# CHAPTER 4
# PUBLIC KEY CRYPTO

PREPARED BY:

DR. MUHAMMAD IQBAL HOSSAIN

ASSISTANT PROFESSOR

DEPARTMENT OF CSE, BRAC UNIVERSITY

# APPENDIX

DIFFIE-HELLMAN KEY EXCHANGE

ELLIPTIC CURVE CRYPTOGRAPHY

USES FOR PUBLIC KEY CRYPTO

INFORMATION HIDING

STEGANOGRAPHY

HASH FUNCTION

# Diffie-Hellman Key exchange

# DIFFIE-HELLMAN

- Invented by Williamson (GCHQ) and, independently, by D and H (Stanford)
- A "key exchange" algorithm
  - Used to establish a shared symmetric key
- **Not for encrypting or signing**

- Security rests on difficulty of **discrete log problem:** (Not known: NP-complete)

  given $g$, $p$, and $g^k \bmod p \rightarrow$ find $k$

# DIFFIE-HELLMAN

- Let p be prime, let g be a **generator** (p,g are public)
  - For any $x \in \{1,2,\ldots,p-1\}$ there is n s.t. $x = g^n \bmod p$
- Alice selects secret value a
- Bob selects secret value b
- Alice sends $g^a \bmod p$ to Bob
- Bob sends $g^b \bmod p$ to Alice
- Both compute shared secret $g^{ab} \bmod p$
  - $(g^b)^a = g^{ba} = g^{ab} \bmod p$
- Shared secret can be used as symmetric key

# DIFFIE-HELLMAN

- Suppose that Bob and Alice use $g^{ab} \bmod p$ as a symmetric key
- Trudy can see $g^a \bmod p$ and $g^b \bmod p$
- Note $g^a g^b \bmod p = g^{a+b} \bmod p \neq g^{ab} \bmod p$
- If Trudy can find a or b, system is broken
- If Trudy can solve **discrete log** problem, then she can find a or b

# DIFFIE-HELLMAN

- **Public:** g and p
- **Secret:** Alice's exponent a, Bob's exponent b

$g^a \bmod p$ →

← $g^b \bmod p$

Alice, a

Bob, b

- Alice computes $(g^b)^a = g^{ba} = g^{ab} \bmod p$
- Bob computes $(g^a)^b = g^{ab} \bmod p$
- Could use $K = g^{ab} \bmod p$ as symmetric key

# DIFFIE–HELLMAN KEY EXCHANGE: EXAMPLE

Domain parameters $p$=29, $g$=2

**Alice**

Choose random private key
$a = 5$

Compute corresponding public key
$A = g^a = 2^5 = 3 \bmod 29$

$\xrightarrow{A}$

**Bob**

Choose random private key
$b = 12$

$\xleftarrow{B}$

Compute correspondig public key
$B = g^b = 2^{12} = 7 \bmod 29$

**Alice**

Compute common secret
$k_{AB} = B^a = g^{ba} = 7^5 = 16 \bmod 29$

**Bob**

Compute common secret
$k_{AB} = A^b = g^{ab} = 3^{12} = 16 \bmod 29$

Proof of correctness:

*Alice computes:* $B^a = (g^b)^a \bmod p$
*Bob computes:* $A^b = (g^a)^b \bmod p$

*i.e., Alice and Bob compute the same key* $k_{AB}$!

| Alice | | Bob |
| --- | --- | --- |
| Choose random private key $a \in \{1,2,\ldots,p\text{-}1\}$ | | Choose random private key $b \in \{1,2,\ldots,p\text{-}1\}$ |
| Compute corresponding public key $A = \alpha^a \bmod p$ | $\xrightarrow{A}$ | |
| | $\xleftarrow{B}$ | Compute correspondig public key $B = \alpha^b \bmod p$ |
| Compute common secret $k_{AB} = B^a = (g^a)^b \bmod p$ | | Compute common secret $k_{AB} = A^b = (g^b)^a \bmod p$ |

---

We can now use the joint key $k_{AB}$ for encryption, e.g., with AES

$$y = AES_{kAB}(x) \xrightarrow{y} x = AES^{-1}{}_{kAB}(y)$$

# DIFFIE-HELLMAN

- Subject to **man-in-the-middle (MiM)** attack

$g^a \bmod p$ →

← $g^t \bmod p$

$g^t \bmod p$ →

← $g^b \bmod p$

Alice, a

Trudy, t

Bob, b

- Trudy shares secret $g^{at} \bmod p$ with Alice
- Trudy shares secret $g^{bt} \bmod p$ with Bob
- *Alice and Bob don't know Trudy exists!*

# DIFFIE-HELLMAN

- How to prevent MiM attack?
- Solutions
  1. Encrypt DH exchange with symmetric key
  2. Encrypt DH exchange with public key
  3. Sign DH values with private key
  4. Other?
- You **MUST** be aware of MiM attack on Diffie-Hellman

# ECC:
# ELLIPTIC CURVE CRYPTOGRAPHY

# ELLIPTIC CURVE CRYPTO (ECC)

- “Elliptic curve” is **not** a cryptosystem
  - Elliptic curves are a different way to do the math in public key system
- Elliptic curve versions of DH, RSA, etc.
- Elliptic curves may be more efficient
  - Fewer bits needed for same security
  - But the operations are more complex

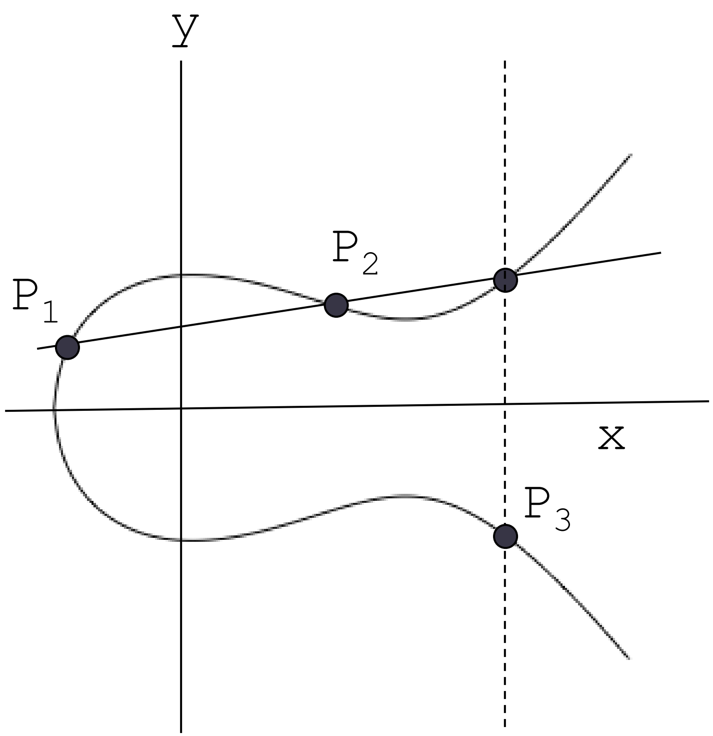
# WHAT IS AN ELLIPTIC CURVE?

- An elliptic curve E is the graph of an equation of the form

  $$\mathbf{y^2 = x^3 + ax + b}$$

- Also includes a "point at infinity" : $\infty$
- What do elliptic curves look like?
- See the next slide!

# ELLIPTIC CURVE PICTURE




- Consider elliptic curve

  $E$: $y^2 = x^3 - x + 1$

- If $P_1$ and $P_2$ are on E, we can define

  $P_3 = P_1 + P_2$

  as shown in picture

- **Addition is all we need**
- For discrete points, we add "mod p" to the EC

# KEY SIZE COMPARISON

| *Symmetric* | *ECC* | *RSA, DL* | *Remark* |
|---|---|---|---|
| 64 Bit | 128 Bit | ≈ 700 Bit | Only short term security (a few hours or days) |
| 80 Bit | 160 Bit | ≈ 1024 Bit | Medium security (except attacks from big governmental institutions etc.) |
| 128 Bit | 256 Bit | ≈ 3072 Bit | Long term security (without quantum computers) |

# USES FOR PUBLIC KEY CRYPTO

# USES FOR PUBLIC KEY CRYPTO

- Confidentiality
  - Transmitting data over insecure channel
  - Secure storage on insecure media
- Authentication (later)
- Digital signature provides integrity and **non-repudiation**
  - **No** non-repudiation with symmetric keys
  - Who has the secret key is the key for non-repudiation.

# NON-NON-REPUDIATION

- Alice orders 100 shares of stock from Bob
- Alice computes **MAC** using symmetric key
- Stock drops, Alice claims she did not order
- Can Bob prove that Alice placed the order?
- **No!** Since Bob also knows symmetric key, he could have forged message
- **Problem:** Bob knows Alice placed the order, but he can't prove it

# NON-REPUDIATION

- Alice orders 100 shares of stock from Bob
- Alice **signs** order with her private key
- Stock drops, Alice claims she did not order
- Can Bob prove that Alice placed the order?
- **Yes!** Only someone with Alice's private key could have signed the order
- This assumes Alice's private key is not stolen (revocation problem)

# PUBLIC KEY NOTATION

- **Sign** message M with Alice's **private key:** $[M]_{Alice}$
- **Encrypt** message M with Alice's **public key:** $\{M\}_{Alice}$
- Then

$$\{[M]_{Alice}\}_{Alice} = M$$ Sign than Encrypt

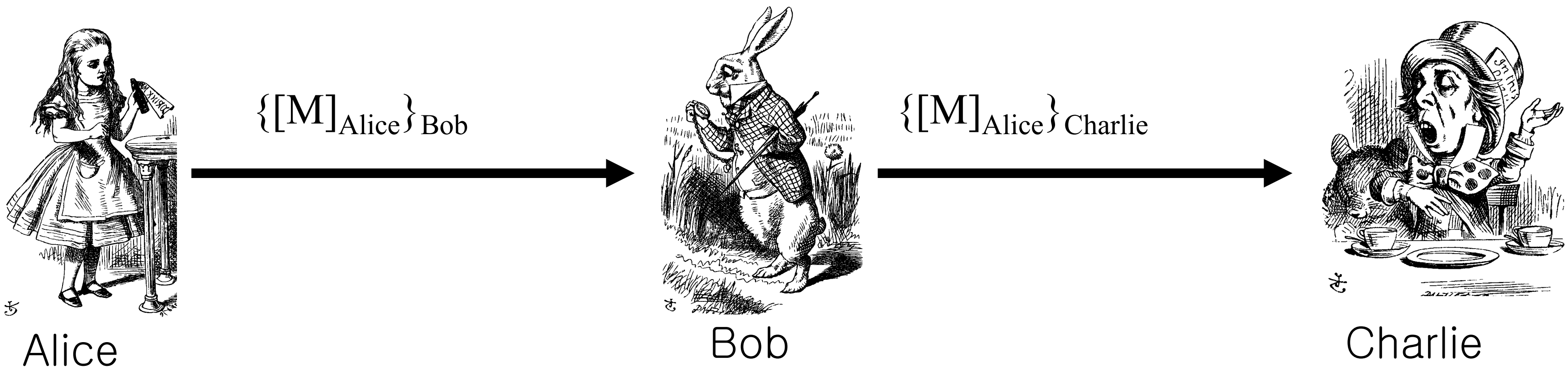
$$[\{M\}_{Alice}]_{Alice} = M$$ Encrypt than Sign

# CONFIDENTIALITY AND NON-REPUDIATION?

- Suppose that we want confidentiality and integrity/non-repudiation
- Can public key crypto achieve both?
- Alice sends message to Bob
  - **Sign and encrypt:** $\{[M]_{Alice}\}_{Bob}$
  - **Encrypt and sign:** $[\{M\}_{Bob}]_{Alice}$
- Can the order possibly matter?

# SIGN AND ENCRYPT

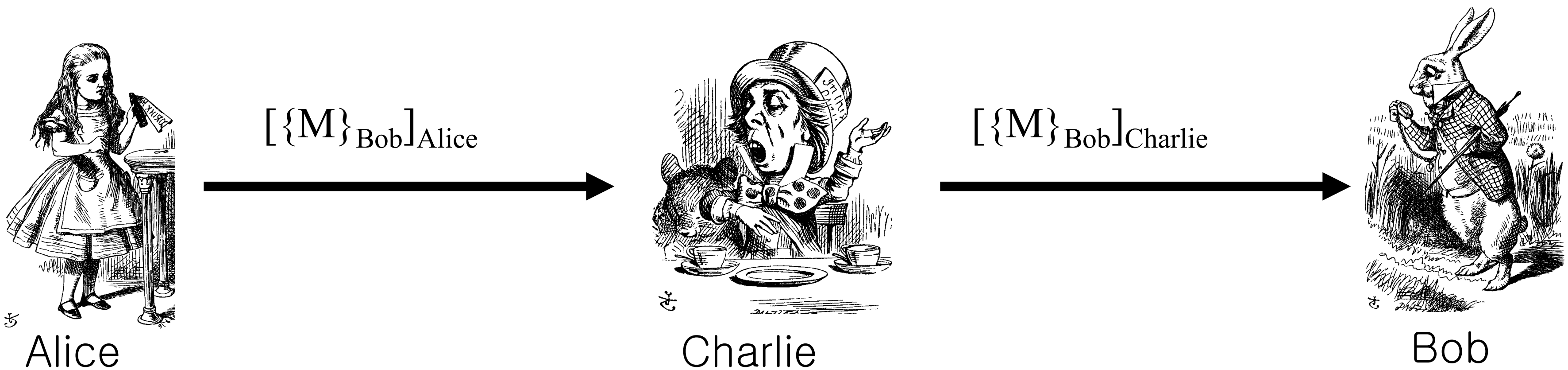
- M = “I love you”




- **Q:** What’s the problem?
- **A:** No problem — public key is public

# ENCRYPT AND SIGN

- M = "My theory, which is mine…."




- Note that Charlie cannot decrypt M
- Q: What is the problem?
- A: No problem — public key is public

# PUBLIC KEY CERTIFICATE

- Digital **certificate** contains name of user and user's public key (possibly other info too)
- It is ***signed*** by the issuer, a ***Certificate Authority*** (CA), such as VeriSign

  $M$ = (Alice, Alice's public key), $S = [M]_{CA}$

  **Alice's Certificate** = $(M, S)$
- Signature on certificate is verified using CA's public key

  Must verify that $M = \{S\}_{CA}$

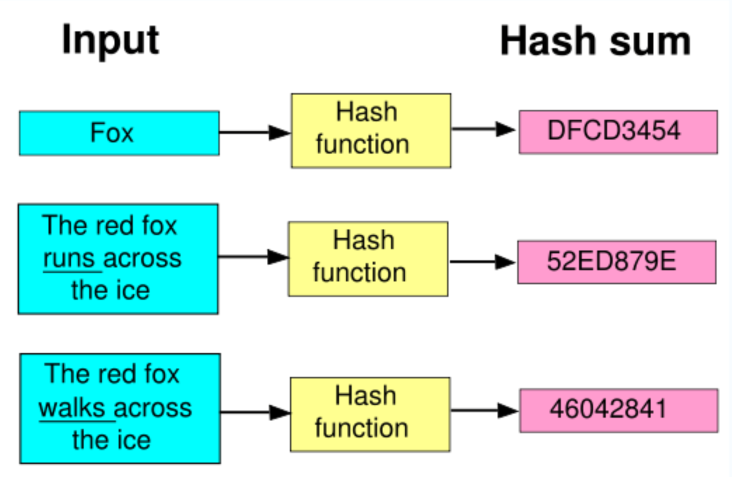
# CERTIFICATE AUTHORITY

- Certificate authority (CA) is a trusted 3rd party (TTP) — creates and signs certificates
- Verify signature to verify **integrity** & identity of **owner of corresponding private key**
  - Does **not** verify the identity of the **sender** of certificate — certificates are public!
- Big problem if CA makes a mistake
  - CA once issued Microsoft cert. to someone else
- A common format for certificates is X.509

# HASH FUNCTION?

- Hash function
  - a reproducible method of turning some kind of data into a (relatively) small number that may serve as a digital "fingerprint" of the data.

- Crypto Hash function
  - a hash function with certain additional security properties to make it suitable for use as various info security applications



# HASH FUNCTION MOTIVATION

- Suppose Alice signs **M**
  - Alice sends M and S = $[M]_{Alice}$ to Bob
  - Bob verifies that M = $\{S\}_{Alice}$
  - Aside: Is it OK to just send S?
- If M is big, $[M]_{Alice}$ is costly to compute
- Suppose instead, Alice signs h(M), where h(M) is much smaller than M
  - Alice sends M and S = $[h(M)]_{Alice}$ to Bob
  - Bob verifies that h(M) = $\{S\}_{Alice}$

# CRYPTO HASH FUNCTION

- Crypto hash function h(x) must provide the following properties
- **Compression**
  - output length is small
- **Efficiency**
  - h(x) easy to computer for any x
- **One-way**
  - given a value y it is infeasible to find an x such that h(x) = y

# CRYPTO HASH FUNCTION

- **Weak collision resistance**
  - given $x$ and $h(x)$, infeasible **to find $y$** with $y \neq x$ such that $h(y) = h(x)$
- **Strong collision resistance**
  - infeasible **to find any $x$ and $y$,** with $x \neq y$ such that $h(x) = h(y)$
- Lots of collisions exist, but hard to find one

# POPULAR CRYPTO HASHES

- **MD(Message Digest) 5**
  - invented by Rivest
  - 128 bit output
  - MD2 → MD4 → MD5
  - MD2 and MD4 are no longer secure, due to collision found
  - Note: even MD5, collision recently found

# POPULAR CRYPTO HASHES

- **SHA(Secure Hash Algorithm)-1**
  - A US government standard (similar to MD5)
  - "The world's most popular hash function"
  - 180 bit output
  - SHA-0 $\rightarrow$ SHA-1
- Many others hashes, but MD5 and SHA-1 most widely used

# Hash usages

# HASH USES

- Authentication (HMAC)
- Message integrity (HMAC)
- Message fingerprint
- Data corruption detection
- Digital signature efficiency
- Anything you can do with symmetric crypto ???

# ONLINE AUCTION

- Suppose Alice, Bob and Charlie are bidders
- Alice plans to bid A, Bob B and Charlie C
- They don’t trust that bids will stay secret
- Solution?
  - Alice, Bob, Charlie submit **hashes** h(A), h(B), h(C)
  - All hashes received and posted online
  - Then bids A, B and C revealed
- Hashes don’t reveal bids (one way)
- Can’t change bid after hash sent (collision)

# INFORMATION HIDING DIGITAL WATERMA

- Digital Watermarks
  - The "Digital Watermarking" name from watermarking of paper or money as a security measure
  - A technique which allows an individual to add hidden copyright notices to digital audio, video, or image signals and documents
  - Defense against music or software piracy
  - Example: Add "invisible" identifier to data
  - Digital watermarking can be a form of steganography

# INFORMATION HIDING DIGITAL WATERMA

BRAC UNIVERSITY
Inspiring Excellence

- Add a “mark” to data
- Several types of watermarks
- Type 1
  - Invisible — Not obvious the mark exists
  - Visible — Such as **TOP SECRET** stamp
- Type 2
  - Robust — Readable even if attacked
  - Fragile — Mark destroyed if attacked

# INFORMATION HIDING DIGITAL WATERMA

- Add robust invisible mark to digital music
  - If pirated music appears on Internet, can trace it back to original source
- Add fragile invisible mark to audio file
  - If watermark is unreadable, recipient knows that audio has been tampered (integrity)
- Combinations of several types are sometimes used
  - E.g., visible plus robust invisible watermarks

# INFORMATION HIDING STEGANOGRAPHY

- Steganography – "Hidden writing"
  - The art and science of writing hidden messages
    - recipient does not know of the existence of the mssg
  - Hide the fact that information is being transmitted – a kind of covert channel (Ch8)
    - Secret communication channel
    - Cryptography, where the existence of the message itself is not disguised, but the content is obscured.
  - Example: Hide data in image or music file

# INFORMATION HIDING STEGANOGRAPHY

- According to Herodotus (Greece 440BC)
  - Shaved slave's head
  - Wrote message on head
  - Let hair grow back
  - Send slave to deliver message
  - Shave slave's head to expose message (warning of Persian invasion)
- Historically, Steganography has been used more than cryptography!

# INFORMATION HIDING STEG EXAMPLE

- Images use 24 bits for color: RGB
  - 8 bits for red, 8 for green, 8 for blue
- For example
  - **0x7E 0x52 0x90** is this color
  - **0xFE 0x52 0x90** is this color
- While
  - **0xAB 0x33 0xF0** is this color
  - **0xAB 0x33 0xF1** is this color
- Low-order bits are unimportant!

# INFORMATION HIDING STEG EXAMPLE

- Given an uncompressed image file
  - For example, BMP format
- Then we can insert any information into low-order RGB bits
- Since low-order RGB bits don't matter, result will be "invisible" to human eye
- But a computer program can "see" the bits

# INFORMATION HIDING STEG EXAMPLE 1

- Left side: plain Alice image
- Right side: Alice with entire *Alice in Wonderland* (pdf) “hidden” in image

# INFORMATION HIDING STEG EXAMPLE 2

BRAC UNIVERSITY Inspiring Excellence

## Non-Steganography Example

- Walrus.html in web browser

"The time has come," the Walrus said,
"To talk of many things:
Of shoes and ships and sealing wax
Of cabbages and kings
And why the sea is boiling hot
And whether pigs have wings."

- View source

```
<font color="#000000">"The time has come," the Walrus said,</font><br>
<font color="#000000">"To talk of many things:</font><br>
<font color="#000000">Of shoes and ships and sealing wax</font><br>
<font color="#000000">Of cabbages and kings</font><br>
<font color="#000000">And why the sea is boiling hot</font><br>
<font color="#000000">And whether pigs have wings."</font><br>
```

# INFORMATION HIDING STEG EXAMPLE 2

- stegoWalrus.html in web browser

"The time has come," the Walrus said,
"To talk of many things:
Of shoes and ships and sealing wax
Of cabbages and kings
And why the sea is boiling hot
And whether pigs have wings."

- View source

```
<font color="#010100">"The time has come," the Walrus said,</font><br>
<font color="#000100">"To talk of many things:</font><br>
<font color="#010100">Of shoes and ships and sealing wax</font><br>
<font color="#000101">Of cabbages and kings</font><br>
<font color="#000000">And why the sea is boiling hot</font><br>
<font color="#010001">And whether pigs have wings."</font><br>
```

- "Hidden" message: **110 010 110 011 000 101**

# INFORMATION HIDING STEGANOGRAPHY

BRAC UNIVERSITY
Inspiring Excellence

- Some formats (jpg, gif, wav, etc.) are more difficult (than html) for humans to read
- Easy to hide information in **unimportant bits**
- Easy to **destroy** or remove info stored in unimportant bits!

# INFORMATION HIDING STEGANOGRAPHY

- To be robust, information must be stored in **important bits**
- But stored information must not damage data!
- Collusion attacks also a major concern
  - The original and watermarked object can be compared
- Robust steganography is trickier than it seems

# THE BOTTOM LINE OF INF HIDING

- If information hiding is suspected
  - Attacker can probably make information/watermark unreadable
  - Attacker may be able to read the information, given the original document (image, audio, etc.)

# CHAPTER 5
# HASH FUNCTIONS


PREPARED BY:

DR. MUHAMMAD IQBAL HOSSAIN

ASSISTANT PROFESSOR

DEPARTMENT OF CSE, BRAC UNIVERSITY

- SHA -Secure Hash Algorithm
- MD5 – Message Digest
- The process
  - Sender use MD5
  - Append Message Digest to plain text
  - Send it to receiver
  - Receiver compute with MD5
  - Receiver compare MD5, MD5

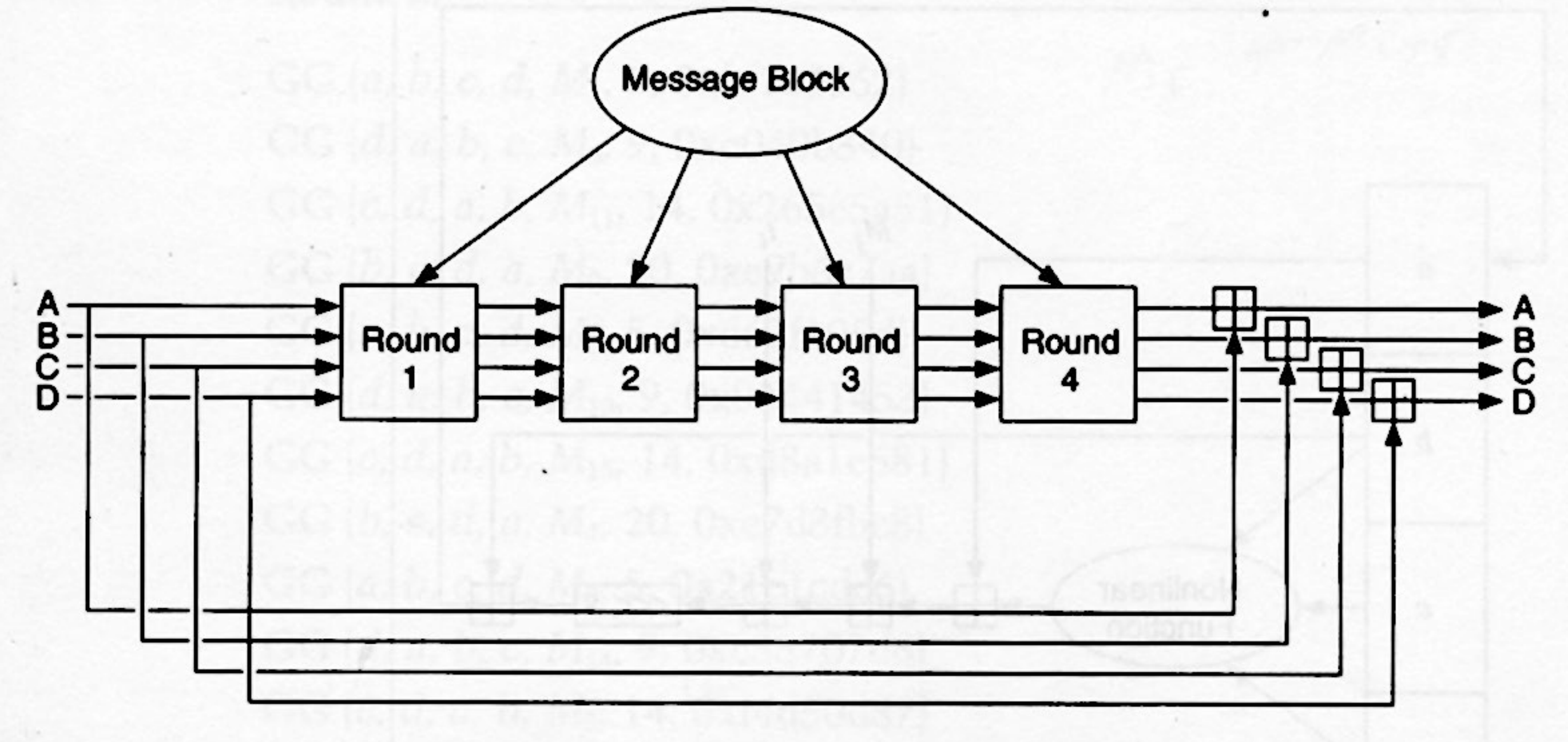
# MD5 ALGORITHM (128 BIT)

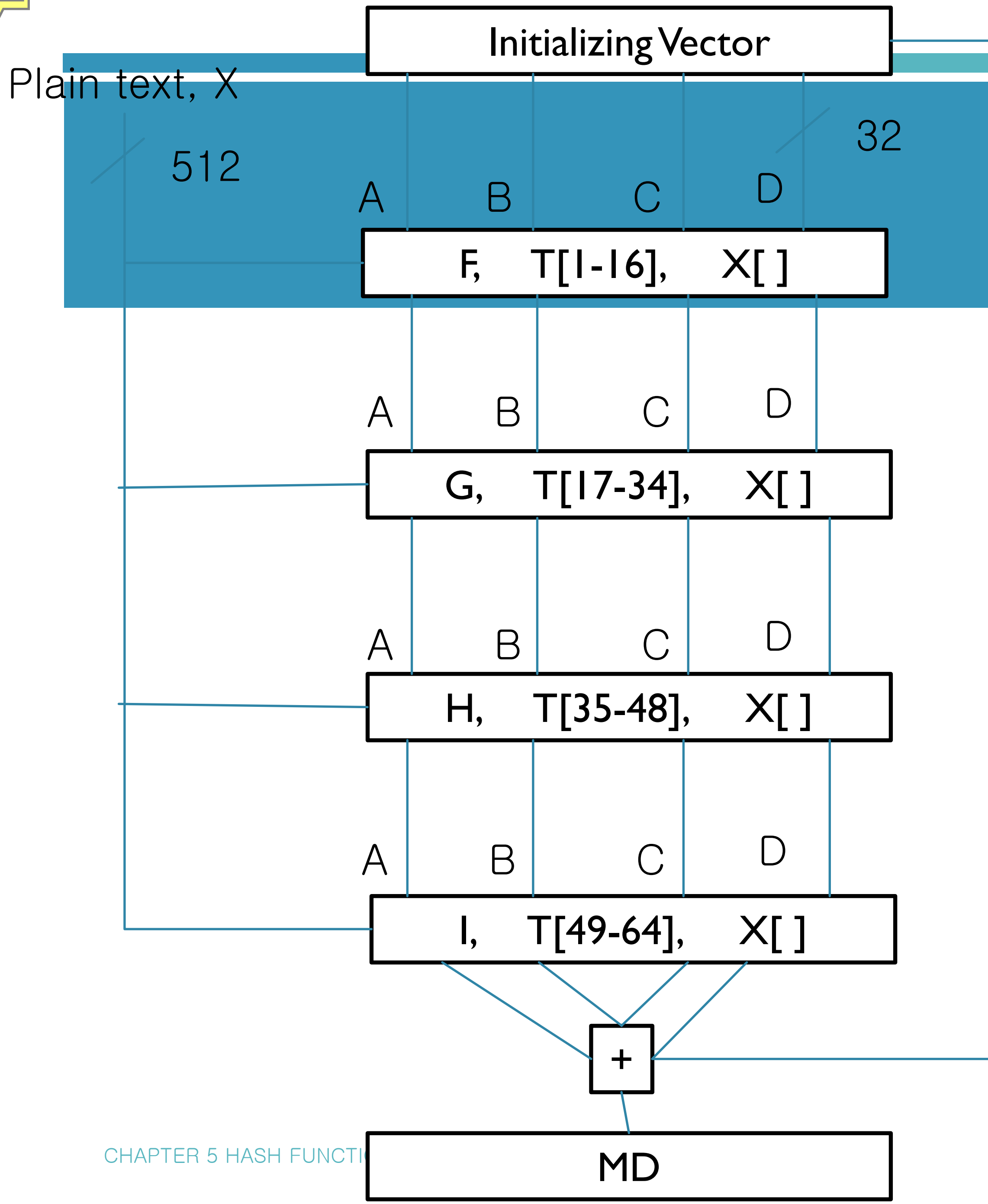
1. The Message is padded to an exact multiple of 512 bit blocks.
   a. Append 64 bit representation
2. Initiate the MD buffers (32-bit, 4 buffer, A, B, C, D)
3. Process the each block (512)
4. Output (message digest in buffers)

# MAIN MD5 LOOP

IVs

$A = 0x01234567$
$B = 0x89abcdef$
$C = 0xfedcba98$
$D = 0x76543210$






Each round have 16 constraints and 16 steps. Each constraint will use in each step.

Uses 4 32-bit inputs: *a, b, c, d*
Also uses a 32-bit sub-block of the message block
Generates 4 32-bit outputs for the next round step or the next round

# DIFFICULTY IN CRACKING

- Md5, with its 128bit encryption algorithm has 1,280,000,000,000,000,000 possible combinations.
- Even if the exact same hash value found, possible other string combination could have created it.
- It is considered that the md5 message digest would take an unrealistic time to crack via brute force attack.

# PROS/CONS MD5

- Easy to use
- Widely used
- Considered secure
- Difficult to crack

- Is susceptible to brute force attacks
- Hash collisions is a known flaw
- Quantum computers would make such an algorithm worthless