ELECTRONIC MAIL SECURITY

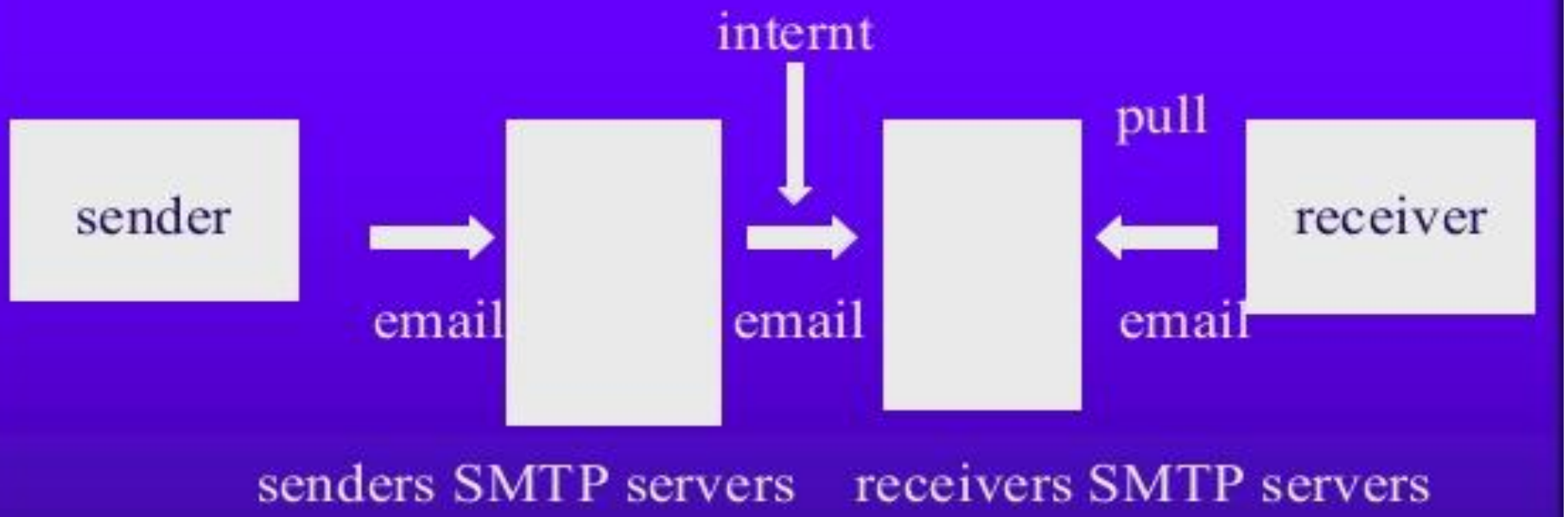
Introduction

- email is one of the most widely used and regarded network services
- currently message contents are not secure
  - may be inspected either in transit
  - or by suitably privileged users on destination system

Email security requirements

- Confidentiality
- Authentication
- Integrity
- Non-repudiation

Basic phase of Email



Privacy enhanced mail

PEM adopted by the **Internet Architecture Board (IAB)** to provide secure electronic mail communication over the internet

Steps of PEM

- Canonical conversion
- Digital signature
- Encryption
- Base 64 encoding

Canonical conversions

- PEM transforms each email message into an abstract canonical representation. This means that regardless of the architecture and the operating system of the sending and receiving computers, the email message always travels in a uniform, independent format.

Digital signature

Email message
To:abc@xyz.com
From:mnc@xyz.com
Subject:meeting

→ Message digest Algorithm(MD2 Or MD5) →

1010101010101
01010……..
……

Message digest creation

1010101010101
01010……..
……

→ encrypt →

digital signature

↑

Sender's private key

encryption

Email message
To:abc@xyz.com
From:mnc@xyz.com
Subject:meeting

Symmetric key

encrypt

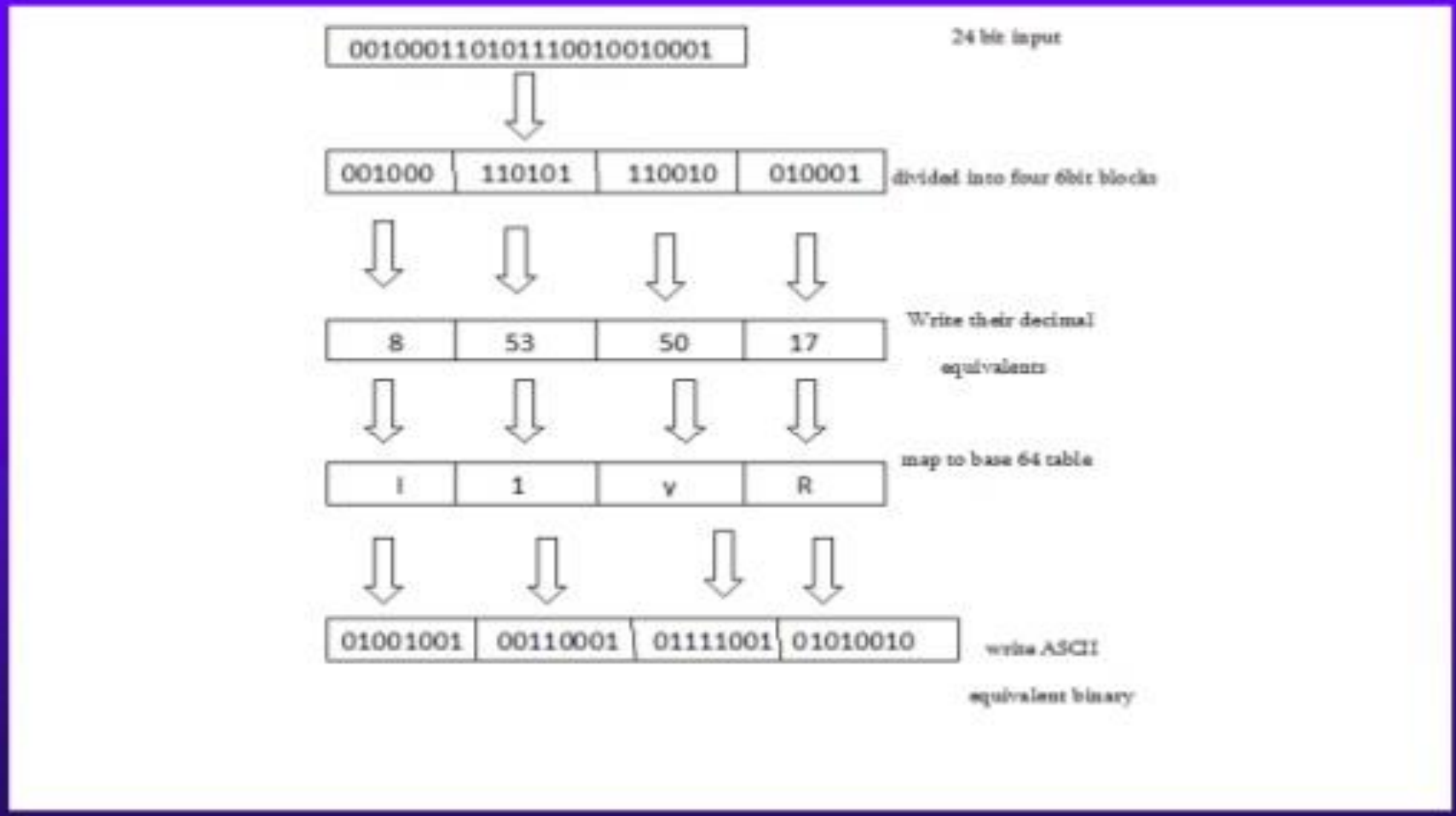
Encrypted result

+

digital signature

DES in CBC mode

Base 64encoding



Printable encoding characters

0 A	8 I	16 Q	24 Y	32 g	40 o	48 w	56 4
1 B	9 J	17 R	25 Z	33 h	41 p	49 x	57 5
2 C	10 K	18 S	26 a	34 i	42 q	50 y	58 6
3 D	11 L	19 T	27 b	35 j	43 r	51 z	59 7
4 E	12 M	20 U	28 c	36 k	44 s	52 0	60 8
5 F	13 N	21 V	29 d	37 l	45 t	53 1	61 9
6 G	14 O	22 W	30 e	38 m	46 u	54 2	62 +
7 H	15 P	23 X	31 f	39 n	47 v	55 3	63 /

Pretty Good Privacy (PGP)

PGP provides a confidentiality and authentication service that can be used for electronic mail and file storage applications.

A number of reasons can be cited for this growth.

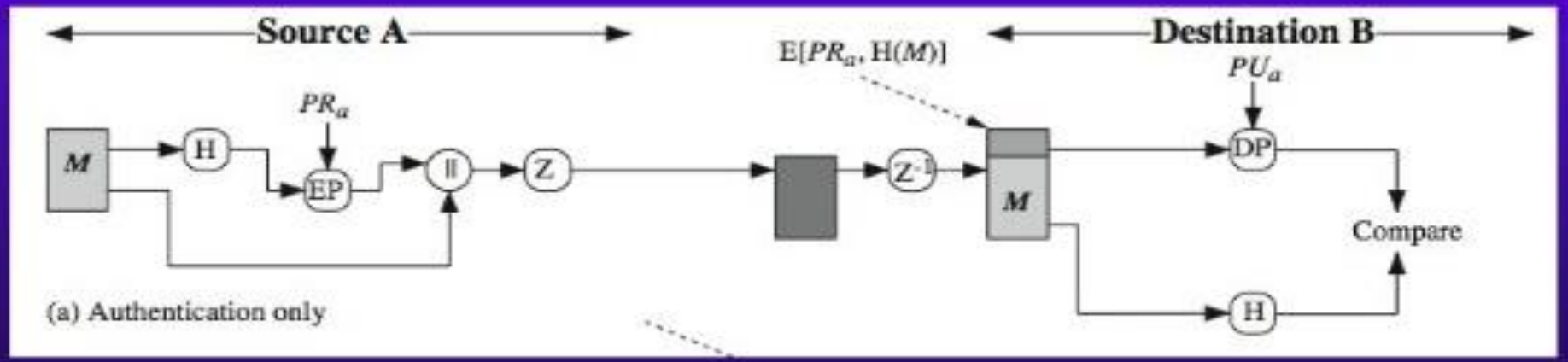
- available free worldwide
- It is based on extremely secure algorithm.
- wide range of applicability
- not developed by governmental organization.

Operational description

The actual operation of PGP, consists of five services: authentication, confidentiality, compression, e-mail compatibility, and segmentation

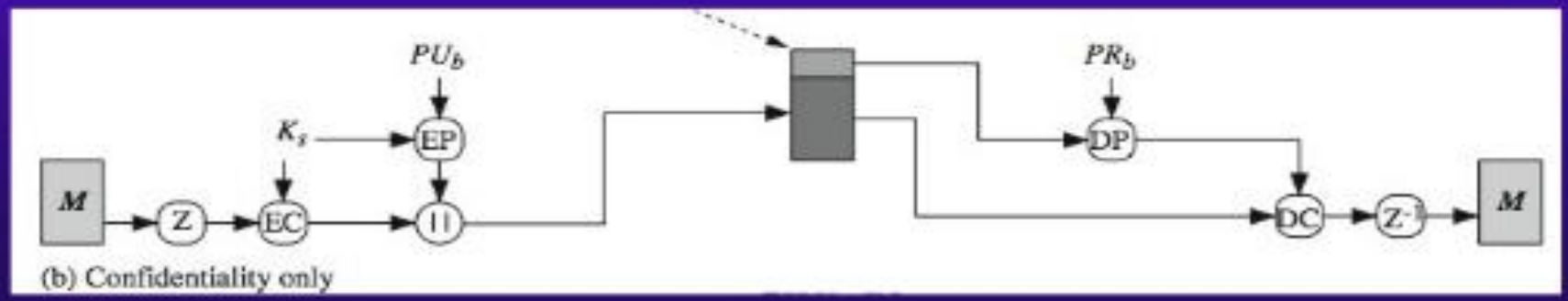
Authentication

1. sender creates message
2. make SHA-1160-bit hash of message
3. attached RSA signed hash to message
4. receiver decrypts & recovers hash code
5. receiver verifies received message hash



Confidentiality

1. sender forms 128-bit random session key
2. encrypts message with session key
3. attaches session key encrypted with RSA
4. receiver decrypts & recovers session key
5. session key is used to decrypt message



(b) Confidentiality only

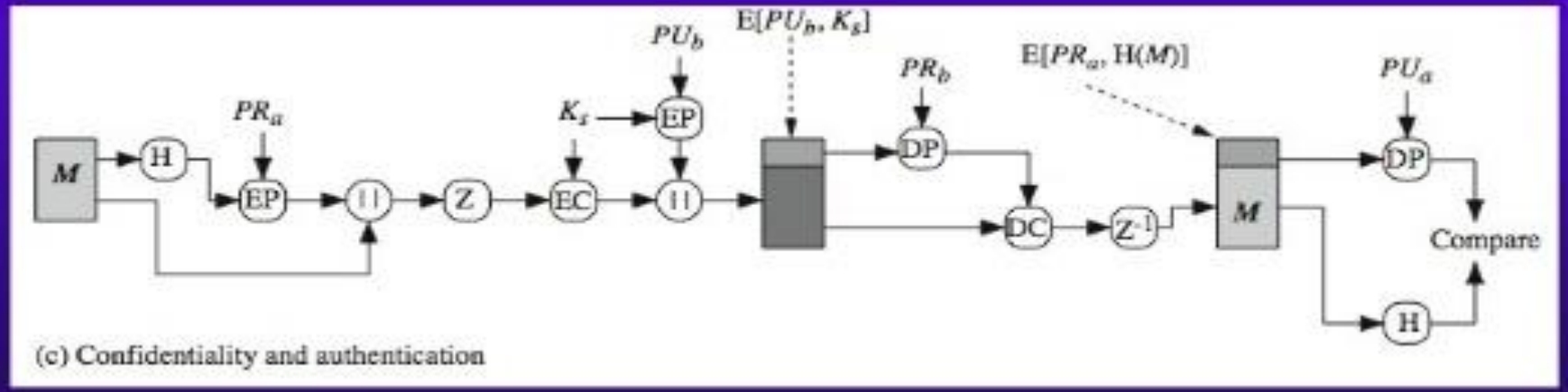
Confidentiality & Authentication

can use both services on same message

create signature & attach to message

encrypt both message & signature

attach RSA/ElGamal encrypted session key



(c) Confidentiality and authentication

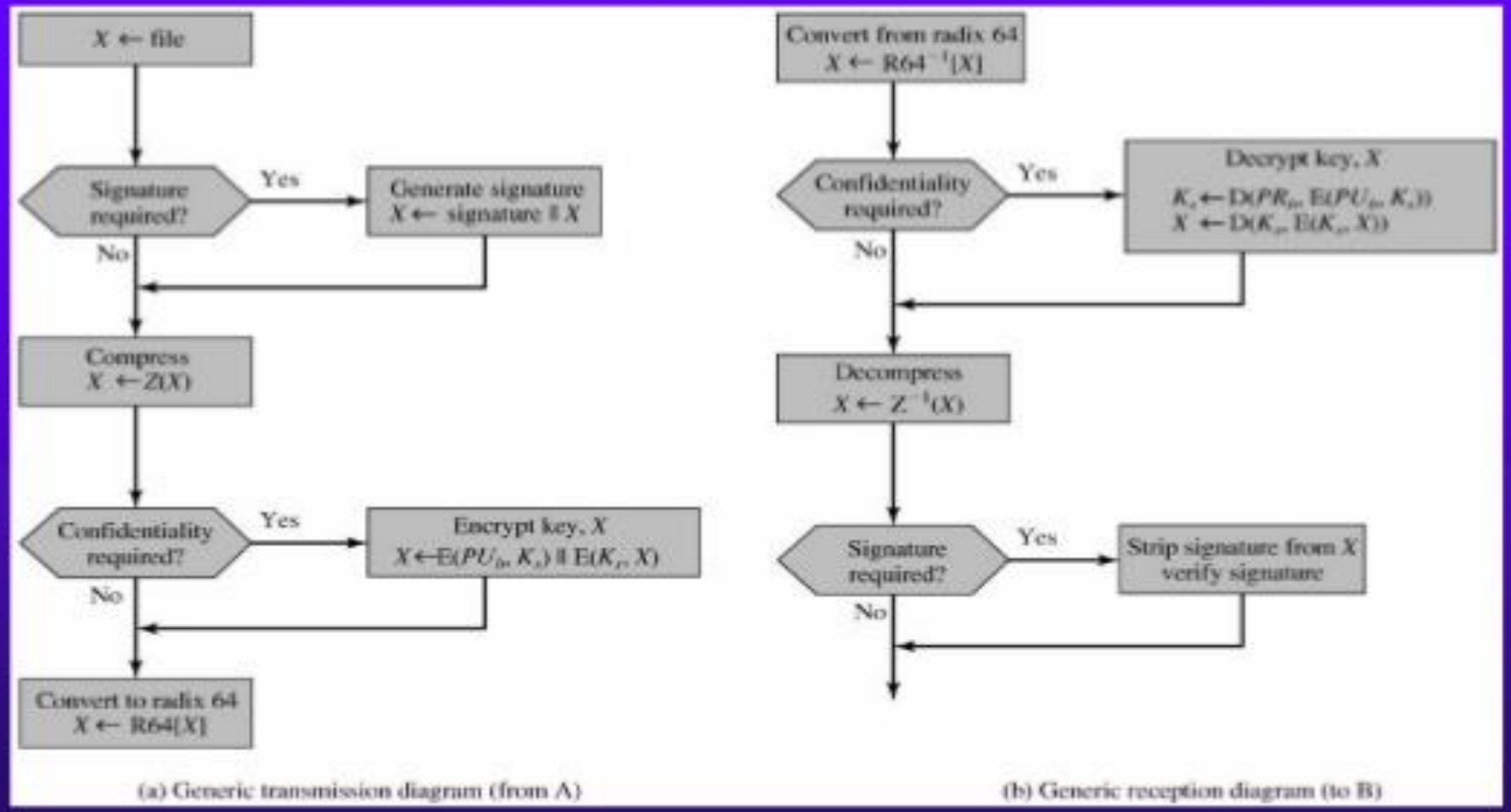
Compression

- by default PGP compresses message after signing but before encrypting
  - so can store uncompressed message & signature for later verification
  - & because compression is non deterministic
- uses ZIP compression algorithm

Email Compatibility

- when using PGP will have binary data to send (encrypted message etc)
- however email was designed only for text
- hence PGP must encode raw binary data into printable ASCII characters
- uses radix-64 algorithm
  - maps 3 bytes to 4 printable chars
  - also appends a CRC
- PGP also segments messages if too big

PGP Operation – Summary



(a) Generic transmission diagram (from A)

(b) Generic reception diagram (to B)

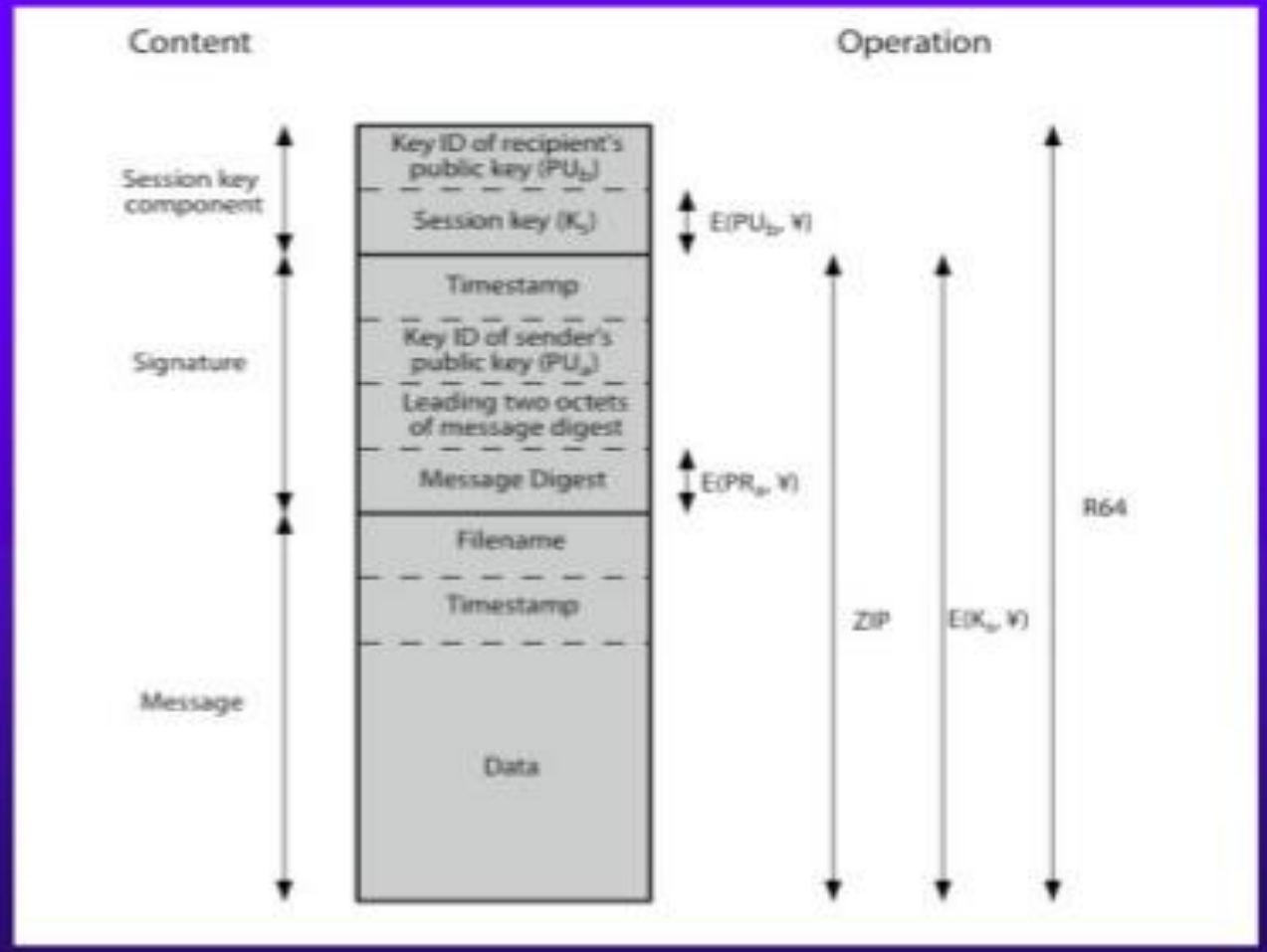
PGP Session Keys

- need a session key for each message
  - of varying sizes: 56-bit DES, 128-bit CAST or IDEA, 168-bit Triple-DES
- generated using ANSI X12.17 mode
- uses random inputs taken from previous uses and from keystroke timing of user

PGP Public & Private Keys

- since many public/private keys may be in use, need to identify which is actually used to encrypt session key in a message
  - could send full public-key with every message
  - but this is inefficient
- rather use a key identifier based on key
  - is least significant 64-bits of the key
  - will very likely be unique
- also use key ID in signatures

PGP Message Format



PGP Key Rings

Private Key Ring

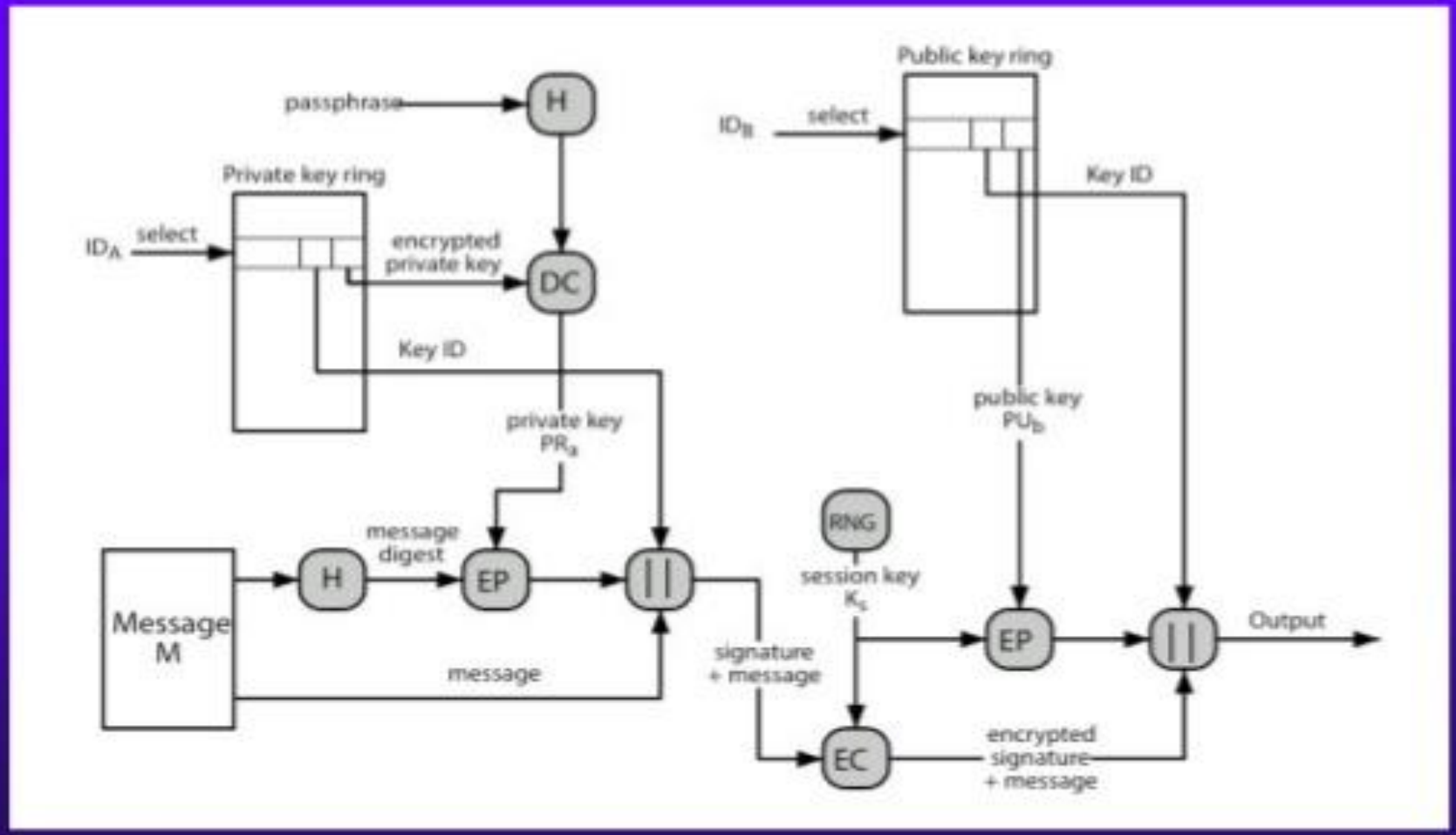
Timestamp	Key ID*	Public Key	Encrypted Private Key	User ID*
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•
Ti	$PU_i \bmod 2^{64}$	PU_i	$E(H(P_i), PR_i)$	User i
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•

Public Key Ring

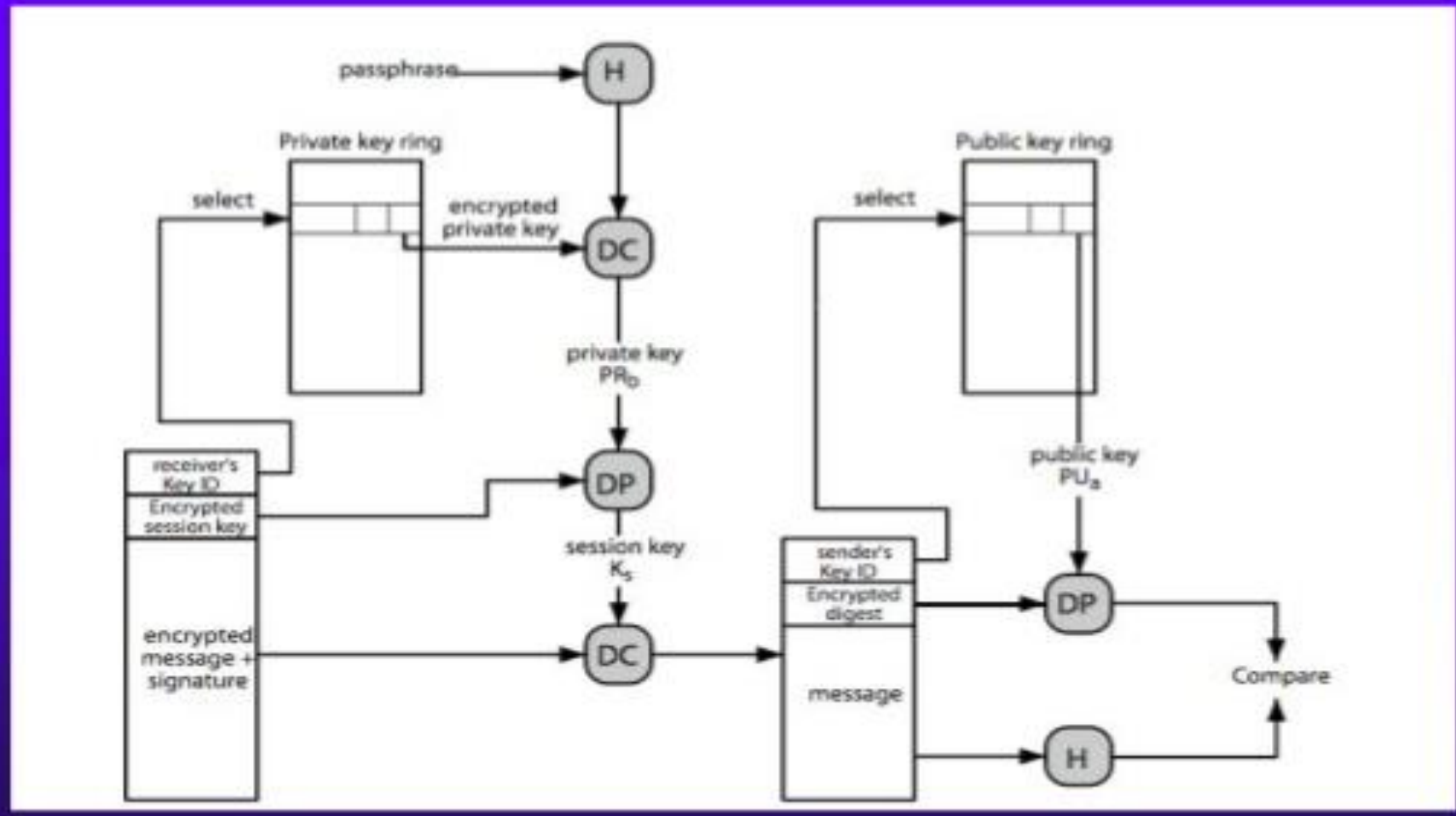
Timestamp	Key ID*	Public Key	Owner Trust	User ID*	Key Legitimacy	Signature(s)	Signature Trust(s)
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
T_i	$PU_i \bmod 2^{64}$	PU_i	$trust\_flag_i$	User i	$trust\_flag_i$		
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

* = field used to index table

PGP Message Generation



PGP Message Reception



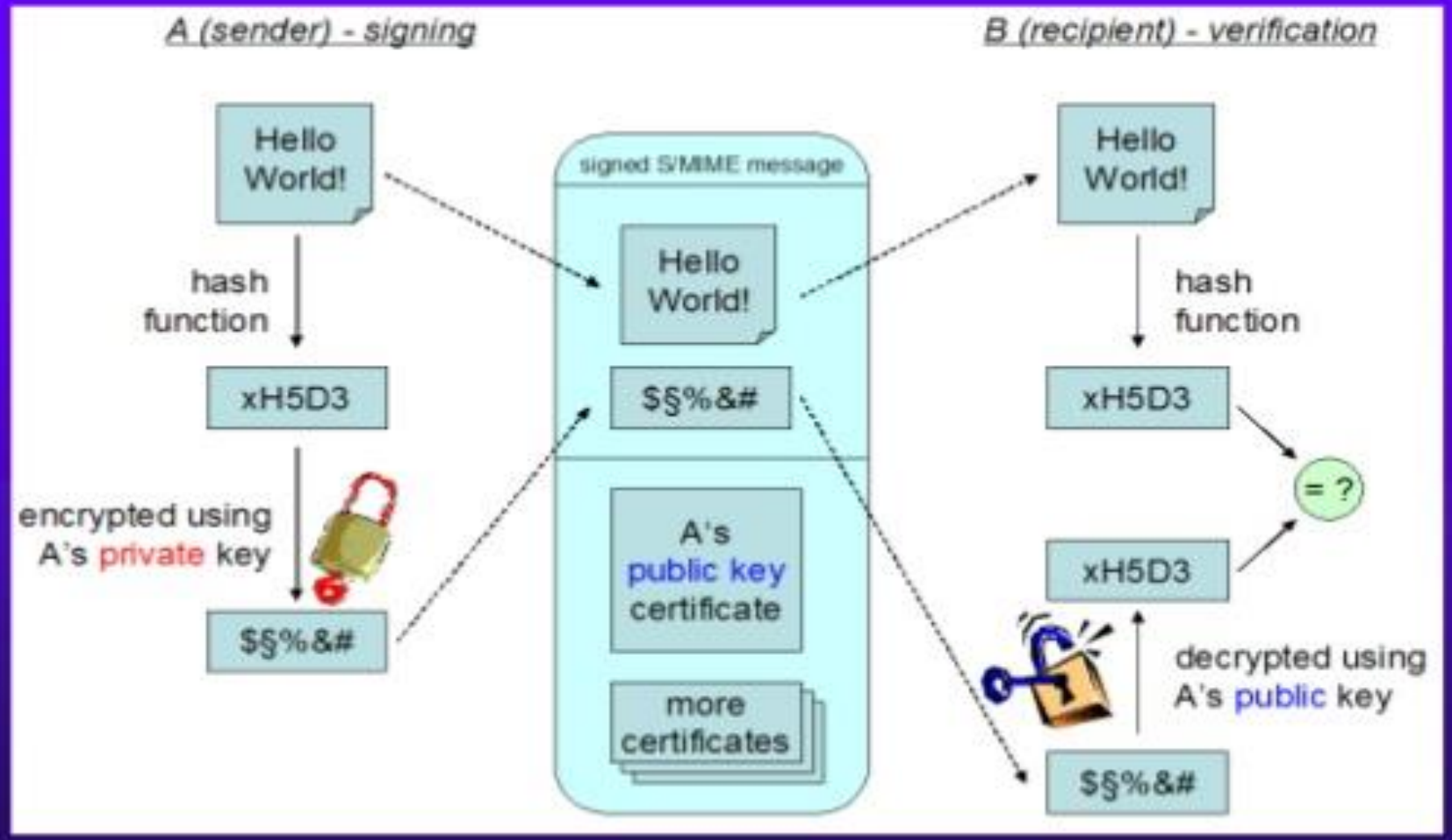
S/MIME (Secure/Multipurpose Internet Mail Extensions)

- security enhancement to MIME email
  - original Internet RFC822 email was text only
  - MIME provided support for varying content types and multi-part messages
  - with encoding of binary data to textual form
  - S/MIME added security enhancements
- have S/MIME support in many mail agents
  - eg MS Outlook, Mozilla, Mac Mail etc

Signed mail

1. The user writes the message as clear-text.
2. The message digest is being calculated (using SHA-1[2] or MD5[3]).
3. The message digest is being encrypted using the signer's private key (DSS[4] or RSA[5]).

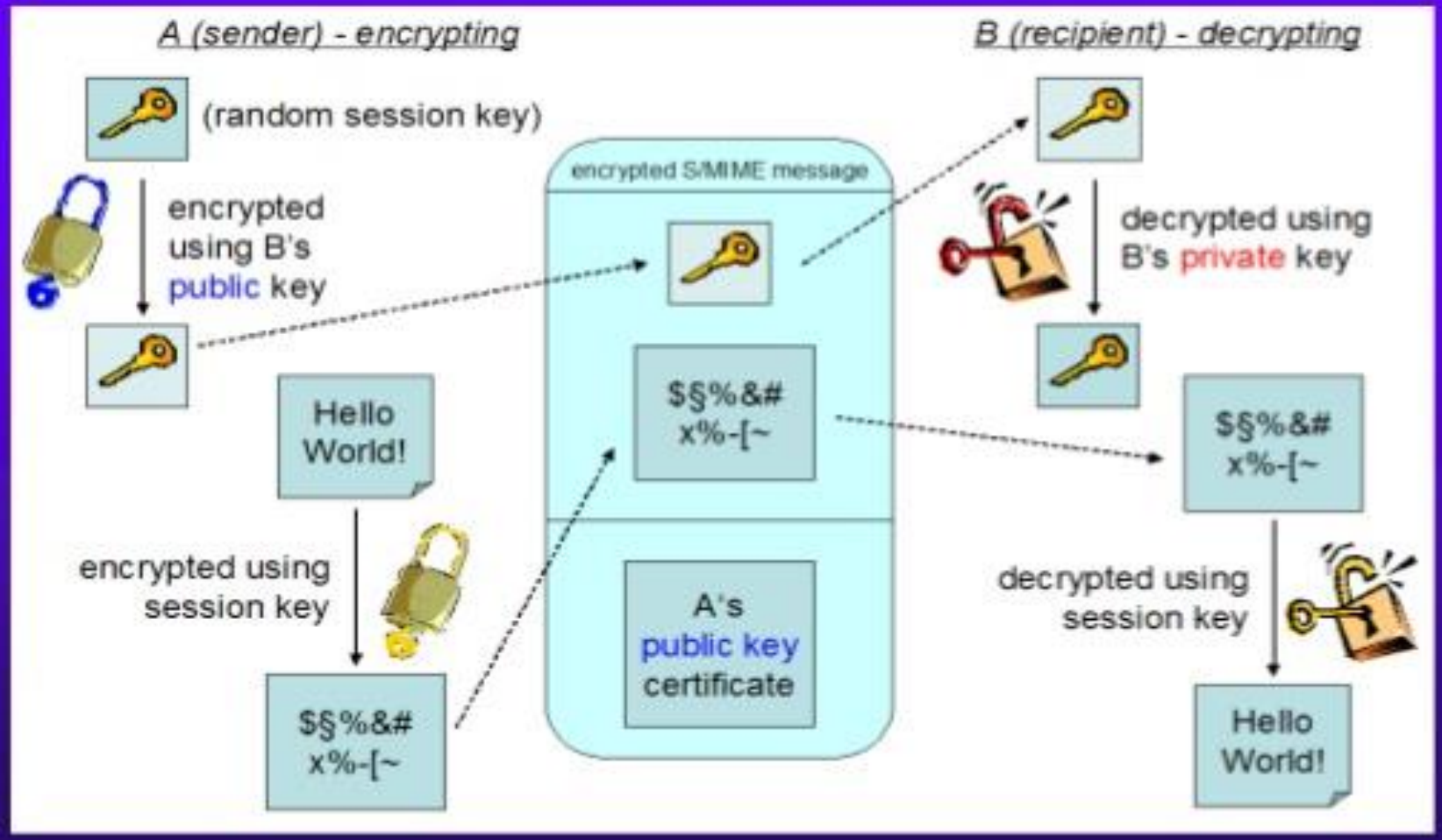
Signed mail


A (sender) - signing
B (recipient) - verification
Hello World!
hash function
xH5D3
encrypted using A's private key
$§%&#
signed S/MIME message
Hello World!
$§%&#
A's public key certificate
more certificates
Hello World!
hash function
xH5D3
= ?
xH5D3
decrypted using A's public key
$§%&#

Encrypted mail

1. The user writes the message as clear-text.
2. A random session key is being created (tripleDES[6] or RC2[7])
3. The message is being encrypted using the random session key.
4. For every recipient, the session key is being encrypted using the recipient's public key (DH[8] or RSA[5]).

Encrypted mail



S/MIME Cryptographic Algorithms

- digital signatures: DSS & RSA
- hash functions: SHA-1 & MD5
- session key encryption: ElGamal & RSA
- message encryption: AES, Triple-DES, RC2/40 and others
- MAC: HMAC with SHA-1

S/MIME Functions

- enveloped data
  - encrypted content and associated keys
- signed data
  - encoded message + signed digest
- clear-signed data
  - cleartext message + encoded signed digest
- signed & enveloped data
  - nesting of signed & encrypted entities