EE 122: Network Security I

Ion Stoica (and Brighten Godfrey)

TAs: Lucian Popa, David Zats and Ganesh Ananthanarayanan

http://inst.eecs.berkeley.edu/~ee122/

(Materials with thanks to Vern Paxson, Jennifer Rexford, and colleagues at UC Berkeley)

1

Security Requirements

- Authentication
  - Ensures that the sender and the receiver are who they are claiming to be
- Data integrity
  - Ensure that data is not changed from source to destination
- Confidentiality
  - Ensures that data is read only by authorized users
- Non-repudiation
  - Ensures that the sender has strong evidence that the receiver has received the message, and the receiver has strong evidence of the sender identity, strong enough such that the sender cannot deny that it has sent the message and the receiver cannot deny that it has received the message

2

Outline

- ➢ Cryptographic Algorithms (Confidentiality and Integrity)
- Authentication
- System examples

3

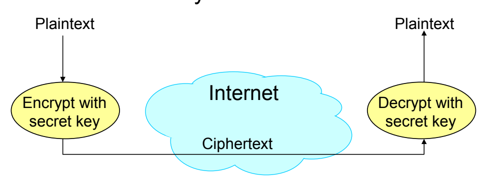
Cryptographic Algorithms

- Security foundation: cryptographic algorithms
  - Secret key cryptography, Data Encryption Standard (DES)
  - Public key cryptography, RSA algorithm
  - Message digest, MD5

4

Symmetric Key

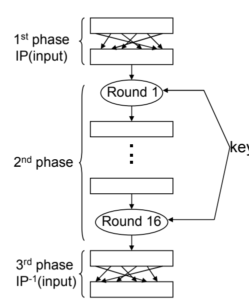
- Both the sender and the receiver use the same secret keys



5

Data Encryption Standard (DES)

- DES encrypts a 64-bit block of plain text using a 64-bit key
- Three phases
  1. Permute the 64 bits in the block
  2. Apply a given operation 16 times on the 64 bits
  3. Permute the 64 bits using the inverse of the original permutation



6

Initial Permutation (IP)

- IP: bit 58 of input becomes 1st bit, bit 50 becomes 2nd bit, etc

58 50 42 34 26 18 10 2 60 52 44 36 28 20 12 4
62 54 46 38 30 22 14 6 64 56 48 40 32 24 16 8
57 49 41 33 25 17 9 1 59 51 43 35 27 19 11 3
61 53 45 37 29 21 13 5 63 55 47 39 31 23 15 7

- IP^{-1}: inverse of IP, e.g., IP(1) = 58, IP^{-1} (58) = 1

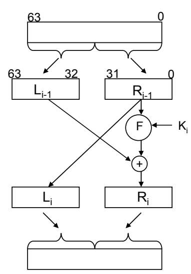
40 8 48 16 56 24 64 32 39 7 47 15 55 23 63 31
38 6 46 14 54 22 62 30 37 5 45 13 53 21 61 29
36 4 44 12 52 20 60 28 35 3 43 11 51 19 59 27
34 2 42 10 50 18 58 26 33 1 41 9 49 17 57 25

2nd Phase: Operation In Each Round

- Key K is 64 bits
- 16 rounds
- Each round i select a 48 bit key K_i from the original 64 bit key K. Perform (F is a given function):

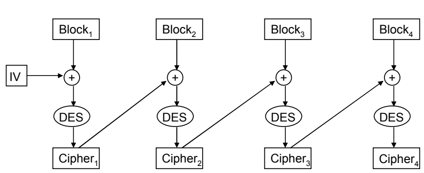
$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$



Encrypting Larger Messages

- Initialization Vector (IV) is a random number generated by sender and sent together with the ciphertext

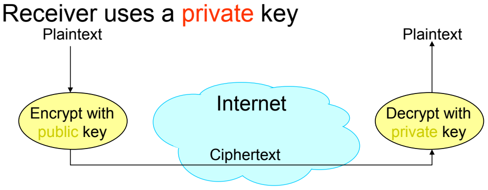


DES Properties

- Confidentiality
  - No mathematical proof, but practical evidence suggests that decrypting a message without knowing the key requires exhaustive search
  - To increase security use triple-DES, i.e., encrypt the message three times

Public-Key Cryptography: RSA (Rivest, Shamir, and Adleman)

- Sender uses a public key
  - Advertised to everyone
- Receiver uses a private key



Generating Public and Private Keys

- Choose two large prime numbers p and q (~ 256 bit long) and multiply them: $n = p*q$
- Chose encryption key e such that e and $(p-1)*(q-1)$ are relatively prime
- Compute decryption key d as $d = e^{-1} \bmod ((p-1)*(q-1))$ (equivalent to $d*e = 1 \bmod ((p-1)*(q-1))$)
- Public key consist of pair (n, e)
- Private key consists of pair (d, n)

RSA Encryption and Decryption

- Encryption of message block *m*:
  - $c = m^e \bmod n$
- Decryption of ciphertext *c*:
  - $m = c^d \bmod n$

Example (1/2)

- Choose p = 7 and q = 11 → n = p*q = 77
- Compute encryption key e: (p-1)*(q-1) = 6*10 = 60 → chose e = 13 (13 and 60 are relatively prime numbers)
- Compute decryption key d such that 13*d = 1 mod 60 → d = 37 (37*13 = 481)

Example (2/2)

- n = 77; e = 13; d = 37
- Send message block m = 7
- Encryption: c = m^e mod n = 7^{13} mod 77 = 35
- Decryption: m = c^d mod n = 35^{37} mod 77 = 7

Properties

- Confidentiality
- A receiver A computes n, e, d, and sends out (n, e)
  - Everyone who wants to send a message to A uses (n, e) to encrypt it
- How difficult is to recover d ? (Someone that can do this can decrypt any message sent to A!)
- Recall that

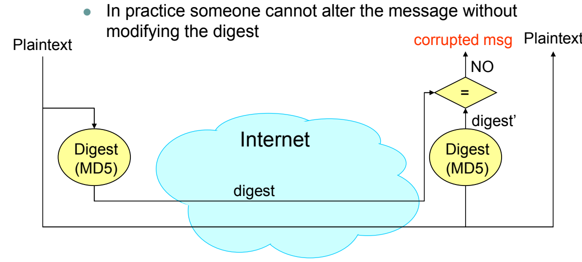
  $d = e^{-1} \bmod ((p-1)*(q-1))$
- So to find d, you need to find primes factors p and q
  - This is provable hard

Message Digest (MD) 5

- Provide data integrity: make sure that message was not altered by a 3rd party
- Idea:
  1) Sender computes a digest of message m, i.e., compute H(m), where H() is a publicly known hash function
  2) Send digest (d = H(m)) to the receiver in a secure way, e.g.,
     - Using another physical channel
     - Using encryption

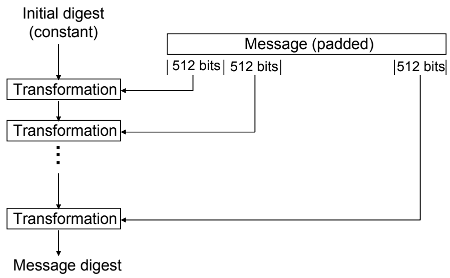
MD 5 (cont'd)

- Basic property: digest operation (i.e., H()) very hard to invert
  - In practice someone cannot alter the message without modifying the digest



Message Digest Operation

- Transformation contains complex operations (see textbook)



Outline

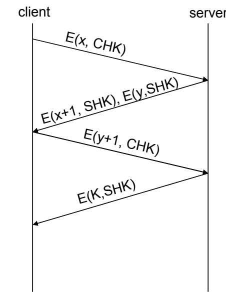
- Cryptographic Algorithms (Confidentiality and Integrity)
- ➢ Authentication
- System examples

Authentication

- Goal: Make sure that the sender an receiver are the ones they claim to be
- Two solutions based on secret key cryptography (e.g., DES)
  - Three-way handshaking
  - Trusted third party
- One solution based on public key cryptography (e.g., RSA)
  - Public key authentication

Simple Three-Way Handshaking

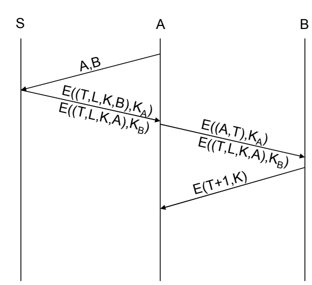
- Client and server share two secret keys: CHK and SHK, respectively
- K – session key used for data communication
  - reduce # of messages containing CHK and SHK
- x, y: nonce (random) values
  - Avoid reply attacks, e.g., attacker impersonating the server
- Notation: E(m,k) – encrypt message m with key k



Trusted Third Party

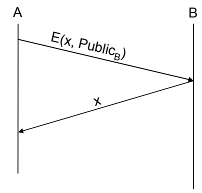
- Trust a third party entity, authentication server
- Scenario: A wants to communicate with B
- Assumption: both A and B share secrete keys with S: K_A and K_B
- Notations:
  - T: timestamp (also serves the purpose of a random number)
  - L: lifetime of the session
  - K: session's key

Trusted Third Party (cont'd)



Public Key Authentication

- Based on public key cryptography
- Each side need only to know the other side's public key
  - No secrete key need to be shared
- A encrypts a random number x and B proves that it knows x
- A can authenticate itself to be in the same way



25

Outline

- Cryptographic Algorithms (Confidentiality and Integrity)
- Authentication
- System examples

26

Public Key Infrastructure (PKI)

- System managing public key distribution on a wide-scale
- Trust distribution mechanism
- Allow any arbitrary level of trust

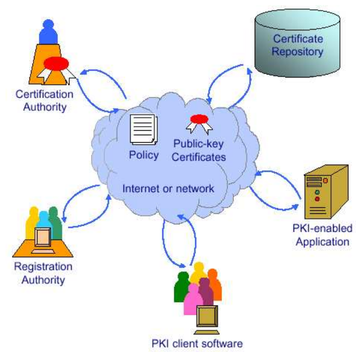
27

PKI Properties

- Authentication → via Digital Certificates
- Confidentiality → via Encryption
- Integrity → via Digital Signatures
- Non–Repudiation → via Digital Signatures

28

Components of a PKI



29

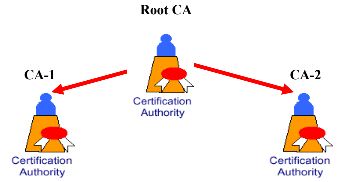
Digital Certificate

- Signed data structure that binds an entity with its corresponding public key
  - Signed by a recognized and trusted authority, i.e., Certification Authority (CA)
  - Provide assurance that a particular public key belongs to a specific entity
- Example: certificate of entity E = $E((name_E, KE_{public}), KCA_{private})$
  - $KCA_{private}$: private key of Certificate Authority
  - KE_{public}: public key of entity E
  - $name_E$: name of entity E

30

Certification Authority

- People, processes responsible for creation, delivery and management of digital certificates
- Organized in an hierarchy



Registration Authority

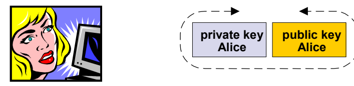
- People, processes and/or tools that are responsible for
  - Authenticating the identity of new entities (users or computing devices), e.g.,
    - By phone, physical presence, etc
  - Requiring certificates from CA's.

Certificate Repository

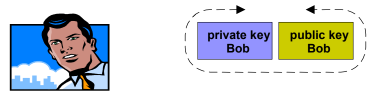
- A database which is accessible to all users of a PKI, contains:
  - Digital certificates,
  - Certificate revocation information
  - Policy information

Example

- Alice generates her own key pair.



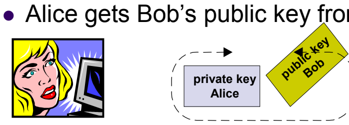
- Bob generates his own key pair.



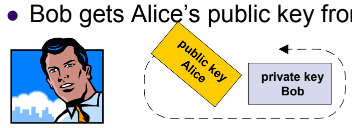
- Both sent their public key to a CA and receive a digital certificate

Example

- Alice gets Bob's public key from the CA

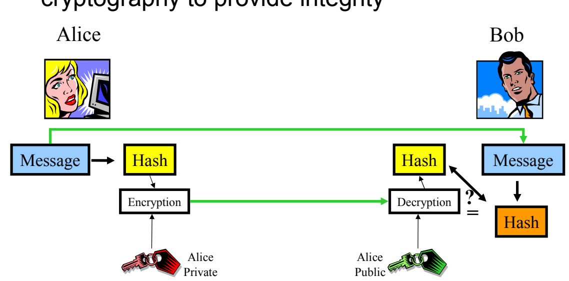


- Bob gets Alice's public key from the CA



Example

- Alice use private key to sign: use public key cryptography to provide integrity



Certificate Revocation

- Process of publicly announcing that a certificate has been revoked and should no longer be used.
- Approaches:
  - Use certificates that automatically time out
  - Use certificate revocation list
  - Use list that itemizes all revoked certificates in an on-line directory

37

What do You Need To Know

- Security requirements
- Cryptographic algorithms
  - How does DES and RSA work
- Authentication algorithms
- Public key management, digital certificates (high level)

38