Email Security

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

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

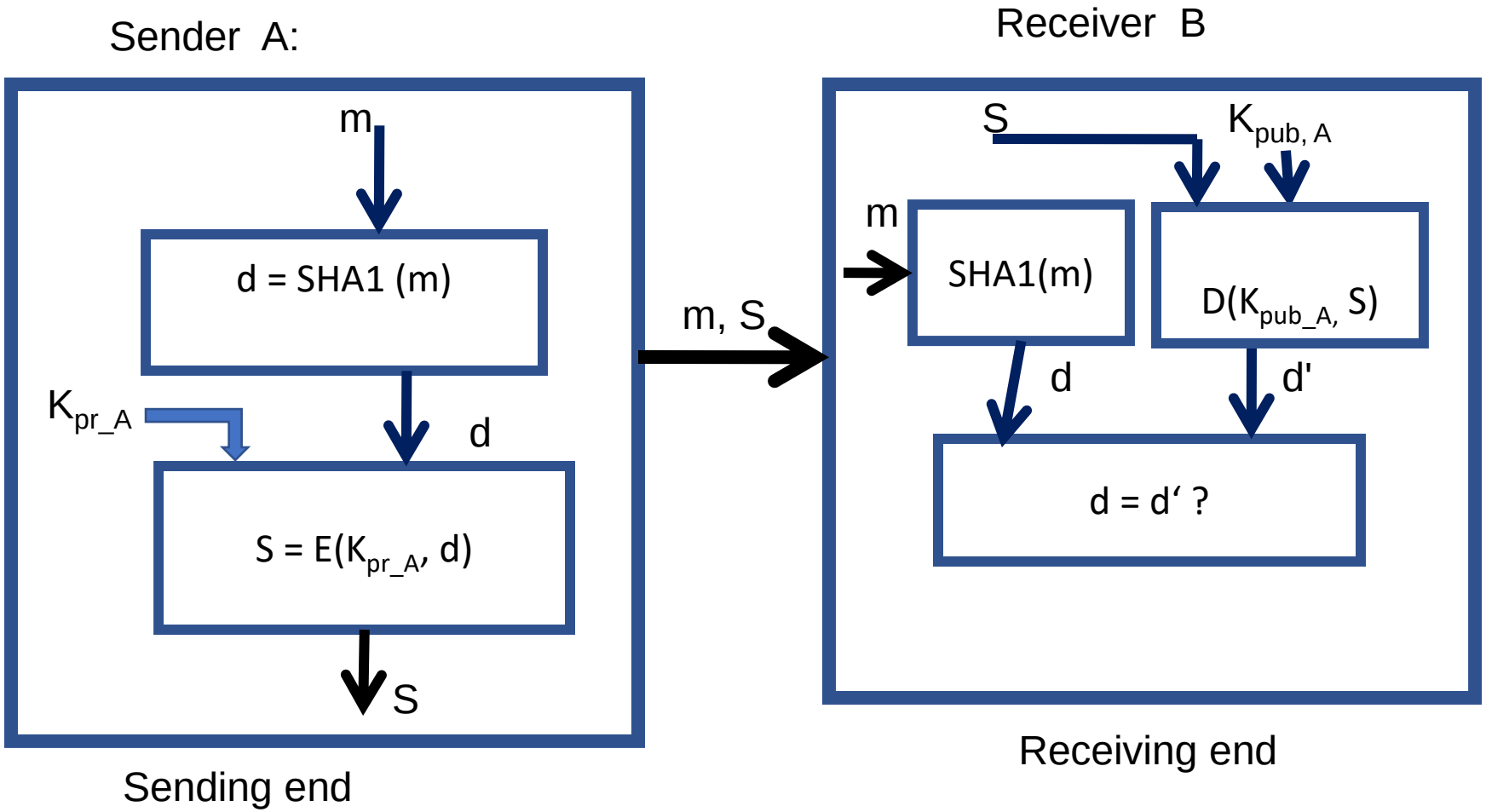
Pretty Good Privacy (PGP)

- widely used de facto secure email
- developed by Phil Zimmermann
- selected best available crypto algs to use
- integrated into a single program
- on Unix, PC, Macintosh and other systems
- originally free, now also have commercial versions available

PGP Operation – Authentication

1. sender creates message
2. use SHA-1 to generate 160-bit hash of message
3. signed hash with RSA using sender's private key, and is attached to message
4. receiver uses RSA with sender's public key to decrypt and recover hash code
5. receiver verifies received message using hash of it and compares with decrypted hash code

PGP Authentication



PGP Authentication

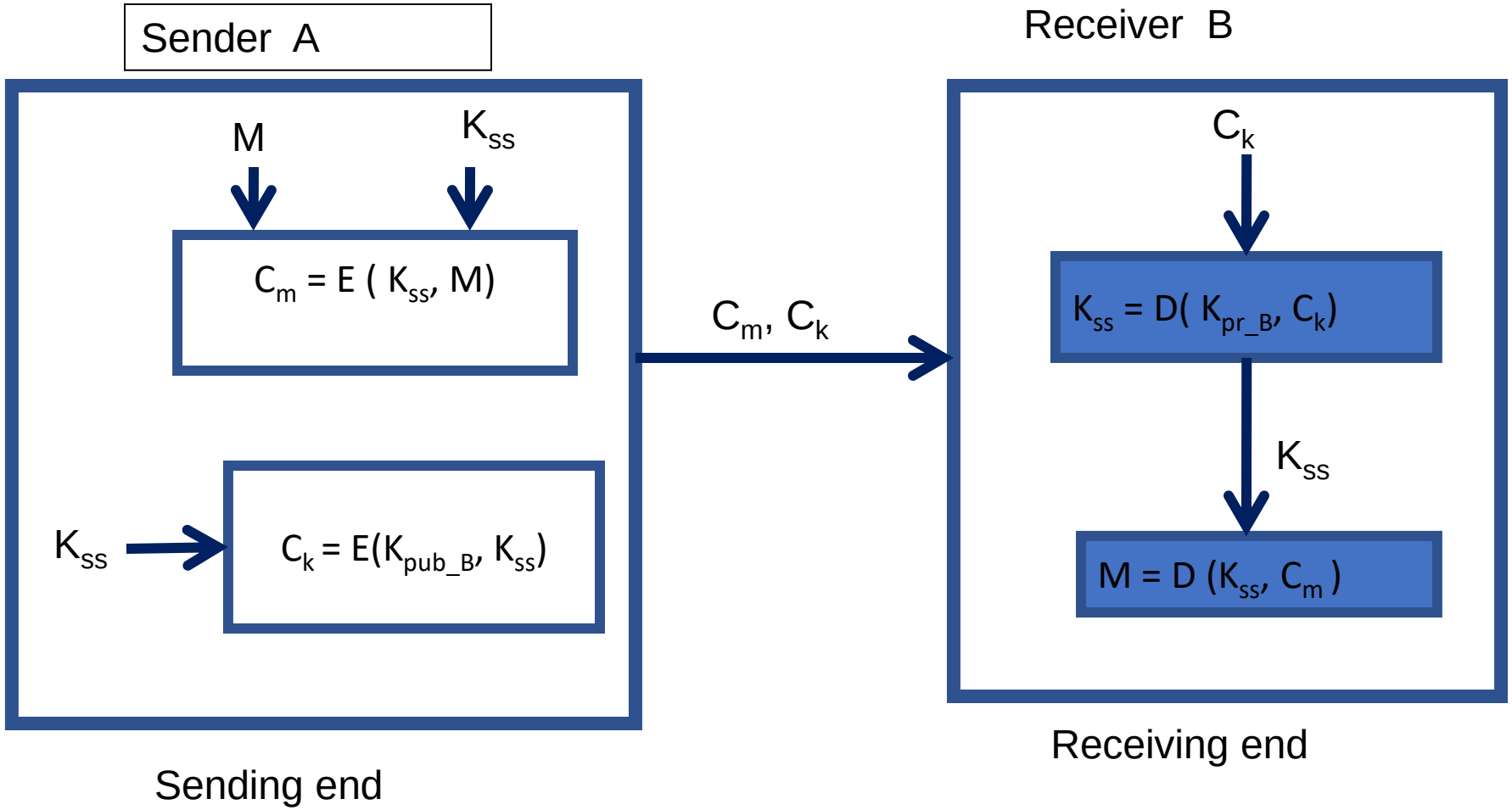
1. Signature (S) of sender is decrypted using sender's (A's)public key (K_{pub_A}), so the receiver identified and authenticated the sender.

2. If hash code d and d' is same (equal) the original message has been received. The integrity of the message has been verified,.

PGP Operation – Confidentiality

1. sender generates message and 128-bit random number as session key for it
2. encrypt message using CAST-128 / IDEA / 3DES with session key
3. session key encrypted using RSA with recipient's public key, & attached to msg
4. receiver uses RSA with private key to decrypt and recover session key
5. session key is used to decrypt message

PGP Confidentiality



PGP Operation – Confidentiality & Authentication

- can use both services on same message
  - create signature & attach to message
  - encrypt both message & signature
  - attach RSA/ElGamal encrypted session key

PGP Operation – 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

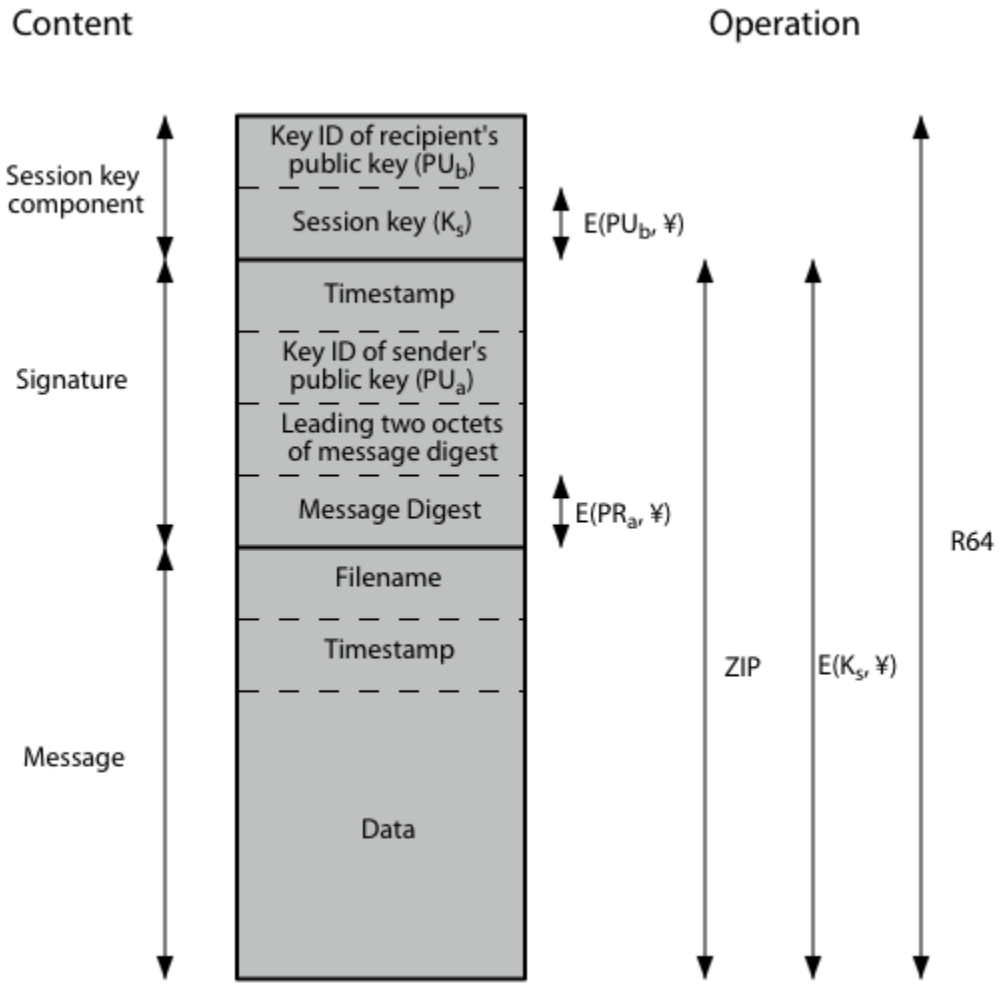
PGP Operation – Email Compatibility

- 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 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

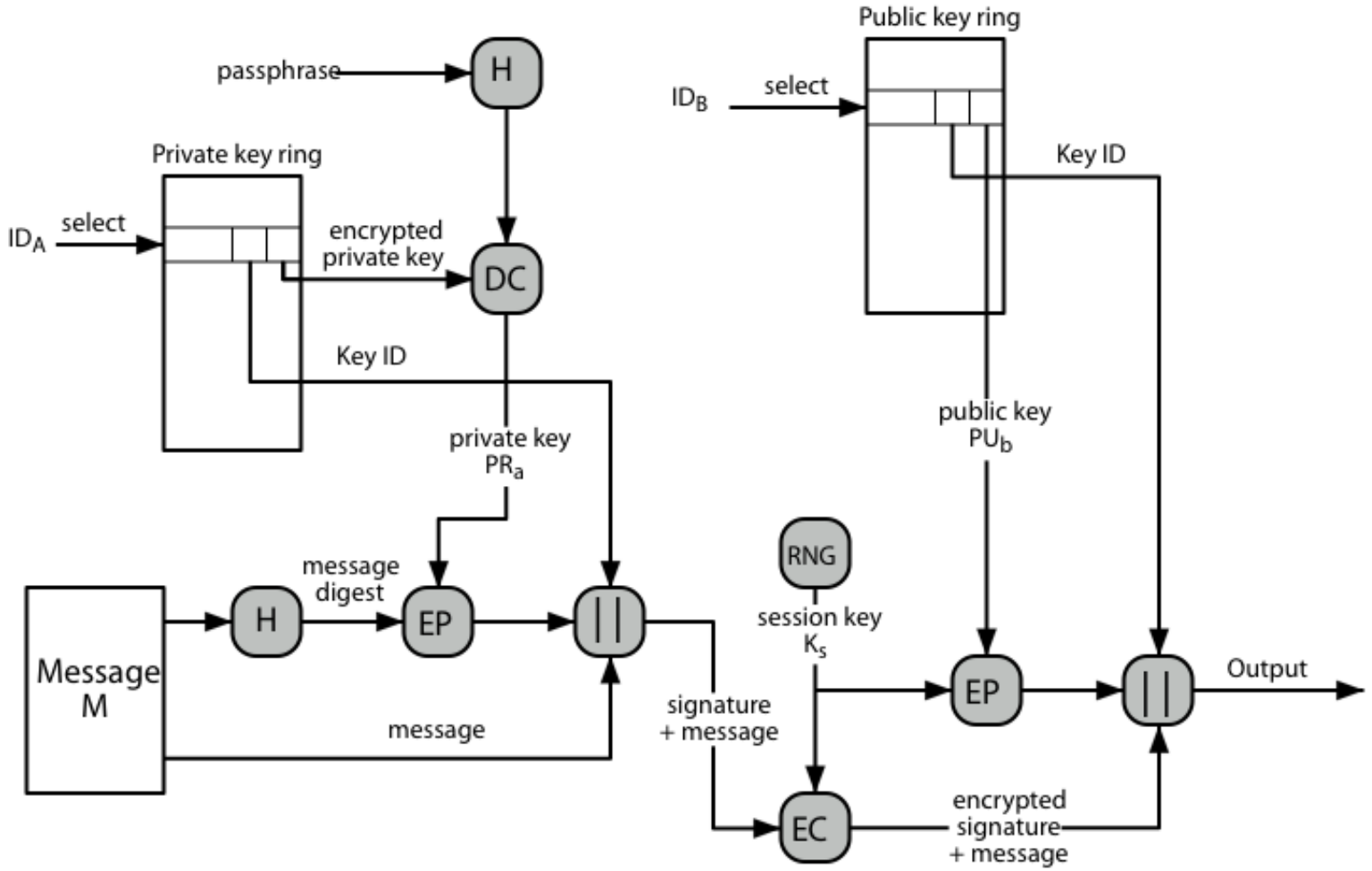
PGP Message Format



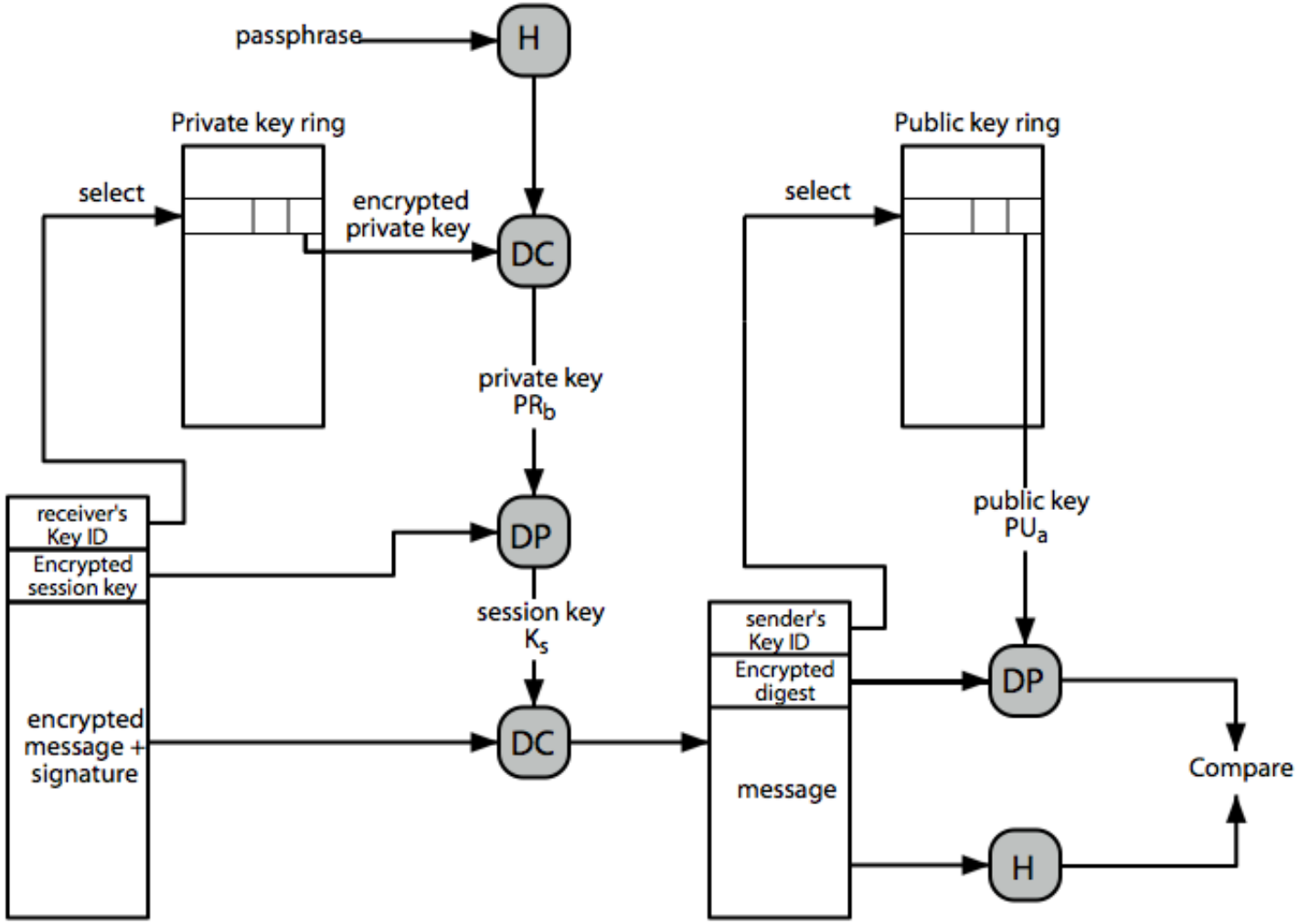
PGP Key Rings

- each PGP user has a pair of keyrings:
  - public-key ring contains all the public-keys of other PGP users known to this user, indexed by key ID
  - private-key ring contains the public/private key pair(s) for this user, indexed by key ID & encrypted keyed from a hashed passphrase
- security of private keys thus depends on the pass-phrase security

PGP Message Generation



PGP Message Reception



PGP message

- K_s – Session key
- PU_a – Public key of A
- EP – Public key encryption
- DP – Public key decryption
- EC - Symmetric (private) key encryption

PGP message

- DC - Symmetric (private) key decryption
- H – Hash function
- || - Concatenation
- Z – Compression using ZIP algorithm
- RNG – Random number generator

Description of PGP Message generation

- **Sender is A and Receiver is B.**
- **From Sender A:**
- 1. a) Take private key (K_{pr_A}) from the private key ring.
- b) Encrypt the hash value of the message
- d= H(M)
- 2. Encrypt d to create signature
- $S = E(K_{pr\_A}, d)$

PGP message generation

3. Do concatenation

   $CN\_1 = ID_{K\_A} \mid\mid S \mid\mid M$ (ID_{K_A} key ID of A)

4. Generate session key Ks.

5. a) Encrypt session key using K_{pub_B}

   b) Encrypt CN_1 Using Ks

      $C_{CN\_1} = E\ (Ks, CN\_1)$

6. Do concatenation and send it to B

   $CN\_2 = ID_{K\_B} \mid\mid C_{CN\_1}$

PGP Message reception

- **B has received CN_2 from A**
- 1. Take private key of B from key ring using ID_{K_B}
- 2. a) Find Session key from CKs
- Ks = D (K_{pr_B}, C_{Ks})
- b) Decrypt C_{CN_1} using Ks
- 3. a) Use IDK_A and get public key of A from public key ring.

Message reception

- b) Decrypt S to get message diget
- $d = D (K_{pub\_A}, S)$
- [since S was created using K_{pr_A}]
- c) Find Hash value of M
- $d' = H (M)$
- 4. Compare, d = d'?

PGP Key Management

- rather than relying on certificate authorities
- in PGP every user is own CA
  - can sign keys for users they know directly
- forms a “web of trust”
  - trust keys have signed
  - can trust keys others have signed if have a chain of signatures to them
- key ring includes trust indicators
- users can also revoke their keys