Module 1: Cryptography

Dr. Natarajan Meghanathan
Associate Professor of Computer Science
Jackson State University, Jackson, MS 39217
E-mail: natarajan.meghanathan@jsums.edu

Security Fundamentals

- **Confidentiality:** Data should be accessible only to entities (users/machines/processes) with the valid permissions (also includes privacy)
- **Integrity:**
  - Data should be modified only by entities with the valid permissions
  - A system should perform its function without any deliberate manipulation by entities without valid permissions
- **Availability:** Data and service should be accessible (timely and reliable) to entities with the valid permissions
- **Authentication:**
  - ***Entity authentication*** – validating user/machine identity
  - ***Message authentication*** – validating whether a message came from the user/machine/source who claims to have sent it
- **Access control:** Validating the permissions a user claims to have on a resource
- **Non-repudiation:** Actions of an entity should be uniquely traced back to that entity.

Security Fundamentals

- **Cryptography (Encryption and Decryption):**
  - Transform information from plaintext to ciphertext (encryption) so that it is not comprehensible for unauthorized entities during transmission or at the end systems (more towards confidentiality)
  - Every encryption algorithm needs to have a corresponding decryption algorithm to get back the plaintext
- **Digital Signature**: A form of encryption/ decryption that ensures the message came from the appropriate entity
  - *Non-repudiation*, *Message Authentication*
- **Hashing**: A digest of the message such that even if a bit changes in the message, the hash value should change
  - *Integrity*
- **Notarization**: Vouching for a user/machine – the notarizing authority is trusted by the associated entities
  - *Entity authentication*
- **Steganography:** Replace certain bits in a media file with the plaintext bits and transmit them
  - *Weak confidentiality* (but not very obvious to unauthorized users)

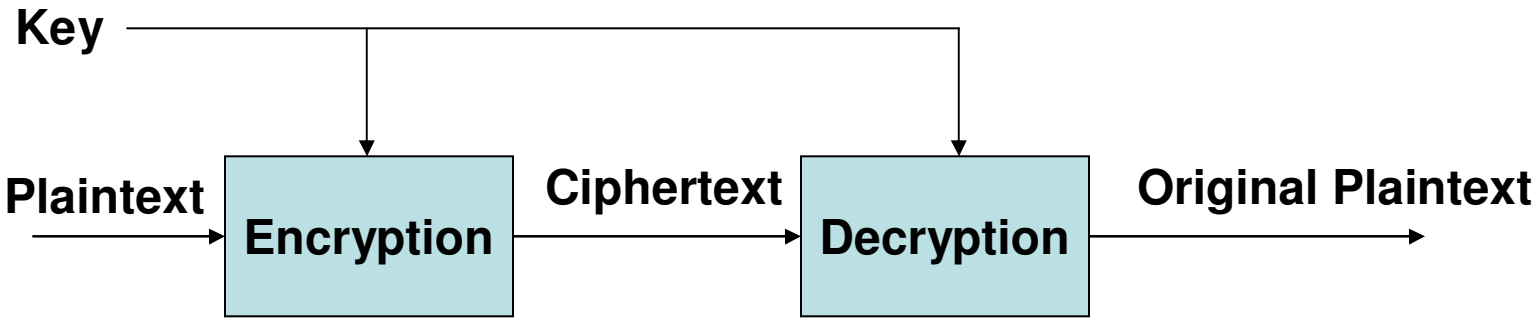
Security Mechanism vs. Security Service

- **Security Mechanism:** A mechanism that is designed to detect, prevent or recover from a security attack.
  - **Examples:** Encryption, Hashing, Digital signature, Notarization, Steganography

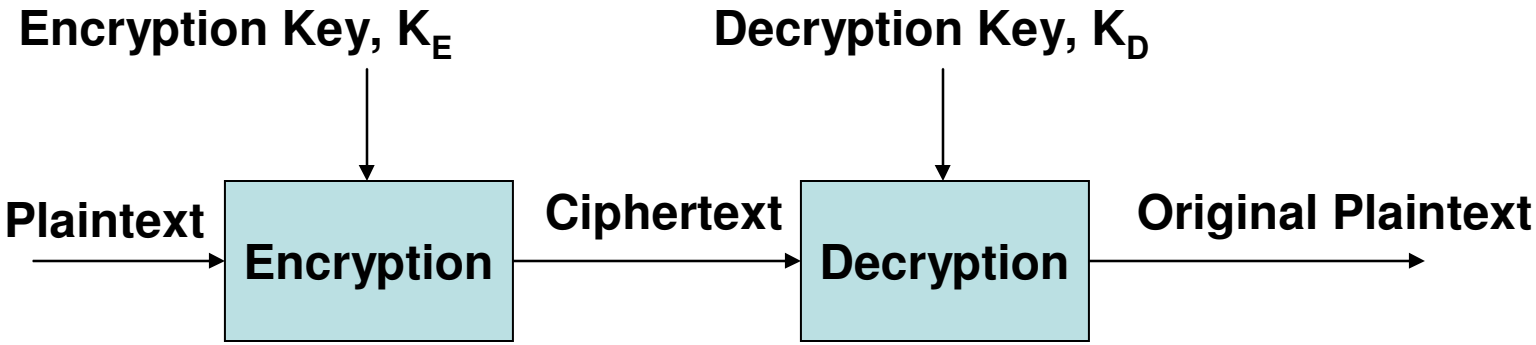
- **Security Service:** A service that enhances the security of the data processing systems and the information transfers of an organization.
- The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.
  - **Examples:** Authentication, Access control, Data confidentiality, Data integrity, Non-repudiation, Availability

Types of Encryption

- Symmetric encryption: The same key performs, both encryption and decryption.
  - P = D(K, E(K, P))



- Asymmetric encryption: distinct, very different keys, one for encryption and the other for decryption only



Cryptanalysis

- Analyzing the ciphertext (along with the encryption/ decryption algorithms, sometimes the plaintext, known plaintext-ciphertext pairs, etc) to deduce the key used for encryption.

Average Time Required for Exhaustive Key Search

Key Size (bits)	Number of Alternative Keys	Time Required at 1 Decryption/μs	Time Required at 10^6 Decryptions/μs
32	$2^{32} = 4.3 \times 10^9$	$2^{31}\ \mu s$ = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55}\ \mu s$ = 1142 years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127}\ \mu s$ = 5.4×10^{24} years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167}\ \mu s$ = 5.9×10^{36} years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26}\ \mu s = 6.4 \times 10^{12}$ years	6.4×10^6 years

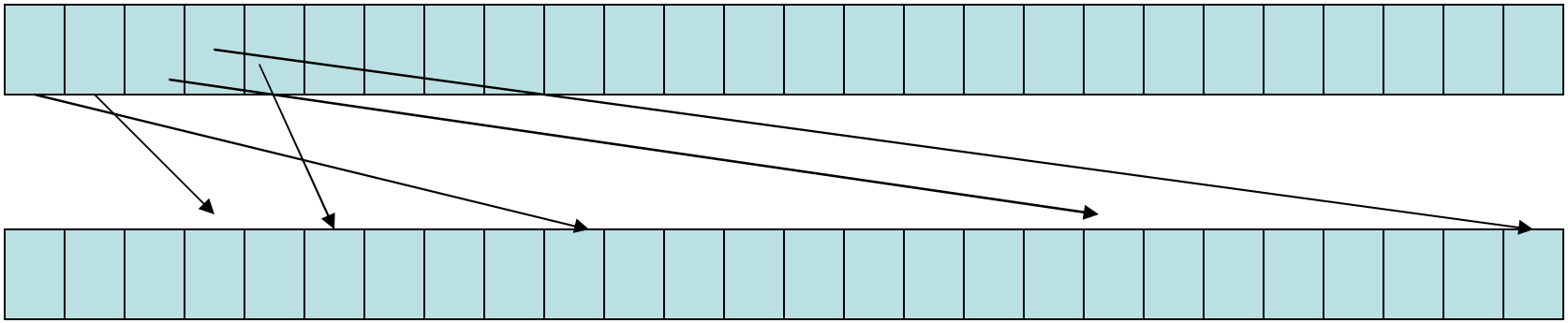
Source: Table 2.2 from William Stallings – Cryptography and Network Security, 5th Ed.

Principles of Symmetric Key Encryption Algorithms

- **Substitution:** Replace every character in the plaintext with a corresponding ciphertext.
  - The encryption and decryption of one character is independent of the others
    - Used for stream ciphers (more faster)
  - However, if the same key is used for encrypting every plaintext character to a ciphertext character (Ceasar Cipher), then cryptanalysis is quite straightforward.
  - Substitution-based algorithms are designed to cause more confusion. The keys are to be chosen such that the ciphertext for a particular plaintext character is different at different instants (like in Vigenere Cipher).
  - Classical ciphers like Caesar cipher, Vigenere Cipher are substitution-based.

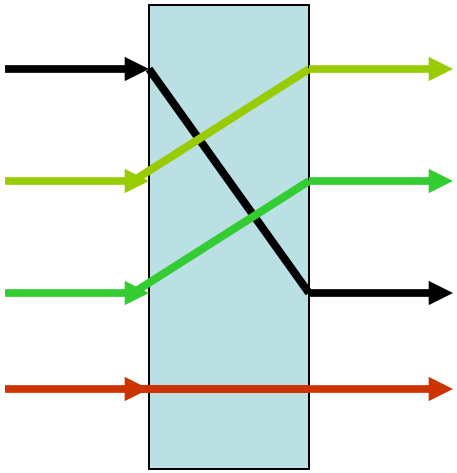
Principles of Symmetric Key Encryption Algorithms

- **Permutation:** The plaintext characters are moved here and there in a certain fashion that is known only to the sender and receiver.
  - The characters are not replaced one after the other.
  - Encryption are decryption are done to be in a block (relatively slow). Characters are moved within a block in a certain fashion.
  - Permutation-based ciphers are designed to produce more diffusion such that the adjacent plaintext characters are moved to far away locations in the ciphertext so that it becomes difficult to recover the plaintext.

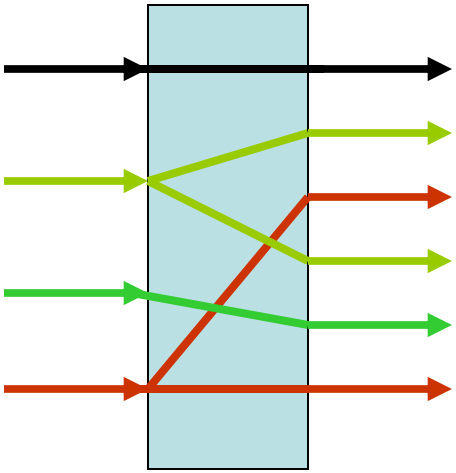


Example: Columnar transposition Cipher

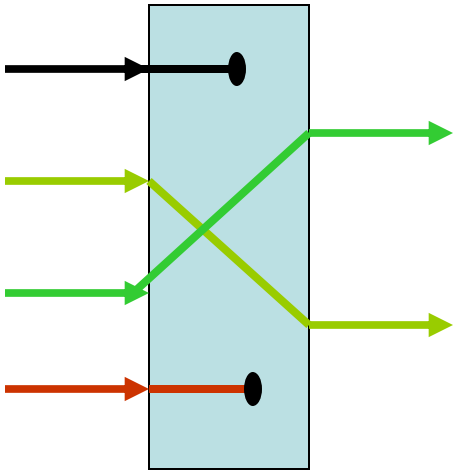
Types of Permutations



Permutation



Expansion
Permutation



Permuted Choice

Representing Characters

- Conventions/ Assumptions:
  - The plaintext is written in UPPERCASE letters and the ciphertext in lowercase letters
  - We use a numeric encoding for the letters as shown below as most encryption algorithms are based on mathematical transformations.

Letter	A	B	C	D	E	F	G	H	I	J	K	L	M
Code	0	1	2	3	4	5	6	7	8	9	10	11	12

Letter	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Code	13	14	15	16	17	18	19	20	21	22	23	24	25

  - We can perform arithmetic on the letters of a message. For example, A + 3 = D, K – 1 = J
  - Arithmetic is performed as if the above alphabetic table were circular. In other words, all arithmetic is with respect to modulo 26. The result of every arithmetic operation is between 0 and 25.
    - For example, Y + 3 = B

Substitution Ciphers

- Idea: Use a correspondence table and substitute a character or symbol for each character of the original message
- Goal of substitution is Confusion: an attempt to make it difficult for a cryptanalyst or an intruder to determine how a message and key were transformed into ciphertext.
- Caesar Cipher
  - Each letter is translated to the letter a fixed number of times after it in the alphabet table
  - Caesar cipher uses shift by 3.
  - $C_i = E(p_i) = p_i + 3$

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	d	e	f	g	h	i	j	k	l	m	n	o	p

Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	q	r	s	t	u	v	w	x	y	z	a	b	c

  - Example:
    - Plaintext: TREATY IMPOSSIBLE
    - Ciphertext: wuhdwb lpsrvvleoh
  - Complexity of the encryption algorithm: Length of the message

Cryptanalysis of the Caesar Cipher

- On a closer look at the result of applying Caesar's encryption technique to "TREATY IMPOSSIBLE", we get the following clues from the ciphertext even if did not know the plaintext:
  - The break in between the two words is preserved in the ciphertext
  - Double letters are preserved (SS is translated to vv)
  - When a letter is repeated, it maps to the same ciphertext as before (look at letters T, I, E in the plaintext)
- Consider you are given the following ciphertext and you want to determine the plaintext: "`wklv phvvdjh lv qrw wrr kdug wr euhdn`"
  - As a start, assume that the coder was lazy, and has allowed the blank space to be translated to itself. Hence, the message has actually been enciphered with a 27-symbol alphabet: A through Z and a blank-space separating the words.
  - If this assumption is true, knowing where the spaces helps to find out what are the small words.
  - The English language has very few short words like *am*, *is*, *to*, *be*, *he*, *she*, *we*, *and*, *you*, *are*, and so on.
- There is a strong clue in the repeated `r` of the word `wrr`.
  - Two very common three-letter words having the pattern `xyy` are `see` and `too`; other less common possibilities are `add`, `odd` and `off`. Try the more common word first.

Cryptanalysis of the Caesar Cipher

- Also, the combination `wr` appears in the ciphertext too, so you can determine whether the first two letters of the three-letter word form a separate word by themselves.
- `wklv phvvdjh lv qrw wrr kdug wr euhdn`
- `T--- ------- -- -OT TOO ---- TO -----`
- The `-OT` could be `cot`, `dot`, `got`, `hot`, `lot`, `not`, `pot`, `rot` or `tot`. A likely choice is not. So `q` = `N`
- The word lv is also the end of the word wklv.
  - lv cannot be SO, because then wklv is T-SO. There is no such word
  - lv cannot be IN, because we have q = N
  - lv has to be IS, so wklv is THIS
- `wklv phvvdjh lv qrw wrr kdug wr euhdn`
- `THIS --SS--- IS NOT TOO H--- TO -----`

- By now, we should be able to figure out that the shift has been by three characters for each character in the plaintext. So, the plaintext for the given ciphertext is:
  - `wklv phvvdjh lv qrw wrr kdug wr euhdn`
  - `THIS MESSAGE IS NOT TOO HARD TO BREAK`

Cryptanalysis of Substitution Ciphers

- Some clues to break the code more quickly
  - The frequency with which certain letters are used can help us to break the code more quickly.
    - The letters E, T, O, A occur more often the letters J, Q, X, Z
  - The nature and context of the text being analyzed affects the distribution
    - In a medical article in which the term x-ray may be used often, the letter x would have an uncommonly high frequency
  - Letters appear to each other with predictable frequency
  - In usual English, EN, RE, ER,…, and ENT, ION, AND,… are most frequently-occurring coincident pairs (digrams) and triples (trigrams) of letters
  - Digrams and trigram frequencies are well-known for all written languages
  - Frequency distribution may not give complete decryption, due to peculiarities of plaintext, but considerably narrows down choices.

- Short messages give a cryptanalyst little to work with as the latter works by finding patterns (possible to obtain more with long messages). So, shorter messages are fairly more secure with simple encryption algorithms.

Useful English Language Statistics

Order and Frequency of Single Letters

E 12.31%	L 4.03%	B 1.62%
T 9.59	D 3.65	G 1.61
A 8.05	C 3.20	V 0.93
O 7.94	U 3.10	K 0.52
N 7.19	P 2.29	Q 0.20
I 7.18	F 2.28	X 0.20
S 6.59	M 2.25	J 0.10
R 6.03	W 2.03	Z 0.09
H 5.14	Y 1.88	

Letter Groups Percentages

A E I O U	38.58%
L N R S T	33.43%
J K Q X Z	1.11%
E T A O N	45.08%
E T A O N I S R H	70.02%

Order and Frequency of Leading DIGRAMS

TH	3.15%	TO	1.11%	SA	0.75%	MA	0.56%
HE	2.51	NT	1.10	HI	0.72	TA	0.56
AN	1.72	ED	1.07	LE	0.72	CE	0.55
IN	1.69	IS	1.06	SO	0.71	IC	0.55
ER	1.54	AR	1.01	AS	0.67	LL	0.55
RE	1.48	OU	0.96	NO	0.65	NA	0.54
ES	1.45	TE	0.94	NE	0.64	RO	0.54
ON	1.45	OF	0.94	EC	0.64	OT	0.53
EA	1.31	IT	0.88	IO	0.63	TT	0.53
TI	1.28	HA	0.84	RT	0.63	VE	0.53
AT	1.24	SE	0.84	CO	0.59	NS	0.51
ST	1.21	ET	0.80	BE	0.58	UR	0.49
EN	1.20	AL	0.77	DI	0.57	ME	0.48
ND	1.18	RI	0.77	LI	0.57	WH	0.48
OR	1.13	NG	0.75	RA	0.57	LY	0.47

Book Ciphers

- Book cipher is a variation of the well-known Vignere cipher
- The key comes from a text portion starting from a certain page of a book. Both the sender and receiver should have the same edition of the book.
- Consider encrypting the message `MACHINES CANNOT THINK`
- Using the Key: `i am i exist that is certain`
- The ciphertext is the character corresponding to the cell at the intersection of the row of the plaintext character and the column of the character in the key
- Cryptanalysis becomes difficult with more flatter frequency distribution.
- Encryption: $C_i \equiv P_i + K_i \pmod{26}$
- Decryption: $P_i \equiv C_i - K_i \pmod{26}$

- Example for Book Cipher: Use a character grouping of size 5
  - Plaintext: `MACHI NESCA NNOTT HINK`
  - Key: `iamie xistt hatis cert`
  - Ciphertext: `uaopm kmkvt unhbl jmed`

Cryptanalysis of Book Ciphers

- The probability that a given character in the plaintext is any one of E, A, O, T, N or I is close to 50%.
- Similarly, the probability that a given character in the key (taken from a book) is any one of E, A, O, T, N or I is close to 50%.
- To break the cipher, assume that each letter of the ciphertext comes from a situation in which the plaintext letter (row selector) and the key letter (column selector) are both one of the six most frequent letters.
- A sub-table of the Vigenere tableau table that lists the intersections between these six characters is given below:

	a	**e**	**o**	**t**	**n**	**i**
A	a	e	o	t	n	i
E	e	i	s	x	r	m
O	o	s	c	h	b	w
T	t	x	h	m	g	b
N	n	r	b	g	a	v
I	i	m	w	b	v	q

Cryptanalysis of Book Ciphers

- Searching through the sub-table for possibilities, we have:

– Ciphertext:	uaopm kmkvt unhbl jmed
– Possible Plaintexts:	?**A**A?E ?**E**?N**A** ?A**O**O? ?EA? NO T T IT **N**T**T** TE **I** I **I**
- Actual Plaintext:	M**A**CH**I** N**E**SC**A** N**NOT**T H**I**NK

- Out of the 25 predictions, 8 were correct. Hence, the overall percentage correctness in the predictions is 8/25 = 32%.

- Fraction of the plaintext characters correctly predicted = 8/19.

Example for Columnar Transposition

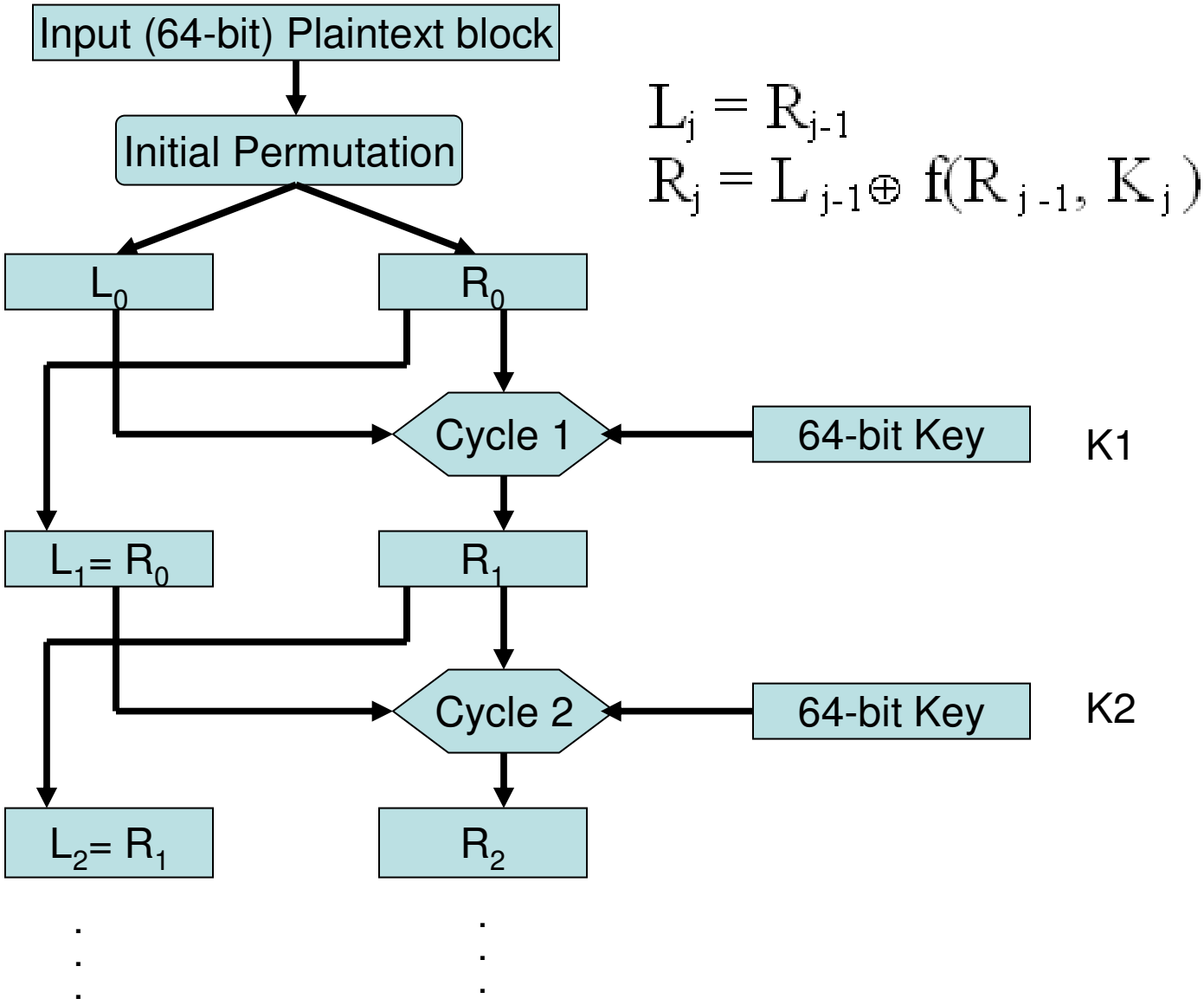
- Plain text: THIS IS A MESSAGE TO SHOW HOW COLUMNAR TRANSPOSITION WORKS
- Ciphertext: TSSOHLRSTOHAASOUTPIRIMGHWMROOKSEEOC NASNSISTWOANIWX
- At the receiver:
  - To figure out each column and the number of rows, the receiver divides the message length by the number of columns agreed upon.

T	H	I	S	I
S	A	M	E	S
S	A	G	E	T
O	S	H	O	W
H	O	W	C	O
L	U	M	N	A
R	T	R	A	N
S	P	O	S	I
T	I	O	N	W
O	R	K	S	X
1	2	3	4	5

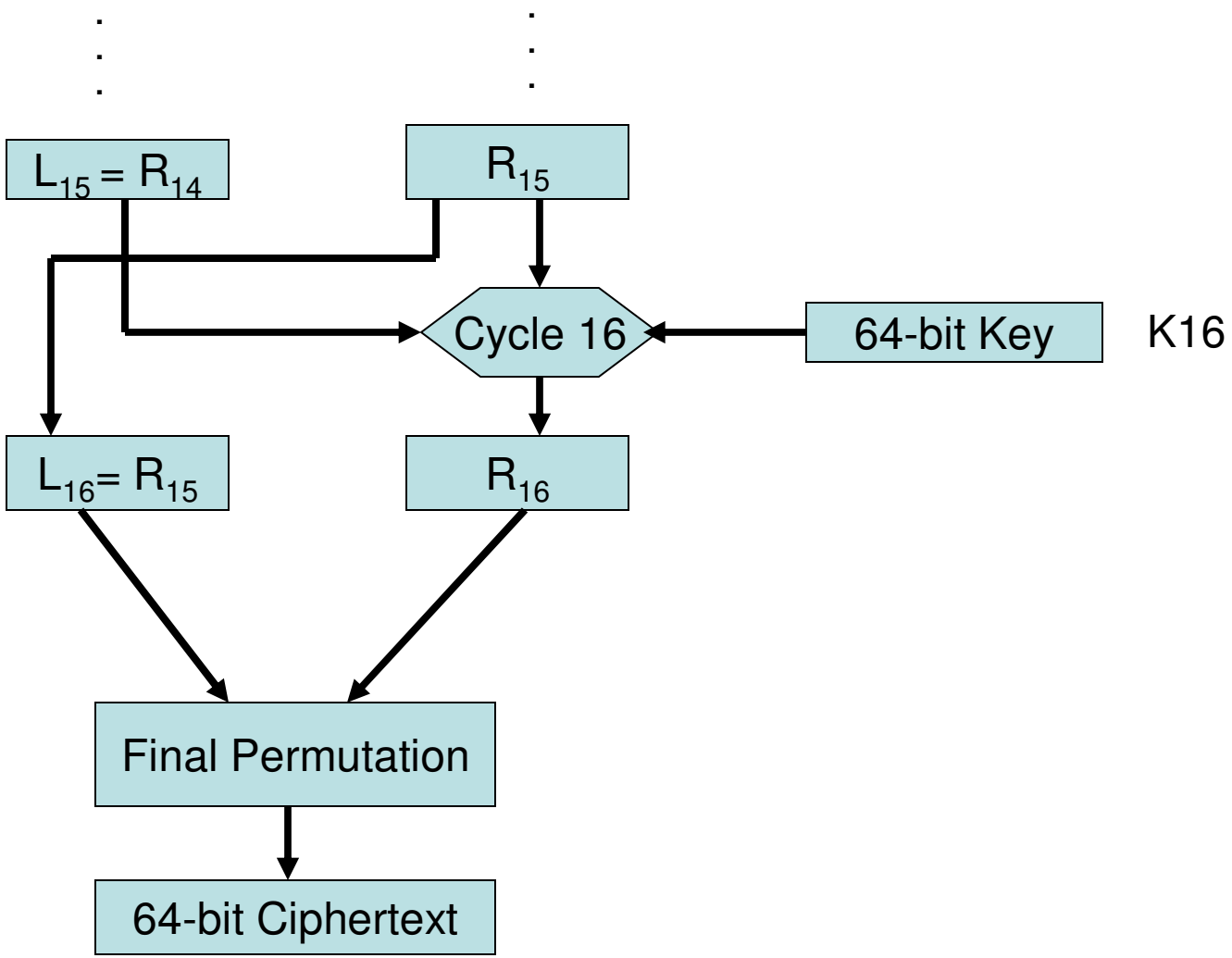
- To make it more secure, the sender and receiver could agree on a code word of length equal to the number of columns and then send the columns in the alphabetical order of the characters in the key word.
- Let the code word be ZEBRA. Then, the fifth column would be sent first, followed by the third column, followed by the second column and so on.
- Ciphertext: ISTWOANIWXIMGHWMROOKHAASOUTPIRSEEOC NASNSTSSOHLRSTO

Z	E	B	R	A
↓	↓	↓	↓	↓
T	H	I	S	I
S	A	M	E	S
S	A	G	E	T
O	S	H	O	W
H	O	W	C	O
L	U	M	N	A
R	T	R	A	N
S	P	O	S	I
T	I	O	N	W
O	R	K	S	X
5	3	2	4	1

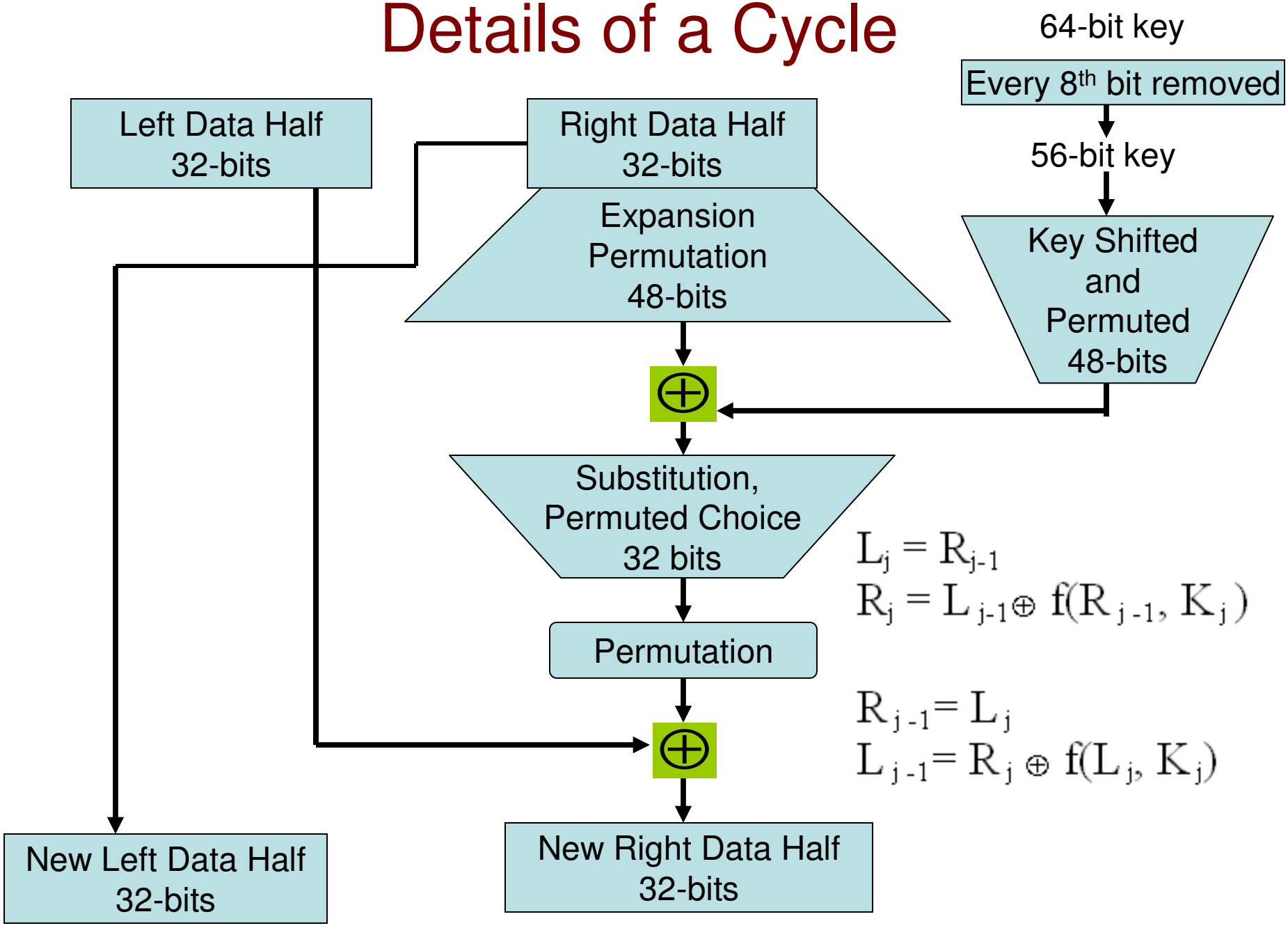
Data Encryption Standard (DES): Cycles of Substitution and Permutation



Cycles of Substitution and Permutation



Details of a Cycle



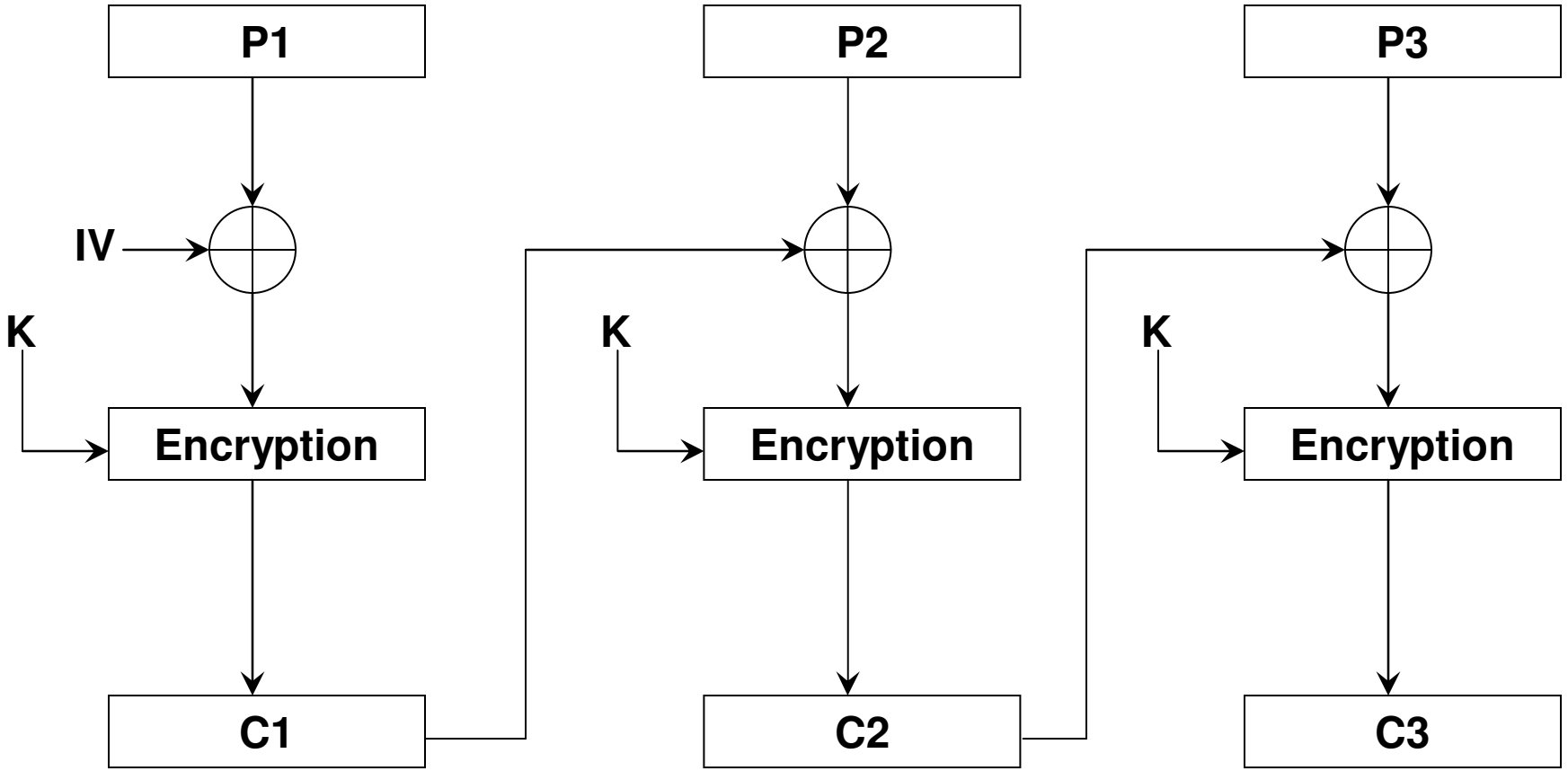
Double DES and Triple DES

- The DES algorithm is fixed for a 56-bit key.
- As the computing power has increased rapidly these days and hopefully will continue in the near future too, it may not be that time consuming to do an exhaustive search of all the 2^{56} keys, when an attacker gets a plaintext and the corresponding ciphertext.
  Double DES:
  - To encrypt: C = E(K2, E(P, K1))
  - To decrypt: P = D(K1, D(K2, C))
  - The encryption/ decryption algorithm used is DES.
- Triple DES:
  - To encrypt: C = E(K3, D(K2, E(K1, P)))
  - To decrypt: P = D(K1, E(K2, D(K3, C)))
  - The encryption/ decryption algorithm used is DES.
  - With 3 keys, 3DES uses 168-bits and is more robust; but, also slow.
  - 3DES has also been adopted for Internet applications like PGP, S/MIME.

Meet-in-the-Middle Attack with Double DES

- It is a known-plaintext attack where the <plaintext, ciphertext> pair and the encryption algorithm (DES) is known and the key(s) need to be determined.
  - $C = E_{K2}(E_{K1}(P))$
- Since $X = E_{K1}(P) = D_{K2}(C)$, the attack consists of encrypting P with all possible values of 56-bit keys (K_1) and storing the resulting X values. Similarly, we decrypt C with all possible values of 56-bit keys (K_2) and compare the resulting values for a match with the set obtained based on K_1. The 56-bit key values (K_1 and K_2) for which $E_{K1}(P) = D_{K2}(C)$, constitute the 112-bit key K_1 K_2.
- The time complexity for cryptanalysis is thus $O(2^{56})$ and not $O(2^{112})$.

Message Authentication Code (CBC: Cipher Block Chaining)



CBC for Data Integrity and Message Authentication

- A ciphertext block depends on all blocks before it
- Any change to a plaintext block affects all of the succeeding ciphertext blocks – creates an Avalanche effect. This property can be used to compute a "Message Authentication Code" (MAC) for the entire plaintext and sent as part of the message.
- If the "integrity" of the message is the only required criterion, then we can send P1, P2, …, P_{last_block}, MAC.
  - If any intruder changes any of the plaintext, the Avalanche Effect property of CBC requires that the MAC value computed by the destination to be different than what is sent by the sender as part of the message.

- The encryption Key K and the Initialization Vector are the secret keys to be known only to the sender and receiver.

One-way Hash Function

- A hash function accepts a variable-size message *M* as input and produces a fixed-size message digest H(*M*) as output.
- Unlike a MAC, a hash function does not take a secret key as input;
  - Used for integrity check
  - Combined with encryption for authentication check
- The length of the message (in bits) is padded along with the message to compute the hash value. This is to make it complicated for an attacker to come up with a message of the same hash value.
- The hash value is a “finger print” of the file, message or block of data.

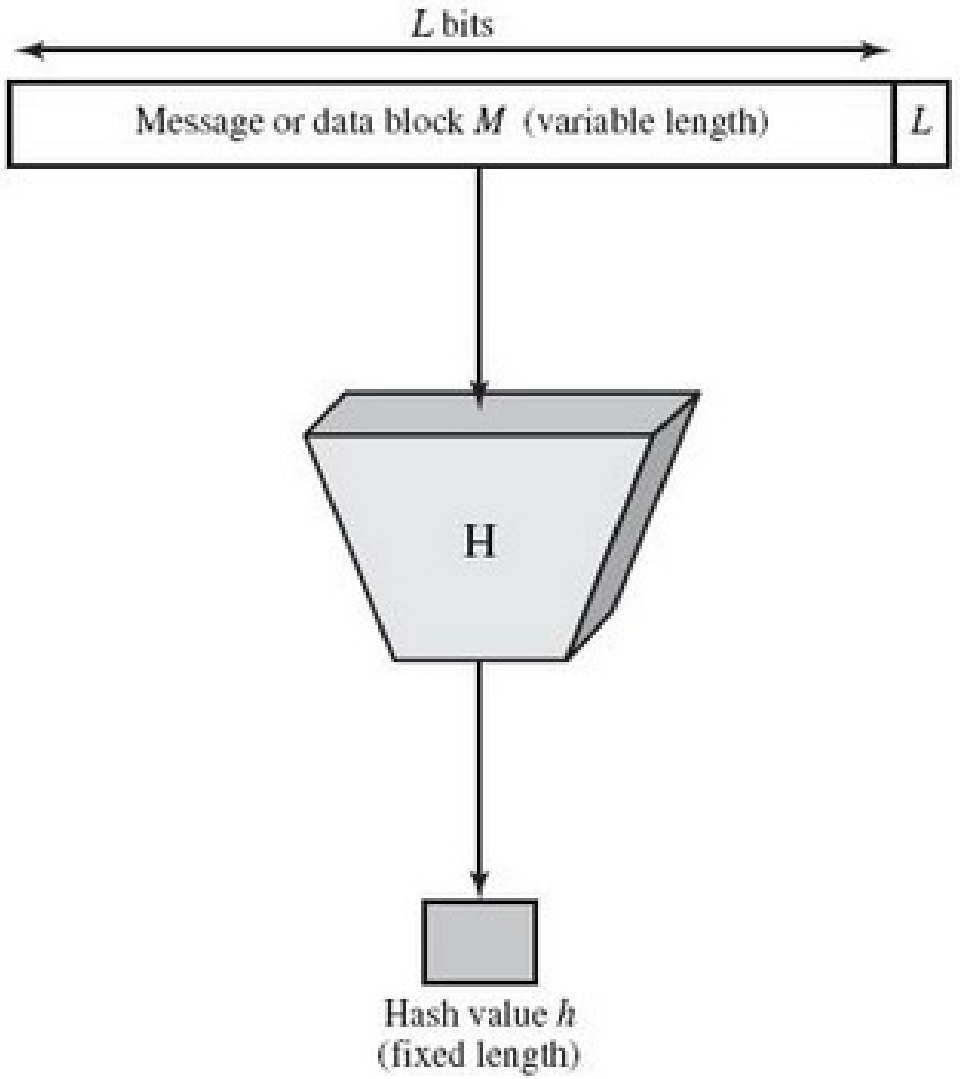


Figure 2.5: W. Stallings: Computer Security: Principles and Practice

Secure Hash Function: Requirements

- To be useful for data integrity, a hash function *H* must have the following properties:
  - 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.
  - One-way property: Given a hash value *h*, it is computationally infeasible to compute the underlying message x such that H(x) = h.
  - Weak-collision resistant: For any given block x, it is computationally infeasible to find another block y, where y ≠ x and H(y) = H(x).
  - Strong-collision resistant: It is computationally infeasible to find any pair of blocks x and y, such that y ≠ x and H(y) = H(x).
- Hash functions that satisfy the first five properties (listed above) are said to be *weak hash functions*. Hash functions that satisfy all of the above properties are said to be *strong hash functions*.
- Secure Hash Algorithm (SHA) and its variants (SHA-256, 384, 512) are the commonly used hash functions.
- Other uses: (1) Store passwords for operating systems; (2) Periodically compute/ verify the hash values of files; the hash values are stored in a secure location or disc.

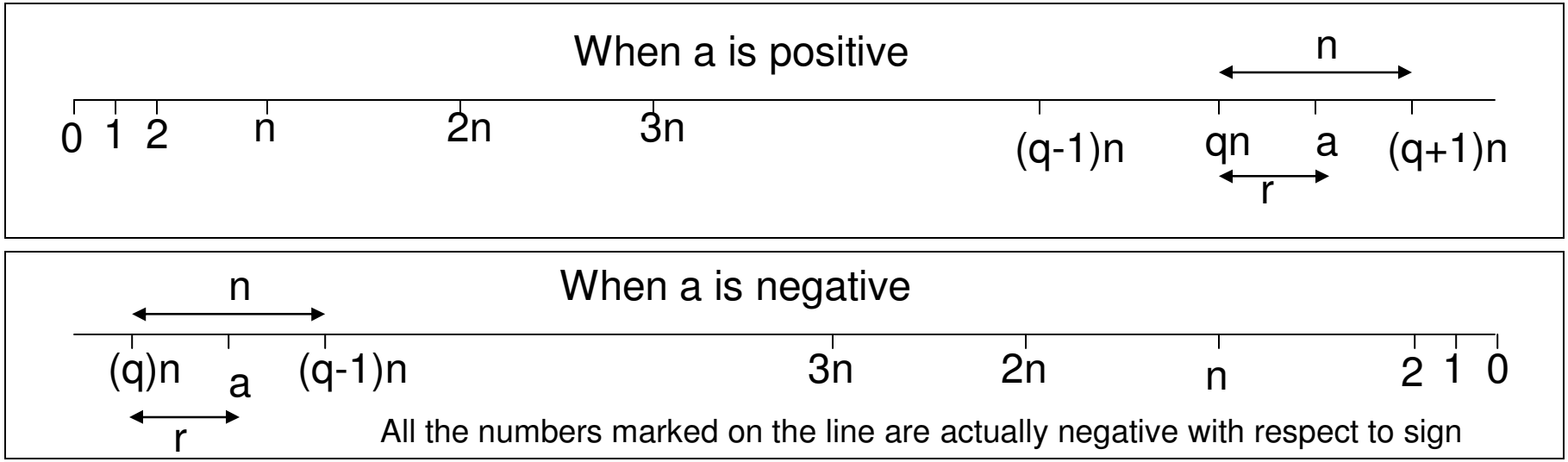
Public Key Encryption

- Motivation: Key distribution problem of symmetric encryption system
- Let K_{PRIV} and K_{PUB} be the private key and public key of a user. Then,
  - $P = D(K_{PRIV}, E(K_{PUB}, P))$
  - With some, public key encryption algorithms like RSA, the following is also true: $P = D(K_{PUB}, E(K_{PRIV}, P))$
- In a system of n users, the number of secret keys for point-to-point communication is $n(n-1)/2 = O(n^2)$. With the public key encryption system, we need 2 keys (one public and one private key) per user. Hence, the total number of keys needed is $2n = O(n)$.

	Secret Key (Symmetric)	Public Key (Asymmetric)
Number of Keys	1	2
Protection of Key	Must be secret	One key must be secret; the key can be publicly exposed
Best uses	Cryptographic workhorse; secrecy and integrity of data	Key exchange, authentication
Key distribution	Must be out-of-band	Public key can be used to distribute other keys
Speed	Fast	Slow

Modular Arithmetic

- Given any positive integer n and any integer a, if we divide a by n, we get a quotient q and a remainder r that obey the following relationship:
  - a = q * n + r, 0 ≤ r < n and r is the remainder, q is the quotient



  - Example:
    - a = 59; n = 7; 59 = (8)*7 + 3 r = 3; q = 8
    - a = -59; n = 7; -59 = (-9)*7 + 4 r = 4; q = -9
    - 59 mod 7 = 3
    - -59 mod 7 = 4

Modular Arithmetic

- Two integers a and b are said to be congruent modulo n, if a mod n = b mod n. This is written as $a \equiv b \bmod n$.
  - We say "a and b are equivalent to each other in class modulo n"

- Example:
  - $73 \equiv 4 \bmod 23$, because $73 \bmod 23 = 4 = 4 \bmod 23$
  - $21 \equiv -9 \bmod 10$, because $21 \bmod 10 = 1 = -9 \bmod 10$

- Properties of the Modulo Operator
  - If $a \equiv b \bmod n$, then $(a - b) \bmod n = 0$
  - If $a \equiv b \bmod n$, then $b \equiv a \bmod n$
  - If $a \equiv b \bmod n$ and $b \equiv c \bmod n$, then $a \equiv c \bmod n$

- Example:
  - $73 \equiv 4 \bmod 23$, then $(73 - 4) \bmod 23 = 0$
  - $73 \equiv 4 \bmod 23$, then $4 \equiv 73 \bmod 23$, because $4 \bmod 23 = 73 \bmod 23$
  - $73 \equiv 4 \bmod 23$ and $4 \equiv 96 \bmod 23$, then $73 \equiv 96 \bmod 23$.

Modular Arithmetic

- Now, that we have studied the meaning of "equivalency" or "congruent modulo n", it is see that the "mod n" operator maps "all integers" (negative and positive) into the set of integers [0, 1, …., n-1].
- Example: Class of modulo 15

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
-60	-59	-58	-57	-56	-55	-54	-53	-52	-51	-50	-49	-48	-47	-46
-45	-44	-43	-42	-41	-40	-39	-38	-37	-36	-35	-34	-33	-32	-31
-30	-29	-28	-27	-26	-25	-24	-23	-22	-21	-20	-19	-18	-17	-16
-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
45	46	47	48	49	50	51	52	53	54	55	56	57	58	59
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…

- From the above table, we could say things like
  - -38 ≡ 22 mod 15 — 24 ≡ 54 mod 15
  - -38 mod 15 = 7 [-38 = (-3)*15 + 7] — 24 mod 15 = 9 [24 = (1)*15 + 9]
  - 22 mod 15 = 7 [22 = (1)*15 + 7] — 54 mod 15 = 9 [54 = (3)*15 + 9]

Multiplicative Inverse Modulo n

- If (a * b) modulo n = 1, then
  - a is said to be the multiplicative inverse of b in class modulo n
  - b is said to be the multiplicative inverse of a in class modulo n
- Example:
  - Find the multiplicative inverse of 7 in class modulo 15
  - Straightforward approach:
    - Multiply 7 with all the integers [0, 1, …, 14] in class modulo 15
    - There will be only one integer x for which (7*x) modulo 15 = 1

X	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
(7 * X) modulo 15	0	7	14	6	13	5	12	4	11	3	10	2	9	1	8

  - Find the multiplicative inverse of 9 in class modulo 13
    - Multiply 9 with all the integers [0, 1, …, 12] in class modulo 13
    - There will be only one integer x for which (9*x) modulo 13 = 1

X	0	1	2	3	4	5	6	7	8	9	10	11	12
(9 * X) modulo 13	0	9	5	1	10	6	2	11	7	3	12	8	4

- A more efficient approach to find multiplicative inverse in class modulo n is to use the Extended Euclid Algorithm

Example for Modular Exponentiation

- To compute 5^{41} mod 9
  - Straightforward approach:
    - 5^{41} mod 9 = (4547473508864641189575195312 5) mod 9 = 2
    - Number of multiplications - 40
  - Using the Right-to-Left Binary Algorithm
    - Write 41 in binary: 101001
    - $5^{41} = 5^{32} * 5^{8} * 5^{1}$

32	16	8	4	2	1
1	0	1	0	0	1

$5^1 \bmod 9 = 5 \bmod 9 = 5$
$5^2 \bmod 9 = (5^1 * 5^1) \bmod 9 = (5 \bmod 9 * 5 \bmod 9) \bmod 9 = (5 * 5) \bmod 9 = 25 \bmod 9 = 7$
$5^4 \bmod 9 = (5^2 * 5^2) \bmod 9 = (5^2 \bmod 9 * 5^2 \bmod 9) \bmod 9 = (7 * 7) \bmod 9 = 49 \bmod 9 = 4$
$5^8 \bmod 9 = (5^4 * 5^4) \bmod 9 = (5^4 \bmod 9 * 5^4 \bmod 9) \bmod 9 = (4 * 4) \bmod 9 = 16 \bmod 9 = 7$
$5^{16} \bmod 9 = (5^8 * 5^8) \bmod 9 = (5^8 \bmod 9 * 5^8 \bmod 9) \bmod 9 = (7 * 7) \bmod 9 = 49 \bmod 9 = 4$
$5^{32} \bmod 9 = (5^{16} * 5^{16}) \bmod 9 = (5^{16} \bmod 9 * 5^{16} \bmod 9) \bmod 9 = (4 * 4) \bmod 9 = 16 \bmod 9 = 7$

$$
\begin{aligned}
5^{41} \bmod 9 &= (5^{32} * 5^{8} * 5^{1}) \bmod 9 \\
&= (7 * 7 * 5) \bmod 9 \\
&= ((49 \bmod 9) * (5 \bmod 9)) \bmod 9 \\
&= (4 * 5) \bmod 9 \\
&= 20 \bmod 9 \\
&= 2
\end{aligned}
$$

Number of multiplications: 5 + 2 = 7

Multiplications is dependent on the # bits in the binary representation of the exponent n

Multiplications = Θ(logn)

Example for Modular Exponentiation

- To compute 8^{35} mod 11
  - Straightforward approach:
    - 8^{35} mod 11 = (40564819207303340847894502572032) mod 11 = 10
    - Number of multiplications - 34

32	16	8	4	2	1
1	0	0	0	1	1

$8^1 \bmod 11 = 8$

$8^2 \bmod 11 = (8^1 \times 8^1) \bmod 11 = (8 \times 8) \bmod 11$
$= 64 \bmod 11 = 9.$

$8^4 \bmod 11 = (8^2 \times 8^2) \bmod 11 = (9 \times 9) \bmod 11$
$= 81 \bmod 11 = 4.$

$8^8 \bmod 11 = (8^4 \times 8^4) \bmod 11 = (4 \times 4) \bmod 11$
$= 16 \bmod 11 = 5.$

$8^{16} \bmod 11 = (8^8 \times 8^8) \bmod 11 = (5 \times 5) \bmod 11$
$= 25 \bmod 11 = 3$

$8^{32} \bmod 11 = (8^{16} \times 8^{16}) \bmod 11 = (3 \times 3) \bmod 11 = 9$

$8^{35} \bmod 11 = (8^{32} \times 8^2 \times 8^1) \bmod 11$
$= (9 \times 9 \times 8) \bmod 11 = (81 \times 8) \bmod 11$
$= ((81 \bmod 11) \times 8) \bmod 11$
$= (4 \times 8) \bmod 11 = 10$

Multiplications = 2 + 5 = 7

RSA Algorithm

- Given the two keys (e, n) and (d, n).
- The two keys are e and d are related as follows:
  - d is the multiplicative inverse of e in the class modulo (p-1)*(q-1), where n = p*q; also, p and q are prime integers
  - The complexity of breaking RSA lies in the fact that p, q, n are large integers of the order of 100 digits, 200 digits; it becomes difficult to factorize a large integer into two prime factors
- A plaintext message P is encrypted to ciphertext by:
  - $C = P^e \bmod n$
- The plaintext is recovered by:
  - $P = C^d \bmod n$
- Because of symmetry in modular arithmetic, encryption and decryption are mutual inverses and commutative. Therefore,
  - $P = C^d \bmod n = (P^e)^d \bmod n = (P^d)^e \bmod n$
- Thus, one can apply the encrypting transformation first and then the decrypting one, or the decrypting transformation first followed by the encrypting one.
- One of the two keys is known publicly and the other one is known only to the user.

Another Example for RSA Algorithm

- Let the encryption and decryption keys be (13, 391) and (325, 391) respectively. Show the encryption and decryption for Plaintext 127

8	4	2	1
1	1	0	1

- Encryption for Plaintext P = 127
- Ciphertext C = $P^e \bmod n$

$= 127^{13} \bmod 391$

$127^1 \bmod 391 = 127 \bmod 391 = 127$

$127^2 \bmod 391 = (127^1 * 127^1) \bmod 391 = (127 \bmod 391 * 127 \bmod 391) \bmod 391 = (127 * 127) \bmod 391 = 16129 \bmod 391 = 98$

$127^4 \bmod 391 = (127^2 * 127^2) \bmod 391 = (127^2 \bmod 391 * 127^2 \bmod 391) \bmod 391 = (98 * 98) \bmod 391 = 9604 \bmod 391 = 220$

$127^8 \bmod 391 = (127^4 * 127^4) \bmod 391 = (127^4 \bmod 391 * 127^4 \bmod 391) \bmod 391 = (220 * 220) \bmod 391 = 48400 \bmod 391 = 307$

$127^{13} \bmod 391 = (127^8 * 127^4 * 127^1) \bmod 391$

$= (307 * 220 * 127) \bmod 391$

$= (((307 * 220) \bmod 391) * (127)) \bmod 391$

$= (((67540) \bmod 391) * (127)) \bmod 391$

$= ((288) * (127)) \bmod 391$

$= (36576) \bmod 391$

$= 213$

Ciphertext is 213

Another Example for RSA Algorithm

- Decryption for Ciphertext C = 213
- Plaintext $P = C^d \bmod n$

$= 213^{325} \bmod 391$

256	128	64	32	16	8	4	2	1
1	0	1	0	0	0	1	0	1

$213^1 \bmod 391 = 213 \bmod 391 = 213$

$213^2 \bmod 391 = (213 * 213) \bmod 391 = 45369 \bmod 391 = 13$

$213^4 \bmod 391 = (13 * 13) \bmod 391 = 169 \bmod 391 = 169$

$213^8 \bmod 391 = (169 * 169) \bmod 391 = 28561 \bmod 391 = 18$

$213^{16} \bmod 391 = (18 * 18) \bmod 391 = 324 \bmod 391 = 324$

$213^{32} \bmod 391 = (324 * 324) \bmod 391 = 104976 \bmod 391 = 188$

$213^{64} \bmod 391 = (188 * 188) \bmod 391 = 35344 \bmod 391 = 154$

$213^{128} \bmod 391 = (154 * 154) \bmod 391 = 23716 \bmod 391 = 256$

$213^{256} \bmod 391 = (256 * 256) \bmod 391 = 65536 \bmod 391 = 239$

$$\begin{aligned} 213^{325} \bmod 391 &= (213^{256} * 213^{64} * 213^4 * 213^1) \bmod 391 \\ &= (239 * 154 * 169 * 213) \bmod 391 \\ &= (52 * 169 * 213) \bmod 391 \\ &= (186 * 213) \bmod 391 \\ &= 127 \end{aligned}$$

Plaintext is 127

Applications of Public-Key Encryption

- Diffie-Hellman Key Exchange
  - Used to allow two parties that have to establish a shared secret key over an insecure communication channel.
  - Alice and Bob agree on a field size n and a starting number g.
  - Alice generates a secret integer a and sends g^a mod n to Bob. Alice sends this encrypted using its private key, so that Bob can decrypt it using Alice's public key, thereby authenticating that the message came from Alice. $E(K_{PRI-ALICE}, g^a \text{ mod } n)$
  - At the same time, Bob generates a secret integer b and sends g^b mod n to Alice. Bob sends this encrypted using its private key, thereby authenticating to Alice that the message came from Bob. $E(K_{PRI-Bob}, g^b \text{ mod } n)$
  - When Bob gets Alice's message, it computes $(g^a)^b$ mod n and uses it as the secret key.
  - Similarly, when Alice gets Bob's message, it computes $(g^b)^a$ mod n and uses it as the secret key.
  - According to Modular arithmetic, $(g^a)^b \text{ mod } n = (g^b)^a \text{ mod } n$. Hence, both Alice and Bob have agreed on a shared secret key.

Example for Diffie-Hellman Key Exchange

- Assume the secret integers used by Alice and Bob to be 15 and 29 respectively. The values of *g* and *n* are 13 and 45 respectively. What would be the secret key they will be agreeing with?

g = 13; n = 45; a = 15; b = 29

Alice Side

	8	4	2	1
15 is:	1	1	1	1

Compute $g^a \bmod n = 13^{15} \bmod 45$

$13^1 \bmod 45 = 13$
$13^2 \bmod 45 = (13^1 \bmod 45 * 13^1 \bmod 45) = 169 \bmod 45 = 34$
$13^4 \bmod 45 = (13^2 \bmod 45 * 13^2 \bmod 45) = (34*34) \bmod 45 = 31$
$13^8 \bmod 45 = (13^4 \bmod 45 * 13^4 \bmod 45) = (31*31) \bmod 45 = 16$

$13^{15} \bmod 45 = (13^8 * 13^4 * 13^2 * 13^1) \bmod 45 = (16 * 31 * 34 * 13) \bmod 45$
$= (1 * 34 * 13) \bmod 45$
$= 37$

Example for Diffie-Hellman Key Exchange (continued…)

g = 13; n = 45; a = 15; b = 29

	16	8	4	2	1
29 is:	1	1	1	0	1

Alice sends 37 to Bob

Bob computes $(g^a \bmod n)^b \bmod n = 37^{29} \bmod 45$

$37^1 \bmod 45 = 37$
$37^2 \bmod 45 = (37^1 \bmod 45 * 37^1 \bmod 45) = 19$
$37^4 \bmod 45 = (37^2 \bmod 45 * 37^2 \bmod 45) = (19*19) \bmod 45 = 1$
$37^8 \bmod 45 = (37^4 \bmod 45 * 37^4 \bmod 45) = (1*1) \bmod 45 = 1$
$37^{16} \bmod 45 = (37^8 \bmod 45 * 37^8 \bmod 45) = (1*1) \bmod 45 = 1$

$37^{29} \bmod 45 = (37^{16} * 37^8 * 37^4 * 37^1) \bmod 45 = (1 * 1 * 1 * 37) \bmod 45$
$= 37$

Example for Diffie-Hellman Key Exchange (continued…)

- Assume the secret integers used by Alice and Bob to be 15 and 29 respectively. The values of *g* and *n* are 13 and 45 respectively. What would be the secret key they will be agreeing with?

g = 13; n = 45; a = 15; b = 29

Bob Side

	16	8	4	2	1
29 is:	1	1	1	0	1

Compute $g^b \bmod n = 13^{29} \bmod 45$

$13^1 \bmod 45 = 13$
$13^2 \bmod 45 = (13^1 \bmod 45 * 13^1 \bmod 45) = 19$
$13^4 \bmod 45 = (13^2 \bmod 45 * 13^2 \bmod 45) = (19*19) \bmod 45 = 1$
$13^8 \bmod 45 = (13^4 \bmod 45 * 13^4 \bmod 45) = (1*1) \bmod 45 = 1$
$13^{16} \bmod 45 = (13^8 \bmod 45 * 13^8 \bmod 45) = (1*1) \bmod 45 = 1$

$13^{29} \bmod 45 = (13^{16} * 13^8 * 13^4 * 13^1) \bmod 45 = (1 * 1 * 1 * 13) \bmod 45$
$= 13$

Example for Diffie-Hellman Key Exchange (continued…)

g = 13; n = 45; a = 15; b = 29

	8	4	2	1
15 is:	1	1	1	1

Bob sends 13 to Alice

Alice computes $(g^b \bmod n)^a \bmod n = 13^{15} \bmod 45$

$13^1 \bmod 45 = 13$
$13^2 \bmod 45 = (13^1 \bmod 45 * 13^1 \bmod 45) = 169 \bmod 45 = 34$
$13^4 \bmod 45 = (13^2 \bmod 45 * 13^2 \bmod 45) = (34*34) \bmod 45 = 31$
$13^8 \bmod 45 = (13^4 \bmod 45 * 13^4 \bmod 45) = (31*31) \bmod 45 = 16$

$13^{15} \bmod 45 = (13^8 * 13^4 * 13^2 * 13^1) \bmod 45 = (16 * 31 * 34 * 13) \bmod 45$
$= (1 * 34 * 13) \bmod 45$
$= 37$

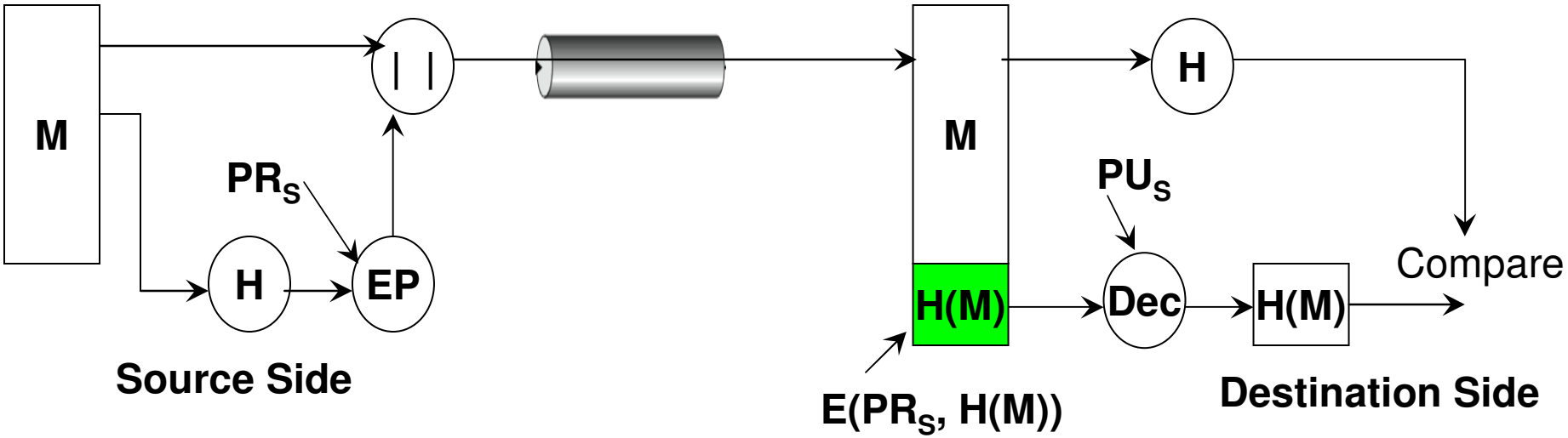
37 is the **session key** agreed upon by both sides

Public-Key Encryption

- Let K_{PRIV} and K_{PUB} be the private key and public key of a user. Then,
  - $P = D(K_{PRIV}, E(K_{PUB}, P))$
  - $P = D(K_{PUB}, E(K_{PRIV}, P))$
- Exchange of Secret Message using Asymmetric Encryption
  - Let K_{PUB-S}, K_{PRI-S} denote the public and private keys of Sender S. Similarly, let K_{PUB-R} and K_{PRI-R} be the public and private key of Receiver R. Let M be the secret message to be sent from S to R.
  - S sends to R the following:
    - $E\ (K_{PUB-R}\ E(K_{PRI-S}, M)\ )$
  - The inner encryption guarantees that the secret message M came from S and the outer encryption guarantees that only the receiver R could open the outer encryption of the message and get access to the inner encryption.

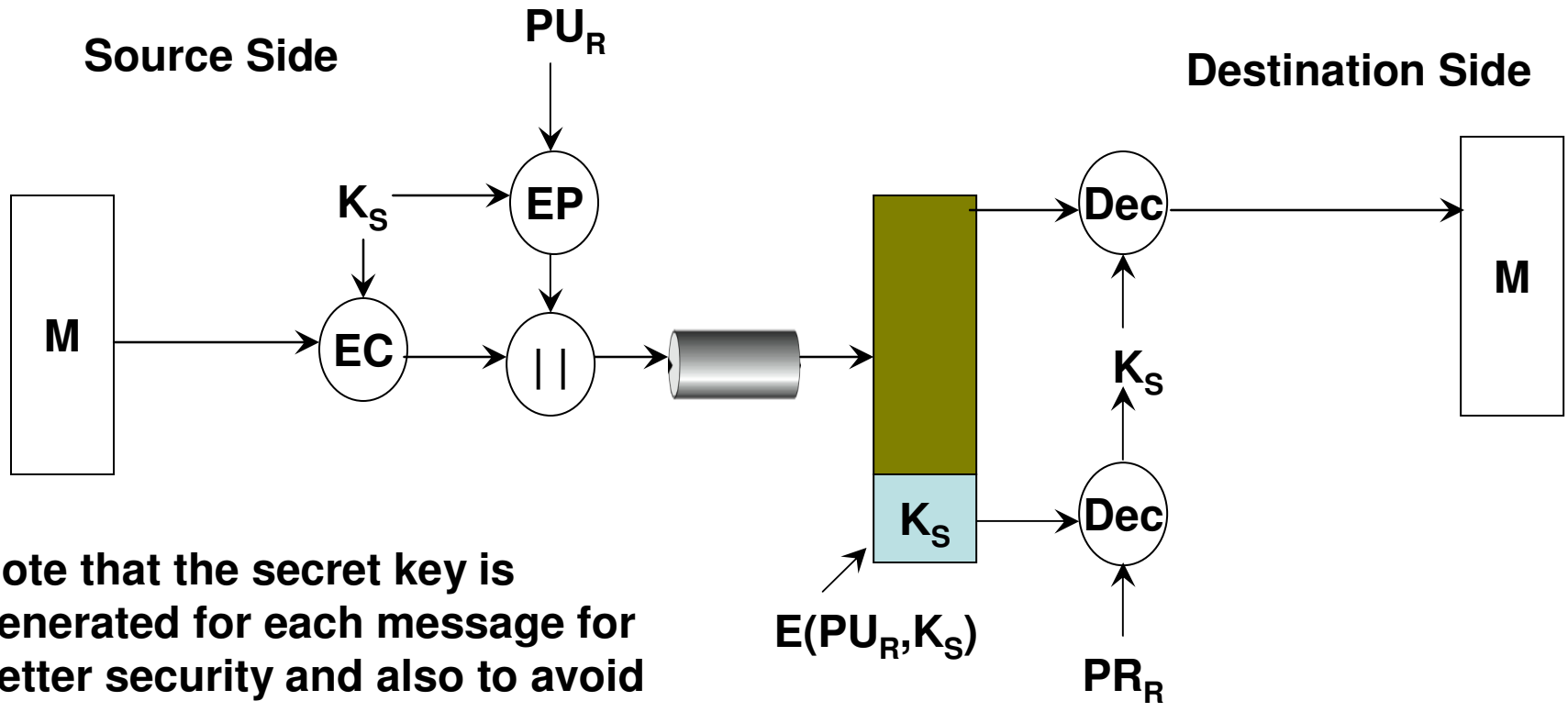
Use of Public-Key Encryption to Provide Integrity and Authentication

- $M \,||\, E_{Pri\text{-}S}(Hash(M))$



Use of Public-Key Encryption to Provide Confidentiality

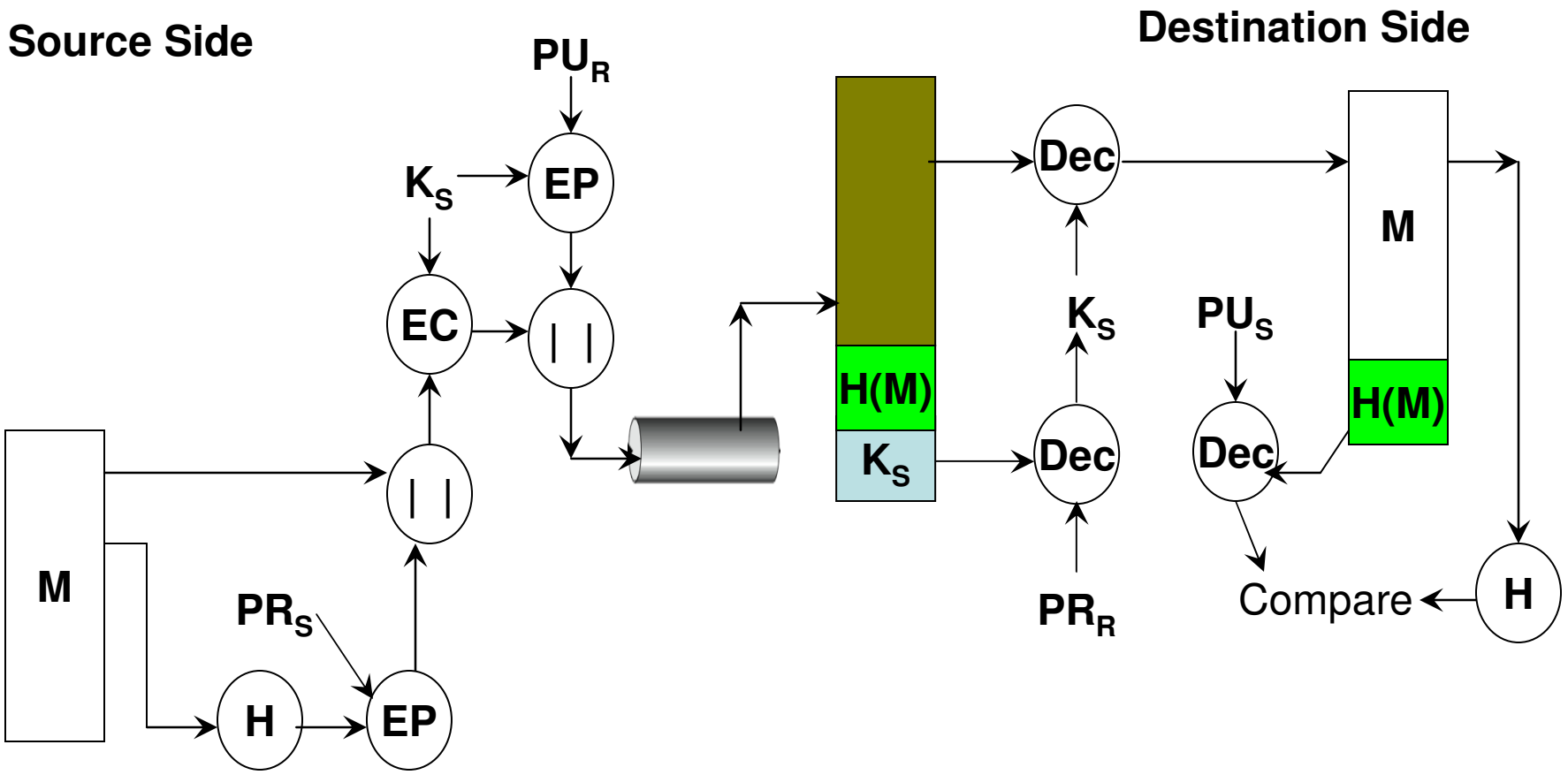
- $E_{Secret\text{-}Key}(M) \,||\, E_{Pub\text{-}R}(Secret\text{-}key)$



Note that the secret key is generated for each message for better security and also to avoid the need for key distribution.

Use of Public-Key Encryption to Provide Confidentiality, Integrity and Authentication

$E_{Secret\text{-}Key}(M \,||\, E_{Pri\text{-}S}(Hash\ (M)\ )\ )\ ||\ E_{Pub\text{-}R}(Secret\text{-}key)$



Man-in-the-Middle Attack

- Man-in-the-middle (MITM) attack is an attack in which an attacker is able to read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised.
- The attacker must be able to observe and intercept messages going between the two victims.
- Example: (MITM attack on public-key cryptography)
  - Suppose Alice wishes to communicate with Bob.
  - Mallory wants to eavesdrop their conversation or also possibly deliver a false message to Bob.
  - First, Alice must ask Bob for his public key.
  - If Bob sends his public key to Alice, but Mallory is able to intercept it, a MITM attack can begin.
  - Mallory sends a forged message to Alice that claims to have come from Bob, but contains Mallory's public key
  - Alice believes the public key received to be that of Bob's. So, Alice encrypts the message she wishes to send to Bob using the public key received and transmits on the link to Bob.
  - Mallory could now intercept the message, decrypt it with his private key and get the actual contents of the message.
  - Mallory now again encrypts the message (could be even altered too) with Bob's public key and transmits the message to Bob.
  - Bob on receiving the message, decrypts the message with his private key and reads the contents of the message assuming it came from Alice

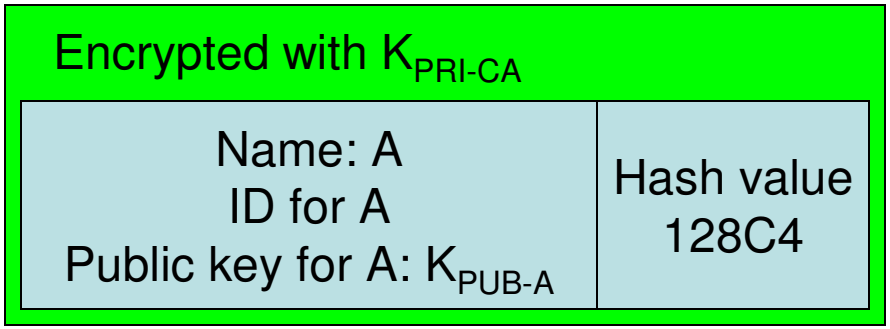
Public-Key Certificates

- Each of us adopt a “trust threshold” – a degree to which we are willing to believe an unidentified individual.
- We will use the concept of “vouching for” by a third party as the basis of thrust in settings where two parties do not know about each other.

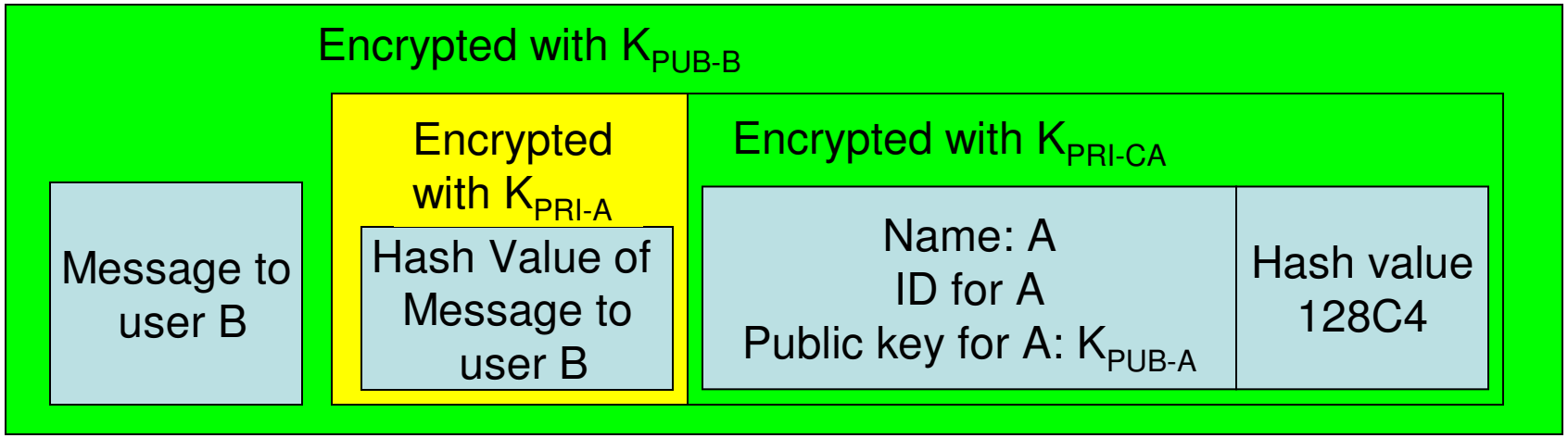
- Certification Authority (CA): Is an entity that issues digital certificates that contain a public key and the identity of the owner.
- The CA attests that the public key contained in the digital certificate belongs to the person (CA is a sort of digital notary).

Certificates

Digital Certificate for the Public Key of A



User A sending to user B



Note: The certificates are created and formatted based on the X.509 standard, which outlines the necessary fields of a certificate and the possible values that can be inserted into these fields. The latest X.509 version is v.3.