Chapter 12

Pretty Good Privacy (PGP)

With the explosively growing reliance on electronic mail for every conceivable purpose, there grows a demand for authentication and confidentiality services. Two schemes stand out as approaches that enjoy widespread use: Pretty Good Privacy (PGP) and Secure/Multipurpose Internet Mail Extension (S/MIME). The latter is a security enhancement to the MIME Internet e-mail format standard, based on technology from RSA Data Security. Although both PGP and S/MIME are on an IETF standards track, it appears likely that S/MIME will emerge as the industry standard for commercial and organisational use, while PGP will remain the choice for personal e-mail security for many users. In this course we will only be looking at PGP. S/MIME is discussed in detail in the recommended text.

12.1 Background

PGP is a remarkable phenomenon. Largely the effort of a single person, Phil Zimmermann, PGP provides a confidentiality and authentication service that can be used for electronic mail and file storage applications. In essence what Zimmermann has done is the following:

1. Selected the best cryptographic mechanisms (algorithms) as building blocks.

2. Integrated these algorithms into a general purpose application that is independent of operating system and processor and that is based on a small set of easy to use commands.

3. Made the package and its source code freely available via the Internet, bulletin boards, and commercial networks such as America On Line (AOL).

4. Entered into an agreement with a company (Viacrypt, now Network Associates) to provide a fully compatible low cost commercial version of PGP.

From its beginnings about 15 years ago, PGP has grown explosively and is now very widely used. A number of reasons are sited for such growth:

1. It is available free worldwide in versions that run on many different platforms, Windows, UNIX, Mac etc. In addition the commercial version satisfies those who want vendor support.

2. It is based on algorithms that have survived extensive public review and are considered secure. Specifically, the package includes RSA, DSS and Diffie-Hellman for public-key encryption; CAST-128, IDEA, and 3DES for symmetric encryption; and SHA-1 for hash coding.

3. It has a wide range of applicability, from corporations that wish to select and enforce a standardised scheme for encrypting files and messages to individuals who wish to communicate securely with others worldwide over the Internet.

4. It was not developed by, nor is it controlled by, any government or standards organisation. For those with an instinctive distrust of “the establishment”, this makes PGP attractive. In the last few years commercial versions have become available.

5. PGP is now on an Internet standards track (RFC 3156). Nevertheless, PGP still has an aura of an anti-establishment endeavor.

12.2 Operational Description

PGP consists of the following five services:

1. Authentication

2. Confidentiality

3. Compression

4. E-mail compatibility

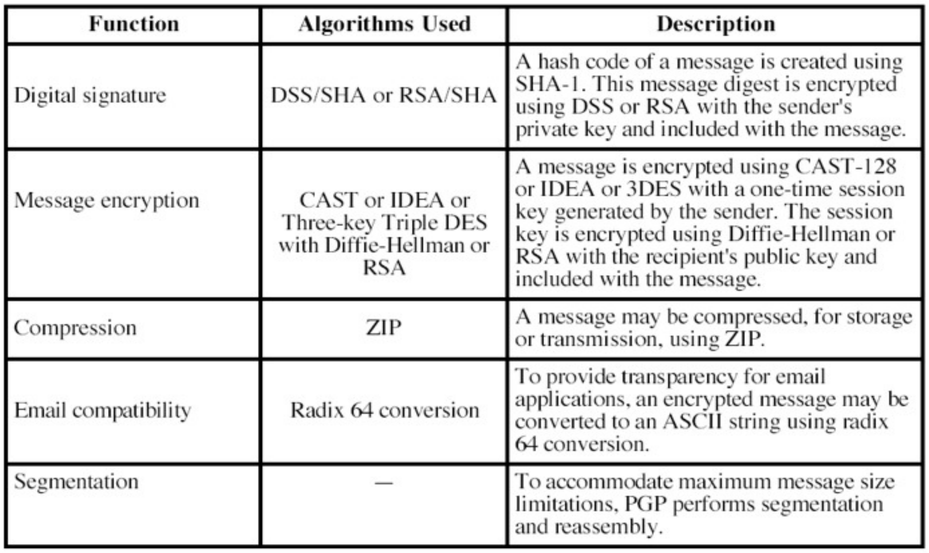
5. Segmentation

Table 12.1 shows a summary of these services and the algorithms used to implement them. Each service will be discussed in turn and then we will look at the problem of key management.

12.2.1 Authentication

Figure 12.2a illustrates the digital signature service provided by PGP. The figure is similar to ones we have looked at earlier. The hash function used is SHA-1 which creates a 160 bit message digest. EP (DP) represents public encryption (decryption) and the algorithm used can be RSA or DSS (recall that the DSS can only be used for the digital signature function and unlike RSA cannot be used for encryption or key exchange). The message may be compressed using and algorithm called **ZIP**. This is represented by “Z” in the figure.

The combination of SHA-1 and RSA provides an effective digital signature scheme. Due to the strength of RSA the recipient is assured that only the possessor of the



Function	Algorithms Used	Description
Digital signature	DSS/SHA or RSA/SHA	A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key and included with the message.
Message encryption	CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA	A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and included with the message.
Compression	ZIP	A message may be compressed, for storage or transmission, using ZIP.
Email compatibility	Radix 64 conversion	To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.
Segmentation	—	To accommodate maximum message size limitations, PGP performs segmentation and reassembly.

Figure 12.1: Summary of PGP services.

matching private key can generate the signature. Because of the strength of SHA-1 the recipient is assured that no one else could generate a new message that matches the hash code and hence, the signature of the original message.

12.2.2 Confidentiality

Another basic service provided by PGP is confidentiality which is provided by encrypting messages to be transmitted or to be stored locally as files. In both cases, the user has a choice of CAST-128, IDEA or 3DES in 64 bit cipher feedback (CFB) mode. The symmetric key is used only once and is created as a random number with the required number of bits. It is transmitted along with the message and is encrypted using the recipients public key. Figure 12.2c illustrates the sequence:

1. The sender generates a message and a random number to be used as a session key for this message only.

2. The message is encrypted using CAST-128, IDEA or 3DES with the session key.

3. The session key is encrypted with RSA (or another algorithm known as ElGamal) using the recipients public key and is prepended to the message.

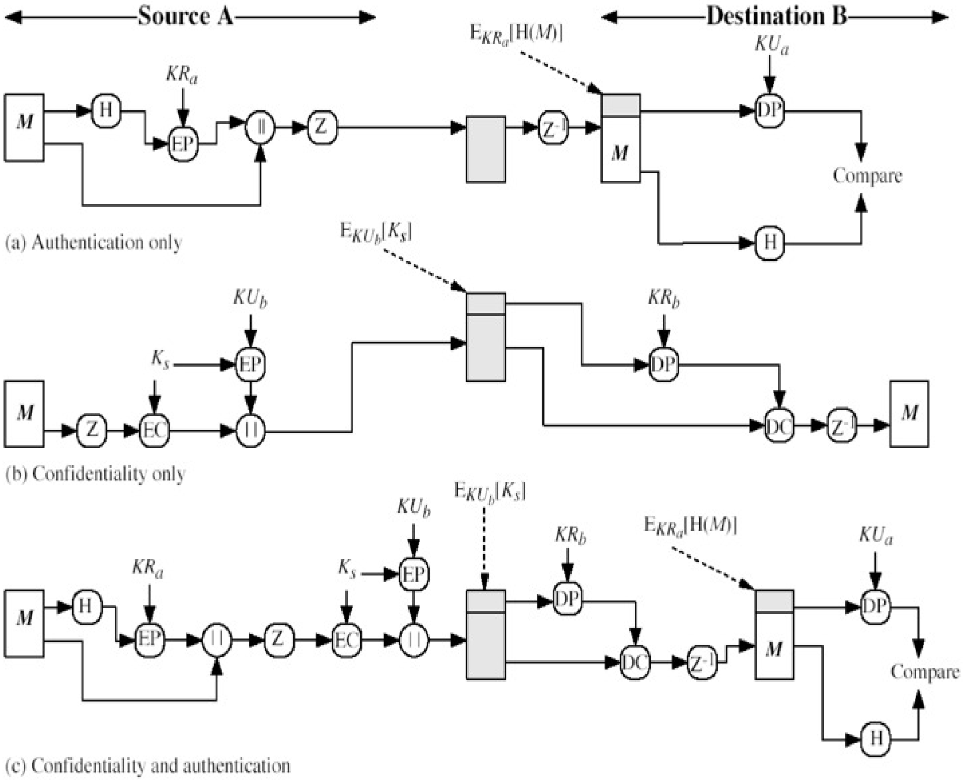


Figure 12.2: PGP cryptographic functions.

4. The receiver uses RSA with its private key to decrypt and recover the session key.

5. The session key is used to decrypt the message.

As mentioned before, public key encryption is a lot more computationally intensive than symmetric encryption. For this reason both forms are used as public key encryption solves the key distribution problem. However as will be noticed, the message itself (which is the largest part of the transmission) is encrypted using symmetric key cryptography whereas only the key is encrypted using the public key algorithm.

12.2.3 Confidentiality and Authentication

As figure 12.2c illustrates, both services may be used for the same message. First, a signature is generated for the plaintext message and prepended to the message. Then the plaintext message plus signature is encrypted using CAST-128 (or IDEA or 3DES), and the session key is encrypted using RSA (or ElGamal). This sequence is preferable to the opposite: encrypting the message and then generating a signature of the encrypted message. It is generally more convenient to store a signature with a plaintext version of a message. Furthermore, for purposes of third party verification, if the signature is performed first, a third party need not be concerned with the symmetric key when verifying the signature.

12.2.4 Compression

As a default, PGP compresses the message after applying the signature but before encryption. This has the benefit of saving space both for e-mail transmission and for file storage.

The placement of the compression algorithm, indicated by Z for compression and Z^{-1} for decompression in figure 12.2 is critical:

1. The signature is generated before compression for two reasons:
   (a) It is preferable to sign an uncompressed message so it is free of the need for a compression algorithm for later verification.
   (b) Different version of PGP produce different compressed forms. Applying the hash function and signature after compression would constrain all PGP implementation to the same version of the compression algorithm.
2. Message encryption is applied after compression to strengthen cryptographic security. Because the compressed message has less redundancy than the original plaintext, cryptanalysis is more difficult.

The compression algorithm used is called **ZIP** which is described in the recommended text.

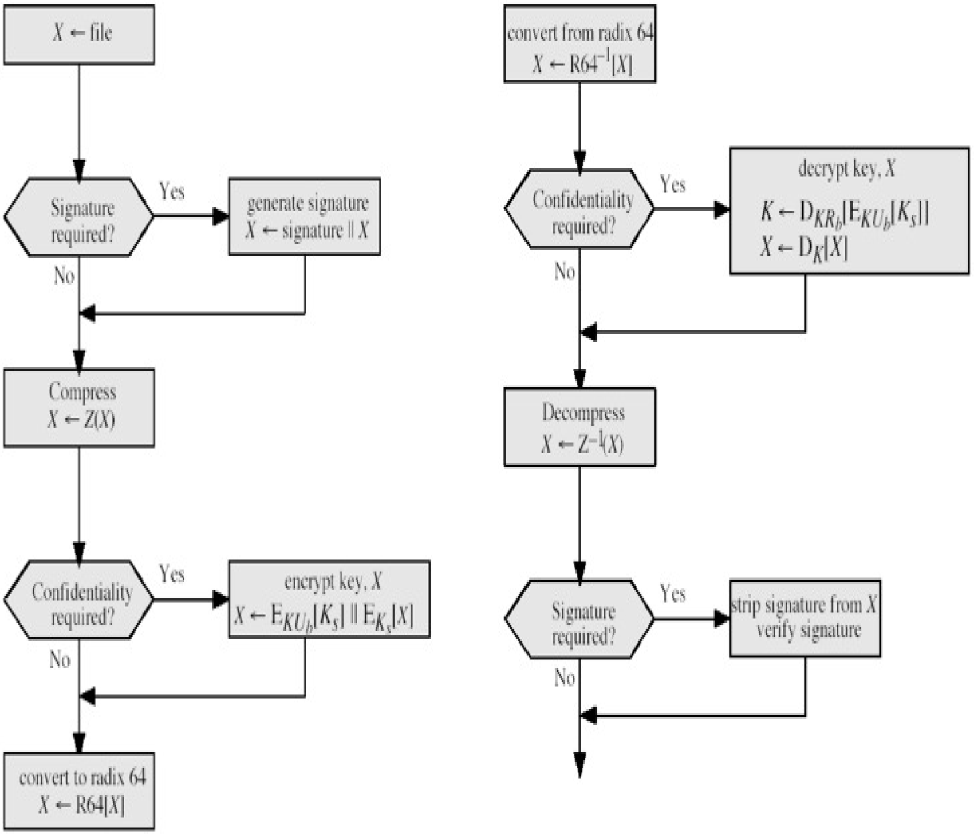
12.2.5 E-mail compatibility

Many electronic mail systems only permit the use of blocks consisting of ASCII text. When PGP is used, at least part of the block to be transmitted is encrypted. This basically produces a sequence of arbitrary binary words which some mail systems won't accept. To accommodate this restriction PGP uses and algorithm known as **radix64** which maps 6 bits of a binary data into and 8 bit ASCII character. Unfortunately this expands the message by 33% however, with the compression algorithm the overall compression will be about one third (in general).

12.2.6 Segmentation

E-mail facilities are often restricted to a maximum message length. For example, many of the facilities accessible throughout the Internet impose a maximum length of 50,000 octets. Any message longer than that must be broken up into smaller segments, each of which is mailed separately.

To accommodate this restriction, PGP automatically subdivides a message that is too large into segments that are small enough to sent via e-mail. The segmentation is done after all the other processing, including the radix-64 conversion. Thus the session key component and signature component appear only once, at the beginning of the first segment. At the receiving end, PGP must strip off all e-mail headers and reassemble the entire original block before performing the steps illustrated in figure 12.3.



(a) Generic Transmission Diagram (from A)

(b) Generic Reception Diagram (to B)

Figure 12.3: Transmission and Reception of PGP messages.

12.3 Cryptographic Keys and Key Rings

PGP makes use of four types of keys:

1. One-time session symmetric keys
2. Public keys
3. Private keys
4. Passphrase based symmetric keys

Three separate requirements can be identified with respect to these keys:

1. A means of generating unpredictable session keys is needed
2. We would like to allow a user to have multiple public-key/private-key pairs. As a result there is not a one-to-one correspondence between users and their public keys. Thus, some means is needed for identifying particular keys.
3. Each PGP entity must maintain a file of its own public/private key pairs as well as a file of public keys of correspondents.

12.3.1 Session key generation

Each session key is associated with a single message and is used only for the purpose of encryption and decrypting that message. Recall that message encryption/decryption is done with a symmetric encryption algorithm. Assuming it is a 128 bit key that is required, the random 128 bit numbers are generated using CAST-128. The input to the random number generator consists of as 128-bit key (this is a random number using the keystroke input from the user) and two 64-bit blocks that are treated as plaintext to be encrypted. Using CFB mode two 64-bit cipher text blocks are produced and concatenated to form the 128 bit session key. The algorithm that is used is based on the one specified in ANSI X12.17.

12.3.2 Key Identifiers

As mentioned it is possible to have more than one public/private key pair per user. Each one therefore needs an ID of some kind. The key ID associated with each public key consists of its least significant 64 bits. That is, the key ID of public key KU_a is ($\text{KU}_a \bmod 2^{64}$). This is a sufficient length that the probability of duplicate key IDs is very small. A key ID is also used for the PGP digital signature as the sender may use one of a number of private keys to encrypt the message digest and the recipient must know which one was used.

With the key ID concept understood, a more detailed look at the format of a transmitted message is shown in figure 12.4.

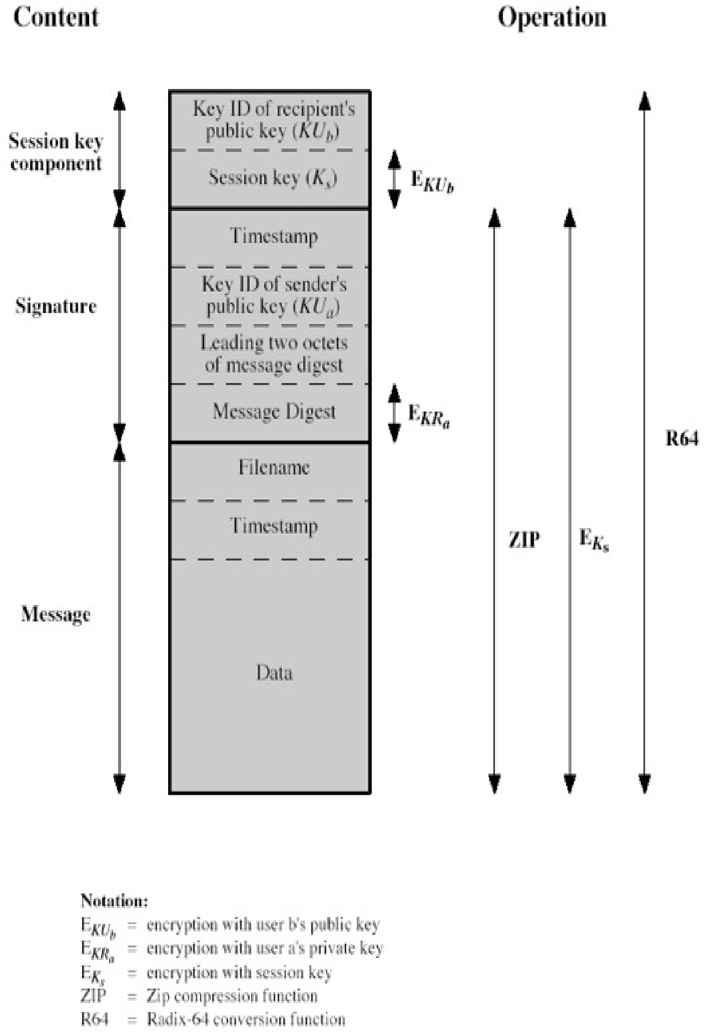
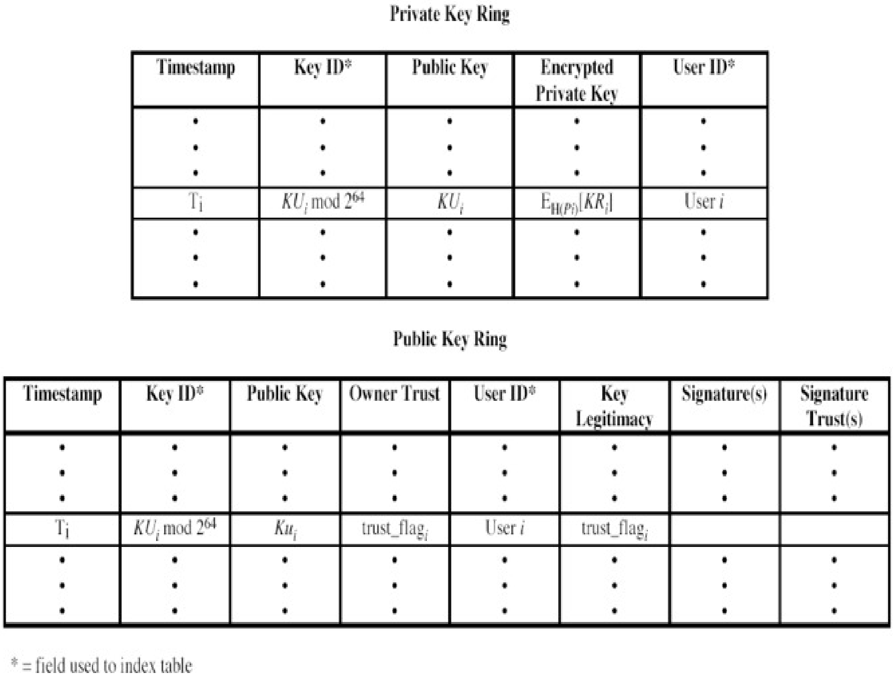

Content
Operation
Session key component
Signature
Message
Key ID of recipient's public key (KU_b)
Session key (K_s)
Timestamp
Key ID of sender's public key (KU_a)
Leading two octets of message digest
Message Digest
Filename
Timestamp
Data
E_{KU_b}
E_{KR_a}
ZIP
E_{K_s}
R64
Notation:
E_{KU_b} = encryption with user b's public key
E_{KR_a} = encryption with user a's private key
E_{K_s} = encryption with session key
ZIP = Zip compression function
R64 = Radix-64 conversion function

Figure 12.4: General format of PGP message (from A to B).

12.3.3 Key Rings

Key IDs are critical to the operation of PGP. From figure 12.4 it can be seen that two key IDs are included in any PGP message that provides both confidentiality and authentication. These keys need to be stored and organised in a systematic way for efficient and effective use by all parties. The scheme used in PGP is to provide a pair of data structures at each node, one to store the public/private key pairs owned by that node and one to store the public keys of other users known at this node. These data structures are referred to, respectively as the private-key ring and the public key ring. They can be seen in figure 12.5.

We can view the ring as a table where each row represents one of the public/private key pairs owned by this user. Each row contains the following:



Private Key Ring

Timestamp	Key ID*	Public Key	Encrypted Private Key	User ID*
• • •	• • •	• • •	• • •	• • •
T_i	KU_i mod 2^{64}	KU_i	$E_{H(Pi)}[KR_i]$	User i
• • •	• • •	• • •	• • •	• • •

Public Key Ring

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

* = field used to index table

Figure 12.5: General structure of private and public-key rings.

- **Timestamp:** The date/time when this key pair was generated.
- **Key ID:** The least significant 64 bits of the public key for this entry.
- **Public Key:** The public-key portion of the pair.
- **Private key:** The private-key portion of the pair.
- **User ID:** Typically a user's e-mail address.

The private key ring can be indexed by either User ID, key ID or both. However for security the value of the key is not stored in the key ring but and encrypted version of it which requires a pass phrase to decrypt. As with all passphrase schemes, the security of this method depends on the strength of the passphrase. Figure 12.6 show PGP message generation (without compression or radix64 conversion) using all the terms we have met (reception is similar).

A freeware PGP version can be downloaded here (I recommend downloading it and trying it out):

http://www.pgpi.org/products/pgp/versions/freeware/

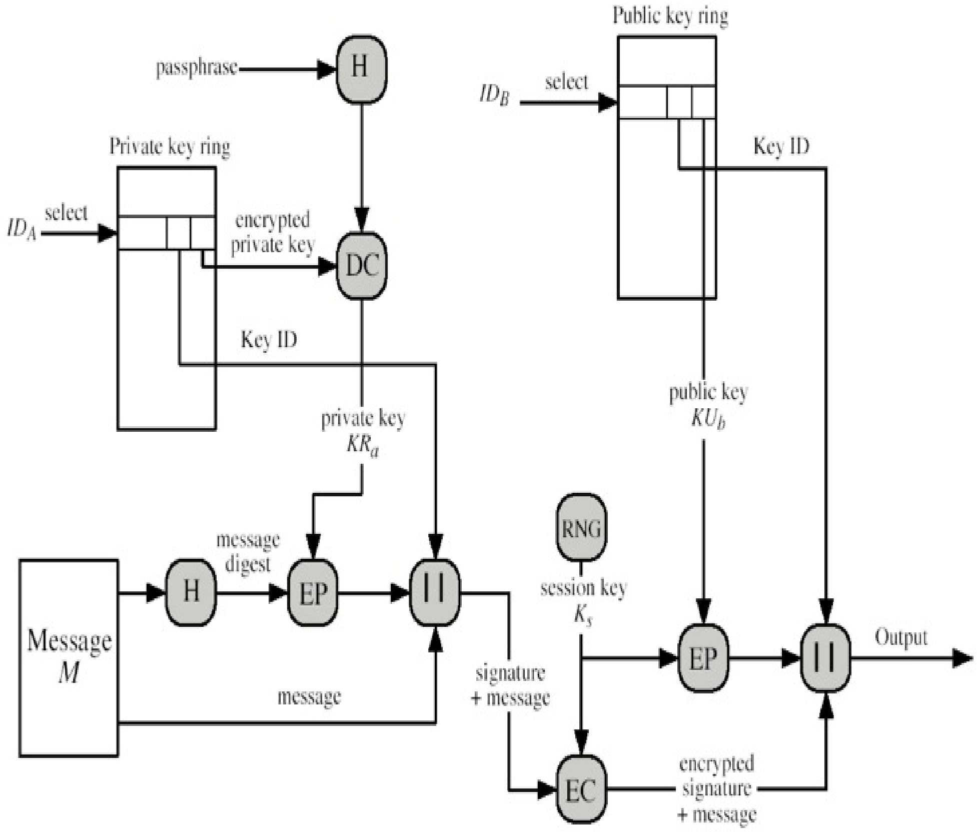


Figure 12.6: PGP Message generation (from User A to User B; no compression or radix64 conversion).