Segurança de Sistemas e dados (MSI 2021/2022)

Aula 1

Rolando Martins
DCC – FCUP
Slides Adaptados do Prof. Manuel Eduardo Correia

Authentication

User Authentication

* fundamental security building block
  * basis of access control & user accountability
* is the process of verifying an identity claimed by or for a system entity
* has two steps:
  * identification - specify identifier
  * verification - bind entity (person) and identifier

Means of User Authentication

* **four means of authenticating user's identity**
* based on something the individual
  1) **knows** - e.g. password, PIN
  2) **possesses** - e.g. key, token, smartcard
  3) **is** (static biometrics) - e.g. fingerprint, retina
  4) **does** (dynamic biometrics) - e.g. voice, sign
* can use alone or combined
* all can provide user authentication
* all have issues

Passwords

Password Authentication

* widely used user authentication method
  * user provides name/login and password
  * system compares password with that saved for specified login
* authenticates ID of user logging and
  * that the user is authorized to access system
  * determines the user's privileges
  * is used in discretionary access control

Trouble with Passwords

* “Passwords are one of the biggest practical problems facing security engineers today.”
* “Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations. (They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed.)”

Password Vulnerabilities

* offline dictionary attack
* specific account attack
* popular password attack
* password guessing against single user
* workstation hijacking
* exploiting user mistakes
* exploiting multiple password use
* electronic monitoring

Why Passwords?

* Why is “something you know” more popular than “something you have” and “something you are”?
* **Cost**: passwords are free
* **Convenience**: easier for admin to reset pwd than to issue a new thumb

Countermeasures

* stop unauthorized access to password file
* intrusion detection measures
* account lockout mechanisms
* policies against using common passwords but rather hard to guess passwords
* training & enforcement of policies
* automatic workstation logout
* encrypted network links

Password File?

* Bad idea to store passwords in a file
* But we need to verify passwords
* Cryptographic solution: **hash** the pwd
  * Store $y = h(\text{password})$
  * Can verify entered password by hashing
  * If Trudy obtains "password file," she does not obtain passwords
* But Trudy can try a *forward search*
  * Guess x and check whether $y = h(x)$

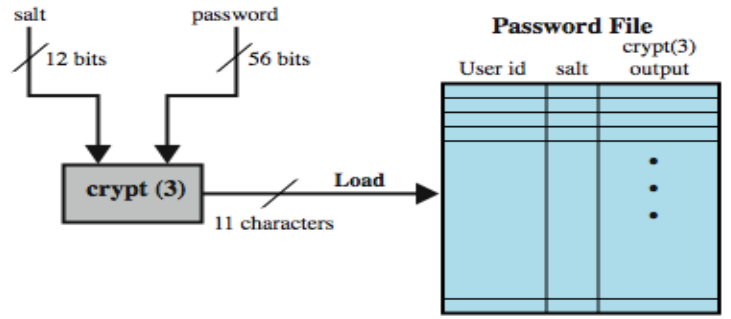
Dictionary Attack

* Trudy pre-computes $h(x)$ for all x in a **dictionary** of common passwords
* Suppose Trudy gets access to password file containing hashed passwords
  * She only needs to compare hashes to her pre-computed dictionary
  * After one-time work, actual attack is trivial
* Can we prevent this attack? Or at least make attacker's job more difficult?

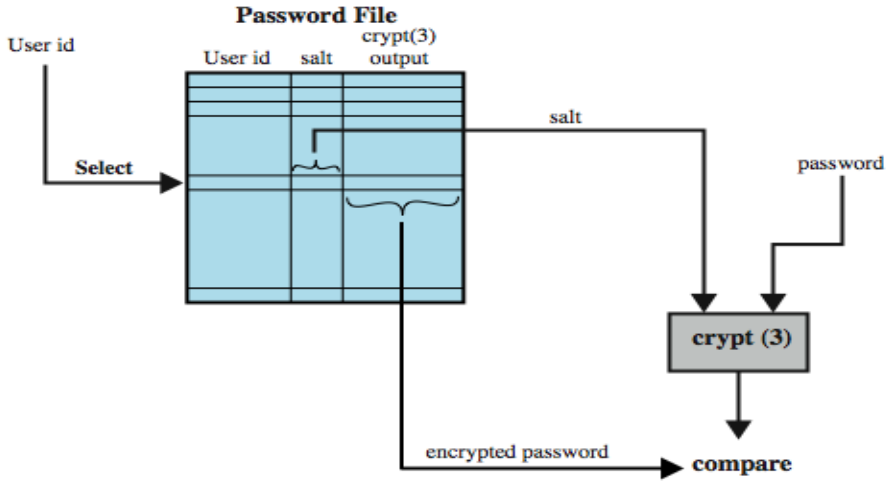
Salt

* Hash password with **salt**
* Choose random salt s and compute
$$y = h(\text{password}, s)$$
and store (s,y) in the password file
* Note: The salt s is not secret
* Easy to verify salted password
* But Trudy must re-compute dictionary hashes for each user
  * Lots more work for Trudy!

Use of Hashed Passwords



(a) Loading a new password



(b) Verifying a password

Figure 3.1 UNIX Password Scheme

UNIX Implementation

* original scheme
  * 8 character password form 56-bit key
  * 12-bit salt used to modify DES encryption into a one-way hash function
  * 64 bit block of Ⓞ (zero) repeatedly encrypted 25 times
  * output translated to 11 character sequence
* now regarded as woefully insecure
  * e.g. supercomputer, 50 million tests, 80 min
* sometimes still used for compatibility

Improved Implementations

* have other, stronger, hash/salt variants
* many systems now use MD5/SHA-1/SHA-2
  * with 48-bit salt
  * password length is unlimited
  * is hashed with 1000 times inner loop
  * produces 128-bit hash
* OpenBSD uses Blowfish block cipher based hash algorithm called Bcrypt
  * uses 128-bit salt to create 192-bit hash value

Password Cracking

* dictionary attacks
  * try each word then obvious variants in large dictionary against hash in password file
* rainbow table attacks
  * pre-compute tables of hash values for all salts
  * a mammoth table of hash values, aka rainbow tables
  * **e.g. 1.4GB table cracks 99.9% of alphanumeric Windows passwords in less then 5 secs**
  * not feasible if larger salt values used

Users…

Sigarra: S!9@rr@
info: uW!5H
banco: $$$$$$

Keys vs Passwords

Crypto keys

* key is 64 bits
* Then 2^{64} keys
* Choose key at random…
* …then attacker must try about 2^{63} keys

Passwords

* passwords are 8 characters, and 256 different characters
* Then $256^8 = 2^{64}$ pwds
* Users do not select passwords at random
* Attacker has far less than 2^{63} pwds to try (**dictionary attack**)

Password Choices

* users may pick short passwords
  * e.g. 3% were 3 chars or less, easily guessed
  * system can reject choices that are too short
* users may pick guessable passwords
  * so crackers use lists of likely passwords
  * e.g. one study of 14000 encrypted passwords guessed nearly 1/4 of them
  * would take about 1 hour on fastest systems to compute all variants, and only need 1 break!

Good and Bad Passwords

* Bad passwords
  * frank
  * Fido
  * password
  * 4444
  * Pikachu
  * 102560
  * AustinStamp
* Good Passwords?
  * jf1ej,43j-EmmL+y
  * 09864376537263
  * P0kem0N
  * FSa7Yago
  * 0nceuP0nAt1m8
  * PokeGCTall150

Password Experiment

- Three groups of users — each group advised to select passwords as follows
  - **Group A:** At least 6 chars, 1 non-letter
  - winner → **Group B:** Password based on passphrase
  - **Group C:** 8 random characters
- Results
  - **Group A:** About 30% of pwds easy to crack
  - **Group B:** About 10% cracked
    - Passwords easy to remember
  - **Group C:** About 10% cracked
    - Passwords hard to remember

Password Experiment

* User compliance hard to achieve
* In each case, 1/3rd did not comply
  * And about 1/3rd of those easy to crack!
* Assigned passwords sometimes best
* If passwords not assigned, best advice is…
  * Choose passwords based on passphrase
  * Use pwd cracking tool to test for weak pwds
* Require periodic password changes?

Attacks on Passwords

* Attacker could…
  * Target one particular account
  * Target any account on system
  * Target any account on any system
  * Attempt denial of service (DoS) attack
* Common attack path
  * Outsider → normal user → administrator
  * May only require **one** weak password!

Password Retry

* Suppose system locks after 3 bad passwords. How long should it lock?
  * 5 seconds
  * 5 minutes
  * Until SA restores service
* What are +'s and -'s of each?

Password File Access Control

* can block offline guessing attacks by denying access to encrypted passwords
  * make available only to privileged users
  * often using a separate shadow password file
* still have vulnerabilities
  * exploit O/S bug
  * accident with permissions making it readable
  * users with same password on other systems
  * access from unprotected backup media
  * sniff passwords in unprotected network traffic

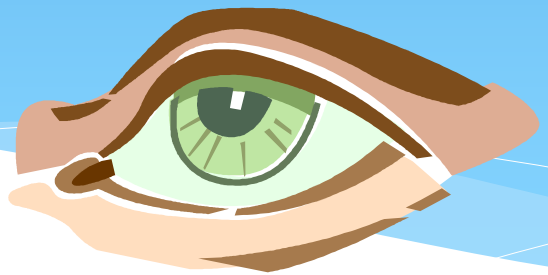
Using Better Passwords

* clearly have problems with passwords
* goal to eliminate guessable passwords
* whilst still easy for user to remember
* techniques:
  * user education
  * computer-generated passwords
  * reactive password checking
  * proactive password checking

Proactive Password Checking

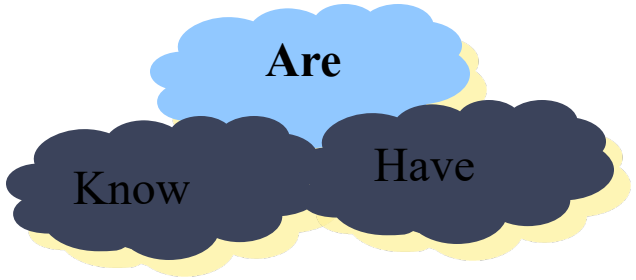
* rule enforcement plus user advice, e.g.
  * 10+ chars, upper/lower/numeric/punctuation
  * may not suffice
* password cracker
  * time and space issues
* Markov Model
  * generates guessable passwords
  * hence reject any password it might generate
* Bloom Filter
  * use to build table based on dictionary using hashes
  * check desired password against this table

Biometrics



Something You Are

* Biometric
  * **"You are your key"** — Schneier
* Examples
  * Fingerprint
  * Handwritten signature
  * Facial recognition
  * Speech recognition
  * Gait (walking) recognition
  * "Digital doggie" (odor recognition)
  * Many more!



Biometric Authentication

* authenticate user based on one of their physical characteristics

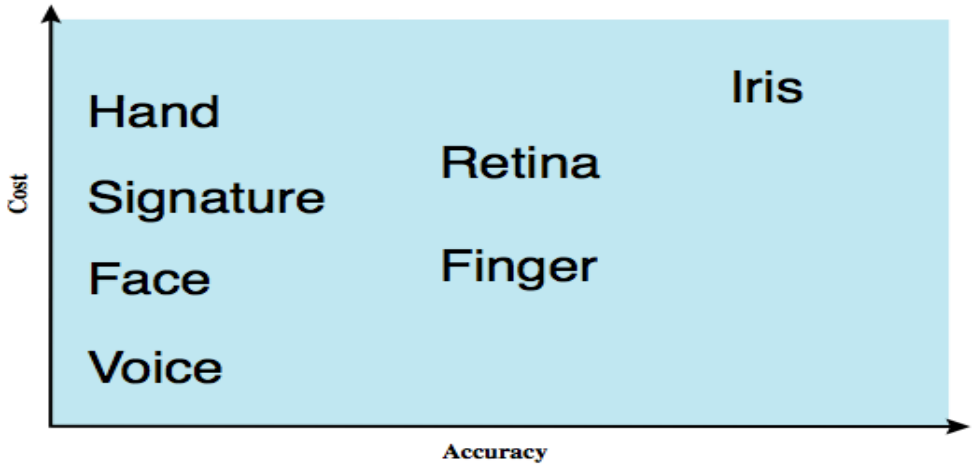


Figure 3.6 Cost versus accuracy of various biometric characteristics in user authentication schemes.

Ideal Biometric

* **Universal** — applies to (almost) everyone
  * In reality, no biometric applies to everyone
* **Distinguishing** — distinguish with certainty
  * In reality, cannot hope for 100% certainty
* **Permanent** — physical characteristic being measured never changes
  * In reality, OK if it to remains valid for long time
* **Collectable** — easy to collect required data
  * Depends on whether subjects are cooperative
* Safe, user-friendly, etc., etc.

Biometric Modes

* **Identification** — Who goes there?
  * Compare **one-to-many**
  * Example: The FBI fingerprint database
* **Authentication** — Are you who you say you are?
  * Compare **one-to-one**
  * Example: Thumbprint mouse
* Identification problem is more difficult
  * More "random" matches since more comparisons
* We are interested in authentication

Cooperative Subjects?

* Authentication — cooperative subjects
* Identification — uncooperative subjects
* For example, facial recognition
  * Used in Las Vegas casinos to detect known cheaters (terrorists in airports, etc.)
  * Often do not have ideal enrollment conditions
  * Subject will try to confuse recognition phase
* Cooperative subject makes it much easier
  * We are focused on authentication
  * So, subjects are generally cooperative