

ISA 666

Internet Security Protocols

Basic Number Theory
Hash Functions

Recommended Reference text on Number theory for Cryptography

- Number Theory for Computing
 - Second edition
- Song Y. Yan
- Springer-Verlag,
 - ISBN = 3-540-43072-5

Basic Number Theory

- We are talking about integers!
 - Integers have many interesting properties that have been widely used in modern cryptography
 - Wonder how to manually calculate
 - $101^{115} \bmod 33$?
 - $115^{101} \bmod 33$?
- Divisor
 - We say that $b \neq 0$ divides a if $a = mb$ for some m , denoted $b|a$. b is a divisor of a .
 - If $a|1$, then $a = 1$ or -1 .
 - If $a|b$ and $b|a$, then $a = b$ or $-b$.
 - Any $b \neq 0$ divides 0 .
 - If $b|g$ and $b|h$, then $b|(mg+nh)$ for arbitrary integers m and n .

Prime Numbers

- Prime numbers
 - An integer $p > 1$ is a prime number if its only divisors are 1 , -1 , p , and $-p$.
 - Examples: $2, 3, 5, 7, 11, 13, 17, 19, 31, \dots$

Theorem: There are infinitely many prime numbers

Proof: Suppose there are only finitely many, say n many and the primes are p_1, \dots, p_n . Then $1 + p_1 \times \dots \times p_n$ is not divisible by either of p_1, \dots, p_n . \square

Factorization

- Any integer $a > 1$ can be factored in a unique way as
 - where each $p_1 > p_2 > \dots > p_t$ are prime numbers and where each $a_i > 0$.
 - Examples: $91 = 13 \times 7$, $11011 = 13 \times 11^2 \times 7$.
- The general expression is of the form:
where each p_i is a prime.

$$a = p_1^{a_1} p_2^{a_2} \dots p_t^{a_t}$$

Factorization of Numbers

- Another view of $a|b$:
 - Let P be the set of all prime numbers
 - Represent a as $a = \prod_{p \in P} p^{a_p}$, where $a_p \geq 0$.
 - Represent b as $b = \prod_{p \in P} p^{b_p}$, where $b_p \geq 0$.
 - $a|b$ means that $a_i \leq b_i$.
- Factoring an integer
 - Not trivial for large, say 128 bit, integers!
 - The very foundation of RSA security

Basic Number Theory (Cont'd)

- Greatest common divisor: $\gcd(a,b)$
 - $\gcd(a,b) = \max \{k \mid k|a \text{ and } k|b\}$
 - Examples
 - $\gcd(6,15)=3$.
 - $\gcd(60,24)=\gcd(60,-24)=12$.
 - $\gcd(a,0) = a$.
 - $\gcd(a,b)$ can be easily derived if we can factor a and b .
- Relatively Prime Numbers
 - Integers a and b are relatively prime iff $\gcd(a,b) = 1$.
 - Non-prime numbers can be relative primes!
 - Example: 8 and 15 are relatively prime.

How To Find Greatest Common Divisor? Euclid's Algorithm

- Observation
 - $\gcd(a, b) = \gcd(b, a - b)$
 - $\gcd(a, b) = \gcd(b, a \bmod b)$
- Euclid (d, f) , $d > f > 0$.
 - $X \leftarrow d; Y \leftarrow f$
 - While $Y \neq 0$ do
 - $R = X \bmod Y$
 - $X \leftarrow Y$
 - $Y \leftarrow R$
 - return $X = \gcd(d, f)$

Example: $\gcd(12,9)$

First Round

• $x=12, y=9$

• $R = 12 \bmod 9 = 3$

Second Round

• $x=9, y=3$

• $R = 9 \bmod 3 = 0$

Third round ($y=0$)

Return (3)

Modulo Operator

- Given any positive integer n and any integer a , we have $a = qn + r$, where $0 \leq r < n$ and $q = \lfloor a/n \rfloor$.
 - We write $r = a \bmod n$.
 - The remainder r is often referred to as a residue.
 - Example:
 - $2 = 12 \bmod 5$.
- Two integer a and b are said to be **congruent modulo n** if $a \bmod n = b \bmod n$.
 - We write $a \equiv b \bmod n$
 - Example:
 - $7 \equiv 12 \bmod 5$.

Modulo Operator (Cont'd)

- Properties of the modulo operator
 - $a \equiv b \bmod n$ iff a differs b by multiple of n .
 - $a \equiv b \bmod n$ iff $n \mid (a - b)$
 - $a \equiv b \bmod n$ implies $b \equiv a \bmod n$.
 - $a \equiv b \bmod n$ and $b \equiv c \bmod n$ imply $a \equiv c \bmod n$.

Modular Arithmetic

- Observation:
 - The $(\text{mod } n)$ operator maps all integers into the set of integers $\{0, 1, 2, \dots, (n-1)\}$. **Creates a partition on the integer set**
- Modular addition.
 - $[(a \text{ mod } n) + (b \text{ mod } n)] \text{ mod } n = (a+b) \text{ mod } n$
- Modular subtraction.
 - $[(a \text{ mod } n) - (b \text{ mod } n)] \text{ mod } n = (a - b) \text{ mod } n$
- Modular multiplication.
 - $[(a \text{ mod } n) \times (b \text{ mod } n)] \text{ mod } n = (a \times b) \text{ mod } n$

An Exercise (n=5)

- Addition

	0	1	2	3	4
0					
1					
2					
3					
4					

- Multiplication

	0	1	2	3	4
0					
1					
2					
3					
4					

Exponentiation

$$92^{10} \text{ mod } 5 = \underline{\hspace{2cm}}$$

Properties of Modular Arithmetic

- $Z_n = \{0, 1, \dots, (n-1)\}$
- Commutative laws
 - $(w + x) \bmod n = (x + w) \bmod n$
 - $(w \times x) \bmod n = (x \times w) \bmod n$
- Associative laws
 - $[(w + x) + y] \bmod n = [w + (x + y)] \bmod n$
 - $[(w \times x) \times y] \bmod n = [w \times (x \times y)] \bmod n$
- Distributive law
 - $[w \times (x + y)] \bmod n = [(w \times x) + (w \times y)] \bmod n$
- Identities
 - $(0 + w) \bmod n = w \bmod n$
 - $(1 \times w) \bmod n = w \bmod n$
- Additive inverse ($-w$)
 - For each $w \in Z_n$, there exists a z such that $w + z = 0 \bmod n$.

Modular Multiplicative Inverse

- Why do we care about modular multiplicative inverse?
 - Used in RSA...
- Don't always exist
 - Example: There doesn't exist a z such that $6 \times z = 1 \bmod 8$.

Z_8	0	1	2	3	4	5	6	7
$\times 6$	0	6	12	18	24	30	36	42
Residues	0	6	4	2	0	6	4	2

- An integer $a \in Z_n$ has a modular multiplicative inverse iff $\gcd(a, n) = 1$.
- In particular, if n is a prime number, then all elements in Z_n have modular multiplicative inverse.

Extended Euclid Algorithm:

Computing modular inverses: $Y_2 = d^{-1} \pmod f$

Extended Euclid (d, f)

1. $(X_1, X_2, X_3) \leftarrow (1, 0, f)$;
2. $(Y_1, Y_2, Y_3) \leftarrow (0, 1, d)$
3. If $Y_3 = 0$
 return ("no inverse")
4. If $Y_3 = 1$
 return $(Y_2) /* = d^{-1} \pmod f */$
5. $Q = \lfloor X_3 / Y_3 \rfloor$
6. $(T_1, T_2, T_3) \leftarrow$
 $(X_1 - QY_1, X_2 - QY_2, X_3 - QY_3)$
7. $(X_1, X_2, X_3) \leftarrow (Y_1, Y_2, Y_3)$
8. $(Y_1, Y_2, Y_3) \leftarrow (T_1, T_2, T_3)$
9. Goto 2

Compute $5^{-1} \pmod 6$

1. $(x_1, x_2, x_3) = (1, 0, 6)$
2. $(y_1, y_2, y_3) = (0, 1, 5)$
- 3, 4 no effect
5. $q = \lfloor 6/5 \rfloor = 1$
6. $(T_1, T_2, T_3) = (5, 5, 1)$
7. $(x_1, x_2, x_3) = (0, 1, 5)$
8. $(y_1, y_2, y_3) = (5, 5, 1)$
9. Goto 2
2. $(y_1, y_2, y_3) = (5, 5, 1)$
3. no effect
4. **return** $(y_2) = 5$

NOTE: $5 \cdot 5 = 25 = 1 \pmod 6$

Observations

In the previous Algorithm:

$$fX_1 + dX_2 = X_3; fY_1 + dY_2 = Y_3 - \text{loop invariants}$$

If $Y_3 = 1$, then

$$fY_1 + dY_2 = 1, \text{ and therefore}$$

$$Y_2 = d^{-1} \pmod f$$

$$(\text{also } Y_1 = f^{-1} \pmod d)$$

Chinese Remainder Theorem

- Suppose m_1, \dots, m_n are pairwise relatively prime and a_1, \dots, a_n are any integers. Then there is solution to the following set of equations:

$$x = a_1 \pmod{m_1}$$

$$x = a_2 \pmod{m_2}$$

..... =,

$$x = a_n \pmod{m_n}$$

If x and x' are two solutions then $x' = x \pmod{(z_1, \dots, z_n)}$

Chinese Remainder Theorem

- Two simultaneous congruences
 $\mathbf{X = n_1 \pmod{m_1}}$ and $\mathbf{X = n_2 \pmod{m_2}}$
are solvable only when
 $\mathbf{n_1 = n_2 \pmod{\gcd(m_1, m_2)}}.$
- The solution is unique modulo $\mathbf{lcm(m_1, m_2)}$

Example

- Solve:

$$p1: x = 2 \pmod{3}$$

$$p2: x = 3 \pmod{5}$$

$$p3: x = 2 \pmod{7}$$

Note:

$$\gcd(3,5)=\gcd(5,7)=\gcd(3,7)=1$$

- From p1, $x = 3t + 2$, for some integer t .
- Substituting this into p2 gives $3t = 1 \pmod{5}$
- Same as p4: $t = 2 \pmod{5}$
{Why?: implies $3t=6 \pmod{5} \rightarrow t=2 \pmod{5}$ }
- Let $t=2+5s$
- Substitution into $x = 3t + 2$ yields $x = 15s + 8$

Continuing the example ...

- Again: $p3: x = 2 \pmod{7}$
- Substituting in p3: $15s + 8 = 2 \pmod{7}$
- Divide out by 7 gives $s = 1 \pmod{7}$
[How?: $(2*7+1)s + (7+1) = 2 \pmod{7}$]
- So $s = 7u + 1$
- Hence $x = 105u + 23$
- That is, any solution where $x = 105u + 23$ satisfies p_1, p_2 and p_3 .

Proof of the Chinese Remainder Theorem

- Let $k_i = m_1 \dots m_i \dots m_{i+1} \dots m_n$
- Then k_i and m_i are relatively prime
- So there are integers r and s satisfying $rk_i + sm_i = 1$
- Hence
$$rk_i = 0 \pmod{k_i} \text{ and } rk_i = 1 \pmod{m_i}.$$
- Because $m_1, \dots, m_i, \dots, m_{i+1}, \dots, m_n$ divide k_i we get $X_i = rk_i$ satisfy the congruences
$$X_i = 0 \pmod{m_1}, X_i = 0 \pmod{m_2}, X_i = 0 \pmod{m_3}, \dots$$

except $X_i = 0 \pmod{m_i}$

Continued...

- To solve the set of equations,
 - Let $X = a_1x_1 + \dots + a_nx_n$
Then $X = a_ix_i = a_i \pmod{m_i}$
 - Hence, this x is a solution.
- For the uniqueness:
- If x' is another solution, then $x' = x \pmod{m_i}$ for each i
 - Hence $x - x' = 0 \pmod{m_i}$, and
 - Hence $\text{lcm}(m_1, \dots, m_n)$ divides $x - x'$

Z_n and Z_n^*

- $Z_n = \{0, 1, \dots, n-1\}$
- $Z_n^* = \{x \mid x \in Z_n \text{ and } x \text{ is relatively prime to } n\}$
- **Example:** $Z_{10}^* = \{1, 3, 7, 9\}$

Theorem: Z_n^* is closed under multiplication

Proof: Assume $a, b \in Z_n^*$. Need to show $a \cdot b \in Z_n^*$

$$Ua + Vn = 1 \text{ -----(1)}$$

$$Xb + Yn = 1 \text{ -----(2)}$$

Multiply (1) and (2)

$$[Ua + Vn][Xb + Yn] = 1$$

$$UVab + [VXb + UYn + VYn]n = 1$$

Euler's Totient Function

- **Totient function $\phi(n)$:** $|Z_n^*|$
 - number of integers less than n and relatively prime to n
 - If n is prime, $\phi(n) = n-1$
 - If $n = p \cdot q$, and p, q are primes, $\phi(n) = (p-1)(q-1)$
 - If p is prime and $k > 0$, $\phi(p^k) = (p-1)p^{k-1}$
- **Examples:**
 - $\phi(7) = 6$
 - $\phi(21) = \phi(3 \cdot 7) = 2 \cdot 6 = 12$

Why is $\phi(n)=(p-1)(q-1)$ valid?

- Show:

$\gcd(m,pq)=1$ iff $\gcd(m,p)=1$ and $\gcd(m,q)=1$

\Rightarrow obvious, because any factor of pq is a factor of p or a factor of q

\Leftarrow Suppose $m_p \in \mathbb{Z}_p$ and $m_q \in \mathbb{Z}_q$

Then $U_p m_p + V_p p = 1$ and $U_q m_q + V_q q = 1$.

Let $m_p = m - kp$ and $m_q = m - lq$

Hence $U_p m + (V_p - U_p k)p = 1$ and $U_q m + (V_q - U_q l)q = 1$

Multiply and get $[U_p U_q + **]m + [***]pq = 1$

So $m_p \in \phi(p)$ and $m_q \in \phi(q)$ iff $m_p m_q \in \phi(pq)$

Euler's Theorem

- For every a and n that are relatively prime, $a^{\phi(n)} \equiv 1 \pmod n$.

- Examples

– $a=3, n=10, \phi(10)=\underline{\quad}, 3^{\phi(10)} \pmod{10} = \underline{\quad}$

– $a=2, n=11, \phi(11)=\underline{\quad}, 2^{\phi(11)} \pmod{11} = \underline{\quad}$

- What if a and n that are **NOT** relatively prime?

- Generalization of Euler's Theorem

– if $n=pq$ (p, q are prime numbers) $a^{k\phi(n)+1} \equiv a \pmod n$ for all a in \mathbb{Z}_n .

– $\phi(33) = \phi(3 \cdot 11) = 2 \cdot 10 = 50$,

– $101^{101} \pmod{33} = \underline{\quad}??$

Hint: $101^{101} = 101^{2 \cdot \phi(33) + 1} = 101 \pmod{33} = 2$

Why is $a^{\phi(n)} \equiv 1 \pmod n$ valid?

- Let X be the product of all $\phi(n)$ integers in Z_n^*
- Remember Z_n^* is closed under multiplication
- Now multiply each elements of Z_n^* by a .
- Now multiply all of them together.
- We get $a^{\phi(n)} X$
- Multiplying all elements only rearranges them
- Hence the result is X
- Hence $a^{\phi(n)} X = X$

Modular Exponentiation

- If x, n are relative primes, then
$$x^y \pmod n = x^{y \pmod{\phi(n)}} \pmod n$$
- if $y \pmod{\phi(n)} = 1$ and x is relative prime with n , then $x^y \pmod n = x \pmod n$
- Example:
$$2^{101} \pmod{33} = 2^{101 \pmod{\phi(33)}} \pmod{33} = 2^2 \pmod{33} = 4$$

The Power of An Integer Modulo n

- Given $a, n > 0$, want to see if there exist $m > 0$ such that
- $a^m \equiv 1 \pmod{n}$
- If a and n are relatively prime, then there is at least one integer m that satisfy $a^m \equiv 1 \pmod{n}$
 - That is, the Euler's totient function $\phi(n)$.
 - But $\phi(n)$ may be not the smallest!
 - The least positive exponent m for which the above equation holds is referred to as:
 - The order of $a \pmod{n}$
 - The exponent to which a belongs \pmod{n}
 - The length of the period generated by a .

Understanding The Order of $a \pmod{n}$

- Powers of Integers Modulo 19

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Observations in The Previous Table

- All sequences end in 1.
- The length of a sequence divides $\phi(19) = 18$.
 - Lengths: 1, 2, 3, 6, 9, 18.
- Some of the sequences are of length 18.
 - The base integer a generates (via powers) all nonzero integers modulo 19 (with different order)

Primitive Root

- The highest possible order of $a \pmod{n}$ is $\phi(n)$.
- **Primitive root:** If the order of $a \pmod{n}$ is $\phi(n)$, then a is referred to as **a primitive root** of n .
 - The powers of a : a, a^2, \dots, a^{n-1} are distinct \pmod{n} and are all relatively prime to n .
- For a prime number p , if a is a primitive root of p , then a, a^2, \dots, a^{p-1} are all the distinct numbers \pmod{p} .

Discrete Logarithm

- Given a primitive root a for a prime number p :
 - The expression $b \equiv a^i \pmod{p}$, $0 \leq i < (p-1)$, produces the integers from 1 to $(p-1)$.
- The exponent i is referred to as the index of b for the base $a \pmod{p}$, denoted as $\text{ind}_{a,p}(b)$.
 - Inverse of exponentiation = logarithm
- Some properties:
 - $\text{ind}_{a,p}(1)=0$, because $a^0 \pmod{p} = 1$.
 - $\text{ind}_{a,p}(a)=1$, because $a^1 \pmod{p} = a$.

Discrete Logarithm

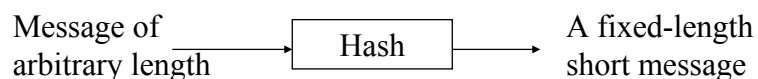
- Example:
 - Integer 2 is a primitive root of prime number 19

Number	1	2	3	4	5	6	7	8	9
Index	0	1	13	2	16	14	6	3	8
Number	10	11	12	13	14	15	16	17	18
Index	17	12	15	5	7	11	4	10	9

Discrete Logarithm (Cont'd)

- Consider $x = a^{\text{ind}_{a,p}(x)} \bmod p$, $y = a^{\text{ind}_{a,p}(y)} \bmod p$, and $xy = a^{\text{ind}_{a,p}(xy)} \bmod p$,
 - $a^{\text{ind}_{a,p}(xy)} \bmod p = (a^{\text{ind}_{a,p}(x)} \bmod p)(a^{\text{ind}_{a,p}(y)} \bmod p)$
 - $a^{\text{ind}_{a,p}(xy)} \bmod p = (a^{\text{ind}_{a,p}(x) + \text{ind}_{a,p}(y)}) \bmod p$
- By Euler's theorem: $a^z \equiv a^q \bmod p$ iff $z \equiv q \bmod \phi(p)$.
 - $\text{ind}_{a,p}(xy) = \text{ind}_{a,p}(x) + \text{ind}_{a,p}(y) \bmod \phi(p)$.
 - $\text{ind}_{a,p}(x^y) = [x \cdot \text{ind}_{a,p}(y)] \bmod \phi(p)$.
- Discrete logarithm mod p : index mod p .
- **Computing a discrete logarithm mod a large prime number p is in general difficult**
 - **Used as the basis of some public key cryptosystems.**

Hash Function



- Also known as
 - Message digest
 - One-way transformation
 - One-way function
 - Hash
- Length of $H(m)$ much shorter than length of m
- Usually fixed lengths: 128 or 160 bits

Requirements for a Hash Function

- Consider a hash function H
 - **Flexibility**: Can be applied to a block of data of any size
 - **Convenience (for check)**: produce a fixed-length short output.
 - **Performance**: Easy to compute $H(m)$
 - **One-way property**: Given $H(m)$ but not m , it's difficult to find m
 - **Weak collision resistance (freedom)**: Given $H(m)$, it's difficult to find m' such that $H(m') = H(m)$.
 - **Strong collision resistance (freedom)**: Computationally infeasible to find m_1, m_2 such that $H(m_1) = H(m_2)$

Birthday Paradox

- **Question**:
 - What is the minimum value of k such that the probability of **at least two people in a group of k people have the same birthday** is greater than 0.5 that?
 - Ignore the leap years and assume each birthday is equally likely.
- Probability of k people having k different birthdays:
 $Q(365, k) = {}^n C_k (1/365)^k = 365! / (365-k)! 365^k$
- Probability that at least two people have the same birthday:
 $P(365, k) = 1 - Q(365, k) > 0.5$ yields the integer solution to the equation
 $365! / (365-k)! 365^k > 0.5$
- K is **about 23**.

Generalization of the Birthday Paradox

- Given a random variable that is an integer with uniform distribution between 1 and n and a selection of k instances of the random variables, what is the least value of k such that the probability $P(n,k)$ is greater than 0.5 that there is at least one duplicate?

- $P(n,k) > 1 - e^{-k(k-1)/2n}$
- For large n and k , we have

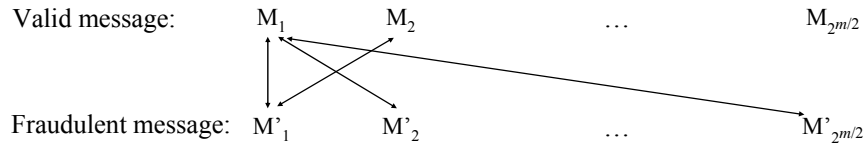
$$k = \sqrt{2(\ln 2)n} = 1.18\sqrt{n} \approx \sqrt{n}$$

- **Intuition:** How many k do we need to have a collision with $P=0.5$?
- **Implication**
 - For a hash function H with 2^m possible outputs, if we apply H to $k=(2^m)^{1/2}=2^{m/2}$ random inputs, the probability that there is at least one duplicate is greater than 0.5.

Birthday Attack

- **Objective= Find a fraudulent message with the same digest as a genuine message**
- The source, A, is prepared to sign a message
- The opponent generates $2^{m/2}$ variations of the message, and prepares $2^{m/2}$ variations on the fraudulent message.
- The opponent compares the two sets of messages to find a pair of messages that produces the same hash value. The opponent repeats generating variations until a match is found. The probability of success is greater than 0.5.
- The opponent offers the valid variation to A for signature, but attaches the signature to the fraudulent variation.

Birthday Attack (Cont'd)



- Check to see if $H(M_i) = H(M'_j)$?
- Send M_i to A to get his signature $H(M_i)$
- Append $H(M_i)$ to M'_j to get $M'_j \parallel H(M_i)$, which appears a valid message with valid signature by A

How Many Bits for Hash?

- For m bits, takes $2^{m/2}$ messages to find two with the same hash
- For 64 bits, takes 2^{32} messages to search duplicates
- Need a total of 128 bits (two of 64)

Building Hash Using Cipher Block Chaining Techniques

- Use Block Cipher for Hashing
- Use Message Blocks as Keys to the Block Cipher
- Divide M into fixed-size blocks M_1, M_2, \dots, M_n
- Compute the hash as follows
 - H_0 =Initial value
 - $H_i = E_{M_i}(H_{i-1})$
 - Hash value $G = H_n$
- Weakness
 - Birthday attack (reason: hash value is too short)
 - Meet-in-the-middle attack

Building Hash Using Cipher Block Chaining Techniques (Cont'd)

- **Meet-in-the-middle attack**
 - Get the correct hash value G
 - Construct any message in the form Q_1, Q_2, \dots, Q_{n-2}
 - Compute $H_i = E_{Q_i}(H_{i-1})$ for $1 \leq i \leq (n-2)$.
 - Generate $2^{m/2}$ random blocks; for each block X , compute $E_X(H_{n-2})$.
 - Generate $2^{m/2}$ random blocks; for each block Y , compute $D_Y(G)$.
 - With high probability there will be an X and Y such that $E_X(H_{n-2}) = D_Y(G)$.
 - Form the message $Q_1, Q_2, \dots, Q_{n-2}, X, Y$. It has the hash value G .

Modern Hash Functions

- MD2, MD4, MD5
 - Developed by Ron Rivest on 1989, 1990 and 1991 respectively. Specified in RFCs 1319 - 1321
 - Take arbitrarily long message, and generate 128-bit digest
 - MD2, MD4 have known weakness, and are considered broken
- SHA (Secure Hash Algorithm), SHA-1
 - Developed by NIST, very similar to MD4
 - SHA-1 was a revision to SHA that was published in 1994
 - Take up to 2^{64} bit message, and generate 160-bit digest
- HMAC
- RIPEMD-160

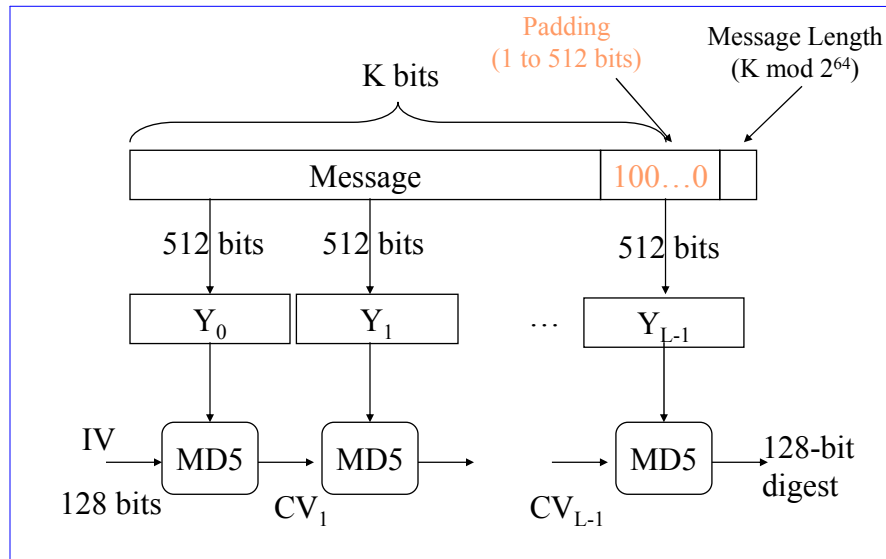
MD5: Message Digest Version 5

input Message



Output 128 bits Digest

MD5: A High-Level View



ISE at George Mason University

ISA 666 Duminda Wijesekera

47

Padding

- Given original message M , add padding bits "10*" such that resulting length is 64 bits less than a multiple of 512 bits.
- Append (the *length of the original message in bits mod 2^{64}*), represented in 64 bits to the padded message
- Final message is chopped 512 bits a block
- Exercise:
 - How to add padding bits to a message that is already a multiple of 512 bits?

ISE at George Mason University

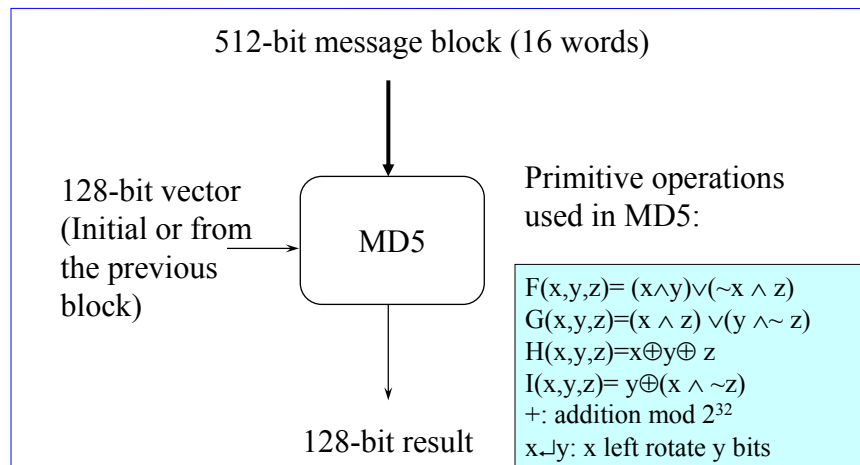
ISA 666 Duminda Wijesekera

48

MD5 (Intermediate) Buffer

- Used to hold intermediate and final result of MD5.
- 128 bits
- Represented as four 32-bit words (128=4 x 32)
 - (A,B,C,D)
 - Initially, A=0x67452301, B=0xEFCDAB89, C=0x98BADCFE, D=0x10325476
 - Stored in little-endian format:
 - A=0x01234567
 - B=0x89ABCDEF
 - C=0xFEDCBA98
 - D=0x76543210.

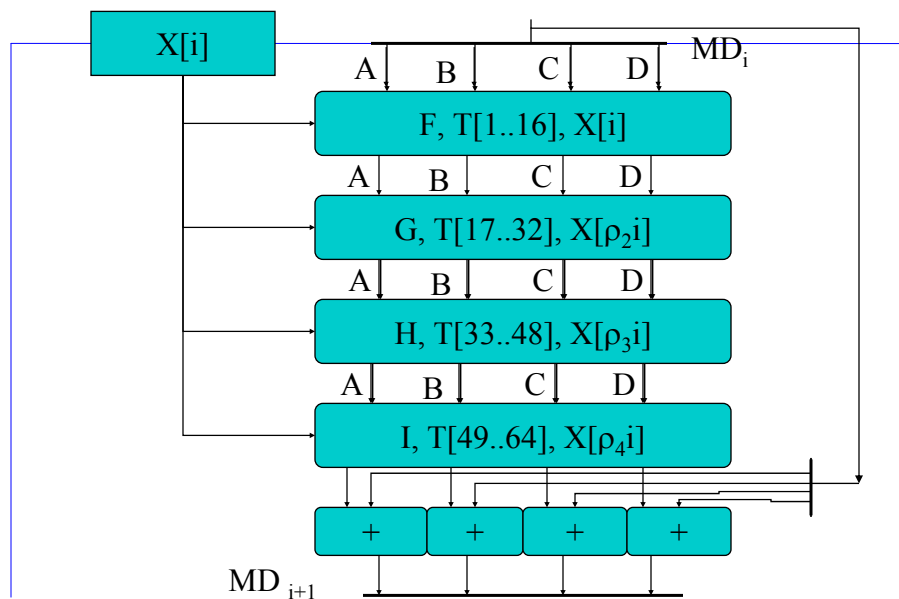
Processing of A Single Block



Processing of A Single Block (Cont'd)

- Every message block contains **16 32-bit words**:
 - $X[0] X[1] X[2] \dots X[15]$
- Every stage consists of **4 rounds** over the message block, each modifying the MD5 buffer (A,B,C,D).
 - The four rounds use functions F, G, H, I, respectively.
- Each round uses one-fourth of a 64-element table $T[1 \dots 64]$.
 - $T[i] = 2^{32} * \text{abs}(\sin(i))$ represented in 32 bits.
- The output of the fourth round is added to the *next stages'* input to the first round.

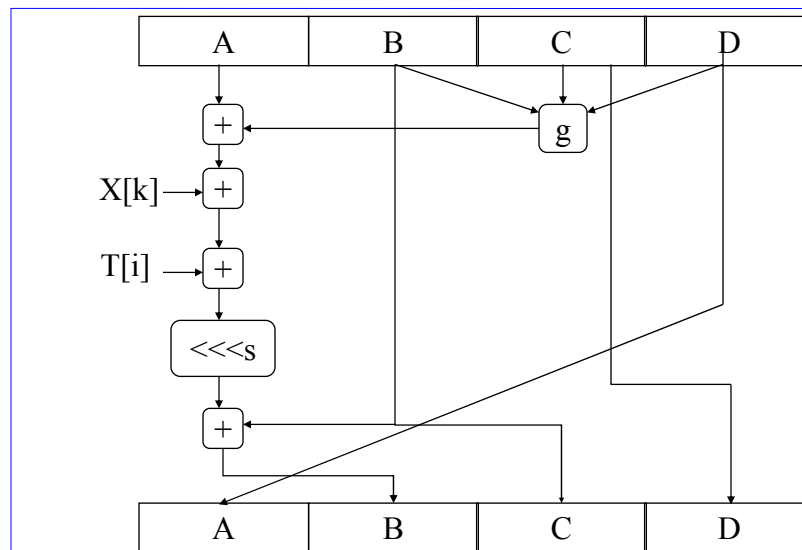
Processing of Block m_i : 4 Rounds



Logic of Each Round

- Each round consists of 16 steps
- Each step is of the form
 - $A \leftarrow B + ((A + g(B, C, D) + X[k] + T[i]) \lll s)$
 - Function g is one of F, G, H, I
 - $X[k]$ is the word in the input
 - $T[i]$ is the i^{th} word in T
 - $\lll s$: circular left shift by s bits.
 - Followed by a word level circular right shift of one word.

Logic of Each Step



Logic of Each Step

- Within a round, each of the 16 words of $X[i]$ is used exactly
 - First round, $X[i]$ are used in the order of I
 - Round 2, in the order of $\rho_2(i)$, where $\rho_2(i)=(1+5i)\bmod 16$;
 - Round 3, in the order of $\rho_3(i)$, where $\rho_3(i)=(5+3i)\bmod 16$;
 - Round 4, in the order of $\rho_4(i)$, where $\rho_4(i)=7i\bmod 16$;
- Each word of $T[i]$ is used exactly once.

Security of MD5

- No known method that breaks MD5 (till 08/2004).
- However, there are methods that are close to breaking MD5
 - Technique that enables the generation of a collision for a single 512-bit block.
- Birthday attack
 - 2^{64}

Latest Result on Collision of MD5, SHA

- In Crypto'2004,
 - Chinese researchers reported a family of collisions in MD5 (after fixing the previous bug in their analysis), and also reported that their method can efficiently (2^{40} hash steps) find a collision in SHA-0. Paper is available at <http://eprint.iacr.org/2004/199/>
 - Eli Biham announced new results in finding collision in a reduced-round version of SHA-1. The full SHA-1 algorithm does 80 rounds of scrambling. At present, Biham and Chen can break versions of SHA-1 that use up to about 40 rounds
- MD5 is fatally wounded! its use will be phased out soon.
- SHA-1 is still alive, NIST plans to phase out SHA-1 by 2010
 - brief is at http://csrc.nist.gov/hash_standards_comments.pdf