Authentication Protocols

CS 470
Introduction to Applied Cryptography
Ali Aydın Selçuk

Entity Authentication

• Authentication of people, processes, etc.
• Non-cryptographic
  – Address-based (E-mail, IP, etc.)
  – Passwords
  – Biometrics
• Cryptographic
  – Symmetric key
  – Public key

Authentication Tokens

• What you know (password schemes)
• What you have (keys, smart cards, etc.)
• What you are (fingerprints, retinal scans, etc.)

Two-factor authentication: Identification of users by two different components

• Many examples:
  – ATM card & PIN
  – password & mobile phone (by SMS-OTP)
  – fingerprint & PIN

Password Problems

• Eavesdropping
• Stealing password files
• On-line password guessing
• Off-line guessing attacks
  – Dictionary attacks
  – Exhaustive search
• Careless users writing down passwords
Eavesdropping

- Watching the screen
- Watching the keyboard
- Login Trojan horses. Solutions:
  - Different appearance
  - Interrupt command for login
- Keyboard sniffers. Solutions:
  - Good system administration
- Network sniffers. Solutions:
  - Cryptographic protection
  - One-time passwords

On-line Password Guessing

Careless choices (first names, initials, etc.); poor initial passwords

Defenses: After wrong guesses,
- Lock the account
  - Not desirable, can be used for DoS
- Slow down
- Alert users about unsuccessful login attempts
- Don’t allow short or guessable passwords

Off-line Password Guessing

- Stealing & using password files
- Passwords should not be stored in clear. Typically, they’re hashed and stored.
- Attacks:
  - Exhaustive search
  - Dictionary attacks
- Defenses:
  - Don’t allow short/guessable passwords
  - Don’t make password files readable
  - Salting: Mix a random number to each hash
  - More complex hash functions (Normal users will not notice the difference, but attackers will be slowed down significantly.)

Salting

- Salting: Mixing a random number to each hash
- Store <username, rand, H(pwd,rand)> and use rand to hash the input password at each login.
  - Why? How does this help to slow down the attack?
- Several target passwords cannot be tried at once. (Say, there are 20,000 entries in the password file of a university…)
- Also, precomputed password tables cannot be used.
More Complex Hash Functions

- **PBKDF2**
  - Standard by RSA Labs, PKCS #5 v2.0.
  - Repeats a given hash function for a given number of times (minimum 1000, as recommended in year 2000).
  - Used in WPA/WPA2 (with 4096 iterations).
  - Can be attacked efficiently by GPU or FPGA.

- **bcrypt**
  - Based on Blowfish cipher (has a slow key schedule).
  - Uses a large internal table, needs 4 KB of fast RAM.
  - Resilient against GPUs, which are not good at making a lot of memory accesses in parallel.

- **scrypt**
  - bcrypt is resilient against GPUs but not so much against FPGAs. FPGA chips have a lot of small embedded RAM blocks.
  - scrypt allows to increase the memory usage as desired.
  - Resilient against FPGAs as well as GPUs.

Cryptographic Authentication

- Password authentication subject to eavesdropping

- Alternative: Cryptographic challenge-response
  - Symmetric key
  - Public key

Symmetric Key Challenge-Response

An example protocol:

- Alice sends Bob a challenge \( R \).
- Bob computes \( F(K_{AB}, R) \).

- \( F \) is either:
  - block cipher (how?)
  - hash function (how?)

- What about a stream cipher?! (As in WEP)
Mutual Authentication

Both Alice and Bob authenticate each other

An example protocol:

Alice

I'm Alice

R1

F(K_{AB}, R1)

R2

F(K_{AB}, R2)

Bob

Some saving:

Alice

I'm Alice, R2

R1, F(K_{AB}, R2)

F(K_{AB}, R1)

Bob

Reflection attack:

Trudy

I'm Alice, R2

R1, F(K_{AB}, R2)

F(K_{AB}, R1)

Bob

Trudy

I'm Alice, R1

R3, F(K_{AB}, R1)

Bob

• Solutions:
  – Different keys for Alice and Bob
  – Formatted challenges, different for Alice and Bob

• Principle: Initiator should be the first to prove its identity
Another weakness: Trudy can do dictionary attack against $K_{AB}$ acting as Alice, without eavesdropping.

Solution against both problems:

(Dictionary attack still possible if Trudy can impersonate Bob.)

By signature:

By decryption:

• Problem: How can the public/private keys be remembered by ordinary users?
  – Keys can be stored in an electronic token (USB), or can be retrieved from a server with password-based authentication & encryption.

• Problem: Bob (or Trudy) can get Alice to sign/decrypt any text he chooses!
  – Never use the same key for different purposes (e.g., for login and signature)
  – Have formatted challenges
Nonces

- **Nonce**: Something created for one particular occasion
- Nonce types:
  - Random numbers
  - Timestamps
  - Sequence numbers
- Random nonces: if unpredictability is needed
- Timestamps: require syn. clocks
- Obtaining random nonces from timestamps: Encrypt/hash the timestamp with a secret key.
- Seq.no.: Fine if predictability is not a problem