CS-552

CYBER FORENSICS

E-mail Security
PGP, S/MIME
Certificates and PKI

E-mail Security

E-mail is one of the most widely used network services

- One of the old killer applications of the Internet

Normally message contents not secured

- Can be read/modified either in transit or at destination by the attacker

E-mail service is like postcard service

- just pick it and read it

Email Security Enhancements

confidentiality
- protection from disclosure

authentication
- of sender of message

message integrity
- protection from modification

non-repudiation of origin
- protection from denial by sender

Quick E-mail History

SMTP and RFC 822 (later RFC 5322)

- ~~SMTP is the email transfer protocol running over TCP~~
- RFC 822/5322 defines the message format and headers
  - only ASCII messages (7-bit)

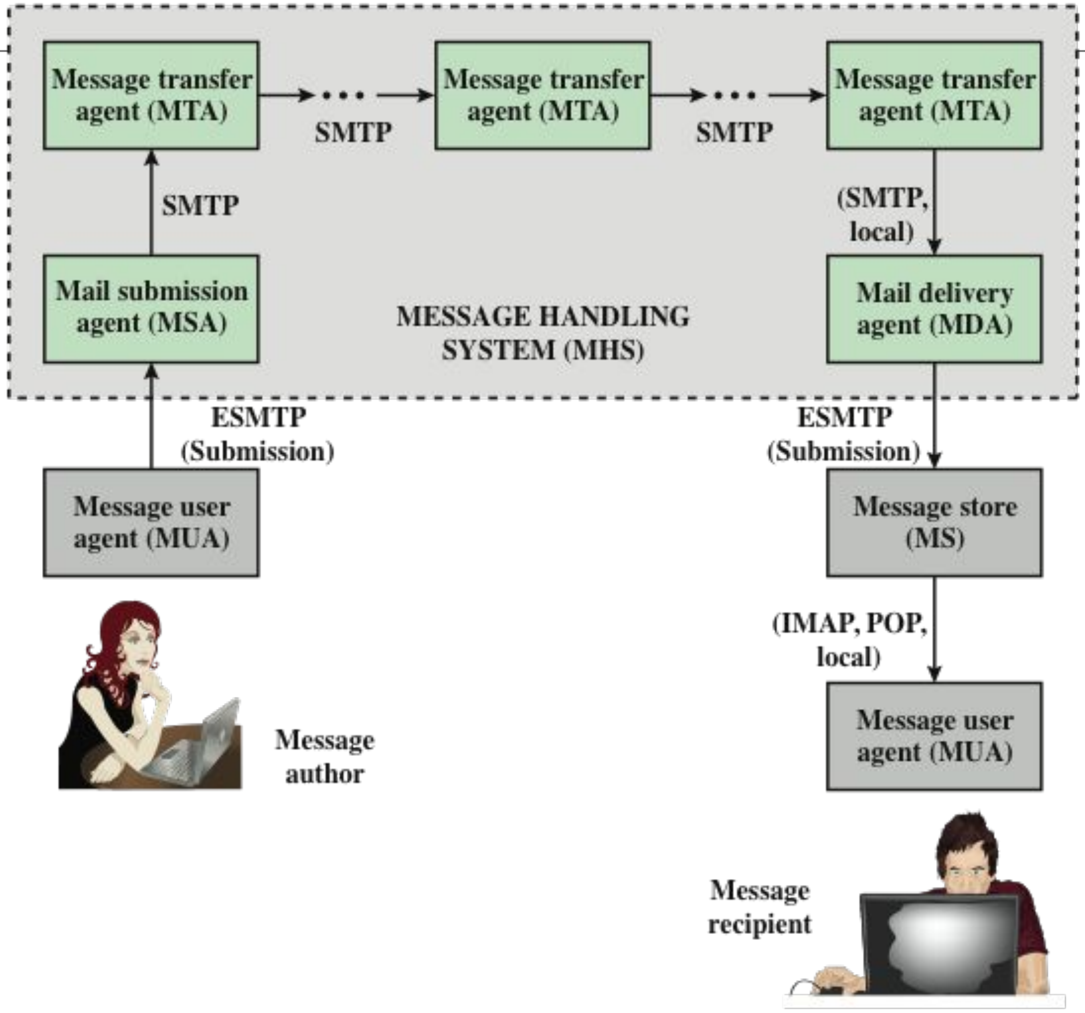
MIME (Multipurpose Internet Mail Extensions)

- content type
  - Almost any type of information can appear in an email message
- transfer encoding
  - specifies how the message body is encoded into textual form (radix64 is common)

S/MIME: Secure MIME

- new content types, like signature, encrypted data

More on Internet Email Architecture



More on MIME

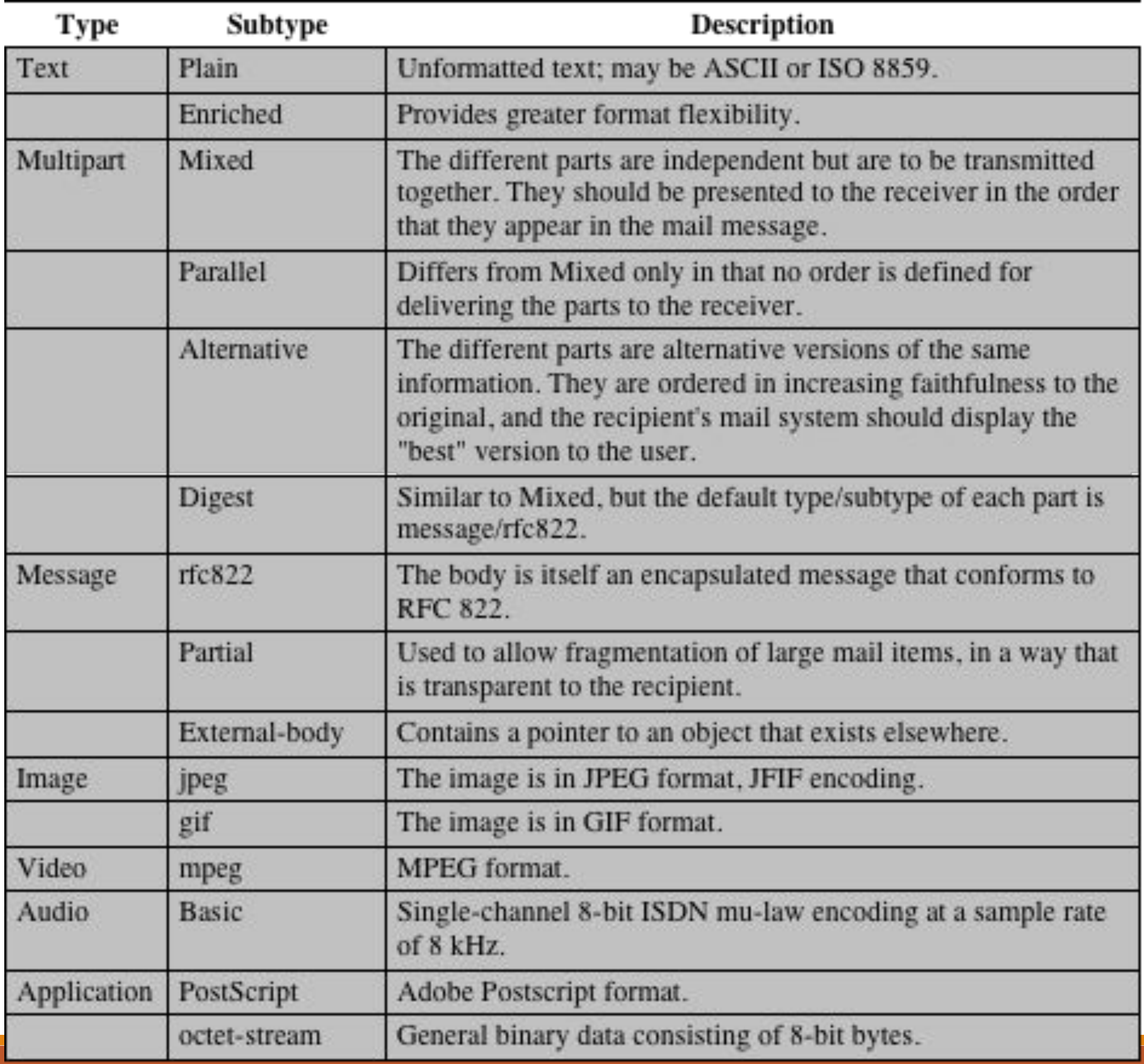
Multipurpose Internet Mail Extensions

- Addresses the limitations of SMTP/5322 scheme
  - Most important one: binary data transfer
  - Attachments

Adds some fields to the email messages: Important ones:

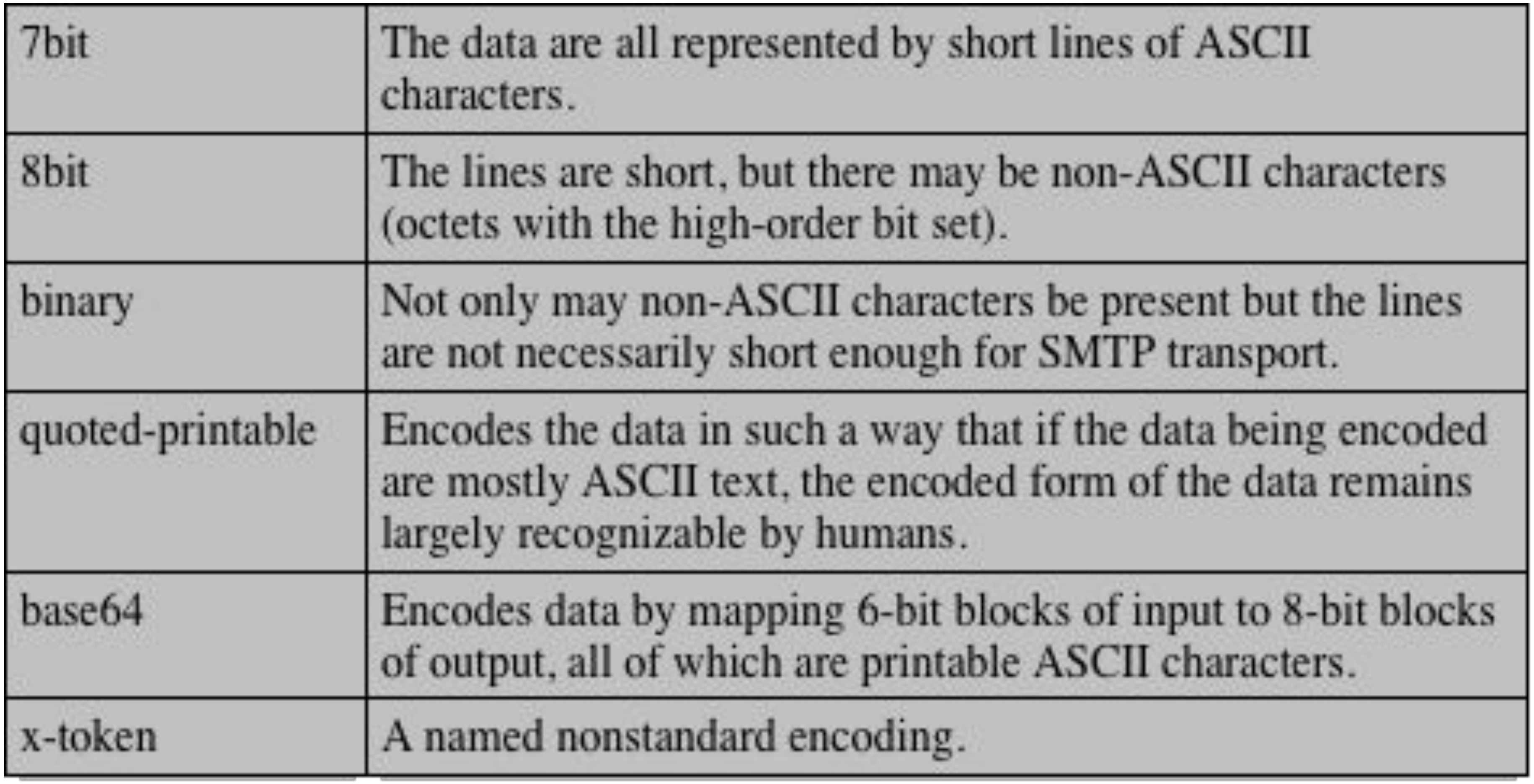
- Content-Type
  - Describes the data contained in the body with sufficient detail that the receiving user agent can pick an appropriate agent or mechanism to represent the data to the user
- Content-Transfer-Encoding
  - Indicates the type of transformation that has been used to represent the body of the message in a way that is acceptable for mail transport

MIME Content Types



Type	Subtype	Description
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
	Enriched	Provides greater format flexibility.
Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order that they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined for delivering the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original, and the recipient's mail system should display the "best" version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822.
Message	rfc822	The body is itself an encapsulated message that conforms to RFC 822.
	Partial	Used to allow fragmentation of large mail items, in a way that is transparent to the recipient.
	External-body	Contains a pointer to an object that exists elsewhere.
Image	jpeg	The image is in JPEG format, JFIF encoding.
	gif	The image is in GIF format.
Video	mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of 8 kHz.
Application	PostScript	Adobe Postscript format.
	octet-stream	General binary data consisting of 8-bit bytes.

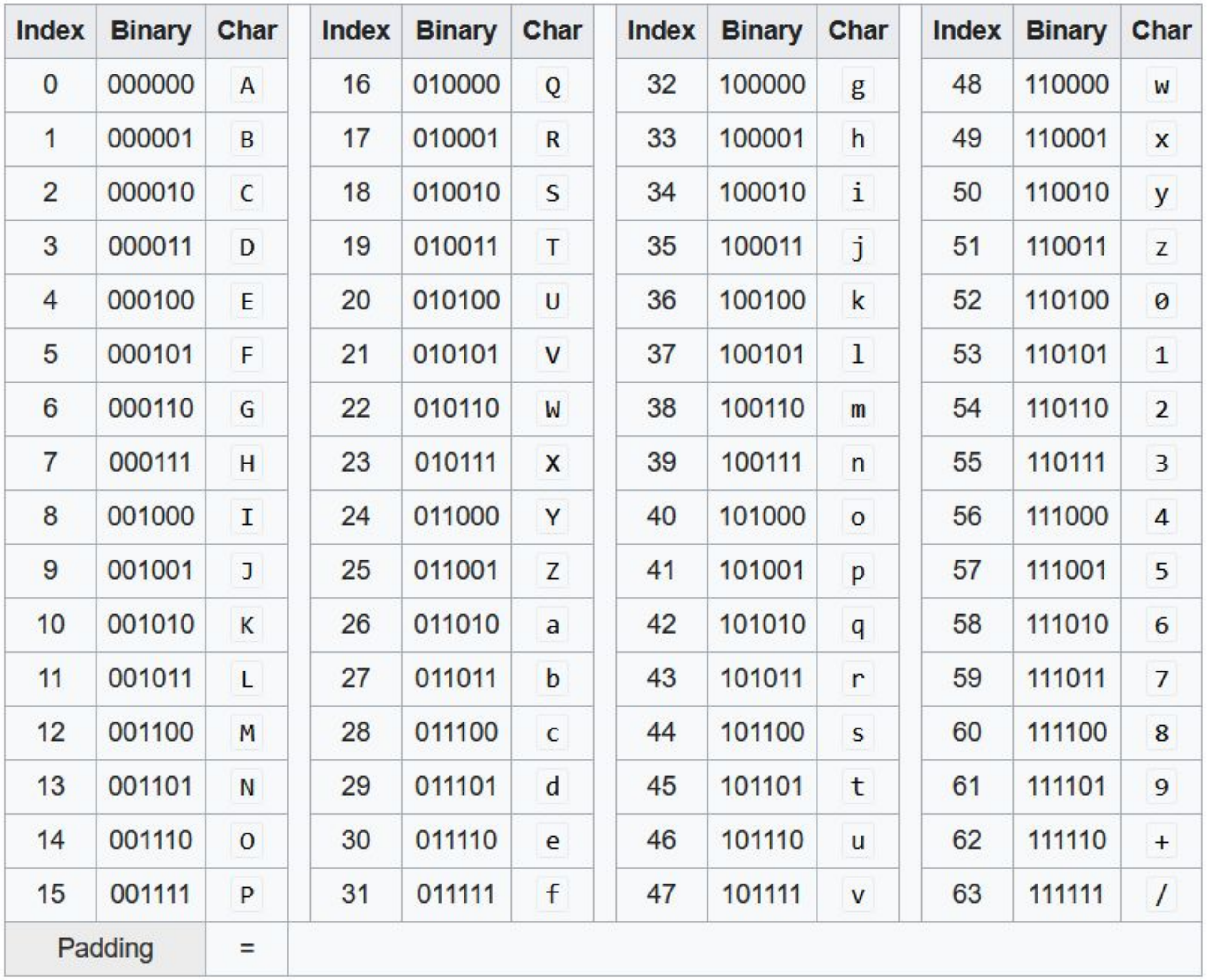
MIME Transfer Encodings



7bit	The data are all represented by short lines of ASCII characters.
8bit	The lines are short, but there may be non-ASCII characters (octets with the high-order bit set).
binary	Not only may non-ASCII characters be present but the lines are not necessarily short enough for SMTP transport.
quoted-printable	Encodes the data in such a way that if the data being encoded are mostly ASCII text, the encoded form of the data remains largely recognizable by humans.
base64	Encodes data by mapping 6-bit blocks of input to 8-bit blocks of output, all of which are printable ASCII characters.
x-token	A named nonstandard encoding.

(Table is on page 605 in textbook)

Base64 Encoding



Index	Binary	Char	Index	Binary	Char	Index	Binary	Char	Index	Binary	Char
0	000000	A	16	010000	Q	32	100000	g	48	110000	w
1	000001	B	17	010001	R	33	100001	h	49	110001	x
2	000010	C	18	010010	S	34	100010	i	50	110010	y
3	000011	D	19	010011	T	35	100011	j	51	110011	z
4	000100	E	20	010100	U	36	100100	k	52	110100	0
5	000101	F	21	010101	V	37	100101	l	53	110101	1
6	000110	G	22	010110	W	38	100110	m	54	110110	2
7	000111	H	23	010111	X	39	100111	n	55	110111	3
8	001000	I	24	011000	Y	40	101000	o	56	111000	4
9	001001	J	25	011001	Z	41	101001	p	57	111001	5
10	001010	K	26	011010	a	42	101010	q	58	111010	6
11	001011	L	27	011011	b	43	101011	r	59	111011	7
12	001100	M	28	011100	c	44	101100	s	60	111100	8
13	001101	N	29	011101	d	45	101101	t	61	111101	9
14	001110	O	30	011110	e	46	101110	u	62	111110	+
15	001111	P	31	011111	f	47	101111	v	63	111111	/
Padding		=									

S/MIME Functions

enveloped data
- encrypted content and associated keys

signed data
- encoded message + encoded signed message digest (hash)

clear-signed data
- cleartext message + encoded signed message digest (hash)

signed and enveloped data
- Nested signed and encrypted entities

S/MIME Cryptographic Algorithms

hash functions: switched to SHA256

digital signatures: Mostly RSA is used

session key encryption: Mostly RSA is used

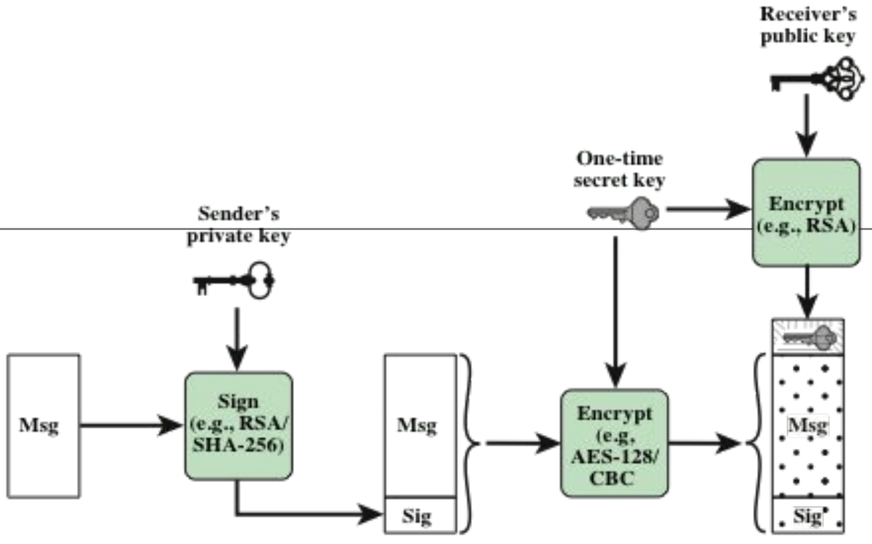
message encryption: Triple-DES, AES and others, but AES-128 is preferred

Base64 (radix64) encoding is used for email compatibility (ASCII conversion)

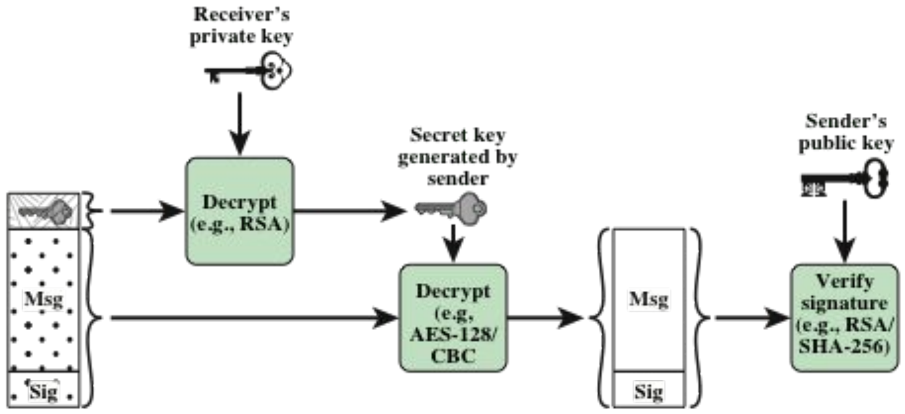
sender should know the capabilities of the receiving entity (public announcement or previously received messages from receiver)

- otherwise sender takes a risk of sending unintelligible e-mail.

S/MIME Security Functionality: Simplified View



(a) Sender signs, then encrypts message


Receiver's private key
Secret key generated by sender
Sender's public key
Decrypt (e.g., RSA)
Msg
Sig
Decrypt (e.g. AES-128/ CBC
Msg
Sig
Verify signature (e.g., RSA/ SHA-256)

(b) Receiver decrypts message, then verifies sender's signature

Scope of S/MIME Security

S/MIME secures a MIME entity

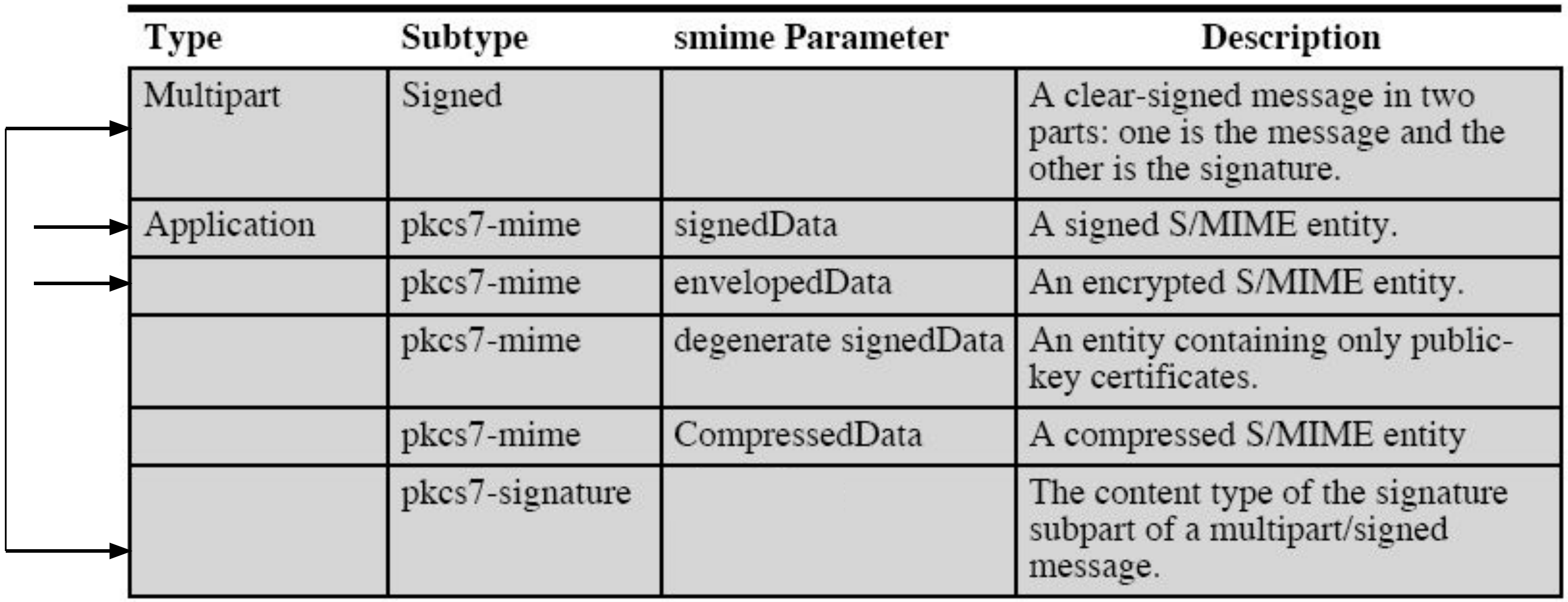
- a MIME entity is entire message except the headers
- so the header is not secured

First MIME message is prepared

This message and other security related data (algorithm identifiers, certificates, etc.) are processed by S/MIME

and packed as one of the S/MIME content type

S/MIME Content Types



Type	Subtype	smime Parameter	Description
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	pkcs7-mime	signedData	A signed S/MIME entity.
	pkcs7-mime	envelopedData	An encrypted S/MIME entity.
	pkcs7-mime	degenerate signedData	An entity containing only public-key certificates.
	pkcs7-mime	CompressedData	A compressed S/MIME entity
	pkcs7-signature		The content type of the signature subpart of a multipart/signed message.

EnvelopedData

For message encryption

Similar to PGP

- create a random session key, encrypt the message with that key and a conventional crypto, encrypt the session key with recipient’s public key

SignedData

For signed message

- both message and signature are encoded so that the recipient only sees some ASCII characters if he does not use an email client with S/MIME support

Similar to PGP

- first message is hashed, then the hash is encrypted using sender’s private key

Message, signature, identifiers of algorithms and the sender’s certificate are packed together

- again difference between S/MIME and PGP in trust management

S/MIME

Secure/Multipurpose Internet Mail Extensions

A standard way for email encryption and signing

IETF effort (RFCs 2632, 2633 – for version 3.0; RFCs 3850, 3851 for version 3.1; 5750, 5751 for version 3.2)

Industry support

Not a standalone software, a system that is to be supported by email clients

- such as MS Outlook and Thunderbird

S/MIME handles digital signatures

- Also provides encryption

S/MIME Certificate Processing

S/MIME uses X.509 v3 certificates

- Certification Authorities (CAs) issue certificates
- unlike PGP, a user cannot be a CA

each client has a list of trusted CA certificates

- actually that list comes with e-mail client software or OS

and own public/private key pairs and certs

It is very hard for an average user to maintain the list of trusted CAs

- Generally OS and/or email client software default trusted CA certificate lists are directly used.
- So trust management is not user-centric in practice

S/MIME Certificate Processing and CAs

One should obtain a certificate from a CA in order to send signed messages

Certificates classes (common practice by most CAs)

- Class 1
- Class 2
- Class 3

↓ Stronger identity validation

↑ Easier to issue

CA certification policies (Certificate Practice Statement)

- ID-control practices
  - Class 1: only email address check
  - Class 2: class1 + against third party database / fax documents
  - Class 3: class1 + apply in person and submit picture IDs and/or paper documents

Pretty Good Privacy (PGP)

widely used secure e-mail software
- originally a file encryption/decryption facility

developed by Phil Zimmermann
- a security activist who has had legal problems due to PGP

best available crypto algorithms are employed

available on several platforms with source code

originally free, now commercial versions exist

not controlled by a standardization body
- although there are RFCs

PGP Mechanisms

Digital Signatures (and consequently message authentication and integrity)

- RSA, DSS, and others

Message Encryption

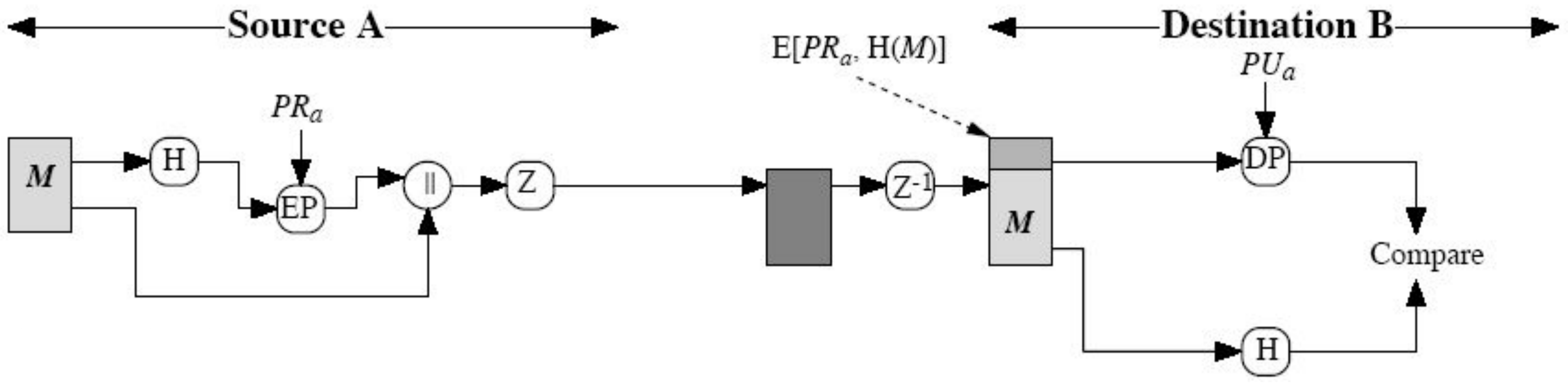
- CAST, IDEA, 3DES, AES, etc. (all at least 128 bits)
- symmetric keys are used once and encrypted using RSA or ElGamal (based on discrete logs)

Compression using ZIP

Radix-64 conversion (to ASCII)

- for e-mail compatibility

PGP Operation – Digital Signatures

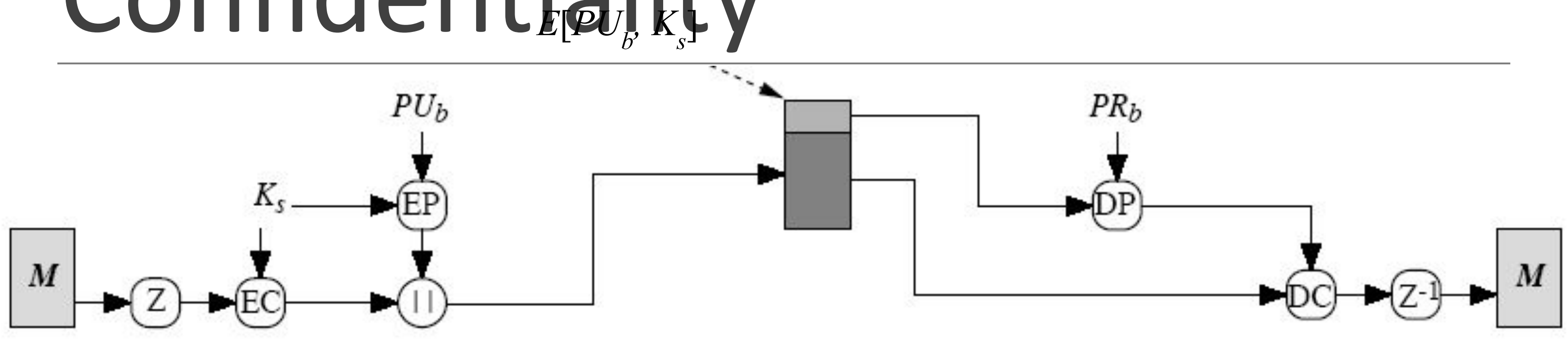


Classical application of public key crypto
- This figure is actually for RSA
- for DSA refer to previous lectures

Z is zip function

radix-64 conversion is done after zip at sender, before Z^{-1} at receiver
- may be done only for signature or for the whole message

PGP Operation – Confidentiality



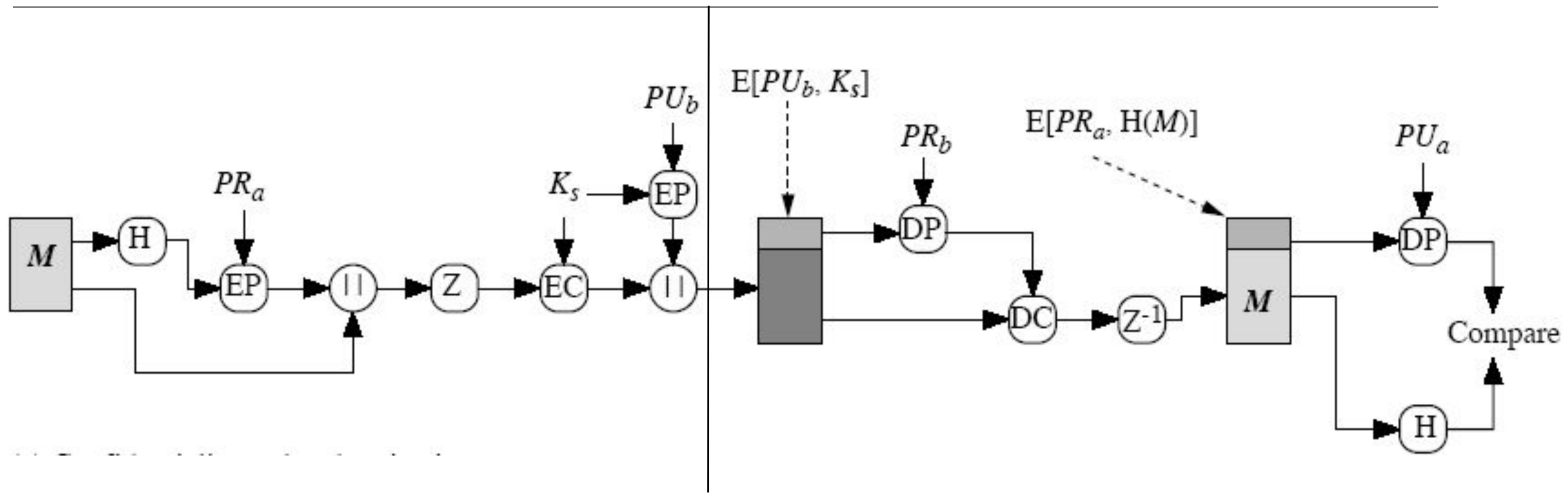
One-time session key, K_s

- generated at random
- encrypted using a public key cryptosystem, EP
  - RSA or ElGamal

Message is compressed before encryption

- This is the default case

PGP Operation – Confidentiality and Authentication



uses both services on same message

- create signature and attach to message
- compress and encrypt both message & signature
- attach encrypted session key
- radix-64 conversion is for everything at the end

PGP Operation – radix-64 conversion

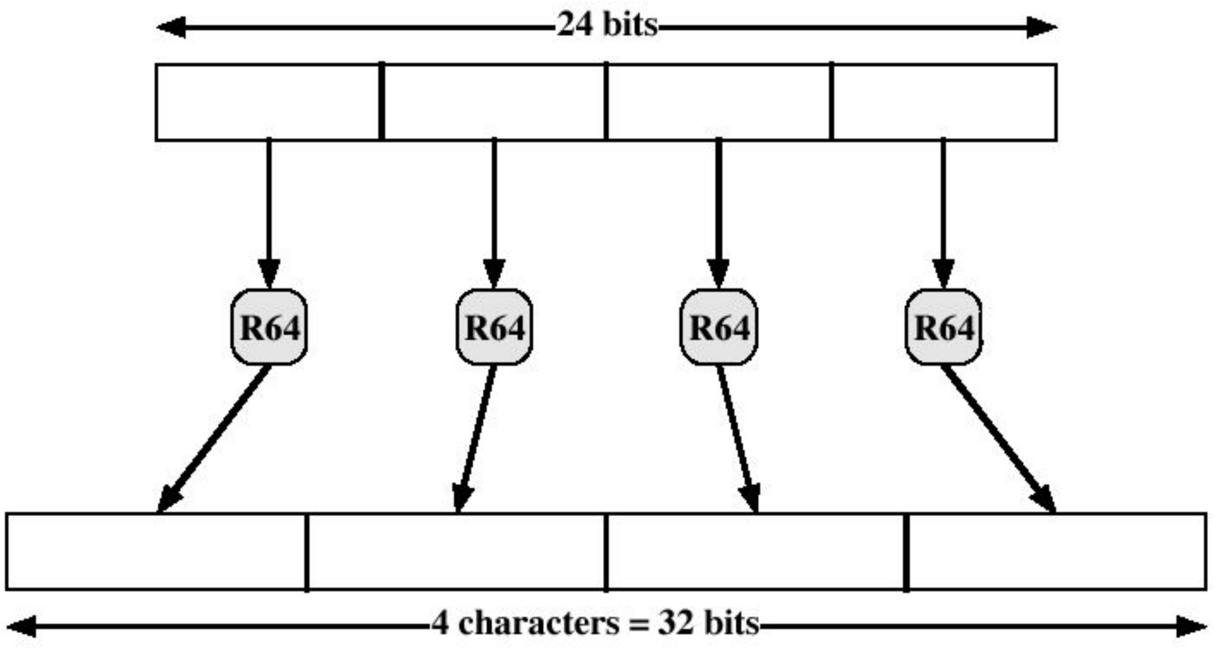
Encrypted text and signatures create binary output

however email was designed only for text

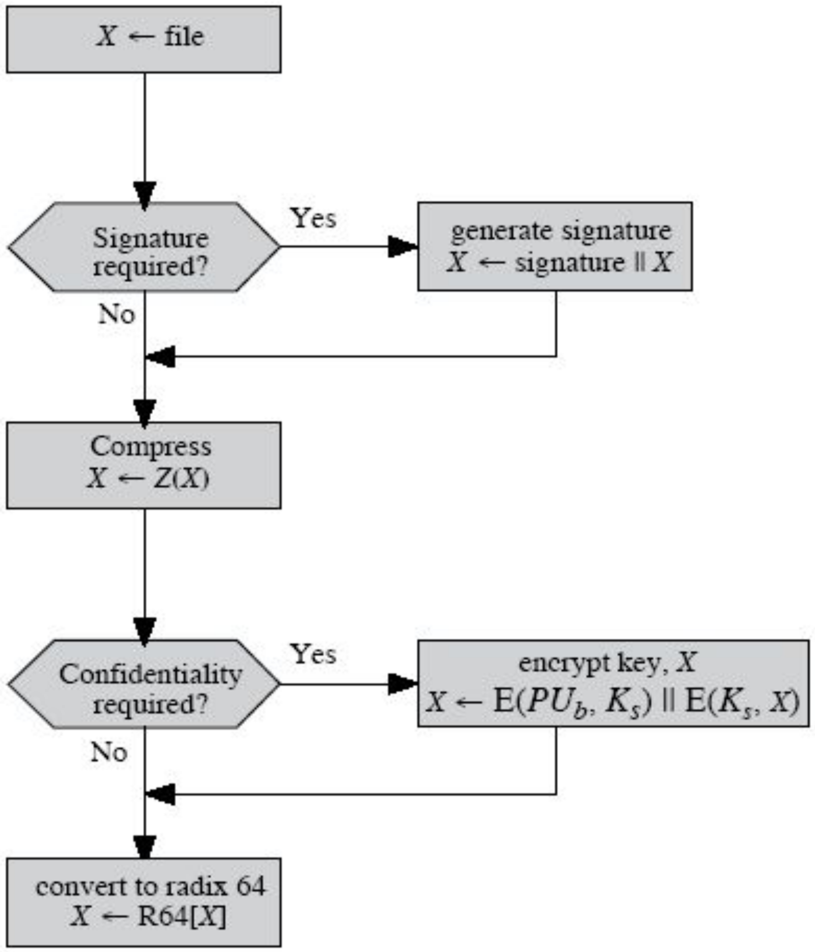
- hence PGP must encode raw binary data into printable ASCII characters

uses radix-64 algorithm

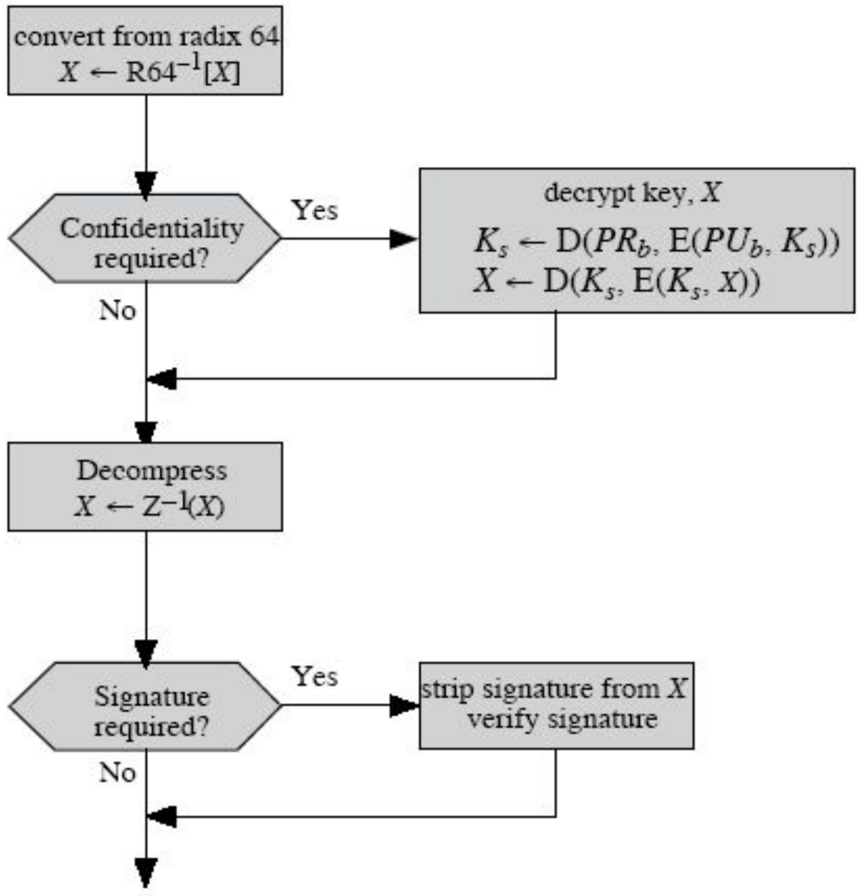
maps 3 bytes to 4 printable chars



PGP Operation – Summary


$X \leftarrow$ file
Signature required?
Yes
generate signature $X \leftarrow$ signature $\Vert$ X
No
Compress $X \leftarrow Z(X)$
Confidentiality required?
Yes
encrypt key, X $X \leftarrow E(PU_b, K_s) \Vert E(K_s, X)$
No
convert to radix 64 $X \leftarrow R64[X]$

(a) Generic Transmission Diagram (from A)


convert from radix 64 $X \leftarrow R64^{-1}[X]$
Confidentiality required?
Yes
decrypt key, X $K_s \leftarrow D(PR_b, E(PU_b, K_s))$ $X \leftarrow D(K_s, E(K_s, X))$
No
Decompress $X \leftarrow Z^{-1}(X)$
Signature required?
Yes
strip signature from X verify signature
No

(b) Generic Reception Diagram (to B)

PGP Key ID concept

since a user may have many public/private keys in use, there is a need to identify which is actually used to encrypt the session key in a message

- PGP uses a key identifier which is least significant 64-bits of the public key
- uniqueness?
  - very likely, at least for a particular user ID (e-mail address)

Key IDs are used in signatures too

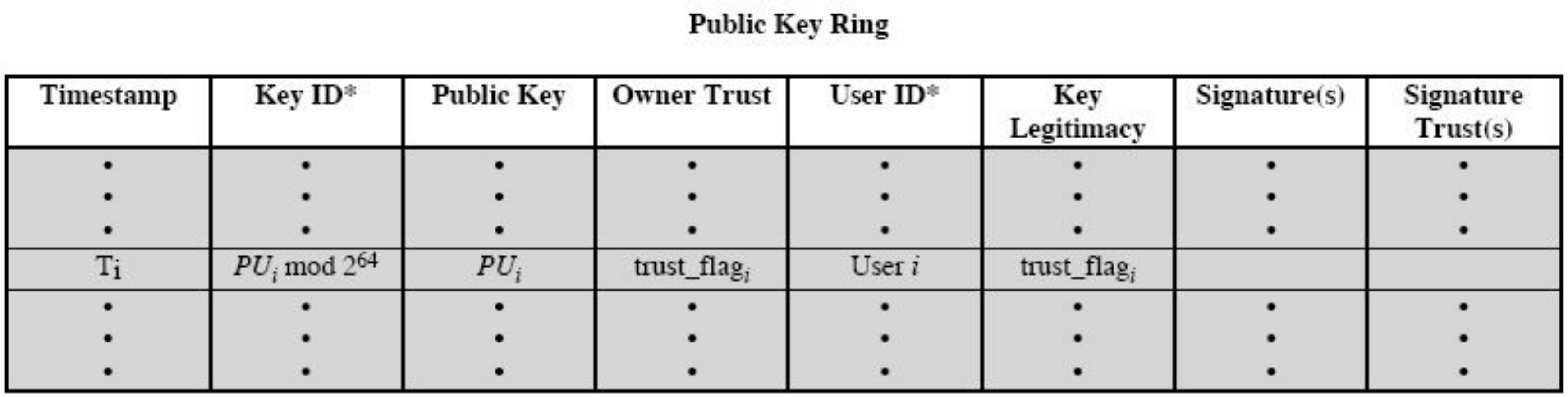
- key ID for the public key corresponding to the private key used for signature

Key IDs are sent together with messages

PGP Key Rings

each PGP user has a pair of keyrings to store public and private keys

- public-key ring contains all the public-keys of other PGP users known to this user



Public Key Ring

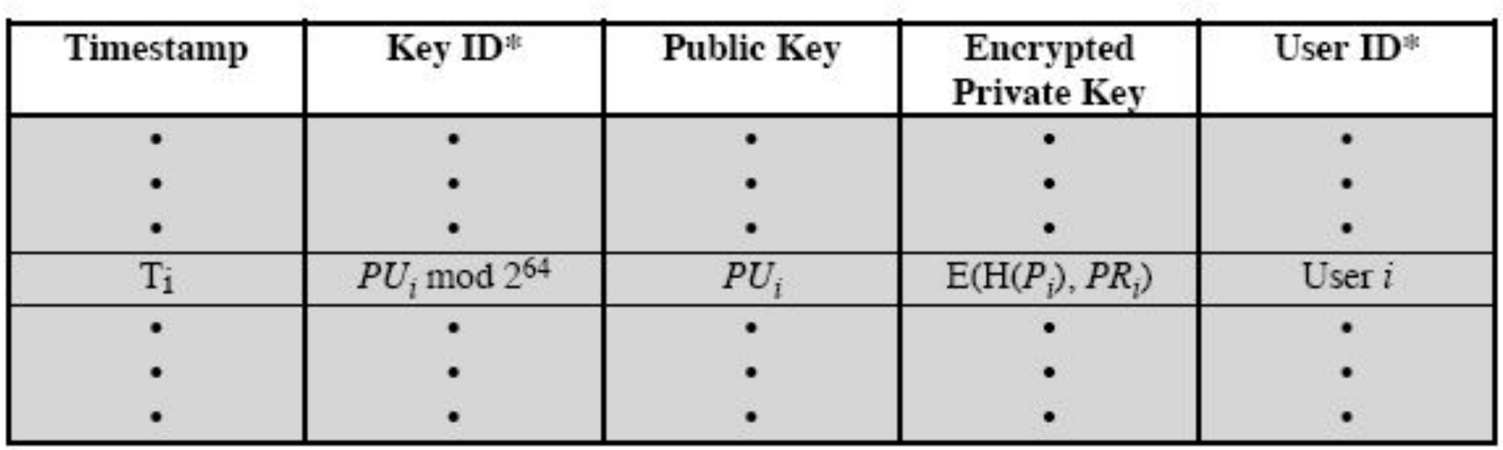
Timestamp	Key ID*	Public Key	Owner Trust	User ID*	Key Legitimacy	Signature(s)	Signature Trust(s)
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •
T_i	PU_i mod 2^{64}	PU_i	trust_flag$_i$	User i	trust_flag$_i$		
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •

* Key ID, Used ID combination uniquely identifies a key

PGP Key Rings

private-key ring contains the public/private key pair(s) for this user, private keys are encrypted using a key derived from a hashed passphrase

Private Key Ring



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

* Key ID, Used ID combination uniquely identifies a key

PGP Key Management - 1

From PGP documentation:

“This whole business of protecting public keys from tampering is the most difficult problem in practical public key applications”

You have to make sure about the legitimacy of the public key of your party

- exchange public-keys manually (using CDs, USB sticks, etc.)
- verify fingerprint of a public key over the phone
- trust another individual who signs public keys
  - public key signatures

PGP Key Management - 2

Public keys could be signed by

- Certification Authorities (CA)
  - trusted entities
  - the mechanism of S/MIME, not in PGP
- in PGP each user is a CA
  - everybody can sign keys of users they know directly
  - other users’ key signatures can also be used, if those users are trusted

The only ultimately trusted entity is yourself

- all other keys should either be directly signed by you or there should be a trusted path of key signatures
- you reflect your own trust assessment in your public key ring (no system enforcement)
- key ring includes trust indicators
- “web of trust”

PGP Key Management - 3

A trusted signature on a public key means that

- the key really belongs to its owner

But does not mean that key owner is trusted to sign other keys

- key owner can sign other keys, but their trustworthiness is determined by the verifier (the owner of the pubkey ring)

Making sure about the legitimacy of a key and trusting the key owner to find out other keys are two different concepts

Keys and signatures on them are generally obtained from PGP public keyservers

- there might be several signatures on a single key

Certificates

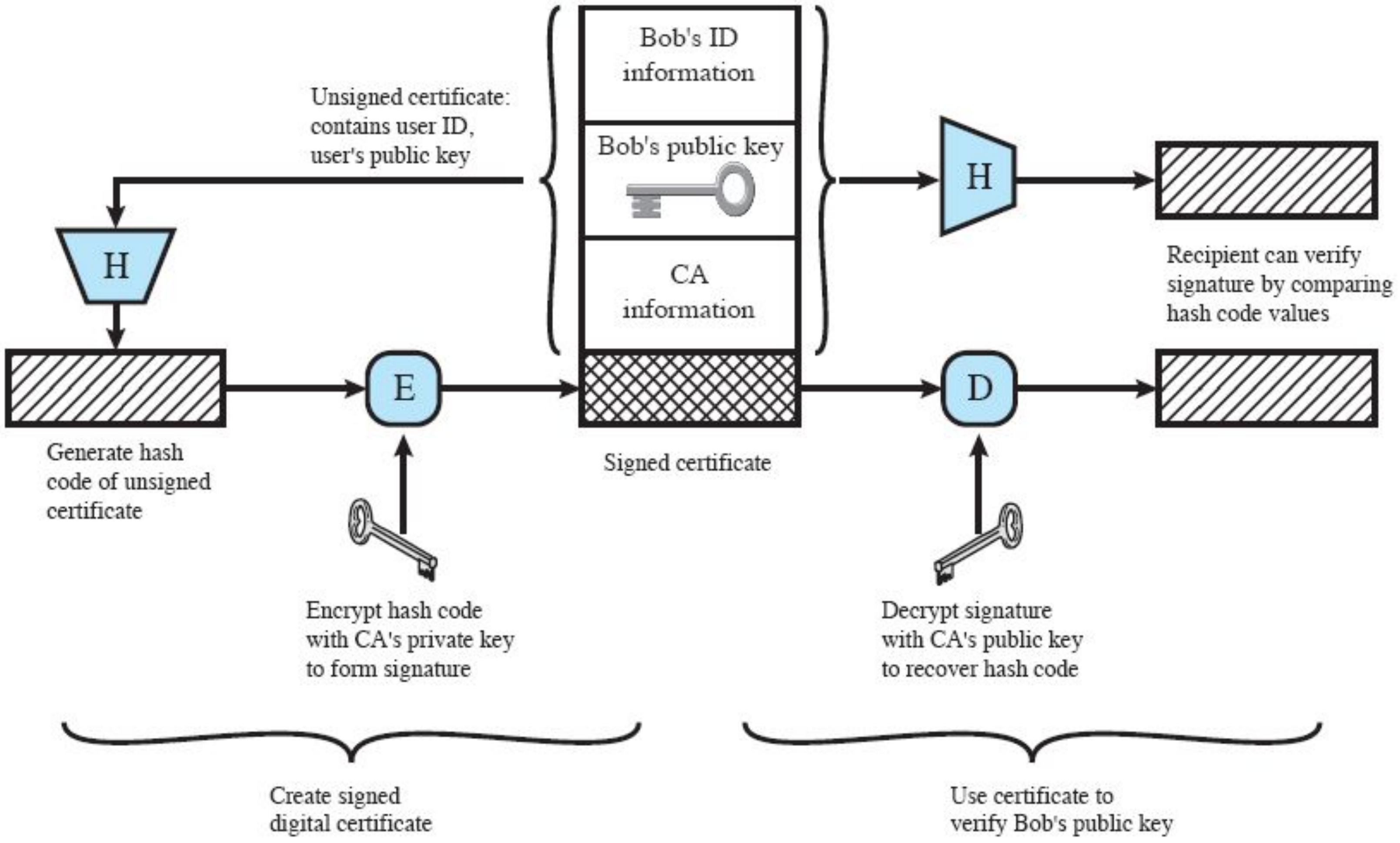
Yet another public-key distribution method

- first (conceptually) offered by Kohnfelder (1978)

Binding between the public-key and its owner

Issued (digitally signed) by the Certificate Authority (CA)

Certificates


Bob's ID information
Bob's public key
CA information
Unsigned certificate: contains user ID, user's public key
H
H
E
D
Generate hash code of unsigned certificate
Signed certificate
Recipient can verify signature by comparing hash code values
Encrypt hash code with CA's private key to form signature
Decrypt signature with CA's public key to recover hash code
Create signed digital certificate
Use certificate to verify Bob's public key

Certificates

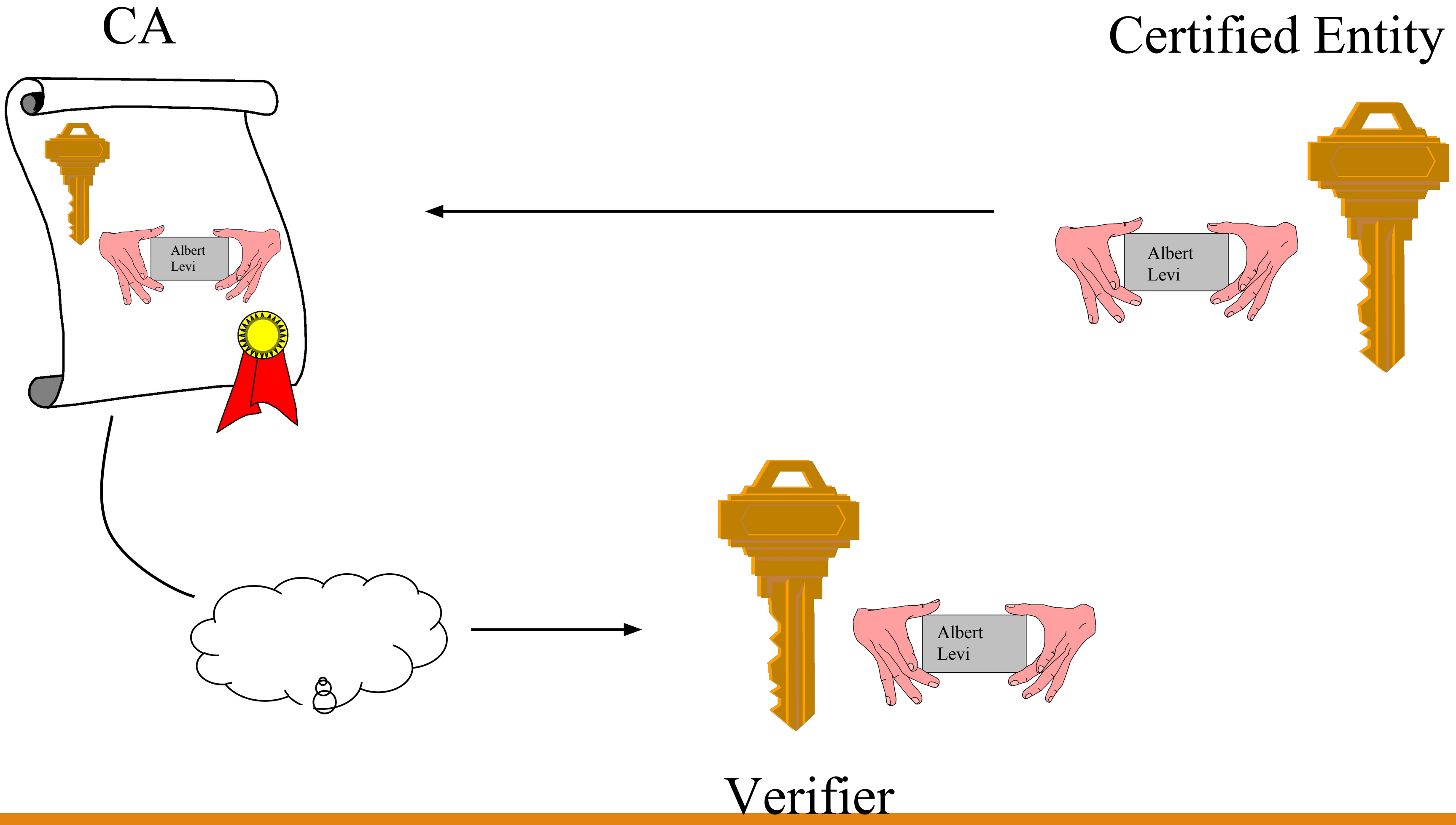
Certificates are verified by the verifiers to find out correct public key of the target entity

Certificate verification is the verification of the signature on certificate

In order to verify a certificate, the verifier

- must know the public key of the CA
- must trust the CA

Certificates



Issues Related Certificates

CA certification policies (Certificate Practice Statement)

- how reliable is the CA?
- certification policies describe the methodology of certificate issuance
- ID-control practices
  - loose control: only email address
  - tight control: apply in person and submit picture IDs and/or paper-based documentation

Issues Related Certificates

TRUST

- verifiers must trust CAs
- CAs need not trust the certified entities
- certified entities need not trust its CA

What is “trust” in certification systems?

- Answer to the question: “How correct is the certificate information?”
- related to certification policies

Issues Related Certificates

Certificate Revocation

- certificates have lifetimes, but they may be revoked before the expiration time
- Reasons:
  - certificate holder key compromise/lost
  - CA key compromise
  - end of contract (e.g. certificates for employees)
- Certificate Revocation Lists (CRLs) hold the list of certificates that are not expired but revoked
  - each CA periodically issues such a list with digital signature on it

Real World Analogies

Is a certificate an “electronic identity”?

Concerns

- a certificate is a binding between an identity and a key, not a binding between an identity and a real person
- anyone can submit someone else’s certificate
- one must submit its certificate to identify itself, but submission is not sufficient, the key must be used in a protocol

Real World Analogies

Result: Certificates are not picture IDs

So, what is the real world analogy for certificates?

- Endorsed document/card that serves as a binding between the identity and signature

Public Key Infrastructure (PKI)

PKI is a complete system and well-defined mechanisms for certificates

- certificate issuance
- certificate revocation
- certificate storage
- certificate distribution

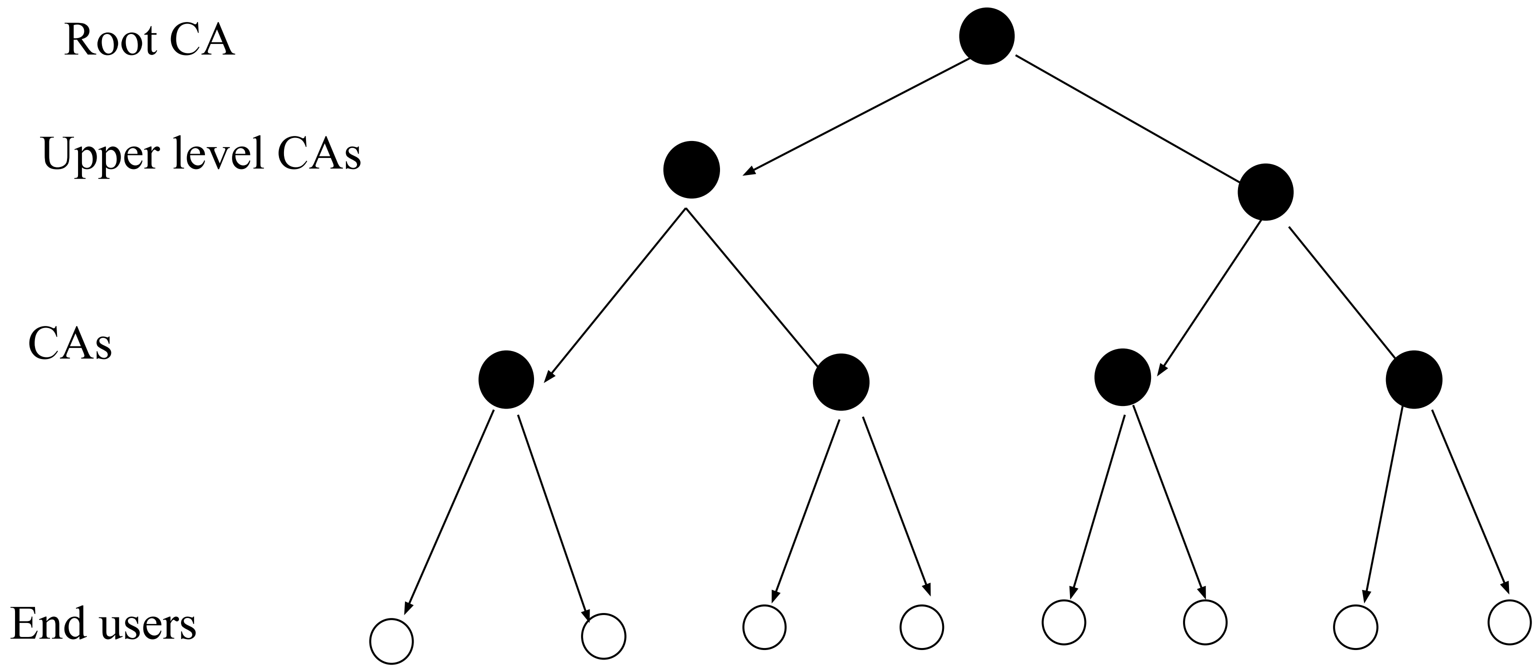
PKI

Business Practice: Issue certificates and make money
- several CAs

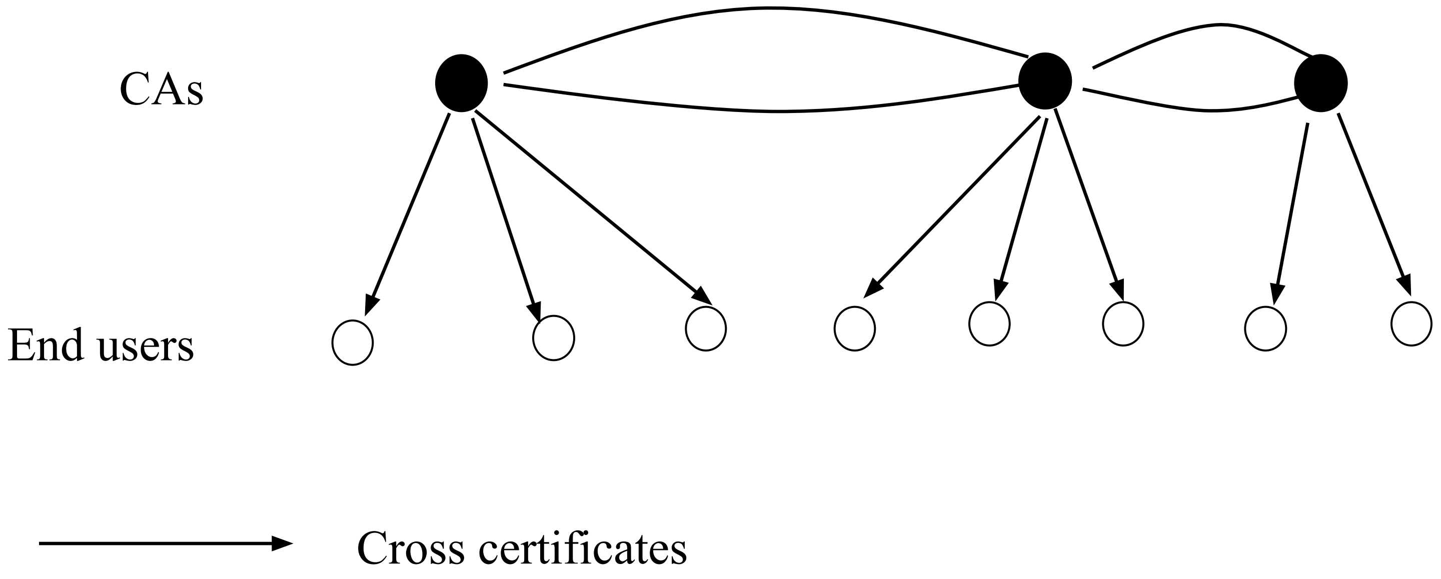
Several CAs are also necessary due to political, geographical and trust reasons

3 interconnection models
- hierarchical
- cross certificates
- hybrid

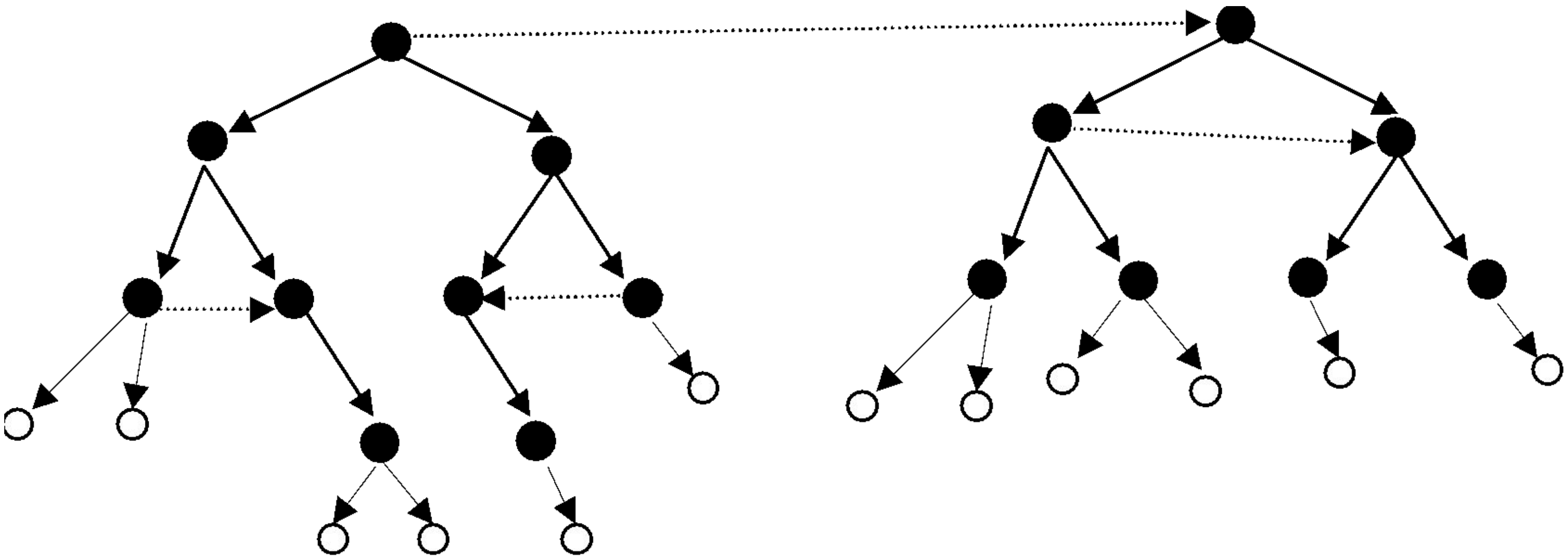
Hierarchical PKI Example



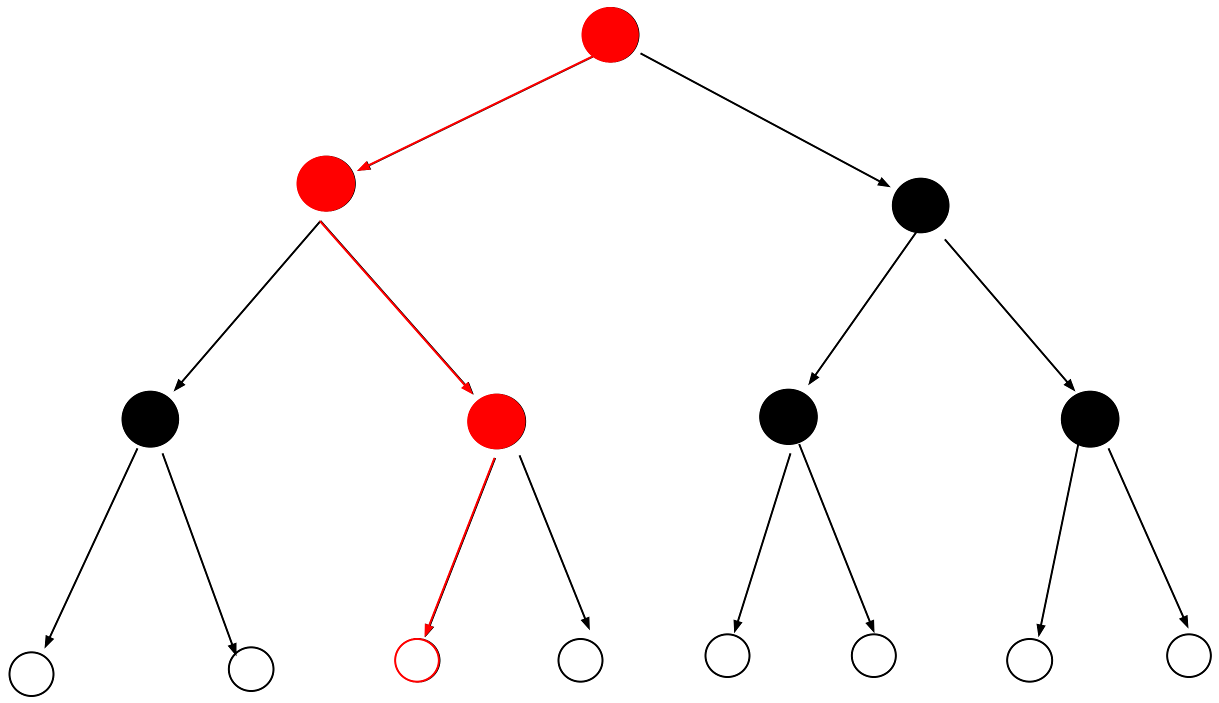
Cross Certificate Based PKI Example



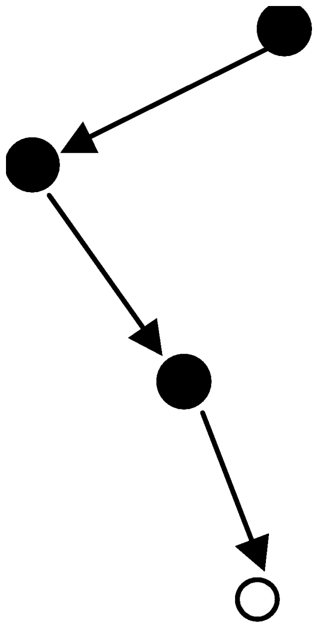
Hybrid PKI example



Certificate Paths



Certificate Paths



Verifier must know public key of the first CA

Other public keys are found out one by one

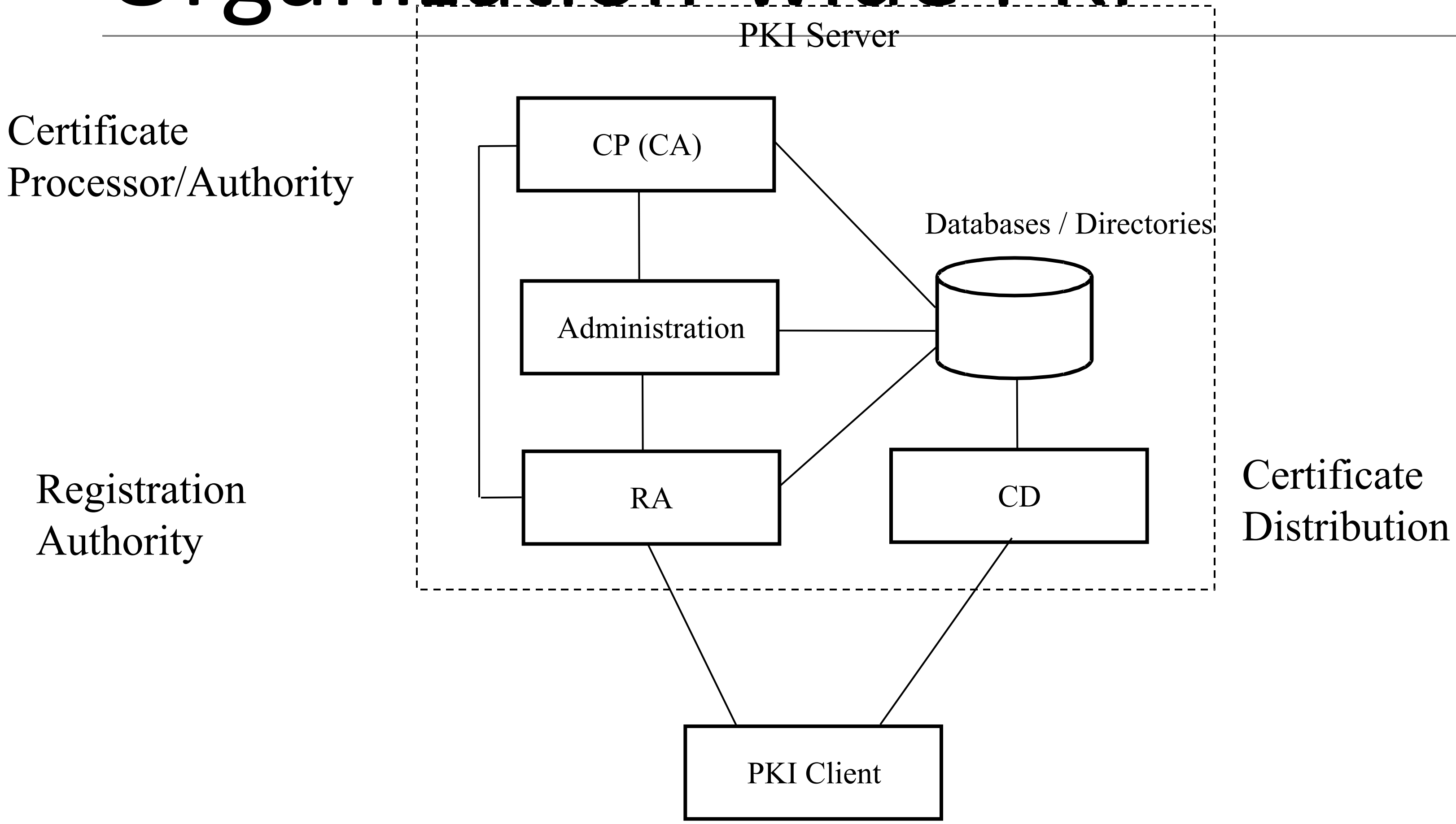
All CAs on the path must be trusted by the verifier

Organization-wide PKI

Local PKI for organizations

- may have global connections, but the registration facilities remain local
- generally to solve local problems
  - local secure access to resources

Organization-wide PKI



Architecture of a typical organization-wide PKI

Hosted vs. Standalone PKI

Hosted (outsourced) PKI

- PKI vendor acts as CA
- PKI owner is the RA

Standalone PKI

- PKI owner is both RA and CA

Hosted vs. Standalone PKI

Standalone PKI	**Hosted PKI**
Organization has to have a secure server for certificate issuance and processing.	Organization does not need to run a secure server for certificate processing.
Organization must issue cross certificates or has to have some other arrangements for universal connection of its PKI. Otherwise, the PKI remains local.	PKI provider (host) already has such arrangements. Organization does not have to worry about worldwide visibility of its PKI.
More administrative work for organization.	Less administrative work for organization.
No continuous dependency on the PKI vendor. Organization does not have to pay periodic fees.	Continuous dependency on the PKI vendor (host). The organization must pay regular fees to the host based on the certificate volume.
Security of the PKI is in the organization's hands.	Although the organization is responsible for the security of its PKI, they are dependent on the host's security.
Organization does not have to trust the PKI vendor as different than its other software vendors.	Ultimate trust to host is indispensable.
The only user of the private key is the organization itself.	Private key is being used by the host for certificate issuance.

X.509

ITU-T standard (recommendation)

- ISO 9495-2 is the equivalent ISO standard

part of X.500 family for “directory services”

- distributed set of servers that store user information
  - an utopia that has never been carried out
- X.509 defines the authentication services and the pubic-key certificate structure (certificates are to be stored in the directory)
- so that the directory would contain public keys of the users

X.509

Defines identity certificates

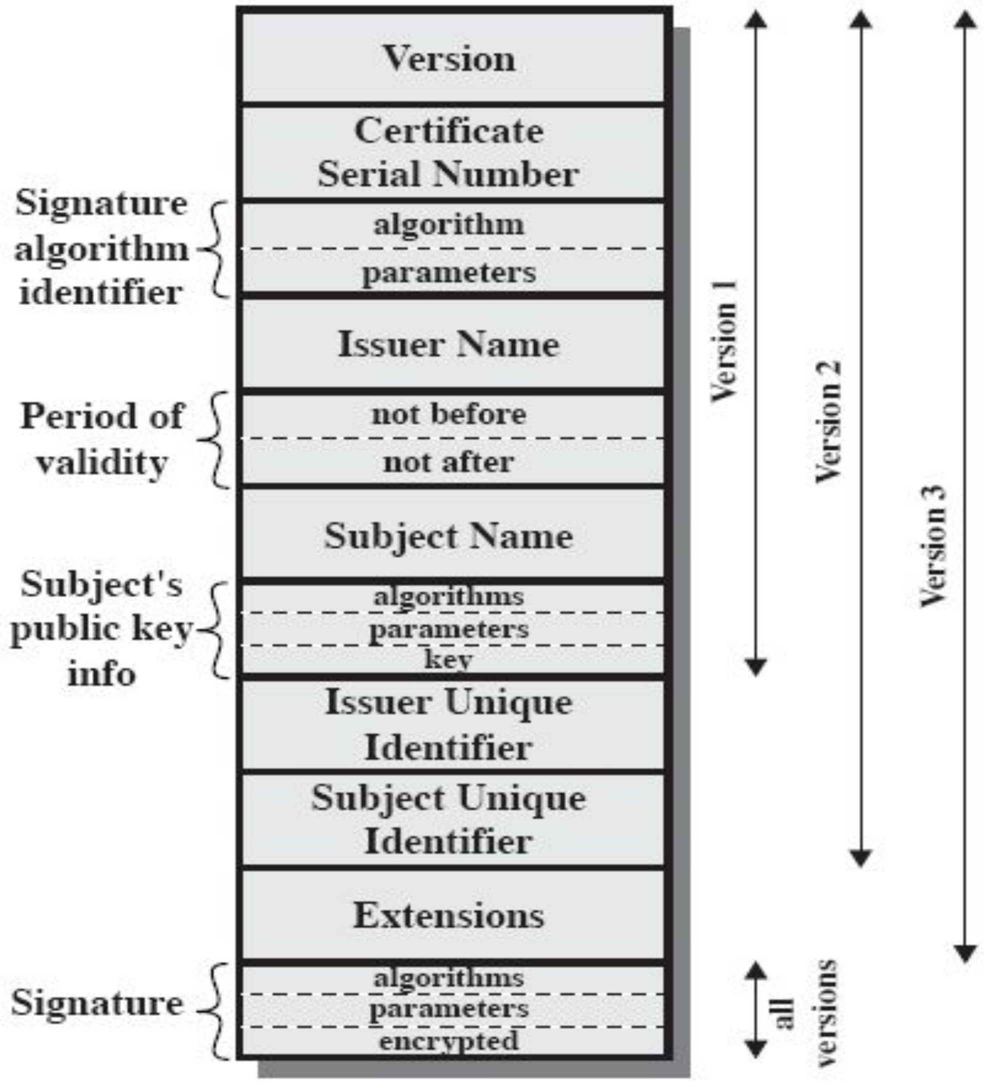
- attribute (authorization) certificates are added in 4th edition (2000)

Defines certificate structure, not PKI

Supports both hierarchical model and cross certificates

End users cannot be CAs

X.509 Certificate Format



X.509v3 Extensions

Not enough flexibility in X.509 v1 and v2

- mostly due to “directory” specific fields
- real-world security needs are different
  - email/URL names should be included in a certificate
  - key identification was missing (so should be included)
  - policy details should indicate under which conditions a certificate can be used (was not the case in v1 and v2)
  - avoidance of blind trust was not possible in v1 and v2

Rather than explicitly naming new fields a general extension method is defined

- An extension consists of an extension identifier, value and criticality indicator

X.509v3 Extensions

Key and policy information
- subject & issuer key identifiers
- indicators of certificate policies supported by the cert
- key usage (list of purposes like signature, encryption, etc)

Alternative names, in alternative formats for certificate subject and issuer

Certificate path constraints
- For CA certs and to restrict certificate issuance based on
  - path length (restricting number of subordinate CAs)
  - policy identifiers
  - names

Verifier could exercise its own restrictions during verification as well
- No blind trust to CAs