Information Security

Computer security encompasses measures and controls that safeguard the confidentiality, integrity, and availability of assets within an information system. Here are the core concepts broken down:

Core Concepts:

1. Confidentiality

- **Data Confidentiality**
  - Ensures that private or confidential information is not disclosed to unauthorized individuals.
- **Privacy**
  - Ensures individuals have control or influence over the collection, storage, and disclosure of their personal information.

2. Integrity

- **Data Integrity**
  - Ensures that information and programs are altered only in a specified and authorized manner.
- **System Integrity**
  - Ensures that systems function as intended without unauthorized manipulations.

3. Availability

- Ensures systems operate promptly and services are not denied to authorized users.

Losses Defined:

- **Loss of Confidentiality**: Unauthorized information disclosure.
- **Loss of Integrity**: Unauthorized modification or destruction of information.

- **Loss of Availability**: Disruption in access to or use of information or an information system.

Expanded Concepts:

While the CIA (Confidentiality, Integrity, Availability) triad is a well-established framework for defining security objectives, it is argued that a more complete picture can be achieved with the addition of two more concepts:

4. Authenticity

- Ensures genuineness and verifiability of a transmission or message.
- Verifies the identity of users and the trustworthiness of input sources.

5. Accountability

- Ensures actions within the system can be uniquely traced back to an entity.
- Supports various security goals such as non-repudiation, deterrence, fault isolation, and intrusion detection/prevention.
- Helps in forensic analysis post-security breaches or transaction disputes.

Importance

Confidentiality

- **High Importance**: Student grade information
  - Regulated by FERPA in the US.
  - Should be available only to students, their parents, and certain employees.
- **Moderate Importance**: Student enrollment information
  - Still under FERPA but accessed more regularly.
  - Less targeted compared to grade info.
  - Less consequential when disclosed.
- **Low/No Importance**: Directory information

    - Lists of students, faculty, or departments.
    - Typically public and found on a school’s website.

Integrity

- **High Importance**: Hospital patient’s allergy information in a database
    - Vital for a doctor to trust this information.
    - Unauthorized changes can lead to patient harm and hospital liability.
- **Moderate Importance**: Website forums for registered users
    - Possible for users or hackers to alter or deface content.
    - Damage isn't severe unless it's a vital research forum.
- **Low Importance**: Anonymous online polls
    - Offered by many websites, like news organizations.
    - Inaccuracy is expected due to lack of safeguards.

Availability

- **High Importance**: Authentication service systems
    - Critical for systems, applications, and devices.
    - Interruption means a loss in resource accessibility, productivity, and potential customers.
- **Moderate Importance**: Public university website
    - Provides info for students and donors.
    - Not critical, but unavailability can be embarrassing.
- **Low Importance**: Online telephone directory
    - Temporary unavailability is a minor inconvenience.
    - Alternative access methods available (e.g., hardcopy directory).

Key Concepts

Adversary (Threat Agent)

- A person, group, organization, or government entity that intends to execute detrimental activities, potentially causing harm to an information system or the data housed within it, utilizing various methods and strategies to accomplish their adverse objectives.

Attack

- Malicious activities or maneuvers that are specifically designed to collect, disrupt, deny, degrade, or destroy information system resources or the information itself, utilizing various techniques ranging from cyber-attacks to physical tampering.

Countermeasure

- Tools, devices, or methodologies implemented to prevent, mitigate or obstruct harmful actions, including espionage, sabotage, theft, or unauthorized access to information systems. These can also function to prevent unauthorized use of sensitive information or system resources, thereby safeguarding organizational assets.

Risk

- A detailed assessment of the potential adverse impacts of certain events or circumstances on an entity or organization. It considers the adverse consequences that could occur if the circumstance or event materializes and the probability of its occurrence. The objective is to create a strategic approach to minimize negative impacts and protect organizational interests.

Security Policy

- A comprehensive set of guidelines and criteria aimed at ensuring the security of system services and data. It dictates and limits the activities within a data processing facility, fostering a secure environment for the operation of systems and the safekeeping of data, thereby minimizing the risk of security breaches.

System Resource (Asset)

- These are significant components such as major applications, general support systems, high-impact programs, physical infrastructure, mission-critical systems,

personnel, equipment, or logically interconnected group of systems that play a crucial role in the functioning and success of an organization.

Threat

- Any potential event or situation that holds the capacity to negatively affect an organization, individuals, other organizations, or even a nation through an information system. This could involve unauthorized access, destruction, disclosure, modification of information, or denial of service, all of which could potentially tarnish the organization's reputation or hamper its operations.

Vulnerability

- A recognized weakness or flaw present in an information system, system security procedures, internal controls, or implementation that could potentially be exploited or triggered by a threat source, thereby compromising the security and functionality of the system.

Types of vulnerability

1. **Corruption**: The system may experience instances where it provides incorrect results or behaves in an erratic manner due to unauthorized or malicious modifications. This scenario emphasizes the necessity to establish robust security measures to protect data integrity.
2. **Leakage**: The system might exhibit gaps that allow unauthorized individuals to gain access to sensitive or confidential information available through the network, pointing towards potential weaknesses in the security protocols implemented.
3. **Unavailability or Slow Performance**: Users might find it either extremely difficult or downright impossible to use the system or network owing to reduced operational speed or complete unavailability, highlighting the importance of ensuring system availability and efficiency as a part of comprehensive security planning.

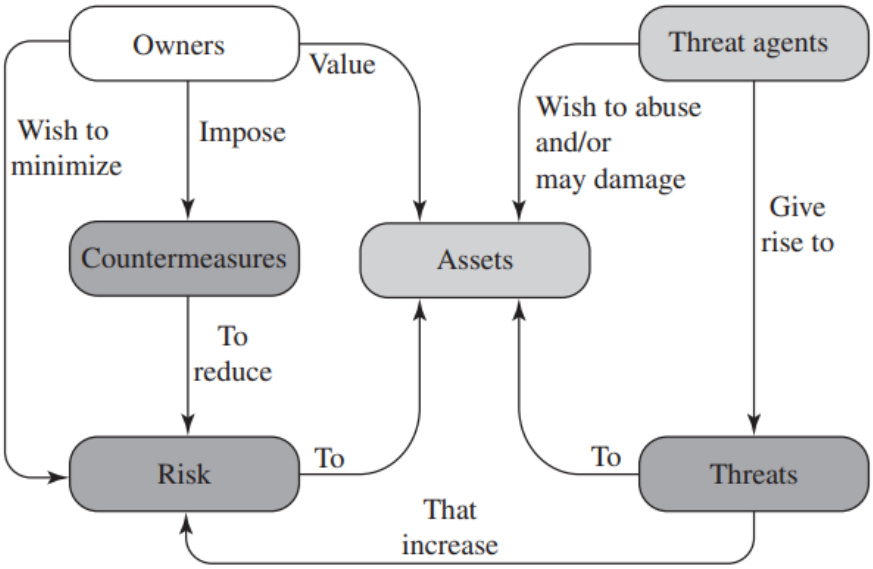


Figure 1.2 **Security Concepts and Relationships**

Definitions

- **Threat**: A potential danger or harm to a system resource, usually exploiting vulnerabilities present in the system.
- **Attack**: The execution of a threat, which, if successful, results in a breach of security, causing undesirable consequences.
- **Attacker or Threat Agent**: The entity responsible for initiating an attack, aiming to compromise the security of a system.
- **Countermeasure**: Strategies or techniques implemented to mitigate the effects of a security attack, preventing future breaches and aiding in recovery post-attack.

Classification of Attacks

Based on the Nature of the Attack

1. **Active Attack**
    - **Objective**: To change system resources or influence their operation.
    - **Impact**: Alters data or system functionalities directly.
2. **Passive Attack**

- **Objective**: To gather or utilize information from the system without affecting its resources.
- **Impact**: Does not directly alter system resources but may breach confidentiality.

Based on the Origin of the Attack

1. **Inside Attack**
   - **Initiator**: An insider, an entity within the security perimeter having authorized access.
   - **Misuse**: Utilizes access rights in a manner not sanctioned by the authorizing entity.
   - **Examples**: Employees misusing their access rights, unauthorized data manipulation.
2. **Outside Attack**
   - **Initiator**: An outsider, an unauthorized entity outside the security perimeter.
   - **Range**: Could range from individuals playing pranks to organized criminal groups, or even hostile governments targeting critical infrastructures.
   - **Examples**: Cyber criminals, international terrorists.

Countermeasures and Considerations

- **Preventive Countermeasures**: Aim to entirely prevent an attack from occurring.
- **Detective and Recovery Countermeasures**: Focus on identifying an attack and recovering from its implications, in case prevention fails.
- **New Vulnerabilities**: Countermeasures might introduce new vulnerabilities that need to be addressed.
- **Residual Vulnerabilities**: These are vulnerabilities that remain even after implementing countermeasures, posing a residual level of risk.
- **Risk Minimization**: Asset owners strive to reduce these risks while also considering other constraints like resources and feasibility.

The primary objective is to devise countermeasures that not only prevent attacks but also aid in the quick recovery and minimize potential residual risks, thereby safeguarding the system resources from both internal and external threats.

Threat Consequences and Corresponding Attacks

1. **Unauthorized Disclosure** (Threat to confidentiality)
    - **Exposure**: Deliberate or accidental release of sensitive data to unauthorized entities, which can stem from various errors or insider actions.
        - Example: Universities accidentally posting confidential student information on the Web.
    - **Interception**: Unauthorized entities accessing sensitive data during transfers between authorized sources and destinations.
        - Example: A hacker accessing e-mail traffic or other data transfers on the Internet.
    - **Inference**: Indirect access to sensitive data through the analysis of data traffic patterns or through exploiting limited access to databases.
        - Example: Gaining information through traffic analysis on a network.
    - **Intrusion**: Gaining unauthorized access to sensitive data by bypassing the system's security measures.
        - Example: Overcoming system's access control protections to access sensitive data.
2. **Deception** (Threat to system/data integrity)
    - **Masquerade**: Unauthorized access by posing as an authorized user or using malicious logic to gain unauthorized access.
        - Example: Using someone's login ID and password to access the system or using a Trojan horse.
    - **Falsification**: Altering, replacing, or introducing false data into a system.
        - Example: A student altering grades in a school database.

- **Repudiation**: Denying the sending, receiving, or possessing of data falsely.
    - Example: A user denying the reception or possession of certain data.

3. **Disruption** (Threat to availability/system integrity)
    - **Incapacitation**: Interrupting or preventing the correct operation of system services, either through physical damage or malicious software.
        - Example: Using viruses or worms to disable a system.
    - **Corruption**: Adversely altering system operation or functions, which can be achieved through unauthorized access or malicious software.
        - Example: Introducing backdoor logic in the system for unauthorized access.
    - **Obstruction**: Interfering with system operation by disrupting communications or overloading system resources.
        - Example: Altering communication control information or causing excess traffic burden.
4. **Usurpation** (Threat to system integrity)
    - **Misappropriation**: Unauthorized use of system resources, often for malicious purposes.
        - Example: Utilizing malicious software in distributed denial-of-service (DDoS) attacks.
    - **Misuse**: Unauthorized access and utilization of system components to compromise security.
        - Example: A hacker using malicious logic for unauthorized access and potential damage.

Table 1.3 **Computer and Network Assets, with Examples of Threats**

	Availability	Confidentiality	Integrity
Hardware	Equipment is stolen or disabled, thus denying service.	An unencrypted USB drive is stolen.	
Software	Programs are deleted, denying access to users.	An unauthorized copy of software is made.	A working program is modified, either to cause it to fail during execution or to cause it to do some unintended task.
Data	Files are deleted, denying access to users.	An unauthorized read of data is performed. An analysis of statistical data reveals underlying data.	Existing files are modified or new files are fabricated.
Communication Lines and Networks	Messages are destroyed or deleted. Communication lines or networks are rendered unavailable.	Messages are read. The traffic pattern of messages is observed.	Messages are modified, delayed, reordered, or duplicated. False messages are fabricated.

Passive Attacks

- **Definition**: Attempts to learn or use information from the system without affecting system resources.
- **Nature**: Eavesdropping or monitoring transmissions with the intention to obtain transmitted information.
- **Types**:
  1. **Release of Message Contents**:
     - Disclosure of sensitive or confidential information in phone conversations, emails, file transfers, etc.
     - Goal: Prevent opponents from understanding the contents of these transmissions.
  2. **Traffic Analysis**:
     - More subtle compared to message content release.
     - Even with encryption protection, opponents might deduce information from the message patterns, including the identity of communicating hosts, and the frequency and length of messages.
     - This deduced information can hint at the nature of the communication taking place.

- **Prevention Strategy**: Mainly focuses on prevention through encryption rather than detection as these attacks do not involve data alteration, making them hard to detect.

Active Attacks

- **Definition**: Attempts to alter system resources or affect their operation.
- **Characteristics**: Contrary to passive attacks, active attacks are challenging to prevent entirely but are easier to detect. The goal is to detect and recover from disruptions or delays caused by them.
- **Types**:
  1. **Replay**:
     - Captures a data unit passively and retransmits it to cause an unauthorized effect.
  2. **Masquerade**:
     - An entity impersonates another entity, often involving other forms of active attacks.
     - Example: Capturing and replaying authentication sequences to gain unauthorized privileges.
  3. **Modification of Messages**:
     - Altering, delaying, or reordering portions of a legitimate message to cause an unauthorized effect.
     - Example: Modifying a message to grant unauthorized access to confidential files.
  4. **Denial of Service (DoS)**:
     - Inhibits normal use or management of communication facilities, either targeting a specific service or disrupting an entire network.
     - Methods: Disabling the network or overloading it with messages to degrade performance.
- **Mitigation Strategy**:

- Detection of these attacks has a deterrent effect, contributing to prevention.
- Recovery from disruptions or delays caused by the attacks is crucial.

Notes on Security Design Principles and Implementation

Introduction

- Challenges in achieving foolproof security design and implementation.
- Importance of having guiding principles for developing protection mechanisms.
- The role of the National Centers of Academic Excellence in Information Assurance/Cyber Defense.

Fundamental Security Design Principles

Reference: [NCAE13], [SALT75]

1. **Economy of Mechanism**
   - Simple and small security designs are preferable.
   - Simple mechanisms: less exploitable flaws, easier updates, and less maintenance.
   - Goal: Reduce unnecessary complexity despite demand for new features.
2. **Fail-Safe Defaults**
   - Base access decisions on permission, not exclusion.
   - Default: No access, defining conditions under which access is permitted.
   - A safer approach to detect failures quickly.
3. **Complete Mediation**
   - Every access checked against the access control mechanism.
   - Avoid reliance on cached access decisions.
   - Consideration needed for changes in authority propagation in continuous systems.
4. **Open Design**

- Security mechanisms should be open to public scrutiny, except for encryption keys.
- Promotion of confidence through expert reviews and widespread adoption.

5. **Separation of Privilege**
    - Use multiple privilege attributes for restricted resource access.
    - Multifactor authentication as an example.
    - Mitigation of potential damages through privilege limitation.
6. **Least Privilege**
    - Users/processes should operate with the minimum necessary privileges.
    - Role-based access control as an example.
    - Temporal aspect: Special privileges should be temporary, preventing unnecessary exposure.
7. **Least Common Mechanism**
    - Minimize shared functions among users to enhance security.
    - Reduction of unintended communication paths and dependency on hardware/software.
8. **Psychological Acceptability**
    - Security mechanisms should not unduly interfere with user work.
    - Need for transparency or minimal obstruction.
    - Encourage user comprehension and adherence to security protocols.
9. **Isolation**
    - Critical resource protection through physical or logical isolation.
    - Prevention of unauthorized access through modern operating systems.
    - Security mechanism isolation to prevent tampering or unauthorized access.
10. **Encapsulation**
    - Object-oriented isolation for protection.

- Access restrictions to internal structures of data objects.

11. **Modularity**
    - Development of security functions as separate modules.
    - Support for technology migration or feature upgrades without complete system redesign.
12. **Layering**
    - Use of overlapping protection approaches.
    - Defense in depth: Multiple barriers against adversaries.
13. **Least Astonishment**
    - Predictable and intuitive responses from programs or user interfaces.
    - Transparency in authorization mechanisms to aid user understanding.

Other Important Points

- **Personnel Security**
  - Trustworthiness and security criteria adherence for personnel in responsible positions.
  - Protection of information during and after personnel actions (e.g., terminations, transfers).
  - Sanctions for non-compliance with security policies.
- **Risk Assessment**
  - Periodic assessment of risks to operations, assets, and individuals.
  - Consideration of the implications of organizational information systems usage.
- **Systems and Services Acquisition**
  - Adequate resource allocation for organizational information system protection.
  - Incorporation of information security considerations in development life cycle processes.
  - Restrictions on software usage and installation, and third-party provider security measures.

- **System and Communications Protection**
  - Monitoring and controlling organizational communications at key boundaries.
  - Use of designs and techniques promoting effective information security.
- **System and Information Integrity**
  - Timely identification and correction of information system flaws.
  - Protection against malicious code and appropriate responses to security alerts.

Computer Security Strategy

Introduction

- **Goal**: Develop a comprehensive strategy for computer security involving three vital aspects:
  1. Specification/Policy
  2. Implementation/Mechanisms
  3. Correctness/Assurance

1. Security Policy

- **Definition**: A statement outlining rules and practices for safeguarding sensitive and critical system resources.
- **Types of Policies**:
  - Informal: Describes the desired system behavior.
  - Formal: Specifies how the system or organization will provide security services.
- **Development Considerations**:
  - Value of assets to be protected
  - System vulnerabilities
  - Potential threats and likelihood of attacks
- **Trade-offs to Consider**:

    - Ease of use vs. security (considering access control mechanisms, network security measures, etc.)
    - Cost of security vs. cost of failure and recovery (direct costs, asset value, damages, risk, etc.)
- **Nature of Policy**: A business decision possibly influenced by legal requirements.

2. Security Implementation

- Encompasses four key courses of action:
    1. **Prevention**: Ideal scenario where no attack is successful; involves secure encryption algorithms and protective measures.
    2. **Detection**: Identifying security attacks when absolute protection is not feasible, through systems like intrusion detection.
    3. **Response**: Initiating actions to halt detected ongoing attacks to prevent further damage.
    4. **Recovery**: Using backup systems to reload correct data copy if data integrity is compromised.

3. Assurance and Evaluation

- **Assurance**:
    - Definition: An attribute providing grounds to believe that the system’s security policy is enforced.
    - Concerns: Whether the security system meets its requirements and specifications.
    - Current State: Not absolute; depends on developed formal models and logical and mathematical techniques but remains a matter of degree.
- **Evaluation**:
    - Definition: Process of examining a computer product or system based on certain criteria.
    - Methods: Involves testing, formal analytic, or mathematical techniques.

- Objective: Develop evaluation criteria applicable to any security system for product comparisons.

Standards

1. **Introduction to Standards**
   - Standards are specified for security techniques, applications, management practices, and overall architecture of security mechanisms and services.
   - Significant standards for various aspects of computer security are either in use or under development.
   - Several global organizations are spearheading the formulation and promotion of these standards.
2. **Key Organizations Involved in Developing Security Standards**
   - **National Institute of Standards and Technology (NIST)**
     - A U.S. federal agency focused on measurement science, standards, and technology.
     - Primarily caters to U.S. government and promotes private sector innovation in the U.S.
     - Has a global impact with Federal Information Processing Standards (FIPS) and Special Publications (SP).
   - **Internet Society (ISOC)**
     - A worldwide professional membership society.
     - Addresses issues concerning the future of the Internet.
     - Serves as the home organization for groups responsible for Internet infrastructure standards, including the Internet Engineering Task Force (IETF) and the Internet Architecture Board (IAB).
     - Develops Internet standards and related specifications, published as Requests for Comments (RFCs).
   - **International Telecommunication Union (ITU) - ITU-T Sector**

    - A UN agency where governments and the private sector collaborate on global telecom networks and services.
    - Focuses on producing standards covering all fields of telecommunications.
    - Standards are published as "Recommendations".
- **International Organization for Standardization (ISO)**
    - A federation of national standards bodies from over 140 countries.
    - A non-governmental entity promoting standardization to facilitate international exchange of goods and services, and to foster cooperation in various spheres such as intellectual, scientific, technological, and economic activities.
    - Develops international agreements published as International Standards.

3. **Additional Information**
    - Detailed discussions on these organizations can be found in Appendix C.
    - A list of referenced ISO and NIST documents are provided at the end of the book.

Cryptography

Introduction

1. **Symmetric Encryption**:
    - Also known as conventional or single-key encryption.
    - Predates public-key encryption which was introduced in late 1970s.
    - Used extensively for secret communication from ancient times to present day across various fields including military, diplomacy, and commerce.
    - Remains more widely used compared to other types of encryption.

Key Concepts

1. **Ingredients of Symmetric Encryption**:

- **Plaintext**: Original message/data which serves as the input to the encryption algorithm.
- **Encryption Algorithm**: Performs various substitutions and transformations on the plaintext.
- **Secret Key**: Fed into the encryption algorithm to dictate the exact transformations performed.
- **Ciphertext**: The scrambled output message that depends on the plaintext and the secret key.
- **Decryption Algorithm**: Runs in reverse of encryption algorithm to convert ciphertext back to original plaintext using the secret key.

2. **Requirements for Secure Use**:
    - A strong encryption algorithm resistant to decipher attempts even with access to ciphertext and the associated plaintext.
    - Secure acquisition and storage of the secret key to prevent unauthorized access and maintain the confidentiality of communications.
3. **Attack Approaches**:
    - **Cryptanalysis**: Involves exploiting algorithm characteristics and potential knowledge of plaintext or plaintext-ciphertext pairs to deduce the key or specific plaintext.
    - **Brute-force Attack**: Trying all possible keys until an intelligible plaintext is found. Needs knowledge of expected plaintext and means to distinguish plaintext from garble.

Symmetric Encryption Algorithms

1. **Block Encryption Algorithms**:
    - Processes plaintext in fixed-size blocks, each producing a block of ciphertext of equal size.
    - Most notable algorithms include DES, Triple DES, and AES.
    - Detailed technical aspects covered in Chapter 20.
2. **Data Encryption Standard (DES)**:

- Adopted by National Bureau of Standards (now NIST) in 1977.
- Processes 64-bit plaintext blocks with 56-bit keys to produce 64-bit ciphertext blocks.
- Concerns include potential cryptanalysis exploits and inadequacy of 56-bit key length given modern processing speeds.
- Despite numerous attempts, no fatal weaknesses reported till now.

Drawbacks of DES (Data Encryption Standard):

1. **Limited Key Size:** DES uses a 56-bit key, which was found to be susceptible to brute-force attacks, especially with the advent of powerful modern computers.
2. **Vulnerability to Certain Attacks:** Over time, several vulnerabilities were identified in the DES algorithm, including susceptibility to linear and differential cryptanalysis.
3. **Slow Performance:** DES can be slow, especially for applications requiring the encryption of large amounts of data.

3. **Triple DES (3DES)**:
   - Extends DES by repeating the DES algorithm thrice with two or three unique keys (112 or 168 bits).
   - Adopted in 1985 for financial applications and incorporated in the Data Encryption Standard in 1999.
   - Overcomes DES vulnerability to brute-force attacks and maintains high resistance to cryptanalysis.
   - Drawbacks include slow performance in software and smaller block size (64-bit) compared to modern standards.

Drawbacks of 3DES (Triple DES):

1. **Processing Overhead:** 3DES, which applies the DES algorithm three times in a series to each data block, inherently takes three times as long as DES for processing, making it considerably slow, especially for applications with large data throughput requirements.

2. **Legacy Issues:** Despite being more secure than DES, 3DES still carries some of the vulnerabilities of the original DES algorithm, though they are much more difficult to exploit due to the triple encryption process.
3. **Key Management:** While 3DES increases the key size to an effective 112 or 168 bits, it requires the management of two or three separate 56-bit keys, adding complexity to key management procedur

4. **Advanced Encryption Standard (AES)**:
    - Developed to replace 3DES due to its drawbacks.
    - NIST initiated in 1997, with requirements for better security strength and efficiency than 3DES.
    - Symmetric block cipher with 128-bit block length and key lengths of 128, 192, and 256 bits.
    - Rijndael algorithm was selected for AES, detailed in Chapter 20.
5. **Practical Security Issues**:
    - Symmetric encryption often applied to data units larger than single blocks.
    - Techniques like electronic codebook (ECB) mode are used to encrypt multiple-block messages.

Stream Ciphers and Block Ciphers

Block Cipher

- **Process**: Encrypts data one block at a time, where each block consists of a fixed number of bits (for example, 64 or 128 bits).
- **Commonality**: These are more common compared to stream ciphers.
- **Reusability of Keys**: One advantage mentioned here is that you can reuse keys.
- **Applications**: Preferred for encrypting data in blocks, like files, emails, or database entries.

Stream Cipher

- **Process**: Encrypts data continuously, one element (like a byte or bit) at a time.

- **Design Flexibility**: It can operate at various unit sizes, from one bit to larger units than a byte.
- **Speed and Code Efficiency**: Typically faster and uses less code compared to block ciphers.
- **Applications**: Suitable for streaming data encryption, like data communication channels or browser/Web links.

Stream Cipher Structure

The structure of a stream cipher, as depicted in the passage, consists of the following components:

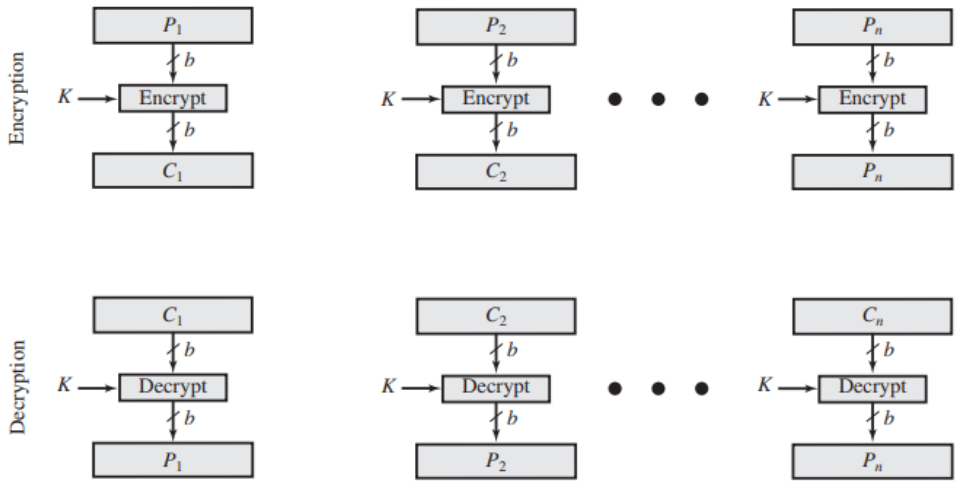
1. **Key Input**: A secret key is used to initialize the cipher. This key is known to both the sender and the receiver.
2. **Pseudorandom Byte Generator (Keystream Generator)**:
    - **Role**: Produces a stream of pseudorandom bytes based on the input key.
    - **Characteristics of Output**: Appears random and is unpredictable without knowledge of the input key.
    - **Output**: A stream of bytes called the keystream, which will be used in the encryption and decryption process.
3. **Encryption Process**:
    - **Plaintext**: The original data that needs to be encrypted.
    - **Operation**: The plaintext is combined one byte at a time with the keystream using a bitwise XOR operation to produce the ciphertext.
4. **Decryption Process**:
    - **Ciphertext**: The encrypted data.
    - **Operation**: Similar to the encryption process, but here the ciphertext is combined with the same keystream to retrieve the original plaintext.

Security Aspect

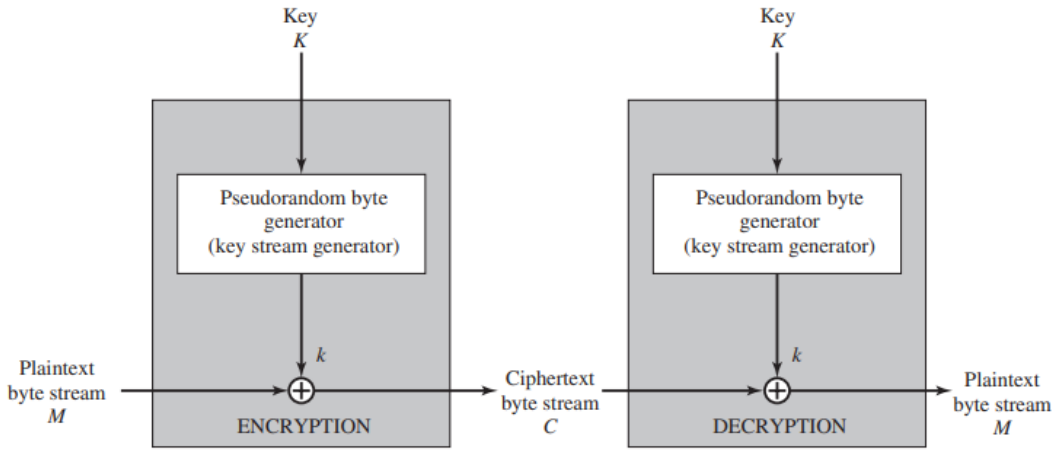
- With a well-designed pseudorandom number generator, a stream cipher can offer a level of security comparable to a block cipher of similar key length.

Application Suitability

- Depending on the specific requirements of an application, either stream ciphers or block ciphers may be more appropriate. Stream ciphers are better suited for streaming data, while block ciphers are more suited for block data encryption.



(a) Block cipher encryption (electronic codebook mode)



(b) Stream encryption

DES

DES is a symmetric-key algorithm for the encryption of electronic data. It works on a block cipher principle, where it encrypts data in fixed-size blocks, which is 64 bits in this case.

Now let's dive into the details:

Step 1: Initial Permutation

1. **Input**: 64-bit plaintext.
2. **Operation**: The bits are rearranged according to a predefined initial permutation table.
3. **Output**: 64-bit block (rearranged).

Step 2: Splitting the Block

1. **Input**: 64-bit block from step 1.
2. **Operation**: The block is split into two halves: a left half (L0) and a right half (R0), each being 32 bits.
3. **Output**: Two 32-bit blocks (L0 and R0).

Step 3: Key Generation

Main Key (64 bits):

1. **56 bits** are used for encryption (8 bits are parity bits and not used in encryption).
2. This 56-bit key is then subjected to permutations and split into two halves (C0 and D0), each of 28 bits.

Round Key Generation:

1. For each round, the halves (C0 and D0) undergo a left circular shift by a predefined number of bits.
2. A 48-bit subkey is generated for each round by selecting bits according to a predefined permutation table (PC-2).
3. This process is repeated to generate 16 round keys, each of 48 bits.

Step 4: Rounds of Encryption (16 rounds)

In each round, the following operations occur:

1. **Expansion Permutation**: The 32-bit right half from the previous round (or initial split) is expanded to 48 bits using an expansion permutation table.
2. **Subkey XOR**: This 48-bit block is XORed with the 48-bit round key generated for this round.
3. **Substitution (S-boxes)**: The resultant 48-bit block is divided into 8 blocks of 6 bits each. Each 6-bit block is then fed into one of the eight S-boxes, which maps it to a 4-bit output (this introduces non-linearity in the encryption process).
4. **Permutation (P-box)**: The 32 bits obtained from all S-boxes are combined and then permuted according to a predefined P-box table, which helps in diffusing the bits further.
5. **XOR with Left Half & Swap**: The output of the P-box is XORed with the left half from the previous round. This becomes the right half for the next round. The previous right half becomes the left half for the next round (a swap).
6. This process is repeated for 16 rounds.

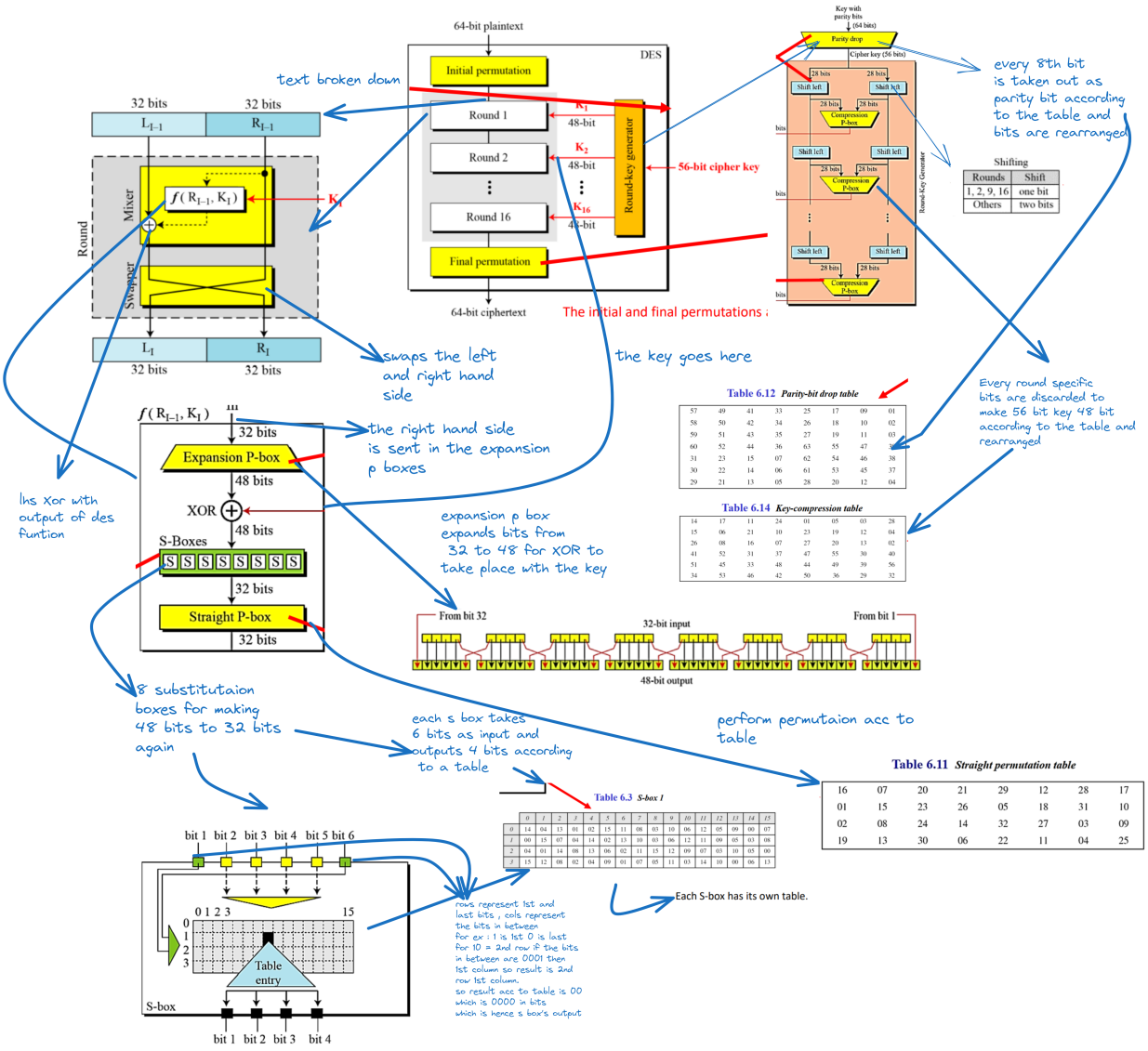
Step 5: Final Permutation

1. **Input**: The combined 64-bit block from the last round (after swapping back the last left and right halves).
2. **Operation**: A final permutation, which is the reverse of the initial permutation, is applied.
3. **Output**: 64-bit ciphertext block.

Summary of Component Details:

1. **S-boxes (Substitution boxes)**: These are tables that map a 6-bit input to a 4-bit output. It introduces non-linearity into the encryption process, making it more resistant to linear cryptanalysis.
2. **P-boxes (Permutation boxes)**: These are used to rearrange the bits, contributing to the diffusion property of the cipher, making it more secure against statistical attacks.
3. **XOR with Subkey**: Introduces the key into the encryption process, ensuring that the encryption is key-dependent.

4. **Round Keys (48 bits)**: Generated through a series of permutations and shifts, these keys are used in each round to provide security, ensuring that reversing the encryption (decryption) is computationally intensive without the key.



AES

The Advanced Encryption Standard (AES) is a symmetric encryption algorithm established by the U.S. National Institute of Standards and Technology (NIST) in 2001. It has become one of the most popular encryption standards used globally. AES operates on blocks of data and supports key sizes of 128, 192, and 256 bits. Here, we'll break down the AES encryption process step by step:

1. Key Expansion

Before the encryption process begins, the key undergoes an expansion phase where additional keys are derived for use in various rounds of the encryption process.

2. Initial Round

2.1 AddRoundKey

The plain text block (128 bits) is XORed with the initial round key (derived from the key expansion process).

3. Main Rounds

The number of rounds depends on the key size: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Each round consists of the following steps:

3.1 SubBytes

Each byte in the block is replaced with another byte according to a predefined substitution table known as the S-box.

3.2 ShiftRows

The rows in the block are shifted by a certain number of steps. The first row is not shifted, the second row is shifted by one step, the third row by two steps, and the fourth row by three steps.

3.3 MixColumns

This step involves mixing the columns of the block using a fixed function; this helps in adding diffusion to the encryption process. This step is not performed in the final round of encryption.

3.4 AddRoundKey

In this step, the block is XORed with the round key derived for that specific round from the key expansion process.

4. Final Round

The final round is similar to the main rounds but excludes the MixColumns step.

4.1 SubBytes

As in the main rounds, the SubBytes step is performed using the S-box.

4.2 ShiftRows

The ShiftRows step is executed just like in the main rounds.

4.3 AddRoundKey

The block is XORed with the final round key derived from the key expansion process.

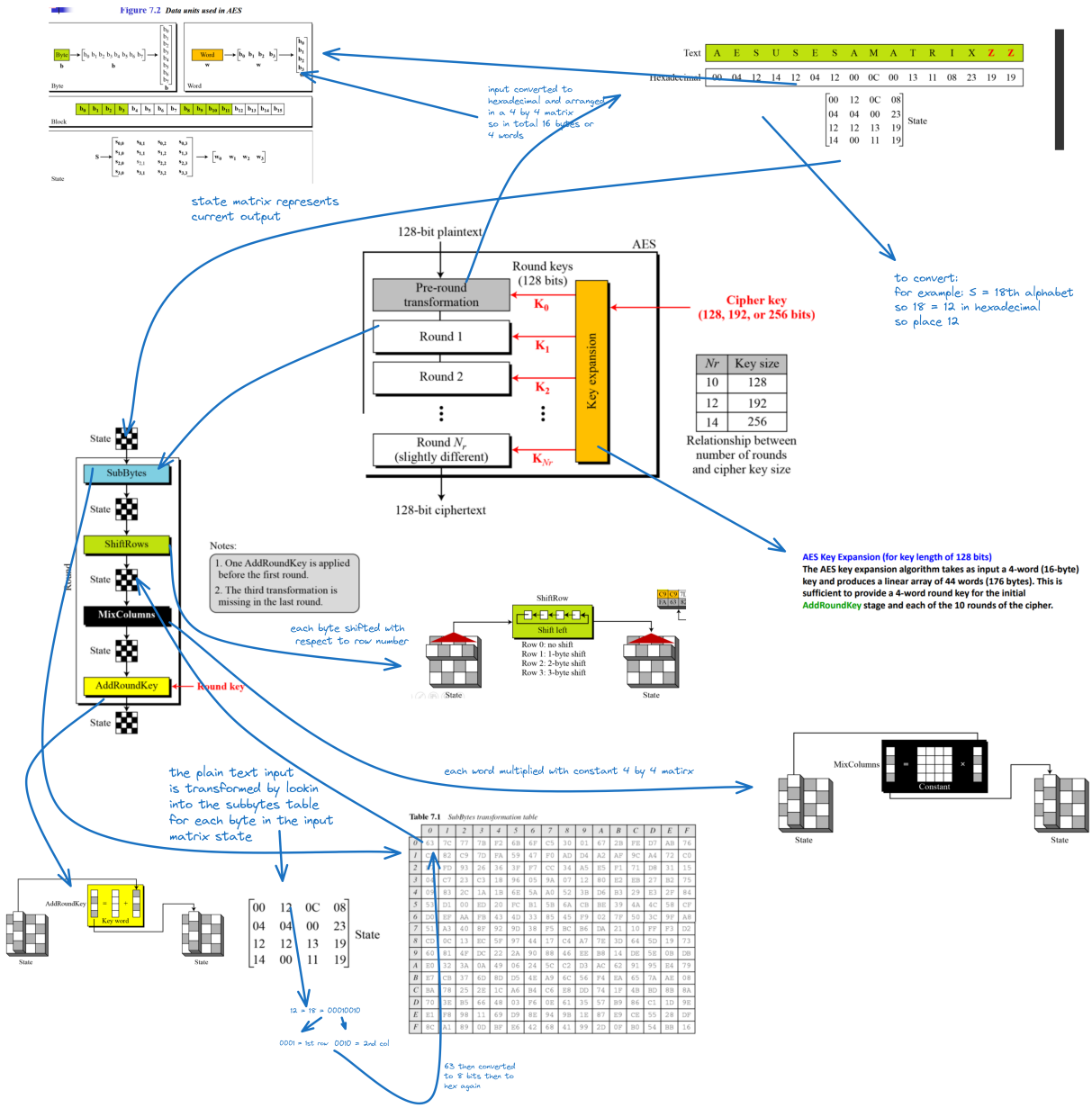
5. Output

The output after the final round is the ciphertext block, which is 128 bits in size regardless of the initial key size.

Decryption

The decryption process in AES is essentially the reverse of the encryption process, with steps executed in reverse order and using inverse operations for SubBytes (InvSubBytes), ShiftRows (InvShiftRows), and MixColumns (InvMixColumns).

This detailed step-by-step process allows AES to provide a high level of security, with resistance against various cryptographic attacks, making it suitable for encrypting sensitive data across a wide range of applications.



Authentication Using Symmetric Encryption

- Symmetric encryption can theoretically be used for message authentication, given that the genuine sender, possessing the shared key, would be able to encrypt the message properly for the intended recipient.
- Adding an error-detection code, sequence number, and timestamp to the message can potentially ensure the message's authenticity and timeliness.
- **Drawback:** Symmetric encryption alone isn't sufficient for secure data authentication. For instance, in ECB encryption mode, ciphertext blocks can be

reordered by an attacker, altering the message's overall meaning without affecting individual block decryption.

Message Authentication without Message Encryption

- **Concept:** Generation of an authentication tag appended to each transmitted message, facilitating message authentication without encryption.
- The authentication function operates separately from the message, allowing the message to be read independently at the destination.
- **Benefits:**
  1. Cost-effective and reliable for broadcasting a single message to multiple destinations with one destination assigned for monitoring authenticity.
  2. Helps in scenarios where selective message authentication is preferable to reduce the load on heavily burdened systems.
  3. Allows for the authentication of plaintext computer programs without needing constant decryption, saving processing resources. Authentication tags can verify program integrity when needed.
- **Limitation:** Does not provide message confidentiality.
- Suggests that combining authentication and confidentiality in a single algorithm could be achieved by encrypting both the message and its authentication tag, though typically these functions are separate.
- Highlights the necessity for both authentication and encryption in securing communications and ensuring data integrity.

Message Authentication Codes (MACs)

Definition and Structure

- A MAC is a small block of data generated using a secret key, which is appended to a message for the purposes of authentication.
- The formula to calculate MAC is given by:

$$MAC_M = F(K_{AB}, M)$$

Properties

1. **Fixed Size**: The MAC is of a small fixed size, regardless of the message size.
2. **Collision Resistance**: It should be practically infeasible to find two different messages that result in the same MAC. This property is essential to prevent attackers from generating valid MACs for fraudulent messages.

Process

1. **Generation**: Sender calculates the MAC using the secret key and the message.
2. **Transmission**: The message along with the generated MAC is transmitted to the receiver.
3. **Verification**: The receiver calculates a new MAC using the received message and the shared secret key. This newly calculated MAC is compared with the received MAC.

Assurances

If the received MAC matches the calculated MAC, the receiver has the following assurances:

1. **Integrity**: The message has not been altered during transmission.
2. **Authenticity**: The message is indeed from the alleged sender, as only they would know the secret key to generate a valid MAC.
3. **Sequence Integrity**: If the message includes a sequence number, the receiver is assured of the proper sequence, preventing attackers from altering the sequence number successfully.

Algorithm Choices

- In the past, algorithms like DES (Data Encryption Standard) were recommended (as per FIPS PUB 113, 1985) for generating MACs.
- Currently, AES (Advanced Encryption Standard) is a more suitable choice due to its enhanced security features.
- The bits of the encrypted version of the message are used to generate the MAC. Previously 16- or 32-bit codes were typical, but modern requirements demand

larger sizes for better collision resistance.

Comparison with Encryption

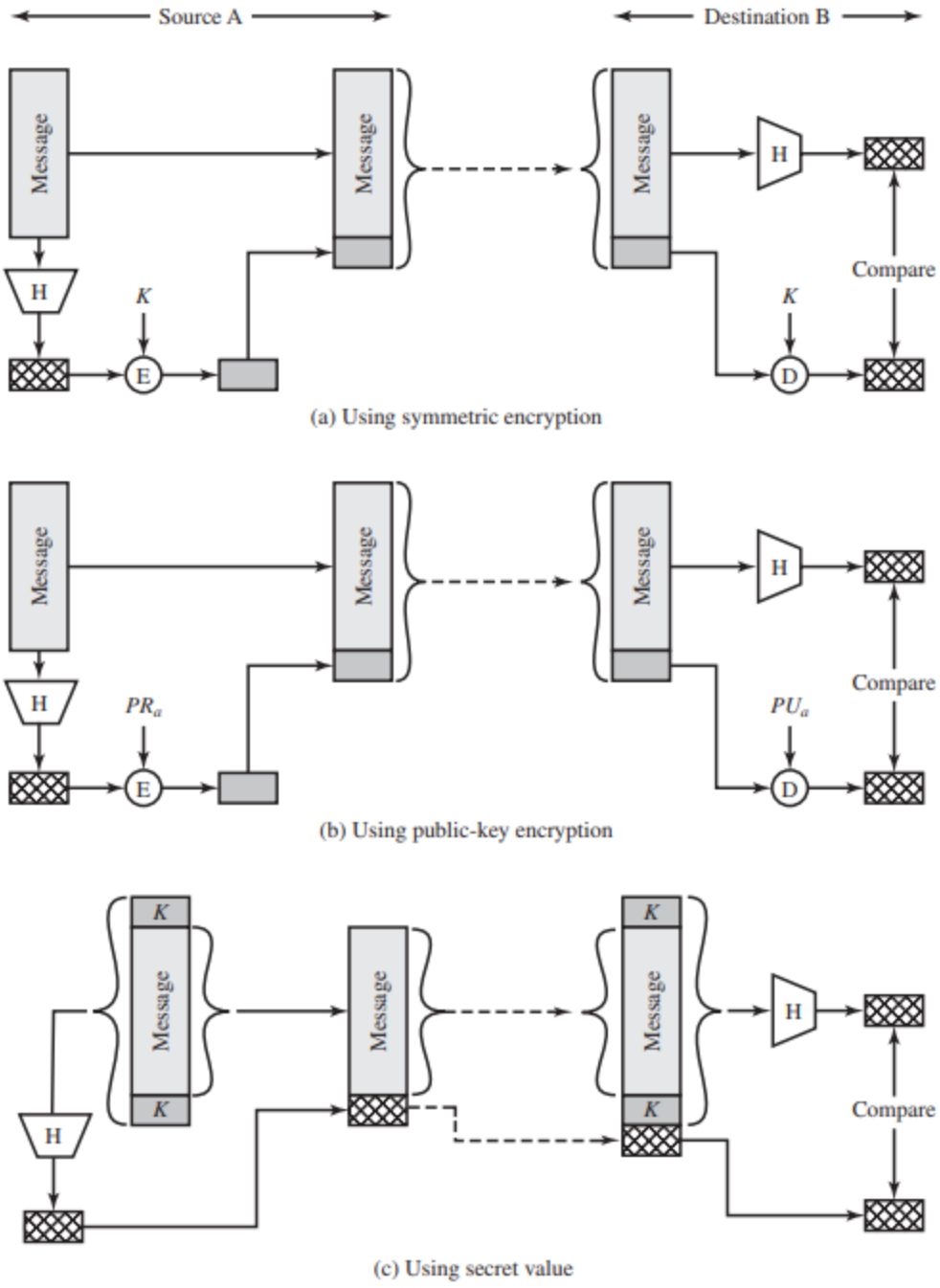
- While similar to encryption, the algorithm generating the MAC doesn't need to be reversible.
- Due to the mathematical properties of the authentication function, it is considered less vulnerable to being broken compared to encryption algorithms.

One-Way Hash Functions for Message Authentication

Definition and Properties

- **One-way hash function** is an alternative to message authentication codes (MACs).
- It takes a variable-size message M as input and produces a fixed-size message digest H(M) as output.
- The message is padded to a fixed length, with padding including the length of the original message, enhancing security.
- It does not use a secret key as an input, unlike MACs.

Approaches to Authentication using Hash Function



1. Symmetric Encryption (Figure 2.5a)

- The message digest is encrypted using symmetric encryption.
- Assumes that only the sender and receiver share the encryption key, assuring authenticity.

2. Public-Key Encryption (Figure 2.5b)

- The message digest is encrypted using public-key encryption.
- Provides a digital signature and message authentication.
- Does not require distribution of keys between communicating parties.

3. Keyed Hash MAC (Figure 2.5c)

- Does not involve encryption, thus reducing computational demands.
- A secret key, shared between communicating parties (A and B), is used in the hash function.
- The hash function is applied over the concatenation of the secret key and the message, formulated as

$$MDM = H(K \oplus M \oplus K)$$

- The message along with the message digest is sent to B.
- B, having the secret key, can recompute the hash function to verify the message digest.
- As the secret key is not transmitted, it offers protection against message modification by attackers.
- Using the secret key as both a prefix and a suffix to the message ensures better security compared to using it only as a prefix or suffix.

Advantages over Encrypting the Entire Message

1. **Less Computation**: Requires less computational power compared to encryption of the entire message.
2. **Cost-Effective**: Avoids the costs associated with encryption hardware.
3. **Faster for Small Data Blocks**: Avoids the initialization/invocation overhead time which is substantial for small data blocks in encryption algorithms.
4. **Avoids Patent Issues**: Skips potential legal issues tied to patented encryption algorithms.

Secure Hash Functions

Purpose

- Create a "fingerprint" of a file, message, or data block.
- Important in message authentication and digital signatures.

Requirements

A secure hash function. H needs to satisfy the following conditions:

1. **Versatility**: Can be applied to any size of data block.
2. **Fixed-Length Output**: Produces a fixed-length output.
3. **Efficient to Compute**: Relatively easy to compute for any input x making both hardware and software implementations feasible.
4. **One-Way or Preimage Resistant**: It is computationally infeasible to find an x given a hash code h such that.

$$H(x) = h$$

5. **Weak Collision Resistant**: It is computationally infeasible to find a different block y where

$$y \neq x$$

that produces the same hash output as x i.e.,

$$H(y) = H(x)$$

1. **Strong Collision Resistant**: It is computationally infeasible to find any pair x, y that gives the same hash output, i.e.

$$H(y) = H(x)$$

The first three properties ensure the practical application of the hash function in message authentication.

Properties Analysis

- **Fourth Property (One-Way Property)**: Important in authentication techniques involving a secret value. It prevents attackers from discovering the secret value by inverting the hash function from an intercepted hash code.
- **Fifth Property**: Protects against forgery by ensuring it is not possible to find an alternate message with the same hash value. If violated, an attacker could generate an alternate message with the same hash code and deceive the recipient.

Hash Function Strength

- **Weak Hash Function**: Satisfies the first five properties mentioned above.
- **Strong Hash Function**: Satisfies all six properties, offering protection against attacks where a message generated for signing can be exploited to validate a different deceptive message.

Additional Benefits

- **Data Integrity**: Besides authentication, hash functions ensure data integrity by checking if any bits in the message were altered during transmission, as any alteration would result in a different hash value.

This section discusses the security aspects of hash functions, specifically focusing on their susceptibility to attacks (cryptanalysis and brute-force) and their application in various security domains. Here's a breakdown of the main points discussed in the excerpt:

Security of Hash Functions

Attack Approaches

1. **Cryptanalysis**: Involves exploiting logical weaknesses in the hash function algorithm, akin to the approach used in symmetric encryption.
2. **Brute-force Attack**: Relies on trying all possible inputs until the correct one is found, the strength against such attacks is determined by the length of the hash code generated by the function.

Strength against Brute-force Attacks

The level of effort required for different types of brute-force attacks is proportional to:

1. **Preimage Resistant**: 2^n
2. **Second Preimage Resistant**: 2^n
3. **Collision Resistant**: $2^{n/2}$

Note: Collision resistance is a desirable property for a general-purpose secure hash function, and the value $2^{n/2}$ determines the strength against brute-force attacks.

Examples

- A design by Van Oorschot and Wiener illustrated the feasibility of finding collisions in MD5 (128-bit hash length) within 24 days using a specially designed machine, indicating the inadequacy of a 128-bit code.
- Considering technological advancements, even a 160-bit hash length might not offer secure protection against brute-force attacks in the present context.

Secure Hash Algorithm (SHA)

1. **Initial Versions**
    - **SHA**: Introduced in 1993 (160-bit hash value).
    - **SHA-1**: Revised version introduced in 1995 to address discovered weaknesses (160-bit hash value).
2. **SHA-2 Family**
    - **Introduction**: Introduced in 2002 with new versions providing hash lengths of 256, 384, and 512 bits.
    - **Structural Similarity to SHA-1**: Despite offering enhanced security, it shares structural similarities with SHA-1, which could potentially be a vulnerability.
3. **SHA-3**
    - **Introduction**: Introduced in 2015 to provide an alternative to SHA-2, with a significantly different structure to mitigate potential vulnerabilities inherent in SHA-2 and SHA-1.

Applications of Hash Functions

Besides message authentication and creation of digital signatures, hash functions find utility in:

1. **Password Security**
    - **Hash Storage**: Passwords are stored as hashes rather than plaintext, enhancing security by preventing direct retrieval of passwords by intruders.
    - **Verification**: Password verification is done by comparing the hash of the entered password with the stored hash.
    - **Required Properties**: Preimage resistance and possibly second preimage resistance.
2. **Intrusion Detection**
    - **Hash Value Storage**: Hash values of files (H(F)) are securely stored to monitor any unauthorized modifications.
    - **File Integrity Verification**: File integrity is verified by recomputing and comparing the hash values at later stages.
    - **Required Property**: Weak second preimage resistance.

PUBLIC-KEY ENCRYPTION

Introduction

- **Important**: Used in message authentication and key distribution.
- **First Proposed**: By Diffie and Hellman in 1976.
- **Nature**: Asymmetric, involving two keys (public and private).
- **Key Characteristics**: Based on mathematical functions rather than simple bit pattern operations.
- **Misconceptions**: Not necessarily more secure than symmetric encryption; hasn't made symmetric encryption obsolete due to computational overhead; key distribution isn't trivial compared to symmetric encryption.

Structure and Components

1. **Plaintext**: The original readable message or data fed into the algorithm.
2. **Encryption Algorithm**: Performs transformations on the plaintext.
3. **Keys**:
    - Public Key: Known to others, used for encryption.
    - Private Key: Known only to the owner, used for decryption.
4. **Ciphertext**: Scrambled message output depending on the plaintext and the key.
5. **Decryption Algorithm**: Uses the matching key to revert the ciphertext back to the original plaintext.

Key Steps in Public-Key Cryptography

1. Users generate a pair of keys for encryption and decryption of messages.
2. Users publish one key (public) in an accessible register; the other (private) is kept secret.
3. To send a confidential message, the sender encrypts it using the receiver's public key.
4. The receiver decrypts the message using their private key, ensuring security and confidentiality.

Modes of Operation

- **Confidentiality Mode**: (Figure 2.6a) Aims to provide confidentiality; decryption is possible only with the intended recipient's private key.
- **Authentication/Data Integrity Mode**: (Figure 2.6b) Aims for authentication and/or data integrity; successful recovery of plaintext using a public key authenticates the sender and ensures data integrity.

Applications

1. **Digital Signature**: Authentication through a unique signature created using private keys.
2. **Symmetric Key Distribution**: Secure distribution of symmetric keys using public-key cryptography.

3. **Encryption of Secret Keys**: Using public-key systems to encrypt secret keys for secure transmission.

Notable Points

- Public-key cryptography has not entirely replaced symmetric encryption due to computational overheads.
- The security of public-key encryption is not fundamentally superior to symmetric encryption, and it depends on key length and the difficulty of breaking the cipher.
- Implementation and protocol security are vital to achieving the intended security level, either for confidentiality or authentication/data integrity.

Applications for Public-Key Cryptosystems

Characteristics

- Utilizes cryptographic algorithms with two keys: a public key (known to others) and a private key (known only to the owner).
- Depending on the application, cryptographic functions are performed using either the sender's private key, the receiver's public key, or both.

Classifications

1. **Digital Signature**: Helps in authenticating the origin of the message.
2. **Symmetric Key Distribution**: Facilitates the secure distribution of symmetric keys.
3. **Encryption of Secret Keys**: Used for the encryption of secret keys to ensure their secure transmission.

Suitability of Algorithms

- Different algorithms are suitable for different applications.
- **Table 2.3** shows which algorithms support which applications:
    - **RSA**: Suitable for Digital Signature, Symmetric Key Distribution, and Encryption of Secret Keys.
    - **Diffie-Hellman**: Only suitable for Symmetric Key Distribution.

- **DSS**: Only suitable for Digital Signature.
- **Elliptic Curve**: Suitable for Digital Signature, Symmetric Key Distribution, and Encryption of Secret Keys.

Requirements for Public-Key Cryptography

According to Diffie and Hellman's postulation in 1976, the algorithm based on two related keys must fulfill the following conditions:

1. It must be computationally easy for a party B to generate a pair of keys (public key PUb, private key PRb).
2. For a sender A, it should be computationally easy to generate the corresponding ciphertext, C, given the public key and the message, M:

   ```
   C = E(PUb, M)
   ```

3. It should be computationally easy for the receiver B to decrypt the ciphertext using the private key to recover the original message:

   ```
   M = D(PRb, C) = D[PRb, E(PUb, M)]
   ```

4. It must be computationally infeasible for an adversary, knowing the public key PUb, to determine the private key PRb.
5. It must be computationally infeasible for an adversary, knowing the public key PUb and a ciphertext C, to recover the original message M.

An additional, useful but not necessary condition:
6. Either of the two related keys can be used for encryption, with the other being used for decryption:

```
M = D[PUb, E(PRb, M)] = D[PRb, E(PUb, M)]
```

Asymmetric Encryption Algorithms

In this section, we delve briefly into some of the most popular asymmetric encryption algorithms, with a promise to explore them in further detail in Chapter 21.

1. **RSA**
    - **Developers**: Ron Rivest, Adi Shamir, and Len Adleman (MIT, 1977).
    - **Publication**: First published in 1978.
    - **Features**:
        - A block cipher where both plaintext and ciphertext are integers between 0 and n-1 (for some n).
        - It remains a popular and widely implemented approach to public-key encryption.
        - Has stood up against various cryptographic challenges, proving that increasing key sizes can provide strong security.
    - **Current Usage**: Key sizes of about 1024-bit (300 decimal digits) are considered strong for almost all applications.
2. **Diffie-Hellman Key Agreement**
    - **Developers**: Whitfield Diffie and Martin Hellman.
    - **Publication**: First publicized in a seminal paper in 1976.
    - **Features**:
        - Facilitates secure agreement on a shared secret between two users.
        - This shared secret can later be used as a key for symmetric encryption.
        - Limited to the exchange of keys and doesn't include encryption or digital signature functionalities.
        - Used in various commercial products.
3. **Digital Signature Standard (DSS)**
    - **Organization**: National Institute of Standards and Technology (NIST).
    - **Publication**: Initially published as FIPS PUB 186 in May 1994, with several revisions thereafter.
    - **Features**:
        - Utilizes SHA-1 and the Digital Signature Algorithm (DSA).

    - Provides only the digital signature functionality.
    - Cannot be employed for encryption or key exchange.

4. **Elliptic Curve Cryptography (ECC)**
    - **Usage**: Recently gaining popularity and has started appearing in standardization efforts including the IEEE P1363 Standard for Public-Key Cryptography.
    - **Features**:
        - Offers similar security levels as RSA but with smaller bit sizes, hence reducing processing overhead.
        - This is particularly beneficial for e-commerce platforms with a high volume of secure transactions.
    - **Current Status**:
        - Despite its promising features, it is relatively new and therefore, has not yet attained a confidence level comparable to RSA.
        - The cryptographic community is actively working on identifying potential weaknesses and solidifying the ECC system.

These algorithms are a cornerstone in securing digital communications and transactions, each with its unique properties and applications. They will be discussed in greater detail in subsequent chapters.

Digital Signature

Digital signatures are a critical application of public-key encryption systems. These signatures, as defined by NIST FIPS PUB 186-4, are cryptographic transformations of data that, when appropriately employed, allow for the verification of the origin of the data, the integrity of the data, and non-repudiation of the signature.

In essence, a digital signature is a bit pattern that is dependent on the data it represents, created by an entity as a representation of a data file or message. Another entity can verify that the data has not been tampered with since the time of signing, and that the signature indeed belongs to the person who claims to have signed it. The signatory cannot later deny the authenticity of the signature.

FIPS 186-4 mandates the use of one of three digital signature algorithms:

1. **Digital Signature Algorithm (DSA)**: This algorithm, originally approved by NIST, relies on the computational complexity of finding discrete logarithms.
2. **RSA Digital Signature Algorithm**: An algorithm grounded on the well-established RSA public-key encryption system.
3. **Elliptic Curve Digital Signature Algorithm (ECDSA)**: This algorithm is based on the elliptic curve cryptography approach.

Process of Creating and Verifying Digital Signatures

Refer to Figure 2.7 for a schematic representation of the digital signature creation and verification process, which generally unfolds as follows:

1. **Signature Creation (Bob)**
    - Bob wishes to send a message to Alice.
    - He generates a hash value of the message using a secure hash function (e.g., SHA-512).
    - This hash value, along with Bob's private key, is used to create a digital signature.
    - The message, along with the signature, is then sent to Alice.
2. **Signature Verification (Alice)**
    - Alice receives the message and the signature.
    - She computes the hash value of the received message.
    - Using Bob's public key, she verifies the signature with a digital signature verification algorithm.
    - If verified, it assures Alice that the message has been sent by Bob and hasn't been altered during transmission.

It is vital to note that while digital signatures guarantee authenticity and integrity, they do not assure confidentiality. Despite the encryption involved in generating the signature, the message itself is transmitted "in the clear," leaving it vulnerable to eavesdropping. However, the signature safeguards the message from alterations, confirming both its origin and its integrity.

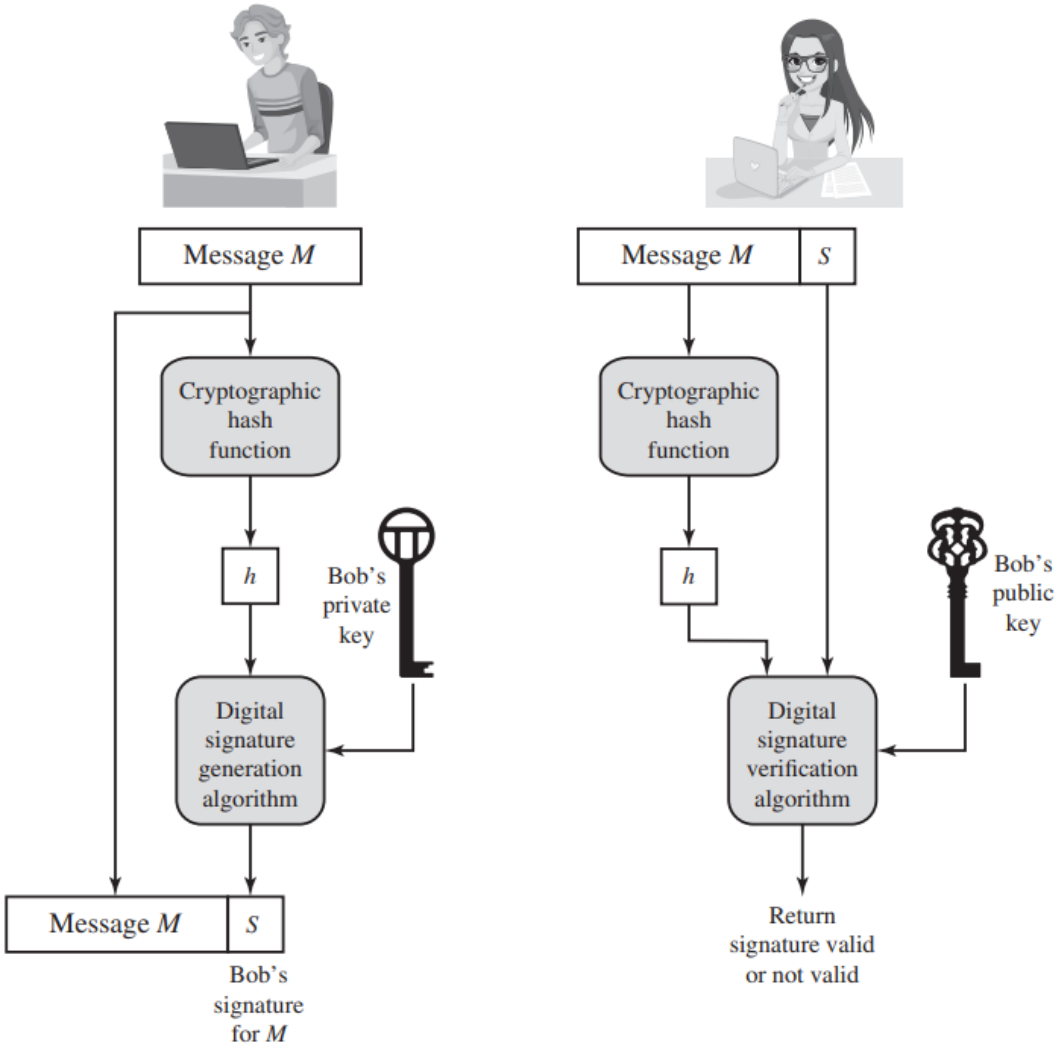


Figure 2.7 Simplified Depiction of Essential Elements of Digital Signature Process

Public-Key Certificates

Overview

Public-key certificates are essential in addressing the security issues related to the sharing of public keys in asymmetric cryptography. They ensure that a public key indeed belongs to the person claiming to own it, preventing forgeries and impersonations.

The Problem

- **Public-key Encryption**: Users share public keys to exchange information securely.
- **Risk**: Anyone can forge public announcements of public keys, impersonating another user and gaining unauthorized access to encrypted messages and authentication processes.

The Solution: Public-Key Certificates

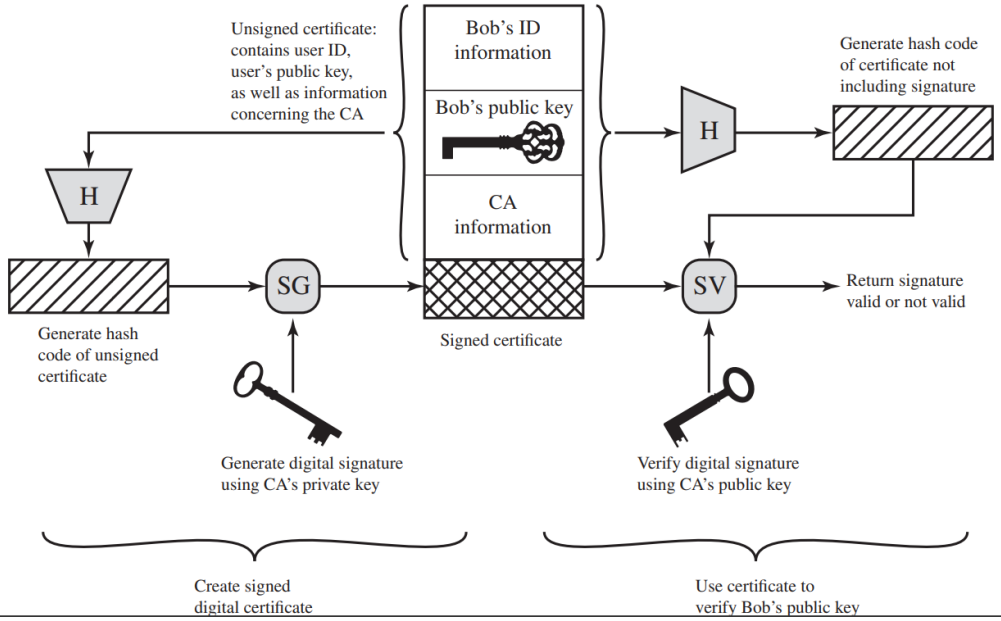
- **Components**:
    1. **Public Key**: The user's public key.
    2. **User ID**: Identifying information of the key owner.
    3. **Signature of a Certificate Authority (CA)**: A trusted third party vouches for the authenticity of the public key and its owner.
    4. **Validity Information**: The period for which the certificate is valid.
- **Certificate Authority (CA)**: A trusted entity like a government agency or a financial institution that issues certificates.
- **Process**:
    1. **Creation**: User software creates a pair of keys (public and private).
    2. **Preparation**: Client prepares an unsigned certificate including the user ID and public key.
    3. **Submission**: User submits the unsigned certificate to a CA via a secure manner.
    4. **Signature Creation**: CA creates a digital signature using a hash function and attaches it to the unsigned certificate.
    5. **Return**: CA returns the signed certificate to the client.
    6. **Verification**: Any user can verify the certificate using the CA’s public key.

Example: X.509 Standard

The X.509 standard is a universally accepted format for public-key certificates. It's used in various network security applications such as IPsec, TLS, SSH, and S/MIME.

Detailed Steps

1. **Key Creation**:
   - User's client software generates a public and a private key.
   - **Example**: RSA key pair generation.
2. **Certificate Preparation**:
   - Client prepares an unsigned certificate with user ID and public key.
   - **Example**: Alice includes her user ID and RSA public key in the certificate.
3. **Submission to CA**:
   - User provides the unsigned certificate to CA securely.
   - **Example**: Alice submits her unsigned certificate via registered email or secure web form to a CA.
4. **Signature Creation by CA**:
   - CA calculates the hash code of the unsigned certificate using a hash function (e.g., SHA-256).
   - CA generates a digital signature using its private key.
   - **Example**: The CA signs Alice’s unsigned certificate after hashing.
5. **Return of Signed Certificate**:
   - CA returns the signed certificate to the client.
   - **Example**: Alice receives her signed certificate from the CA.
6. **Distribution**:
   - Client may provide the signed certificate to any other user.
   - **Example**: Alice can share her signed certificate with Bob.
7. **Verification**:
   - Any user calculates the hash code of the received certificate and verifies the digital signature using CA's public key.
   - **Example**: Bob verifies Alice’s certificate using the CA’s public key, ensuring that the public key indeed belongs to Alice.



Digital User Authentication Principles

Overview

- **Definition**: Digital user authentication is the process of establishing confidence in user identities presented electronically to an information system.
- **Purpose**: It enables systems to determine whether an authenticated individual is authorized to perform specific functions.
- **Scope**: This process can happen both over open networks, like the Internet, and locally across a local area network.

NIST SP 800-63-3 Model

- **Registration Requirement**: The user must be registered with the system.
- **Entities Involved**:
  1. **Applicant**: Applies to become a subscriber of a credential service provider (CSP).
  2. **Registration Authority (RA)**: Establishes and vouches for the identity of an applicant to a CSP.

3. **Credential Service Provider (CSP)**: Issues electronic credentials to the subscriber.
4. **Subscriber/Claimant**: Registered user who needs to be authenticated.
5. **Verifier**: Party verifying the identity of the claimant.
6. **Relying Party (RP)**: Uses authenticated information provided by the verifier to make access control or authorization decisions.

- **Credential and Token**: The credential is a data structure binding an identity to a token (e.g. encryption key, password), which may be issued by the CSP, generated by the subscriber, or provided by a third party.
- **Authentication Process**: The claimant demonstrates possession and control of a token to a verifier through an authentication protocol.

Security Requirements (NIST SP 800-171)

1. **Identify**: Information system users, processes, or devices.
2. **Authenticate**: Verify the identities of users, processes, or devices before allowing access.
3. **Derived Requirements**: Include multifactor authentication, replay-resistant mechanisms, identifier management, password protection, and obscuring feedback of authentication information.

Means of Authentication

1. **Something the Individual Knows**: E.g., password, PIN, answers to prearranged questions.
2. **Something the Individual Possesses (Token)**: E.g., electronic keycards, smart cards.
3. **Something the Individual Is (Static Biometrics)**: E.g., fingerprint, retina, face recognition.
4. **Something the Individual Does (Dynamic Biometrics)**: E.g., voice pattern, handwriting characteristics, typing rhythm.

- **Multifactor Authentication**: The use of more than one means of authentication for added security.

- **Issues with Authentication Methods**: Each method has potential vulnerabilities, such as the risk of theft, loss, forgery, user acceptance, false positives/negatives, cost, and convenience.

Assurance Levels:

- **Definition:**
  - Describes the degree of certainty that a user's presented credential truly refers to their identity.
  - Assurance comprises:
    1. Confidence in the vetting process used to establish the identity.
    2. Confidence that the user presenting the credential is the individual to whom it was issued.
- **Levels of Assurance:**
  - **Level 1:**
    - Little or no confidence in asserted identity’s validity.
    - Suitable for consumer registration on discussion boards.
    - Typically involves user-supplied ID and password.
  - **Level 2:**
    - Some confidence in asserted identity’s validity.
    - Suitable for businesses requiring verified initial identity assertion.
    - Requires secure authentication protocol and one other means of authentication.
  - **Level 3:**
    - High confidence in asserted identity’s validity.
    - Suitable for access to high-value restricted services, e.g., submission of confidential patent information.
    - Requires more than one factor of authentication.
  - **Level 4:**

        - Very high confidence in asserted identity’s validity.
        - Suitable for access to very high-value restricted services or harmful improper access, e.g., law enforcement databases.
        - Requires multiple factors and in-person registration.

Potential Impact:

- **Definition:**
    - Refers to the level of adverse effect on organizations or individuals due to a breach of security, specifically, a failure in user authentication.
- **Levels of Potential Impact:**
    - **Low:**
        - Limited adverse effect, minor damage, or minor financial loss.
    - **Moderate:**
        - Serious adverse effect, significant damage, significant financial loss, or significant harm to individuals without loss of life.
    - **High:**
        - Severe or catastrophic adverse effect, major damage, major financial loss, or severe harm to individuals, including loss of life or serious life-threatening injuries.

Areas of Risk & Assurance Level Impact Profiles:

- **Mapping:**
    - The appropriate level of assurance to deal with potential impact depends on the context and risk areas concerning the organization.
- **Risk Assessment:**
    - For each information system or service asset, determine the level of impact if an authentication failure occurs, based on concerned risk areas.
    - Examples:

        - Financial loss due to unauthorized database access can be Low, Moderate, or High, depending on the nature of the database and the potential liability.

- **Potential Impact Categories for Authentication Errors:**
    - Low: Assurance level 1.
    - Moderate: Assurance level 2 or 3.
    - High: Assurance level 4.
- **Maximum Potential Impacts for Each Assurance Level:**
    - Table 3.2 gives potential impacts for each assurance level in categories like inconvenience, financial loss, harm to organization programs, unauthorized release of sensitive information, personal safety, and civil or criminal violations.
- **Implementation:**
    - Pick an assurance level that meets or exceeds the requirements for assurance in each of the categories listed in Table 3.2.
    - If any category has a potential impact of high or if the personal safety category has a potential impact of moderate or high, then level 4 assurance should be implemented.

Password-Based Authentication

Definition

Password-Based Authentication is a security process that validates the identity of users by comparing entered passwords against a stored password associated with the user ID. It is a common defense mechanism against unauthorized access in multiuser systems, network servers, and web-based services.

Key Components

- **User ID:** Determines user authorization level and privileges.
- **Password:** Authenticates the ID of the individual logging on to the system.

- **Password File:** Stores hashed passwords indexed by user ID.
- **Salt Value:** Combined with a password to create a hashed password. Increases security against offline dictionary attacks.

Vulnerabilities

- **Offline Dictionary Attack:** Attacker obtains system password file and compares the password hashes against commonly used passwords.
- **Specific Account Attack:** Targets a specific account and submits password guesses until the correct password is discovered.
- **Popular Password Attack:** Uses a popular password and tries it against a wide range of user IDs.
- **Password Guessing:** Gains knowledge about account holder and password policies to guess the password.
- **Workstation Hijacking:** Attacker waits until a logged-in workstation is unattended.
- **Exploiting User Mistakes:** Adversaries may read written passwords or trick users into revealing passwords.
- **Exploiting Multiple Password Use:** Damaging if different network devices share the same or a similar password for a given user.
- **Electronic Monitoring:** Vulnerable to eavesdropping if communicated across a network.

Countermeasures

- **Strong Access Controls:** Prevent unauthorized access to the password file.
- **Intrusion Detection Measures:** Identify compromises.
- **Account Lockout Mechanisms:** Lock out access after a number of failed login attempts.
- **Password Policies:** Inhibit the selection of common passwords and ensure passwords are difficult to guess.

- **User Training:** Educate users on keeping passwords secure.
- **Automatic Logout:** Log out users after a period of inactivity.
- **Encryption and Hashing:** Secure password during transmission and storage.

Hashed Passwords

- **Hash Algorithm:** Converts passwords into fixed-length hash codes. It's designed to be slow to thwart attacks.
- **Salt Value:** Prevents duplicate passwords from being visible in the password file.
- **MD5 and Bcrypt:** Examples of stronger hash/salt schemes, using a salt of up to 48 bits and 128 bits respectively, producing a 128-bit and a 192-bit hash value.

UNIX Implementations

- **Crypt(3):** A password scheme based on DES; considered inadequate now due to its vulnerability to dictionary attacks.
- **MD5 Crypt Routine:** Uses a salt of up to 48 bits and has no limitations on password length, producing a 128-bit hash value.
- **Bcrypt:** Developed for OpenBSD; uses the Blowfish symmetric block cipher, allows passwords up to 55 characters, and requires a random salt value of 128 bits to produce a 192-bit hash value.

Password Cracking

- **Traditional Approaches:** Develop a large dictionary of possible passwords and try each against the password file.
- **Rainbow Table:** A precomputed table for reversing cryptographic hash functions, usually for cracking password hashes.

Password File Access Control

- **Definition:** A security method that restricts access to the password file to impede unauthorized access and attacks.

- **Shadow Password File:** Separate file for hashed passwords, intended to be more secure.
- **Vulnerabilities:**
  - Unanticipated break-ins exploiting OS vulnerabilities.
  - Accidents of protection making the password file readable.
  - Users using the same password across different systems.
  - Weakness in physical security enabling unauthorized access.
  - Network traffic sniffing to collect user IDs and passwords.
- **Measures:**
  - The security policy should complement access control with strategies to enforce strong password selection.

Password Selection Strategies

- **Goal:** To help users choose memorable, non-guessable passwords.
- **Strategies:**
  - **User Education:** Educating users on the importance of hard-to-guess passwords and providing guidelines.
  - **Computer-Generated Passwords:** Random or semi-random passwords created by computer programs.
  - **Reactive Password Checking:** The system periodically runs its own password cracker to find and cancel guessable passwords.
  - **Complex Password Policy:** Proactive password checker allowing users to select their passwords within system-guided parameters.
  - **Advice for Users:** Use the first letter of each word of a unique phrase as a password, ensuring it is not a well-known phrase.
- **Challenges:** Balancing between user acceptability and password strength.

Proactive Password Checker & Rule Enforcement

- **Definition:** It allows users to select their password but rejects it if it does not conform to predefined rules.
- **NIST SP 800-63-2 Rules:**
  - Passwords must have at least sixteen characters (basic16).
  - Passwords must have at least eight characters including an uppercase and lowercase letter, a symbol, and a digit (comprehensive8).
  - **Basic16 is considered superior due to higher user acceptance and security against extensive guesses.**
- **Automated Checker:** Tools like pam_passwdqc can automate the enforcement of password rules.

Password Checker & Bloom Filter

- **Password Checker:** It checks user-selected passwords against a dictionary of disapproved or “bad” passwords.
  - **Challenges:** Requires substantial space for the dictionary and time to search through it.
- **Bloom Filter:** A space-efficient probabilistic data structure used to test whether an element is a member of a set.
  - **Advantage:** Offers compression and allows for straightforward password checking.
  - **Functioning:** Utilizes multiple hash functions and a bit array.
  - **False Positives:** A challenge in Bloom Filter; designing the hash scheme is critical to minimize them.
  - **Efficiency:** It is more space-efficient and faster in comparison to storing and searching the entire dictionary.

Token-Based Authentication

Overview

- Tokens are objects used for user authentication.

- There are primarily two types of widely used tokens: Memory Cards and Smart Cards.
- Tokens enhance security by combining something a user knows (e.g., password, PIN) with something a user has (e.g., card).

1. Memory Cards

- **Definition:** Cards capable of storing but not processing data.
- **Common Types:**
  - **Magnetic Stripe Cards:** (e.g., Bank Cards) can store security codes; readable and reprogrammable.
  - **Electronic Memory Cards:** Store data internally.
- **Usage:**
  - Can be used alone for physical access.
  - For authentication, usually combined with a password or PIN (e.g., ATMs).
- **Drawbacks:**
  - Requires special readers, increasing cost and maintenance.
  - Loss of the token can prevent system access.
  - May be inconvenient for computer access leading to user dissatisfaction.

2. Smart Cards

- **Definition:** Cards with embedded microprocessors.
- **Characteristics:**
  - Can have manual interfaces like a keypad and display.
  - Can have electronic interfaces that can be contact or contactless.
  - Contain ROM, EEPROM, and RAM for storing various types of data.
  - Supports encryption, decryption, and digital signatures.

- **Authentication Protocols:**
  - **Static Protocol:** Similar to a memory token.
  - **Dynamic Password Generator:** Generates a unique password periodically.
  - **Challenge-Response:** Token generates a response based on a challenge from the computer system.
- **Electronic Interface Types:**
  - **Contact:** Requires insertion into a smart card reader.
  - **Contactless:** Communicates using radio frequencies and requires proximity to a reader.

Electronic Identity (eID) Cards:

eID Cards serve as national identity cards that can be used by citizens for access to various government and commercial services. These cards are essentially smart cards verified by the national government as valid and authentic.

Features:

1. **Human-Readable Data**: Includes personal data such as name, date of birth, and address along with document number, a unique identifier, and a card access number (CAN).
2. **Machine Readable Zone (MRZ)**: Present at the back of the card containing human- and machine-readable text.
3. **Electronic Functions**: The card is equipped with several electronic functions, each with its protected dataset.
   - **ePass Function**: Reserved for government use for offline identity verification, such as at passport control checkpoints.
   - **eID Function**: Used for general-purpose applications in both online and offline services, allowing authorized service access with cardholder permission.

- **eSign Function**: Optional function used for generating digital signatures, storing a private key and a certificate verifying the key.

Applications:

1. **Offline Use**: Utilized in inspection systems by law enforcement for checks by police or border control officers, providing identification and biometric information like facial image and fingerprints for verification.
2. **Online Use**: Enables user authentication, where a user visits a website requiring authentication. Data is exchanged between the eID card and the terminal reader in encrypted form after the user enters the correct PIN.

Requirements:

- **Software**: Needed for requesting and accepting the PIN number and for message redirection.
- **Hardware**: An eID card reader, either external or internal, contact, or contactless.

Password Authenticated Connection Establishment (PACE):

PACE is a security protocol that ensures secure access to the contactless RF chip in the eID card. It protects unauthorized access to the information on the eID card by establishing a secure and authenticated connection.

Functionality:

1. **Online Applications**: Access to the card is established by the user entering the 6-digit PIN, which should only be known to the holder of the card.
2. **Offline Applications**: Either the MRZ or the six-digit card access number (CAN) is used to establish access.

Purpose:

- **Security**: Provides enhanced security for contactless communication with eID cards by preventing unauthorized reading of the RF chip in the card.
- **Access Control**: Establishes a secure connection only after verifying the correct password (PIN or MRZ or CAN), thus providing an additional layer

of access control to the sensitive information stored on the eID card.

Importance:

- PACE is crucial in maintaining the integrity and confidentiality of the information stored in eID cards.
- It serves as an important measure to ensure that the eID cards are used securely and correctly in various online and offline applications.

Biometric Authentication

Overview:

- **Biometric Authentication**: Relies on unique physical characteristics to authenticate individuals.
- **Types**: Includes static characteristics like fingerprints, facial features, and iris patterns, and dynamic characteristics like voiceprint and signature.
- **Process**: Biometric systems capture these features and compare them against stored templates for authentication.

Characteristics Used:

1. **Facial Characteristics**:
    - Method: Analyzing key facial features or using infrared for face thermograms correlated with the vascular system.
2. **Fingerprints**:
    - Unique across the human population.
    - Features extracted from the patterns of ridges and furrows on fingertips for automated recognition.
3. **Hand Geometry**:
    - Identifies features of the hand, including shape, and lengths and widths of fingers.
4. **Retinal Pattern**:
    - Unique pattern formed by veins beneath the retinal surface.

- Obtained via a digital image from projecting low-intensity light into the eye.

5. **Iris**:
    - Recognized by the detailed structure of the iris.
6. **Signature**:
    - Unique to each individual; reflects writing habits and physical attributes.
7. **Voice**:
    - Reflects the physical and anatomical characteristics of the speaker.

Operation:

- **Enrollment**: Individual's unique biometric characteristic is sensed, digitized, and stored as a template in the system.
- **Authentication**:
    - **Verification**: Compares the presented feature with the stored template for the entered PIN/user.
    - **Identification**: Compares the presented template with all stored templates to identify the user.

Biometric Accuracy:

- **False Match Rate (FMR)**: Frequency at which different source samples are assessed to be from the same source.
- **False Nonmatch Rate (FNMR)**: Frequency at which samples from the same source are assessed to be from different sources.
- **Threshold Value**: Determines when a match or mismatch is declared; influences FMR and FNMR.
- **Operating Characteristic Curve**: Plots FMR vs FNMR and can be used to compare the performance of different systems.

Trade-offs and Performance:

- There is a trade-off between security (FMR) and convenience (FNMR).

- A plausible trade-off is to pick a threshold where the rates are equal.
- A high-security application may require a very low FMR, resulting in decreased convenience.
- Iris biometrics exhibited no false matches in over 2 million cross-comparisons in tests.

Application and Costs:

- **Cost and Accuracy**: Different biometric measures vary in cost and accuracy, with hand geometry being less accurate and less costly, and iris and retina being more accurate and more costly.
- **Applications**: Applied in various specific areas but has yet to mature as a standard tool for user authentication to computer systems due to complexity and expense.