History of e-mail

- Early 1980 :Secure/32, Charli Merritt, using PKC
- 1986 : Mail Safe, RSADSI, DOS
- 1990 :
  - PEM(Privacy Enhanced Mail)
    - ✓ RIPEM (Riordan's Internet PEM)
    - ✓ TIS/PEM
  - PGP (Pretty Good Privacy)
  - S/MIME : Multimedia e-mail

Document of PEM

- RFC 1421, Part I: Message Encryption and Authentication Procedure
- RFC 1422, Part II: Certificate-based Key Management
- RFC 1423, Part III:Algorithms, Modes, and Identifiers
- RFC 1424, Part IV : Key Certification and Related Services

Design Environments of PEM

- Work with existing e-mail system in Internet
- Not restricted to particular host or OS
- Compatible with normal, non secure e-mail
- Performed on PC as well as on large system
- Compatible with a variety of key-management approach including manual distribution, centralized key distribution

Security Services of PEM

- Confidentiality
- Data origin authentication
- Message Integrity
- Non-repudiation of origin
- Key Management

Cryptographic Algorithm

- Data Encryption : DES in CBC
- Key Management : DES in ECB,CBC and RSA
- MIC : RSA+MD2, RSA+MD5
- Digital Signature : RSA+MD2, RSA+MD5

Style of message

- Ordinary, unsecured data
- MIC-Clear : integrity and authentication, but no confidentiality (integrity-protected unmodified data)
- MIC-Only : MIC-Clear + encoding(Integrity-protected encoded data)
- ENCRYPTED : MIC-Only + confidentiality(encoded encrypted integrity-protected data)

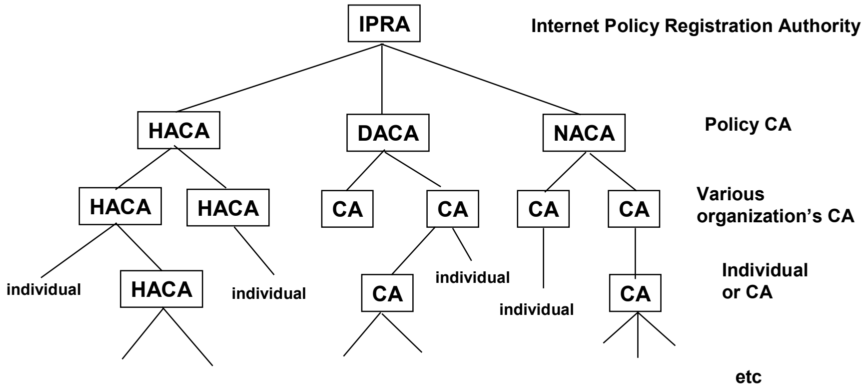
PEM Message

BEGIN-PRIVACY-ENHANCED-MESSAGE
Processing Type
Content Domain
Message text encryption algorithm
Issuing authority
Version/expiration
Origination certificate
Originator key information
Issuer certificate
MIC information
Issuing authority
Version/expiration
Encrypted DEK
User Text
END-PRIVACY-ENHANCED-MESSAGE

Processing steps of PEM Message

- **Sending**
  - Canonicalization
  - Message Integrity and originator authentication
  - Encryption(optional)
  - Transmission encoding(optional)
- **Receiving**
  - Decoding(optional)
  - Decrypting(optional)
  - Verifying message integrity and authenticity
  - Translation

Certification Hierarchy



HA : High Assurance, DA:Discretionary Assurance, NA:No Assurance

PGP

- Program for confidentiality and authentication service
- Select best available algorithm
  - Integrate algorithms into general-purpose
  - Made the package and its document, including source code, freely available via Internet
  - Low-cost commercial version by Viacrypt and Public-domain version

Background of PGP

- Available in various platforms
- Use algorithm survived extensively public review like RSA, DSS, DH, CAST-128, IDEA and 3DES, SHA-1
- Wide range of applicability from cooperation to individual
- Not developed by, nor controlled by, any government and standards organization

History of PGP(I)

- Designed by Phil Zimmerman
  - High security
  - public domain S/W
  - popular for personal use
- PGP Classic : Can't handle Internet Mail
  - PGP v.1.0 : '91.6
  - PGP v.2.0 : '92. 9
  - PGP v.2.3a : '93. 7 (last version of PGP didn't use RSAREF)
  - PGP v.2.4 : original ViaCrypt PGP
  - PGP v.2.5 : Interim release of PGP with RSAREF
  - PGP v.2.6 : Freeware version of PGP
  - PGP v.2.7 : Commercial version by ViaCrypt

History of PGP(II)

- 4 versions
  - PGP Classic : non commercial use
  - PGP 5.0 : Improve security but don't adapt RSA
  - PGP/MIME
    - ✓ MIME-based
    - ✓ Use special certificate
    - ✓ Handle Internet Mail
  - OpenPGP
  - Use DH, DSA, SHA-1
  - Interoperability with S/MIME

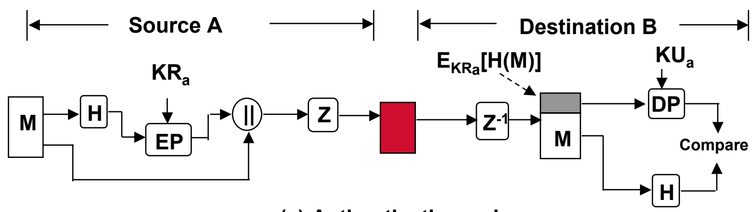
Features of PGP

Function	Algorithm	
Digital Signature	DSS/SHA or RSA/SHA	
Message Encryption	CAST-128 or IDEA or 3DES (64bCFB) w/ DH or RSA	
Compression	ZIP	(Note) Signing before compression Encryption after compression
E-mail compatibility	Radix 64	
Segmentation		

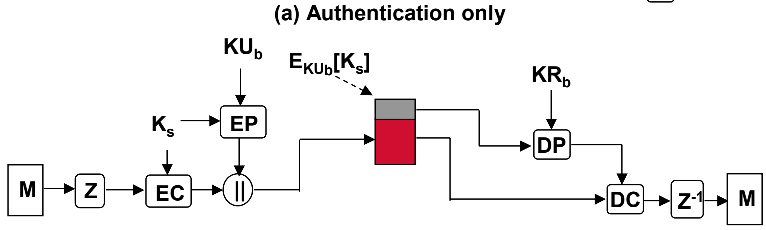
Notation

- Ks : session key for conventional algorithm
- KR_a : Private key of user A for PKC
- KU_a : Public key of user A for PKC
- EP : PK encryption
- DP : PK decryption
- EC : conventional encryption
- DC : conventional decryption
- H : Hash function
- || : concatenation
- Z : compression
- R64 : conversion to radix 64 ASCII format

Security Service in PGP(I)

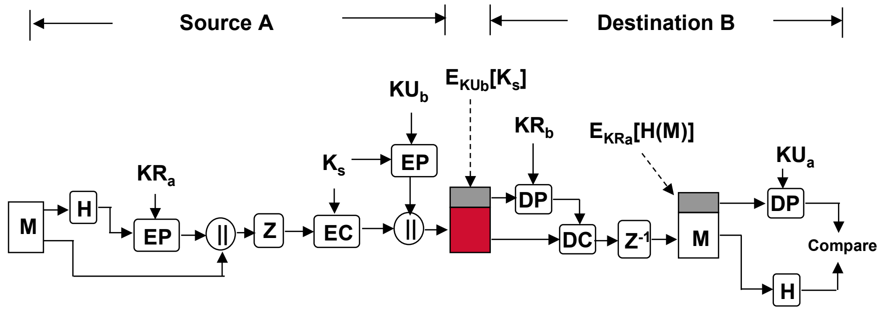


(a) Authentication only



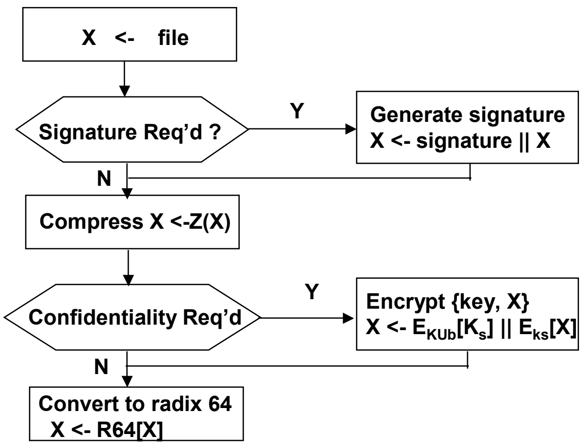
(b) Confidentiality only

Security Service in PGP(II)

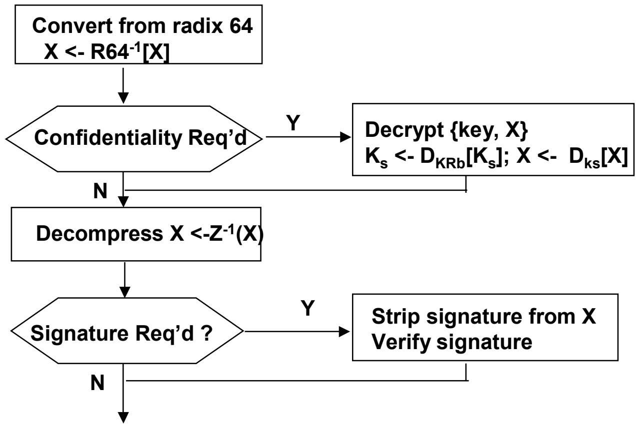


(c) Confidentiality and Authentication

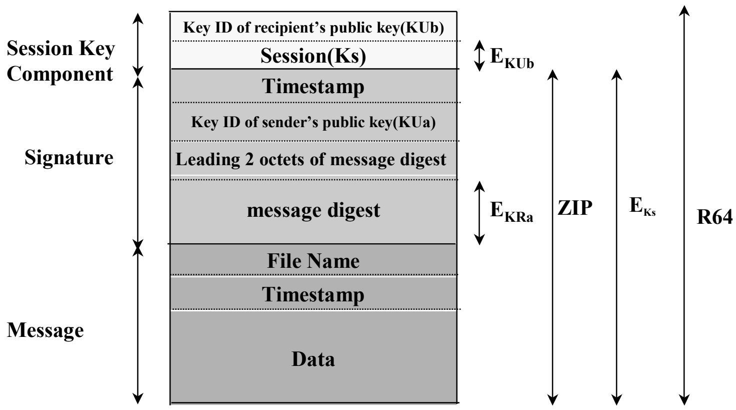
Steps of sending a message



Steps of receiving a message



Message format (A->B)



E_{KUb} : encryption with user b's public key
E_{KRa} : encryption with user b's private key
E_{Ks} : encryption with session key

ZIP : ZIP compression functions
R64 : Radix-94 conversion function

Key Management

- One-time session key : 128bit for CAST or IDEA, 168 bit for 3DES)
- Public Key
- Private Key
- Passphrase-based conventional key

Key Rings of PGP

Private Key ring : store his own public and private keys

Timestamp	KeyID	Public Key	Encrypted Private key	UserID
Ti	KU_i mod 2^{64}	KUi	$E_{H(Pi)}$[KRi]	User i

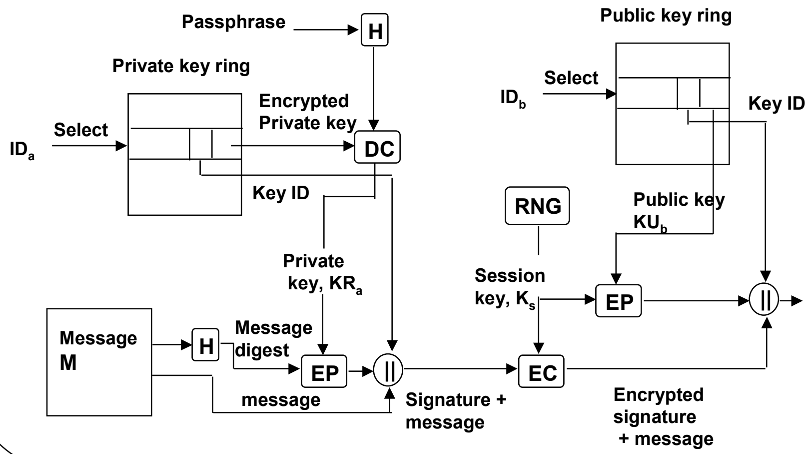
Public Key ring : store all known entities' public key

Time stamp	KeyID	Public Key	Owner Trust	UserID	Key Legitimacy	Signature(s)	Signature Trust(s)
Ti	KU_imod 2^{64}	KUi	trust_flagi	User i	trust_flagi	ERj(H([KUi])) ERk(H([KUi]))	complete marginal

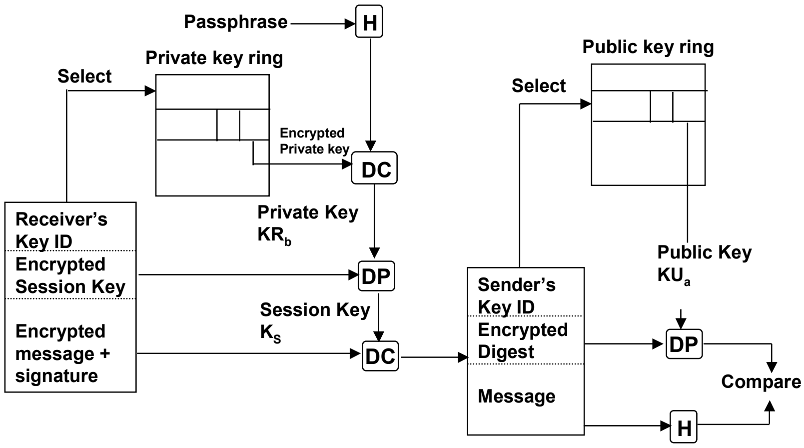
Use of private key

- ❑Using IDEA, store encrypted key
  - ✓User selects passphrase
  - ✓When generating private/public key pairs, use passphrase
  - ✓ Passphrase is inputted to hash ft. MD5 (SHA-1), Use 128 (160)-bit hash value as key of IDEA
- ❑After use, delete it from system

Message sending (Detailed)



Message receiving (Detailed)



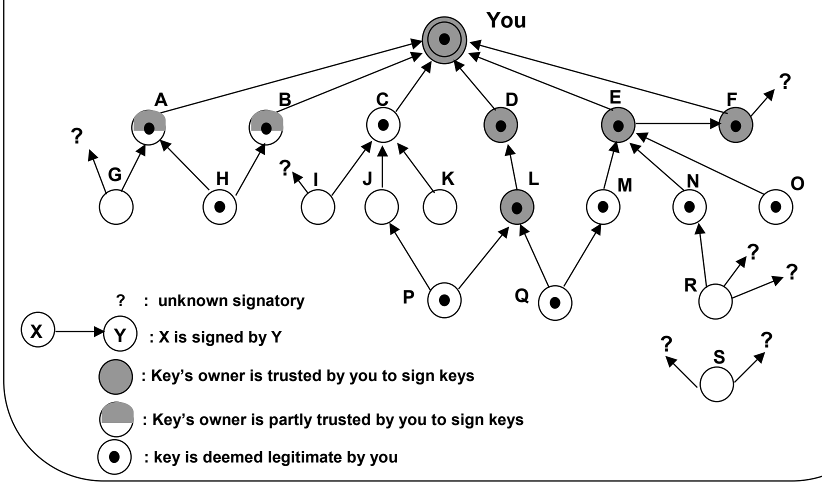
Distribution of Public Key

- Direct delivery (floppy disk, mail,..)
- Sending e-mail and confirm by telephone

B $\xrightarrow[\text{(by e-mail)}]{\text{Key}}$ A $\xrightarrow[\text{fingerprint \quad (by telephone)}]{\text{(128-bit MD5 digest) \quad fingerprint}}$ B (matching)

- TTP
- CA

PGP's Trust Model



Revocation of Public key

- ❑ Issue public key revocation signature
  - – Similar form of usual Signature Certificate
  - – Signature using secret key of public key to be revocated
  - – Propagate as many as possible
- ❑ All public keys signed by revocated key
  - – Make *Owner_trust* and *key_legitimacy* to untrust