# *Computer Security*

- Secure the Network
  - Stop the hackers from getting in
    - Cryptographic service for passwords
    - Physical security
- Secure the network traffic
  - Provide encryption for messages
  - Who can you trust?



# Security Issues

- **Confidentiality**
  - Only authorised people should have access to information held on a computer equipment or other electronic devices. The Data Protection Act covers this area. Different countries have different policies.
- **Authentication**
  - Provides correct identification of the source of a message which is verifiable and reliable.
- **Integrity**
  - Only authorised people with the correct access privileges should have access to viewing, altering delaying or filtering data held or transmitted in an information environment.

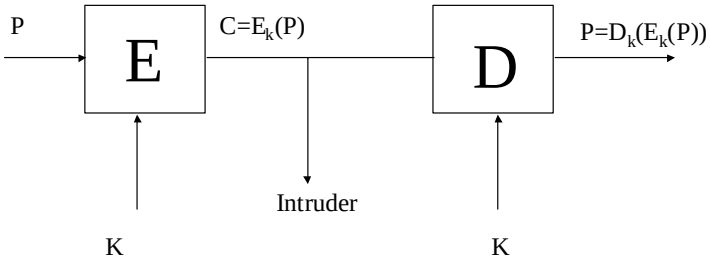
- **Nonrepudiation**
  - Neither the sender or receiver of information may be able to deny that a transmission took place. Useful for vendors of information or financial services and their clients.
- **Access Control**
  - Only authorised people may access a communications medium or information stored on an information system.
- **Availability**
  - Information and media should be available to authorised people when needed.
- **Legal Issues**
  - Many countries in Europe do not even allow the transmission of encrypted data, it may be a criminal offence to be involved in such activities, so check before sending encrypted e-mail and such. The US does not allow the export of cryptographic software or hardware, they regard such systems to be armaments, with severe penalties for infringements.



# Cryptology

- Cryptography: Devising codes (D & E)
- Cryptanalysis: Breaking codes.
- Cryptanalyst works with ….
  - The Ciphertext only problem - the hardest
  - The known plaintext problem - some matched plaintext & Ciphertext is available
  - Chosen plaintext problem - arbitrary amounts of chosen plaintext & ciphertext is available.

# Features of Cryptographic Systems




P: Plaintext
C: Ciphertext
E: Encryption method
D: Decryption method
K: Key



# Substitution Ciphers

- Replace each letter of the alphabet by another one.
- The oldest cipher (Caesar cipher) replaces A by D, B by E, C by F ..... etc., so each letter is shifted 3 places to give the ciphertext.
- The key (K) is then K=3 (shifts)

# Monoalphabetic Substitution

- The number of possible keys is then 26! = $4*10^{26}$.
- This may appear safe, but it is not !
- Language exhibits statistical properties, look for letters, digrams (2 letters), trigrams (3 letters) and words.



# Polyalphabetic Ciphers

- Vignere cipher which uses multiple Caesar type ciphers with a variable value of K. An easily remembered phrase could supply the key

B R E A D A N D B U T T E R B R E A D A N
W E H O L D T H E S E T R U T H S T O B E

# Transposition Ciphers

- Reorder letters but do not otherwise change them

| Key | 7 | 4 | 5 | 1 | 2 | 8 | 3 | 6 |
|---|---|---|---|---|---|---|---|---|
| | P | L | E | A | S | E | T | R |
| | A | N | S | F | E | R | O | N |
| | E | M | I | L | L | I | O | N |
| | D | O | L | L | A | R | S | T |
| | O | M | Y | A | N | S | B | A |
| | C | H | E | R | A | C | C | O |
| | U | N | T | S | I | X | T | W |
| | O | T | W | O | B | L | A | B |

- This yields the ciphertext
  AFLLSRSOSELANAIBTOOSBCTALNMOMHNT.....etc....



# Product Ciphers

- Consist of multiple substitution and transposition type ciphers back-to-back, working on the binary representation of letters (ASCII)
- The data Encryption Standard (**DES**) utilises a product cipher system
- DES however can be broken because of the small key size (56 bits conventionally).

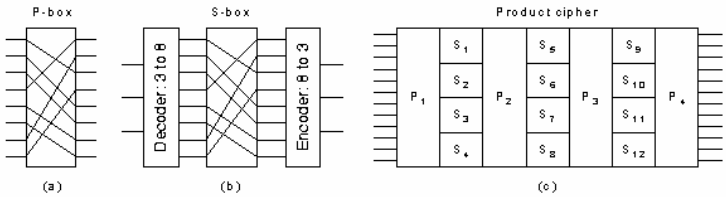



Fig. 7-4. Basic elements of product ciphers. (a) P-box. (b) S-box. (c) Product.

# Key Distribution

- How do two computer users agree upon a secret DES key, in a broadcast network environment, and possibly in the presence of eavesdroppers?
- By using **Diffie-Hellman Key Exchange** or **Exponential Key Exchange**!

# Exponential Key Exchange

- Alice chooses a random large integer x and computes
  - $X=g^x$ MOD n
- Bob chooses a random large integer y and computes
  - $Y=g^y$ MOD n
- Alice sends X to Bob, and Bob sends Y to Alice. They each however keep x and y secret.
- Alice computes $k=Y^x$ MOD n and
- Bob computes $k'=X^y$ MOD n
- Now..…
  - $k = k' = g^{xy}$ MOD n



- The intruder knows n, g, X and Y, but to no avail. The required calculation of x and y is **VERY DIFFICULT**. This is called the discrete logarithm problem, i.e. its easy to raise some number to a power, but not easy to find out to what power the number was raised to get the result.

# Example

- Example:
  - Given $17 = 10^x \text{ MOD } 89$
  - Find x
- NOTE:
  - Generating 200 digit prime numbers is easy.
  - Calculating $g^x$ MOD n is also easy



# Another Example

- Calculate $y = 10^{97} \text{ MOD } 89$

$10 = 10 \text{ MOD } 89$

$10^2 = 11 \text{ MOD } 89$

$10^4 = 32 \text{ MOD } 89$

$10^8 = 32 \text{ MOD } 89$

$10^{16} = 45 \text{ MOD } 89$

$10^{32} = 39 \text{ MOD } 89$

$10^{64} = 8 \text{ MOD } 89$

Therefore

$10^{97} \text{ MOD } 89$

$= 10^{64} . 10^{32}. 10 \text{ MOD } 89$

$= 8 . 39 . 10 . \text{MOD } 89$

$= 3120 \text{ MOD } 89$

$= 5$

- Now try to reverse this !.....DIFFICULT

# Notes

- The number g should be chosen such that it is a **small primitive root** of n, that is .....
  - $x^i$ MOD n
- should generate the numbers 1...(n-1) in some order.



# Primitive Roots

- 2 is NOT a primitive root of 7 while 3 is...

$2^1 = 2$ MOD 7
$2^2 = 4$ MOD 7
$2^3 = 1$ MOD 7
$2^4 = 2$ MOD 7
$2^5 = 4$ MOD 7
$2^6 = 1$ MOD 7

$3^1 = 3$ MOD 7
$3^2 = 2$ MOD 7
$3^3 = 6$ MOD 7
$3^4 = 4$ MOD 7
$3^5 = 5$ MOD 7
$3^6 = 1$ MOD 7

- Theorem
- n is a primitive root of a prime p
  - IFF $n^{(p-1)/q}$ is not equivalent to 1 MOD P
- for any prime divisor q of (p-1)
- EXAMPLE
  - p = 7 and so, (p-1) = 6 = 3*2
  - $2^{6/2} = 2^3 = 1$ MOD 7
- Therefore 2 is **NOT** a primitive root of 7
  - $3^{6/2} \neq 1$ MOD 7
  - $3^{6/3} \neq 1$ MOD 7
- Therefore 3 **IS** a primitive root of 7



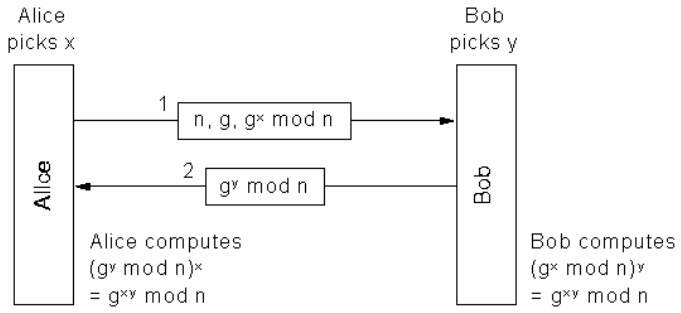



Fig. 7-15. The Diffie-Hellman key exchange.

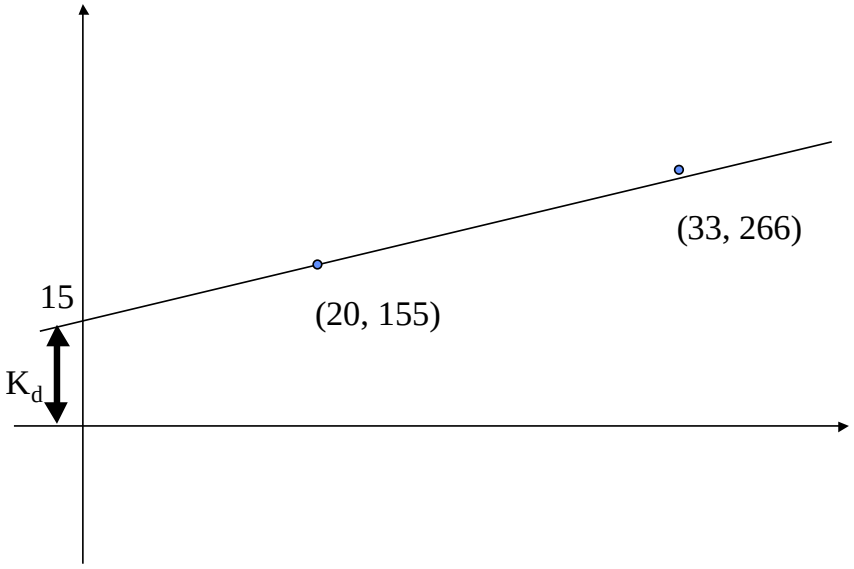
# Sharing Keys

- It is possible to share a key among a group of people, such that all must be present in order to reassemble the key for use
- If the secret key $K_d$ is to be used by *n* people, then we randomly pick a polynomial of degree *n-1*
  - $y = a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + a_{n-3} x^{n-3} + a_1 x + Kd$



# Example

- Assume the secret key $K_d = 15$ is to be shared by two people (simple example), a randomly generated polynomial is generated.….
  - $y = 7x + \mathbf{15}$
- The first person is given the coordinates (20, 155), the second person is given (33, 266)






# DES

- Adopted by US Gov. in 1977
- No longer secure in original form, modified form still useful however
- Uses 19 stages product cipher, 56 bit key
- Uses S-box and P-box mixing

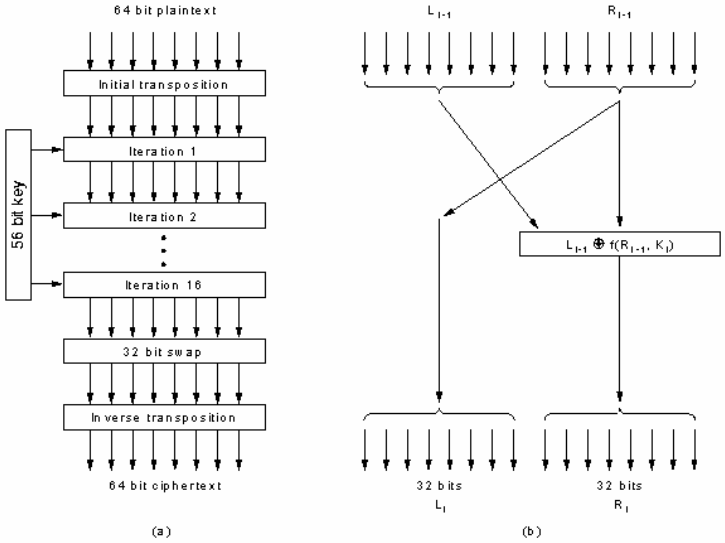



Fig. 7-5. The data encryption standard. (a) General outline. (b) Detail of one iteration.

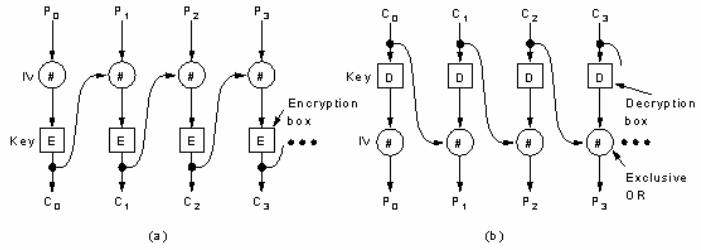



Fig. 7-7. Cipher block chaining

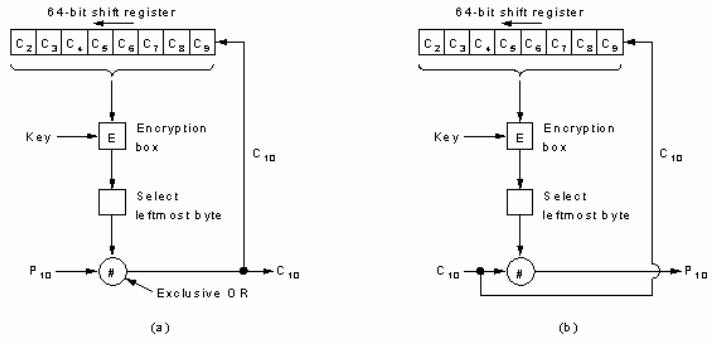



Fig. 7-8. Cipher feedback mode.

# IDEA

- Developed by Lai and Massey in 1990, 1992
- 128 bit key, immune to brute force attacks for decades to come
- Resembles DES, 64 bit block plaintext inputs

# Strength of IDEA

- **Block length** of 64 bits deters statistical analysis of Ciphertext. Complexity of implementation grows exponentially with lengthening of block size, but 64 bit blocks provides sufficient complexity to overcome statistical analysis. Algorithm may be further strengthened by using a cipher feedback mode.
- **Key length** of 128 bits provides security from exhaustive key searches.



- **Confusion** introduced by complicated and involved dependency between the plaintext and the Ciphertext. IDEA uses three different mathematical operations (bitwise XOR, modulo $2^{16}$ addition and modulo $2^{16}+1$ multiplication) in contrast to others such as DES which use only one mathematical operation (bitwise XOR) and the nonlinear S-box.
- **Diffusion** is introduced by having each plaintext exerting an influence over each Ciphertext bit. IDEA implements this very well by spreading each plaintext bit over a large number of Ciphertext bits, thus hiding the statistical structure of the plaintext.

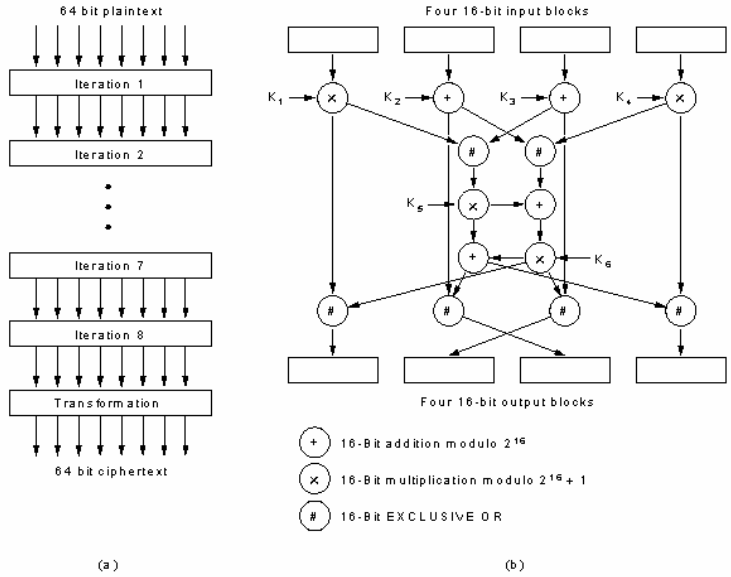



Fig. 7-10. (a) IDEA. (b) Detail of one iteration.

# **P**ublic **K**ey **C**ryptography (PKC)

- Problem: cryptographic key must be kept secret
- In PKC the decryption key is kept secret, but the encryption key may be published.
- Deducing the Decryption key from the encryption key is effectively impossible.

# PKC Gives Strong Encryption

- PKC is based on **trap door functions** (one way functions), easy to solve in one direction, but extremely difficult in the other.
  - E.g. $n = Ft(p, q)$.
- The trap-door function here is the multiplication of two prime numbers, p and q, which is easy to solve.
  - $(p, q) = Ft^{-1}(n)$
- The inverse function, to find the factors of n is a very difficult problem to solve.



# Example

- What are the factors of 11023
- What is 72 * 151

# RSA

- Rivest, Shamir Adleman
- 1978 paper



# RSA Method

- Choose two large primes *p* and *q*
  - typically $>10^{100}$
- Compute
  - n = p * q
  - z = (p-1)(q-1)
- Choose a number relatively prime to *z*
  - call it *d*
- Find *e* such that *e * d = 1 MOD Z*

# RSA Algorithm

- Pre-compute the parameters
- Divide plaintext into blocks (bit-strings) P s.t. $0 <= P < n$
- To encrypt
  - $C = P^e \text{ MOD } n$
- To decrypt
  - $P = C^d \text{ MOD } n$



# RSA Example

- The encryption process may be summarised thus.....
  - Plaintext: $M < n$
  - Ciphertext: $C = M^e \text{ (MOD } n)$
- The decryption process.....
  - Ciphertext: C
  - Plaintext: $M = C^d \text{ (MOD } n)$

- Decryption requires *d* and *n*
- Encryption requires *e* and *n*
- Public key consists of (*e*, *n*)
- Private key consists of (*d*, *n*)
- Given *z* and *e*, *d* can be found using Euclid's algorithm
- Strength of method relies upon difficulty of factoring large numbers
- No mathematical breakthroughs reported on this factoring problem in 300 years!.



# Simple Numeric Example

- Select two primes, $p = 7$ and $q = 17$
- Calculate
  - $n = pq = 7 * 17 = 119$
- Calculate the product
  - $Z = (p-1) * (q-1) = 96$
- Select *e* s.t. *e* is **relatively prime** to 96 and less than 96, e.g. 5
- Determine *d*, s.t. $de = 1 \bmod 96$ and $d<96$, i.e. $d = 77$
  - $77 * 5 = 385 = 4 * 96 + 1$

- For a plaintext input of M = 19, 19 is raised to the 5th power, yielding 2 476 099. When divided by 119 (n), the remainder is calculated to be 66. So.....
  - $19^5 \equiv 66$ MOD 119
- Yields a ciphertext of **66**.
- For decryption it is determined that
  - $66^{77} \equiv 19$ MOD 119
- The encryptor may now proceed to generate Ciphertext using another persons public key *e*, which only the owner of *e* (or their trusted colleagues) may decrypt using *d*.



| Plaintext (P) | | | Ciphertext (C) | | After decryption | |
|---|---|---|---|---|---|---|
| Symbolic | Numeric | $P^3$ | $P^3 \pmod{33}$ | $C^7$ | $C^7 \pmod{33}$ | Symbolic |
| S | 19 | 6859 | 28 | 13492928512 | 19 | S |
| U | 21 | 9261 | 21 | 1801088541 | 21 | U |
| Z | 26 | 17576 | 20 | 1280000000 | 26 | Z |
| A | 01 | 1 | 1 | 1 | 1 | A |
| N | 14 | 2744 | 5 | 78125 | 14 | N |
| N | 14 | 2744 | 5 | 78125 | 14 | N |
| E | 05 | 125 | 26 | 8031810176 | 5 | E |
| Sender's computation | | | | Receiver's computation | | |

Fig. 7-11. An example of the RSA algorithm.

# Authentication

- Authentication may be achieved by a number of different methods.
- PKC provides one such scheme, whereby a message may be signed using the senders secret key, allowing all and sundry to decode the message with the public key
- Only the real Alice has Alice's secret key



# Authentication Using PKC

- Alice encrypts her identity $A$ and a random number $R_A$ using Bob's encryption key $E_B$
- When Bob receives this message, he does not know whether it came from Alice or was inserted by Trudi
- Bob sends back a message containing Alice's $R_A$, his own $R_B$ and a session key $K_S$

- When Alice gets Bob's response, she decrypts it using her secret key $D_A$
- She finds $R_A$ inside, thus she knows that Bob was able to decipher her message using his secret key $D_B$, which only he has and Trudi has not
- Alice agrees to the session by sending back $R_B$ encrypted with the session key $K_S$
- When Bob sees $R_B$ he knows that only Alice could have decrypted it and re-sent it
- What are the weaknesses of this method?



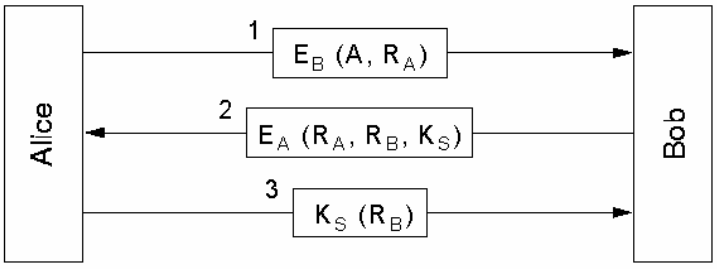



Fig. 7-21. Mutual authentication using public-key cryptography.

# Digital Signatures

- Signed messages are required for electronic documents s.t.
  - Receiver can verify claimed identity of sender
  - Sender cannot later repudiate message
  - Receiver cannot forge the message themselves



# Digital Signatures with PKC

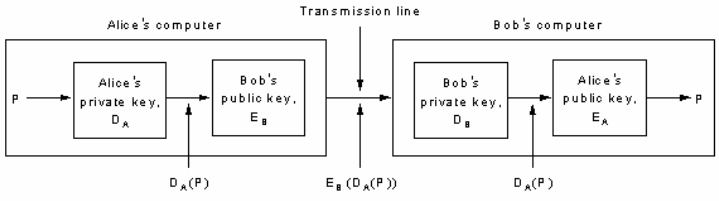



Fig. 7-23. Digital signatures using public-key cryptography.

# MD5

- A message may be authenticated without having to encrypt the entire message, as is done by using PKC
- A one way hash function takes an arbitrarily long piece of text and attaches redundant information at the end which is dependant upon the content of the message. If the message is changed in any way, then the redundant information changes.



# MD5 Hash Function

- MD5 is one message digest function proposed by Ron Rivest (the R in RSA).
- This **Hash Function** has three important properties …
  - Given *P* it is easy to compute *MD(P)*
  - Given *MD(P)* it is effectively impossible to find *P*
  - No one can generate two messages that have the same message digest.
- The hash should be greater than 128 bits to satisfy property 3

- In PKC systems, Alice computes message digest for her plaintext, MD(P). She then signs the message digest (for authentication) and sends the signed digest and the plaintext to Bob.
- If Trudi substitutes or interferes with P, then Bob will know when he computes MD(P) himself.
- Trudi cannot regenerate MD(P) for the tampered message because it was signed with Alice's secret key.



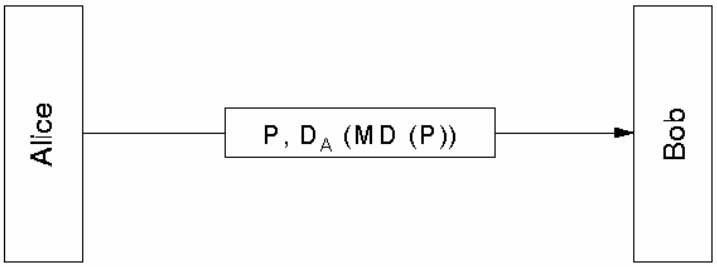



Fig. 7-24. Digital signatures using message digests.

# MD5 Algorithm

- Pad message to 64 bits short of a multiple of 512 bits long
- Append original length as 64 bit integer (total length now multiple of 512 bits)
- Initialise a 128 bit buffer to a fixed value
- Take 512 bits if input and mix with 128 bit buffer
- Throw in a table constructed from the Sine function, for good measure. This is included to help assure users that no trap door functions exist
- Four rounds are performed for each block.
- GOTO 4 until all input consumed
- The contents of the 128 bit buffer form the message digest.



# PGP

- PGP is based upon the **IDEA** encryption algorithm and **RSA** public key generation and **MD5**
- Developed by **Phil Zimmerman**, famous world-wide for failed attempts of US government to prosecute him for alleged export of strong cryptography (regarded in US as exportation of munitions)

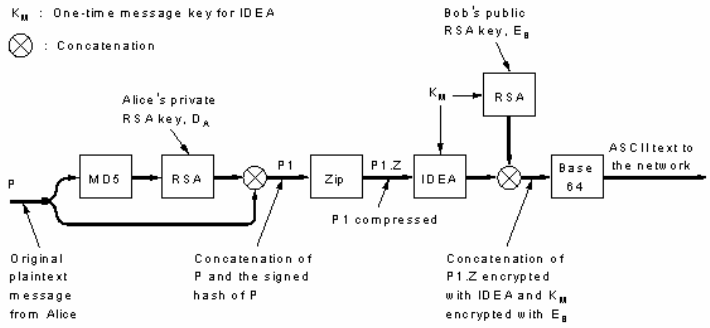



Fig. 7-49. PGP in operation for sending a message.