# Announcements

- First Exam Opens Thursday
  - Will have until Sunday at midnight to take it
  - Covers Chapters 1 and 2
  - Take the practice quizzes
- DETER Invitation
  - Should have received an email from DETER
  - Have 72 hours to accept before the registration expires
- Programming Assignment 1:
  - Due Feb 9
    - START NOW!!
  - Requires C/C++
  - Posted helpful links for those who need them in the general teams
  - If you struggle with C/C++, please attend TA office hours to get the help you need
    - The purpose of this assignment is to get hands on experience with encryption NOT getting bogged down in C/C++ so get help early if you're not comfortable with the language.

# Elliptic Curve Cryptography

- Operates on Elliptic Curve mathematical equations
- Utilizes a set of points to form ECC operations
- Each key corresponds to a point on the elliptic curve
    - A private key (random number)
    - A public key (derived from the private key)
        - Via scalar multiplication AKA a series of dot products
- Public Key Generation
    - Perform repeated additions (dot products) of a point on the elliptic curve a certain number of times
        - The number of repetitions is determined by the private key
            - Via the binary representation => the number of bits set to 1 is the number of repeated additions
            - If K = 13 = 1101 => the public key will be derived with 3 additions

# Elliptic Curve Cryptography

- Advantages
  - Efficiency
    - Shorter key lengths
  - Resistance to Quantum Attacks
  - Scalability
    - Widely used in modern cryptographic protocols (TLS)
    - Suitable for resource-constrained environments (like IOT)

# MESSAGE AUTHENTICATION

# Message Authentication

**protects against active attacks**

**verifies received message is authentic**

- **contents have not been altered**
- **from authentic source**
- **timely and in correct sequence**

**can use conventional encryption**

- **only sender & receiver share a key**

# Message Authentication Codes

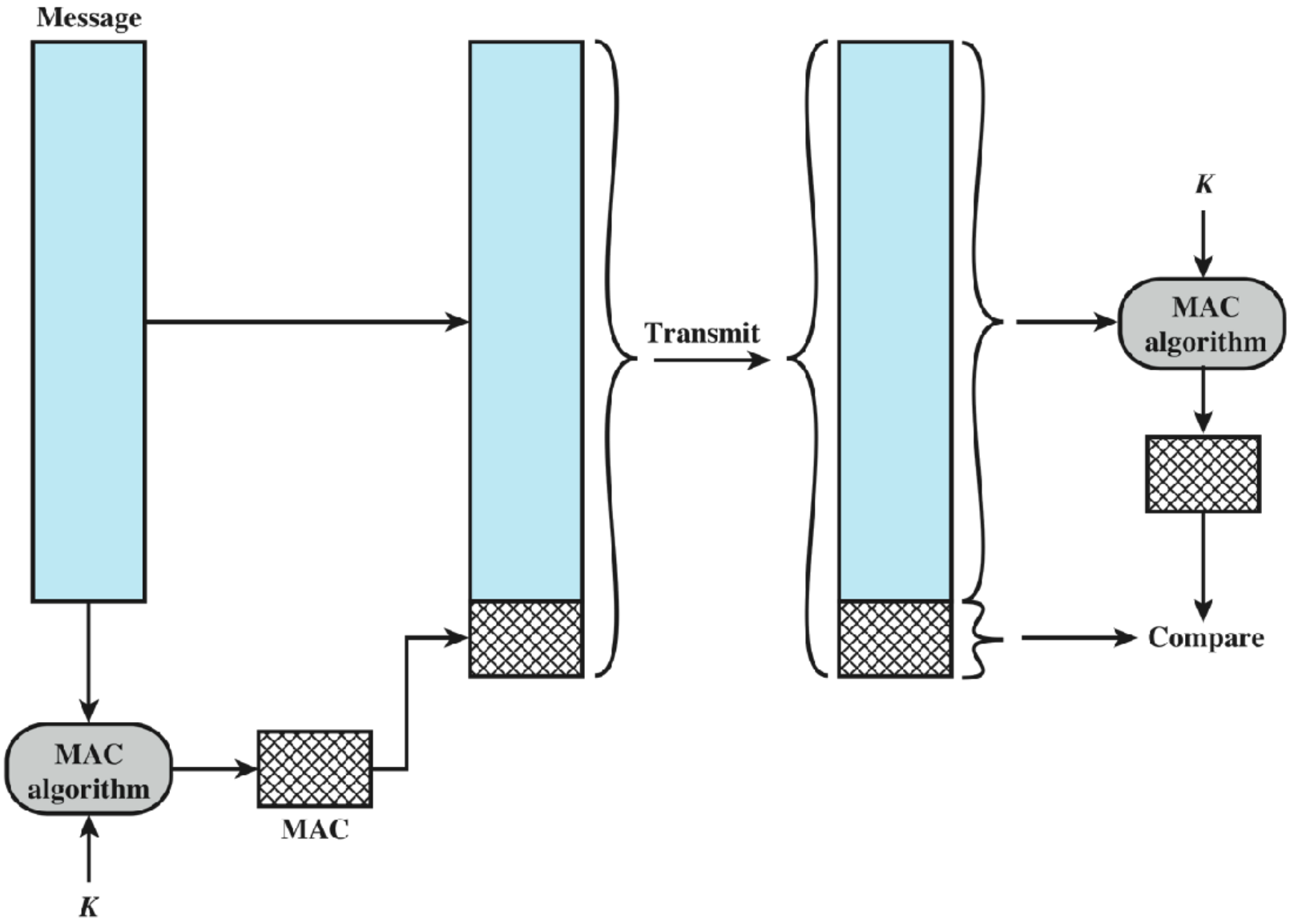



Figure 2.4 Message Authentication Using a Message Authentication Code (MAC). The MAC is a function of an input message and a secret key.

# Secure Hash Functions

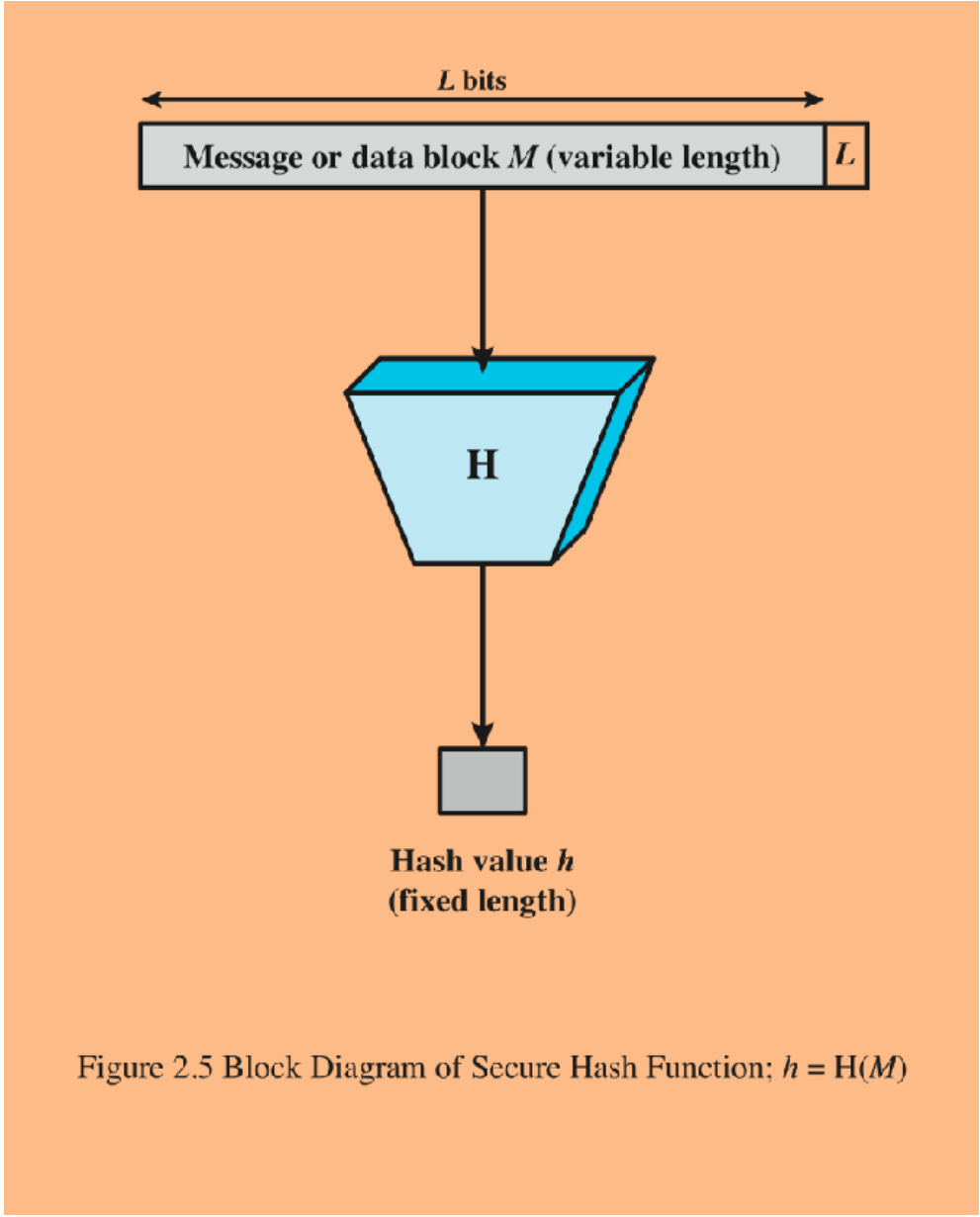



Figure 2.5 Block Diagram of Secure Hash Function; $h = H(M)$

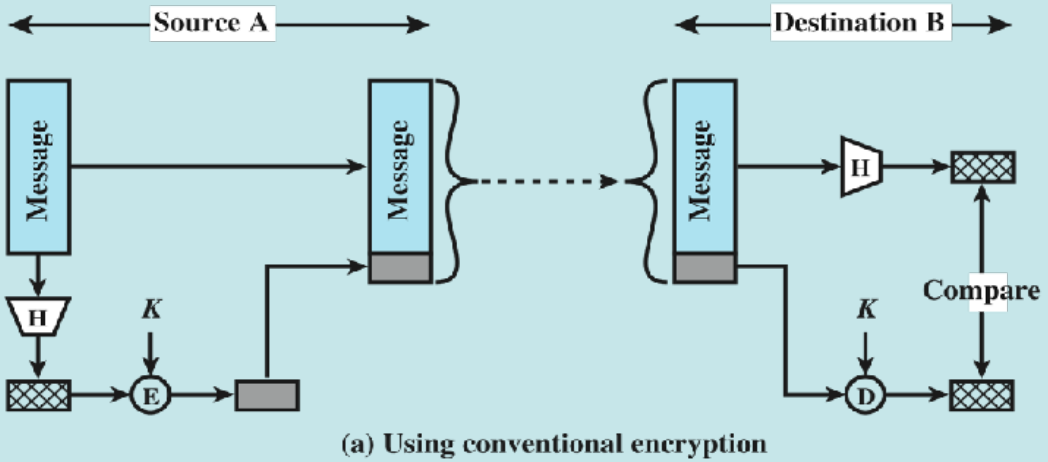
# Hash Function Requirements

- can be applied to a block of data of any size
- produces a fixed-length output
- H($x$) is relatively easy to compute for any given $x$
- one-way or pre-image resistant
  - computationally infeasible to find $x$ such that H($x$) = $h$
- second pre-image resistant or weak collision resistant
  - computationally infeasible to find $y \neq x$ such that H($y$) = H($x$)
- collision resistant or strong collision resistance
  - computationally infeasible to find any pair ($x$, $y$) such that H($x$) = H($y$)

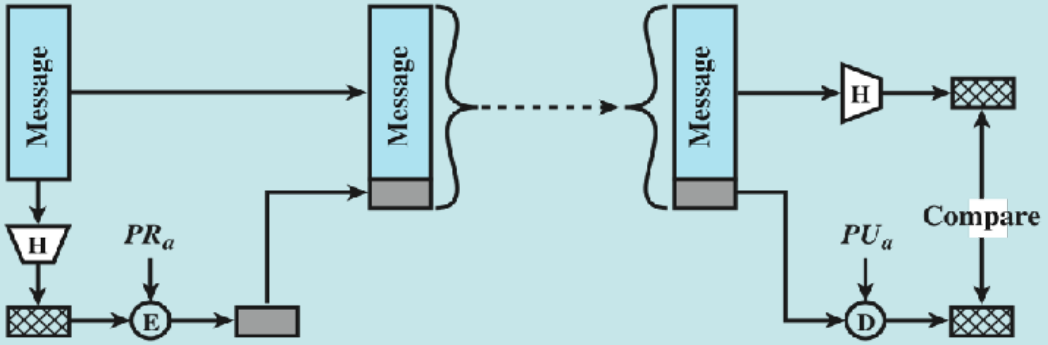
# Security of Hash Functions

- there are two approaches to attacking a secure hash function:
  - cryptanalysis
    - exploit logical weaknesses in the algorithm
  - brute-force attack
    - strength of hash function depends solely on the length of the hash code produced by the algorithm
- SHA most widely used hash algorithm
- additional secure hash function applications:
  - passwords
    - hash of a password is stored by an operating system
  - intrusion detection
    - store H(F) for each file on a system and secure the hash values

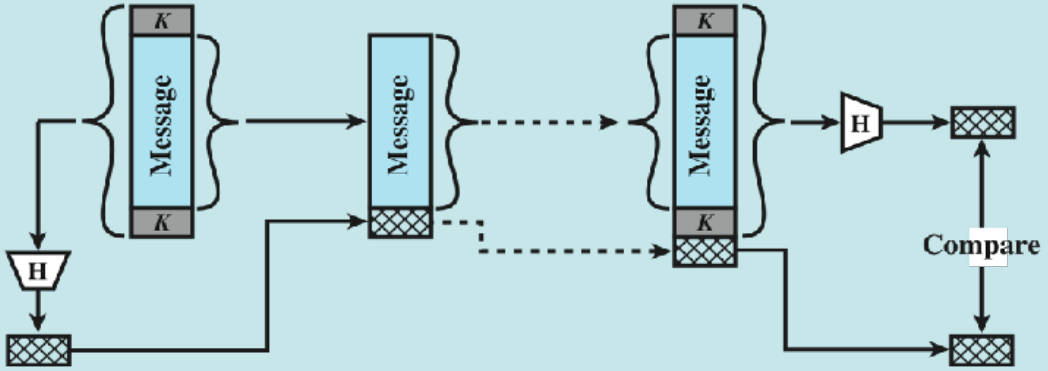
# Message Authentication Using a One-Way Hash Function




(a) Using conventional encryption




(b) Using public-key encryption

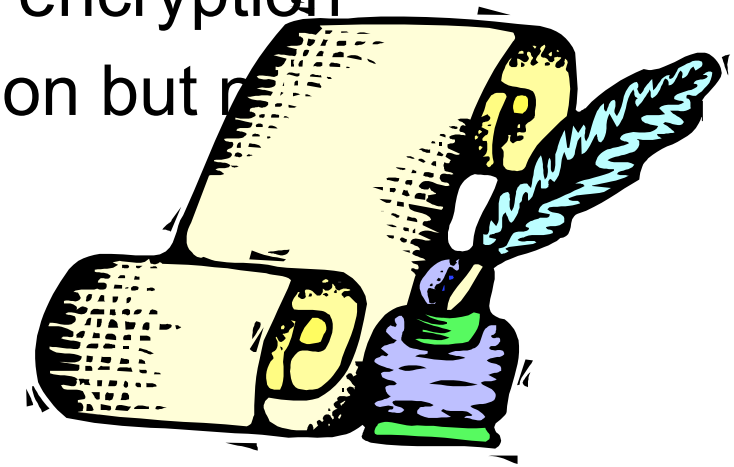



(c) Using secret value

Figure 2.6 Message Authentication Using a One-Way Hash Function. The hash function maps a message into a relatively small, fixed-size block.

# Digital Signatures

- used for authenticating both source and data integrity
- created by encrypting hash code with private key
- does not provide confidentiality
  - even in the case of complete encryption
  - message is safe from alteration but n
    eavesdropping



# Public Key Certificates

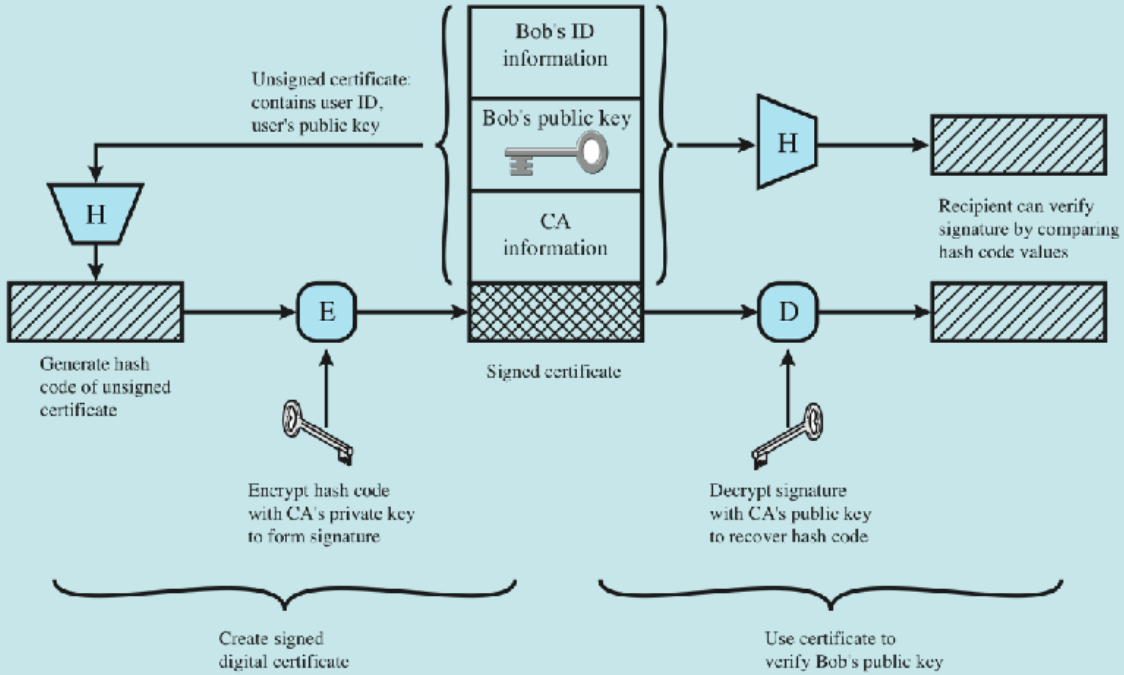
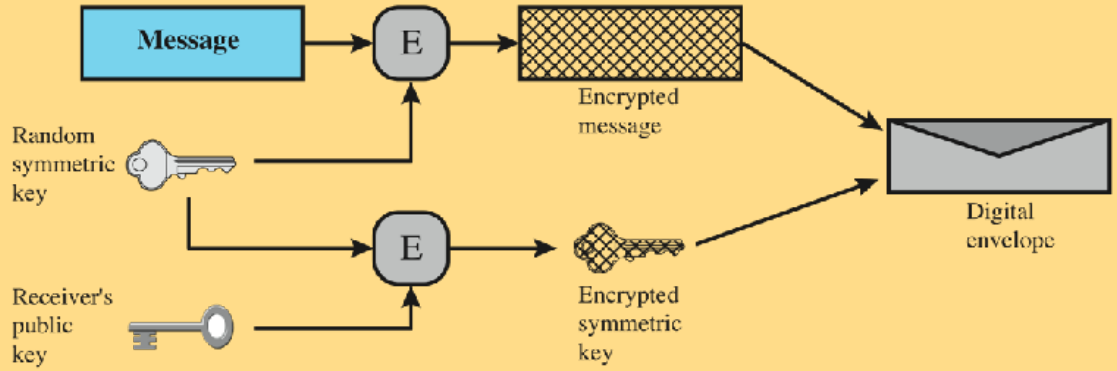


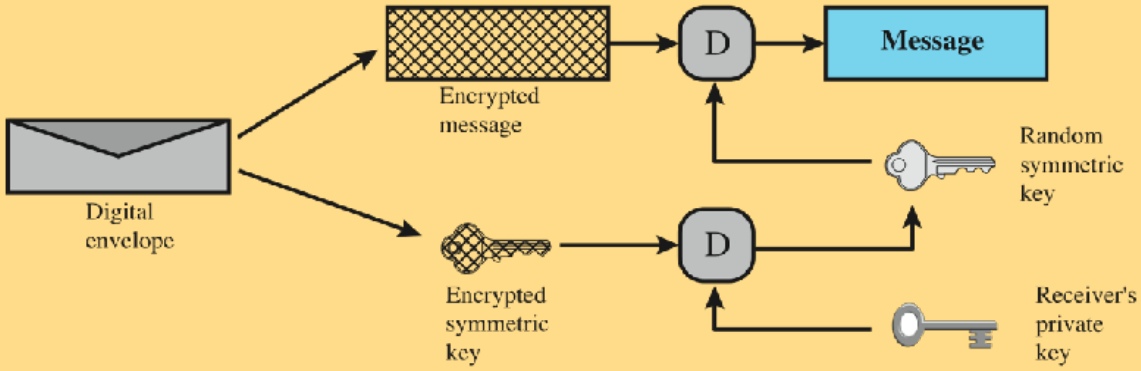

**Figure 2.8 Public-Key Certificate Use**

# Digital Envelopes

- **protects a message without needing to first arrange for sender and receiver to have the same secret key**
- ***equates to the same thing as a sealed envelope containing an unsigned letter**



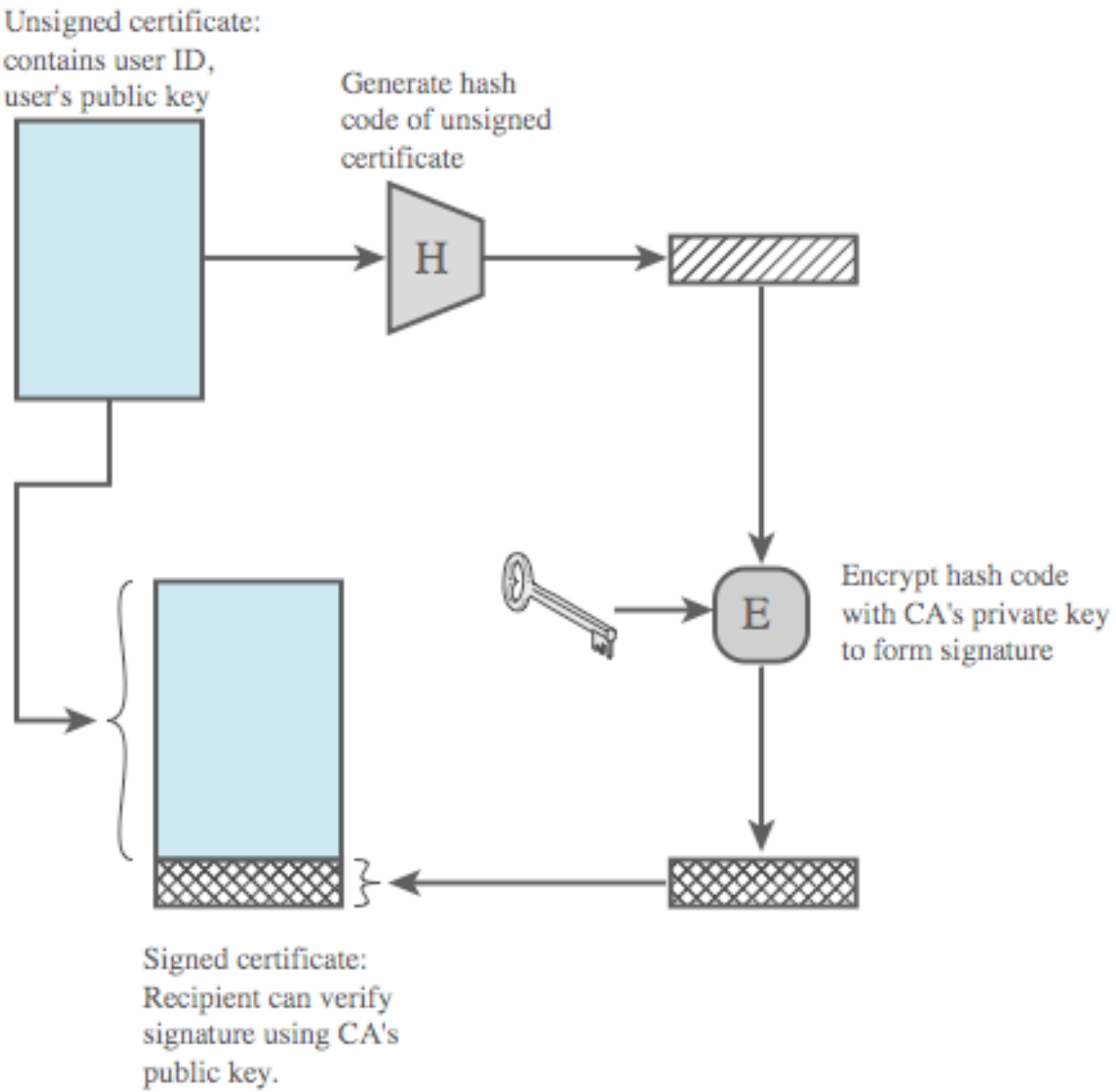

(a) Creation of a digital envelope




(b) Opening a digital envelope

**Figure 2.9 Digital Envelopes**

# Public Key Certificates



Page Info — https://www.oracle.com/index.html

General Media Permissions Security

**Website Identity**

Website: www.oracle.com

Owner: This website does not supply ownership information.

Verified by: DigiCert Inc View Certificate

**Privacy & History**

Have I visited this website prior to today? No

Is this website storing information on my computer? Yes, cookies and 64.3 KB of site data Clear Cookies and Site Data

Have I saved any passwords for this website? No View Saved Passwords

**Technical Details**

Connection Encrypted (TLS_AES_256_GCM_SHA384, 256 bit keys, TLS 1.3)

The page you are viewing was encrypted before being transmitted over the Internet.

Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.

?

# RANDOM NUMBERS

# Random Numbers

- **Uses include generation of:**
  - keys for public-key algorithms
  - stream key for symmetric stream cipher
  - symmetric key for use as a temporary session key or in creating a digital envelope
  - handshaking to prevent replay attacks
  - session key

# Random Numbers

- random numbers have a range of uses
- requirements:
  - randomness
    - based on statistical tests for uniform distribution and independence
  - unpredictability
    - successive values not related to previous
    - clearly true for truly random numbers
    - but more commonly we use PRNG generator

# Pseudorandom versus Random Numbers

- often use algorithmic technique to create pseudorandom numbers
  - which satisfy statistical randomness tests
  - but likely to be predictable
- true random number generators use a nondeterministic source
  - e.g. radiation, gas discharge, leaky capacitors
  - increasingly provided on modern processors

# Simple pseudo-random number generator (PRNG)

```
seed = constant  //* if you want to repeat an experiment
seed = timestamp // if you want different results for each experiment
```

$x_0 = \text{given}, \qquad x_{n+1} = P_1 x_n + P_2 \pmod N \quad n = 0,1,2,... \quad (*)$

$x_0 = 79, N = 100, P_1 = 263, \text{ and } P_2 = 71$

Then

$x_1 = 79*263 + 71 \pmod{100} = 20848 \pmod{100} = 48,$

$x_1 = 48*263 + 71 \pmod{100} = 12695 \pmod{100} = 95,$

$x_1 = 95*263 + 71 \pmod{100} = 25056 \pmod{100} = 56,$

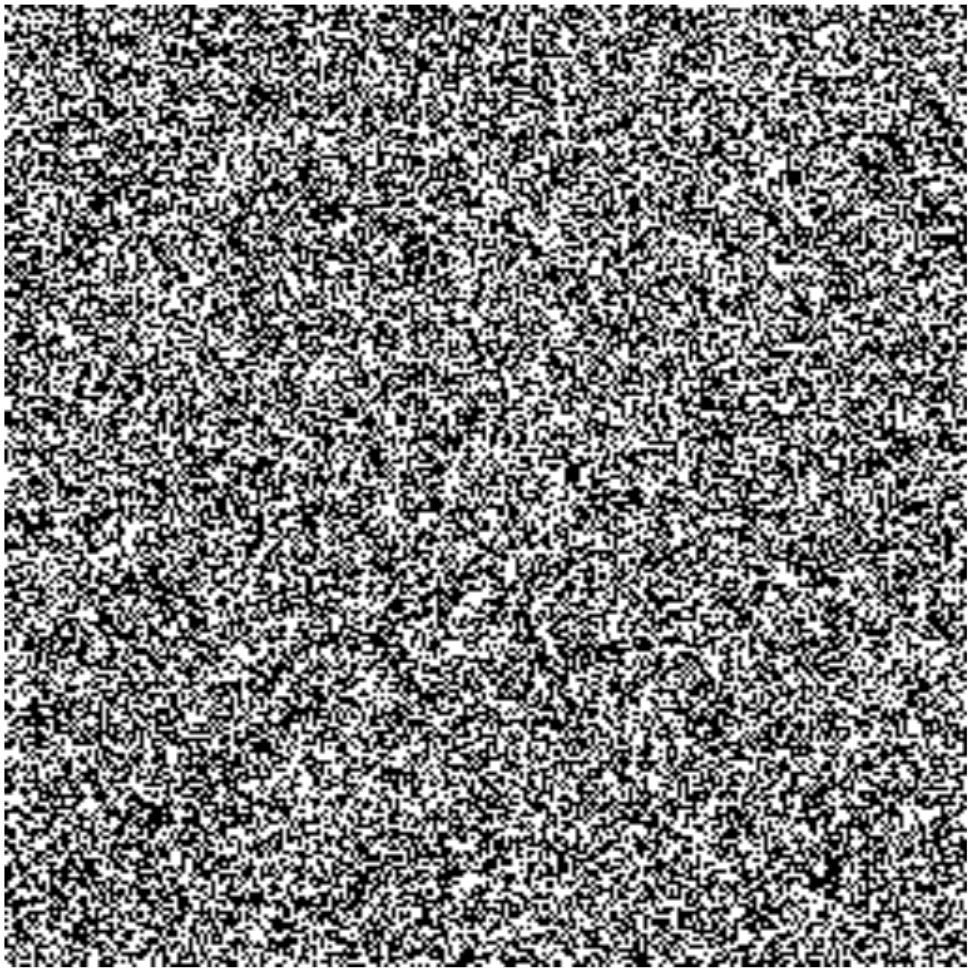
$x_1 = 56*263 + 71 \pmod{100} = 14799 \pmod{100} = 99,$

Subsequent numbers are: **8, 75, 96, 68, 36, 39, 28, 35, 76, 59, 88, 15, 16, 79, 48.** The sequence then **repeats** .

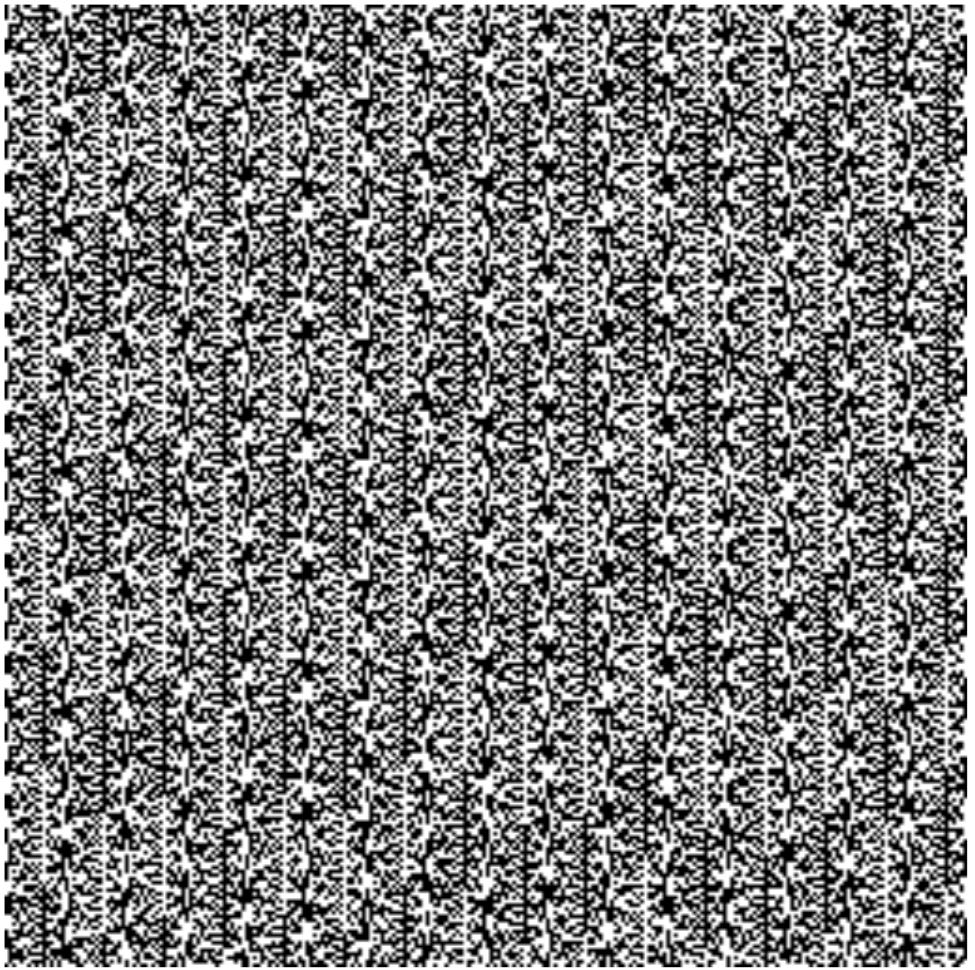
**$P_1 = 16807$, $P_2 = 0$, and $N = 2^{31} - 1 = 2147483647$. (a better random number generator)**

# Randomness Tests

# Randomness Tests

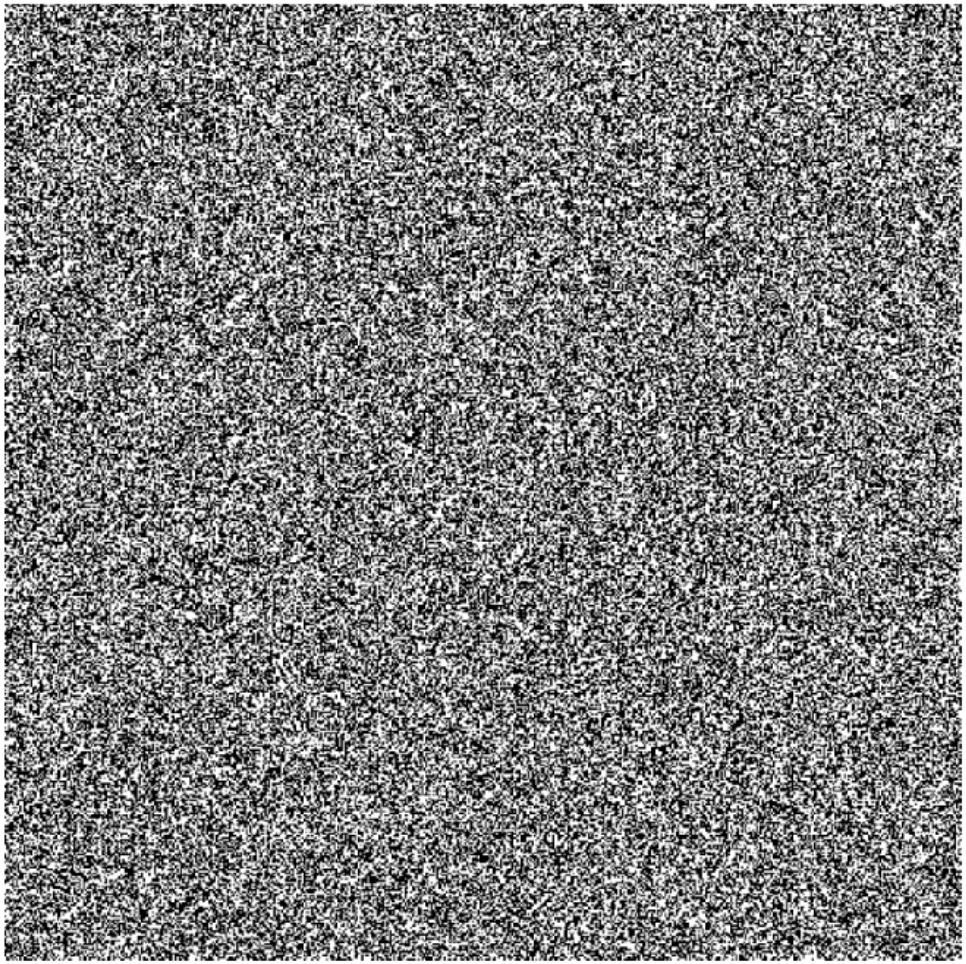


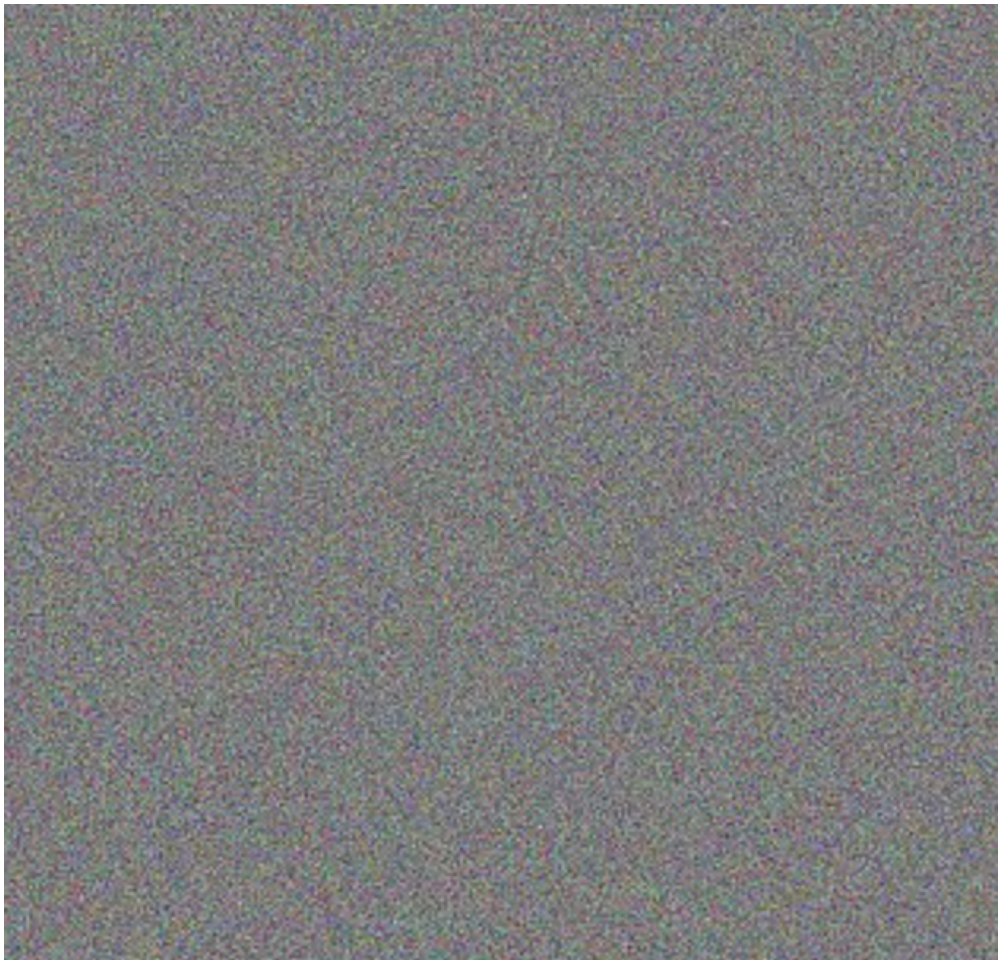
RANDOM.ORG



PHP rand() on Microsoft Windows

# Which one shows better randomness?





# Summary

- introduced cryptographic algorithms
- symmetric encryption algorithms for confidentiality
- message authentication & hash functions
- public-key encryption
- digital signatures and key management
- random numbers

# Exam Recap

- Introduction to Computer Security
  - What is Computer Security
  - Terminology
    - What is the CIA triad and how it applies to security concepts
    - AAA and how it applies to security concepts
    - Kinds of attacks
      - Passive vs Active
  - Functional Requirements
  - Strategy

# Exam Recap

- Cryptography
  - Symmetric and Asymmetric encryption
  - Public Key Generation
  - Digital signatures
  - Key management
  - Block and Stream Ciphers
    - ECB vs CBC
  - Algorithms
    - DES, 3DES, AES
  - Diffie Helman Key Exchange and RSA
  - Prime Numbers
    - Importance and use
  - Applications
- Message Authentication
- Random Numbers

# CS 356 – Lecture 4
# User Authentication

Spring 2024