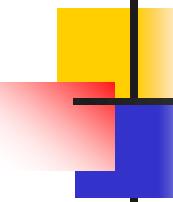




Chapter 16

Security at the Application Layer: PGP and S/MIME



Chapter 16

Objectives

- To explain the general structure of an e-mail application program
- To discuss how PGP can provide security services for e-mail
- To discuss how S/MIME can provide security services for e-mail
- To define trust mechanism in both PGP and S/MIME
- To show the structure of messages exchanged in PGP and S/MIME

16-1 E-MAIL

Let us first discuss the electronic mail (e-mail) system in general.

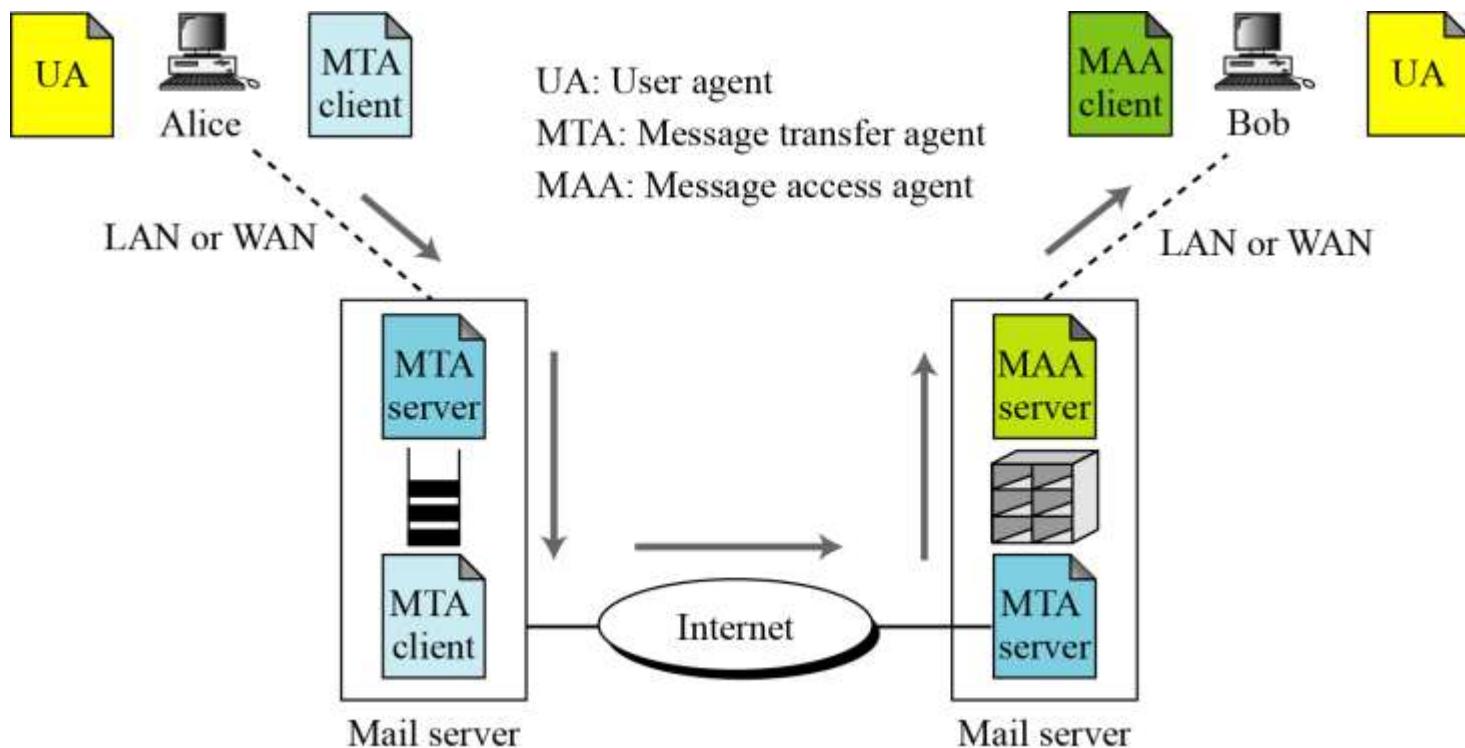
Topics discussed in this section:

16.1.1 E-mail Architecture

16.1.2 E-mail Security

16.1.1 E-mail Architecture

Figure 16.1 E-mail architecture



16.1.2 E-mail Security

Cryptographic Algorithms

Note

In e-mail security, the sender of the message needs to include the name or identifiers of the algorithms used in the message.

Certificates

It is obvious that some public-key algorithms must be used for e-mail security.

16.1.2 Continued

Cryptographic Secrets

Note

In e-mail security, the encryption/decryption is done using a symmetric-key algorithm, but the secret key to decrypt the message is encrypted with the public key of the receiver and is sent with the message.

16-2 PGP

Pretty Good Privacy (PGP) can be used to create a secure e-mail message or to store a file securely for future retrieval.

Topics discussed in this section:

16.2.1 Scenarios

16.2.2 Key Rings

16.2.3 PGP Certificates

16.2.4 Key Revocation

16.2.5 Extracting Information from Rings

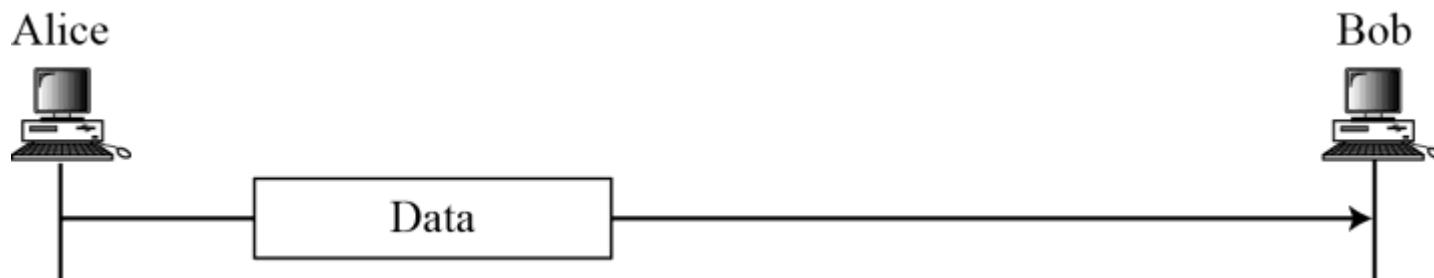
16.2.6 PGP Packets

16.2.7 PGP Messages

16.2.1 Scenarios

Plaintext

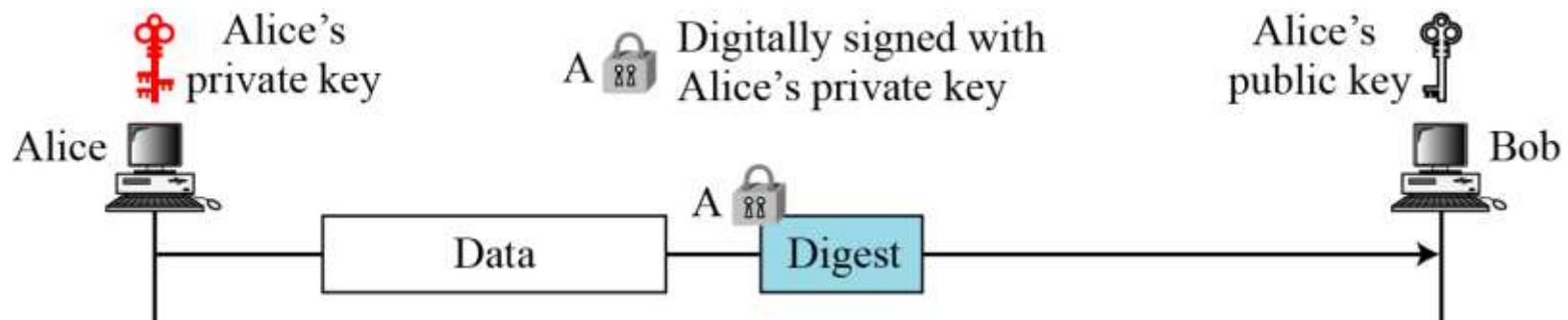
Figure 16.2 *A plaintext message*



16.2.1 *Continued*

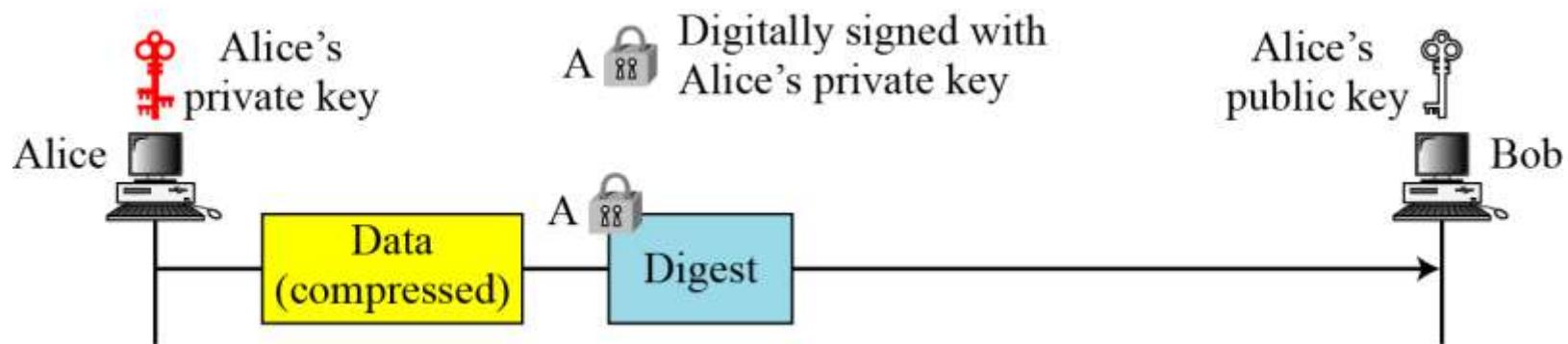
Message Integrity

Figure 16.3 *An authenticated message*



16.2.1 *Continued* *Compression*

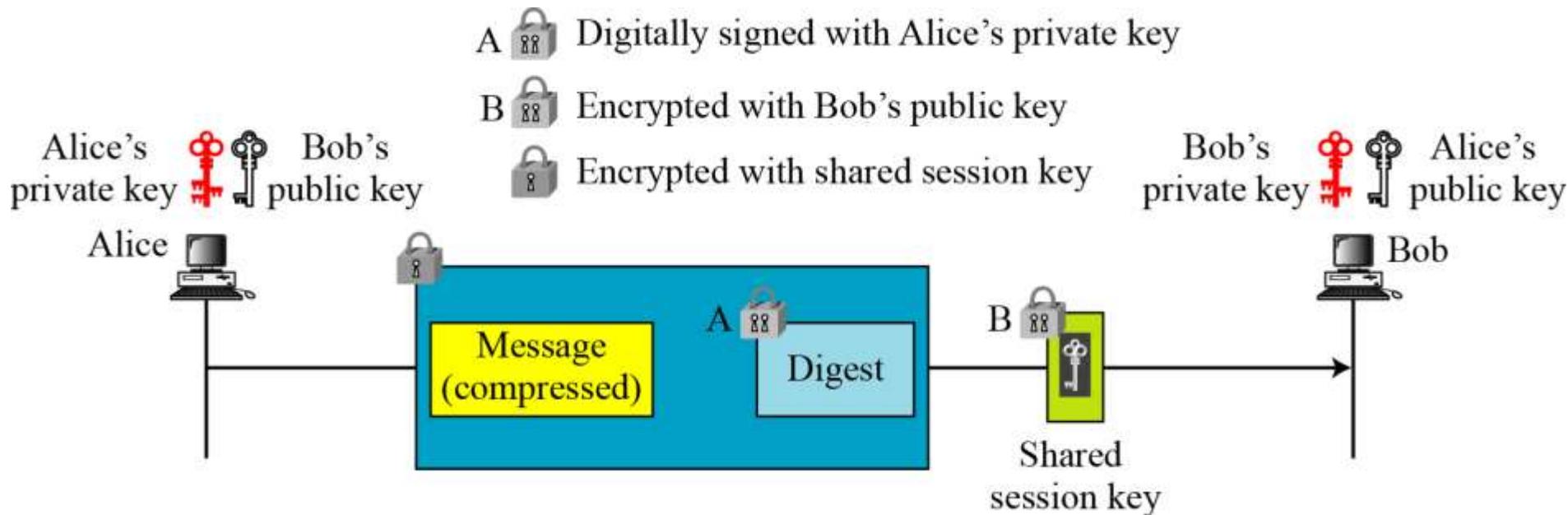
Figure 16.4 *A compressed message*

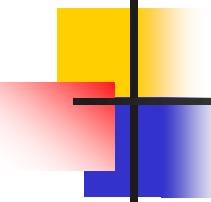


16.2.1 Continued

Confidentiality with One-Time Session Key

Figure 16.5 A confidential message





16.2.1 Continued

Code Conversion

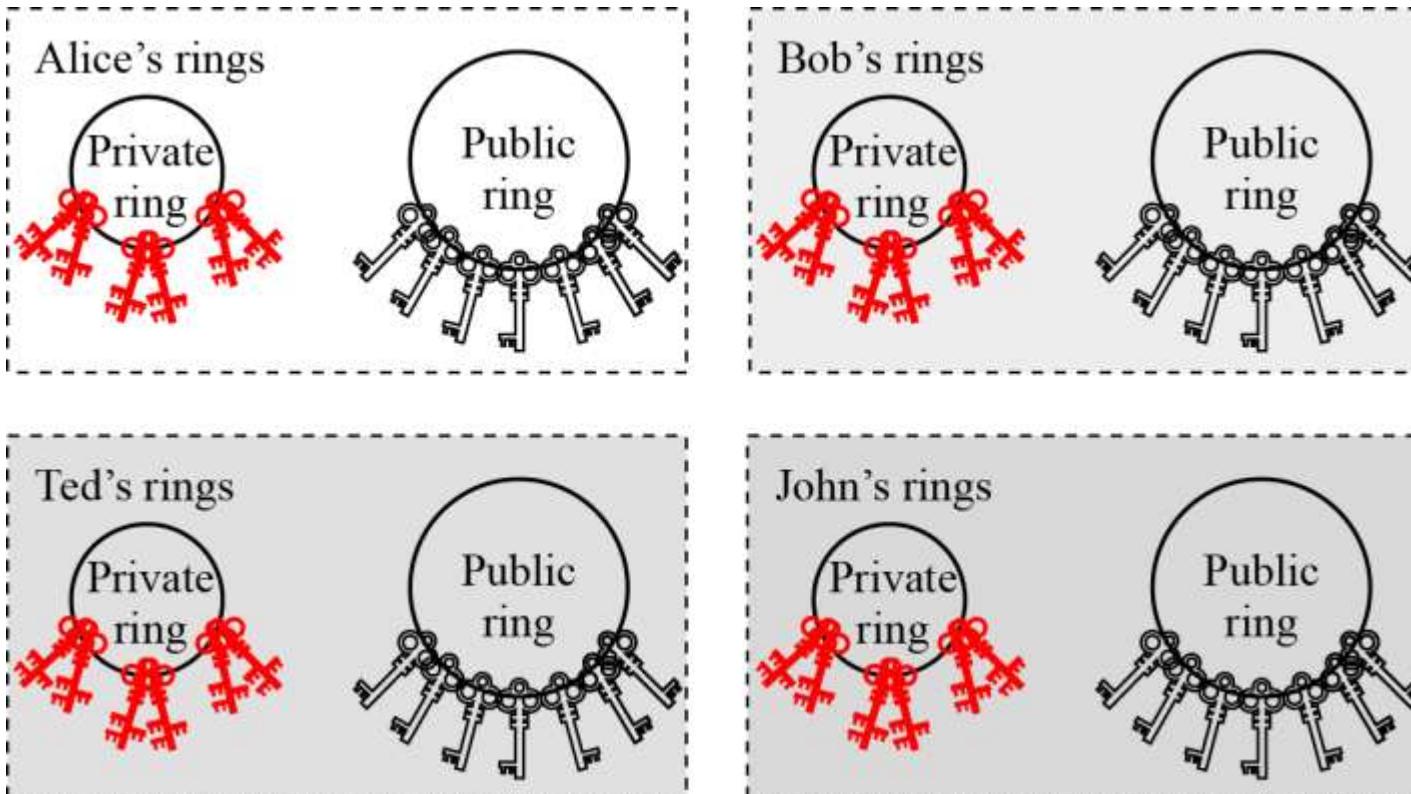
*Another service provided by PGP is code conversion.
PGP uses Radix-64 conversion.*

Segmentation

PGP allows segmentation of the message.

16.2.2 Key Rings

Figure 16.6 Key rings in PGP



16.2.2 *Continued*

PGP Algorithms

Table 16.1 *Public-key algorithms*

<i>ID</i>	<i>Description</i>
1	RSA (encryption or signing)
2	RSA (for encryption only)
3	RSA (for signing only)
16	ElGamal (encryption only)
17	DSS
18	Reserved for elliptic curve
19	Reserved for ECDSA
20	ElGamal (for encryption or signing)
21	Reserved for Diffie-Hellman
100–110	Private algorithms

16.2.2 *Continued*

Table 16.2 *Symmetric-key algorithms*

<i>ID</i>	<i>Description</i>
0	No Encryption
1	IDEA
2	Triple DES
3	CAST-128
4	Blowfish
5	SAFER-SK128
6	Reserved for DES/SK
7	Reserved for AES-128
8	Reserved for AES-192
9	Reserved for AES-256
100–110	Private algorithms

16.2.2 *Continued*

Table 16.3 *Hash Algorithms*

<i>ID</i>	<i>Description</i>
1	MD5
2	SHA-1
3	RIPE-MD/160
4	Reserved for double-width SHA
5	MD2
6	TIGER/192
7	Reserved for HAVAL
100–110	Private algorithms

16.2.2 *Continued*

Table 16.4 *Compression methods*

<i>ID</i>	<i>Description</i>
0	Uncompressed
1	ZIP
2	ZLIP
100–110	Private methods

16.2.3 PGP Certificates

X.509 Certificates

Protocols that use X.509 certificates depend on the hierarchical structure of the trust.

Note

In X.509, there is a single path from the fully trusted authority to any certificate.

16.2.3 Continued

PGP Certificates

In PGP, there is no need for CAs; anyone in the ring can sign a certificate for anyone else in the ring.

Note

In PGP, there can be multiple paths from fully or partially trusted authorities to any subject.

Trusts and Legitimacy

*The entire operation of PGP is based on **introducer trust**, the **certificate trust**, and the **legitimacy of the public keys**.*

16.2.3 Continued

Figure 16.7 Format of private key ring table



User ID	Key ID	Public key	Encrypted private key	Timestamp
⋮	⋮	⋮	⋮	⋮

16.2.3 *Continued*

Example 16.1

Let us show a private key ring table for Alice. We assume that Alice has only two user IDs, **alice@some.com** and **alice@anet.net**. We also assume that Alice has two sets of private/public keys, one for each user ID.

Table 16.5 *Private key ring table for Example 1*

User ID	Key ID	Public Key	Encrypted Private Key	Timestamp
alice@anet.net	AB13...45	AB13...45...59	32452398...23	031505-16:23
alice@some.com	FA23...12	FA23...12...22	564A4923...23	031504-08:11

16.2.3 Continued

Figure 16.8 Format of a public key ring table



User ID	Key ID	Public key	Producer trust	Certificate(s)	Certificate trust(s)	Key Legitimacy	Timestamp
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

16.2.3 *Continued*

Example 16.2

A series of steps will show how a public key ring table is formed for Alice.

Table 16.6 Example 2, starting table

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F

Table 16.7 Example 2, after Bob is added to the table

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F
Bob...	12...	12.....	F			F

16.2.3 *Continued*

Example 16.2 *Continued*

Table 16.8 Example 2, after Ted is added to the table

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F
Bob...	12...	12.....	F			F
Ted...	48...	48.....	F	Bob's	F	F

Table 16.9 Example 2, after Anne is added to the table

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F
Bob...	12...	12.....	F			F
Ted...	48...	48.....	F	Bob's	F	F
Anne...	71...	71.....	P	Bob's	F	F

16.2.3 *Continued*

Example 16.2 *Continued*

Table 16.10 Example 2, after John is added to the table

User ID	Key ID	Public key	Prod. Trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F
Bob...	12...	12.....	F			F
Ted...	48...	48.....	F	Bob's	F	F
Anne...	71...	71.....	P	Bob's	F	F
John...	31...	31.....	N	Anne's	P	P

16.2.3 *Continued*

Example 16.2 *Continued*

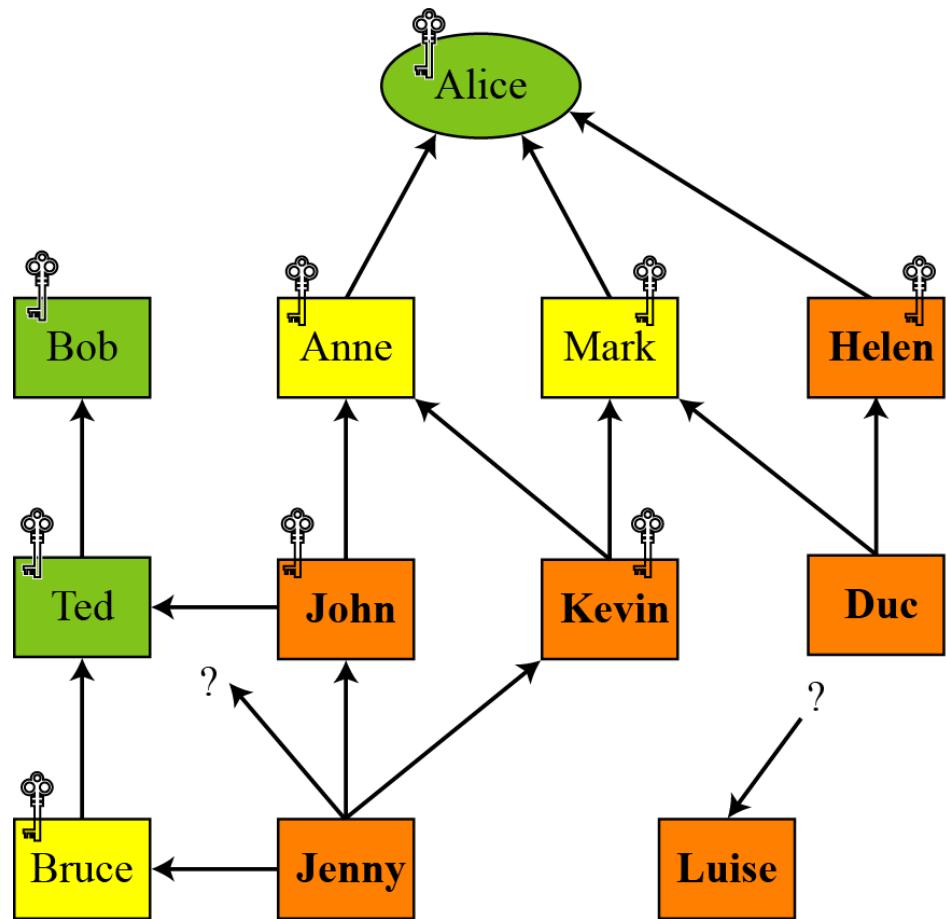
Table 16.11 Example 2, after one more certificate received for John

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F
Bob...	12...	12.....	F			F
Ted...	48...	48.....	F	Bob's	F	F
Anne...	71...	71.....	P	Bob's	F	F
John...	31...	31.....	N	Anne's Ted's	P F	F

16.2.3 Continued

Trust Model in PGP

Figure 16.9 Trust model



X has legitimate key

X → Y X introduced by Y

X → ? X introduced by an unknown entity

Fully trusted entity

Partially trusted entity

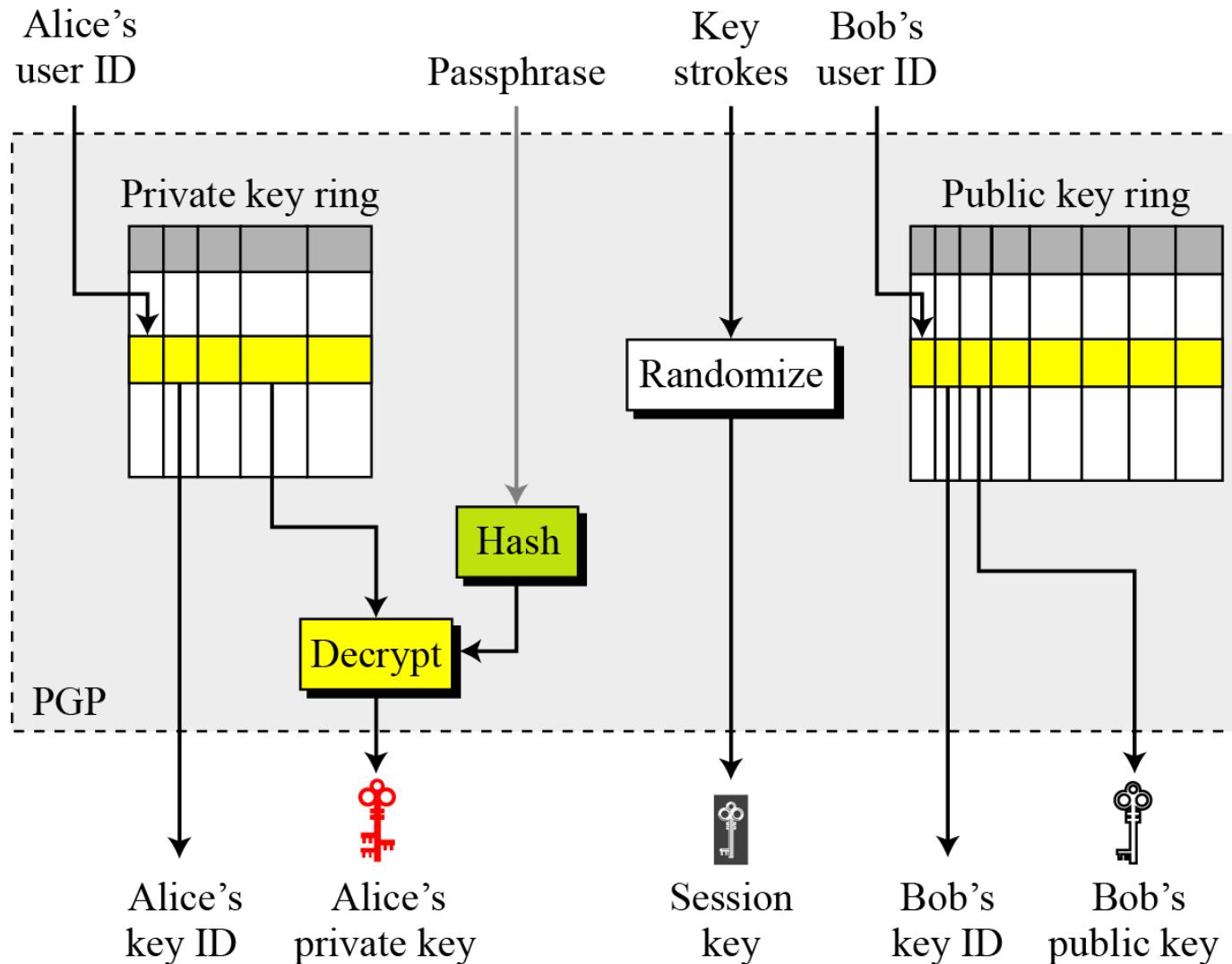
Untrusted entity

16.2.4 Key Revocation

It may become necessary for an entity to revoke his or her public key from the ring. This may happen if the owner of the key feels that the key is compromised (stolen, for example) or just too old to be safe.

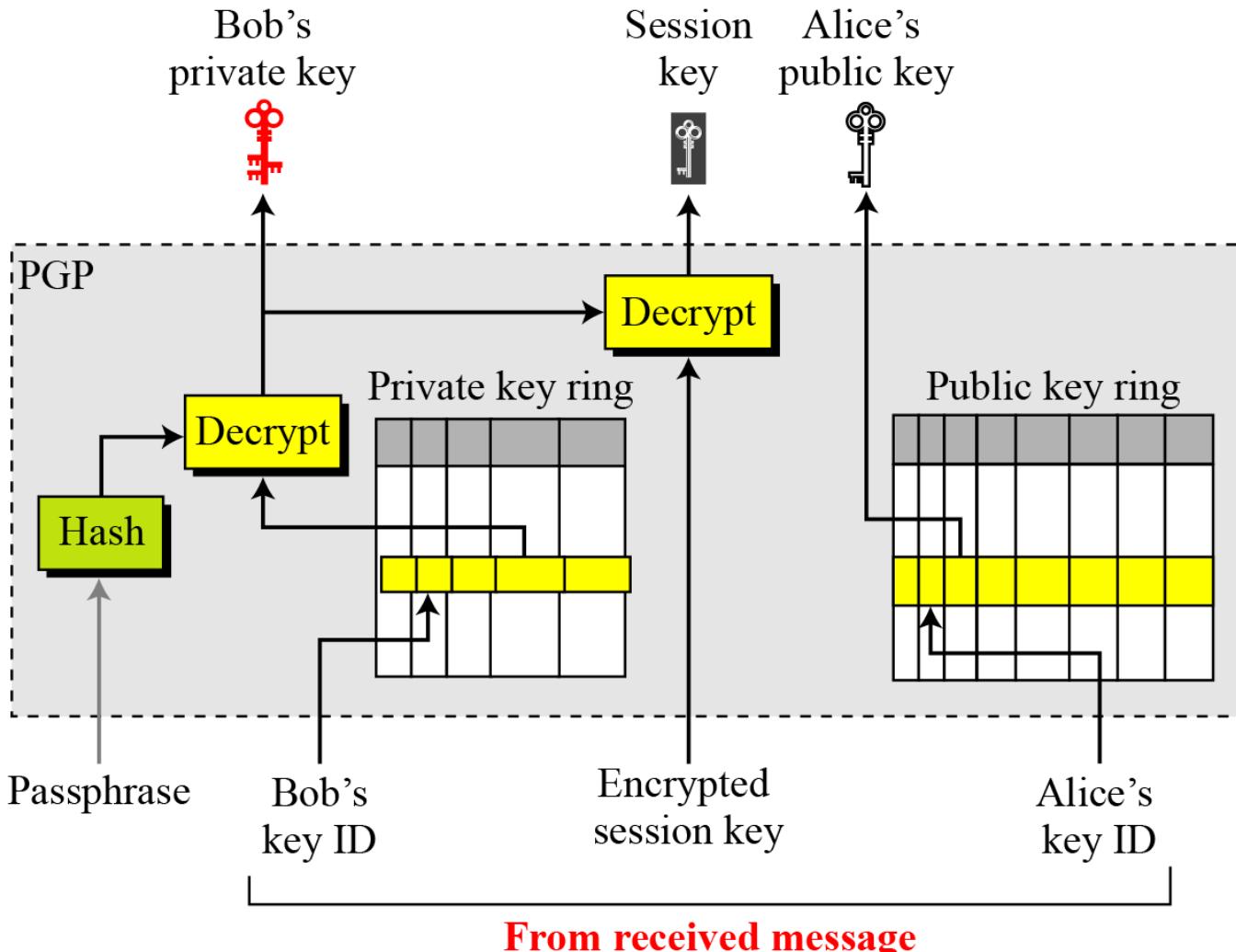
16.2.5 Extracting Information from Rings

Figure 16.10 Extracting information at the sender site



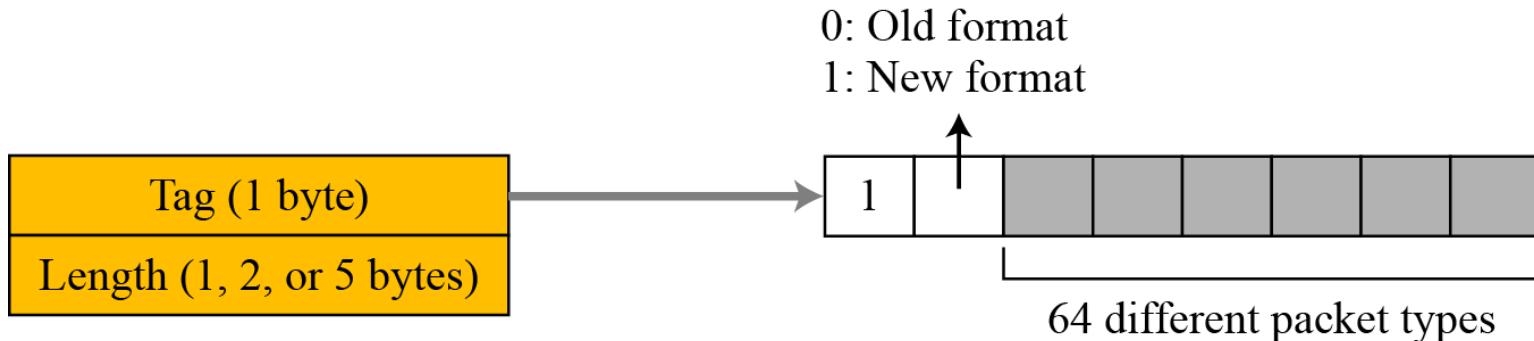
16.2.5 Continued

Figure 16.11 Extracting information at the receiver site



16.2.6 PGP Packets

Figure 16.12 Format of packet header



- a. If the value of the byte after the tag field is less than 192,

body length = first byte

- b. If the value of the byte after the tag field is between 192 and 223 (inclusive),

body length = (first byte - 192) << 8 + second byte + 192

- c. If the value of the byte after the tag field is between 224 and 254 (inclusive),

partial body length = 1 << (first byte & 0x1F)

- d. If the value of the byte after the tag field is 255,

Body length = second byte << 24 | third byte << 16 | fourth byte << 8 | fifth byte

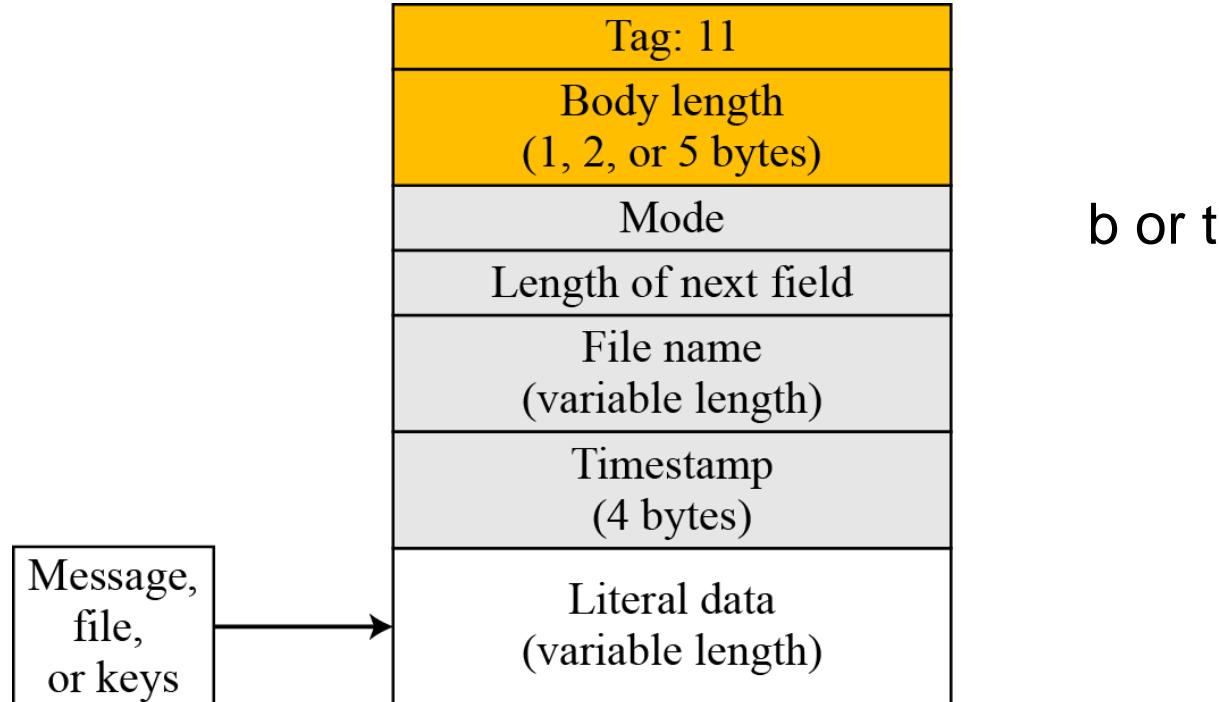
16.2.6 *Continued*

Table 16.12 *Some commonly used packet types*

<i>Value</i>	<i>Packet type</i>
1	Session key packet encrypted using a public key
2	Signature packet
5	Private-key packet
6	Public-key packet
8	Compressed data packet
9	Data packet encrypted with a secret key
11	Literal data packet
13	User ID packet

16.2.6 Continued

Figure 16.13 Literal data packet



16.2.6 Continued

Figure 16.14 Compressed data packet

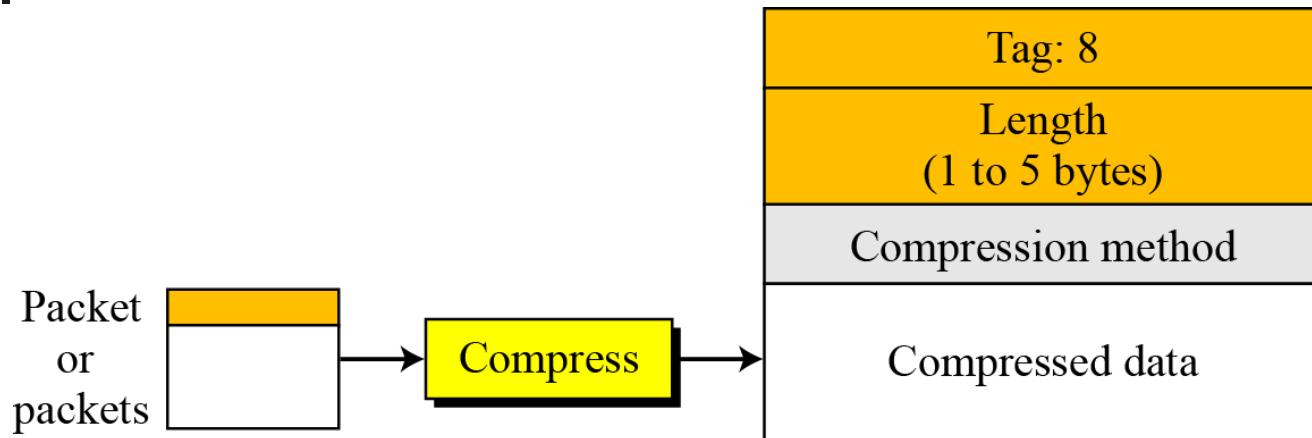
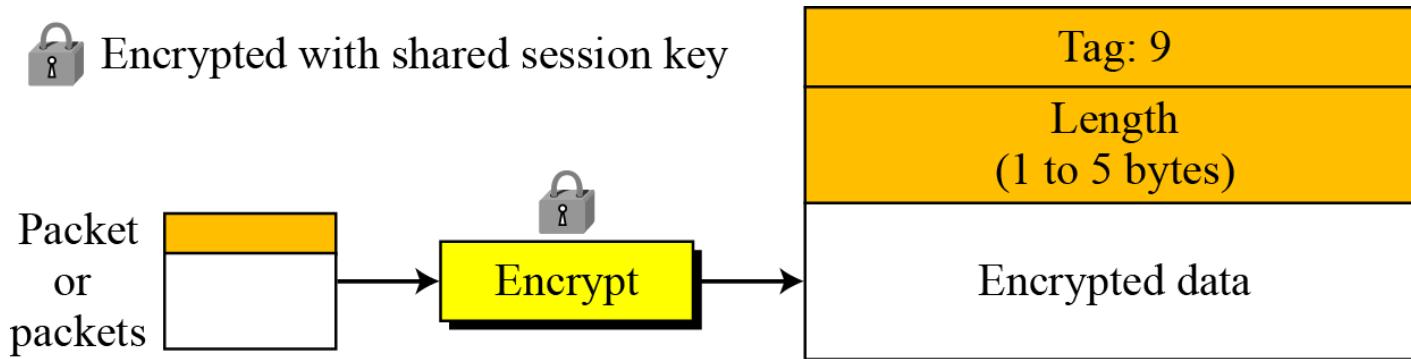


Table 16.4 Compression methods

<i>ID</i>	<i>Description</i>
0	Uncompressed
1	ZIP
2	ZLIP
100–110	Private methods

16.2.6 Continued

Figure 16.15 Encrypted data packet



16.2.6 Continued

Figure 16.16 Signature packet

A Encrypted with Alice's private key

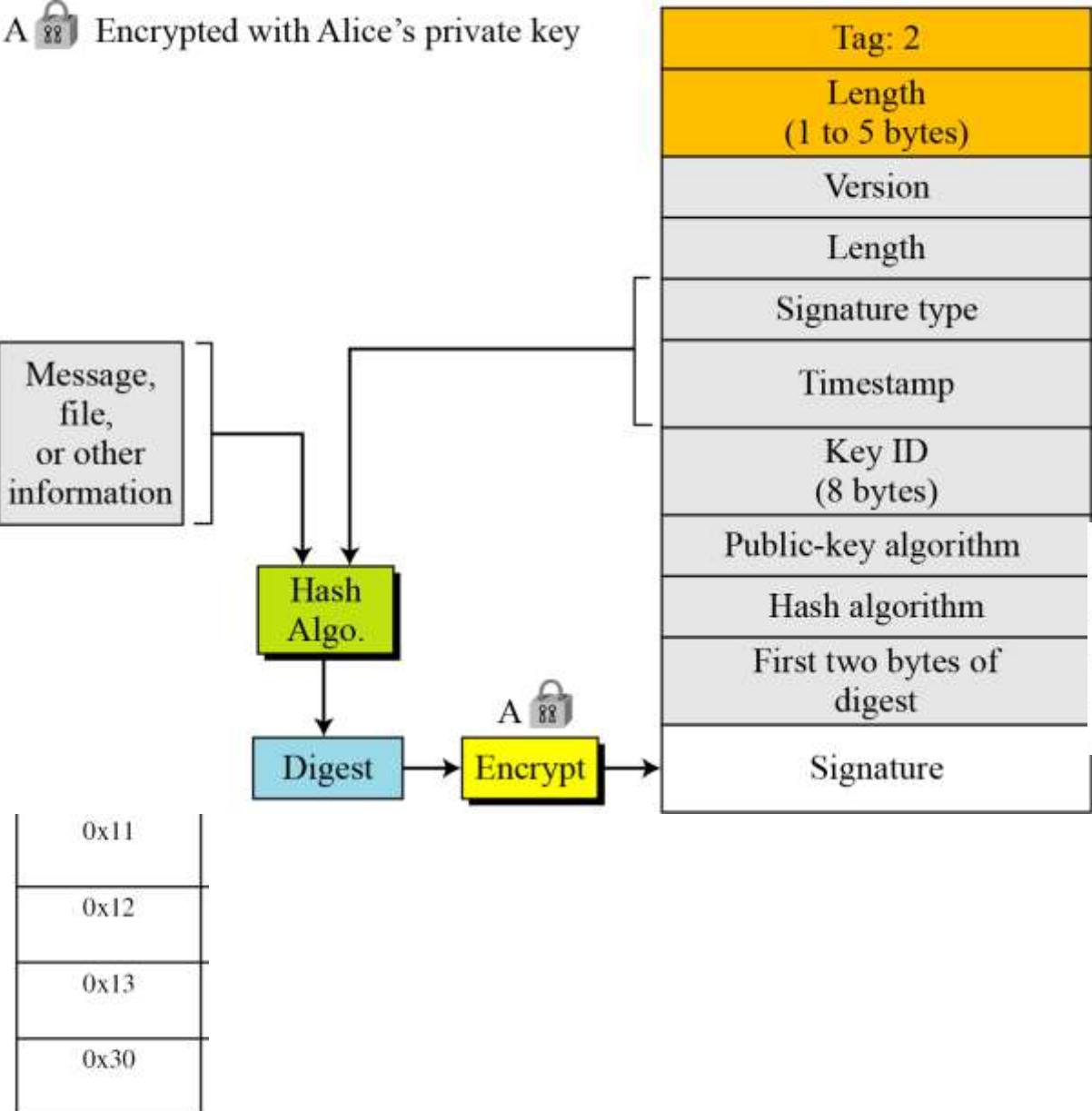


Table 16.3 Hash Algorithms

ID	Description
1	MD5
2	SHA-1
3	RIPE-MD/160
4	Reserved for double-width SHA
5	MD2
6	TIGER/192
7	Reserved for HAVAL
100–110	Private algorithms

ID	Description
1	RSA (encryption or signing)
2	RSA (for encryption only)
3	RSA (for signing only)
16	ElGamal (encryption only)

ID	Description
17	DSS
18	Reserved for elliptic curve
19	Reserved for ECDSA
20	ElGamal (for encryption or signing)
21	Reserved for Diffie-Hellman
100–110	Private algorithms

16.2.6 *Continued*

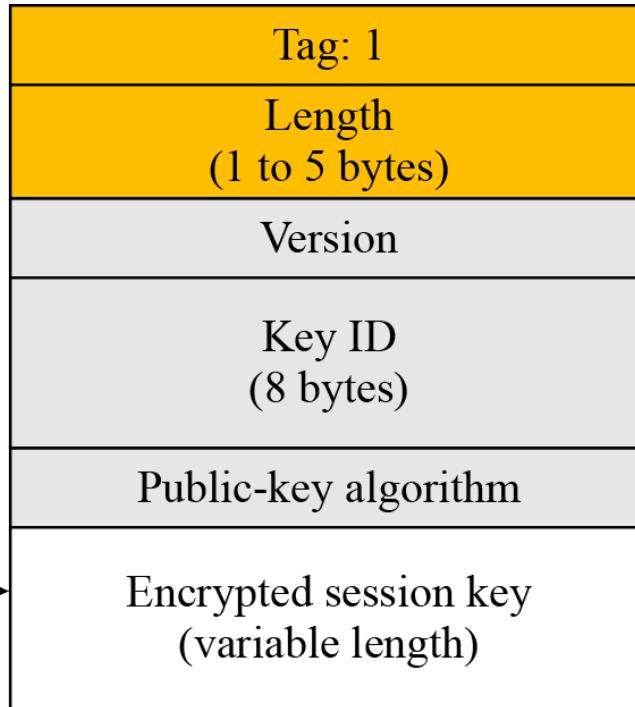
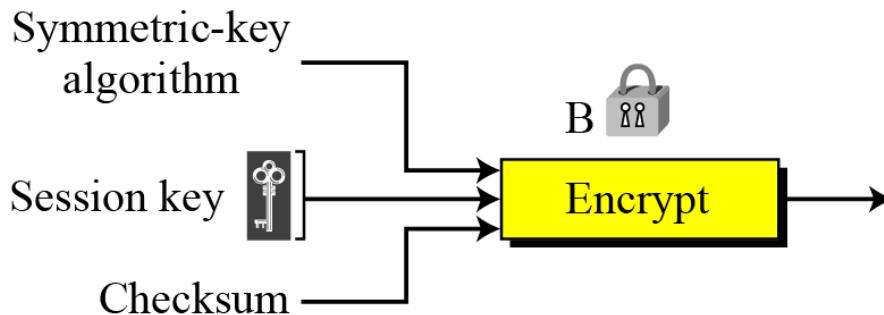
Table 16.13 *Some signature values*

<i>Value</i>	<i>Signature</i>
0x00	Signature of a binary document (message or file).
0x01	Signature of a text document (message or file).
0x10	Generic certificate of a user ID and public-key packet. The signer does not make any particular assertion about the owner of the key.
0x11	Personal certificate of a user ID and public-key packet. No verification is done on the owner of the key.
0x12	Casual certificate of a User ID and public-key packet. Some casual verification done on the owner of the key.
0x13	Positive certificate of a user ID and public-key packet. Substantial verification done.
0x30	Certificate revocation signature. This removes an earlier certificate (0x10 through 0x13).

16.2.6 Continued

Figure 16.17 Session-key packet

B Encrypted with Bob's public key

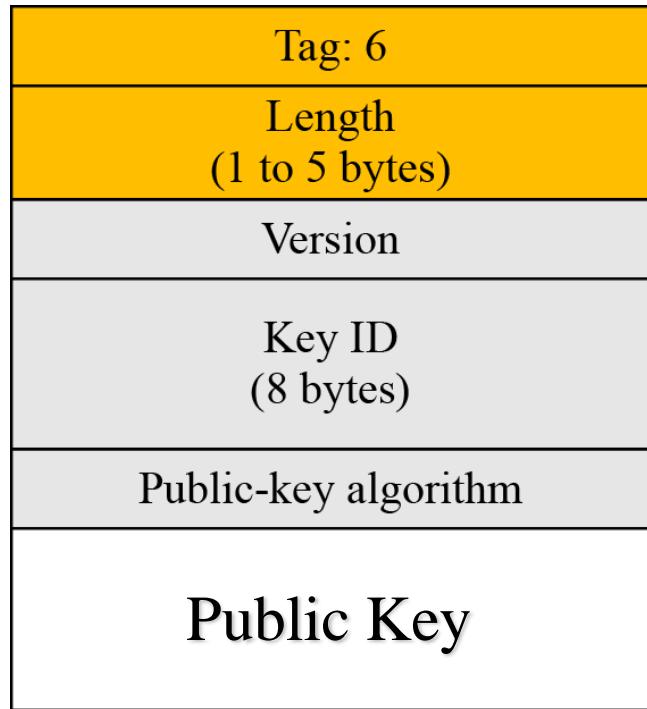


ID	Description
1	RSA (encryption or signing)
2	RSA (for encryption only)
3	RSA (for signing only)
16	ElGamal (encryption only)

ID	Description
17	DSS
18	Reserved for elliptic curve
19	Reserved for ECDSA
20	ElGamal (for encryption or signing)
21	Reserved for Diffie-Hellman
100–110	Private algorithms

16.2.6 Continued

Figure 16.18 Public-key packet

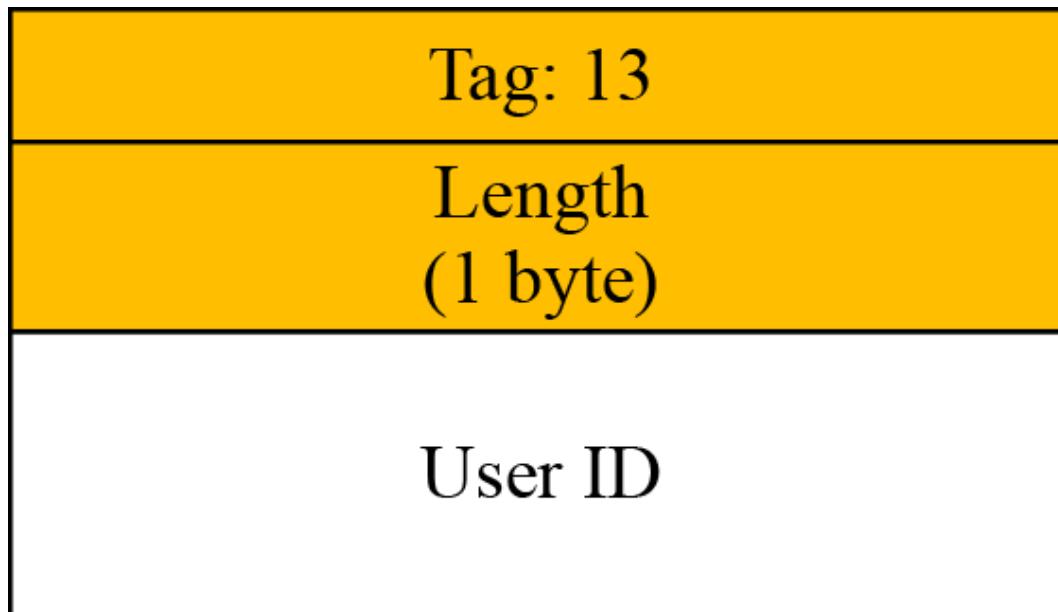


<i>ID</i>	<i>Description</i>
1	RSA (encryption or signing)
2	RSA (for encryption only)
3	RSA (for signing only)
16	ElGamal (encryption only)

<i>ID</i>	<i>Description</i>
17	DSS
18	Reserved for elliptic curve
19	Reserved for ECDSA
20	ElGamal (for encryption or signing)
21	Reserved for Diffie-Hellman
100–110	Private algorithms

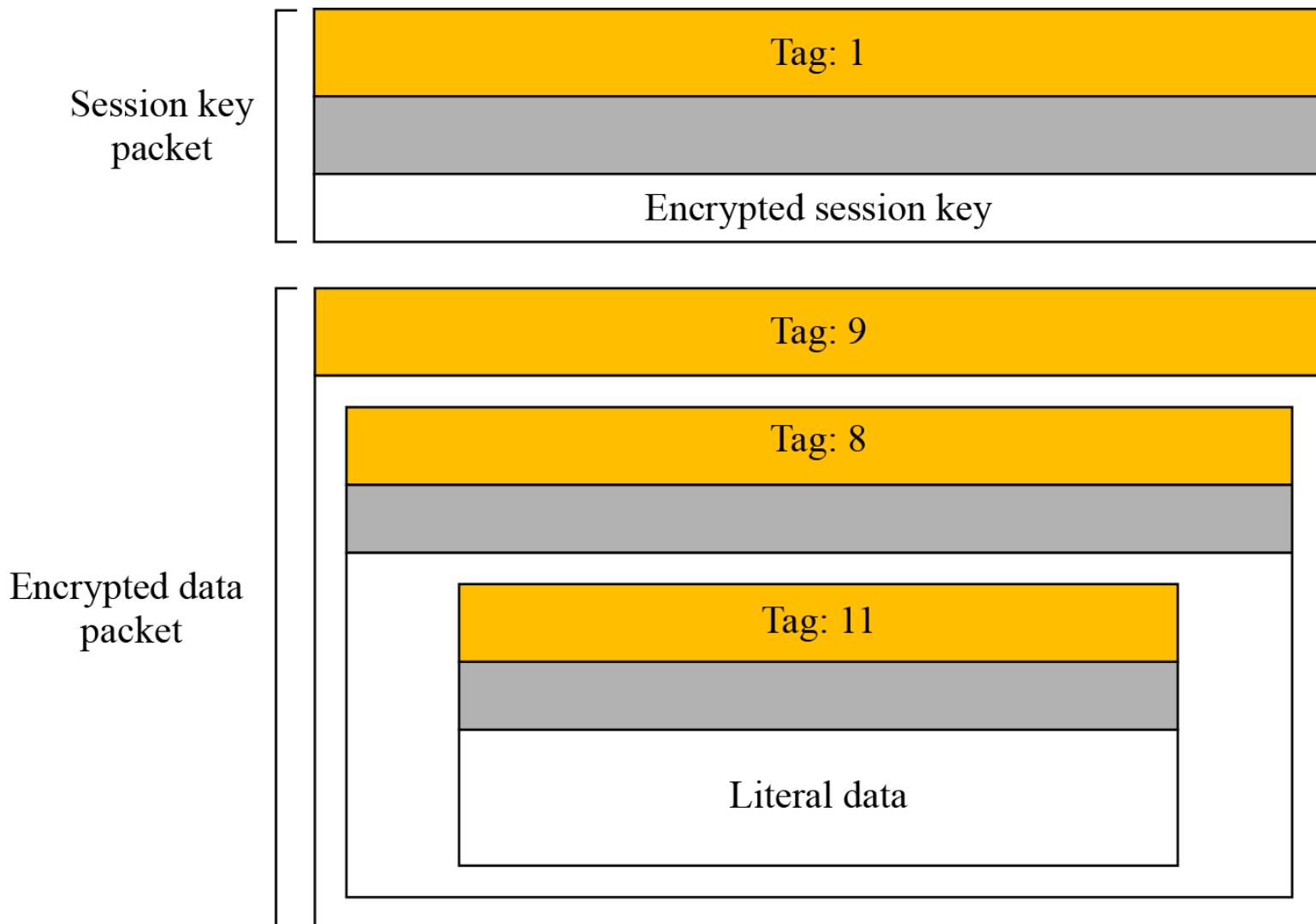
16.2.6 *Continued*

Figure 16.19 *User ID packet*



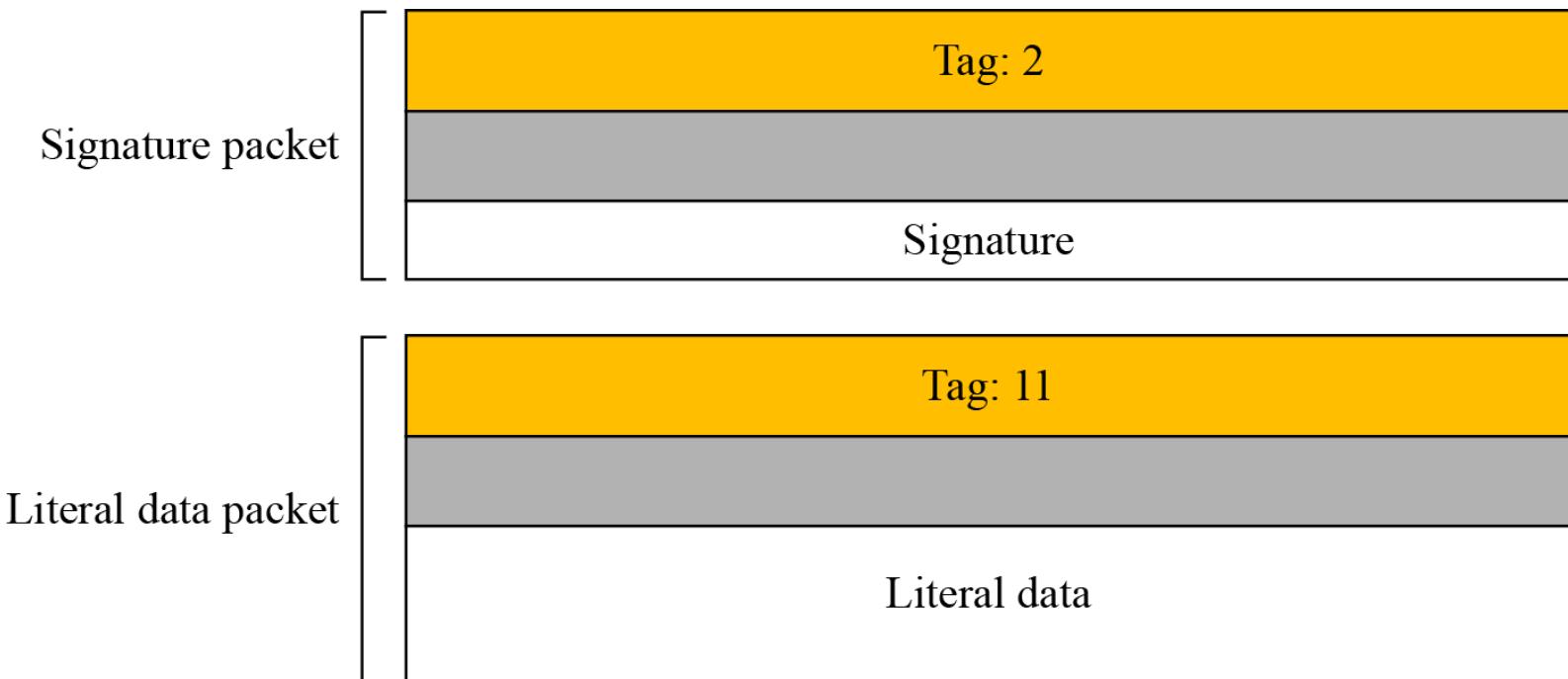
16.2.7 PGP Messages

Figure 16.20 Encrypted message



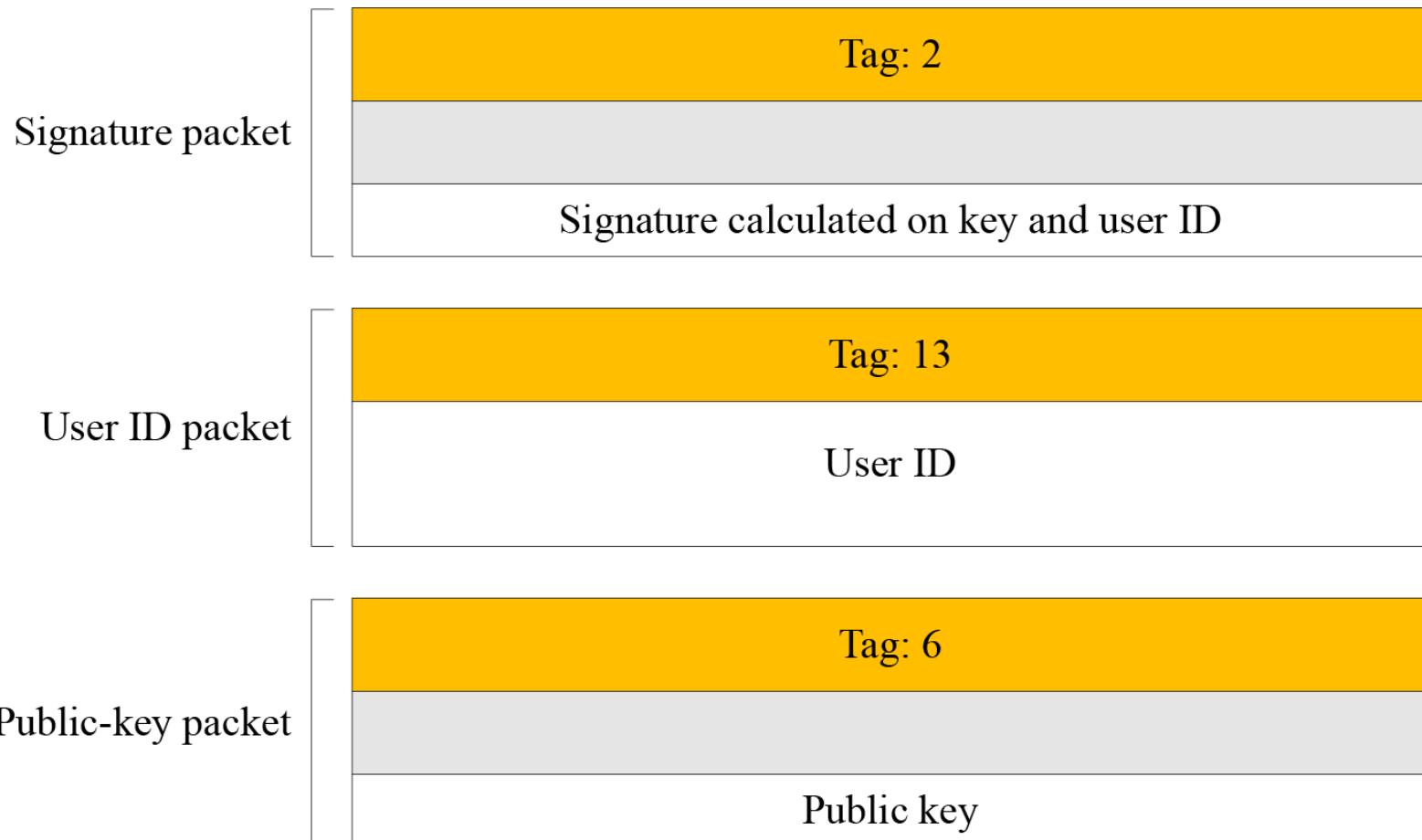
16.2.7 *Continued*

Figure 16.21 *Signed message*



16.2.7 *Continued*

Figure 16.22 Certificate message



16-3 S/MIME

Another security service designed for electronic mail is Secure/Multipurpose Internet Mail Extension (S/MIME). The protocol is an enhancement of the Multipurpose Internet Mail Extension (MIME) protocol.

Topics discussed in this section:

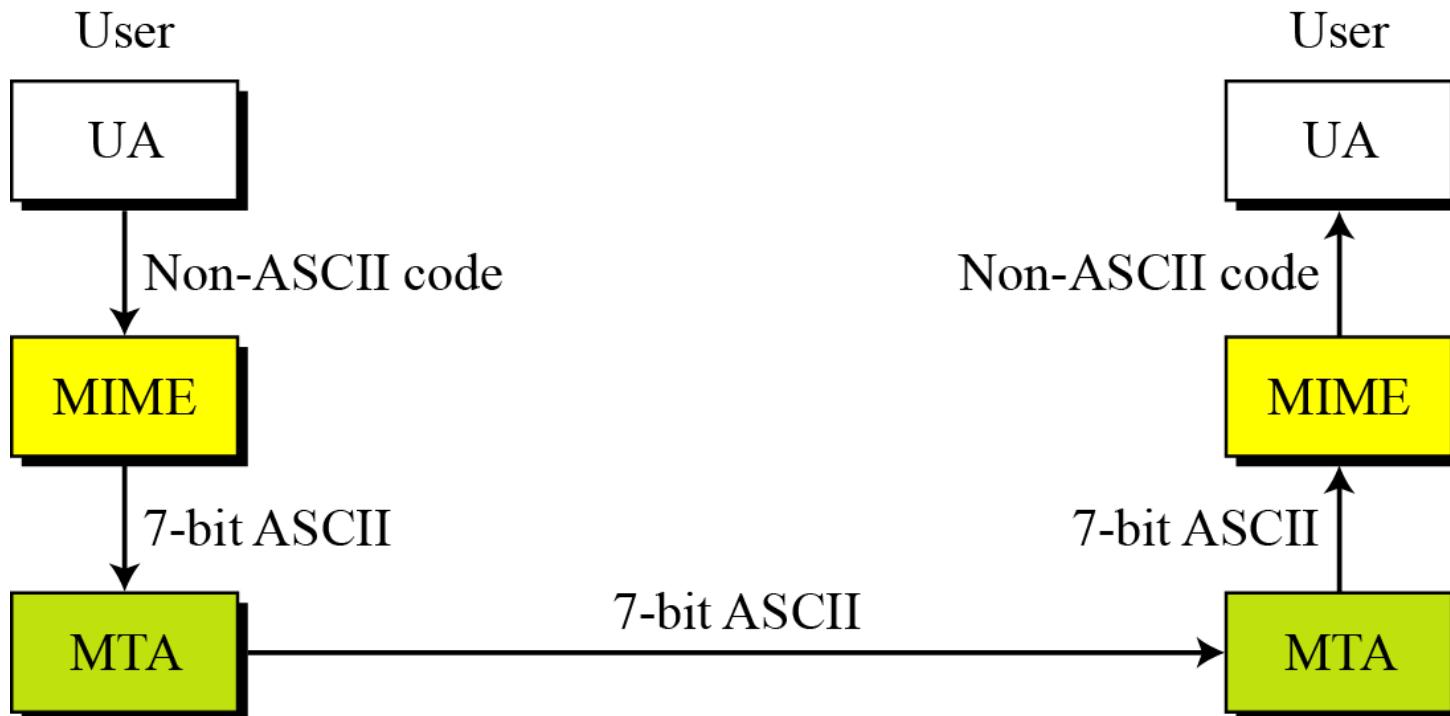
16.3.1 MIME

16.3.2 S/MIME

16.3.3 Applications of S/MIME

16.3.1 *Continued*

Figure 16.23 *MIME*



16.3.1 *Continued*

Figure 16.24 Teledesic

E-mail header	MIME headers
MIME-Version: 1.1 Content-Type: type/subtype Content-Transfer-Encoding: encoding type Content-Id: message id Content-Description: textual explanation of nontextual contents	
E-mail body	

16.3.1 *Continued*

MIME-Version

This header defines the version of MIME used. The current version is 1.1.

MIME-Version: 1.1

Content-Type

The content type and the content subtype are separated by a slash. Depending on the subtype, the header may contain other parameters.

Content-Type: <type / subtype; parameters>

16.3.1 Continued

Table 16.14 Data types and subtypes in MIME

Type	Subtype	Description
	Plain	Unformatted.
	HTML	HTML format.
Multipart	Mixed	Body contains ordered parts of different data types.
	Parallel	Same as above, but no order.
	Digest	Similar to Mixed, but the default is message/RFC822.
	Alternative	Parts are different versions of the same message.
Message	RFC822	Body is an encapsulated message.
	Partial	Body is a fragment of a bigger message.
	External-Body	Body is a reference to another message.
Image	JPEG	Image is in JPEG format.
	GIF	Image is in GIF format.
Video	MPEG	Video is in MPEG format.
Audio	Basic	Single channel encoding of voice at 8 KHz.
Application	PostScript	Adobe PostScript.
	Octet-stream	General binary data (eight-bit bytes).

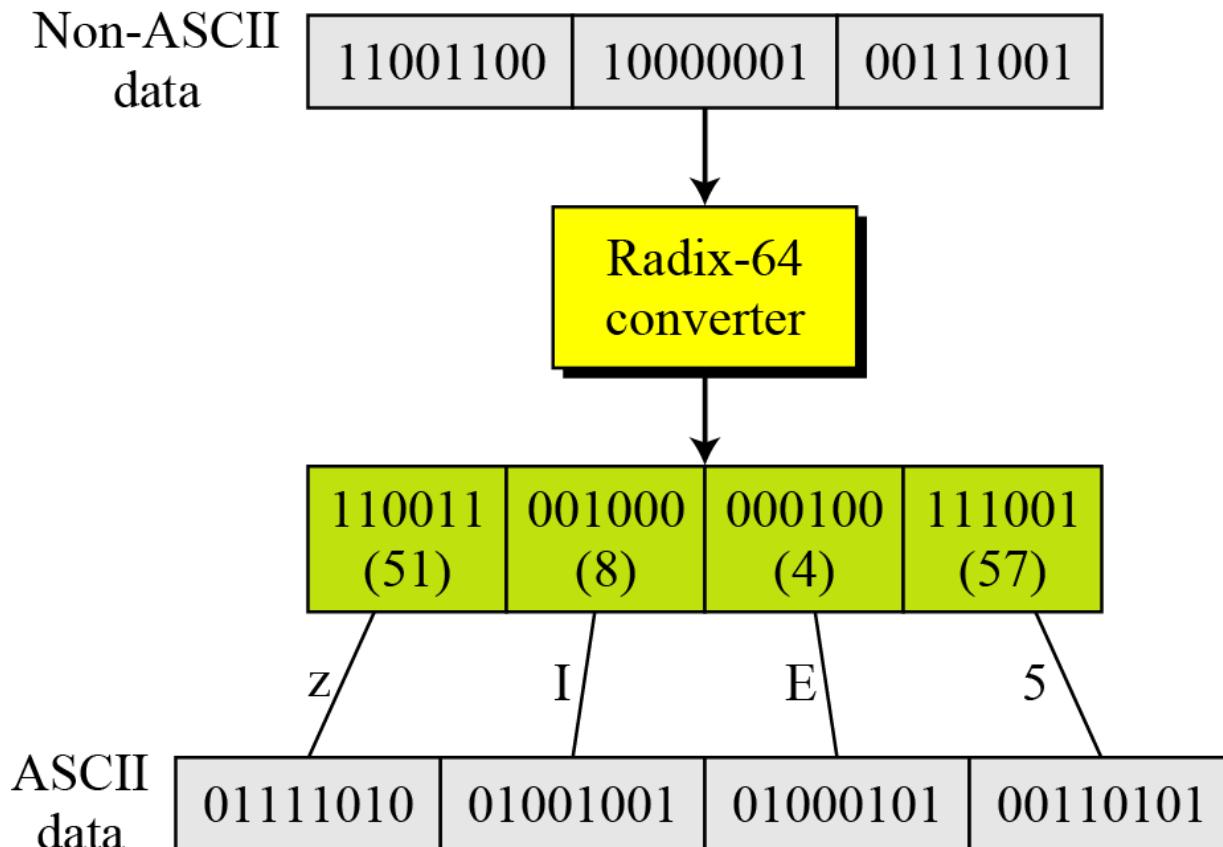
16.3.1 *Continued*

Table 16.15 *Content-transfer-encoding*

Type	Description
7bit	NVT ASCII characters and short lines.
8bit	Non-ASCII characters and short lines.
Binary	Non-ASCII characters with unlimited-length lines.
Radix-64	6-bit blocks of data are encoded into 8-bit ASCII characters using Radix-64 conversion.
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code.

16.3.1 Continued

Figure 16.25 Radix-64 conversion



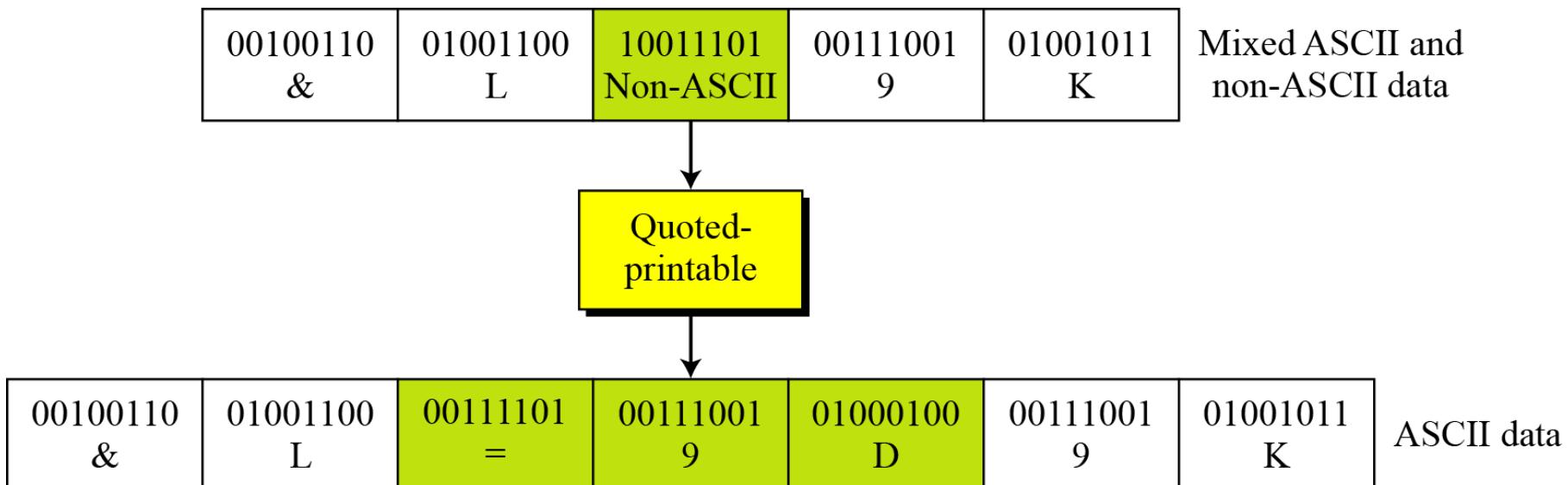
16.3.1 Continued

Table 16.16 Radix-64 encoding table

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

16.3.1 Continued

Figure 16.26 Quoted-printable



16.3.2 S/MIME

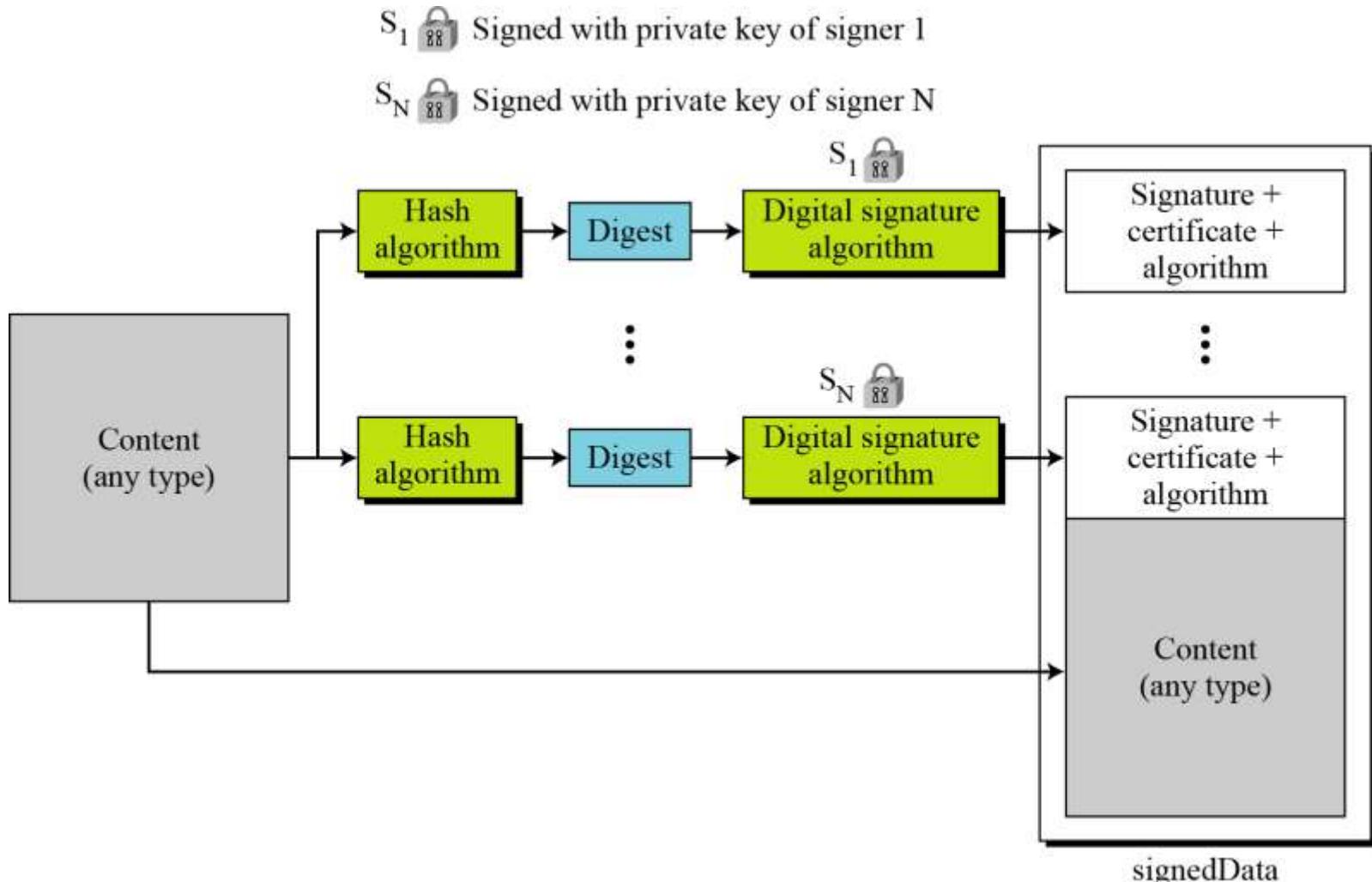
*S/MIME adds some new content types to include security services to the MIME. All of these new types include the parameter “application/pkcs7-mime,” in which “pkcs” defines “**Public Key Cryptography Specification.**”*

Cryptographic Message Syntax (CMS)

*To define how security services, such as confidentiality or integrity, can be added to MIME content types, S/MIME has defined Cryptographic Message Syntax (CMS). The syntax in each case defines the exact encoding scheme for each content type. For details, the reader is referred to **RFC 3369** and **3370**.*

16.3.2 *Continued*

Figure 16.27 *Signed-data content type*



16.3.2 Continued

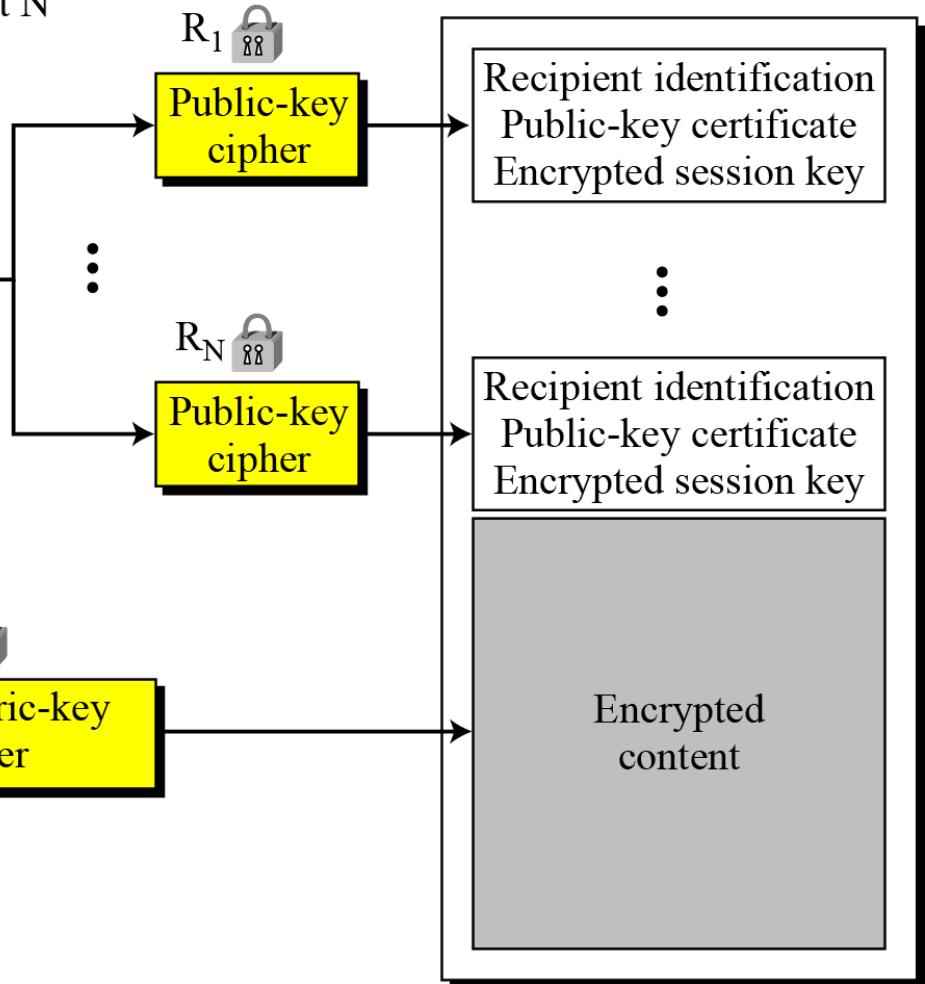
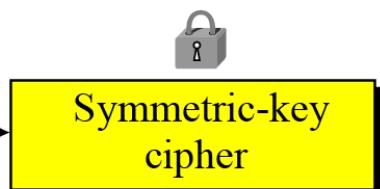
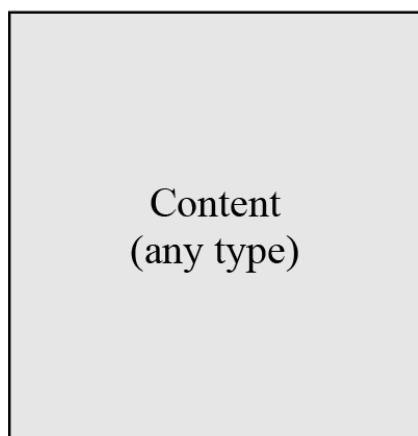
Figure 16.28 Enveloped-data content type

R_1 Encrypted with public key of recipient 1

R_N Encrypted with public key of recipient N

Encrypted with session key

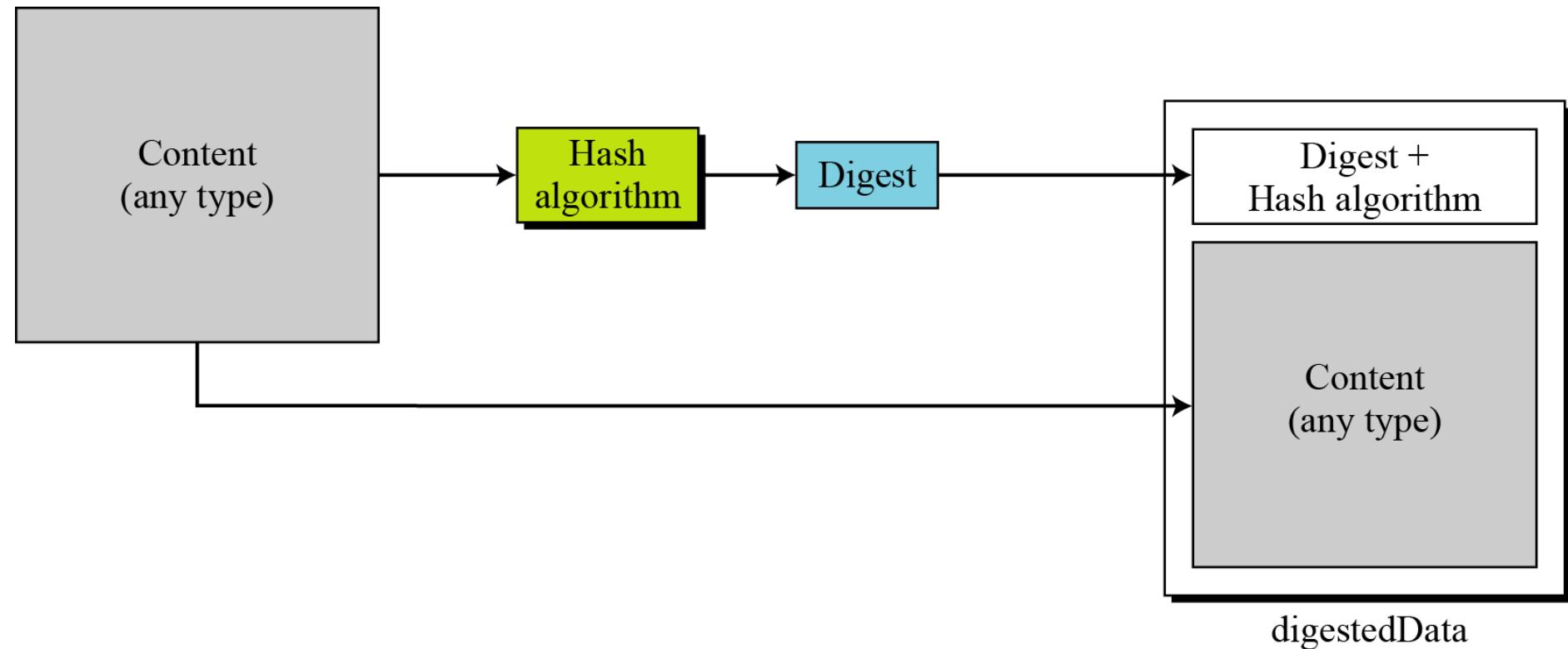
Session key created by
pseudorandom
generator



envelopedData

16.3.2 *Continued*

Figure 16.29 *Digest-data content type*

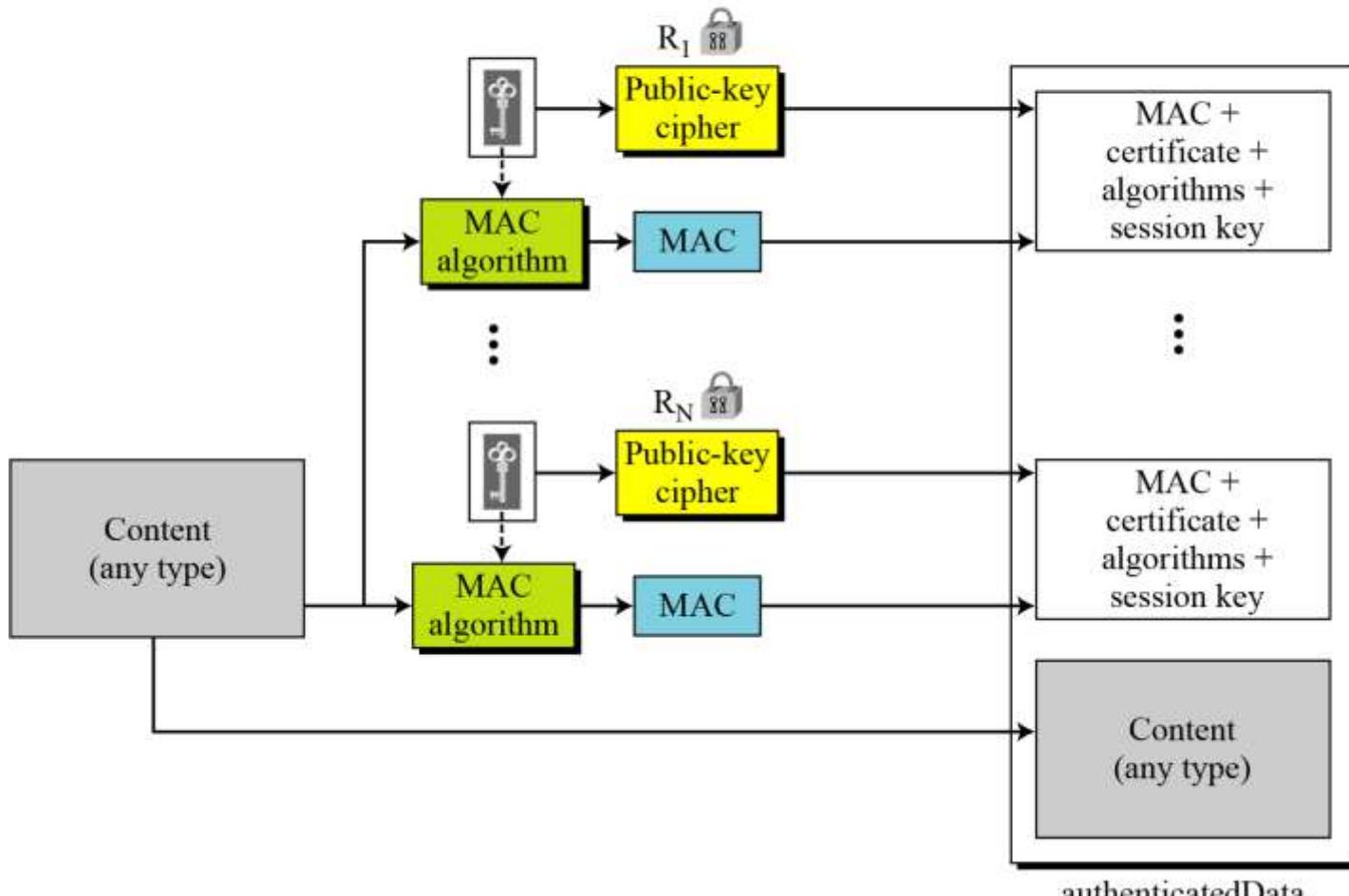


16.3.2 Continued

Figure 16.30 Authenticated-data content type

R_1 Encrypted with public key of recipient 1

R_N Encrypted with public key of recipient N



16.3.2 Continued

Cryptographic Algorithms

S/MIME defines several cryptographic algorithms. The term “**must**” means an absolute requirement; the term “**should**” means recommendation.

Table 16.17 Cryptographic algorithm for S/MIME

Algorithm	Sender must support	Receiver must support	Sender should support	Receiver should support
Content-encryption algorithm	Triple DES	Triple DES		1. AES 2. RC2/40
Session-key encryption algorithm	RSA	RSA	Diffie-Hellman	Diffie-Hellman
Hash algorithm	SHA-1	SHA-1		MD5
Digest-encryption algorithm	DSS	DSS	RSA	RSA
Message-authentication algorithm		HMAC with SHA-1		

16.3.2 Continued

Example 16.3

The following shows an example of an enveloped-data in which a small message is encrypted using triple DES.

Content-Type: application/pkcs7-mime; mime-type=enveloped-data

Content-Transfer-Encoding: Radix-64

Content-Description: attachment

name="report.txt";

cb32ut67f4bhijHU21oi87eryb0287hmnklsxFDoY8bc659GhIGfH6543mhjkdsAH23YjBnmNybmlkjhgfdyhGe23Kjk34XiuD678Es16se09jy76jHuytTMDcbnmlkjgfFdiuyu678543m0n3hG34un12P2454Hoi87e2ryb0H2MjN6KuyrlsgFDoY897fk923jljk1301XiuD6gh78EsUyT23y