Cryptography

Part 1, Basics

Introduction

- Cryptography is the art of designing mathematical methods for protecting information
- A means of transmitting confidential information securely over open networks: encryption
- Classical cryptographical algorithms have been known for hundreds of years
- Innovations in the last two decades have created a whole new branch of cryptography: asymmetric (public key) cryptography
  – Digital signatures
  – Secure on-line key exchange

Symmetric Cryptography

- Based on using a shared secret, a secret piece of information
  - Used to transform information to an encrypted form
  - Only those who know the secret are able to recover the content of encrypted data
- Two facts determine the procedure:
  - Transformation method: the encryption (enciphering) algorithm
  - Shared piece of information: the secret key

Example: Caesar Cipher

- Alice and Bob agree that they use Caesar Cipher to encrypt their communication, with the number 4 as their secret key
  - Alice encrypts the message M = YES

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

  - Alice sends Bob the encrypted message C = E(M) = UAO
  - Now that Bob knows the secret key 4, he can calculate the inverse transformation E^{-1}

```
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

  - Bob decrypts C (applies the inverse transformation) to recover the original message:
    $E^{-1}(C) = M = YES$

Example: Analysis of Caesar Cipher

- Is it possible for an attacker to find out the content of the encrypted message?
- Kerkhoff's Principle: we assume that the attacker knows the encryption algorithm used
  - The set of good encryption algorithms is small, the potential attacker can try all of them relatively easily
- Assume that an attacker, Eve, gets to know C=UAP
  - Since Eve knows that the algorithm is Caesar Cipher, she may start ciphering C with different keys
  - Using the English alphabet, there are only 26 possible keys
  - In at most 26 trials, Eve breaks the cipher
- The method of trying all possible keys is called brute force

One-Time Pad

- Instead of using Caesar Cipher, with the constant shift of 4 for every letter, Alice and Bob may alternatively agree:
  - Left-shift the first letter by 11 positions, the second letter by 1 position and the third letter by 3 positions
  - This algorithm would produce, when applied to the word M = YES, C = E(M) = ODP
  - Here the secret key constitutes of the triplet (11,1,3)
- Now the number of all possible keys is 26*26*26=17576
  - This is still very small a key space to be exhaustively searched with today's computing resources
  - However, trying every possible key would yield all English three letter words!

One-Time Pad (cont.)

- This algorithm, called One-Time Pad, is perfectly secure against brute force attacks (and all other imaginable cryptographic attack methods)
- In computer-based cryptography, the set of letters is 0 and 1
- For example, the bit-string K = 001101 could be used as a secret key for One-Time Pad to encrypt a 6-bit string M = 011001, C = E(M) = M xor K:

```
M 011001
K 001101
C 010100
```

- One-Time Pad has two major limitations
  - The same key may be used only once (otherwise there is a statistical attack for breaking the cipher)
  - The length of the key is equal to the length of the message

Cryptanalysis

- Cryptanalysis is the art of finding techniques for breaking cryptographical algorithms
- Usually based on statistical methods
  - Counting frequencies of different patterns in ciphertext may reveal some redundancies in the original plaintext
- Cryptographical attacks may be categorized as follows:
  - Ciphertext only attack means determining the secret key from the knowledge of a set of encrypted messages
  - Known plaintext attack means determining the secret key using the knowledge of a set of plaintext/ciphertext -pairs
  - In a chosen plaintext attack the attacker has gained knowledge of plaintext/ciphertext -pairs for chosen plaintexts

Building Blocks for Good Algorithms

- Design of symmetrical cryptographical algorithms aims at obscuring the redundancies in the plaintext
- Two basic methods for implementing this are confusion and diffusion
  - Confusion is produced using substitution; when a long block of plaintext is substituted for a different block of ciphertext, the statistical patterns of plaintext become hard to detect
  - Diffusion dissipates the redundancy of the plaintext by spreading it out over the ciphertext; this can be produced using permutation, i.e. reordering the parts of a plaintext message

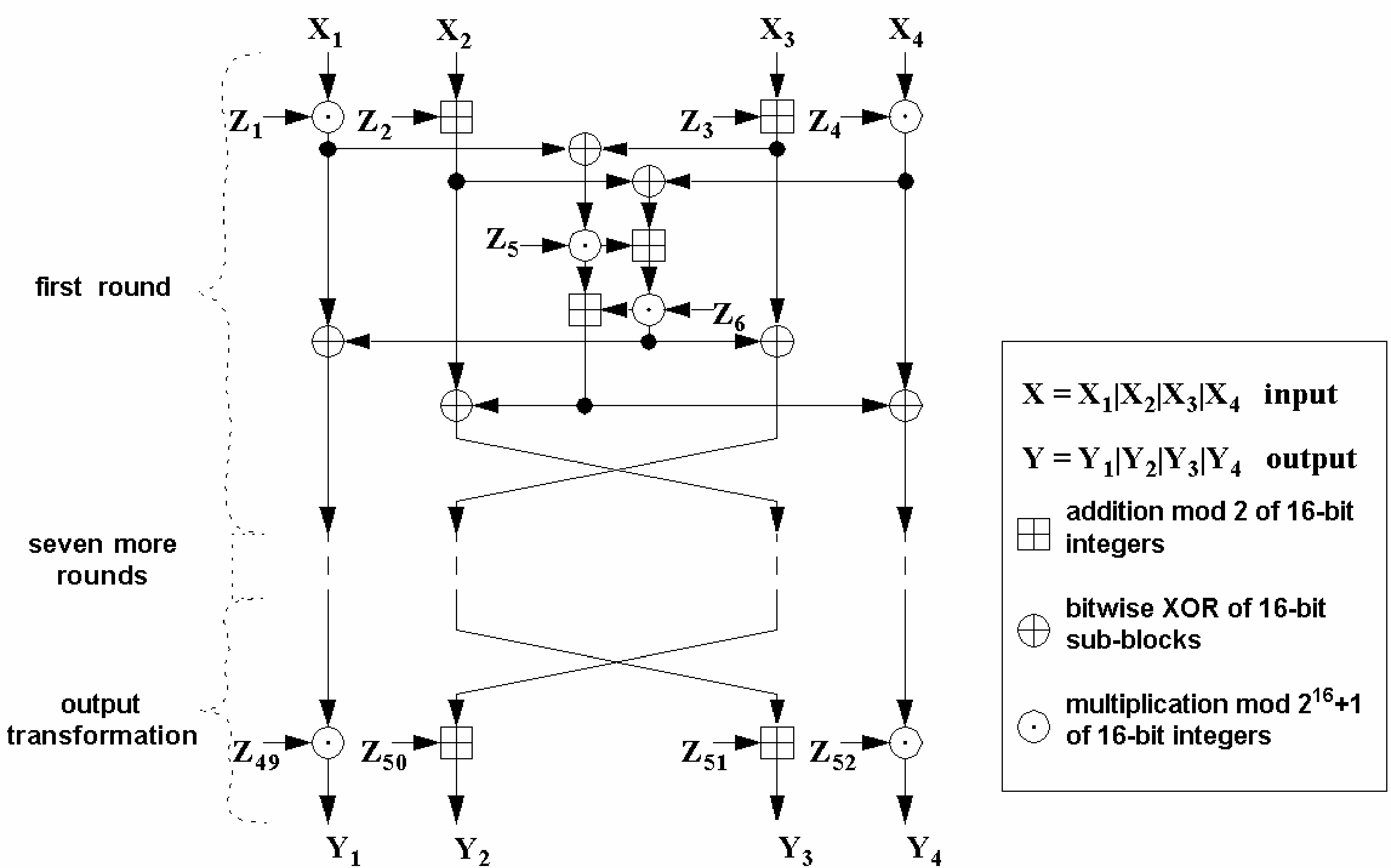
IDEA

- Designed by X. Lai and J. Massey in 1992
- 64-bit blocks in, 64-bit blocks out
- Same 128-bit key used for encryption and decryption
- Involves mixed use of three algebraic operations on 16-bit integers:
  – Addition mod 2 (bitwise XOR)
  – Addition mod 2^{16}
  – Multiplication mod $2^{16}+1$
- Fast implementations in software and hardware
- A default cipher in PGP and SSH

IDEA Subkey Calculation

- For encryption, 52 16-bit subkeys are generated from the input 128-bit key, to be used in the 8 rounds of IDEA operation
  - The 128-bit input key K is first divided to 8 16-bit blocks: $K=K_1|K_2|K_3|K_4|K_5|K_6|K_7|K_8$
    - These are stored into memory: $Z_1=K_1$, $Z_2=K_2$, ..., $Z_8=K_8$
  - Then the bits of K are rotated 25 positions to the left and split again into 8 16-bit blocks
    - The resulting blocks are stored into memory as Z_9, ..., Z_{16}
  - The above step is repeated until all Z_1, ..., Z_{52} are calculated
- For decryption, the corresponding 52 16-bit subkeys are generated similarly, in a slightly modified way

IDEA Operation



Security of IDEA

- To date, the brute force attack is the best known way for attacking IDEA
- The key length of 128-bits makes the number of all possible keys incredibly large: $2^{128} = 10^{38}$
  - With one billion hardware devices each capable of performing billion encryptions per second the brute force attack would take 10^{13} years, longer than the age of universe
  - An array with 10^{24} such chips could perform the brute force attack in a day, but there are not enough silicon atoms in the universe for that
- However, IDEA is still a relatively new algorithm; in cryptography, there is no such thing as absolute truth:
  - As new mathematics is invented, old algorithms may be broken, and become replaced by new ones

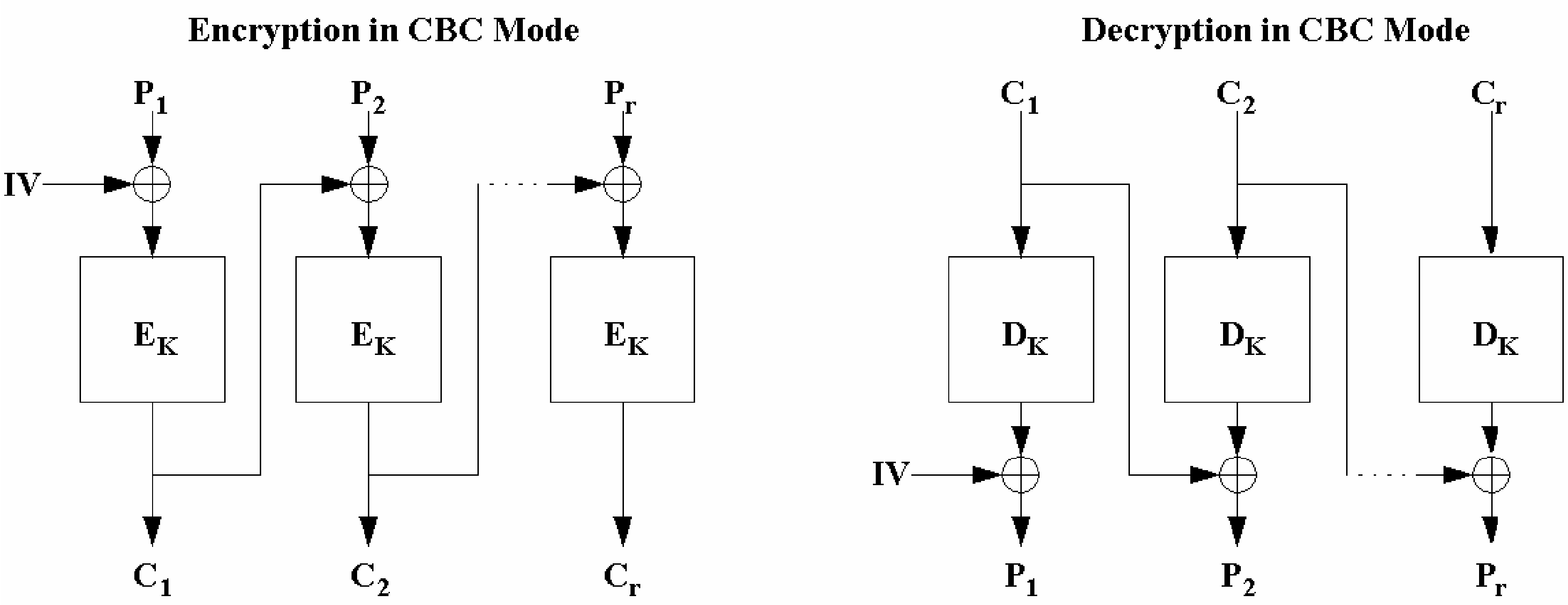
Other Symmetric Encryption Algorithms

- DES (Digital Encryption Standard) is perhaps the most widely used encryption algorithm
  – The relatively small key length of 56 bits makes DES vulnerable to a brute force attack with today's computing resources
- 3DES (Triple DES) performs DES encryption three times, yielding effective key length of 112 bits
- RC2 and RC4 can be used with variable key lengths
  – With the default 40-bit keys, these ciphers are relatively easy to break
- New encryption algorithms are being published all the time
  – Most of them are trivial to break
  – It is recommended to use older, widely known algorithms

Cipher Feedback

- Normally, encryption is done one 64-bit block at a time
  - Every 8-octet string of plaintext encrypts to 8 octets of ciphertext
  - This way of applying a symmetric block encryption algorithm is called Electronic Codebook (ECB) Mode
- The ECB mode has a potential security risk: encrypting similar blocks of 8 octets always produces similar ciphertext blocks
  - If the input plaintext contains e.g., formatted text, collecting large amounts of ciphertext may help an attacker to gain information on the content of the plaintext
- Cipher Block Chaining (CBC) is a mode that can be used to add feedback to produced ciphertext
  - An Initialization Vector (IV) is a random block of data used in the beginning to prevent identical plaintexts producing same output

Cipher Block Chaining Mode



One-Way Hash Functions

- A hash function is a mapping that compresses an input bit-string to a smaller bit-string of fixed length
- Cryptographic hash functions satisfy two requirements: they are one-way and collision-free
  - A hash function H is called one-way, if for a given hash value C it is practically impossible to find out an input M such that H(M) = C
  - A collision-free hash function H is such that it is computationally infeasible to find out two values M_1 and M_2 such that $H(M_1)=H(M_2)$
- Most widely used cryptographic hash functions are SHA-1 (160-bit output) and MD5 (128-bit output)

Message Authentication Codes

- Cryptographic hash functions are used to produce Message Authentication Codes (MAC)
  – To verify integrity of a message, i.e. that it has not been altered in transit
  – To authenticate the identity of a message's originator
- Example:
  – Let's assume that Alice and Bob share a secret key K
  – Alice creates a message M of 1000 bits
  – Alice uses SHA-1 to compute a 160-bit hash V=H(M|K) of the message concatenated with the secret key
  – Alice transmits Bob the strings M and V
  – When Bob receives the message, he also computes H(M|K) from the received message M and verifies that this equals V

Two Problems with Symmetric Cryptosystems

- Key agreement
  - We need an efficient way to agree upon a common secret key
    - distant locations
    - peers not previously known to us
- Identification (authentication of identity)
  - How can we be sure about the peer’s identity?
    - in the Internet we are "blind"

Asymmetric Cryptography

- The requirement of a shared secret with symmetric encryption algorithms is a serious limitation in large open networks:
  - If every user wishes to communicate with each other privately in a network of n users, a total of n(n-1)/2 keys are needed
  - Every key needs to be distributed using a reliable channel (a courier, etc.) before communication can start
- Asymmetric (public key) cryptography addresses these issues
- Asymmetric encryption algorithm involves using separate keys for encryption and decryption
  - The knowledge of either key does not reveal any information about the other key

Asymmetric Cryptography (cont.)

- In practice, one of the keys is kept secret and the other is published to other parties of the system
  – The secretly stored key is called private key
  – The other key is called public key
- Asymmetric algorithms are mainly used for:
  – Encryption
  – Digital signatures
  – Key agreement
- Other applications include user authentication schemes and anonymous digital cash

Public Key Encryption

- Let E be an asymmetric encryption algorithm
- When Alice wants to send Bob a confidential message M, she may act as follows:
  - Alice fetches Bob's public key K_{Bob} from a publicly available directory
  - She then encrypts M using K_{Bob}: $C = E(M, K_{Bob})$ and send this to Bob
  - Now Bob is the only person being able to decrypt the message

Digital Signatures

- The order of keys with asymmetric encryption can be reversed, resulting in a digital signature
  - Alice calculates a hash H(M) of M
  - She then encrypts the hash with her private key: $S = E(H(M), K^{-1}_{Alice})$ and sends Bob the pair M, S; where S is Alice's signature
  - Now Bob may verify Alice's signature using Alice's public key, by checking that $E(S, K_{Alice}) = H(M)$

RSA

- An asymmetric encryption algorithm invented by Rivest, Shamir and Adleman in 1978
- Key pair generation:
  - Choose two large prime numbers, p and q; denote their product pq=n, known as the modulus of the RSA algorithm
  - Choose a random integer e, such that e and (p-1)(q-1) have no common divisors; this will be the public key
  - Compute the integer d, which satisfies ed=1 (mod (p-1)(q-1)); this is the private key
  - Discard all data about p and q

RSA Encryption

- Encryption of a message (bit-string) M
  - Divide M into k-bit blocks: $M=m_1m_2\ldots m_r$
    - $k = |n|-1$ ($|n|$ is the length of the modulus n in bits)
  - Regard m_1 as an integer $0<m_1<n$, and compute $c_1 = m_1^e \bmod n$
  - Repeat the same for $m_2, m_3, \ldots, m_r$
  - Convert the integers $c_1, c_2, \ldots, c_r$ to binary
  - The encrypted message is now (the concatenated bit-string) $C = c_1c_2\ldots c_r$

RSA Decryption

- Decryption of the ciphertext bit-string $C=c_1c_2...c_r$
  – Interpret the k-bit strings c_1, c_2, ..., c_r as integers represented in binary notation
  – Compute $c_1^d \bmod n$, $c_2^d \bmod n$, ..., $c_r^d \bmod n$
  – Compute $c_1^d \bmod n = (m_1^e \bmod n)^d \bmod n = m_1^{ed} \bmod n = m_1 \bmod n = m_1$, and repeat this for c_2, c_3, ..., c_r
  – The original message is now recovered by concatenating the binary representations of the intermediary results: $M = m_1m_2...m_r$
- Decryption works, because e and d were chosen so that ed = 1 (mod (p-1)(q-1)):
  – this implies $m_i^{ed} \bmod n = m_i \bmod n$, a basic fact from elementary number theory

Security of RSA

- It is assumed that the attacker knows the public key e and the modulus n
- After several years of cryptological study, the best known method for computing the private key d using the knowledge of n and e is to factor n into primes
  – That is, find the numbers p and q in the original composition of n=pq
- If the modulus is very large ($n > 2^{1000}$) the integer factorization problem is very hard to solve
  – In February 1999 a 465-bit (140-digit) integer was factorized with hundreds of workstations running in parallel for several months
  – A result (April 1999) by Adi Shamir (the “S” of RSA) describes principles for constructing a hardware device potentially capable of factoring a 600-bit integer

Diffie-Hellman Key Exchange

- In 1976, W. Diffie and M. Hellman offered a solution to the on-line key distribution problem:
  - Alice and Bob agree on using a large prime number p and a base number $1<g<p$
  - Alice chooses a random number a: $1<a<p$, and computes $g^a \bmod p$
  - similarly, Bob chooses b: $1<b<p$, and computes $g^b \bmod p$
  - Alice sends Bob $g^a \bmod p$, and Bob sends Alice $g^b \bmod p$
  - Alice computes $(g^b \bmod p)^a \bmod p$
  - Bob computes $(g^a \bmod p)^b \bmod p$
  - Now, Alice and Bob share a common (secret) value, since $(g^b \bmod p)^a \bmod p = g^{ab} \bmod p = (g^a \bmod p)^b \bmod p$

Diffie-Hellman Analysis

- If someone eavesdropped the transaction between Alice and Bob, he/she would have gained knowledge of p, g, g^a mod p and g^b mod p:
  – The problem of determining g^{ab} mod p from the above information is known as the Diffie-Hellman problem
  – To date, no computationally efficient algorithm for solving this, if the prime $p > 2^{1000}$
- Alice and Bob and may start using symmetric encryption with g^{ab} as their shared secret key
- This Diffie-Hellman method is used today in SSH and IKE (Internet Key Exchange), for instance
- A generalized version runs on elliptic curves

Problems with Asymmetric Cryptosystems

- A asymmetric cryptosystem solves the key agreement problem
- We still have no solution to the identification problem
  – its easy to create fake public keys
- Other problems
  – are average Internet users competent enough to create good cryptographic keys?
  – what if I lost my private key?
- We need a public key infrastructure (PKI)
  – an organized system to securely distribute public keys

PGP

- Designed by Phil Zimmermann for providing cryptographic protection of e-mail and file storage
- Combines strong cryptographic algorithms with e-mail byte conversion and key management techniques
- Original versions written for Unix
  – Includes versions up to 2.6.x
  – Open source
  – Free of charge
- Commercial version available from Network Associates
  – Support for a variety of platforms; including Unix, Mac, and Windows
  – Graphical user interface

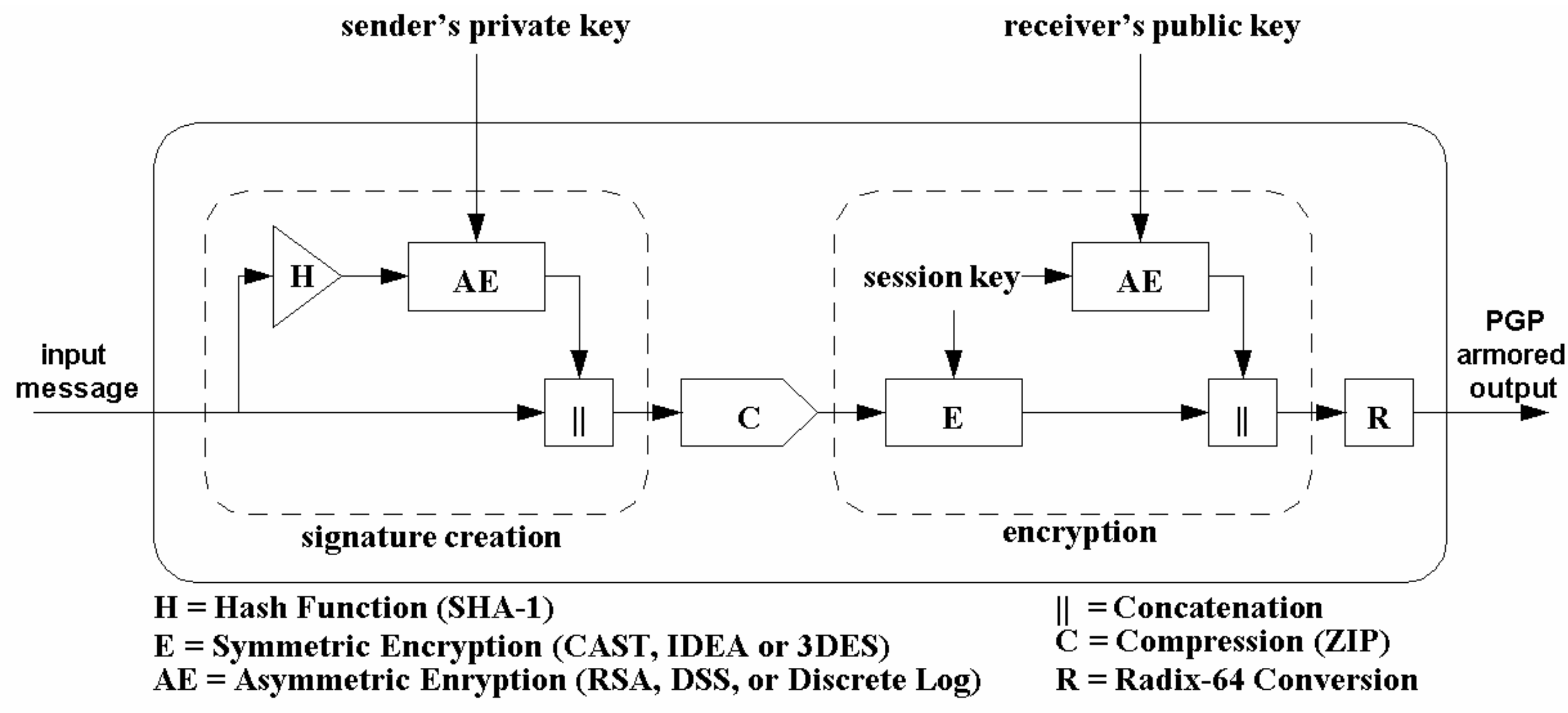
PGP Design Philosophy

- Written for individual end-users
  - Every user creates and manages his/her own keys
  - Every user has a freedom to choose, who to trust
- No administrative organization or governments involved
  - No hierarchy in trust relationships
- No standardization organizations involved
  - The product has become a de facto standard
  - OpenPGP is a recent attempt by IETF to create standards for establishing interoperability between different PGP implementations and mail clients

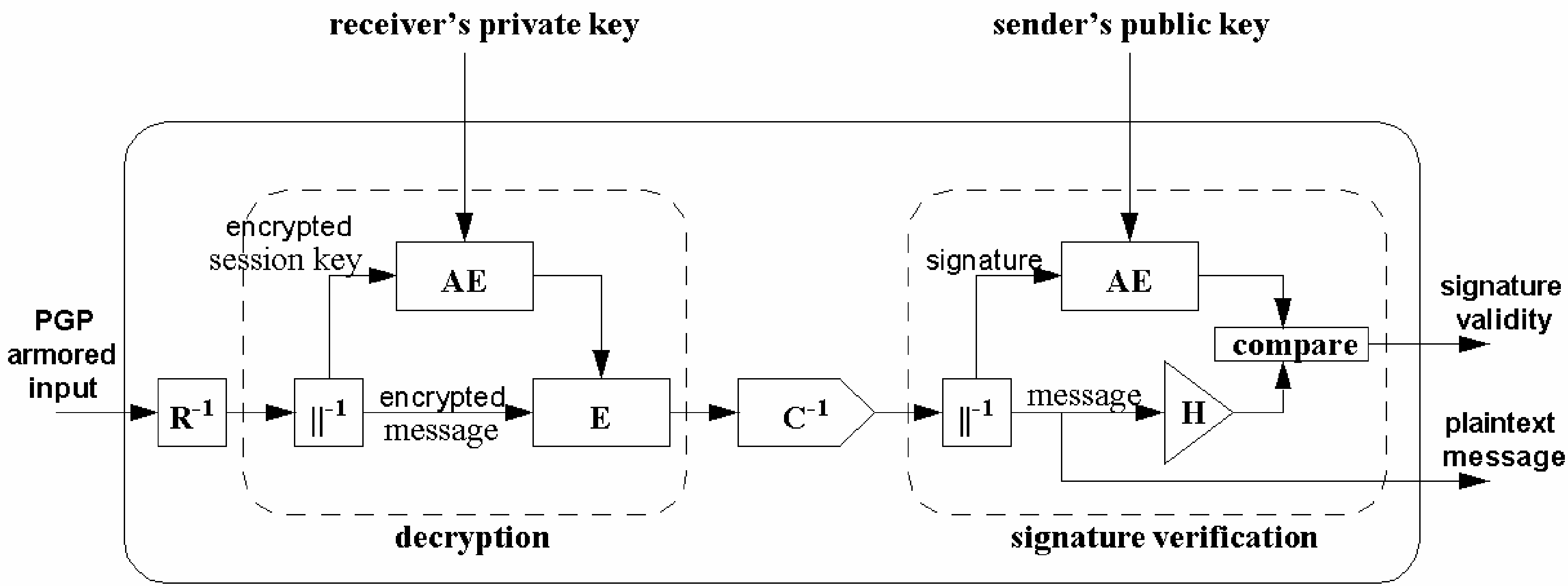
PGP Functionality

- PGP offers five services for mail messages:
  - Authentication using digital signatures
  - Confidentiality with the use of encryption
  - Compression
  - E-mail compatibility by converting binary data to 7-bit ASCII
  - Segmentation by fragmenting long messages into smaller chunks

Standard PGP Operation: Sending



Standard PGP Operation: Receiving



H = Hash Function (SHA-1)
E = Symmetric Encryption (CAST, IDEA or 3DES)
AE = Asymmetric Enryption (RSA, DSS, or Discrete Log)

$||^{-1}$ = Inverse Concatenation
C^{-1} = Uncompression (ZIP)
R^{-1} = Inverse Radix-64 Conversion

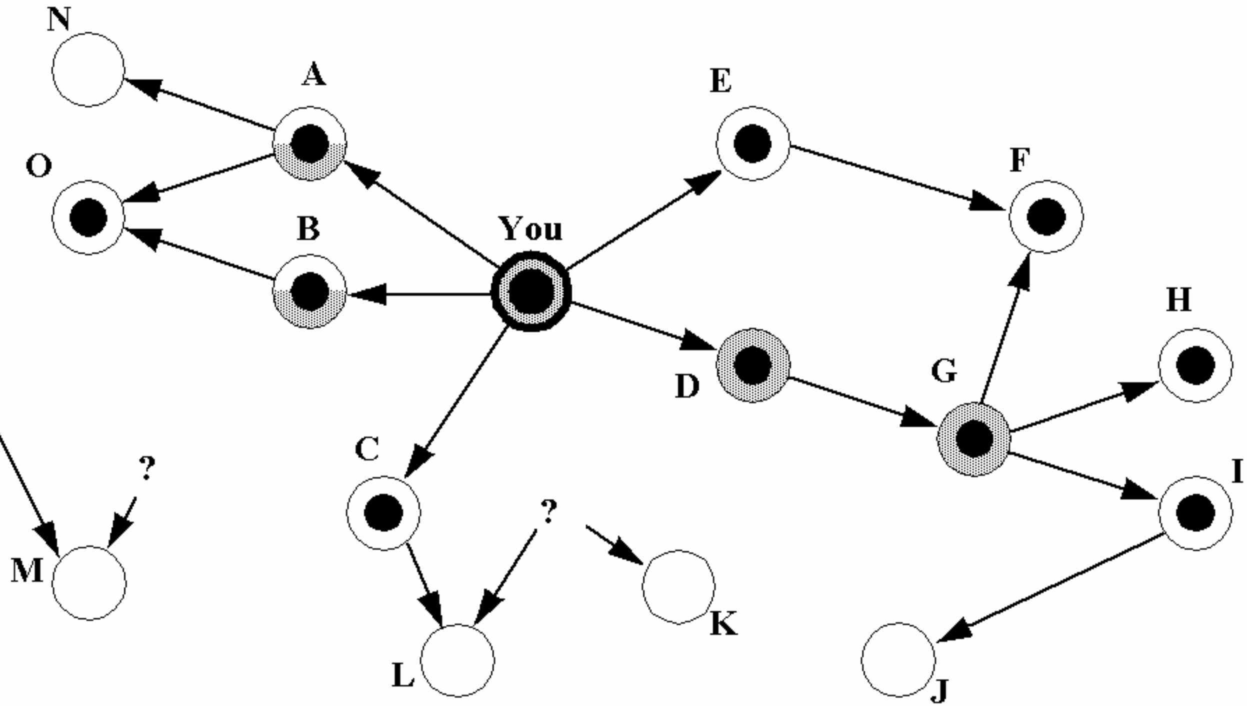
PGP Key Management

- Every user manages a local key ring in his/her disk, containing:
  - The public keys known to the user
  - Trust information about the keys
- When adding a new public key to the key ring, the user is prompted to assign a trust level to the key and its owner
  - How much the user trusts that the public key really belongs to its claimed owner
  - How much the user trusts the key owner to introduce new public keys

PGP Key Transport

- Public keys are generally distributed via e-mail or by submitting them to a key server
- How do we know that a public key really belongs to its claimed owner?
- We can call the claimed owner by telephone and ask him/her if the public key we got is correct
  - For convenience, we may ask the owner if the fingerprint (a 128-bit hash) of the public key is correct
- An alternative way is to obtain a public key from a mutually trusted individual
  - The trusted third party may indicate his/her trust for a public key by creating a signed certificate
  - The trusted party may be a widely-known certificate authority

A Web of Trust



= key is deemed legitimate by you

= key's owner is partially trusted to sign keys

= key's owner is fully trusted to sign keys

X ⟶ Y = X has signed Y's public key

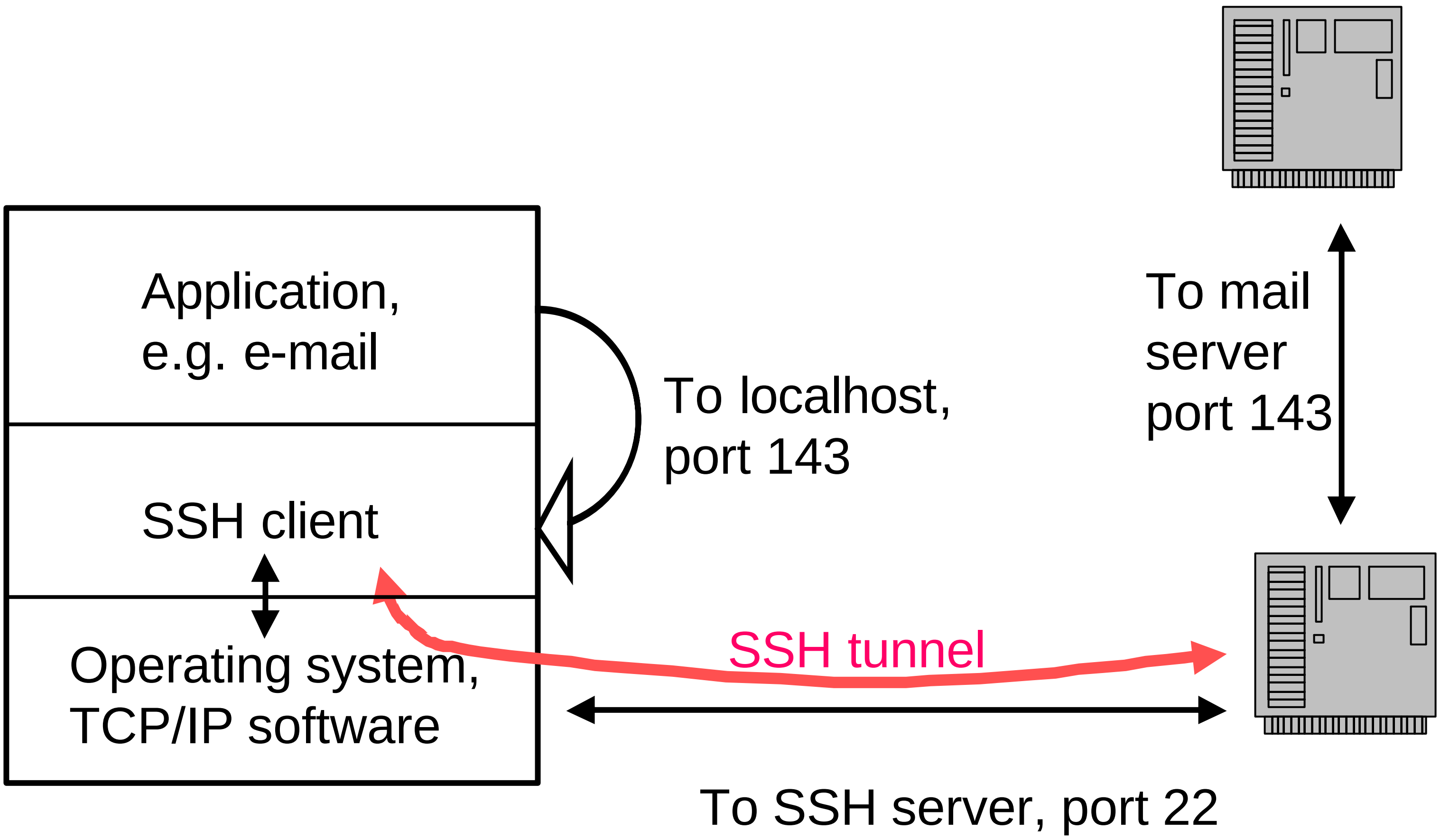
SSH

- SSH (Secure SHell) provices an encrypted TCP connection between two hosts on the network
  - Replaces Berkeley R-tools (rlogin, rcp, rsh)
  - Protects X-Window system traffic
  - Any TCP-connection can be tunneled over SSH
- Vulnerable to "Man in the Middle" -attack
  - SSH client does not know the host key until first connection
- Used mostly for systems administration and for tunneling traffic from external hostst to the protected domain
- No standard yet, drafts exist
  - http://www.ietf.org/html.charters/secsh-charter.html

TCP Tunneling

- A local SSH client can be configured to tunnel TCP connections from a local TCP port to a SSH server host and from there to another host
- This protects the traffic between the two SSH hosts
- Use requires changes to software settings, local host appears to be the server
- SSH server is usually in port 22

Tunneling with SSH



SSH 2.0 protocol key exchange

- Client contacts a server (a TCP connection is initiated)
- Server sends two public keys (server and host) and available algorithms
- Client simultaneously sends available algorithms
- Client creates a session key (symmetric), encrypts it with server's public keys and sends it to server
- A shared secret is now formed and a session is started
- Either side may request a renegotiation of keys
- User authentication is done after this