sent over that connection, the computer adds the email to its local file system. The filesystem is carefully implemented to provide write-once semantics: once a _le is created, it can never be overwritten or deleted. We might also set up a packet filter so that no other connections to that server are allowed. What's in the TCB now? Well, the TCB includes the computer running this server, the code of this server application, the operating system and filesystem and other privileged code on this machine, the system administrators of this machine, the packet firewall, the physical security mechanisms (locks and so on) protecting the machine room where this computer is located, and so on. The TCB is a little bigger than with a printer, but it is vastly better than an approach where the TCB includes all the privileged software and privileged users on every government machine. This sounds manageable. I think you've earned your consulting fee.

In summary, some good principles are:

i. know what is in the TCB. Design your system so that the TCB is clearly identifiable;

ii. try to make the TCB as unbypassable, tamper-resistant, and veritable as possible;

iii. keep It Simple, Stupid (KISS). The simpler the TCB, the greater the chances you can get it right; and

iv. decompose for security. Choose a system decomposition/modularization based not just on functionality or performance grounds. choose an architecture that makes the TCB as simple and clear as possible.

Self-Assessment Exercise (SAE)

Please insert self-Assessment Exercise

Self-Assessment Answers (SAA)

Please insert self-Assessment Answers

4.0 Conclusion

In this study unit, you have learnt about the principles of secure software. You have also learnt what is meant by the terms the trusted computing base (tcb) A trusted component is a part of the system that we rely upon to operate correctly, if the whole system is to be secure; to turn it around, a trusted component is one that is able to violate our security goals.

5.0 Summary

1. Explain principles of secure software.
2. Define what is meant by the terms the trusted computing base (tcb).

6.0 Tutor-Marked Assignment (TMA)

1. Give a brief description what is meant by the terms the Trusted Computing Base (TCB)? Explain in details.

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html

http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

Dark Reading - Tech Insight: Database Activity Monitoring

Unit 3

Cryptography

Content

1.0 Introduction

Cryptography is the practice and study of techniques for secure communication in the presence of third parties (called adversaries). More generally, it is about constructing and analyzing protocols that overcome the influence of adversaries and which are related to various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. define what is meant by the terms cryptography

ii. computer era of cryptography

3.0 Learning Contents

3.1 What is Cryptography?

Cryptography is the practice and study of techniques for secure communication in the presence of third parties (called adversaries). More generally, it is about constructing and analyzing protocols that overcome the influence of adversaries and which are related to various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

Cryptography prior to the modern age was effectively synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The originator of an encrypted message shared the decoding technique needed to recover the original information only with intended recipients, thereby precluding unwanted persons to do the same. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.

Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary. It is theoretically possible to break such a system but it is infeasible to do so by any known practical means. These schemes are therefore termed computationally secure; theoretical advances (e.g., improvements in integer factorization algorithms) and faster computing technology require these solutions to be continually adapted. There exist information-theoretically secure schemes that provably cannot be broken even with unlimited computing power—an example is the one-time pad—but these schemes are more difficult to implement than the best theoretically breakable but computationally secure mechanisms.

Cryptology-related technology has raised a number of legal issues. In the United Kingdom, additions to the Regulation of Investigatory Powers Act 2000 require a suspected criminal to hand over their encryption key if asked by law enforcement. Otherwise the user will face a criminal charge. The Electronic Frontier Foundation (EFF) is involved in a case in the Supreme Court of the United States, which may determine whether requiring suspected criminals to provide their encryption keys to law enforcement is unconstitutional. The EFF is arguing that this is a violation of the right of not being forced to incriminate oneself, as given in the Fifth Amendment

3.1.1 History of Cryptography and Cryptanalysis

Before the modern era, cryptography was concerned solely with message confidentiality (i.e. encryption)—conversion of messages from a comprehensible form into an incomprehensible one and back again at the other end, rendering it unreadable by interceptors or eavesdroppers without secret knowledge (namely the key needed for decryption of that message). Encryption was used to (attempt to) ensure secrecy in communications, such as those of spies, military leaders, and diplomats. In recent decades, the field has expanded beyond confidentiality concerns to include techniques for message integrity checking, sender/receiver identity authentication, digital signatures, interactive proofs and secure computation, among others.

Self-Assessment Exercise (SAE)

1. What is Cryptography?
2. Before the modern era, cryptography was only concern with what?

Self-Assessment Answers (SAA)

1. Cryptography is the practice and study of techniques for secure communication in the presence of third parties (called adversaries).
2. message confidentiality (i.e. encryption)—conversion of messages from a comprehensible form into an incomprehensible one and back again at the other end, rendering it unreadable by interceptors or eavesdroppers without secret knowledge (namely the key needed for decryption of that message).

3.2 Computer Era Cryptography

Cryptanalysis of the new mechanical devices proved to be both difficult and laborious. In Great Britain, cryptanalytic efforts at Bletchley Park during WWII spurred the development of more efficient means for carrying out repetitious tasks. This culminated in the development of the Colossus, the world's first fully electronic, digital, programmable computer, which assisted in the decryption of ciphers generated by the German Army's Lorenz SZ40/42 machine.

Just as the development of digital computers and electronics helped in cryptanalysis, it made possible much more complex ciphers. Furthermore, computers allowed for the encryption of any kind of data representable in any binary format, unlike classical

ciphers which only encrypted written language texts; this was new and significant. Computer use has thus supplanted linguistic cryptography, both for cipher design and cryptanalysis. Many computer ciphers can be characterized by their operation on binary bit sequences (sometimes in groups or blocks), unlike classical and mechanical schemes, which generally manipulate traditional characters (i.e., letters and digits) directly. However, computers have also assisted cryptanalysis, which has compensated to some extent for increased cipher complexity. Nonetheless, good modern ciphers have stayed ahead of cryptanalysis; it is typically the case that use of a quality cipher is very efficient (i.e., fast and requiring few resources, such as memory or CPU capability), while breaking it requires an effort many orders of magnitude larger, and vastly larger than that required for any classical cipher, making cryptanalysis so inefficient and impractical as to be effectively impossible.

Credit card with smart-card capabilities. The 3-by-5-mm chip embedded in the card is shown, enlarged. Smart cards combine low cost and portability with the power to compute cryptographic algorithms.

Extensive open academic research into cryptography is relatively recent; it began only in the mid-1970s. In recent times, IBM personnel designed the algorithm that became the Federal (i.e., US) Data Encryption Standard; Whitfield Diffie and Martin Hellman published their key agreement algorithm, and the RSA algorithm was published in Martin Gardner's Scientific American column. Since then, cryptography has become a widely used tool in communications, computer networks, and computer security generally. Some modern cryptographic techniques can only keep their keys secret if certain mathematical problems are intractable, such as the integer factorization or the discrete logarithm problems, so there are deep connections with abstract mathematics. There are no absolute proofs that a cryptographic technique is secure (but see one-time pad); at best, there are proofs that some techniques are secure if some computational problem is difficult to solve, or this or that assumption about implementation or practical use is met.

Modern Cryptography

The modern field of cryptography can be divided into several areas of study. The chief ones are discussed here;

Symmetric-Key Cryptography

Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key (or, less commonly, in which their keys are different, but related in an easily computable way). This was the only kind of encryption publicly known until June 1976.

One round (out of 8.5) of the patented IDEA cipher, used in some versions of PGP for high-speed encryption of, for instance, e-mail.

Symmetric key ciphers are implemented as either block ciphers or stream ciphers. A block cipher enciphers input in blocks of plaintext as opposed to individual characters, the input form used by a stream cipher.

The Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) are block cipher designs which have been designated cryptography standards by the US government (though DES's designation was finally withdrawn after the AES was adopted). Despite its deprecation as an official standard, DES (especially its still-approved and much more secure triple-DES variant) remains quite popular; it is used across a wide range of applications, from ATM encryption to e-mail privacy and secure remote access. Many other block ciphers have been designed and released, with considerable variation in quality. Many have been thoroughly broken, such as FEAL.

Stream ciphers, in contrast to the 'block' type, create an arbitrarily long stream of key material, which is combined with the plaintext bit-by-bit or character-by-character, somewhat like the one-time pad. In a stream cipher, the output stream is created based on a hidden internal state which changes as the cipher operates. That internal state is initially set up using the secret key material. RC4 is a widely used stream cipher; see Category: Stream ciphers Block ciphers can be used as stream ciphers; see Block cipher modes of operation.

Public-Key Cryptography

Symmetric-key cryptosystems use the same key for encryption and decryption of a message, though a message or group of messages may have a different key than others. A significant disadvantage of symmetric ciphers is the key management necessary to use them securely. Each distinct pair of communicating parties must, ideally, share a different key, and perhaps each cipher text exchanged as well. The number of keys required increases as the square of the number of network members, which very quickly requires complex key management schemes to keep them all straight and secret. The difficulty of securely establishing a secret key between two communicating parties, when a secure channel does not already exist between them, also presents a chicken-and-egg problem which is a considerable practical obstacle for cryptography users in the real world.

Whitfield Diffie and Martin Hellman, authors of the first published paper on public-key cryptography In a groundbreaking 1976 paper, Whitfield Diffie and Martin Hellman proposed the notion of public-key (also, more generally, called asymmetric key) cryptography in which two different but mathematically related keys are used—a public key and a private key. A public key system is so constructed that calculation of one key (the 'private key') is computationally infeasible from the other (the 'public key'), even though they are necessarily related. Instead, both keys are generated secretly, as an interrelated pair. The historian David Kahn described public-key cryptography as "the most revolutionary new concept in the field since polyalphabetic substitution emerged in the Renaissance".

In public-key cryptosystems, the public key may be freely distributed, while its paired private key must remain secret. In a public-key encryption system, the public key is used for encryption, while the private or secret key is used for decryption. While Diffie and Hellman could not find such a system, they showed that public-key cryptography was indeed possible by presenting the Diffie–Hellman key exchange protocol, a

solution that is now widely used in secure communications to allow two parties to secretly agree on a shared encryption key.

Diffie and Hellman's publication sparked widespread academic efforts in finding a practical public-key encryption system. This race was finally won in 1978 by Ronald Rivest, Adi Shamir, and Len Adleman, whose solution has since become known as the RSA algorithm.

Self-Assessment Exercise (SAE)

1. What are the several areas of study of the modern cryptography?

Self-Assessment Answers (SAA)

1. Symmetric Key Cryptography
2. Public Key Cryptography

4.0 Conclusion

In this study unit, you have learnt about the meaning of cryptography. The Modern cryptography, Public-key cryptography. You have also learnt the Computer era of cryptography

5.0 Summary

i. Define what is meant by the terms cryptography
ii. Computer era of cryptography

6.0 Tutor-Marked Assignment (TMA)

1. What is meant by the terms cryptography?
2. Give a brief description of the Computer era of cryptography

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html
http://iase.disa.mil/stigs/stig/index.html
http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf
Dark Reading - Tech Insight: Database Activity Monitoring

Module 5

Principles for Secure Systems

Unit 1

Principles for Secure Systems

Content

1.0 Introduction

Understanding the principles for secure system is core knowledge needed to know and understand.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. explain the principles for secure systems.

3.0 Learning Contents

3.1 Principles for Secure Systems

Here are some general principles for secure system design:

Security is economics.

No system is completely, 100% secure against all attacks. Rather, systems may only need to resist a certain level of attack. There is no point buying a $10,000 firewall to protect $1,000 worth of trade secrets.

Also, it is often helpful to quantify the level of effort that an attacker would need to expend to break the system. Adi Shamir once wrote. There are no secure systems, only degrees of insecurity. A lot of the science of computer security comes in measuring the degree of insecurity.

Analogy: Safes come with a rating of their level of security. For instance, a consumer-grade safe might indicate that it will resist attack for up to 5 minutes by anyone without tools. A high-end safe might be rated TL-30: it is secure against a burglar with safecracking tools and limited to 30minutes access to the safe. (With such a safe, we know that we need to hire security guards who are able to respond to any intrusion within 30 minutes.) A corollary of this principle is you should focus your energy on securing the weakest links. Security is like a chain: it is only as secure as the weakest link. Attackers follow the path of least resistance, and they will attack the system at its weakest point. There is no sense putting an expensive high-end deadbolt on a screen door; an attacker isn't going to bother trying to pick the lock when he can just rip out the screen and step through.

Least privilege.

Give a program the set of access privileges that it legitimately needs to do its job. But nothing more, try to minimize how much privilege you give each program and system component. Least privilege is an enormously powerful approach. It doesn't reduce the probability of failure, but it can reduce the expected cost of failures. The less privilege that a program has, the less harm it can do if it goes awry or runs amok. You can think of this as the computer-age version of the shipbuilder's notion of watertight compartments. Even if one compartment is breached, we want to minimize the

damage to the integrity of the rest of the system. For instance, the principle of least privilege can help reduce the damage caused by buffer overruns. If a program is compromised by a buffer overrun attack, then it will probably be completely taken over by an intruder, and the intruder will gain all the privileges the program had. Thus, the fewer privileges that a program has, the less harm is done if it should someday be penetrated by a buffer overrun attack.

Example: How does Unix do, in terms of least privilege? Answer: Pretty lousy. Every program gets all the privileges of the user that invokes it. For instance, if I run an editor to edit a single _le, the editor receives all the privileges of my user account, including the powers to read, modify, or delete all my _les. That's much more than is needed; strictly speaking, the editor probably only needs access to the _le being edited to get the job done.

Example: How is Windows, in terms of least privilege? Answer: Just as lousy. Arguably worse, because many users run under an Administrator account, and many Windows programs require that you be Administrator to run them. In this case, every program receives total power over the whole computer. Folks on the Microsoft security team have recognized the risks inherent in this and are taking many steps to warn people away from running with Administrator privileges, so things are getting better in this respect.

Use fail-safe defaults.

Use default-deny polices. Start by denying all access, then allow only that which has been explicitly permitted. Ensure that if the security mechanisms fail or crash, they will default to secure behaviour, not to insecure behaviour.

Example: A packet filter is a router. If it fails, no packets will be routed. Thus, a packet filter fails safe. This is good for security. It would be much more dangerous if it had fail-open behaviour, since then all an attacker would need to do is wait for the packet filter to crash (or induce a crash) and then the fort is wide open.

Example: Long ago, SunOS machines used to ship with + in their /etc/hosts.equiv, which allowed anyone with root access on any machine on the Internet to log into your machine as root. Irix machines used to ship with xhost + in their XWindows con_guration _les by default. This violates of the principle of fail-safe defaults, since the machines came with an out-of-the-box configuration that was insecure by default.

Separation of responsibility

Split up privilege, so no one person or program has complete power. Require more than one party to approve before access is granted. Examples: In a nuclear missile silo, two launch officers must agree before the missile can be launched. Example: In a movie theatre, you pay the teller and get a ticket stub; then when you enter the movie theatre, a separate employee tears your ticket in half and collects one half of it, putting it into a lockbox. Why bother giving you a ticket that 10 feet later is going to be collected from you? One answer is that this helps prevent insider fraud. Tellers are low-paid employees, and they might be tempted to under-charge a friend, or to over-charge a

stranger and pocket the difference. The presence of two employees helps keep them both honest, since at the end of the day, the manager can reconcile the number of ticket stubs collected against the amount of cash collected and detect some common shenanigans.

Example: In many companies, purchases over a certain amount must be approved both by the requesting employee and by a separate purchasing office. This control helps prevent fraud, since it is less likely that both will collude and since it is unlikely that the purchasing office will have any conflict of interest in the choice of vendor.

Defence in depth

This is a closely related principle. There's a saying that you can recognize a security guru who is particularly cautious if you see someone wearing both a belt and a set of suspenders. (What better way to avoid getting caught with your trousers around your ankles?) The principle is that if you use multiple redundant protections, then all of them would need to be breached before the system's security will be endangered.

Psychological acceptability.

It is important that your users buy into the security model. Example: Suppose the company _firewall administrator gains a reputation for capriciously, for no good reason, blocking applications that the engineers need to use to get their job done. Pretty soon, the engineers are going to view the _firewall as damage and route around it, maybe setting up tunnels, or bypassing it in any number of other ways. This is not a game that the _firewall administartor is going to win. No system can remain secure for long when all its users actively seek to subvert it. Example: The system administrator issues an edict that, henceforth, all passwords will be automatically generated unmemorizable strings that are at least 17 characters long, and must be changed once a month. What's likely to happen is that users will simply write down their password on a yellow sticky attached to their monitor, visible to anyone who looks. Such well-intentioned edicts can ultimately turn out to be counter-productive.

13 Principles Secure Systems

1) Secure the weakest link -- Spaf (that is, highly respected security expert Gene Spafford of Purdue University) teaches this principle with a funny story. Imagine you are charged with transporting some gold securely from one homeless guy who lives in a park bench (we'll call him Linux) to another homeless person who lives across town on a steam grate (we'll call her Android). You hire an armored truck to transport the gold. The name of the transport company is "Applied Crypto, Inc." Now imagine you're an attacker who is supposed to steal the gold. Would you attack the Applied Crypto truck, Linux the homeless guy, or Android the homeless woman? Pretty easy experiment, huh? (Hint: the answer is, "Anything but the crypto.")

2) Defend in depth – Author and consultant Kenneth van Wyk likes to call this one the "belt and suspenders" approach. Redundancy and layering is usually a good thing in security. Don't count on your firewall to block all malicious traffic; use an intrusion detection system as well. If you are designing an application, prevent single points of

failure with security redundancies and layers of defense. From Building Secure Software, "The idea behind defense in depth is to manage risk with diverse defensive strategies, so that if one layer of defense turns out to be inadequate, another layer of defense will hopefully prevent a full breach." It's a concept preached universally by information security experts, and for good reason: it works.

3) Fail securely -- Make sure that any system you design does not fail "open." My favorite story about this principle comes from the ill-fated Microsoft Bob product of yesteryear. (Bob was the precursor of Clippy the paperclip.) According to legend, if you failed to get your username and password right after three attempts, Bob would helpfully notice and ask whether you wanted to pick a new password to use. Thanks Bob (said the hacker)! Obviously, a better default in this situation is to deny access.

From Building Secure Software, "Any sufficiently complex system will have failure modes. Failure is unavoidable and should be planned for. What is avoidable are security problems related to failure. The problem is that when many systems fail in any way, they exhibit insecure behavior."

4) Grant least privilege -- When you do have to grant permission for a user or a process to do something, grant as little permission as possible. Think about your Outlook contacts. If you need someone to have access to your contacts to see some data, grant them reader permission, but do not grant them edit permission. Or if you want a geekier example, try this: most users of a system should not need root permission for their everyday work, so don't give it to them. Bottom line, avoid unintentional, unwanted, or improper uses of privilege by doling it out in a miserly fashion.

5) Separate privileges -- I once saw a system that divided its authentication front end into an impressive number of roles with different degrees of access to the system. The problem was that when a user of any role had to perform a back-end database action, the software granted each user de-facto administrator privilege temporarily. Not good. Even the lowliest intern could blitzkrieg the database.

Know that if an attacker is able to finagle one privilege but not a second, she may not be able to launch a successful attack. Keep privilege sets apart.

6) Economize mechanism -- Complexity is the enemy of security engineering and the friend of the attacker. It's just too easy to screw things up in a complicated system, both from a design perspective and from an implementation perspective. The irony: Want to see something complicated? Check out just about any piece of modern enterprise software!

Do what you can to keep things simple. From Building Secure Software, "The KISS mantra is pervasive: 'Keep It Simple, Stupid!' This motto applies just as well to security as it does everywhere else. Complexity increases the risk of problems. Avoid complexity and avoid problems."

7) Do not share mechanisms -- Should you plunk your inward-facing business application on the public cloud? Probably not, according to this principle. Why have

your authentication system deal with random Internet traffic when you can limit it to employees who you (supposedly) trust?

Here's a geekier example. If you have multiple users using the same components, have your system create different instances for each user. By not sharing objects and access mechanisms between users, you will lessen the possibility of security failure.

8) Be reluctant to trust -- Assume that the environment where your system operates is hostile. Don't let just anyone call your API, and certainly don't let just anyone gain access to your secrets! If you rely on a cloud component, put in some checks to make sure that it has not been spoofed or otherwise compromised. Anticipate attacks such as command-injection, cross-site scripting, and so on.

This principle can get tricky fast. From Building Secure Software, "One final point to remember is that trust is transitive. Once you dole out some trust, you often implicitly extend it to anyone the trusted entity may trust."

9) Assume your secrets are not safe -- Security is not obscurity, especially when it comes to secrets stored in your code. Assume that an attacker will find out about as much about your system as a power user, maybe more. The attacker's toolkit includes decompilers, disassemblers, and any number of analysis tools. Expect them to be aimed at your system. Ever look for a crypto key in binary code? An entropy sweep can make it stick out like a sore thumb. Binary is just a language.

10) Mediate completely -- Every access and every object should be checked, every time. Make sure your access control system is thorough and designed to work in the multi-threaded world we all inhabit today. Whatever you do, make sure that if permissions change on the fly in your system, that access is systematically rechecked. Don't cache results that grant authority or wield authority. In a world where massively, distributed systems are pervasive and machines with multiple processors are the norm, this principle is a doozy to think about.

11) Make security usable -- If your security mechanisms are too odious, your users will go to great length to circumvent or avoid them. Make sure that your security system is as secure as it needs to be, but no more. If you affect usability too deeply, nobody will use your stuff, no matter how secure it is. Then it will be very secure, and very near useless.

Spaf has always laughed at the line that mentions how the most secure system in the world is one with its hard drive demagnetized that is buried in a 30-foot hole filled with concrete poured around a Faraday grid. Such a system is, ahem, difficult to use.

12) Promote privacy -- Yeah, I know, everybody talks about privacy, but most people don't actually do anything about it. You can help fix that. When you design a system, think about the privacy of its ultimate users. Are you collecting personally identifiable information (PII) just because somebody from the marketing team said to do so? Is it a good thing to do? Do you store PII in a place where it can be compromised? Shouldn't that be encrypted? Information security practitioners don't

always have to provide the answers to these privacy questions (that's what CIOs get paid for), but it's important for infosec to put forth these kinds of questions if no one else does.

13) Use your resources -- As I was taught in troop leader development class when I was 14, "use your resources" is a principle with incredibly wide application. If you're not sure whether your system design is secure, ask for help. Architectural risk analysis is hard, but there are people who have been doing it well for decades. Don't try to go it alone if you can't. And don't feel bad about asking for help; this stuff is tricky

Self-Assessment Exercise (SAE)

1. What are the general principles for secure systems?

Self-Assessment Answers (SAA)

Find answer in page 93-96

4.0 Conclusion

In this study unit, you have learnt about the principles for secure systems. This principle will be able to tell you analysis and secure different systems.

5.0 Summary

i. Explain the principles for secure systems

6.0 Tutor-Marked Assignment (TMAs)

1. List and Explain 4 principles for secure systems

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html

http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

Dark Reading - Tech Insight: Database Activity Monitoring

Unit 2

Operating System Based Security

Content

1.0 Introduction

Operating system provides an interface between the hardware and programs executing on this hardware, and also an interface to the human user here the hardware includes the CPU and the memory. The abstractions that the operating system provides and the illusion of each process having exclusively access to its own CPU and memory that it tries to provide {are not perfect. Such imperfections may lead to security vulnerabilities.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. Explain what an Operating system based security is
ii. Define what is meant by the terms Operating system abstractions and access control
iii. Describe the Imperfections in abstractions
iv. Explain what goes wrong

3.0 Learning Contents

3.1 Operating System Based Security

Every modern computer system, from network servers, workstation desktops, to laptops and handheld devices, has a core piece of software, called kernel or operating system, executed on the top of a bare machine of hardware that allocates the basic resources of the system (e.g., CPU, memory, device driver, communication port, etc.), and supervises the execution of all applications within the system. Some popular commercial and Open Source operating systems are Microsoft Windows, different flavors of Unix (BSD, AIX, HP, UX, Solaris, etc.), Mac OS, and Linux

Before we consider what role, the programming language can play in providing security, it is useful to take a step back and consider the role an operating system (OS) traditionally plays in providing security, and look at:

1. the security mechanisms that operating systems traditionally provide, through separation and various forms of access control;
2. what the basis for this access control is;
3. the role that abstractions play here.

Because of the crucial role of the operating system in the operation of any computer systems, the security (or lack of security) of an operation system will have fundamental impacts to the overall security of a computer system, including the security of all applications running within the system. A compromise of the underneath operating system will certainly expose danger to any application running in the system. Lack of proper control and containment of execution of individual applications in an operating system may lead to attack or break-in from one application to other applications Based

on the “Trusted Computer System Evaluation Criteria” of US government, the security level of most commercially available operating systems are no higher than C2 class, which requires Discretionary Access Control (DAC) Key fingerprint = AF19 FA27 2F94 998D FDB5 DE3D F8B5 06E4 A169 4E46 protection at a per user granularity. Although this level of protection provides safeguard of certain extent among different applications in a multitasking, time sharing environment that is

typical for current mainstream operating systems, no mechanisms are supported by operating systems in this class to enforce strict security policies of individual applications. As a result, in a C2 class operating system the security of applications and users are responsible for their own fates

3.2 Operating System Abstractions and Access Control

The operating system provides an interface between the hardware and programs executing on this hardware, and also an interface to the human user. Here the hardware includes the CPU and the memory, plus a range of other I/O devices, typically including a hard disk, a keyboard, a display, and a network connection. The operating system provides useful abstractions to hide a lot of the underlying complexity. For example, the operating system uses segmentation and page tables to provide virtual memory, and programs can then use a virtual address space for storing data without having to worry about space for storing data without having to worry about where in memory or where on the hard disk the data actually resides. Another example is that the operating system provides a le system to programs and users, which hides the details of how the les are stored over various sectors of the hard disk. The central abstraction that the operating system provides is that of a process.

Each program in execution gives rise to a process, and the operating system tries to create the illusion that this process has exclusive access to the CPU (through time slicing) and to the computer's memory, and typically too far more memory than is physically available as RAM (through segmentation and paging). As part of this abstraction, the operating system tries to ensure separation between the different processes, so that programs cannot interfere with each other. The prime example here is that processes have their own address space that others cannot touch, which is obviously crucial for security. A process that tries to access memory outside its address space will crash with a so-called segmentation fault. This way the operating system provides some encapsulation of processes.

Processes can still interfere with each other when competing for the shared resources of the computer: e.g., a program that claims a lot of memory or a lot of CPU time could interfere with the progress that another process can make. The operating system also provides security by enforcing access control for many of the abstractions it introduces1. The prime example here is the le system, where different users {and the programs they start {have different access rights for reading or writing certain les.

Traditionally, access control provided by the operating system is organized by user and by process, where processes typically inherit the access rights of the user that started them, although modern operating systems will over some ways to tweak this illustrates the security mechanisms discussed above. The abstractions provided by the operating system play a crucial role in any access control for them: if someone gets access to the raw bits on the hard drive, any access control provided by the other code does not have. Any bugs in this code can lead to serious security vulnerabilities; this code has to be trusted, and hopefully it is trustworthy.

This has led to the distinction between user mode and kernel mode, which most operating systems provide. Part of the code that belongs to the operating system is executed in kernel mode, giving it complete and unrestricted access to the underlying hardware. So the access rights of a process do not only depend on the user that started that process, but also on whether the process is executing kernel or user code, which may vary in time.

3.3 Imperfections in Abstractions

The abstractions that the operating system provides {and the illusion of each process having exclusively access to its own CPU and memory that it tries to provide} are not perfect. Such imperfections may lead to security vulnerabilities. For example, a process may be able to detect that some of its memory is in fact swapped out to disk, based on the response time of certain operations. This property was used to break password security of the TENEX, an operating system developed in the early 1970s: depending on the response time of some incorrect password guess, it was possible to work out if the rst n characters of the password guess were correct, by making sure only these rst n characters were in main memory, and the rest were swapped to disk. Another example is that by observing the contents of freshly allocated memory, a process can observe the contents of memory that used to belong to other processes. This is a classic way of snooping on other users in multi-user systems. An example given in [GVW03] shows the unexpected ways in which these im- perfections can lead to security problems. In 1993 it was observed that every tar- ball2 produced on UNIX Solaris .0 contained a fragment of the password le {/etc/passwd on UNIX systems {at the very end. The way that this happened was as follows:

1. To nd out the owner and group permissions for the tar le to be created, the password le was consulted. (The password le tells which group a user belongs to.) This meant this le was read from disk and stored in RAM. This RAM was then released.
2. Then the tar le was created. For this RAM memory was allocated. Because of the way memory allocation worked, this memory included the memory that had just before been used for the password le. And because memory is not wiped on allocation or de-allocation, this memory still contained contents of the password le.
3. The size of a tar le is always a multiple of a xed block size. Unless the actual contents size is precisely a multiple of this block size, the block at the end will not

be completely led. The remainder of this block still contained data from the password le. As tar les are typically used to share les over the internet the security implications are clear. Fortunately, in Solaris 2.0 the le /etc/passwd no longer contained hashed and salted passwords, these were in a separate le (the shadow password le).

A quick x to the problem is replacing a call to malloc (which allocates memory) in the code of tar by a call to calloc (which allocates memory and zeroes itout). That only xes this particular instance of the problem, and people always use malloc by default rather than calloc because it is faster. One can think about more fundamental solutions to this kind of problem, e.g., always zeroing out memory on allocation, or always zeroing out memory upon de-allocation if the memory contained sensitive data.

3.4 What Goes Wrong?

The way that security vulnerabilities normally arise in operating systems is more mundane than the tarball example above. It typically happens due to nd out the owner and group permissions for the tar le to be created, les are stored over various sectors of the hard disk. The central abstraction that the operating system provides is that of a process. Each program in execution gives rise to a process, and the operating system tries to software bugs especially buer overow weaknesses in high priority library calls by the op-erating system are notorious examples of software bugs.Modern operating systems come with very large APIs for user applications touse. Many of the system calls in this interface run with very high privileges. The standard example is the login function: login is invoked by a user who has not been authenticated yet, and who should therefore not be trusted and only be given minimal permission; however, the login procedure needs access to the password le to check if the given password guess is correct, so needs very high privileges. A buer overow weakness in login can possibly be exploited by an attacker to do other things with these privileges.

Complexity

Even if the operating system API was implemented without any bugs, the sheer complexity of modern operating systems means that users will get things wrong or run into unforeseen and unwanted feature interaction, where inter-play of various options introduces security vulnerabilities. Example, introductory textbooks on operating systems typically illustrate the idea of operating system access control with read, write, and execute permissions, with users organised in groups, and one super-user or root who has permissions to do anything; but real-life modern operating systems o er dozens of permissions and all sorts of user groups with di erent privilege levels. The advantage is that this enables very ne-grained access control, the disadvan-tage is that people get it wrong and grant unwanted access right. for an interesting account of how major software vendors got Windows access control wrong.

4.0 Conclusion

In this study unit, you have learnt Explain why this is an Operating system based security. You also have learnt Define what is meant by the terms Operating system abstractions and access control. You have also learnt the various Imperfections in abstractions

5.0 Summary

i. Explain what an Operating system based security is.

ii. Define what is meant by the terms Operating system abstractions and access control.

iii. Describe the Imperfections in abstractions.

iv. Explain what goes wrong.

6.0 Tutor-Marked Assignment (TMAs)

1. Define what is meant by the terms Operating system abstractions and access control Describe the Imperfections in abstractions

2. Describe the Imperfections in abstractions.

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html

http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

Dark Reading - Tech Insight: Database Activity Monitoring

Unit 3

Safe Programming Languages

Content

1.0 Introduction

This unit investigates what safety means here, and discusses the various favours of safety, such as memory safety, type safety, and thread safety. Precisely pinning down what safety means is tricky, and people have different opinions on precisely what constitutes a safe programming language. An alternative characterisation, which is a bit more concrete, is: In a safe programming language, programs always have a precise and well-dened semantics.

2.0 Learning Outcomes

After studying the material in this unit, you should be able to:

i. Explain the term Safe programming languages

ii. Define what is meant by the terms Network Safety

iii. Describe the Safety security and Memory safety

3.0 Learning Contents

3.1 Safe Programming Languages

A fundamental way in which a programming language can help in writing secure programs is by being `safe'. This unit investigates what safety means here, and discusses the various favours of safety, such as memory safety, type safety, and thread safety. Precisely pinning down what safety means is tricky, and people have different opinions on precisely what constitutes a safe programming language. Usually, by a safe programming language people mean one that provides memory safety and type safety, but there are other forms of safety, as we will see. Safe programming languages provide some guarantees about the possible behaviour of programs, which can protect against security vulnerabilities due to unwanted behaviour. In essence, the central idea is: In a safe programming language, you can trust the abstractions provided by the language.

An alternative characterisation, which is a bit more concrete, is: In a safe programming language, programs always have a precise and well-dened semantics. An important goal of safety is that safety makes it possible to reason about programsin a modular way: i.e., to make it possible to make guarantees about the behaviour of certain `parts' of a program without knowing the rest of it. Here, depending on the programming languages, these `parts' can be functions and procedures, objects, classes, packages, or some other notion of module. In a safe programming language, it is possible to understand the behaviour of a program in a modular way, and to make guarantees about part of a program by inspecting only that part and not the entire pro-gram.

The following sections will explain these general ideas in more detail and give more speci c examples. We begin by considering the issue of undenedness, before turning to the more speci c notions of memory safety and type safety.

3.2 Network Safety

Many program statements only make sense under certain circumstances. For example, the statement a[i] = (float) x; only makes sense if in the current execution state, a is an array in which we can store oats, i has an integer value within the array bounds, and x can sensibly be converted into an oat value. If one of these conditions is not met, then it is unclear what the semantics of this statement should be. There are roughly two approaches that programming languages can take here:

1. One approach is to accept that a program statement may be executed when it does not really make sense. The semantics of the statement is undened in these cases: essentially, anything may happen. It is then the obligation of the programmer to ensure that statements are only ever executed in situations where they make sense. What actually will happen in the undened cases is down to the compiler and the platform responsible for the actual execution. This platform is the raw hardware (i.e., the CPU) in the case the compiler produces native code. It can also be the (software) execution engine used to execute programs in some intermediate language (e.g., a VM executing bytecode) and the hardware on top of which this engine runs;

2. The other approach is that the language ensures that a statement is only ever executed when it makes sense, or, when it does not, signals some error in a precisely dened manner, for example by throwing an exception. It is now the obligation of the language to somehow prevent or detect the execution of statements when this does not make sense. This can be through measures at compile-time (e.g., type checking), at run-time (by some execution engine that monitors for these conditions, or by the compiler including some checks in the code it produces), or a combination of the two. The first approach leads to an unsafe programming language. The second approach eads to a safe programming language. In safe languages, the semantics of a program {or any program fragment {is always precisely de ned; even in cases where it does not make sense, it is precisely defined what error will be reported.

C and C++ are the prime examples of unsafe languages. Java and C# are meant to be safe, but still have some unsafe features. For instance, the Java Native Interface (JNI) allows Java programs to call native machine code. In C# pointer arithmetic is allowed in code blocks that are marked as unsafe. So, Java and C# programs are only safe if these features are not used. Most functional programming languages, such as Haskell, ML, or LISP, are safe, as are most logic programming languages, such as Prolog.

The main advantage of the unsafe approach is speed. Execution of a safe language typically involves some run-time monitoring that slows execution down. An important disadvantage is security. If the programmer is not very careful and just one program statement is executed in a state where it does not make sense, we have no way of knowing what may happen: it may re-format the hard disk, sent all your private email correspondence to wikileaks, etc. This means we can no longer make any security

guarantees about the program. Conicts between security and some practical consideration such as speed (or`convenience') are common. Almost always people sacrifice security in favour of the short-term practical advantage, and then live to regret the consequences in the long term.

3.3 Safety and Security

It is possible to write insecure programs in any programming language. But without safety, writing secure programs gets a lot harder. Consider a procedure login (uname,passwd) written in some unsafe program-ming language. Only if this procedure is very carefully written to check for any of the error conditions that would lead to undened behaviour, we can make some guarantees about it (for instance to return true i it is given a matching username and password). If not, then we need to know the precise circumstances in which the procedure is called in order to make any guarantees. In the best case, this is a lot of work: we have to check all the places in the code where login is called, and check that these calls cannot trigger any unde ned behaviour. When doing this, we should be careful not to make any assumptions on user input that the program may receive. For non-trivial programs this quickly becomes infeasible to check. Note that this reveals a fundamental problem with unsafe languages, apart from any security issues: we cannot always understand code in a modular fashion. E.g., we may not be able to know what the procedure login will do without knowing the precise context in which it is called. Things get worse if the procedure is part of some library that may be called from many places, or if login is an operating system call that may be called by any application program. In these cases, we do not know the specic circumstances of calls to login, so we cannot rely on these to avoid running into undened behaviour.

3.4 Memory Safety

One important form of safety is memory safety. A programming language is memory-safe if programs are guaranteed to only access memory locations that they are permitted to access. Language features that break memory safety include:

a. Pointer arithmetic;

b. unconstrained casting (e.g., casting a oating-point number to an array);

c. lack of array bounds checks;

d. programmer-controlled de-allocation (as this allows dangling pointers to mem-ory that has been de-allocated).

Memory safety is a special case of the notion of safety as discussed in section 3.1: accessing some random memory location that you shouldn't will not have a well-dened semantics, and can break any abstractions the language provides. Memory unsafety makes it hard { if not impossible { to understand programs in a modular fashion. Consider a program consisting of two modules, P and Q, written in a language that is not memory safe. Code belonging to module Q can makes changes anywhere in memory, including data belonging to module P. So, code belonging to module Q could

corrupt data that module P relies on, for instance breaking data invariants. This means we cannot make guarantees about P just by looking at P, we also have to make sure Q does not interfere with P. Note that this means we cannot make guarantees about extensible programs, as any extension could corrupt existing behaviour.

If a language is memory-safe then programs can never crash with a segmentation fault. One might even consider it safe to switch to the checks the operating system performs on memory access for these programs. However, the execution of the memory-safe program might rely on some interpreter or execution engine that is written in an unsafe language, which could still cause out-of-bounds memory access. Also, memory safety might rely on the correctness of compilers and type checkers, which may contain bugs. So, in line with the security principle of Defence in Depth, keeping the memory access control performed by the operating system switched on is always a wise thing to do.

Self-Assessment Exercise (SAE)

1. What are the language features that break memory safety?

Self-Assessment Answers (SAA)

Find answer in Page 106

4.0 Conclusion

In this study unit, you have learnt about the meaning of Safe programming languages. A fundamental way in which a programming language can help in writing secure programs is by being `safe'. You have also learnt the terms Network Safety. You have also learnt the term Safety security and Memory safety.

5.0 Summary

i. Explain the term Safe programming languages

ii. Define what is meant by the terms Network Safety

iii. Describe the Safety security and Memory safety

6.0 Tutor-Marked Assignment (TMAs)

1. Explain the term Safe programming languages
2. Define what is meant by the terms Network Safety

7.0 References/Further Reading

http://iase.disa.mil/stigs/checklist/index.html

http://iase.disa.mil/stigs/stig/index.html

http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

Dark Reading - Tech Insight: Database Activity Monitoring