Protection and Security

Arvind Krishnamurthy
Spring 2004

Introduction

- Types of misuse of computers:
  - Accidental
  - Intentional
- Protection is to prevent either accidental or intentional misuse; security is to prevent intentional misuse
- Four approaches to security: (Denning & Denning)
  - Access controls: Authorization and enforcement (who can do what?)
  - Flow control: no flow from high security to lower security
  - Inference controls: control access to database
  - Encryption and authorization

Authentication

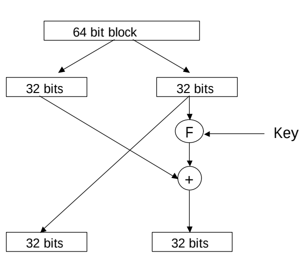
- Common approach: passwords
  - Shared secret between two parties
  - Since only user knows the password, machine can “authenticate”
- Problem 1: system must keep copy of secret to check against user input
  - What if malicious user gains access to this list?
  - What if a copy of the password file is accidentally made/misplaced
- Encryption: transformation that is difficult to reverse without the right key
  - Password → one way transform → encrypted password
  - System stores only encrypted version, so ok even if someone reads the file
  - Even make the encryption algorithm public

Data Encryption Standard

- Encrypts a 64-bit block of plaintext using a 64-bit key
- For passwords:
  - Plaintext is known
  - Key is user password
- DES algorithm steps:
  - Step 1: permute 64-bit block
  - Steps 2-17: Transform block based on key
  - Step 18: reverse permute 64-bit block
- Cannot determine the key just given the plaintext and encrypted version of plaintext
- Can obtain plaintext from encrypted version by applying the reverse algorithm if the key is available

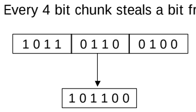
DES Details

- Key is actually only 56 bits long (rest 8 are parity)
- Steps 2-17:



DES Details Contd.

- Function F: takes 2 inputs
  - 32 bit block
  - 56 bit key
- Expands 32 bit block into 48 bits
  - Every 4 bit chunk steals a bit from adjoining chunks



- Shift key (by amount that is round specific), prune it to 48 bits (by dropping certain round specific bits), and permute (in a round specific manner)
- XOR two results, take 48 bit result and construct a 32 bit value by substituting 6 bit chunks with 4 bit chunks using a “substitution table”

DES

- Hard to figure out what the algorithm does!
- Apparently steps 1 and 18 (permutation and reverse permutation) are not so useful
- “Achieves” security by confusion and obfuscation
- Given the plaintext and encrypted text, have to try 2^56 combinations to find password that is used as the key
- How long to perform a single DES?
  - In 1975, about 10ms
  - Now it costs about 1us

DES for large blocks of text

- Referred to as “cipher block chaining” (CBC)
- Algorithm:
  - Break into 64 bit chunks
  - Plaintext for block j is XORed with cipher-text for block j-1 before running it through DES
  - Cipher-text for non-existent block 0 is generated randomly and is referred to as Initialization Vector (IV)
  - IV is sent along with encrypted data
- Question: why do we need IV?

Password Issues

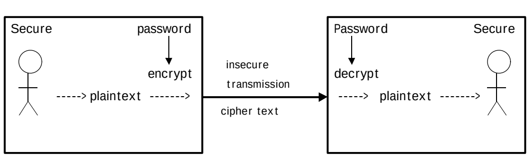
- Typically not necessary to cycle through 2^56 combinations
- Most passwords are:
  - Small, mostly letters
  - Chosen from dictionaries (or some small modifications of it)
  - Exhaustive search is possible
    How long for an exhaustive search? 26^5 = 10 million
    In 1975, 1 day. Now about 10 seconds
- More importantly, an exhaustive search could reveal all the passwords in the entire password file
- Partial solution: extend each password with a unique number (stored in password file), so can’t crack multiple passwords at a time
  - Referred to as “salt”
- Further modifications:
  - Delay all remote login attempts by 1 second
    - Hacker cannot attempt passwords at a fast rate
  - Have password program reject “simple” passwords

Announcements

- Background readings for this material:
  - Unix security paper
  - Data security paper by Denning and Denning

Authentication in Distributed Systems

- Two roles for encryption
  - Authentication
  - Secrecy --- I don’t want anyone to know this data

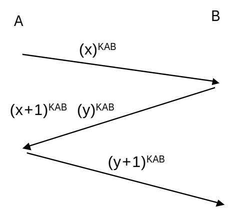


- Guard against:
  - Snooping messages on the network
  - Altering messages (or emitting false material)
  - Replaying messages

Dangers

- Eavesdropper listening to messages over a channel
  - Solution: encryption
- Interloper: someone can inject messages into the network
  - Solution: encryption
- Replaying: save the packets and replay them later
  - Solution: have something unique about each conversation
- Other pieces of security protocols:
  - Trusted servers
  - Signature functions or cryptographic checksums
  - Double encryption

Basic Secret Key Protocol



- KAB – shared key between A and B
- m encrypted by a key K is represented by $(m)^K$
- x and y are random numbers generated by the nodes
- They are sometimes called nonces (use once numbers)
- x and y can then be sequence numbers for future communication

Authentication Server Protocol

- How do you get shared secret in both places? Use **authentication server**
- Main idea: Server keeps list of passwords, provides a way for parties, A and B, to talk to one another, as long as they trust server.
- A asks server for key

  A → S (Hi, I'd like a key for talking between A and B)

- Server gives back special session key encrypted using B's key

  S → A $[\text{ use Kab; } [\text{ This is A! Use Kab}]^{KSB}\ ]^{KSA}$

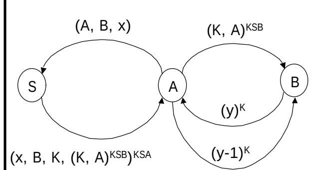
- A gives B the ticket

  A → B $[\text{ This is A! Use Kab }]^{KSB}$

Needham and Schroeder protocol

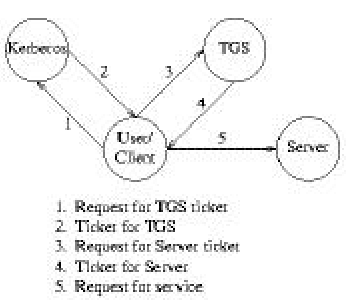
- Goal: obtain a shared key for communication between two nodes
- Bootstrapping: each node has a shared secret (or key) with an authentication server
- One of the nodes communicates with the server to obtain a session key

Needham and Schroeder Protocol



- Initial trust: KSA, KSB
- x, y are nonces to guard against replay
- First message is (A, B, x): which is minimal amount of information
- Server sends back x to guarantee freshness, B to guarantee original message was uncorrupted (tradeoff between message size and encryption costs)
- But B has no guarantee regarding the freshness of K (so Kerberos uses timestamps)

Kerberos Protocol



- Ticket contains:
  - Granting authority
  - Name of client
  - Start time of ticket
  - End time of ticket
  - Session key
- Minimize the use of password
- Obtain new ticket when old ticket expires

Public Key Encryption

- Alternative to secret key encryption
- Has strong number theoretic foundations:
  - Cracking public key scheme is as hard as factorization
- Scheme involves a pair of keys
- Message is encrypted by one key and decrypted with the other
  - Symmetric – message encrypted with the second key can be decrypted with the first one
- Typically, one key is made public while the other is kept private
  - Cannot derive the private key from the knowledge of the public one

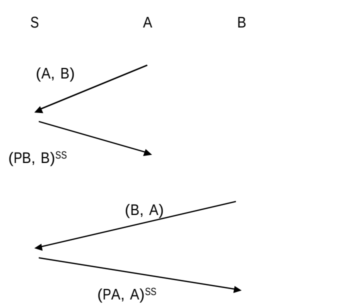
RSA Public Key Algorithm

- Designed by Rivest, Shamir, and Adleman
- With 512 bit keys:
  - Choose two large primes p and q that are roughly 256 bits long
  - Multiply p and q to get N
  - Next choose "e" such that e and (p-1)*(q-1) are relatively prime
  - Finally compute d such that:
    e * d = 1 mod ((p-1)*(q-1))
  - Throw away p and q (do not disclose them)
- Encrypt message m: $c = m^e \bmod n$
- Decrypt: $m = c^d \bmod n$
- Number theoretic property that you get back m
- m needs to be less than n; large messages are treated as concatenation of multiple 512 bit blocks

Public Key Scheme

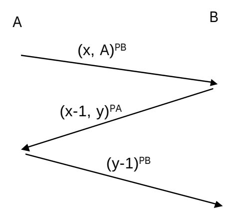
- Properties:
  $[\text{text}]^{KPUB}$ = ciphertext $[\text{ciphertext}]^{KPRIV}$ = text
  $[\text{text}]^{KPRIV}$ = ciphertext' $[\text{ciphertext'}]^{KPUB}$ = text
  KPRIV kept secret, KPUB put in a telephone directory
- Authentication:
  [I will hold office hours tomorrow. $]^{KPRIV}$
  Everyone can read it, but only I can send it!
- Secrecy:
  [Hi, can I get hold of tomorrow's exam questions? $]^{KPUB}$
  Anyone can send it, but only the target can read it
- Secure authenticated communication:
  [[Hi, this is X -- can I get hold of the exam questions? $]^{KPUB}]^{KXPRIV}$
  Only source could have sent it, and only target can read it!

Public Key based Protocols



- Let PA be public key of A and let SA be private key
- To lookup B's public key contact S
- Authentication server returns B's key signed with its private key

Protocol (contd.)



- Both A and B know each other's public key
- Can exchange nonces to begin communicating with each other

Taxonomy of network security