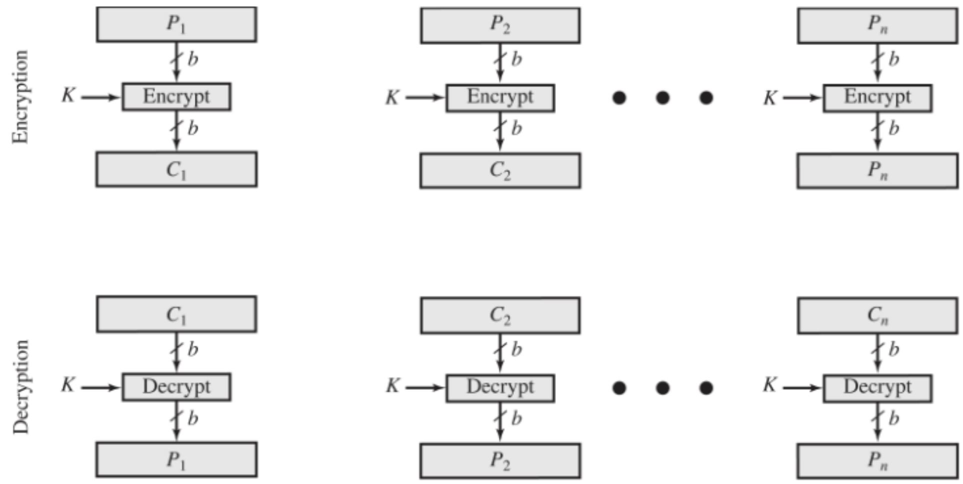
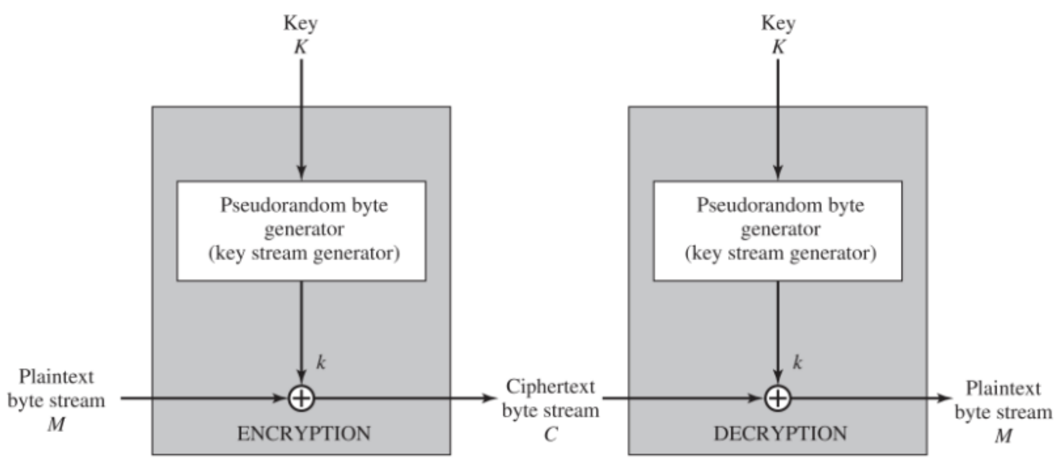
- Confidentiality with symmetric encryption
    - Symmetric encryption
        - strong algorithm that if an opponent who knows the algorithm and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key
        - Sender and receiver must have obtained copies of secret key in a secure fashion and must keep the key secure
    - Two approaches to attacking a symmetric encryption
        - cryptanalysis: relies on the nature of the algorithm plus perhaps knowledge of general characteristics of the plaintext, or even some sample plaintext-ciphertext pairs
        - brute-force attack: try every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained
    - Systemic block encryption algorithm:
        - Block cipher processes the plaintext input in fixed-size blocks and produces a block of ciphertext of equal size blocks
            - DES: Data Encryption Standard
                - Takes plaintext block of 64 bits and a key of 56 bits to produces a ciphertext block of 64 bits
            - Triple DES:
                - involves repeating the basic DES algorithm three times, using either two or three unique keys for a size of 112 or 168 bits
            - AES: Advanced Encryption Standard
                - must be a symmetric block cipher with a block length of 128 bits and support key lengths of 128, 192, and 256 bits
    - Practical safety issues:
        - Email messages, network packets, database records, and other plaintext resources must be broken up into a series of fixed-length block for encryption by a symmetric block cipher



(a) Block cipher encryption (electronic codebook mode)



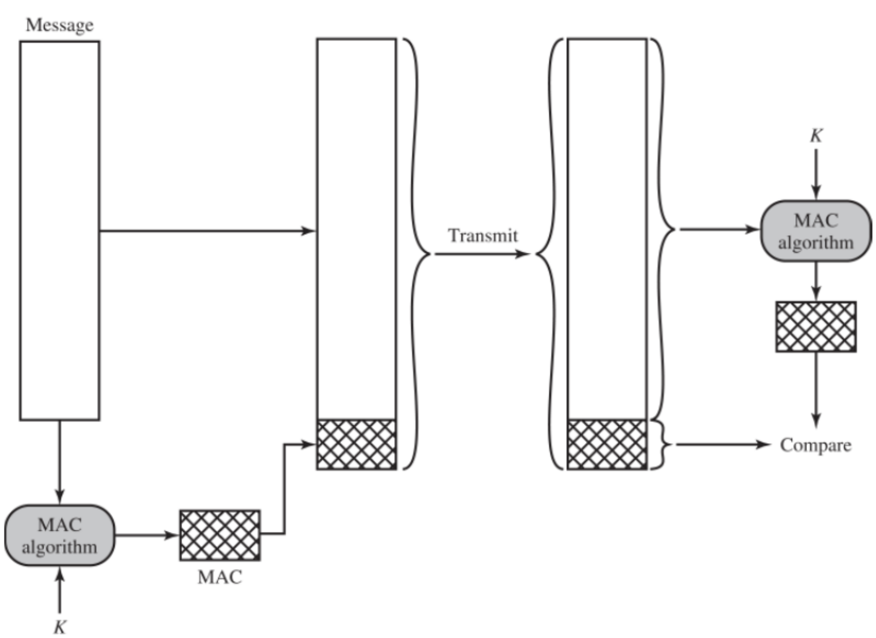
(b) Stream encryption

        - ECB may not be best practice for long messages
    - Stream Ciphers:
        - processes the input elements continuously, producing output one element at a time
        - key is input into pseudo random bit generator that produces a stream of 8-bit numbers, output, keystream, is combined one byte at a time with the plaintext stream using the bitwise exclusive OR (XOR) operation

2.2 Message Authentication and Hash Functions:

- Authentication Using Symmetric Encryption:
    - In ECB mode an attacker may alter the order of the blocks altering the meaning of the data
- Message Authentication without Message Encryption:

- Authentication tags may be generated and appended to each message in transmission
- Normally authentication and encryption are done separately and not apart of the same process
- 3 instances of when message confidentiality is preferable:
  - when messages are sent using a broadcast there can be an authentication tag provided, the responsible system performs the authentication and if there is a violation other destination systems are alerted
  - Authentication is carried out on a selective basis, with messages being chosen at random for checking. Usually with high volume loads
  - computer programs may also use authentication tags, if one were attached to the program, it could be checked whenever assurance is required of the integrity of the program
- Message Authentication with Code:
  - Small block of data that generates a secret key
    - two communicating parties must share a common secret key KAB
    - calculates message authentication code as complex function: MACM = F(KAB, M)
    - recipients perform the same calculation using the secret key to generate a new message authentication code



- Receiver assures message has not been altered, even if message was altered the code would not be and result in a different calculation
- Receiver assures that the message is from the alleged sender
- If the message includes a sequence number (like TCP), then the receiver can be assured of the proper sequence, because an attacker cannot successfully alter the sequence number

        - Authentication is less vulnerable to being broken than encryption because of mathematical properties
- One-way Hash Function:
    - Accepts a variable-size message M as input and produces a fixed-size message digest H(M) as output

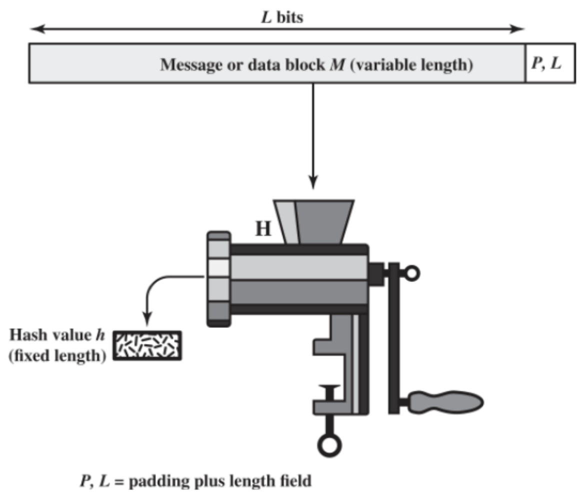


Figure 2.4 Cryptographic Hash Function; h=H(M)

    - Hash function does not take a secret key as input
    - Can be encrypted using symmetric encryption
    - can be encrypted using public-key encryption
    - public key has two advantages: digital signature and message authentication.

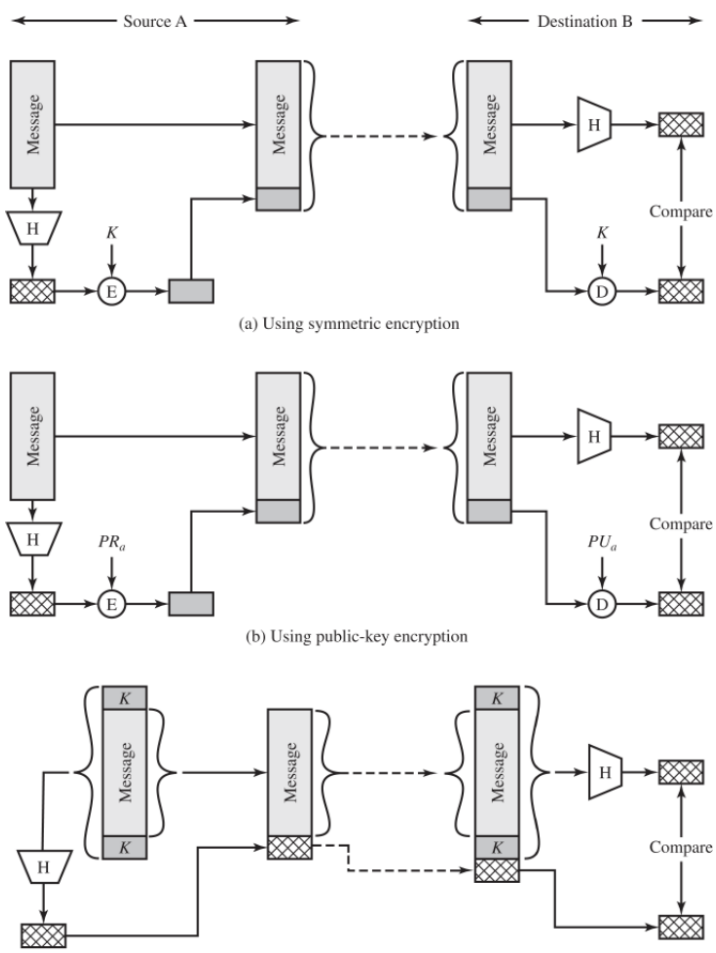


Figure 2.5 Message Authentication Using a One-Way Hash Function

    - Sometimes encryption is not necessarily the best method, for example it is slow software
    - encryption hardware is not cheap
    - hardware is optimized toward large data sizes
    - encryption algorithm may be protected by a patent

    - c. assumes that both parties share a secret key, which is incorporated into the process of generating a hash code

- Secure Hash Functions:
    - Hash function requirements:
        - H can be applied to a block of data of any size
        - H produces a fixed-length output
        - H(x) is relatively easy to compute for any given x
        - For any given code h, it is computationally infeasible to find x, such that H(x) = x (one way hash)
            - generates a code given a message, but virtually impossible to generate a massage given a code
        - For any given block x, it is computationally infeasible to find y != x with H(y) = H(x) (weak collision resistant)
            - guarantees that it is impossible to find an alternative message with the same hash value
        - Computationally infeasible to find any pair (x, y) such that H(y) = H(x) (collision restraint)
- Security of Hash Functions:
    - Strength of hash depends on length of the hash code produced by the algorithm
        - Preimage resistant: 2n
        - Second preimage resistant: 2n
        - Collision Resistant: 2n/2
- Secure Hash Function Algorithms:
    - SHA (Secure Hash Algorithm)
- Other Applications of Hash Functions:
    - Passwords: when a user enters a password the hash of that password is compared to the stored hash value for verification
    - intrusion detection: store the hash value for a file for each file on a system and secure the hash values

2.3 Public-Key Encryption:

- Public Encryption Structure
    - Public-key algorithms are based on mathematical functions rather than on simple operations on bit patterns
    - asymmetric: involving the use of two separate keys, in contrast to symmetric encryption, which uses only one key
    - security of any encryption depends on two things:
        - length of key
        - computational work involved in breaking the cipher

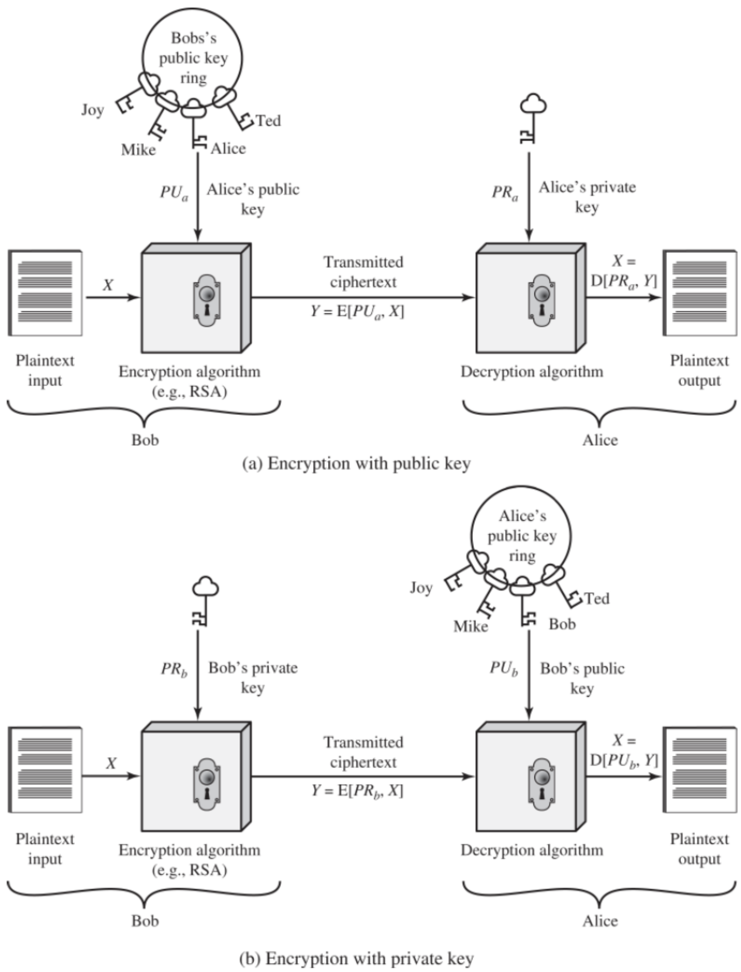


Figure 2.6 Public-Key Cryptography

- Plaintext: readable message or data that is fed into the algorithm as input
- Encryption algorithm: performs various transformations on the plaintext
- public and private key: selected keys so that if one is used for encryption, the other is used for decryption
- ciphertext: scrambled message produced as output. depends on the plaintext and key
- Decryption algorithm: accepts the ciphertext and matching key and produces the original plaintext

- Public-key algorithms rely on:
  - Each user generates a pair of keys to be used for encryption and decryption
  - each user places one of the two keys in a public register or another accessible file and the other is kept private
  - if bob wants to send a private message to alice, then bob encrypts the message using alice's public key
  - when alice receives the message, she decrypts it using her private key, no other recipient can decrypt
- Applications For Public Key Cryptosystems:
  - We classify the use of public-key cryptosystems into three categories: digital signature, symmetric key distribution, and encryption of secret keys

Table 2.3 Applications for Public-Key Cryptosystems

Algorithm	Digital Signature	Symmetric Key Distribution	Encryption of Secret Keys
RSA	Yes	Yes	Yes
Diffie–Hellman	No	Yes	No
DSS	Yes	No	No
Elliptic Curve	Yes	Yes	Yes

- Asymmetric Encryption Algorithms
    - RSA:
        - most widely accepted and implemented approach to public-key encryption
        - block cipher in which the plaintext and ciphertext are integers between 0 and n-1 for some n
    - Diffie-Hellman Key Agreement
        - enables two users to securely reach agreement about a shared secret that can be used as a secret key for subsequent symmetric encryption
    - Digital Signature Standard:
        - makes use of the SHA-1 and presents a new digital signature technique, the DSA (digital signature algorithm).
        - uses an algorithm that is designed to provide only the digital signature function
        - cannot be used for encryption or key exchange
    - Elliptic Curve Cryptography:
        - Appears to offer equal security to RSA for a smaller bit size, reducing processing overhead

2.4 Digital Signatures and Key Management

- Three distinct aspects to use of public-key encryption:
    - secure distribution of public keys
    - use of public-key encryption to distribute secret keys

    - use of public-key encryption to create temporary keys for message encryption
- Digital Signature:
    - known as the result of a cryptographic transformation of data that, when properly implemented, provides a mechanism for verifying origin authentication, data integrity, and signatory non-repudiation (data-dependent bit pattern)
- DSA (Data Signature Algorithm) The original NIST-approved algorithm, which is based on the difficulty of computing discrete logarithms
- RSA Digital Signature Algorithm: based on RSA public-key algorithm
- Elliptic Curve Digital Signature: based on elliptic-curve cryptography

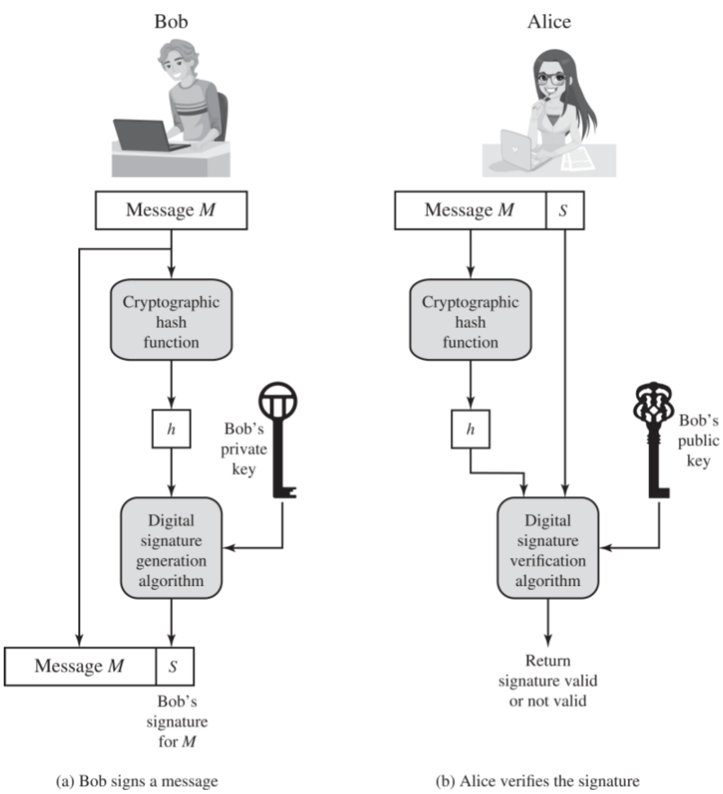


Figure 2.7 Simplified Depiction of Essential Elements of Digital Signature Process

- digital signatures do not provide confidentiality, for example they can sometimes be eavesdropped

- Public-Key Certificates:
    - A major weakness is that anyone can forge such a public announcement, meaning some users could pretend to be Bob and send a public key to another participant or broadcast such a public key
    - A certificate consists of a public key plus a user ID of the owner, with the whole block signed by a trusted third party

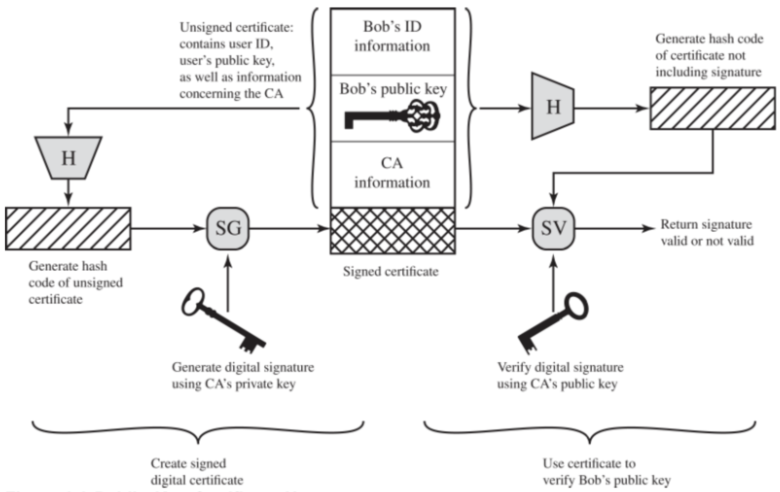


Figure 2.8 Public-Key Certificate Use

The key steps can be summarized as follows:

- key steps:
    - user software (client) creates a pair of keys: one public and another private
    - client prepares an unsigned certificate that includes the user ID and user's public key
    - user provides the unsigned cert to a CA in some secure manner
    - CA creates a signature as follows:
        - CA uses a hash function to calculate the hash code of the unsigned certificate
        - CA generates digital signature using the CA's private key and a signature generation algorithm
    - CA attaches the signature to the unsigned cert to create a signed cert
    - CA returns the signed cert to client
    - Client may provide the signed certificate to any other user
    - Any user may verify that the cer is valid as follows:
        - user calculates the hash code of certificate (not including signature)
        - user verifies digital signature using CA's public key and the signature verification algorithm
- Digital Envelopes
    - Digital envelopes can be used to protect a message without needing to first arrange for sender and receiver to have the same secret key

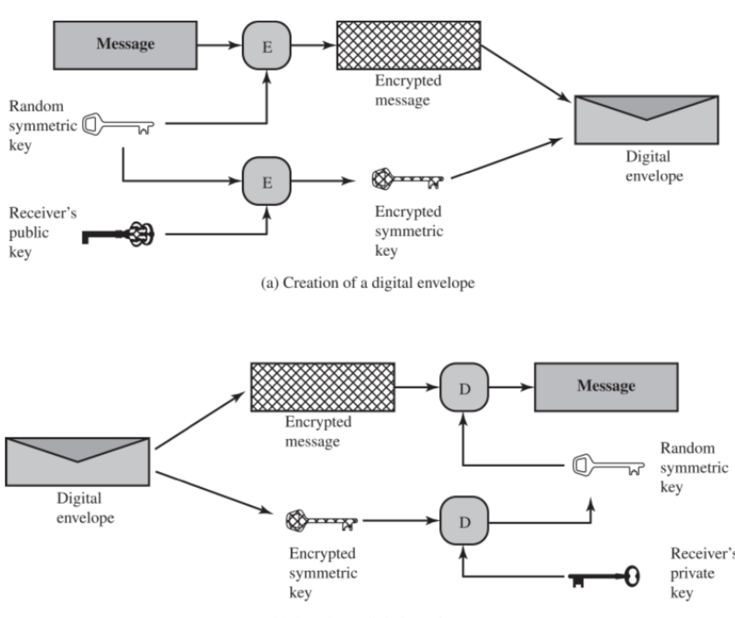


Figure 2.9 Digital Envelopes

- Prepare a message
- generate a random symmetric key that will be used this one time only
- encrypt that message using symmetric encryption the one-time key
- encrypt the one-time key using public-key encryption with Alice's public key
- Attach the encrypted one-time key to the encrypted message and send it to Alice

2.5 Random and Pseudorandom Numbers

- Use of random numbers:
    - Generation of keys for RSA
    - Generation of a stream key for symmetric stream cipher
    - Generation of a symmetric key for use as a temporary session key
    - in a number of key distribution scenarios
    - session key generation, whether done by a key distribution center or by one of the principals
- Randomness:
    - Validating random numbers
        - uniform distribution: the distribution of numbers in the sequence should be uniform, the frequency of occurrence of each of the numbers should be approximately the same
        - independence: no one value in the sequence can be inferred from the others
    - If a problem is too hard or time-consuming to solve exactly, a simpler, shorter approach based on randomization is used to provide an answer with any desired level of confidence
- Unpredictability:
    - With “true” random sequences, each number is statistically independent of other numbers in the sequence and therefore unpredictable
- Random Versus Pseudorandom:
    - algorithms are deterministic and therefore produce sequences of numbers that are not statistically random
    - True random number generator uses a nondeterministic source to produce randomness

2.6 Practical Application: Encryption of Stored Data

- It’s best practice to encrypt even data that is at rest combined with encryption key management
- Back-end appliance: hardware that sits between servers and storage systems and encrypt all data going from the server to the storage system, and decrypts in opposite direction
- Library based tape encryption: co-processor board embedded in the tape drive and tape library hardware, co-processor encrypts data using a non readable key configured into the board, key can be exported via secure email, or a small flash drive that is transported securely
- Background laptop and PC data encryption: some programs encrypt all or designated files and folders