CIPHER CHRONICLES: UNRAVELING CYBER THREATS IN THE MODERN AGE

Dr. S. Suganyadevi
Ms. M. Nivedha
Ms. Ganga T G

Cipher Chronicles: Unraveling Cyber Threats in the Modern Age

EMPYREAL
PUBLISHING HOUSE

India | UAE | Nigeria | Uzbekistan | Montenegro | Iraq | Egypt | Thailand | Uganda | Philippines | Indonesia

www.empyrealpublishinghouse.com

Cipher Chronicles: Unraveling Cyber Threats in the Modern Age

By:

Dr. S. Suganyadevi
Assistant Professor of Computer Science at Sri Krishna Arts and Science College in Coimbatore, Tamil Nadu, India

Ms. M. Nivedha
Assistant Professor of Computer Science at Sri Krishna Arts and Science College in Coimbatore, Tamil Nadu, India

Ganga T G
Assistant Professor of Computer Applications at Nilgiri college of Arts and Science, Thaloor, Tamilnadu India

First Impression: August 2023

Cipher Chronicles: Unraveling Cyber Threats in the Modern Age

ISBN: 978-93-93810-96-0

Rs. 1000/- ($80)

No part of the book may be printed, copied, stored, retrieved, duplicated and reproduced in any form without the written permission of the editor/publisher.

DISCLAIMER

Information contained in this book has been published by Empyreal Publishing House and has been obtained by the editors from sources believed to be reliable and correct to the best of their knowledge. The authors are solely responsible for the contents of the articles compiled in this book. Responsibility of authenticity of the work or the concepts/views presented by the authors through this book shall lie with the authors and the publisher has no role or claim or any responsibility in this regard. Errors, if any, are purely unintentional and readers are requested to communicate such error to the author to avoid discrepancies in future.

Published by:
Empyreal Publishing House

PREFACE

In today's fast-changing tech world, Information Security is a key part of our digital society. As we move through the era of cybersecurity, keeping sensitive data safe has become really important. From early computer security to complex operating system and network defense, our ongoing mission is to protect our digital world from ongoing threats, showing how smart humans can be.

When computers came along, new risks emerged, leading to computer security. This area looks after hardware, software, and data to stop unauthorized access, tampering, or damage. Operating system security is important too, focusing on making the software that controls how hardware and apps work more secure.

In our digital world, network security is like a safety net for information. With more internet and devices connecting, we see where networks can be vulnerable. Network security experts work hard to block cyber threats, so data stays private, safe, and accessible.

Trust in digital talk and transactions relies on cryptographic protocols and checks. These tools make sure communication, data sharing, and user identification are secure. They use clever math, encryption, and digital signatures to create a safe base for online transactions and private data sharing.

This introduction sets the scene for our exploration of the wide world of Information Security. As we go through this landscape, we'll look deeper into each part, discovering the challenges, smart ideas, and tricky parts that shape our plans to stay safe from cyber threats in our connected global world.

ACKNOWLEDGEMENT

This is about a topic that has become increasingly vital in our digital age: Information Security. In a world where technology permeates every aspect of our lives, safeguarding our digital assets and personal information is of utmost importance.

In light of this, I wish to convey my heartfelt gratitude to all those who have played a pivotal role in the triumphant realization within the realm of information security.

I am indebted to Sri Krishna Arts and Science College for providing the necessary resources and facilities that facilitated this process.

I would also like to acknowledge the contributions of my colleagues and peers who engaged in stimulating discussions and provided a supportive environment for intellectual growth.

Last but not least, I extend my gratitude to my family and friends for their unwavering encouragement and understanding during the challenges.

This work would not have been possible without the collective efforts and support of all these individuals and entities. Thank you for being an integral part of this journey toward advancing our understanding of information security in the era of cyber security.

Dr. S. Suganyadevi

Ms. M. Nivedha

Ms. Ganga T G

Table of Contents

Preface	**IV**
Acknowledgement	**V**
Table of Contents	**VI**

Title of the chapter	**Page No.**
INTRODUCTION TO COMPUTER SECURITY	1 – 39
OPERATING SYSTEM SECURITY	40 – 60
NETWORK SECURITY	61 – 78
SOFTWARE SECURITY	79 – 88

 ISBN: 978-93-93810-96-0

CHAPTER - 1

INTRODUCTION TO COMPUTER SECURITY

1. INTRODUCTION

Computer Security is the protection of computing systems and the data that they store or access. It's very important for protecting personal and sensitive information.

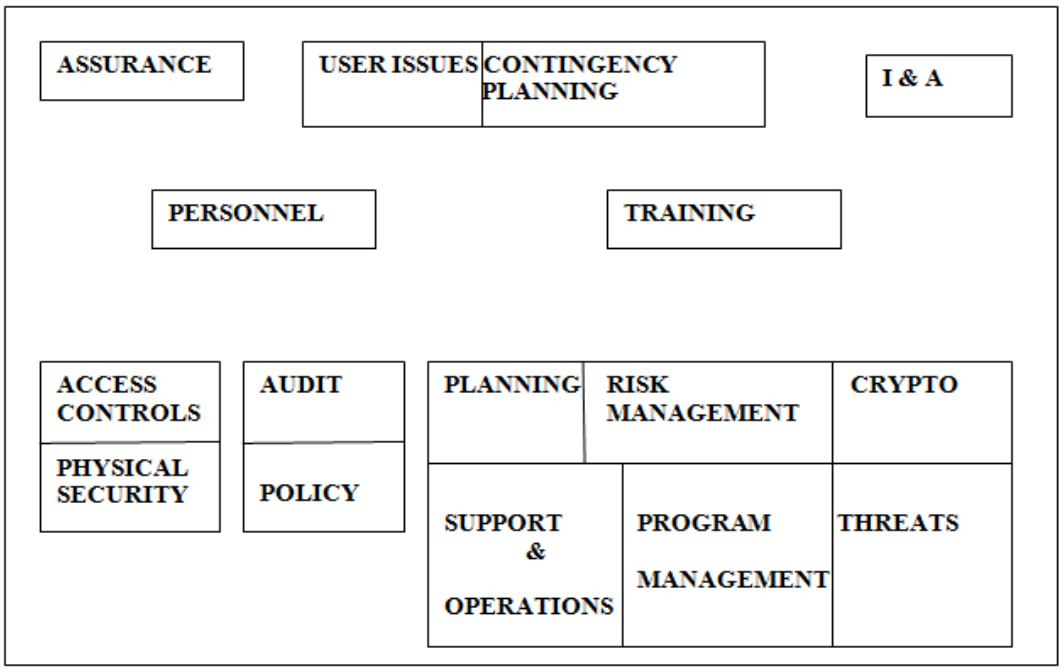


Fig 1.1 Computer Securities

The Computer Security Act of 1987 requires agencies to identify sensitive systems, conduct computer security training, and develop computer security plans.

1.2 BASIC CONCEPTS

General approach to computer security is based on eight major elements:

1. Computer security should support the mission of the organization.

2. Computer security is an integral element of sound management.

3. Computer security should be cost-effective.

4. Computer security responsibilities and accountability should be made explicit.

5. System owners have computer security responsibilities outside their own organizations.

6. Computer security requires a comprehensive and integrated approach.

7. Computer security should be periodically reassessed.

 ISBN: 978-93-93810-96-0

8. Computer security is constrained by societal factors.

Computer Security Supports the Mission of the Organization: The purpose of computer security is to protect an organization's valuable resources, such as information, hardware, and software. they are put in place to protect important assets and thereby support the overall organizational mission.

Computer Security is an Integral Element of Sound Management: Information and computer systems are often critical assets that support the mission of an organization. Protecting them can be as critical as protecting other organizational resources, such as money, physical assets, or employees.

When an organization's information and computer systems are linked with external systems, management's responsibilities also extend beyond the organization. This may require that management

(1) Know what general level or type of security is employed on the external system(s) or

(2) seek assurance that the external system provides adequate security for the using organization's needs.

Computer Security Should Be Cost-Effective: The costs and benefits of security should be carefully examined in both monetary and nonmonetary terms to ensure that the cost of controls does not exceed expected benefits. Security should be appropriate and proportionate to the value of and degree of reliance on the computer systems and to the severity, probability and extent of potential harm. Requirements for security vary, depending upon the particular computer system.

Computer Security Responsibilities and Accountability Should Be Made Explicit: The responsibilities and accountability of owners, providers, and users of computer systems and other parties11 concerned with the security of computer systems should be explicit. The assignment of responsibilities may be internal to an organization or may extend across organizational boundaries.

Systems Owners Have Security Responsibilities Outside Their Own Organizations: If a system has external users, its owners have a responsibility to share appropriate knowledge about the existence and general extent of security measures so that other users can be confident that the system is adequately secure.

Computer Security Requires a Comprehensive and Integrated Approach: Providing effective computer security requires a comprehensive approach that considers a variety of areas both within and outside of the computer security field. Information lifecycle is

1. **Interdependencies of Security Controls**
2. **Other Interdependencies**

Computer Security Should Be Periodically Re assessed: Computers and the environments they operate in are dynamic. System technology and users, data and information in the systems, risks associated with the system and, therefore, security requirements are ever-changing.

Computer Security is constrained by Societal Factors: The ability of security to support the mission of the organization(s) may be limited by various factors, such as social issues. For example, security and workplace privacy can conflict. Security can also increase the privacy afforded to an individual or help achieve other goals set by society.

1.3 THREAT MODELS – COMMON SECURITY GOALS

Computer systems are vulnerable to many threats that can inflict various types of damage resulting in significant losses. This damage can range from errors harming database integrity to fires destroying entire computer centers.

To control the risks of operating an information system, managers and users need to know the vulnerabilities of the system and the threats that may exploit them. Knowledge of the threat environment allows the system manager to implement the most cost-effective security measures. In some cases, managers may find it more cost-effective to simply tolerate the expected losses.

Such decisions should be based on the results of a risk analysis.

1.3.1Errors and Omissions

Errors and omissions are an important threat to data and system integrity. These errors are caused not only by data entry clerks processing hundreds of transactions per day, but also by all types of users who create and edit data. Many programs, especially those designed by users for personal computers, lack quality control measures. Users, data entry clerks, system operators, and programmers frequently make errors that contribute directly or indirectly to security problems.

Errors can occur during all phases of the systems life cycle. A long-term survey of computer-related economic losses conducted by Robert Courtney, a computer security consultant and former member of the Computer System Security and Privacy Advisory Board, found that 65 percent of losses to organizations were the result of errors and omissions. Programming and development errors, often called **"bugs,"** can range in severity from benign to catastrophic.

 ISBN: 978-93-93810-96-0

Installation and maintenance errors are another source of security problems. For example, an audit by the President's Council for Integrity and Efficiency (PCIE) in 1988 found that every one of the ten mainframe computer sites studied had installation and maintenance errors that introduced significant security vulnerabilities.

1.3.2 Fraud and Theft

Computer systems can be exploited for both fraud and theft both by "automating" traditional methods of fraud and by using new methods. For example, individuals may use a computer to skim small amounts of money from a large number of financial accounts, assuming that small discrepancies may not be investigated. Computer fraud and theft can be committed by insiders or outsiders.

The U.S. Department of Justice's Computer Crime Unit contends that "insiders constitute the greatest threat to computer systems." Since insiders have both access to and familiarity with the victim computer system authorized system users are in a better position to commit crimes. Insider scan be both general users and technical staff members. An organization's former employees, with their knowledge of an organization's operations, may also pose a threat, particularly if their access is not terminated promptly. In addition to the use of technology to commit fraud and theft, computer hardware and software may be vulnerable to theft.

1.3.3 Employee Sabotage

Employees are most familiar with their employer's computers and applications, including knowing what actions might cause the most damage, mischief, or sabotage. The downsizing of organizations in both the public and private sectors has created a group of individuals with organizational knowledge, who may retain potential system access. The number of incidents of employee sabotage is believed to be much smaller than the instances of theft, but the cost of such incidents can be quite high.

Common examples of computer-related employee sabotage include:

- Destroying hardware or facilities,
- Planting logic bombs that destroy
- Programs or data,
- Entering data incorrectly,
- "Crashing" systems,
- Deleting data,
- Holding data hostage, and

 ISBN: 978-93-93810-96-0

❖ Changing data.

1.3.4 Loss of Physical and Infrastructure Support

The loss of supporting infrastructure includes power failures (outages, spikes, and brownouts), loss of communications, water outages and leaks, sewer problems, lack of transportation services, fire, flood, civil unrest, and strikes.

1.3.5 Malicious Hackers

The term malicious hackers, sometimes called **crackers,** refer to those who break into computers without authorization. They can include both outsiders and insiders. Much of the rise of hacker activity is often attributed to increases in connectivity in both government and industry. One 1992study of a particular Internet site (i.e., one computer system) found that hackers attempted to break in at least once every other day.

The hacker threat should be considered in terms of past and potential future damage. Although current losses due to hacker attacks are significantly smaller than losses due to insider theft and sabotage, the hacker problem is widespread and serious. One example of malicious hacke activity is that directed against the public telephone system.

The hacker threat often receives more attention than more common and dangerous threats. The

U.S. Department of Justice's Computer Crime Unit suggests three reasons for this.

- The hacker threat is a more recently encountered threat. Organizations have always had to worry about the actions of their own employees and could use disciplinary measures to reduce that threat. However, these measures are ineffective against outsiders who are not subject to the rules and regulations of the employer.
- Organizations do not know the purposes of a hacker some hackers browse, some steal, some damage. These inabilities to identify purposes can suggestthat hacker attacks have no limitations.
- Hacker attacks make people feel vulnerable, particularly because their identity is unknown.

1.3.6 Industrial Espionage

Industrial espionage is the act of gathering proprietary data from private companies or the government for the purpose of aiding another companies. Industrial espionage can be perpetrated either by companies seeking to improve their competitive advantage or by governments seeking to aid their domestic industries. Foreign

industrial espionage carried out by a government is often referred to as economic espionage. Since information is processed and stored on computer systems, computer security can help protect against such threats; it can do little, however, to reduce the threat of authorized employees selling that information.

A 1992 study sponsored by the American Society for Industrial Security (ASIS) found that proprietary business information theft had increased 260 percent since 1985. The three most damaging types of stolen information were 1) pricing information, 2) manufacturing process information, and3) product development and specification information.

1.3.7 Malicious Code

Malicious code refers to viruses, worms, Trojan horses, logic bombs, and other "uninvited “software. Sometimes mistakenly associated only with personal computers, malicious code can attack other plat forms. A 1993 study of viruses found that while the number of known viruses is increasing exponentially, the number of virus incidents is not.39 The study concluded that viruses are becoming more prevalent, but only "gradually."

Actual costs attributed to the presence of malicious code have resulted primarily from system outages and staff time involved in repairing the systems. Nonetheless, these costs can be significant.

Malicious Software: A Few Key Terms:

- **Virus:** A code segment that replicates by attaching copies of itself to existing executables. The new copy of the virus is executed when a user executes the new host program. The virus may include an additional "payload" that triggers when specific conditions are met.
- **Trojan horse:** A program that performs a desired task, but that also includes unexpected (and undesirable) functions. This program could be modified to randomly delete one of the users' files each time they perform a useful function (editing), but the deletions are unexpected and definitely undesired.
- **Worm:** A self-replicating program that is self-contained and does not require a host program. The program creates a copy of itself and causes it to execute- Worms commonly use network services to propagate to other host systems.

1.3.8 Foreign Government Espionage

Threats posed by foreign government intelligence services may be present. In addition to possible economic espionage, foreign intelligence services may target unclassified systems to further their intelligence missions. Some unclassified information that may be of interest includes travel plans of senior officials, civil

 ISBN: 978-93-93810-96-0

defense and emergency preparedness, manufacturing technologies, satellite data, personnel and payroll data, and law enforcement, investigative, and security files.

1.3.9 Threats to Personal Privacy

The threat to personal privacy arises from many sources. In several cases federal and state employees have sold personal information to private investigators or other "**information brokers**."

One such case was uncovered in 1992 when the Justice Department announced the arrest of over two dozen individuals engaged in buying and selling information from Social Security Administration (SSA) computer files.42 During the investigation, auditors learned that SSA employees had unrestricted access to over 130 million employment records. Another investigation found that 5 percent of the employees in one region of the IRS had browsed through tax records of friends, relatives, and celebrities.43 Some of the employees used the information to create fraudulent tax refunds, but many were acting simply out of curiosity.

1.3.10 Threat Modeling

Structured approach to identifying, quantifying, and addressing threats and Essential part of development process are:

- Just like specing and designing
- Just like coding and testing

One technique presented here there are others (e.g., OCTAVE). The Threat Modeling Process is:

 ISBN: 978-93-93810-96-0

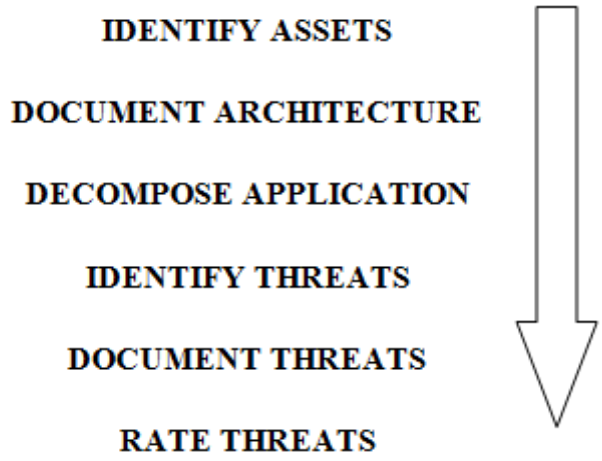


Fig 1.2 Threat modeling process

Identifying Assets:

■ What is it that you want to protect?

☐ Private data (e.g., customer list)

☐ Proprietary data (e.g., intellectual property)

☐ Potentially injurious data (e.g., credit card numbers, decryption keys)

■ These also count as "assets"

☐ Integrity of back-end databases

☐ Integrity of the Web pages (no defacement)

☐ Integrity of other machines on the network

☐ Availability of the application

Documenting Architecture

■ Define what the app does and how it's used

☐ Users view pages with catalog items

☐ Users perform searches for catalog items

☐ Users add items to shopping carts

☐ Users check out

■ Diagram the application

☐ Show subsystems

☐ Show data flow

 ISBN: 978-93-93810-96-0

☐ List assets

Decomposing the App

■ Refine the architecture diagram

☐ Show authentication mechanisms

☐ Show authorization mechanisms

☐ Show technologies (e.g., DPAPI)

☐ Diagram trust boundaries

☐ Identify entry points

■ Begin to think like an attacker

☐ Where are my vulnerabilities?

☐ What am I going to do about them?

Identifying Threats

■ Method #1: Threat lists

☐ Start with laundry list of possible threats

☐ Identify the threats that apply to your app

■ Method #2: STRIDE

☐ Categorized list of threat types

☐ Identify threats by type/category

■ Optionally draw threat trees

☐ Root nodes represent attacker's goals

☐ Trees help identify threat conditions

➢ **STRIDE**

 ISBN: 978-93-93810-96-0

Fig 1.3 STRIDE

Documenting Threats

- Document threats using a template

Table 1.1 Threat

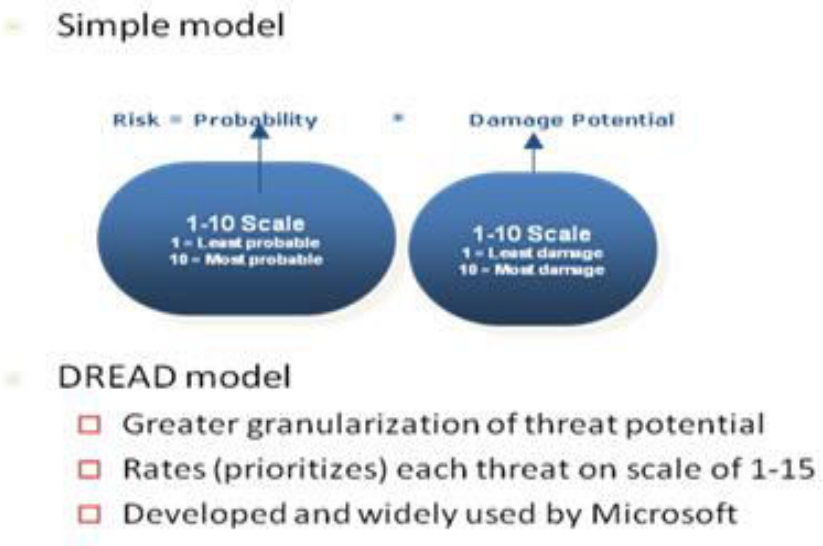
Theft of Auth Cookies by Eavesdropping on Connection	
Threat target	Connections between browsers and Web server
Risk	
Attack techniques	Attacker uses sniffer to monitor traffic
Countermeasures	Use SSL/TLS to encrypt traffic

 ISBN: 978-93-93810-96-0

Theft of Auth Cookies via Cross-Site Scripting	
Threat target	Vulnerable application code
Risk	
Attack techniques	Attacker sends e-mail with malicious link to users
Countermeasures	Validate input; HTML-encode output

Rating Threats

Simple model



➢ **DREAD**

Fig 1.4 DREAD

 ISBN: 978-93-93810-96-0

Example

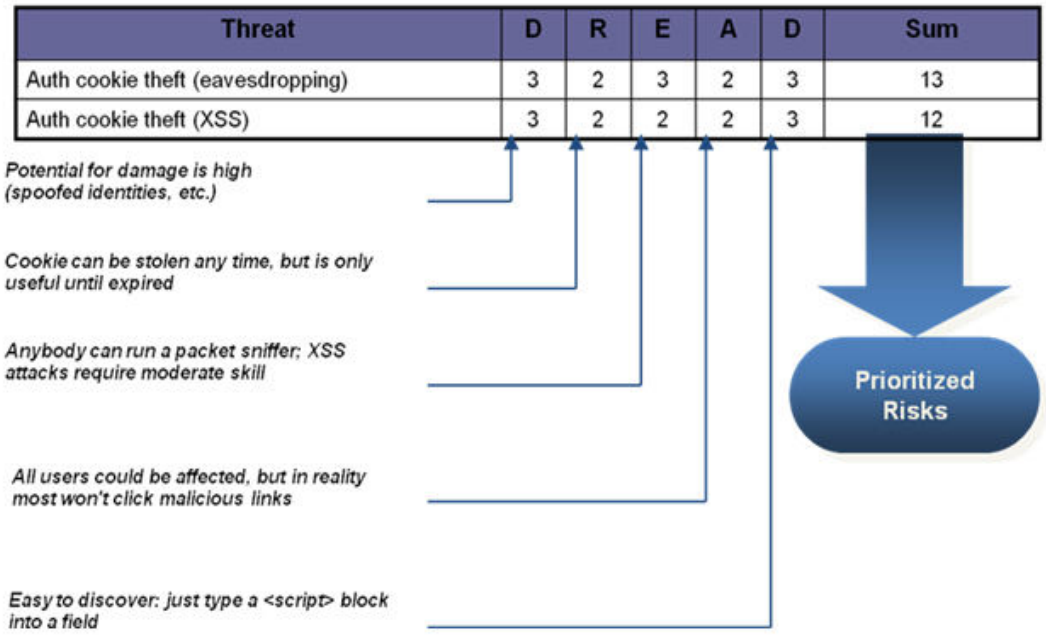


Fig 1.5 Thread Model

1.4 CRYPTOGRAPHY AND CRYPTOGRAPHIC PROTOCOLS

Cryptography is a branch of mathematics based on the transformation of data. It provides an important tool for protecting information and is used in many aspects of computer security. For example, cryptography can help provide data confidentiality, integrity, electronic signatures, and advanced user authentication.

1.4.1Basic Cryptographic Technologies

Cryptography relies upon two basic components: an algorithm and a key. A cryptographic algorithm is a well-defined transformation, which on a given input value produces an output value, achieving certain security objectives.

There are two basic types of cryptography: **Secret Key Systems** (also called symmetric systems) and **Public Key Systems** (also called asymmetric systems).

- **Secret Key Cryptography:** In secret key cryptography, two (or more) parties share the same key, and that key is used to encrypt and decrypt data. As the name implies, secret key cryptography relies on keeping the key secret. If The best known secret key system is the **Data Encryption Standard** (DES), published by NIST as Federal Information Processing Standard (FIPS).Secret key cryptography has been in use for centuries. Early forms merely transposed the written characters to hide the message.

- **Public Key Cryptography:** Public key cryptography is a modern invention and requires the use of advanced mathematics. Whereas secret key cryptography uses a single key shared by two (or more) parties, public key cryptography uses a pair of keys for each party. The public key can be made known to other parties; Public key cryptography is particularly useful when the parties wishing to communicate cannot rely upon each other or do not share a common key.
- **Hybrid Cryptographic Systems:** To maximize the advantages and minimize the disadvantages of both secret and public key cryptography, a computer system can use both types in a complementary manner, with each performing different functions. Secret key systems are often used for bulk data encryption and public key systems for automated key distribution.
- **Key Escrow:** Because cryptography can provide extremely strong encryption, it can thwart the government's efforts to lawfully perform electronic surveillance. NIST has published the **Escrowed Encryption Standard** as FIPS. Under the Federal Government's voluntary key escrow initiative, the decryption keys are split into parts and given to separate escrow authorities. Access to one part of the key does not help decrypt the data; both keys must be obtained.

1.5 ENCRYPTION

One of the best ways to obtain cost effective data confidentiality is through the use of encryption. Encryption transforms intelligible data, called **plaintext** into an unintelligible form, called **ciphertext.** This process is reversed through the process of decryption. Once data is encrypted, the ciphertext does not have to be protected against disclosure.

In cryptography, **encryption** is the process of encoding messages or information in such a way that only authorized parties can read it. Encryption does not of itself prevent interception, but denies the message content to the interceptor. In an encryption scheme, the message or information, referred to as plaintext, is encrypted using an encryption algorithm, generating ciphertext that can only be read if decrypted. For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm. It is in principle possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, large computational resources and skill are required. An authorized recipient can easily decrypt the message with the key provided by the originator to recipients, but not to unauthorized interceptors.

 ISBN: 978-93-93810-96-0

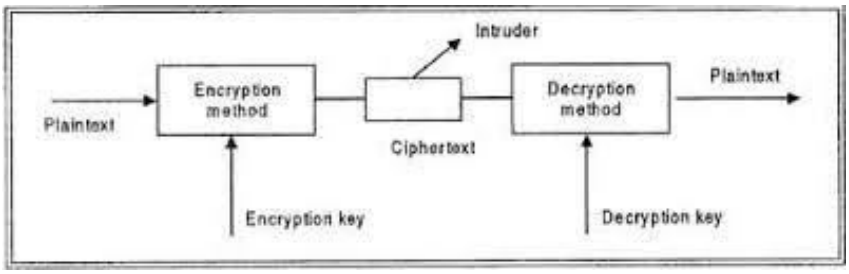


Fig 1.6 Data Encryption

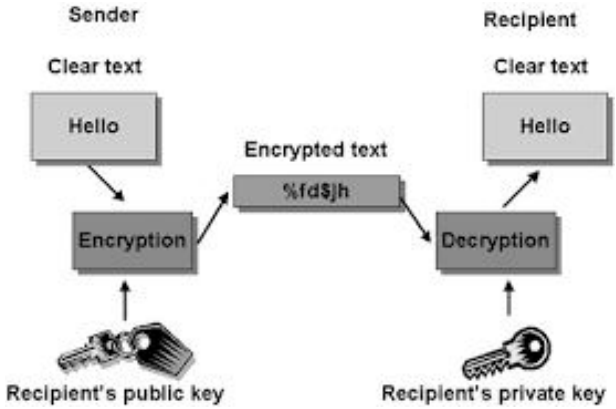


Fig 1.7 Process of Encryption

1.6 AUTHENTICATION

Cryptography is the basis for several advanced authentication methods. Instead of communicating passwords over an open network, authentication can be performed by demonstrating knowledge of a cryptographic key. (like a human with a fingerprint)

Fig 1.8 Example of Authentication

The term Authentication is derived from the Greek word of authentikos- meaning of this Greek term is 'Genuine'. Authentication is the process of actually confirming that identity. In other words, authentication often involves verifying the validity of at least one form of identification.(Ex: Password type, Eyre's,etc)

1.7 MESSAGE AUTHENTICATION CODES (MAC)

In cryptography, a **Message Authentication Code** (**MAC**) is a short piece of information used to authenticate a message and to provide integrity and authenticity

assurances on the message. Integrity assurances detect accidental and intentional message changes, while authenticity assurances affirm the message's origin.

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- will consider the security requirements then three alternative functions used:
  - hash function
  - message encryption
  - message authentication code (MAC)
- a small fixed-sized block of data
- generated from message + secret key
- MAC = C(K,M)
- appended to message when sent

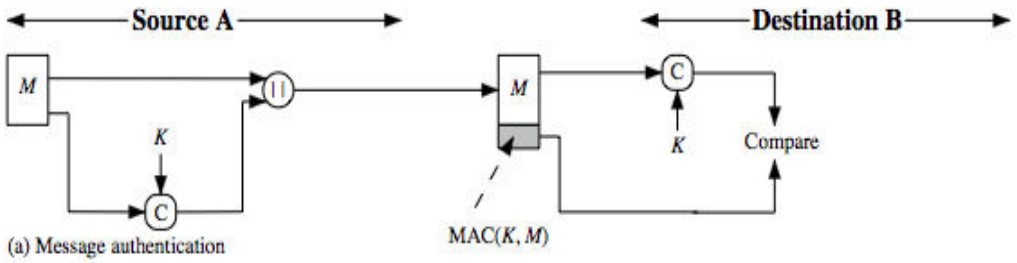


Fig 1.9 Processing of MAC

- The MAC provides authentication can also use encryption for secrecy
  - generally use separate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before
- Why use a MAC?
  - sometimes only authentication is needed
  - sometimes need authentication to persist longer than the encryption (eg. archival use)
- MAC is not a digital signature

MAC Properties

- MAC is a cryptographic checksum

$MAC = C_K(M)$

- condenses a variable-length message M
- using a secret key K
- to a fixed-sized authenticator
- It is a many-to-one function
- potentially many messages have same MAC
- but finding these needs to be very difficult

Requirements for MACs

- Taking into account the types of attacks need the MAC to satisfy the following:

1. knowing a message and MAC, is infeasible to find another message with same MAC
2. MACs should be uniformly distributed
3. MAC should depend equally on all bits of the message

Security of MACs

It handles the following attacks

- **brute-force** attacks exploiting
- strong collision resistance hash have cost $2^{m/2}$
- 128-bit hash looks vulnerable, 160-bits better
- MACs with known message-MAC pairs
- can either attack keyspace (cf key search) or MAC
- at least 128-bit MAC is needed for security
- **cryptanalytic attacks** exploit structure
- like block ciphers want brute-force attacks to be the best alternative
- more variety of MACs so harder to generalize about cryptanalysis

1.7.1 Keyed Hash Functions as MACs (HMAC)

In cryptography, a **keyed-hash message authentication code** (**HMAC**) is a specific construction for calculating a Message Authentication Code (MAC) involving a cryptographic hash function in combination with a secret cryptographic key. As with any MAC, it may be used to simultaneously verify both the data integrity and

the authentication of a message. Any cryptographic hash function, such as MD5 or SHA-1, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-MD5 or HMAC-SHA1 accordingly. The cryptographic strength of the HMAC depends upon the cryptographic strength of the underlying hash function, the size of its hash output, and on the size and quality of the key.

An iterative hash function breaks up a message into blocks of a fixed size and iterates over them with a compression function. For example, MD5 and SHA-1 operate on 512-bit blocks. The size of the output of HMAC is the same as that of the underlying hash function (128 or 160 bits in the case of MD5 or SHA-1, respectively), although it can be truncated if desired.

The definition and analysis of the HMAC construction was first published in 1996 by Mihir Bellare, Ran Canetti, and Hugo Krawczyk, who also wrote RFC 2104. This paper also defined a variant called NMAC that is rarely, if ever, used. FIPS PUB 198 generalizes and standardizes the use of HMACs. HMAC-SHA1 and HMAC-MD5 are used within the IPSec and TLS protocols.

Definition:

KeyedHash = Hash (Key|Message)

Objectives:

- use, without modifications, hash functions
- allow for easy replaceability of embedded hash function
- preserve original performance of hash function without significant degradation
- use and handle keys in a simple way.
- have well understood cryptographic analysis of authentication mechanism strength

hash function on the message:

$HMAC_K(M)$= Hash [(K^+ XOR opad) ||

Hash [(K^+ XOR ipad) || M)]]

- where K^+ is the key padded out to size
- opad, ipad are specified padding constants
- overhead is just 3 more hash calculations than the message needs alone
- any hash function can be used
- eg. MD5, SHA-1, RIPEMD-160, Whirlpool

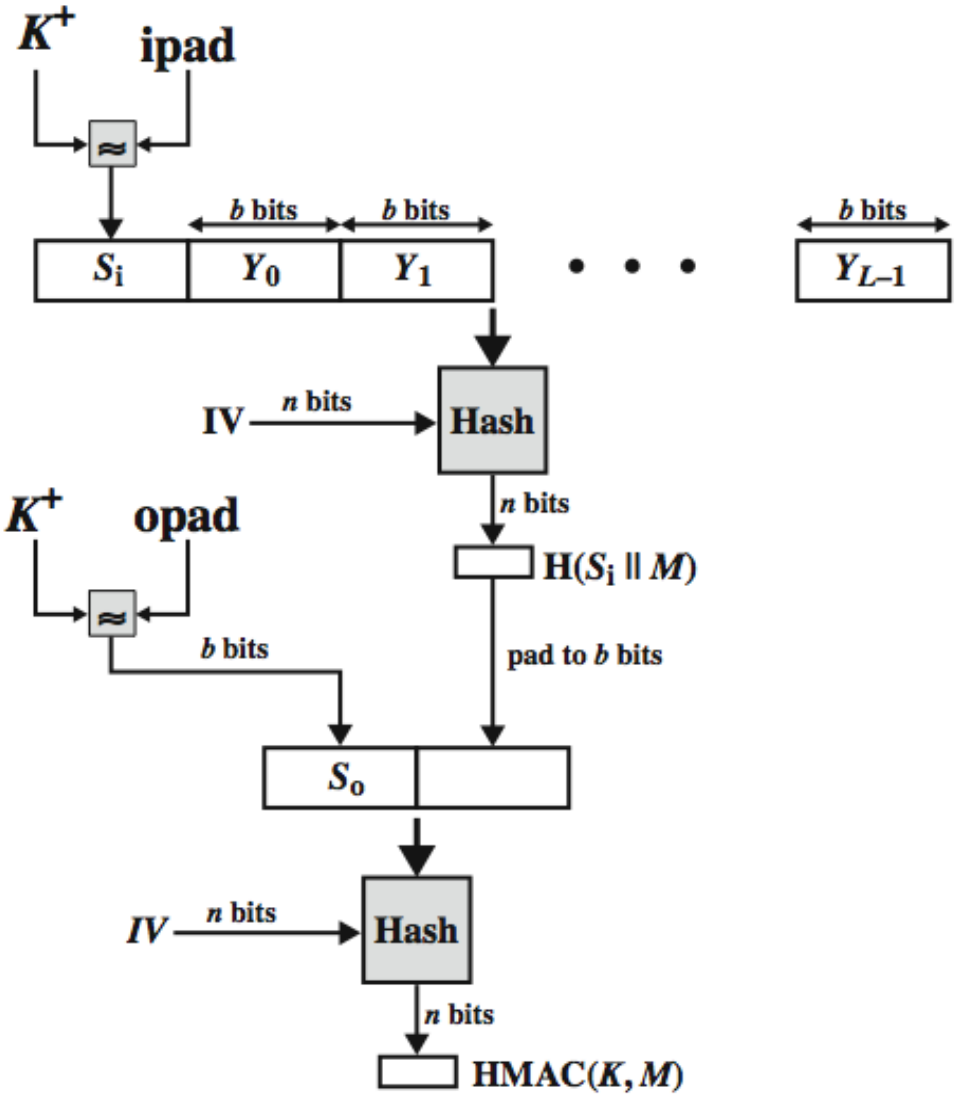


Fig 1.10 HMAC – Process Flow

HMAC Security:

- proved security of HMAC relates to that of the underlying hash algorithm
- attacking HMAC requires either:
  - brute force attack on key used
  - birthday attack (but since keyed would need to observe a very large number of messages)
- choose hash function used based on speed verses security constraints

1.7.2 Using Symmetric Ciphers for MACs

- Using Symmetric Ciphers can use any block cipher chaining mode and use final block as a MAC

 ISBN: 978-93-93810-96-0

- Data Authentication Algorithm (DAA) is a widely used MAC based on DES-CBC
- using IV=0 and zero-pad of final block
- encrypt message using DES in CBC mode
- and send just the final block as the MAC
- or the leftmost M bits (16≤M≤64) of final block
- but final MAC is now too small for security

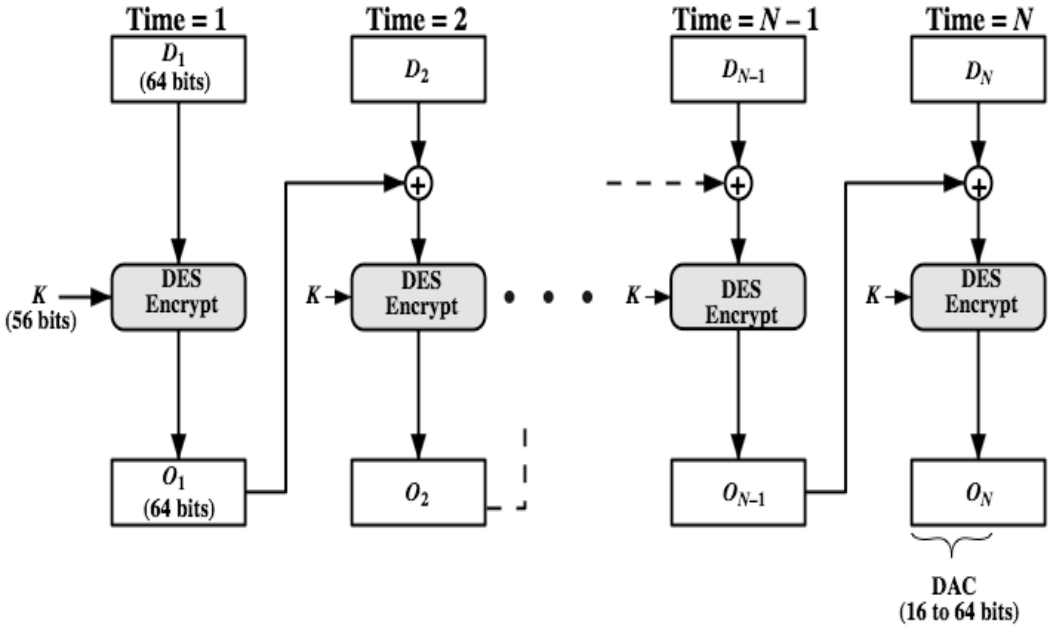


Fig 1.11 DAA Algorithm

Previously saw the DAA (CBC-MAC).It is widely used in govt & industry. But has message size limitation. It can overcome using 2 keys & padding. Thus forming the Cipher-based Message Authentication Code (CMAC).adopted by NIST SP800-38B

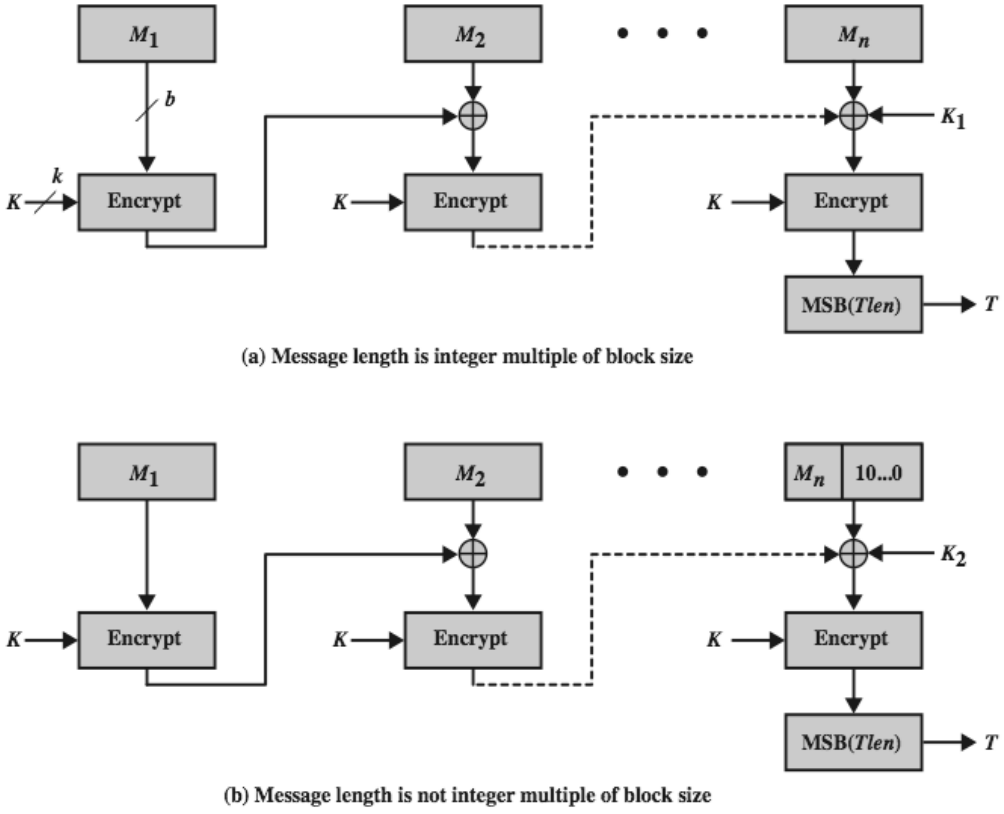


Fig 1.12 CMAC Overview

Stallings Figure 1.12 shows the structure of CMAC. It uses the block size of the underlying cipher (ie 128-bits for AES or 64-bits for triple-DES). The message is divided into n blocks M1..Mn, padded if necessary. The algorithm makes use of a k-bit encryption key K and an n-bit constant K1 or K2 (depending on whether the message was padded or not). For AES, the key size k is 128,192, or 256 bits; for triple DES, the key size is 112 or 168 bits. The two constants K1 & K2 are derived from the original key K using encryption of 0 and multiplication in GF (2^n), as detailed in the text.

1.7.3 Authenticated Encryption

- It is focused for simultaneously protect confidentiality and authenticity of communications and often required but usually separate.
- approaches

- Hash-then-encrypt: E(K, (M || H(M))
- MAC-then-encrypt: E(K2, (M || MAC(K1, M))
- Encrypt-then-MAC: (C=E(K2, M), T=MAC(K1, C)
- Encrypt-and-MAC: (C=E(K2, M), T=MAC(K1, M)

 ISBN: 978-93-93810-96-0

- decryption /verification straightforward
- but security vulnerabilities with all these

1.7.4 Counter with Cipher Block Chaining-Message Authentication Code (CCM)

The CCM consists of

- ✓ NIST standard SP 800-38C for WiFi
- ✓ variation of encrypt-and-MAC approach
- ✓ algorithmic ingredients
- ✓ AES encryption algorithm
- ✓ CTR mode of operation
- ✓ CMAC authentication algorithm
- ✓ single key used for both encryption & MAC

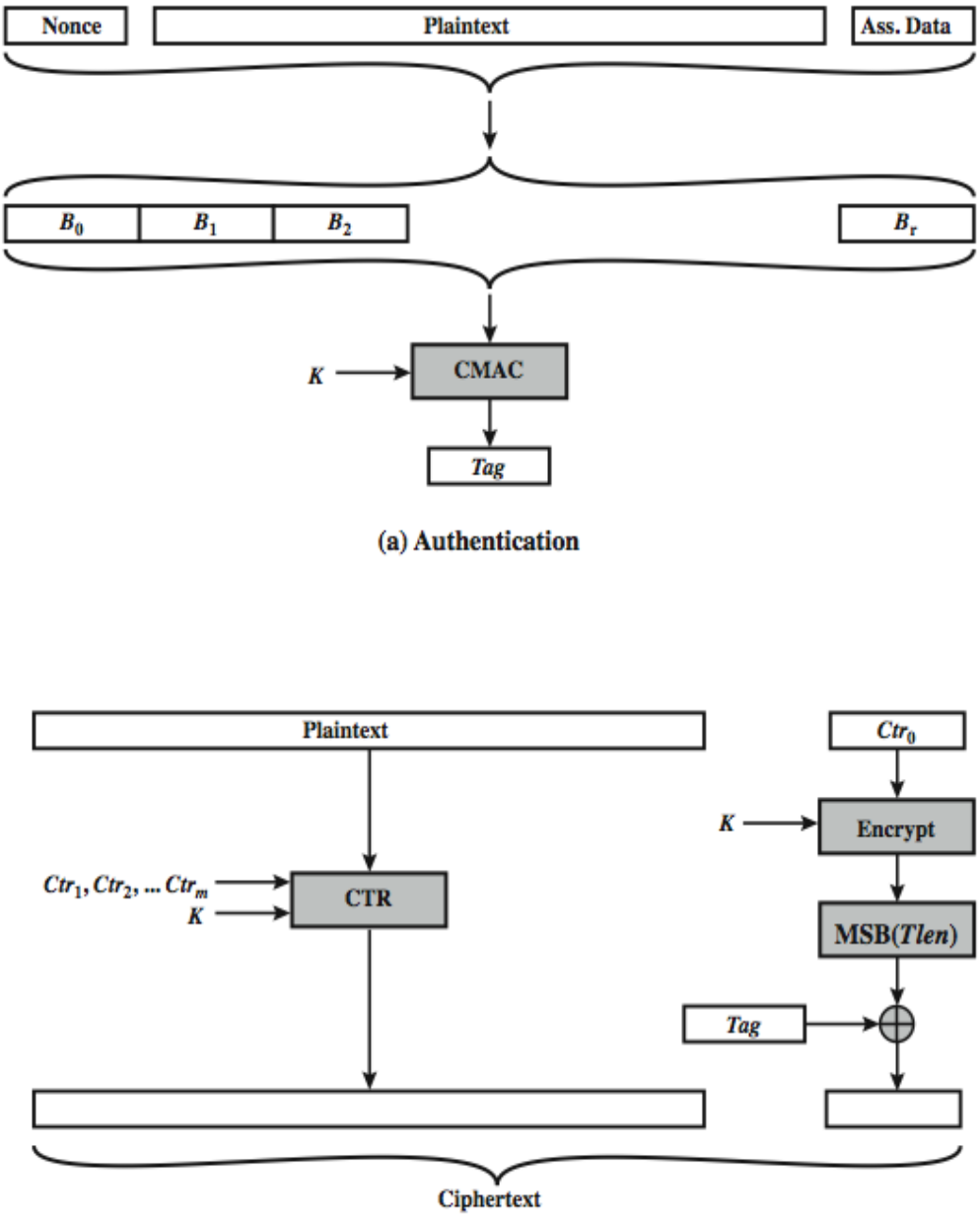


Fig 1.13 the CCM Operation

Stallings Figure 1.13 illustrates the operation of CCM. For authentication, the input includes the nonce, the associated data, and the plaintext. This input is formatted as a sequence of blocks B0 through Br. The first block contains the nonce plus some formatting bits that indicate the lengths of the N, A, and P elements. This is followed by zero or more blocks that contain A, followed by zero of more blocks that contain P. The resulting sequence of blocks serves as input to the CMAC algorithm, which produces a MAC value with length Tlen, which is less than or equal to the block

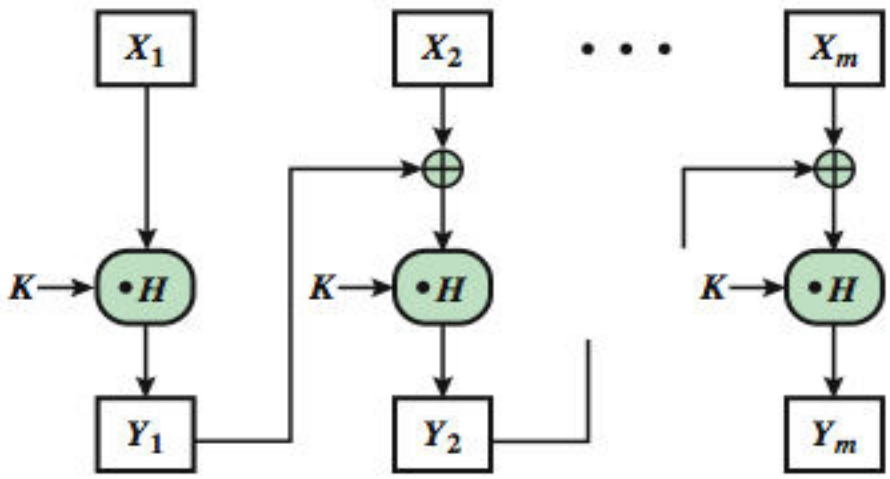
length (Figure 1.13a). For encryption, a sequence of counters is generated that must be independent of the nonce. The authentication tag is encrypted in CTR mode using the single counter Ctr0. The Tlen most significant bits of the output are XORed with the tag to produce an encrypted tag. The remaining counters are used for the CTR mode encryption of the plaintext . The encrypted plaintext is concatenated with the encrypted tag to form the ciphertext output (Figure 12.9b). CCM is a relatively complex algorithm. Note that it requires two complete passes through the plaintext, once to generate the MAC value, and once for encryption. Further, the details of the specification require a tradeoff between the length of the nonce and the length of the tag, which is an unnecessary restriction. Also note that the encryption key is used twice with the CTR encryption mode, once to generate the tag and once to encrypt the plaintext plus tag. Whether these complexities add to the security of the algorithm is not clear. In any case, two analyses of the algorithm conclude that CCM provides a high level of security.

1.7.5 Galois/Counter Mode (GCM)

The NIST standard SP 800-38D - parallelizable. Here message is encrypted in variant of CTR. ciphertext multiplied with key & length over in (2^{128}) to generate authenticator tag. It have GMAC MAC-only mode also

Two functions:

- ✓ GHASH - a keyed hash function
- ✓ GCTR - CTR mode with incremented counter



(a) $\mathbf{GHASH}_H(X_1 \| X_2 \| \ldots \| X_m) = Y_m$

Fig 1.14 GCM Function

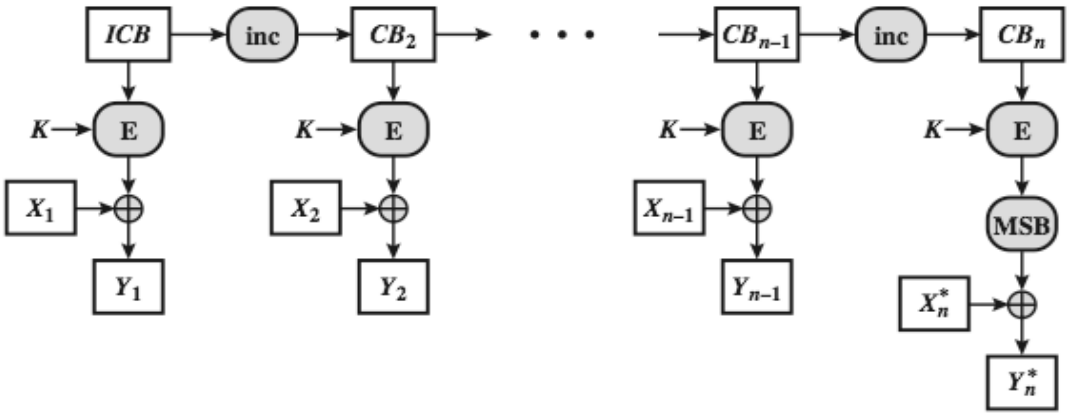
 ISBN: 978-93-93810-96-0

$GHASH_H(X)$ takes a input the hash key H and a bit string X such that len(X) = 128m bits for some positive integer m and produces a 128-bit MAC value. The function may be specified as shown in Stallings Figure 1.14a. The GHASH H(X) function can be expressed as:

$(X_1 \bullet H^m)$ XOR $(X_2 \bullet H^{m-1})$ XOR ... XOR $(X_{m-1} \bullet H^2)$ XOR $(X_m \bullet H)$

Where • designates multiplication in GF (2^{128})

This formulation has desirable performance implications. If the same hash key is to be used to authenticate multiple messages, then the values H^2, H^3 can be precalculated one time for use with each message to be authenticated. Then, the blocks of the data to be authenticated $(X_1, X_2, \ldots X_m)$ can be processed in parallel, because the computations are independent of one another.



(b) $GCTR_K(ICB, X_1 \| X_2 \| \ldots \| X_n^*) = Y_n^*$

Fig 1.15 GCM Function

$GCTR_K(ICB, X)$ takes a input a secret key K and a bit string X arbitrary length and returns a ciphertext Y of bit length len(X). The function may be specified as as shown in Stallings Figure 12.10b. Here the $inc_{32}(S)$ function increments the rightmost 32 bits of S by 1 mod 2^{32}, and the remaining bits are unchanged. Note that the counter values can be quickly generated and that the encryption operations can be performed in parallel.

 ISBN: 978-93-93810-96-0

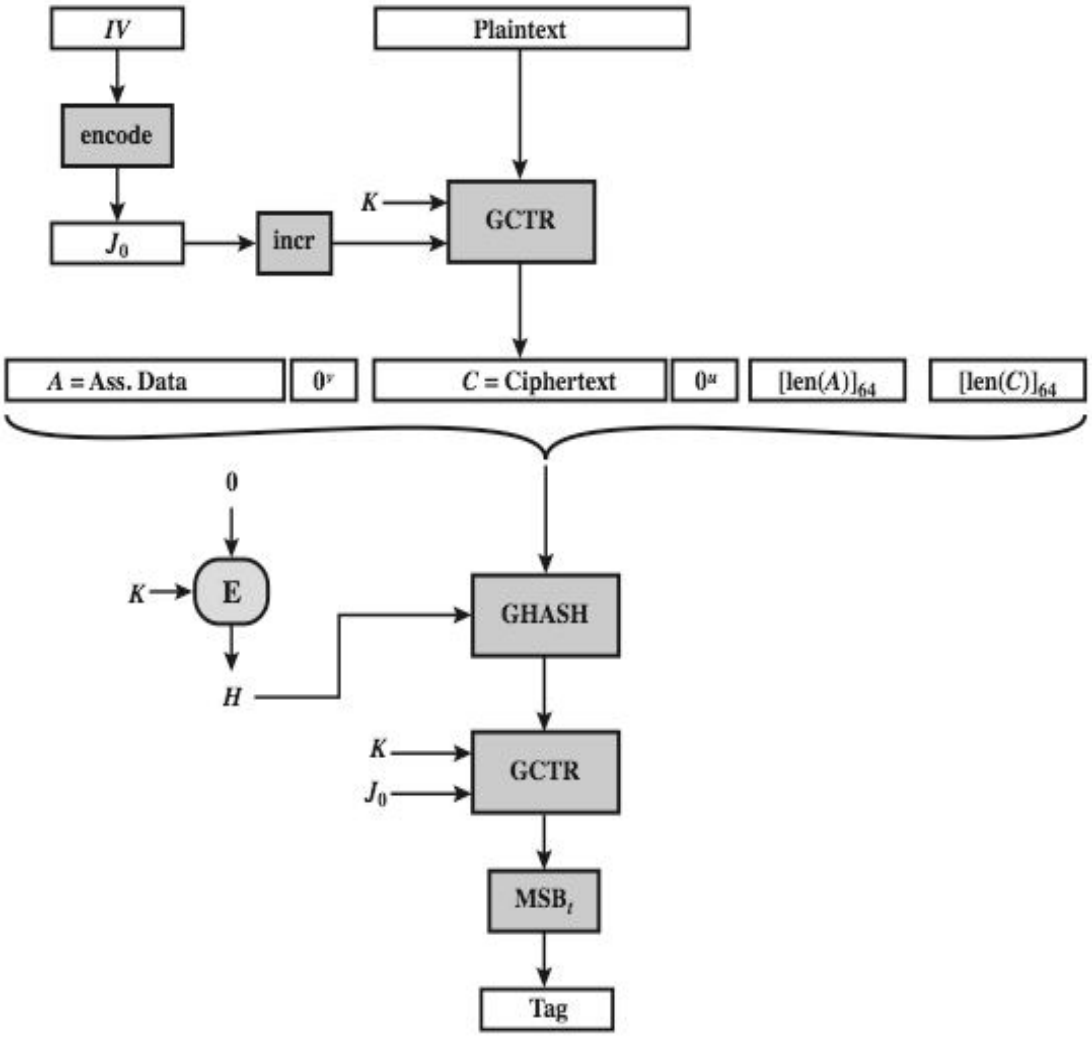


Fig 1.16 GCM Overview

We can now define the overall authenticated encryption function as shown in Stallings Figure 1.16 The input consists of a secret key K, an initialization vector IV, a plaintext P, and additional authenticated data A that will be authenticated but not encrypted. In step 1, the hash key is generated by encrypting a block of all zeros with the secret key K. In step 2, the pre-counter block (J_0) is generated from the IV. In particular, when the length of the IV is 96 bits, then the padding string 0 ‖ 1 is appended to the IV to form the pre-counter block. Otherwise, the IV is padded with the minimum number of 0 bits, possibly none, so that the length of the resulting string is a multiple of 128 bits (the block size); this string in turn is appended with 64 additional '0' bits, followed by the 64-bit representation of the length of the IV, and the GHASH function is applied to the resulting string to form the pre-counter block.

Thus, GCM is based on the CTR mode of operation, and adds a MAC that authenticates both the message and additional data that requires only authentication. The function that computes the hash uses only multiplication in a Galois field. This choice was made because the operation of multiplication is easy to perform within a Galois field and is easily implemented in hardware. CTR- based authenticated encryption is the most efficient mode of operation for high-speed packet networks. The GCM mode meets a high level of security requirements.

1.7.6 Pseudorandom Number Generation (PRNG) Using Hash Functions and MACs

It is essential elements of PRNG are seed value and deterministic algorithm. The seed must be known only as needed. It can base PRNG on

- encryption algorithm (Chs 7 & 10)
- hash function (ISO18031 & NIST SP 800-90)
- MAC (NIST SP 800-90)

PRNG using a Hash Function:

- It hash PRNG from SP800-90 and ISO18031
- take seed V
- repeatedly add 1
- hash V
- use n-bits of hash as random value
- secure if good hash used

 ISBN: 978-93-93810-96-0

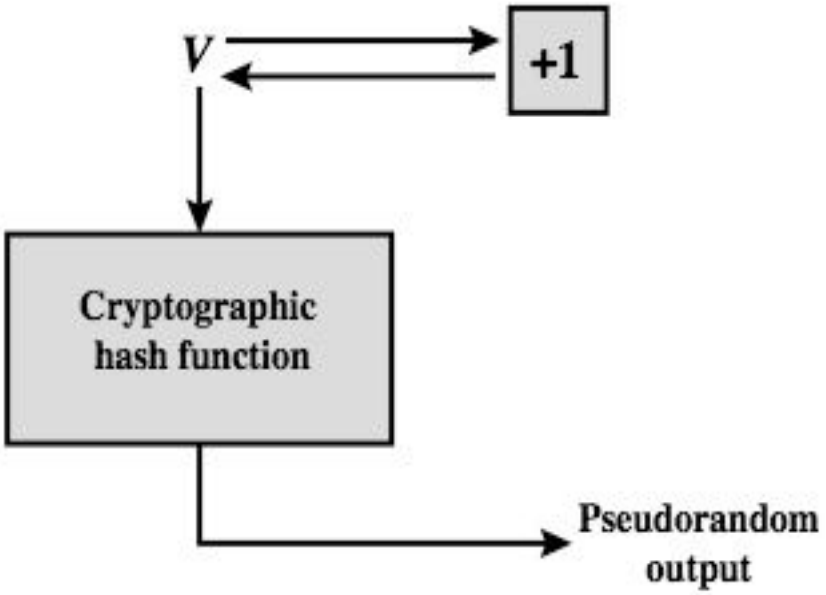


Fig1.17 PRNG Using hash function

Stallings Figure 1.14a shows the basic strategy for a hash-based PRNG specified in SP 800-90 and ISO 18031. The algorithm takes as input: V = seed; seedlen = bit length of V >= k + 64, where k is a desired security level expressed in bits; n = desired number of output bits. The algorithm uses the cryptographic hash function H with an hash value output of outlen bits.

The basic operation of the algorithm is:

m = ceil (n/outlen)

Data = V

W = the null string

For i = 1 to m

w_i = H(data)

W = W || w_i

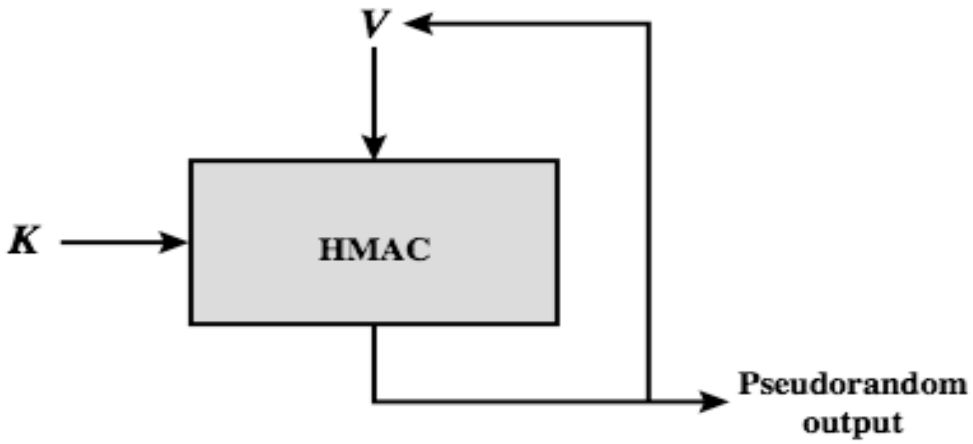
Data = (data + 1) mod $2^{seedlen}$

Return leftmost n bits of W

Thus, the pseudorandom bit stream is w 1 || w2 || … || wm, with the final block truncated if required. The SP 800-90 specification also provides for periodically updating V to enhance security. The specification also indicates that there are no known or suspected weaknesses in the hash- based approach for a strong cryptographic hash algorithm, such as SHA-2.

PRNG using a MAC :

- MAC PRNGs in SP800-90, IEEE 802.11i, TLS
- use key
- input based on last hash in various ways



(b) PRNG using HMAC

Fig 1.18 Prng Using Hmac

Hash function is one of the methods that can be used to generate a message authentication code.

This is done with the help of a generating a hash value of secret key & and a message to be authenticated. The message to be authenticated can be of any length. But the output of the hash function done on the message, will be of fixed length. This generated MAC, will also confirm the authenticity, because only the person that has the secret key can open that MAC value for verifying the data(as mentioned before the MAC is generated with the secret key value).As mentioned, the sender and the receiver will negotiate a secret key, before beginning the communication. This is similar to symmetric encryption, where same keys are used to decrypt at both the ends.

 ISBN: 978-93-93810-96-0

1.8 HASH FUNCTIONS

A hash function is any function that can be used to map digital data of arbitrary size to digital data of fixed size. The values returned by a hash function are called **hash values**, **hash codes**, **hash sums**, or simply **hashes**.

- ❑ Crypto hash function h(x) must provide
  - o **Compression** — output length is small
  - o **Efficiency** — h(x) easy to compute for any x
  - o **One-way** — given a value y it is infeasible to find an x such that h(x) = y
  - o **Weak collision resistance** — given x and h(x), infeasible to find $y \neq x$ such that h(y) = h(x)
  - o **Strong collision resistance** — infeasible to find any x and y, with $x \neq y$ such that h(x) = h(y)
- ❑ Many collisions exist, but cannot find any

1.8.1 Popular Crypto Hashes

- ❑ **MD5 — invented by Rivest**
  - o **128 bit output**
  - o **Note: MD5 collision recently found**
- ❑ **SHA-1 — A US government standard (similar to MD5)**
  - o **160 bit output**
- ❑ **Many others hashes, but MD5 and SHA-1 most widely used**
- ❑ **Messages are hashed in blocks**

1.8.2 Properties

- Determinism
- Uniformity
- Defined range
- Variable range
- Dynamic hash function
- Data normalization
- Continuity
- Non-invertible

1.8.3 The hash function SHA1

The hash function known as SHA1 is a simple but strange function from strings of almost arbitrary length to strings of 160 bits. The function was finalized in 1995, when a FIPS (Federal Information Processing Standard) came out from the US National Institute of Standards that specified SHA1.Let $\{0, 1\}<\ell$ denote the set of all strings of length strictly less than ℓ. The function SHA1:

$\{0, 1\}<264 \rightarrow \{0, 1\}160$

It begins by padding the message via the function shapad, and then iterates the compression function sha1 to get its output.

SHA1 is derived from a function called MD4 that was proposed by Ron Rivest in 1990, and the

Key ideas behind SHA1 are already in MD4. Besides SHA1, another well-known "child" of MD4 is MD5, which was likewise proposed by Rivest. The MD4, MD5, and SHA11 algorithms are all quite similar in structure. The first two produce a 128-bit output, and work by "chaining" a compression function that goes from 512+128 bits to 128 bits, while SHA1 produces a 160 bit output and works by chaining a compression function from 512 + 160 bits to 160 bits. First and foremost, it is supposed to be the case that nobody can find distinct strings M and M′ such that SHA1 (M) = SHA1 (M′). This property is called **collision resistance**.

$X \wedge Y$ bitwise AND of X and Y

$X \vee Y$ bitwise OR of X and Y

$X \oplus Y$ bitwise XOR of X and Y

$\neg X$ bitwise complement of X

$X + Y$ integer sum modulo 232 of X and Y

ROTLl(X) circular left shift of bits of X by l positions ($0 \leq l \leq 31$)

Fig 1.19: Operations on 32-bit words used in sha1.

1.9 ONE-WAY FUNCTIONS

A **one-way function** is a function that is easy to compute on every input, but hard to invert given the image of a random input. Here, "easy" and "hard" are to be understood in the sense of computational complexity theory, specifically the theory of polynomial time problems. Not being one-to-one is not considered sufficient of a function for it to be called one-way.

 ISBN: 978-93-93810-96-0

1.10 PUBLIC – KEY CRYPTOGRAPHY

Public-key cryptography, also known as **asymmetric cryptography**, is a class of cryptographic protocols based on algorithms that require two separate keys, one of which is secret and one of which is public. Although different, the two parts of this key pair are mathematically linked. The public key is used, for example, to encrypt plaintext or to verify a digital signature.

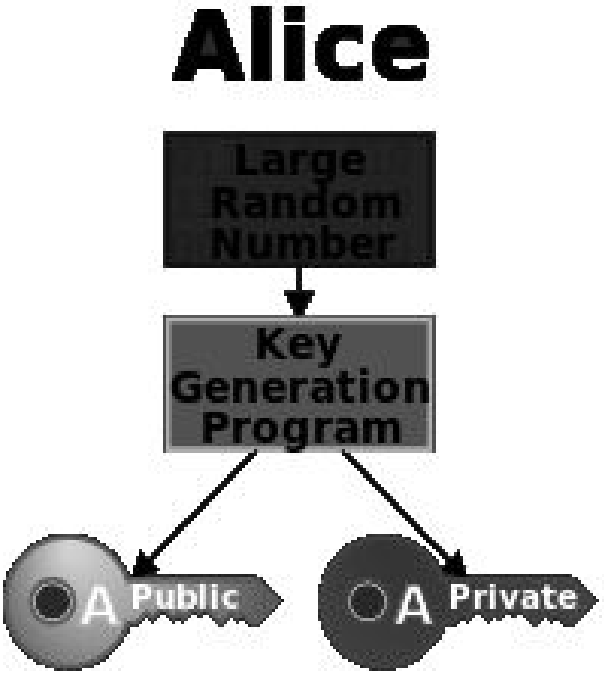


Fig 1.20 An unpredictable number is used to begin generation of an acceptable pair of keys suitable for use by an asymmetric key algorithm.

Public-key algorithms are based on mathematical problems that currently admit no efficient solution and are inherent in certain integer factorization, discrete logarithm, and elliptic curve relationships. Thus the public key may be published without compromising security. Security depends only on keeping the private key private. Public key algorithms, unlike symmetric key algorithms, do not require a secure for the initial exchange of one secret keys between the parties.

 ISBN: 978-93-93810-96-0

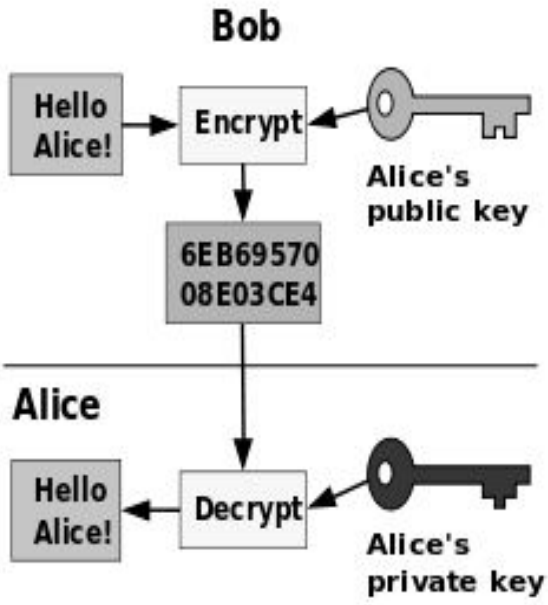


Fig 1.21 In an asymmetric key encryption scheme, anyone can encrypt messages using the public key, but only the holder of the paired private key can decrypt. Security depends on the secrecy of the private key.

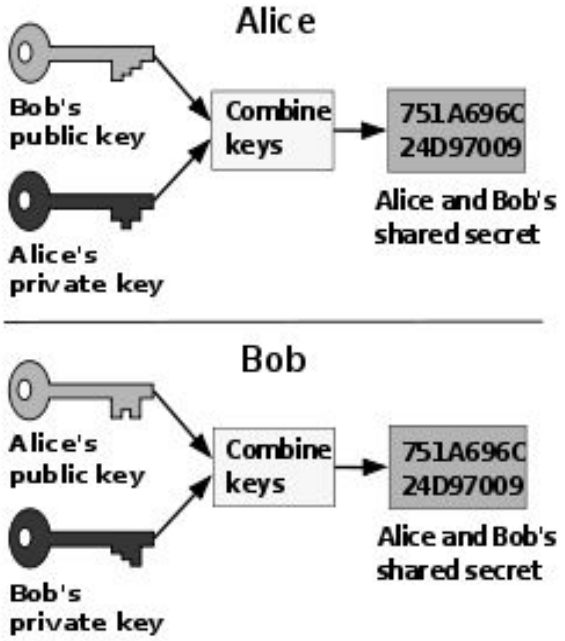


Fig 1.22 The Diffie–Hellman key exchange scheme

Because of the computational complexity of asymmetrical encryption, it is typically used only to transfer a symmetrical encryption key by which the message is encrypted. The symmetrical encryption/decryption is based on simpler algorithms and is much faster.

Message authentication involves hashing the message to produce a "digest," and encrypting the digest with the private key to produce a digital signature. Thereafter anyone can verify this signature by

(1) computing the hash of the message

(2) decrypting the signature with the signer's public key, and

(3) comparing the computed digest with the decrypted digest.

Equality between the digests confirms the message is unmodified since it was signed, and that the signer, and no one else, intentionally performed the signature operation — presuming the signer's private key has remained secret to the signer.

Public-key algorithms are fundamental security ingredients in cryptosystems, applications and protocols. They underpin various Internet standards, such as Transport Layer Security (TLS), S/MIME, PGP, and GPG. Public-key cryptography is used as a method of assuring the confidentiality, authenticity and non-reputability of electronic communications and data storage.

1.11 SECURE CHANNELS

In cryptography, a secure channel is a way of transferring data that is resistant to overhearing and tampering. A confidential channel is a way of transferring data that is resistant to overhearing but not necessarily resistant to tampering. An authentic channel is a way of transferring data that is resistant to tampering but not necessarily resistant to overhearing.

- Authentication
- Message Integrity: msg is protected against modification

– More than authentication of communication parties. e.g. protection of the integrity of on-line transaction agreement

- Confidentiality: Msg won’t be intercepted and read by evaesdroppers

– Cryptographic keys are not enough

Secure Channel: Authentication

- Alice initiates in setting up a channel between Alice and Bob. Once it is done, Alice and Bob know for sure whom they are talking to.
- Authentication based on shared secrete keys (Session Keys): Challenge-Response Protocol

 ISBN: 978-93-93810-96-0

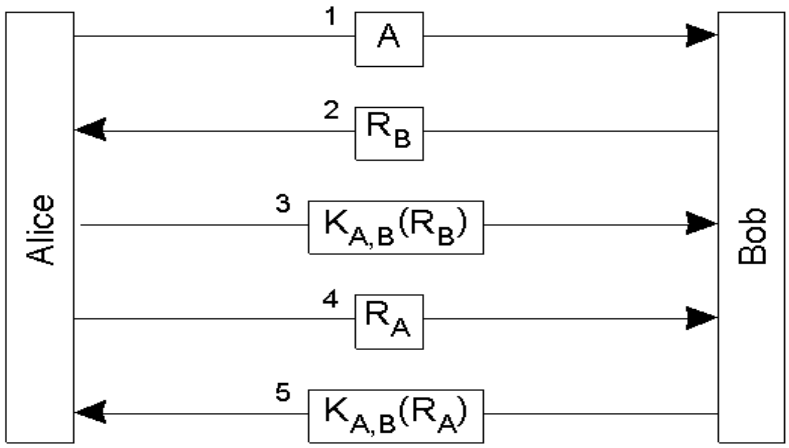


Fig 1.23 Security authentication

1: identify of A 2: Challenge of B 3: Encrypted challenge 4: Challenge of A

5: Encrypted challenge

Optimized Authentication?

- Authentication based on a shared secret key, but using three instead of five keys

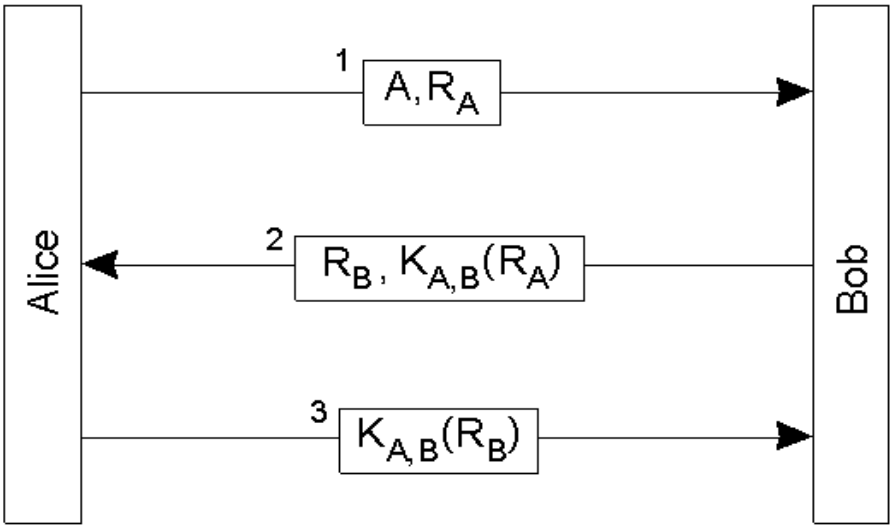


Fig 1.24 Security

 ISBN: 978-93-93810-96-0

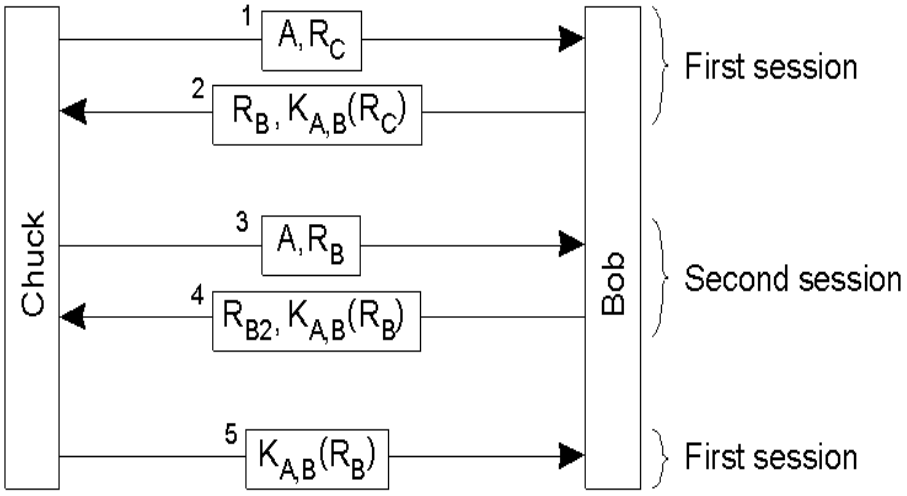


Fig 1.25 Reflection Attack

- Two comm parties use the same challenge in different runs of the protocol
- Also, valuable info. Ka,b(Rc) is released to unknown person

Key Distributed Center

- Shared-key based authentication is not scalable. In a system with n hosts, n(n-1)/2 keys are needed and each host needs to manage n-1 keys
- Alternative is to assume a trusted third party,like KDC, which shares a secret key with each host

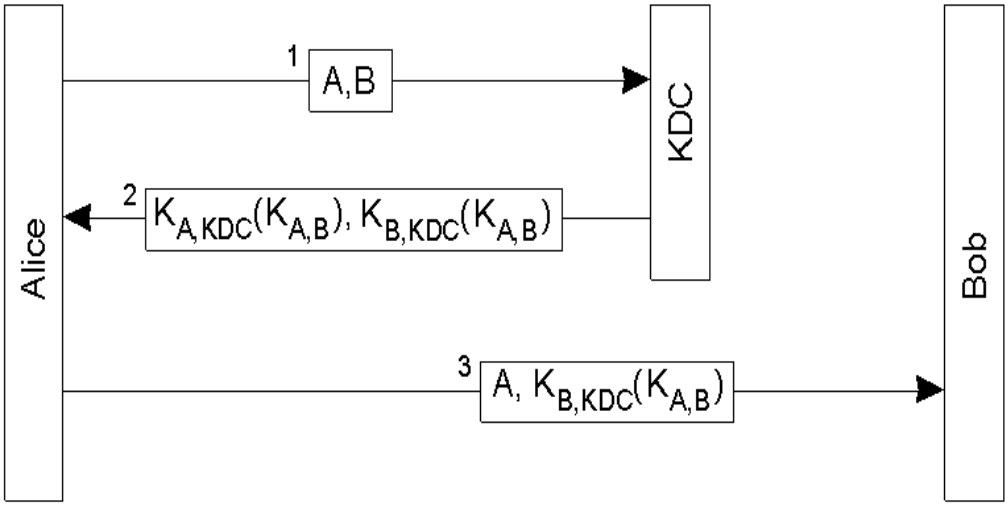


Fig 1.26 Key Distributed Systems

- The message $K_{B,KDC}(K_{A,B})$ is called a ticket
- Alice uses this ticket to establish connection with Bob

 ISBN: 978-93-93810-96-0

Needham-Schroeder Protocol

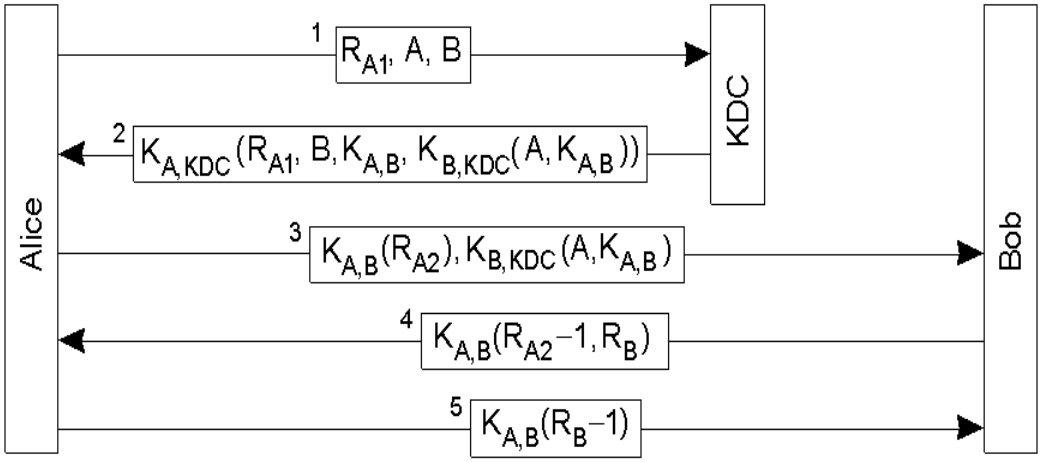


Fig 1.27 Schroder Protocol

- R_{A1} is a nonce (random number, "number use for once") to uniquely related msg 1 and msg 2 to each other.
- The identify B of Bob is included in msg 2 to confirm the return ticket between A and B.
- Returning R_{A2}-1 in msg 4 proves Bob knows the shared key and he actually has used the key to decrypted the challenge.

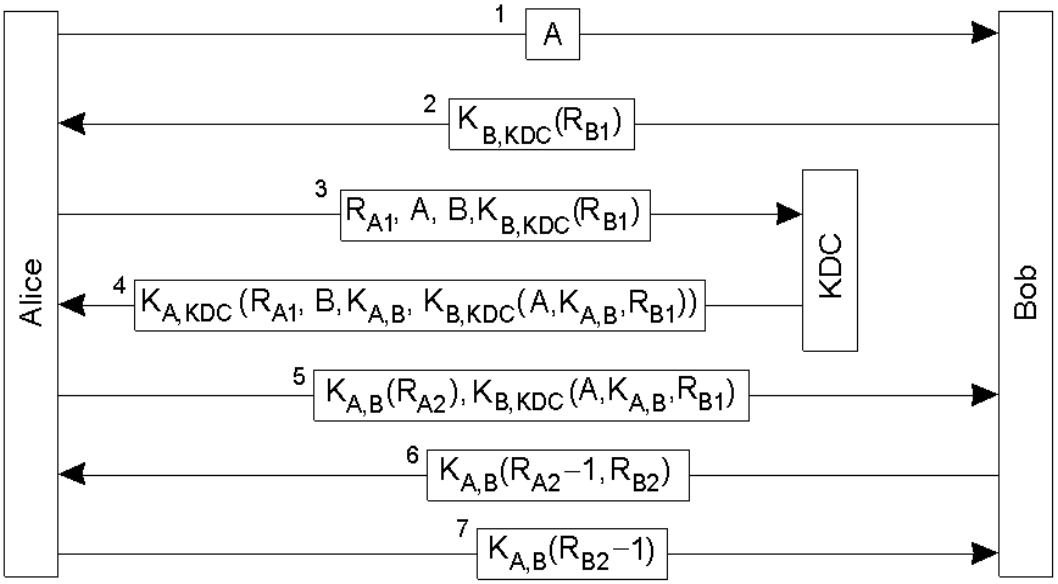


Fig 1.28 Improved Schroder Protocol

- Using an extra nonce R_{B1} to protect against malicious reuse of a previously generated session key

Shared Key Setup by Public-Key

- Mutual authentication, assuming knowledge of public keys of each other

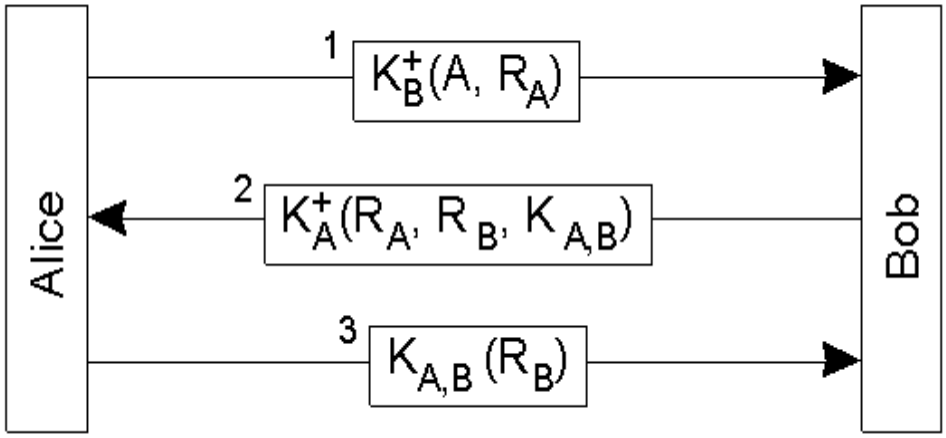


Fig 1.29 Shared Public Key

Initial Key Establishment

- Diffie-Hellman Key Exchange

– Alice and Bob agree on two large public numbers n and g

– Alice and Bob pick up two large random numbers, x and y, as their private keys

– Alice send g^x mod n to Bob and Bob sends g^y mod n to Alice, along with n ang g

- g^x mod n is one-way function: x is impossible to be computed

– Established shared key: $(g^x \text{ mod } n)^y = g^{xy} \text{ mod } n$

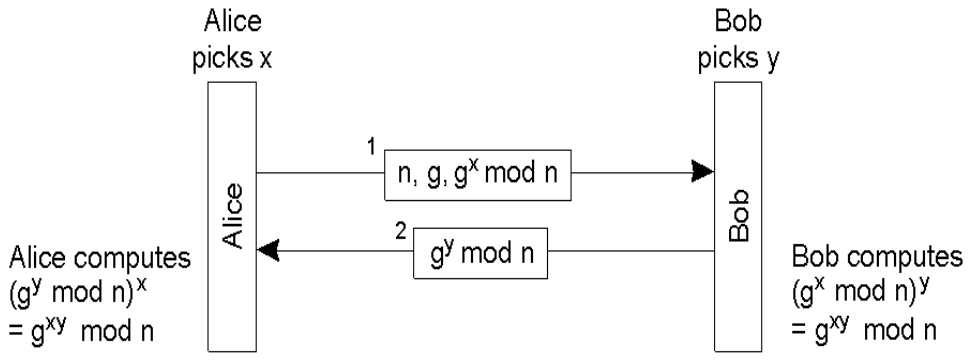


Fig 1.30 Hellman Key exchange

Diffie-Hellman can also be viewed as a public-key cryptography,

Where x and y are private keys, g^x mod n and g^y mod n are public keys.

Session key for Confidentiality

- Confidentiality: msg won't be intercepted and read by eavesdroppers
- Cryptographic keys are subject to "wear and tear"
- Lose of cryptographic keys leads to replay attacks
- Replacing cryptographic keys are expensive

Create a unique session key for each secure communication channel.

1.12 ZERO KNOWLEDGE

Zero-knowledge proofs are a general class of protocols between two parties, called the prover and the verifier. By means of a zero-knowledge proof, the prover convinces the verifier of the validity of a given statement without revealing any information beyond the truth of the Statement.

1.13 CRYPTOGRAPHIC PROTOCOLS

In zero-knowledge identification protocols, the statement proved is something like -I know the private key for this public key." But much more involved statements are possible, such as I know the private key for this public key or for that public key, and that in any case the private keys are different." In fact, the theory of zero-knowledge proofs tells us that any NP-statement can be proved efficiently in zero-knowledge.

1.14 APPLICATIONS

Some few Cryptography applications are listed below

- ❖ Anonymity networks
- ❖ Authentication methods
- ❖ Crypto-anarchism
- ❖ Digital rights management
- ❖ E-commerce
- ❖ Financial cryptography
- ❖ Kleptography
- ❖ Secret broadcasting
- ❖ WikiLeaks
- ❖ Cipherspace
- ❖ FISINT

 ISBN: 978-93-93810-96-0

- Identification friend or foe
- Pseudonymity
- Publicly Verifiable Secret Sharing
- Punchscan
- Radiofrequency MASINT
- Scantegrity
- Signals intelligence operational platforms by nation
- Steganography tools
- Traitor tracing
- X.1035

 ISBN: 978-93-93810-96-0

CHAPTER 2

OPERATING SYSTEM SECURITY

2.1 Memory Protection

Memory protection is a way to control memory access rights on a computer, and is a part of most modern operating systems. The main purpose of memory protection is to prevent a process from accessing memory that has not been allocated to it.

This prevents a bug or malware within a process from affecting other processes, or the operating system itself. An attempt to access unowned memory results in a hardware fault, called a segmentation fault or storage violation exception, generally causing abnormal termination of the offending process. Memory protection for computer security includes additional techniques such as address space layout randomization and executable space protection.

Memory protection keeps user programs from crashing one another and the OS .There are Two hardware-supported mechanisms

▫ Address translation ▫ Dual-mode operation

2.1.1 Address Translation

• Recall that each process is associated with an address space, or all the physical addresses a

Process can touch.

• However, each process s believes that it owns the entire memory, starting with the virtual address 0.

• The missing piece is a translation table to translate every memory reference from virtual to physical addresses

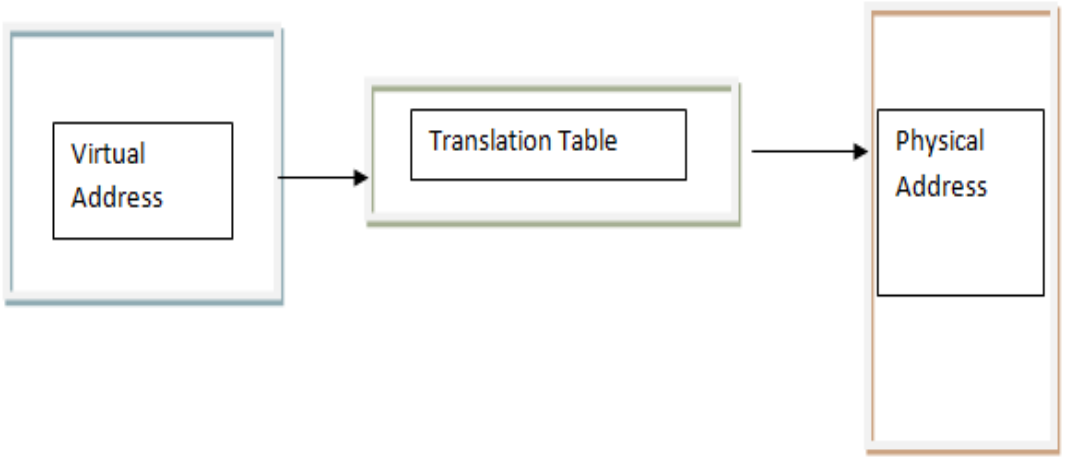


Fig 2.1 Address Translation Visualized

 ISBN: 978-93-93810-96-0

- **Translation provides protection**

❖ Processes cannot talk about other processes' addresses, nor about the OS addresses

❖ OS uses physical addresses directly

❖ No translations

2.1.2 Dual-Mode Operation Revisited

❖ Translation tables offer protection if they cannot be altered by applications

❖ An application can only touch its address space under the user mode

❖ Hardware requires the CPU to be in the kernel mode to modify the address translation tables

- **Details of Dual-Mode Operations**

Mode1 is Switching from the Kernel to User Mode is to run a user program, the kernel and Creates a process and initialize the address space .Loads the program into the memory with initializes translation tables' .Sets the hardware pointer to the translation table after that Sets the CPU to user mode and Jumps to the entry point of the program.

Mode2 is switching from User Mode to Kernel Mode contains **Voluntary** and **System calls**: a user process asks the OS to do something on the process's behalf Involuntary, Hardware interrupts (e.g., I/O) and Program exceptions (e.g., segmentation fault).

For all cases, hardware atomically performs the following steps

❖ Sets the CPU to kernel mode

❖ Saves the current program counter

❖ Jumps to the handler in the kernel

❖ The handler saves old register values

Unlike context switching among threads, to switch among processes it Need to save and restore pointers to translation tables to resume process execution at that time Kernel reloads old register values. Sets CPU to user mode process Jumps to the old program counter.

2.1.3 Communication between Address Spaces

Communication between Address Spaces should following process

❖ Processes communicate among address spaces via interprocess communication (IPC)

- ❖ Byte stream (e.g., pipe): A two-way data stream between two processes interfaced through standard input and output and read in one character at a time.
- ❖ Message passing (send/receive)
- ❖ File systems (e.g., read and write files)
- ❖ Shared memory
- ❖ Bugs can propagate from one process to another

- **Interprocess Communication**

It acts as Exchange of data between two or more separate, independent processes/threads.

Direct

- ❖ **Send (P1, message):** Pass a information to the requester or next one.
- ❖ **Receive (P2, message):** Get the acknowledgement from the receiver.
- ❖ **One-to-one communication :**

Indirect

- ❖ **Mailboxes or ports -** a port is software construct serving as a communications endpoint in a computer's host operating system.
- ❖ **Send (mailbox_A, message) -** Pass a information to the requester or next one.
- ❖ **receive(mailbox_A, message) -** Get the acknowledgement from the receiver
- ❖ **Many-to-many communication -**

2.1.4 Protection without Hardware Support

Hardware-supported protection can be slow

- ✓ Requires applications be separated into address spaces to achieve fault isolation
- ✓ What if your applications are built by multiple vendors? (e.g., Firefox plug-ins)
- ✓ Can we run two programs in the same address space, with safety guarantees? Protection via Strong .

2.1.5 Protection via Strong Typing

- ✓ Programming languages may disallow the misuse of data structures (casting) e.g., LISP and Java
- ✓ Java has its own virtual machines
- ✓ A Java program can run on different hardware and OSes
- ✓ Need to learn a new language

2.1.6 Protection via Software Fault Isolation

- ✓ Compilers generate code that is probably safe e.g., a pointer cannot reference illegal addresses
- ✓ With aggressive optimizations, the overhead can be as low as 5%
- ✓ A malicious user cannot jump to the last line and do damage, since safe is a legal address

2.2 ACCESS CONTROL

2.2.1 Introduction

Access control is the traditional center of gravity of computer security. It iswhere security engineering meets computer science. Its function is to control which principals (persons, processes, machines, ...) have access to which resources in the system — which files they can read, which programs they can execute, how they share data with other principals, and so on.

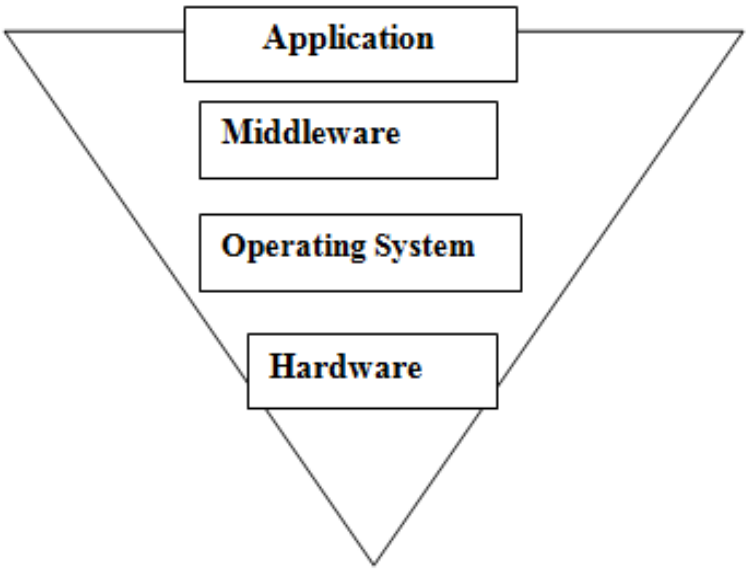


Fig 4.1: Access controls at different levels in a system

1. The access control mechanisms the user sees at the application level may express a very rich and complex security policy. (E.g Online business).
2. The applications may be written on top of middleware such as DBMS which enforces a number of protection properties.(E.g Bookkeeping software).
3. The middleware will use facilities provided by the underlying operating system.
4. Finally the operating system access controls will usually relay on hardware features provided by the processor.

As we work up from the hardware through the operating system and middleware to the application layer, the controls become progressively more complex and less reliable at Access control at the hardware, operating system and database levels.

Access control makes sense only in the context of a protection goal, typically expressed as a security policy. PCs carry an unfortunate legacy, in that the old single-user operating systems such as DOS and Win95/98 let any process modify any data; as a result many applications won't run unless they are run with administrator privileges and thus given access to the whole machine.

Now one of the biggest challenges in computer security is preventing one program from interfering with another. In addition, many everyday reliability problems stem from applications interacting with each other or with the system configuration. The single-user history of personal computing has got people used to all sorts of ways of working that are not really consistent with separating applications and their data in useful ways.

The pendulum is now swinging back. Hosting centers make increasing use of virtualization; having machines exclusively for your own use costs several times what you pay for the same resource on shared machines. The Trusted Computing initiative, and Microsoft Vista, place more emphasis on separating applications from each other; even if you don't care about security, preventing your programs from overwriting each others' configuration files should make your PC much more reliable.

More secure operating systems have led to ever more technical attacks on software other than the operatinEg system. The Trusted computing initiative, and Microsoft Vista, place more emphasis on separating applications from each other. three advanced protection techniques—**sandboxing, virtualization** and **Trusted Computing**.

- **Sandboxing** is an application-level control, run for example in a browser to restrict what mobile code can do.
- **Virtualization** runs underneath the operating system, creating two or more independent virtual machines between which information flows can be controlled or prevented.
- **Trusted Computing** is a project to create two virtual machines side-by-side, one being the 'old, insecure version of an operating system and the second being a more restricted environment in which security-critical opera.

2.2.2 Other Technical Attacks

In 2002, Microsoft announced a security initiative that involved every programmer being trained in how to write secure code. Other tech companies have launched similar training programmes. Despite the training and the tools, memory overwriting attacks are still appearing, to the great frustration of software company managers. However, they are perhaps half of all new vulnerabilities now rather than the 90% they were in 2001.

The other new vulnerabilities are mostly variations on the same general theme, in that they occur when data in grammar A is interpreted as being in grammar B. A stack overflow is when data are accepted as input and end up being executed as machine code. They are essentially failures of type safety.

A **format string vulnerability** arises when a machine accepts input data as a formatting instruction (e.g. %n in the C command printf()). This commonly arises when a programmer tries to print user-supplied data and carelessly allows the print command to interpret any formatting instructions in the string; this may allow the string's author to write to the stack.

SQL insertion attacks commonly arise when a careless web developerpasses user input to a back-end database without checking to see whether itcontains SQL code. The game is often given away by error messages, from which a capable and motivated user may infer enough to mount an attack.Another type of attack called an**integer manipulation attack**. Here, an overflow, underflow, wrap-around or truncation can result in the 'security' code writing an inappropriate number of bytes to the stack.

The late 1990's,most implementations of the Internet protocols allocated a fixed amount of buffer space to process the SYN packets with which TCP/IP connections are initiated. The result was SYN **flooding attacks:** by sending a large number of\ SYN packets, an attacker could exhaust the available buffer space and preventthe machine accepting any new connections. This is now fixed using **syncookies**.

The most recently discovered family of attacks of this kind are on **systemcall wrappers**. These are software products that modify software behavior by intercepting the system calls it makes and performing some filtering or manipulation.

2.2.3 Access Control Lists

Another way of simplifying the management of access rights is to store the access control matrix a column at a time, along with the resource to which the column refers. This is called an access control list or ACL.

 ISBN: 978-93-93810-96-0

2.3AUTHORIZATION

Authorization is the function of specifying access rights to resources related to information security and computer security in general and to access control in particular. More formally, "to authorize" is to define an access policy. For example, human resources staff is normally authorized to access employee records and this policy is usually formalized as access control rules in a computer system. During operation, the system uses the access control rules to decide whether access requests from (authenticated) consumers shall be approved (granted) or disapproved (rejected). Resources include individual files or an item's data, computer programs, computer devices and functionality provided by computer applications. Examples of consumers are computer users, computer programs and other devices on the computer.

Access control in computer systems and networks rely on access policies. The access control process can be divided into the following two phases: 1) policy definition phase where access is authorized, and 2) policy enforcement phase where access requests are approved or disapproved. Authorization is thus the function of the policy definition phase which precedes the policy enforcement phase where access requests are approved or disapproved based on the previously defined authorizations.

2.3.1 AUTHORIZATION SPECIFICATIONS

A) Policy Specification

Security policies must be exchangeable across domains.

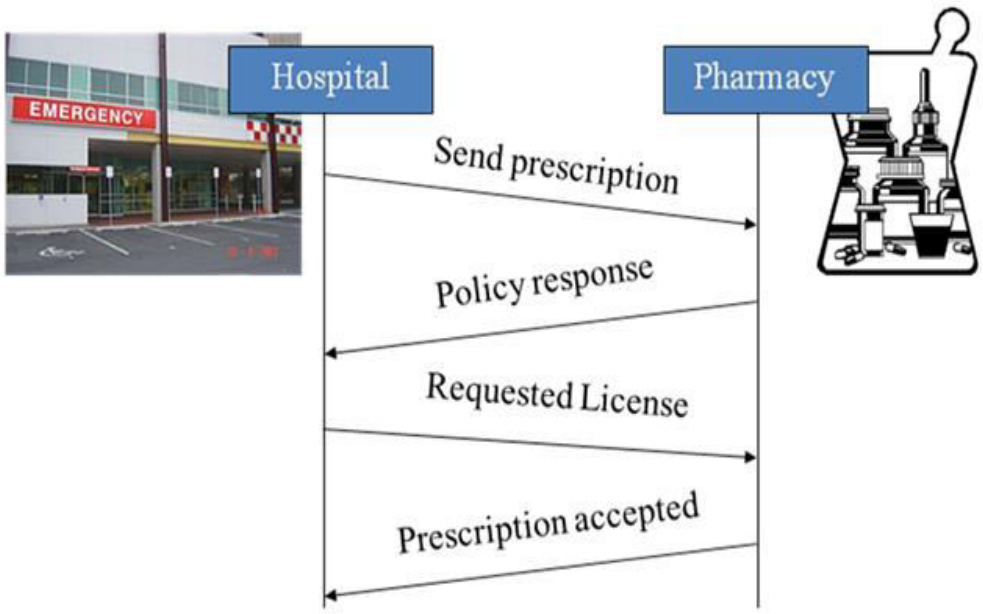


Fig 2.2 Policy Specification

Need policies to be exchangeable and understandable across domains in case attributes need to be gathered from outside the domain.

☐ There are several XML-based policy languages

■ WS-Policy (from Microsoft)

■ XACML (eXtensible Access Control Markup Language)

■ SAML (Security Assertion Markup Language)

In CAAC, WS-Policy was chosen as the specification language because it is inherently supported in the Microsoft .NET framework.

2.3.2 WS-Policy Overview

☐ Why:

■ To describe service requirements, preferences, and capabilities of web services

☐ Goal:

■ Provide the general purpose model and syntax to describe and communicate the policies of a Web service

☐ What:

■ Provides a flexible and extensible grammar for expressing the capabilities, requirements, and general characteristics of Web Services

C) CAAC Policy Specification

☐ Our customized WS-Policy tagsFor any authorization policy AP = (S, P, C)

<wsa:DataType>	specifies the data object or data type of permission P
<wsa:AccessType>	specifies the operation mode of permission P
<wsa:Permission>	specifies the permission P in an AP
<wsse:SubjectToken>	specifies the security token issued to S
<wsse:ContextToken>	specifies one context condition in C
<wsse:ContextType>	specifies which context type is used in one context condition of C

A Sample Policy

<wsp: Policy>

<wsp: AppliesTo>

<wsa: EndpointReference>

 ISBN: 978-93-93810-96-0

<wsa: DataType>PatientRecord</wsa: DataType>

<wsa: AccessType>Delete</wsa: AccessType>

<wsa: Permission>DeletePatientRecord</wsa: Permission>

</wsa: EndpointReference>

</wsp: AppliesTo>

<wsse: SubjectToken wsp: Usage="Required">

<wsse: TokenType>Medical Records Staff</wsse:Token Type>

</wsse: SubjectToken>

<wsp: OneOrMore wsp: Usage="Required">

<wsp: All>

<wsse: ContextToken wsp:Usage="wsp:GreatThan“
wsp:Preference="T(password)">

<wsse: ContextType>Trust Level</wsse: ContextType>

</wsse: ContextToken>

</wsp: All>

</wsp: OneOrMore>

</wsp: Policy>

Fig 2.3 Sample Policy Code

2.3.3 XACML

- OASIS standard version 1.1 (2.0 and 3.0)
- Policy language
- Access control decision request/response language

XACML - Policies

- Policy Set: container of policies (local and remote)
- Policy: a set of rules
- Rule: a target, effect, and condition
- Target: a resource, subject, and action
- Effect: results of rule; “Permit” or “Deny”
- Condition: Boolean; “True,” “False,” or “Indeterminate”

 ISBN: 978-93-93810-96-0

XACML – Access Control

- ☐ Reconciles
- ■ Multiple policies
- ■ Multiple rules per policy
- ■ Multiple control decisions
- ☐ Use a combining algorithm to combine multiple decisions into a single decision
- ☐ Use standard or customized algorithms
- ☐ Policy Combining Algorithms—used by PolicySet
- ☐ Rule Combining Algorithms—used by Policy

XACML – Policy Evaluation

- ❖ Obtain attributes from subject
- ❖ Compare obtained attributes with attributes accepted by the policy
- ❖ Evaluate conditions using standard or customized functions
- ❖ E.g. The function [type]-one-and-only looks in a "bag" of attribute values and returns the single value if there is one or an error if there are zero or multiple.

2.3.4 SAML assertions

- ☐ An assertion is a declaration of facts about a subject
- ☐ SAML has three kinds, all related to security:

1. Authentication
2. Attribute
3. Authorization decision

- ☐ You can extend SAML to make your own kinds of assertions

Some common information in all assertions

- ☐ Issuer and issuance timestamp
- ☐ Assertion ID
- ☐ Subject
- ■ Name plus the security domain
- ■ Optional subject confirmation, e.g. public key
- ☐ "Conditions" under which assertion is valid
- ■ SAML clients must reject assertions containing unsupported conditions

- Special kind of condition: assertion validity period
- Additional "advice"
- E.g., to explain how the assertion was made

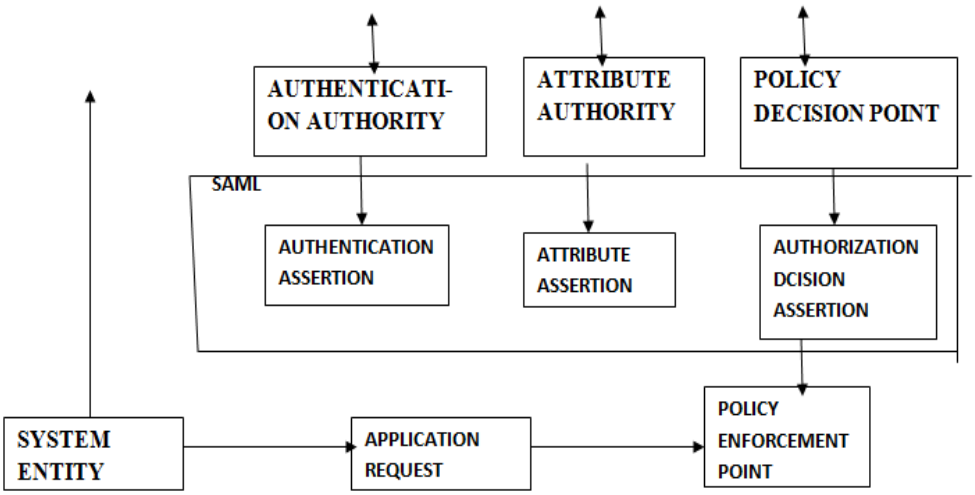


Fig 2.4 SAML conceptual model

Authentication assertion:

- An issuing authority asserts that:
- subject S
- was authenticated by means M
- at time T
- **Caution:** Actually checking or revoking of credentials is not in scope for SAML!
- It merely lets you link back to acts of authentication that took place previously

<saml:Assertion
MajorVersion="1" MinorVersion="0"
AssertionID="128.9.167.32.12345678"
Issuer="University of Virginia"
IssueInstant="2003-12-03T10:02:00Z">
<saml:Conditions
NotBefore="2003-12-03T10:00:00Z"
NotAfter="2003-12-03T10:05:00Z" />
<saml:AuthenticationStatement
AuthenticationMethod="password"
AuthenticationInstant="2003-12-03T10:02:00Z">
<saml:Subject>

<saml:NameIdentifier
SecurityDomain="virginia.edu"
Name="jim" />
</saml:Subject>
</saml:AuthenticationStatement>
</saml:Assertion>

Fig 2.5 Sample authentication assertion

Authorization decision assertion

- An issuing authority decides whether to grant the request:
  - by subject S
  - for access type A
  - to resource R
  - given evidence E
- The subject could be a human or a program
- The resource could be a web page or a web service, for example

2.3 AUTHENTICATING USERS

Authentication is basically the process to confirm that a person (or user) is who they say they are. There are three classic "factors" that can be used to confirm a users' identity:

- Using something only the user knows
- Using something unique the user has
- Using something that only the user is

For example Password and human fingerprint

2.4.1 Swipe Card

A Swipe card is a type of card capable of storing data by modifying the magnetism of tiny iron-based magnetic particles on a band of magnetic material on the card. The magnetic stripe, sometimes called **magstripe**, is read by swiping past a magnetic. Magnetic stripe cards are commonly used in credit cards, identity cards, and transportation tickets.

2.4.2 Smart Card

Smart cards can provide identity documentation, authentication, data storage, and application processing. Smart cards may provide strong security authentication for single sign-on (SSO) within large organization.

 ISBN: 978-93-93810-96-0

Fig 2.6 Smart Card

2.3.3 SIM Cards

A subscriber identity module or subscriber identification module (SIM) is an integrated circuit that is intended to securely store the international mobile subscriber identity (IMSI) and the related key used to identify and authenticate subscribers on mobile telephony devices (such as mobile phones and computers).

Fig 2.7 SIM Card

2.3.4 Iris identification

Iris recognition is an automated method of biometric identification that uses mathematical pattern-recognition techniques on video images of one or both of the irises of an individual's eyes, whose complex random patterns are unique, stable, and can be seen from some distance.

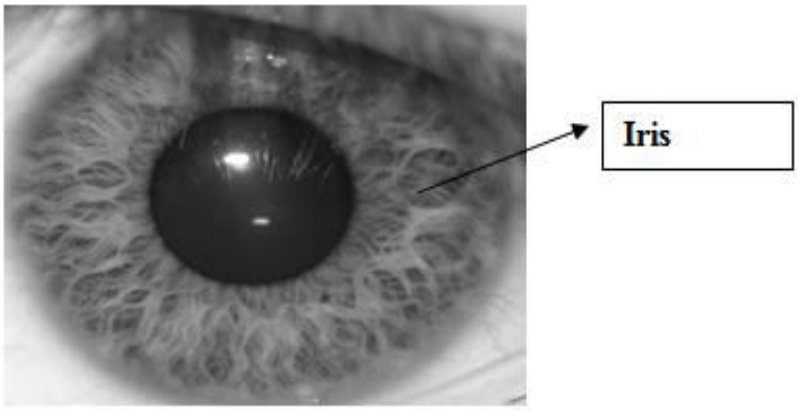


Fig 2.8 Iris-Authentication

2.3.5 Voice Recognition

The term voice recognition or speaker identification refers to identifying the speaker, rather than what they are saying. Recognizing the speaker can simplify the task of translating speech in systems that have been trained on a specific person's voice or it

can be used to authenticate or verify the identity of a speaker as part of a security process.

2.4.6 Fingerprint Authentication

Fingerprint recognition or fingerprint authentication refers to the automated method of verifying a match between two human fingerprints. Fingerprints are one of many forms of biometrics used to identify individuals and verify their identity.

2.4.7 Passwords

A password is a word or string of characters used for user authentication to prove identity or access approval to gain access to a resource (example: an access code is a type of password), which should be kept secret from those not allowed access such as the personal identification number (PIN) commonly used for ATM access. Passwords are generally short enough to be easily memorized and typed.

2.4 SECURITY ENFORCEMENT AND EVALUATION

2.5.1 Introduction

- Development started about 1978 based on work performed in the late 60ies and early 70ies:
- Grace Nibaldi: Proposed Technical Evaluation Criteria for Trusted Computer Systems, 25 October 1979, The Mitre Corporation.
- James P. Anderson: Computer Security Technology Planning Study, October 1972.
- Paul A. Karger, Roger R. Schell: Multics Security Evaluation: Vulnerability Analysis, June 1974
- David E. Bell, Leonard J. LaPadula: Secure Computer System: Unified Exposition and Multics Interpretation

2.5.2 Evolution

- Trusted Computer System Evaluation Criteria (TCSEC) is a United States Government Department of Defense (DoD) standard that sets basic requirements for assessing the effectiveness of computer security controls built into a computer system. The TCSEC was used to evaluate, classify and select computer systems being considered for the processing, storage and retrieval of sensitive or classified information.
- The TCSEC, frequently referred to as the Orange Book, is the centerpiece of the DoD Rainbow Series publications. Initially issued in 1983 by the National Computer Security Center (NCSC), an arm of the National Security Agency, and then updated in 1985,

- TCSEC was replaced by the Common Criteria international standard originally published in 2005.
- Policy - The security policy must be explicit, well-defined and enforced by the computer system. There are two basic security policies:
- Mandatory Security Policy - Enforces access control rules based directly on an individual's clearance, authorization for the information and the confidentiality level of the information being sought. Other indirect factors are physical and environmental. This policy must also accurately reflect the laws, general policies and other relevant guidance from which the rules are derived.
- Marking - Systems designed to enforce a mandatory security policy must store and preserve the integrity of access control labels and retain the labels if the object is exported.
- Discretionary Security Policy - Enforces a consistent set of rules for controlling and limiting access based on identified individuals who have been determined to have a need-to-know for the information.
- Assurance: The computer system must contain hardware/software mechanisms that can be independently evaluated to provide sufficient assurance that the system enforces the above requirements. By extension, assurance must include a guarantee that the trusted portion of the system works only as intended. To accomplish these objectives, two types of assurance are needed with their respective elements:

➢ **Assurance Mechanisms :** Operational Assurance: System Architecture, System Integrity, Covert Channel Analysis, Trusted Facility Management and Trusted Recovery

➢ **Life-cycle Assurance :** Security Testing, Design Specification and Verification, Configuration Management and Trusted System Distribution

2.5 TRUSTED DEVICES

2.6.1 Definitions: Trust vs. Security

Here we discuss the term “trusted operating system” and specify why we prefer the term to something such as “secure operating system”. Basically, security is not a quality that can be quantified easily. Either a system is secure or it is not secure. If a system is called secure, it should be able to resist all attacks. The claim of security is something that has to be taken as is, either one accepts the claim or one does not.

Trust, on the other hand, is something that can be quantified. A system is called trusted if it meets the intended security requirements; thus one can assign a level of

trust to a system depending on the degree to which it meets a specific set of requirements.

The evaluation of the trust to be accorded a system is undertaken by the user of the system and depends on a number of factors, all of which can be assessed.

1) The enforcement of security policy, and

2) The sufficiency of its measures and mechanisms.

6.2.2 Security Policies

A **security policy** is a statement of the security we expect a given system to enforce. A system can be characterized as trusted only to the extent that it satisfies a security policy.

All organizations require policy statements. The evolution of policy is perhaps a fairly dull job, but it is necessary. Policy sets the context for the rules that are implemented by an organization. For this course, we focus on information security policy, used to give a context for the rules and practices of information security. Policy sets the strategy – it is the "big picture", while rules are often seen as the "little picture". The text states that "Policy sets rules".

6.2.3 Trusted Operating System

Designed from the beginning with security in mind, each of these main services is written as a distinct object with its own security controls, especially user **authentication**, **least privilege** and **complete mediation**. Here the UNIX operating system shows its major flaw – users are either not trusted or, being super-users, given access to every resource.

The bottom line here is that computers are fast and memory is cheap. A recent check
(10/31/2003) of the Gateway web site found a server configured with a 3.08 GHz processor, 512 KB of cache memory, and 4GB of main memory. We might as well spend a few of these inexpensive resources to do the job correctly.

Some of the features of a security-oriented operating system are obvious, while other features require a bit of explanation. We discuss those features that are not obvious.

Mandatory access control (MAC) refers to the granting of access by a central authority, not by individual users. If I have SECRET data to show you and you do not have a SECRET clearance, I cannot of my own volition grant you a SECRET clearance. MAC should exist along with **Discretionary Access Control** (DAC) in that objects not managed by the central authority can be managed by the individual user owning them.

Object reuse protection refers to the complete removal of an object before it is returned to the object pool for reuse. The simplest example of this is protection of files. What happens when a file is deleted. In many operating systems, the file allocation table is modified to no longer reference the object and to place its data sectors on the **free list** as available for reuse. Note that the data sectors are not overwritten, so that the original data remains.

Object reuse protection also has a place in large object-oriented systems. In these systems, the creation of some objects is often very computationally intense. This leads to the practice of pooling the discarded objects rather than actually destroying the object and releasing the memory when the object is no longer in use. A program attempting to create a new object of the type in the pool will get an object already created if one exists in the pool. This leads to more efficient operation, but also introduces a security hole.

Audit log management refers to the practice of logging all events with potential security impact, protecting that log from unauthorized access and modification, and creation of procedures and software to examine the log periodically and analyze it for irregularities. A security log is of no use if nobody looks at it.

Intrusion detection refers to the creation and use of system software that scans all activity looking for unusual events. Such software is hard to write, but one should try. For example, this author has a 128 MB flash drive that he occasionally attaches to his computer at work via the USB port. The intrusion detection software always reports that the number of hard drives on the system has changed and says to call the administrator if this was not an intentional act.

6.2.4 Kernelized Design

A **kernel** is the part of an operating system that performs low-level functions. This is distinct from the high-level services part of the operating system that does things such as handle shared printers, provides for e-mail and Internet access, etc. The kernel of an operating system is often called the **nucleus**, and rarely the **core**. In an operating system designed with security in mind there are two kernels: the **security kernel** and the **operating system kernel**, which includes the security kernel.

The security kernel is responsible for enforcing the security mechanisms of the operating system, including the handling of most of the functions normally allocated to the operating system kernel itself, as most of these low-level facilities have impact on security.

The **reference monitor** is one of the most important parts of the security kernel. This is the process that controls access to all objects, including devices, files, memory, interprocess communication, and other objects. Naturally, the

reference monitor must monitor access to it self and include protection against its being modified in an unauthorized way.

6.2.5 The Trusted Computing Base (TCB)

The trusted computing base is the name given to the part of the operating system used to enforce security policy. Naturally, this must include the security kernel. Functions of the TCB include:

1. Hardware management, including processors, memory, registers, and I/O devices.
2. Process management, including process scheduling.
3. Interrupt handling, including management of the clocks and timing functions.
4. Management of primitive low-level I/O operations.

Virtualization is one of the more important tools of a trusted operating system. By this term we mean that the operating system emulates a collection of the computer system's sensitive resources. Obviously virtualized objects must be supported by real objects, but the idea is that these real objects can be managed via the virtual objects.

As an example of a virtualized object, consider a shared printer. The printer is a real object to which it is possible to print directly. Simultaneous execution of several programs, each with direct access to the printer would yield an output with the results of each program intermixed – a big mess. In fact the printer is virtualized and replaced by the **print spooler**, which is the only process allowed to print directly to the printer. Each process accessing the virtualized printer is really accessing the print spooler, which writes the data to a disk file associated with the process. When the process is finished with the printer, the spooler closes the file, and queues it up for being printed on the real printer.

A **virtual machine** is a collection of hardware facilities, each of which could be real or simulated in software. One common feature is **virtual memory**, in which each process appears to have access to all of the memory of the computer, with the possible exception of memory allocated to the operating system.

6.2.6 Assurance in Trusted Operating Systems

For an operating system designed to be secure, **assurance** is the mechanism for convincing others that the security model is correct, as are the design and implementation of the OS. How does one gain confidence that an operating system should be trusted? One way is by gaining confidence that a number of the more obvious security vulnerabilities have been addressed in the design of the system.

Input/Output processing represents one of the larger vulnerabilities in operating systems. There are a number of reasons for the vulnerability of this processing, including

1) The fact that I/O processing is interrupt driven, and

2) The fact that I/O processing is often performed by independent hardware systems, and

3) The complexity of the I/O code itself, and

4) The desire to have the I/O process bypass the security monitors as an efficiency issue.

Methods for gaining assurance include testing by the creator of the software, formal testing by a unit that is independent of the software development process, formal verification (when possible – it is very difficult), and formal validation by an outside vendor. The author of these notes had been part of a software V&V (verification and validation) team, assigned to be sure that the code was written correctly and that it adhered to the requirements.

6.2.7 Formal Evaluation

One of the earliest attempts for formal evaluation was called the **Trusted Computer System Evaluation Criteria (TCSEC)**, more loosely the "Orange Book" because that was the color of the book. This was published in the late 1970's by the U. S. Department of Defense. The TCSEC defined a number of levels of assurance.

D – Basically, no protection. Any system can get this level.

C1 –discretionary access control

C2 – controlled access protection

B1 – labeled security protection

Each object is assigned a security level and mandatory access controls are used.

B2 – structured protection. This is level B1 with formal testing of a verified design.

B3 – security domains. The security kernel must be small and testable.

A1 – verified design. A formal design exists and has been thoroughly examined.

The TCSEC was a good document for its day, but it was overtaken by the arrival of the Internet and connectivity to the Internet. Several operating systems were rated as C1 or better, provided that the system was running without connection to the Internet.

More recently, the U. S. Government has published the Combined Federal Criteria, followed in 1998 by the **Common Criteria**. This document proposed a number of levels of assurance.

2.6 DIGITAL RIGHTS MANAGEMENT

The basic problem is that a PC, being a general-purpose computer, can in principle copy any file and send it to any other computer; unlike with analogue copying, copies are perfect, so millions of copies might be made from one original. The problem is compounded by the fact that, from the viewpoint of the content vendor, the PC owner is the 'enemy'. The music industry believed that unlimited copying would destroy their business; the computer industry told them that DRM was intrinsically impossible on a general-purpose computer, so they'd better get a new business model. The music and film industries, despite being a tenth of the computer industry's size, had much more clout in Congress. So Hollywood got its way. The result is a number of products generally referred to as digital rights management or DRM.

2.7.1 Windows Media Rights Management

WMP replaced an earlier media player application when Windows 98 was released. It enables a user to play music, watch video and view photos, and has all sorts of features from MP3 player support to synchronization of lyrics for Karaoke. However our main interest here is its ability to play files protected using Windows Media Rights Management (WMRM).

A store wanting to sell digital media encrypts each item using a content key and puts the encrypted files on a streaming media server that is typically linked to their web site. In order to access a media object, the customer must get hold of a license, which consists of the object identifier, the license key seed, and a set of instructions in a rights management language which state what she can do with it; how many times she may play it, whether she's allowed to burn it to a CD, and so on. The license is generated by a license server and encrypted using her public key. The license acquisition may involve her going to a registration or payment page, or it may happen silently in the background.

2.7.2 Other Online Rights-Management Systems

The Microsoft offering is fairly typical of rights-management systems. Apple's FairPlay, which is used in the iPod and in its media player QuickTime, also has tunes encrypted under master keys, and when a tune is bought the user is sent the master key encrypted under a random session key, plus the session key encrypted under his iTunes player's RSA public key. Session keys are backed up online on Apple's servers. As with Windows, a number of programs appeared that unlocked

 ISBN: 978-93-93810-96-0

protected content, and Apple duly upgraded iTunes to stop these programs working in September 2006.

In the mid-2000s there were growing calls from consumer and industry groups for DRM systems to become more interoperable. Real Networks is the other major supplier of media player software for PCs, and its RealPlayer uses its own proprietary DRM. In a tussle reminiscent of the word-processing file format wars of the 1980s, Real made their music tracks readable by iTunes in 2004; Apple responded by threatening litigation, and ended the compatibility in a 2006 upgrade.

 ISBN: 978-93-93810-96-0

CHAPTER 3

NETWORK SECURITY

3.1 FIREWALLS

A **firewall** is a network security system that controls the incoming and outgoing network traffic based on an applied rule set. A firewall typically establishes a barrier between a trusted, secure internal network and another network that is assumed not to be secure and trusted.

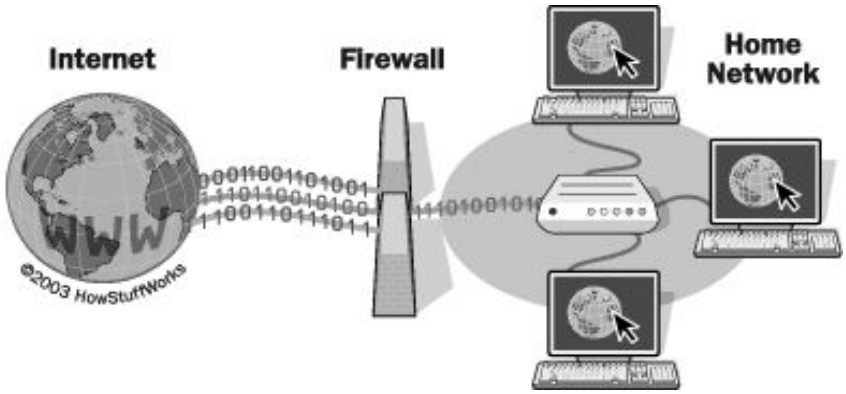


Fig 3.1 Firewall -An illustration of where a firewall would be located in a network.

Firewalls are often categorized as either "network firewalls", where afirewall software appliance running on general purpose hardware, or a hardware-based firewall computer appliances filters traffic between two or more networks - or "host-based firewalls", where a layer of software on one host controls network traffic in and out of that single machine.

3.1.1 Purpose of Firewall

Filtering: The primary purpose of a firewall is packet filtering. When a computer sends a request across the Internet, it takes the form of small packets of data, which travel through the network to their destination. The target server responds with its own packets of data, which return along the same route.

Logging: Another important aspect of a firewall is its ability to log any traffic that passes through it. By recording the information from packets that pass through or that it discards

Internal Threats: While the primary goal of a firewall is to keep attackers out, it also serves a valuable purpose by monitoring outgoing connections.

3.1.2 Why Firewalls?

❖ Internet connectivity has become essential for most organizations.

- ❖ The Internet was not designed to be secure.It was created for open access to research
- ❖ The Internet suffers from major security issues .Allows adversaries to attack or gain access to many private networks

3.1.3 Benefits of a Firewall

- ➢ Protect from vulnerable services
- ■ Allows administrator to deny services deemed vulnerable such as NFS & NIS
- ➢ Network logging & statistics
- ■ Collects information on all traffic passing in/out of network
- ■ Monitors traffic for suspicious activity & attacks
- ➢ Limit external access to internal systems
- ■ Can pick which hosts are accessible from external networks
- ■ All others can be denied access
- ■ Can be done for specific internal and external systems
- ➢ Enhanced privacy
- ■ Ability to block or hide DNS information of all internal hosts
- ■ Only the IP address of the firewall is available from the Internet
- ➢ Concentrated security
- ■ Only need to ensure firewall is void of vulnerabilities to secure network assuming no backdoors exist
- ➢ Policy enforcement
- ■ A firewall offers a method to enforce the network policy of an organization

3.1.4 Disadvantages of Firewalls

- ➢ Backdoors may exist
- ■ Firewalls cannot protect against hosts that connect to ISP through dial-up service, wireless connectively, or other methods
- ➢ No protection from insider attacks
- ■ Offers no solution to protect against disgruntled employees wishing to damage the network
- ■ Internal employees can still download sensitive information and take it offsite

- Blocking of required services
- Could block access to services employees need such as FTP and Telnet
- Considered an "all eggs in one basket" approach
- Adversary who successfully bypasses the firewall will have access to internal hosts
- Does not offer virus protection
- Viruses can be hidden within software or internal authorized users could download viruses
- Firewalls do not offer virus checking
- Would degrade performance
- Constant updates would be required
- Would offer users a false sense of security

3.1.5 Firewall Policy Design

- Two major types of policy:
- Permit all services unless specifically denied
- Deny all services unless specifically permitted
- The first policy is less secure & allows dangerous services not denied by the firewall
- The second is stronger and more secure, but has higher probability of impacting users
- Administrator should find the proper mixture that allows maximum security with minimum user interference

3.1.6 Strong Authentication

- Externally accessing the network using the same username and password is dangerous.
- Valid when sending passwords in the clear or unencrypted
- Protocol analyzers or "sniffers" are used to determine this information and access the network
- One-time passwords avoid the replay of passwords since the same password is never user twice
- Examples include smartcards & authentication tokens

3.1.6 TYPES OF FIREWALLS

Packet-filtering routers

- Applies a set of rules to individual IP packets as they arrive

Application gateways / proxy servers

- Acts as a buffer for services between the internal and external network

Circuit level gateways

- Works by never allowing end-to-end TCP connections

Details of Packet-Filtering Routers

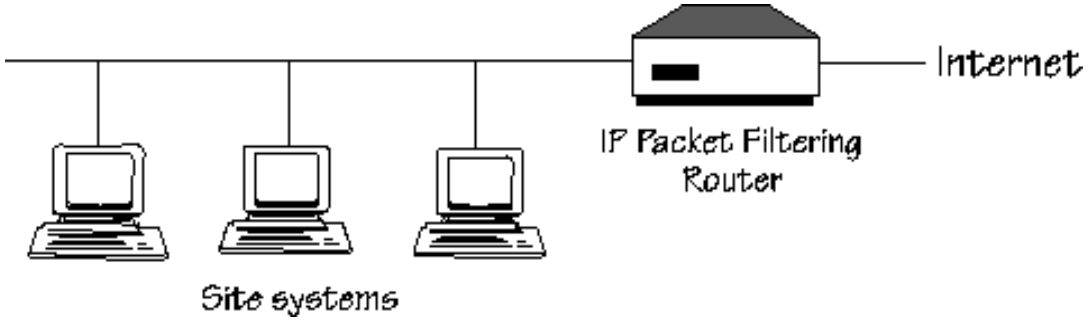


Fig 3.2 Packet filtering techniques

Filtering rules based upon fields:

- Source IP address
- Destination IP address
- TCP/UDP source port
- TCP/UDP destination port

Firewall administrator generates rules at the router to deny or allow access between an internal and external host.Examples of filtered ports include:

- Port 111 – RPC which can be used to steal system information such as passwords
- Port 69 – TFTP which can read system files if improperly configured

Benefits of packet-filtering:

- Fast, flexible, and transparent
- Considered an inexpensive alternative
- Routers are typically in place and only require configuration

Vulnerabilities of Packet-Filtering Routers

- Address & port spoofing
- Some routers can not identify altered address information on network packets

- This allows adversaries to bypass the firewall and gain access to the internal network
- Little or no logging capabilities
- Routers are designed for network performance, not security
- Without logging capabilities, it is almost impossible to identify when the network is under attack
- Lack of strong user authentication
- Typically, this feature is not supported by routers which allows the use of "sniffers" by adversaries to gather passwords
- Router rules are complex
- Some routers do not filter on TCP/UDP source ports which makes filtering more difficult
- It is common for an administrator to modify one rule while unknowingly opening up a vulnerability
- Routers usually offer no testing methods to insure the rules work
- This allows for "holes" in the firewall that can be used to gain access to the network
- RPCs (remote procedure call) are difficult to filter
- A number of RPC services are assigned ports randomly at start-up
- This makes it difficult for the router to determine which ports RPC services reside
- The router will not be able to apply filtering rules without knowing the port information

Details of Application Gateways/Proxy Servers

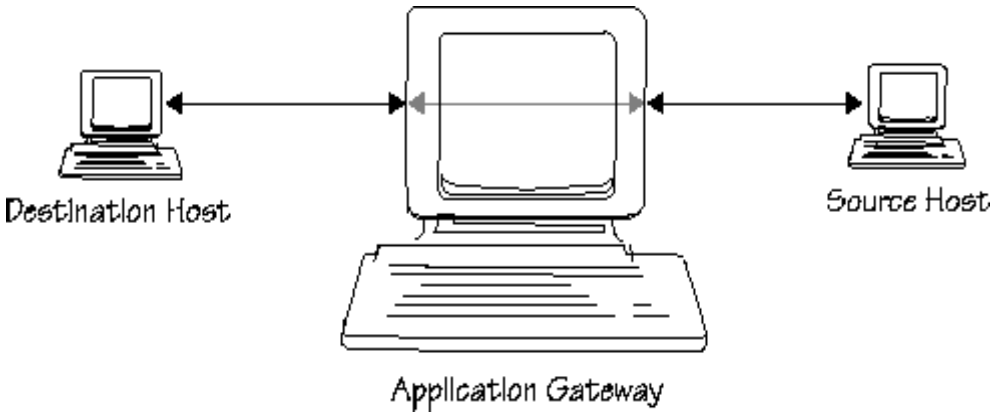


Fig 3.3 Gateways

- Considered a very secure type of firewall
- Application gateway is the only host visible to the outside network
- Requires all connections to pass through the gateway
- Proxies are typically designed & tested to be secure
- Built not to include every feature of the application, but rather to authenticate the requesting user
- Generally supports comprehensive logging & strong authentication practices
- This allows for higher levels of security & protection
- Only allows services to pass through for which there is a proxy
- i.e. – if the gateway only has a proxy for FTP & TELNET then these are the only services allow to pass. All other requests would be denied
- Inability to defend against content related attacks
- i.e. – An authorized user downloading an executable from an untrusted network that contains a virus.
- Not all services are supported by proxies
- If this service is required by an organization, then it will not be protected by the application gateway and leaves the network open to attack

Details of Circuit Level Gateways

- A gateway is system based upon two separate TCP connections
- One between itself & the internal host
- The second between itself & the external host
- Circuit level gateways are used where the administrator trusts internal users
- The advantage is to reduce processing overhead by only examining incoming application data
- Network security function is based upon which incoming connections will be allowed
- Possible to circumvent the firewall if circuit level firewall is configured incorrectly
- Internal users can advertise services on non-standard ports
- These services would then be available to the outside network
- They do not offer any better control than a router

- Operate only on the network layer which means traffic is not monitored or controlled on the application level

Screened Host Firewall

- Allows for more flexibility than a dual-homed firewall
- The cost of the increased flexibility is decreased security
- Flexibility is created because the router is allowed to bypass the application gateway for specified trusted services
- Application gateway's proxy service passes all services for which proxies exist.
- Router filters inherently dangerous protocols from reaching the application gateway
- It accepts or rejects traffic according to a specified set of rules
- The major vulnerability exists within the router due to the complex router rules previously discussed

Distributed Firewalls

- The distributed firewall concept has a centrally defined security policy
- Enforcement occurs at individual endpoints such as hosts & routers
- The goal is to keep the traditional model of the firewall in place while fixing their shortcomings such as:
- Internal traffic cannot be filtered since it is not examined by the network
- Firewalls can become congestion points
- Backdoor access such as dial-up or wireless connections
- End-to-end encryption prevents firewalls from looking at packets for filtering
- Implementation of a distributed firewall requires three components
- A language for expressing policies & resolving requests that supports credentials for delegation of rights & authentication
- A mechanism for safely distributing security policies such as IPSec
- A method for applying security policy to incoming packets or connections
- The research of Ioannidis, Keromytis, Bellovin & Smith (2000) focuses on a system called KeyNote Trust Management System
- Makes use of public key cryptography for authentication in a decentralized environment

- Selected results of a distributed firewall system
- Performance bottleneck is eliminated since network is no longer dependent on a single firewall
- Backdoor connections no longer present vulnerabilities
- End-to-end encryption is possible without compromising security
- Internal network users are no longer automatically trusted on the network
- A distributed firewall system demands the highest quality administration tools in order to function correctly

3.2 INTRUSION DETECTION SYSTEMS

3.2.1 Intrusion Detection Systems (IDS)

- Intrusion Detection Systems look for attack signatures, which are specific patterns that usually indicate malicious or suspicious intent. Different ways of classifying an IDS

IDS based on

– anomaly detection

– signature based misuse

– host based

– network based

3.2.2 Anomaly based IDS

- This IDS models the normal usage of the network as a noise characterization.
- Anything distinct from the noise is assumed to be an intrusion activity.

– E.g flooding a host with lots of packet.

- The primary strength is its ability to recognize novel attacks.

Drawbacks of Anomaly detection IDS:

- Assumes that intrusions will be accompanied by manifestations that are sufficiently unusual so as to permit detection.
- These generate many false alarms and hence compromise the effectiveness of the IDS.

3.2.3 Signature based IDS

- This IDS possess an attacked description that can be matched to sensed attack manifestations.

- The question of what information is relevant to an IDS depends upon what it is trying to detect.

– E.g DNS, FTP etc.

- ID system is programmed to interpret a certain series of packets, or a certain piece of data contained in those packets,as an attack. For example, an IDS that watches web servers might be programmed to look for the string “phf” as an indicator of a CGI program attack.
- Most signature analysis systems are based off of simple pattern matching algorithms. In most cases, the IDS simply looks for a sub string within a stream of data carried by network packets. When it finds this sub string (for example, the ``phf'' in ``GET /cgi-bin/phf?''), it identifies those network packets as vehicles of an attack.

Drawbacks

- They are unable to detect novel attacks.
- Suffer from false alarms
- Have to programmed again for every new pattern to be detected.

3.2.4 Host/Applications based IDS

- The host operating system or the application logs in the audit information.
- These audit information includes events like the use of identification and authentication mechanisms (logins etc.) , file opens and program executions, admin activities etc.
- This audit is then analyzed to detect trails of intrusion.

Drawbacks

- The kind of information needed to be logged in is a matter of experience.
- Unselective logging of messages may greatly increase the audit and analysis burdens.
- Selective logging runs the risk that attack manifestations could be missed.

Strengths of the host based IDS

- Attack verification
- System specific activity
- Encrypted and switch environments
- Monitoring key components
- Near Real-Time detection and response.

- No additional hardware

3.2.5 Stack based IDS

- They are integrated closely with the TCP/IP stack, allowing packets to be watched as they traverse their way up the OSI layers.
- This allows the IDS to pull the packets from the stack before the OS or the application have a chance to process the packets.

3.2.6 Network based IDS

- This IDS looks for attack signatures in network traffic via a promiscuous interface.
- A filter is usually applied to determine which traffic will be discarded or passed on to an attack recognition module. This helps to filter out known un-malicious traffic.

Strengths of Network based IDS

- Cost of ownership reduced
- Packet analysis
- Evidence removal
- Real time detection and response
- Malicious intent detection
- Complement and verification
- Operating system independence

3.2.7 Commercial ID Systems

- ISS – Real Secure from Internet Security Systems:

– Real time IDS.

– Contains both host and network based IDS.

- Tripwire – File integrity assessment tool.
- Bro and Snort – open source public-domain system.

3.2.8 Bro: Real time IDS

- Network based IDS
- Currently developed for six Internet applications: FTP, Finger, Portmapper, Ident, Telnet and Rlogin.

 ISBN: 978-93-93810-96-0

3.3 VIRUSES AND WORMS

3.3.1 VIRUSES

Vital Information Resources Under Seize (VIRUS) A a program that replicates by "infecting" other programs, so that they contain a copy of the virus Or A computer virus is a computer program that can copy itself and infect a computer without permission or knowledge of the user.

Virus is a malicious computer program designed to cause at best annoyance and at worst, damage to computer data. They usually spread from computer to computer through internet or any communication or interacting media. They often hide themselves in innocuous looking files. They lie dormant waiting for a trigger date for launching.

How?

- Viral code is attached or "inserted" into the order of execution so that when the legitimate code is run the viral code is also run or run instead of the legitimate code.
- May be "tacked" on to the end of an executable file or inserted into unused program space.
- Legitimate code must be modified so that the viral code is branched/vectored to.

3.3.2 Most viruses:

- Do not damage the original program or damage the hardware

May damage data files-"trash" firmware and Mess up boot records

- But, some do
- For this reason most can be cleaned up with anti-virus software.

The Normal Virus works like this:

- User call for a legitimate program
- The virus code, having inserted itself in the order of execution, executes instead or in addition to the legitimate program.
- The virus code terminates and returns control to the legitimate program

"In The Wild" A virus is said to be "in the wild" when it has either escaped or been released from its controlled or development environment to the general population.For a virus to be considered In the Wild, it must be spreading as a result of normal day-to-day operations on and between the computers of unsuspecting users.

The Wild list According to wildlist.org:

To be considered "in the wild" a virus must be reported by two or more virus professionals who report to the Wild list Organization. Must also be accompanied by replicated samples and this strictness insures that Wild list viruses are definitely out there doing damage. Its works like as such as follow

Basic structure:

{

look for one or more infectable objects

if (none found)

exit

else

infect object

}

Doesn't remain in memory, but executes all of the viral code at once then returns control to the infected program

Memory Resident Viruses

Virus that installs itself into memory and stays there after the host program terminates so it can infect other programs that come along and Boot sector infectors work this way.

3.3.3 Major Components of Viruses

Infection code: This is the part that locates an infectable object (previous snippet)

Payload: Any operation that any other program can do but is usually something meant to be irritating or possibly destructive.

Trigger: Whatever sets it off, time-of-day, program execution by user.

3.3.4 Classifications

- Boot Sector infectors
- File infectors
- Multipartite viruses
- Macro viruses
- Scripting viruses
- Other

➢ **Boot Sector infectors**

❖ Used to be really popular, but with less people using floppy disks are becoming rare

❖ Hard to write so other methods like scripting and macro virues are more popular

❖ First sector on hard drive partion (first sector on floppy) is Master Boot record, contains info about the drive and the bootstrap loader.

❖ If MBR can be messed up then when boot tries to get drive info from MBR for CMOS it won't be able to boot up.

❖ May keep a copy of MBR around in case other programs need to use info (makes it easier to disinfect)

➢ **File Infectors**

❖ File viruses infect executable files.

❖ Historically haven't been very successful at spreading.

❖ Fast infectors – try to infect as many other files as possible (instant gratification)

❖ Sparse infectors – only infect a few files at a time (in order to not be conspicuous)

❖ Most really successful file infectors are classified as Worms.

➢ **Multipartite Viruses**

❖ Viruses that use more than one infection mechanism

o File and Boot viruses

❖ Becoming more popular with virus writers

➢ **Macro Viruses**

❖ Infect programming environments rather than OSes or files.

❖ Almost any application that has it's own macro programming environment

– MS Office (Word, Excel, Access…)

– Visual Basic

❖ Application loads a file containing macro and executes the macro upon loading – or- runs it based on some application based trigger.

❖ Melissa was really successful macro virus

❖ Usually spread as an e-mail attachment

- **Script Viruses**
  - Usually refers to VBScript but could be any scripting environment as Unix scell scripts, Hypercard scripts, Javascript
  - Usually sent as e-mail attachments with doctored up file name as:
- Filename.doc.bat to fool user into opening it
- **Memetic Viruses**
  - These are not computer viruses but rather attempts at social engineering or getting the user to conform to a certain behavior.
  - Virus Hoaxes
  - "Good Times" hoax (mid 1990s) The story is that a virus called Good Times is being carried by email. Just reading a message with "Good Times" in the subject line will erase your hard drive, or even destroy your computer's processor. Needless to say, it's a hoax, but a lot of people believed it. The original message ended with instructions to "Forward this to all your friends," and many people did just that. Warnings about Good Times have been widely distributed on mailing lists, Usenet newsgroups, and message boards.
  - The original hoax started in early December, 1994. It sprang up again in March of 1995. In mid-April, a new version of the hoax that ment

3.3.5 WORMS

- Worms are a subset of viruses
- The differ in the the method of attachment; rather than attaching to a file like a virus a worm copies itself across the network without attachment.
- Infects the environment rather than specific objects
- Morris Worm, WANK, CHRISTMA EXEC

3.3.6 Christma Exec

- **Christmas Tree EXEC** was the first widely disruptive replicating network program, which paralysed several international computer networks in December 1987.
- Written by a student at the Clausthal University of Technology in the REXX scripting language, it drew a crude Christmas tree - then sent itself to each entry in the target's email contacts file. In this way it spread onto the European Academic Research Network (EARN), the BITNET, and IBM's world-wide VNET. On all of these systems it caused massive disruption.

- Its core mechanism was essentially the same as the worm of 2000 - although running on mainframes rather than PC's, spreading over a different network, and scripted using REXX rather than VBScript.

3.3.7 Morris Worm

- The **Morris worm** or **Internet worm** was one of the first computer worms distributed via the Internet; it is considered the first worm and was certainly the first to gain significant mainstream media attention. It also resulted in the first conviction under the 1986 Computer Fraud and Abuse Act. It was written by a student at Cornell University, Robert Tappan Morris, and launched on November 2, 1988 from MIT. The worm was released from MIT to disguise the fact that the worm originally came from Cornell. (Incidentally, Robert Tappan Morris is now an associate professor at MIT.)
- The Morris worm was not written to cause damage, but to gauge the size of the Internet. An unintended consequence of the code, however, caused it to be more damaging: a computer could be infected multiple times and each additional process would slow the machine down, eventually to the point of being unusable. The Morris worm worked by exploiting known vulnerabilities in UNIX sendmail, Finger, rsh/rexec and weak passwords. The main body of the worm could only infect DEC VAX machines running BSD 4, and Sun 3 systems. A portable C "grappling hook" component of the worm was used to pull over the main body, and the grappling hook could run on other systems, loading them down and making them peripheral victims.

3.3.8 Slapper Worm

- Linux - 2002
- Exploits a problem in OpenSSL to run a shell on a remote computer, this was done in certain versions of the Apache Webserver that use OpenSSL for for https.
- Also had code for DDOS
- Fixes have been issued but is still considered “in the wild”

3.3.9 Comparison between Virus and Worms

A **worm** is a special kind of **computer virus** that propagates by self-replication over a computer network. This propagation can be either via e-mail or other means such as files being copied over a network.

 ISBN: 978-93-93810-96-0

Table 3.1 Viruses Vs Worms

Comparison	Virus	Worm
Definition	The virus is the program code that attaches itself to application program and when application program run it runs along with it.	The worm is code that replicate itself in order to consume resources to bring it down.
How can it spread?	It has to rely on users transferring infected files/programs to other computer systems.	It can use a network to replicate itself to other computer systems without user intervention.
Does it infect files?	Yes, it deletes or modifies files. Sometimes a virus also changes the location of files.	Usually not. Worms usually only monopolize the CPU and memory.
Whose speed is more?	virus is slower than worm.	worm is faster than virus. E.g.The code red worm affected 3 lack PCs in just 14 Hrs.
How does it infect a computer system?	It inserts itself into a file or executable program.	It exploits a weakness in an application or operating system by replicating itself.

3.4 WEB SECURITY

3.4.1 Web Server

- Entry point for clients
- To a variety of services
- Customized for clients (e.g., via cookies)
- Supported by complex backend applications (e.g., databases)

Target of attackers

[1] Common protocol

[2] Supports a wide range of inputs

[3] Complex software interactions and Running with high privilege

Impact – Vulnerabilities, Threats, Risks. Note the multiple application layers and connection to legacy code.

3.4.2 Web Server Vulnerabilities

❖ Not surprisingly, these are numerous

❖ For IIS 5, focus was on function – All services were ON by default – Buffer overflow -- e.g., Code Red

❖ Interactions between components are complex – HTTP input to database queries – SQL Injection -- execute user input directly

❖ Web server permissions – Web servers have broad access – Deface web server -- modify server files – Compromise system -- modify system files

Some examples

"Disable all unnecessary ISAPI filters [services]"

• "Delete DLLs [libraries] associated with disabled filters"

✓ "Website must never be on the system drive"

✓ "Only necessary services" -- only SMTP –

✓ "Remove NTFS write permissions where possible"

✓ Obscurity

• "Don't use obvious names for script and code directories"

• "Set default website to extreme security

3.4.3 What can be done? What can be done?

Checklist for IIS 5 Gives an idea of what must be done for IIS

Some examples

➢ Disable all unnecessary ISAPI filters [services]

- Delete DLLs [libraries] associated with disabled filters
- Website must never be on the system drive
- Only necessary services -- only SMTP
- Remove NTFS write permissions where possible-Obscurity

➢ Don't use obvious names for script and code directories

➢ Set default website to extreme security

➢ IIS 7 does many of these -- automate all

Canonical (common) DOS - Request Flood

Attack: request flooding: Overwhelm some resource with legitimate requests e.g., web-server, phone system Unintentional flood is called a flash crowd.

 ISBN: 978-93-93810-96-0

DOS Prevention - Reverse-Turing Tests

Turing test: measures whether a human can tell the difference between a human or computer (AI)

Reverse Turning Tests: measures whether a user on the internet is a person, a bot, whatever? CAPTCHA - **c**ompletely automated public Turing test to tell computers and humans apart

Contorted image humans can read, computers can't image processing pressing SOA, making these harder.

Fig 3.4 CAPTCHA

DOS Prevention – Puzzles - Puzzles

Make the solver present evidence of "work" don If work is proven, then process request and only useful if request processing significantly more work than. Puzzle design 1) Must be hard to solve 2) Easy to Verify

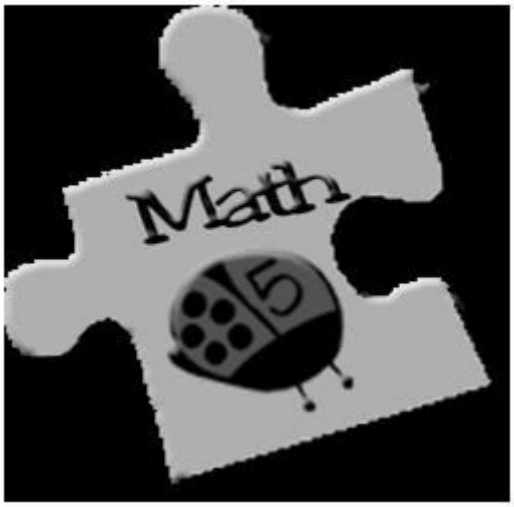


Fig 3.5 PUZZLES

Canonical Example

Puzzle: given x-bits of input r and h(r), where h is a cryptographic hash function.

 ISBN: 978-93-93810-96-0

CHAPTER 4

SOFTWARE SECURITY

The infrastructure for complex applications includes an operating system platform, such as Linux or Windows, other generic applications that run on that system, such as web browsers and e-mail clients, a database management system, middleware that supports distributed computing and database access and libraries of reusable components that are used by the application software. In fact, the majority of attacks focus on system infrastructures because the components (e.g. web browsers) are well known and widely available. In practice, there is an important distinction between application security and infrastructure security:

[1] Application security is a software engineering problem where software engineers should ensure that the system is designed to resist attacks.

[2] Infrastructure security is a systems management problem where system managers should ensure that the infrastructure is configured to resist attacks. System managers have to set up the infrastructure to make the most effective use of whatever infrastructure security features are available. They also have to repair infrastructure security vulnerabilities that have come to light as the software is used.

Security management is not a single task but includes a range of activities such as user and permission management, system software deployment and maintenance and attack monitoring, detection and recovery:

- ✓ User and permission management includes adding and removing users from the system, ensuring that appropriate user authentication mechanisms are in place and setting up the permissions in the system so that users only have access to the resources that they need.
- ✓ System software deployment and maintenance includes installing system software and middleware and configuring these properly so that security vulnerabilities are avoided. It also involves updating this software regularly with new versions or patches that repair security problems that have been discovered.
- ✓ Attack monitoring, detection and recovery includes activities which monitor the system for unauthorized access, detect and put in place strategies for resisting attacks and backup activities so that normal operation can be resumed after an external attack.

Securi1ty management is vitally important for the maintenance of secure systems. However, the key issues for software engineers are not the management processes but how to design support for security management into their application systems.

4.1 SECURITY CONCEPTS

A hospital information system maintains personal information on patients with mental health problems and their treatments. As mental health issues are particularly sensitive, it is essential that patient confidentiality be maintained. This system has to be accessible from different hospitals and clinics so it has been set up so that users access the system through a web browser. Hospital staff must log on to this system using a username and password. The system requires passwords to be at least eight letters long but allows any password to be set without further checking.

A criminal is told that a well-paid sports star is receiving treatment for mental health problems. He would like to gain illegal access to information in this system so that he can blackmail the star. By posing as a concerned relative and talking with the nurses in the mental health clinic, he discovers how to access the system. By checking name badges, he discovers the names of some of the people allowed access. He then attempts to log on to the system by using these names and systematically guessing possible passwords.

4.1.1 Terms and Description

Asset- A system resource that has a value and has to be protected.

Exposure - The possible loss or harm that could result from a successful attack.This can be loss or damage to data or can be a loss of time and Effort if recovery is necessary after a security breach.

Vulnerability - A weakness in a computer-based system that may be exploited to cause loss or harm.

Attack - An exploitation of a system's vulnerability. Generally, this is from outside the system and is a deliberate attempt to cause some damage.

Threats - Circumstances that have potential to cause loss or harm. You can think of these as a system vulnerability that is subjected to an attack.

Control- A protective measure that reduces a system's vulnerability. Encryption would be an example of a control that reduced a

Vulnerability of a weak access control system.

4.2 Example – Security Concepts

Asset - The records of each patient that is receiving or has received treatment

Exposure - Potential financial loss from future patients who do not seek treatment because they do not trust the clinic to maintain their data. Financial 1055 from legal action by the sports star. Loss of reputation.

Vulnerability - Aweak password system which makes it easy for users to set guessable passwords. User ids that are the same as names.

Attack - An impersonation of an authorised user.

Threat - An unauthorized user will gain access to the system by guessing the credentials login name and password) of an authorised user.

Control – A password checking system that disallows passwords that are set by users which are proper names or words that are normally included in a dictionary.

Security threats fall into three principal categories:

1. Threats to the confidentiality of the system and its data. These can disclose information to people or programs that are not authorised to have access to that information.
2. Threats to the integrity of the system and its data. These threats can damage or corrupt the software or its data.
3. Threats to the availability of the system and its data. These threats can restrict access to the software or its data for authorised users.

The controls that you might put in place to enhance system security also fall into three classes:

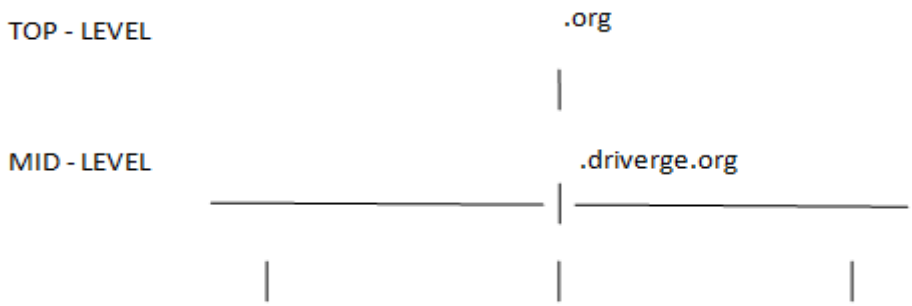
1. Controls that are intended to ensure that attacks are unsuccessful. The strategy here IS to design the system so that security problems are avoided. For example, sensitive military systems are not connected to public networks so that external access is impossible.
2. Controls that are intended to detect and repel attacks. These controls involve including functionality in a system that monitors its operation and checks for unusual patterns of activity. If these are detected, then action may be taken, such as shutting down parts of the system, restricting access to certain users, etc.
3. Controls that support recovery from problems. These can range from automated backup strategies and information 'mirroring' through to insurance policies that cover the costs associated with a successful attack on the system.

Domain Name System (DNS)

Definition: the DNS translate internet domain and host names to ip addresses.DNS automatically converts the names user type in our web browser addresses bar to the ip addresses of web servers hosting those sites .

The domain name system converts machine name to ip addresses. the mapping is done from name to address and address to name. The difference between just plain hosts ip mapping and domain mapping is that DNS uses a hierarchical naming standard. This hierarchy works from right to left with the highest level being on the right.

As an example, here is a simple domain



Bootom-Level strider.diverge.org samwise. diverge.org wormtongue.diverge.org

It seems too simple enough however the system can also be logically divided a even further if one wishes at different points. the example shown above shows three nodes the diverge.org domain but we good even divide diverge.org into sub domain such as strider.net1.diverge.org,samwise.net2.diverge.org and wormtongue.net2.diverge.org,in this cases, two nodes reside on net2.diverge.org and one on net1.diverge.org

4.3 Overview of the DNS

To connect to a system that supports IP, host initiate the connection must know in advance the IP address of the remote system. An IP address is a 32 bit number that represents the location of the system on a network. The 32 bit address is separated into4octets and each octets is typical represents by a decimal number the four decimal numbers are separated from each other by a dot character (“.”) even though four decimal number may be easier to remember then 32 ones and zeros, as with phone numbers, there is a practical; limit as to how many ip addresses a person can remember without the need for some sort of directory assistance. The directory essentially assigns hostname to ip addresses.

The Sanford research institutes network information center (SRI-NIC) become the responsible authority for maintaining unique host name for the internet. SRI-NIC maintained the single file called host.txt, and sites would continuously update SIR-Nic with a host name to IP address mapping to add to, delete from, or change in the file the problem was as the internet grew rapidly, so did the file causing it to become increasingly difficult to manage. Moreover, the host name is needed to be unique throughout the world wide internet. With the growing size of the internet it become more and more impractical grantee the uniqueness of a host name. The need for such things as a hierarchical naming structure and distributed management of host name paved the way for the creation of a new network in protocol that was flexible enough for a use on a global scale.

What evolved form this is an internet distributed database that maps the names of computer system to their respective numerical IP network addresses. This internet lookup facility is the DNS. Important to the concept of the distributed database is delegation of authority no longer is one single organizations responsible for hostname to IP address mappings, but rather those sites that are responsible for maintaining host name for their organizations can now regain that control.

4.3.1 Fundamentals of DNS

The DNS not only supports host name to network address resolution, known as forward resolution, but it also supports network address to host name resolution, known as inverse resolution. Due to its ability to map human memorable system name into computer networks numerical addresses, its distributed nature, and its robustness, the dns has evolved into a critical component of the internet. without it, the only way to reach other computer son the internet it to use the numerical network addresses .using ip addresses to connect t remote computer system is not a very user friendly representation of the system location on the internet and thus the DNS is heavily relied upon to retrieve an IP address by just referencing a computer systems fully qualified domain name (FQDN)

Fully qualified domain name (FQDN): A FQDN is basically a DNS host name and it represents where to resolve this host name within DNS hierarchy**.**

4.3 .2 The Domain name space

The DNS is a hierarchical tree structure whose root node is known as the root domain. A label in a DNS name directly corresponds with a node in the DNS tree structure. A label is a alphanumeric string that uniquely identifies that node from its neighbours. Labels are connected Together with a dot notation, ".", and a DNS containing multiple labels represents its path along the tree to the root. Labels are written from left to right. Only one zero length labels are allowed and are reserved

for the root of the tree. This is commonly referred to as the root zone. Due to the root label begin zero length, all FQDNs end in a dot.

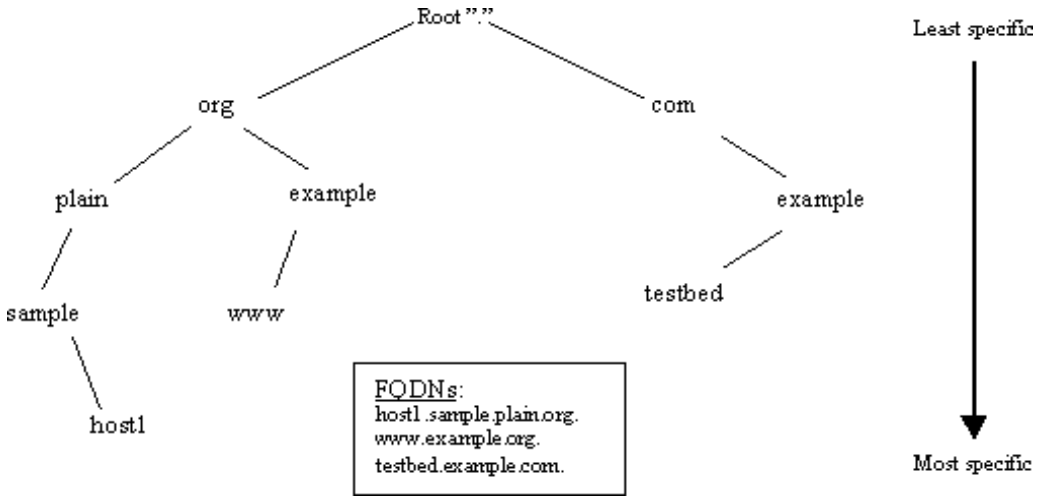


Fig 5.1 Domain Name Space example

As a tree is traversed in an ascending manner (i.e., from the leaf nodes to the root), the nodes become increasingly less specific (i.e., the leftmost label is most specific and the right most label is least specific). Typically in an FQDN, the left most labels is the host name, while the next label to the right is the local domain to which the host belongs. The local domain can be a sub domain of another domain. The name of the parent domain is then the next label to the right to the sub domain (i.e., local domain) name label, and so on, till the root of the tree is reached.

4.3.4 DNS Components

The DNS has three major components,

- **Database**
- **Server**
- **Client**

The database is a distributed database and is comprised of the Domain Name Space, which is essentially the DNS tree, and the Resource Records (RRs) that defines the domain names within the Domain Name Space. The server is commonly referred to as a name server. Name Server are typically responsible for managing some portion of the Domain Name Space and for assisting clients in finding information within the DNS tree. Name servers are authoritative for the domains in which they are responsible. They can also serve as a delegation point to identify other name servers that have authority over sub domains within a given domain.

The RR data found on the name server that makes up a domain is commonly referred to as zone information. Thus, name servers have zones of authority. A single zone can either be a forward zone (i.e., zone information that pertains to a given domain) or an inverse zone (i.e., zone information that maps IP addresses into DNS host names). DNS allows more than one name server per zone, but only one name server can be the primary server for the zone. Primary servers are where the actual changes to the data for a zone tale place. The other entire name server for a zone basically maintains copies of primary server's database for the zone. These servers are commonly referred to as secondary server.

A DNS RR has 6 fields: **NAME, TYPE, CLASS, TTL, RD Length, and RDATA** are presented in table. The NAME field holds the DNS name, also referred to as the owner name, to which the RR belongs. The TYPE field is the TYPE of RR. This field is necessary because it is not uncommon for a DNS name to have more than one type of RR.

The more common type of RR are:

Record	Type	Description Usage
A	An address record	Maps FQDN into an IP address
PTR	A pointer record	Maps an IP address into FQDN
NS	A names server	Denotes a name server for a zone
SOA	record	Specifies many attributes concerning the
CNAME	A Start of Authority	zone, such as the name of the domain
MX	record	(forward or inverse), administrative contact,
	A canonical name	the serial number of the zone, refresh
	record	interval, retry interval, etc.
	A Mail Exchanger	Defines an alias name and maps it to the
	record	absolute(canonically) name
		Used to redirect email for a given domain
		or host to another host

Common DNS Resource Records

- The CLASS in this case is "IN" which stands for Internet. Other classes exist but are omitted for brevity.
- The TTL is the time, in seconds, that a name server can cache a RR. A zero to live means that a server is not to cache the RR.
- RD Length is the RDATA field's length in octets. The RDATA field is the resource data field and is uniquely defines for each TYPE of RR, but in general it

can be thought of as the values into which the entity specified in the NAME field maps.

- The NAME field can be thought of as the subject of a query, although this is not always the case, and the answer Is the data contained in the RDATA field (even though the entire RR is returned in a DNS response).
- RRs are grouped into resources records sets (Resets). RRSets contain 0 or more RRs that have the same DNS name, class, and type, but the data (i,e., RDATA) is different. If the name, class, type, and data are the same for two or more records then duplicate records exist for the same DNS name. Name server should suppress duplicate records.

example.com.	IN	NS	ns1.example.com.
example.com.	IN	NS	ns2.example.com.
example.com.	IN	NS	ns.plain.org.

These three records are grouped into an RRSets.

The client component of the DNS typically contains software routines, known as functions, which are responsible for requesting information from the Domain Name Space on behalf of an application. These functions are bundled together into a software library that is commonly referred to as the resolver library. For this reason, clients are often called resolvers. The resolver library function are responsible for sending a query to a name server requesting information concerning a DNS name and returning the answer to the query back to the requestor.

DNS is hierarchical in structure. A domain is a sub tree of the domain name space. From the root, the assigned top-level domains are:

- **GOV-Government.**
- **EDU-Education.**
- **INT-International organization**
- **NET-Networks**
- **COM-Commercial entity.**
- **MIL-U.S. Military.**
- **ORG-Any other organization not previously listed.**

Each node on the domain name system is separate by a”.”.

Example:”mymachine.mycompany.com.”Note that any name ending in a”.” is an absolute domain name since it goes back to root.

4.3.5 DNS Transactions

DNS transactions occur continuously across the Internet. The two most common transactions are DNS zone transfer and DNS queries/responses. A DNS zone transfer occurs when the secondary server updates its copy of a zone for which it is authoritative. The secondary makes use of information it has on the zone, namely the serial number, and checks to see if the primary server has a more recent version. If it does, the secondary server retrieves a new copy of the zone.

A DNS query is answered by a DNS response. Resolvers use a finite list of name servers,

Usually not more than three, to determine where to send queries. If the first name server in the list is available to answer the query, then the others in the list are never consulted. If it is unavailable, each name server in the list is consulted until one is found that can return an answer to the query. The name server that receives a query from a client can act on behalf of the client to resolve the query. Then the name server can query other name server one at a time, with each server consulted begin presumable closer to the answer. The name server that has the answer sends a response back to the original name server, which then can cache the response and send the answer back to the client. Once an answer is cached, a DNS server can use the cached information when responding to subsequent queries for the same DNS information. Caching makes the DNS more efficient, especially when under heavy load. This efficiency gains its tradeoffs; the most notable is in security.

The DNS has a defined message protocol for queries and responses. A DNS message has five sections, a Header section, a Question section, an Answer section, an Authority section and an Additional section. The header section contains information such as the type of message and what other section are present in the message. The Question section contains the information concerning the object of the query. The last three sections are filled with RRs when appropriate. The Answer section contains RRs specifically pertaining to the answer. The Authority section is filled with either SOA or NS records belonging to the zone of authority for the owner name of the RR(s) in the Answer section. The Additional section may potentially have additional information that the receiver may find of profit.

HEADERQUESTIONANSWERAUTHRITYADDITIONAL

HEADER	QUESTION	ANSWER	AUTHORITY	ADDITIONAL

Fig 4.6 DNS Message Format

4.3.6 Threads to the Domain Name System

The original DNS specifications did not include security based on the fact that it contain, namely host names and IP addresses, is used as a means of communicating data [SPAF]. As more and more IP applications developed, the trend for using IP addresses and host names as a basis for allowing or disallowing access (i.e., system based authentication) grew. UNIX saw the advent of Berkeley "r" commands (e.g.,rlogin, rsh, etc.) and their dependencies on host names for authentication. Then many other protocols evolved with similar dependencies, such as Network File System (NFS), X windows, Hypertext Transfer Protocol (HTTP), et al.

Another contributing factor to the vulnerabilities in the DNS is that the DNS is designed to be a public database in which the concept of restricting access to information within the DNS name space is purposely not part of the protocol. Later versions of the BIND implementation allows access controls for such things as zone transfers, but all in all, the concept of restricting who can query the DNS for RRs is considered outside the scope of the protocol.

The existence and widespread use of such protocols as the r-commands put demands on the accuracy of information contained in the DNS. False information within the DNS can lead to unexpected and potentially dangerous exposures. The majority of the weaknesses within the DNS fall into one of the following categories: Cache poisoning, client flooding, dynamic update vulnerability, information leakage, and compromise of the DNS server's authoritative database.

ABOUT THE AUTHORS

Dr. S. Suganyadevi

Assistant Professor of Computer Science at Sri Krishna Arts and Science College in Coimbatore, Tamil Nadu, India. She has nearly 7 years of teaching experience and has published more than 12 papers in various reputed international journals. She has also presented 15 papers in international conferences, published a book, a chapter, and a pattern. She is an active researcher in the areas of image processing, Medical Image Processing, cyber security, and intelligent systems. She is also the chairperson of several national conferences.

Ms. M. Nivedha

Assistant Professor of Computer Science at Sri Krishna Arts and Science College in Coimbatore, Tamil Nadu, India. Formerly she worked as teacher in ARB International School, Coimbatore. She has 3 years of teaching experience. She has presented papers and also participated in various seminar, workshop and conference in various college. She is an active researcher in the areas of Cyber Security.

Ganga T G

Assistant Professor of Computer Applications at Nilgiri college of Arts and Science, Thaloor, Tamilnadu India followed by Research Scholar in Sri Krishna Arts and science College, Coimbatore, Tamil Nadu, India. Formerly she has worked in The Oxford College of Science, Bangalore. She has 2 years of teaching experience and 1 year of Industrial Experience. She has published more than 3 papers in various reputed international journals. She has presented papers and also participated in various seminar, workshop and conference in various college. She is an active researcher in the areas of Networking.

ABOUT THE BOOK

Cipher Chronicles: Unraveling Cyber Threats In The Modern Age is a comprehensive guide to the latest cyber threats and how to protect yourself from them. The book covers a wide range of topics, including:

- The history of cryptography and code breaking
- The different types of cyber threats, such as ransomware, phishing, and data breaches
- The latest trends in cybercrime
- How to protect your personal and business data
- How to create strong passwords and keep them safe
- How to use security software effectively
- How to spot and report suspicious activity

Cipher Chronicles is a must-read for anyone who wants to stay safe in the digital age. It is written in an engaging and easy-to-understand style, and it provides practical advice that you can use to protect yourself from cyber threats. The book is packed with real-world examples of cyber attacks, which will help you to understand the risks and how to mitigate them.

EMPYREAL PUBLISHING HOUSE

ISBN 978-93-93810-96-0

9 789393 810960

India | UAE | Nigeria | Uzbekistan | Montenegro | Iraq | Egypt | Thailand | Uganda | Philippines | Indonesia

Empyreal Publishing House || www.empyrealpublishinghouse.com || info@empyrealpublishinghouse.com