Cryptography and Security

Cunsheng DING

HKUST, Hong Kong

Version 3

Lecture 16: Electronic Mail Security

Outline of this Lecture

1. Email security issues.
2. Detailed introduction of PGP.

About Electronic Mail

1. In virtually all distributed environments, electronic mail is one of the most heavily-used and network-based applications.
2. It is also a distributed application that is widely used across all architectures (processors) and platforms (PC, UNIX, Macintosh, etc).

Consequence: Since many people may exchange sensitive email messages, there is a demand for authentication and confidentiality services.

Developing a System for Electronic Mail Security

Having learned the basics of ciphers, digital signature, and authentication, you are asked to design a system to support the following for electronic email communication:

1. message confidentiality;
2. signer nonrepudiation; and
3. message authentication (message integrity).

Question: How do you design your system?

Developing a System for Electronic Mail Security

Answer: You need to carry out the following:

1. Select the best available cryptographic algorithms as building blocks; and
2. integrate 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.

This is how PGP and S/MIME were developed.

PGP: Pretty Good Privacy
S/MIME: Secure/Multipurpose Internet Mail Extension

PGP: Pretty Good Privacy

1. It is a program for email communication security.
2. Phil Zimmermann started writing PGP in the mid 1980s and finished the first version in 1991.
3. It is available free worldwide in versions than runs on a variety of platforms, including DOS/Windows, UNIX, Macintosh, and many more.
4. It is based on cryptographic algorithms that have survived extensive public review.
5. It has a wide range of applicability: within corporations and for individuals within themselves.

A Summary of PGP Services

1. Signer nonrepudiation, data integrity, and data origin authentication (Digital signature using DSS/SHA or RSA/SHA plus timestamp).
2. Message confidentiality (encryption with CAST or 3DES or AES, and session key encryption with ElGamal or RSA).
3. Compression (using ZIP) – A message may be compressed, for storage or transmission.
4. Email compatibility (using radix-64 conversion) and email segmentation

   These two services will not be covered in this course.

Security Services that can be provided in PGP

- Only signer nonrepudiation, data origin authentication and data integrity:

$$A \to m||D_{k_d^{(A)}}[h(m)] \to B,$$

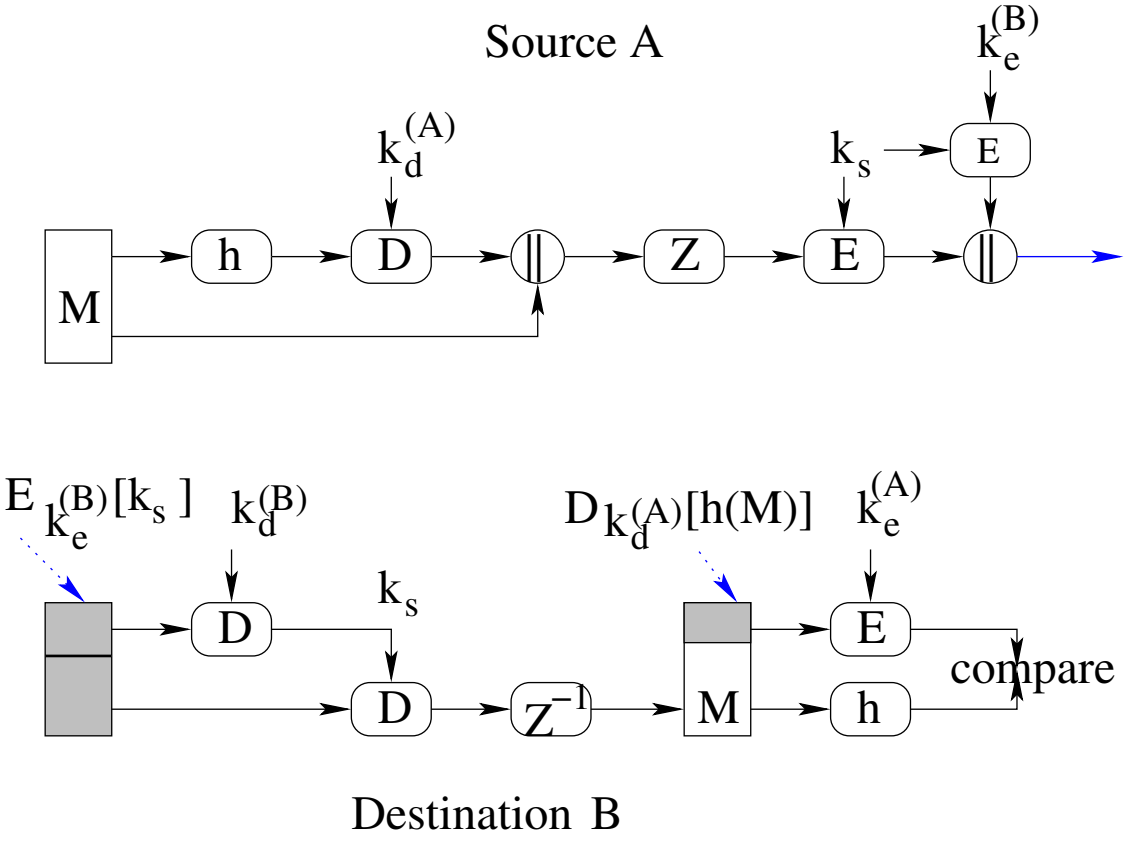
  where signature is generated by DSS/SHA-2 or RSA/SHA-2.

- Only data confidentiality:

$$A \to E_{k_s}(Z(m))||E_{k_e^{(B)}}(k_s) \to B,$$

  where
  - Z = ZIP algorithm, symmetric encryption uses CAST-128 or 3DES or AES. k_s the session key,
  - the encryption of k_s uses RSA or ElGamal.

Authentication, Confidentiality, Nonrepudiation in PGP



DSS/SHA-2 or RSA/SHA-2, Z = ZIP algorithm, RSA or ElGamal, CAST-128 or 3DES or AES. k_s the session key.

Compression in PGP

Why compression? Save space both for email transmission and for file storage, and for enhancing security.

Placement of compression: After applying the signature, but before encryption. Z indicates compression and Z^{-1} decompression.

Why should Z be before encryption? Compression reduces the redundancy of messages and makes cryptanalysis more difficult!

Why signature before compression? A data compression algorithm can be used to compress a document at different ratio. If signature is before compression, the signature verification is independent of compression ratio.

Comment: It is interesting to note that finding the right placement of a building block is quite important for the whole system!

Remark: Details of ZIP are available on the Internet.

Keys Used in PGP

1. One-time session keys for encrypting email messages.
2. Public and private keys for signature generation and verification as well as for session key distribution.
3. Passphrase-based keys for encrypting users' private keys.

Key Requirements in PGP

- A means of generating session keys is needed.
- A user is allowed to have multiple public/private key pairs.

  (A user may wish to have multiple key pairs at a given time to interact with different groups of correspondents or simply to enhance security by limiting the amount of material encrypted with any key.)

  Hence there is not a one-to-one correspondence between users and their public keys. This policy causes some problems (see key identifier later).
- Each PGP entity must maintain a file of its own public/private key pairs as well as a file of public keys of correspondents.
  - See private-key ring and public-key ring later.

Session Key Generation

Definition: Each session key is associated with a single message and is used only for encrypting and decrypting that message using a symmetric cipher.

Symmetric ciphers: CAST-128, 3DES (168-bit key), AES.

Session Key Generation: Using CAST-128 (block size 64) as example

$$k_s = \text{CAST-128}_{CFB}(k, N),$$

where k is a 128-bit key for CAST-128, and $N = N_2||N_1$ consists of two 64-bit blocks. All three (k, N_1, N_2) are based on a keystroke input from the user. N is encrypted using CAST-128 in CFB mode.

Remark: We skip details of the generation of k and N with a keystroke input from the user.

Key Identifiers (1)

Problem: Recall that A sends $E_{k_s}[x]||E_{k_e^{(B)}}[k_s]$ to B if encryption is needed. But B may have more than one private/public key pairs. How could B know which of his public key was used in the digital envelop $E_{k_e^{(B)}}[k_s]$ by A?

Solution 1: Transmit the public key $k_e^{(B)}$ together with that message as $E_{k_s}[x]||E_{k_e^{(B)}}[k_s]||k_e^{(B)}$. Then B could check that it is indeed one of his public keys.

Disadvantage: But it is a waste of resource, as a public key could have hundreds of digits in length.

Key Identifiers (2)

Problem: Recall that A sends $E_{k_s}[x]||E_{k_e^{(B)}}[k_s]$ to B if encryption is needed. But B could have more than one private/public key pairs. How could B know which of his public key was used in the digital envelop $E_{k_e^{(B)}}[k_s]$ by A?

Solution adopted in PGP: Transmit the text $E_{k_s}[x]||E_{k_e^{(B)}}[k_s]||\mathrm{ID}(k_e^{(B)})$, where the ID of the public key $k_e^{(B)}$ is defined to be $k_e^{(B)} \bmod 2^{64}$.

Comments: Hence, with very high probability the IDs of users' public keys are different.

Is key ID needed for PGP signature? Yes. Key ID is also included in the components of PGP signature, so that B can use the correct public key of A for signature verification.

Key Rings

Observation: Two key IDs $ID(k_e^{(A)})$ and $ID(k_e^{(B)})$ are included in any PGP message that provides both confidentiality and authentication.

Question: How do we store and organize key pairs and key IDs in a systematic way for efficient and effective use by all parties?

Scheme used in PGP: It provides a pair of data structure 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.

The data structures are referred to, respectively, as the **private-key ring** and **public-key ring.**

Private Key Ring: 1-Row 1-Key Pair

Timestamp	key ID*	public key	encrypted private key	user ID*
⋮	⋮	⋮	⋮	⋮
T_i	$k_e^{(i)} \bmod 2^{64}$	$k_e^{(i)}$	$E_{h(P_i)}[k_d^{(i)}]$	user i
⋮	⋮	⋮	⋮	⋮

The private-key ring can be indexed by either User ID or Key ID.

The private-key ring is stored only on the machine of the user that created and owns the key pair, and is accessible only to that user.

The user selects a passphrase P_i. The private key is encrypted using part of $h(P_i)$ as the key.

Private Key Ring

More about the passphrase:

1. When a user accesses his/her private key, he/she must supply the passphrase. PGP will retrieve the encrypted private key.
2. When the system generates a new public/private key pair, it also asks the user for the passphrase so that the private key can be encrypted and then stored in the user's private key ring.

Hence the security of the system depends on that of the passphrase!

Public Key Ring: 1-Row per Public Key

Timestamp	key ID*	public key	user ID*
⋮	⋮	⋮	⋮
T_i	$k_e^{(i)} \bmod 2^{64}$	$k_e^{(i)}$	user i
⋮	⋮	⋮	⋮

Definition: It is used to store public keys of other users that are known to this user.

Remark: The public-key ring can be indexed by either User ID or Key ID.

Public-Key Management in PGP

Comment: PGP is intended for use in a variety of formal and informal environments, no rigid public-key management scheme is set up!

Comment: One should update and verify the correctness of the information in his/her public key ring.

Format of Transmitted Messages in PGP

Signature Component (1):

$$D_{k_d^{(A)}}[\mathrm{h}(M||T_1)]||L||ID(k_e^{(A)})||T_2$$

Here M is the message data excluding the header fields (file name and timestamp T_1 of the message M), h is the hash function.

Timestamp: T_1, the time at which the email message was created.

Timestamp: T_2, the time at which the signature was made.

Message digest: $\mathrm{h}(M||T_1)$

Leading two octets of message digest: L

Key ID of sender's public key: $ID(k_e^{(A)})$

Signature Component (2)

Roles of the building blocks:

Message digest: $\mathrm{h}(M||T_1)$

1. Why should T_1 be involved here?

   (Indicate the time at which the email message was created. Any modification of T_1 can be detected with high probability.)

2. Why the filename of the message component is excluded in the computation of the message digest?

   (Ensure that detached signatures are exactly the same as attached signatures prefixed to the message. Detached signatures are calculated as $\mathrm{h}(M||T_1)$ that has no involvement of the file name.)

Signature Component (3)

Roles of the building blocks:

Leading two octets of message digest: L

To enable the recipient to determine if the correct public key ($k_e^{(A)}$) was used to decrypt the message digest for authentication, by comparing this plaintext copy of the first two octets with the first two octets of the decrypted digest.

These octets also serve as a 16-bit frame-check sequence for the message, for authentication and error detection.

The timestamp T_2: Indicate the time at which the digital signature was created. It can be used for antireplay if the email message is encrypted.

Format of Transmitted Messages in PGP

Roles of the building blocks:

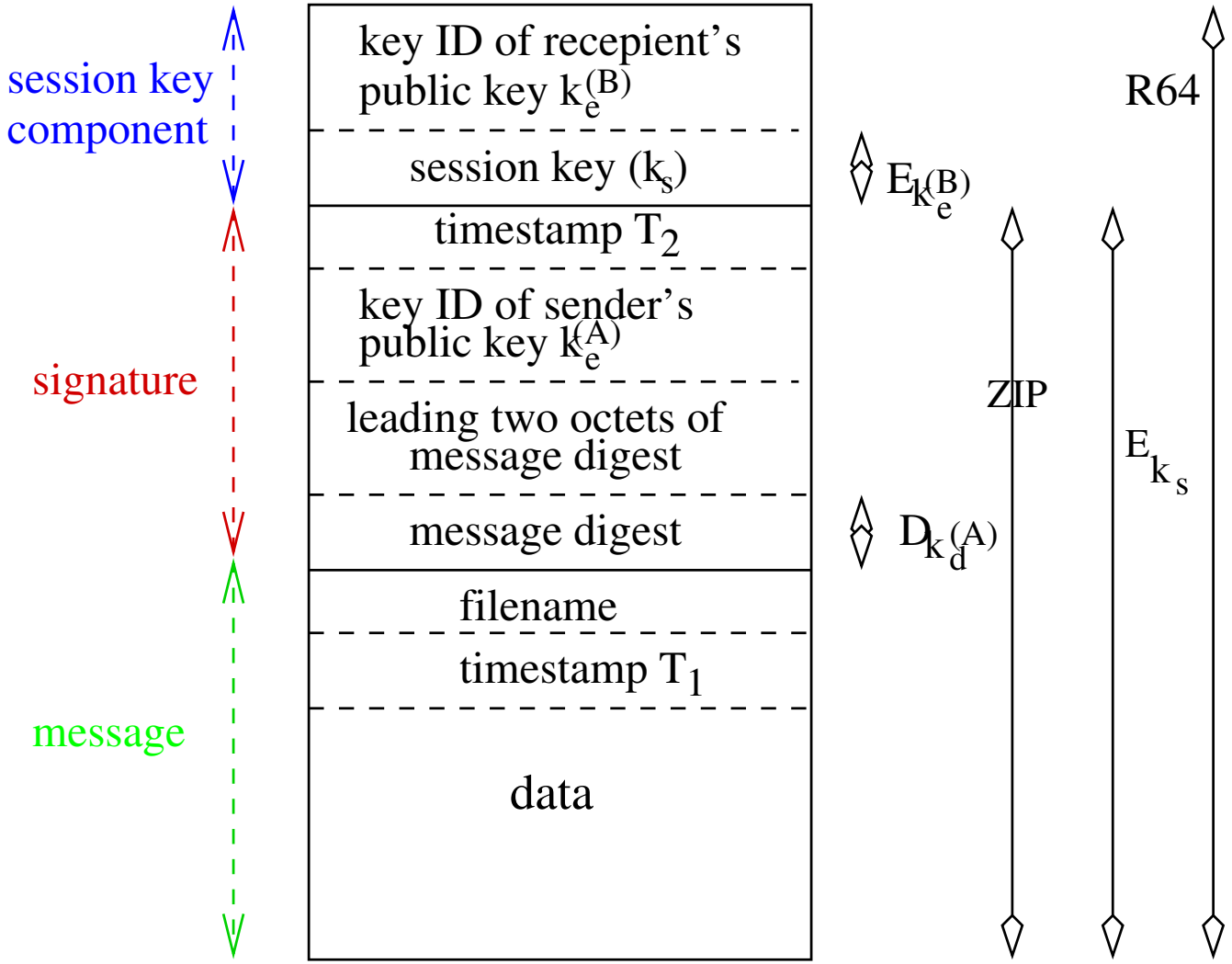
Session Key Component:

$$E_{k_e^{(B)}}[k_s]||ID\left(k_e^{(B)}\right).$$

Other operations on the components:

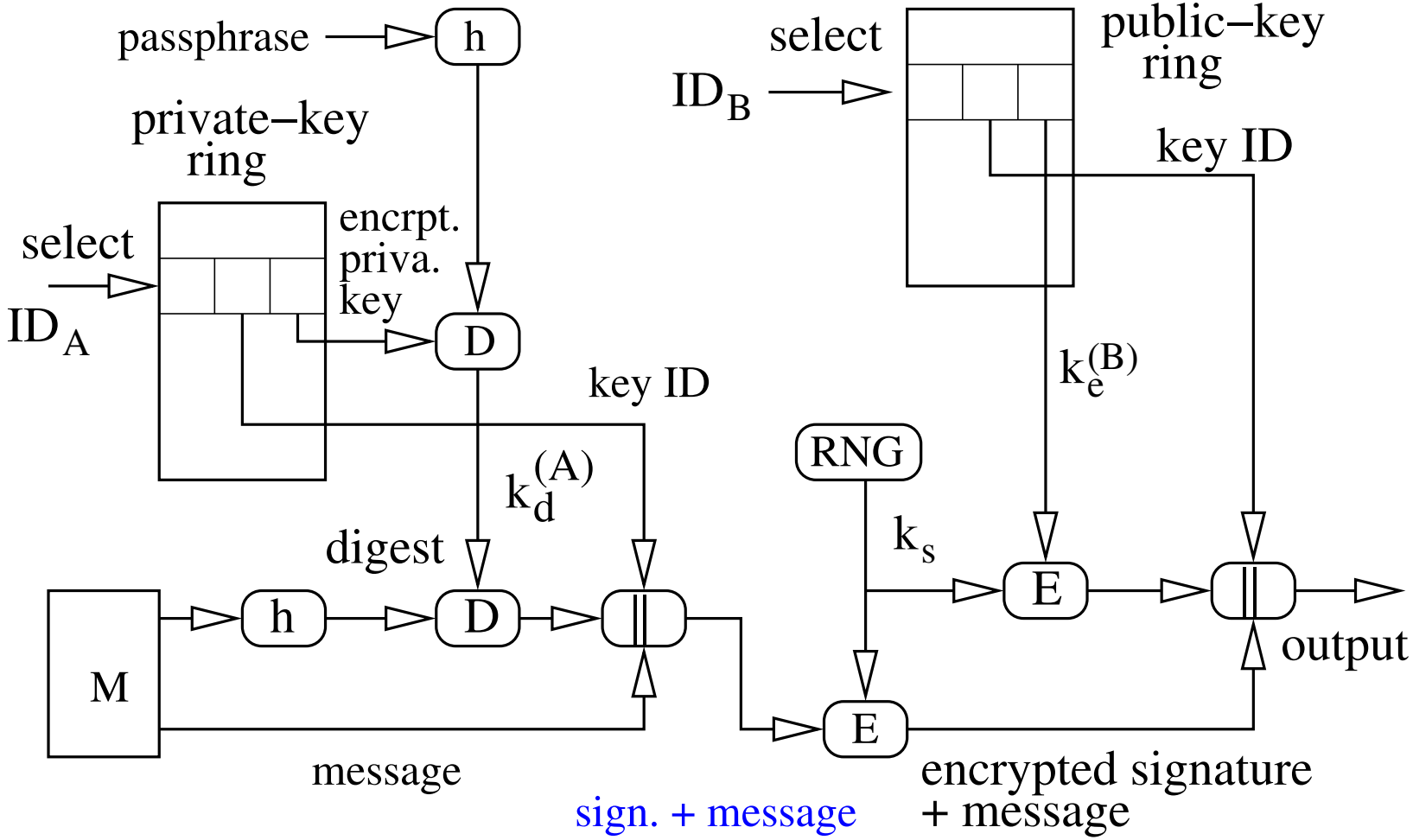
The message component and optional signature component may be compressed using ZIP and may be encrypted using a session key.

Format of Transmitted Messages in PGP: $A \mapsto B$)



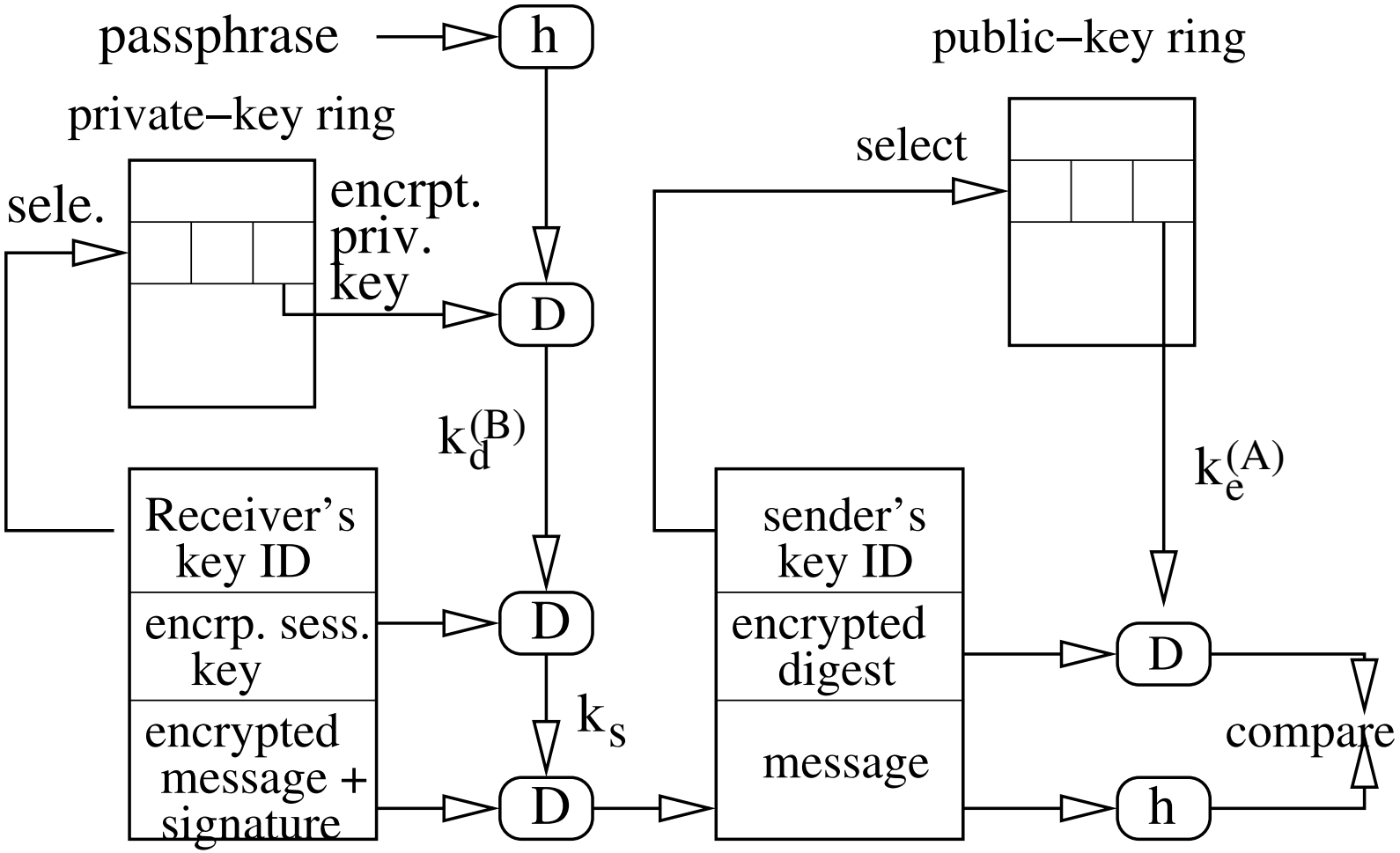
PGP Message Generation

At sender A: ZIP (compression), R64 (conversion) are omitted



PGP Message Reception

At receiver B: ZIP (compression), R64 (conversion) are omitted



Email Systems Supporting PGP

- Use PGP with Pegasus mail
- Use PGP with Simeon (ExacMail)
- Use PGP with Eudora, Outlook
- Use PGP with Herald (WING)
- Use PGP with Pine and ELM on UNIX
- **PGP is now supported in Gmail**.
  - This justifies why you should learn PGP.