

---

---

# Passwords

15 June 2010  
Lecture 12

Some Slides Credit: Steve Zdancewic (UPenn)

---

# Topics for Today

---

- Passwords
  - Best practices, attacks

# Human User Authentication

---

- How do you:
  - Know you're talking to a human?
  - Allow a human user to identify him or herself to a machine?
- Machine
  - Good at authenticating other machines
  - Good at mathematical manipulations, etc.
  - Can handle keys, secrets, etc.
  - Very good memory of things stored in it
- Humans
  - Good at identifying people
  - Use small clues that when combined yield an unmistakable picture
    - Voice
    - Height
    - Stance
    - Shared history

# Identifying Any Human

---

- Problem:
  - How does a machine establish that it's talking to a human?
  - Why?
    - Prevent SPAM, abuse of web accounts, foil bots and web crawlers,...
- Answer: Challenge / Response
  - Challenge is something that only humans can do (quickly):
  - Example: deciphering obscured text



- Read: "Telling Humans and Computers Apart" (von Ahn, Blum, and Langford) [www.captcha.net](http://www.captcha.net)
- Counter strategies:
  - 'Grandmaster chess attack' : get humans to do the decoding

# Identifying a particular human

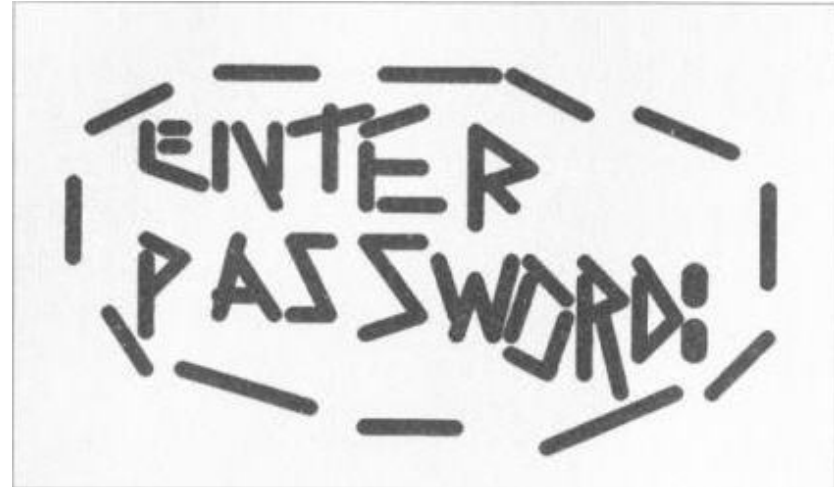
---

- Human Authentication is based on one or more of the following:
  - **Something you know**
    - password
  - **Something you have**
    - driver's license, Student Card
  - **Something inherent about you**
    - Biometrics, location

# Passwords

---

- Shared code/phrase
- Client sends to authenticate
- Simple, right?
- How do you...
  - Establish them to begin with?
  - Stop them from leaking?
  - Stop them from being guessed?



SOURCE: NASA

# Prime Mover Problem

---

- Out of band
  - Physical mail
  - Email
  - Attached to the box
- Piggybacking
  - Swipe ID Card to make Password
  - But where does the chain stop?
    - ID Card -> TZ-> birth certificate

# Leaks & Challenges

---

- Social engineering
- Managing large numbers of passwords:
  - Writing the password down on paper
  - Storing it in an electronic "safe"
  - Using a web browsers 'remember this password' feature
- Legal and responsibility
  - Shared password == shared liability

# Guessing

---

- The "no such user" mistake
  - Gives an attacker information about usernames
- The "here's who we are" mistake
  - Gives an attacker information about who might have an account
- Common words, phrases for passwords
- Null passwords, "password", username, backwards, etc.
- Dictionary attacks
  
- How bad is it?

# 1979 Survey of 3,289 Passwords

---

- With no constraints on choice of password, Morris and Thompson got the following results:
  - 15 were a single ASCII letter.
  - 72 were strings of two ASCII letters.
  - 464 were strings of three ASCII letters.
  - 47 were strings of four alphanumerics.
  - 706 were five letters, all upper-case or all lower-case.
  - 605 were six letters, all lower case.

# Other Surveys of Passwords

---

- Klein (1990) and Spafford (1992) of 15K passwords
  - 2.7% guessed in 15 minutes
  - 21% in a week
  - Sounds ok? Not if the passwords last 30 days
- Schneier (2006) survey of 34,000 MySpace passwords
  - 65% eight characters or less
  - 28% lower case letters followed by a single digit
  - Top 11 passwords: *password1*, *abc123*, *myspace1*, *password*, *blink182*, *qwerty1*, *f\*\*\*you*, *123abc*, *baseball1*, *football1*, *123456*
  - 23% could be cracked in 30 minutes
  - 55% in eight hours
- Tricks
  - Letter substitutions, words backwards, common names, patterns, etc.
  - Anything you can think of off the top of your head, a hacker can think of too
- Lazy users!
  - Weakest link is always the way of the attack

# Heuristics for Guessing Attacks

---

- The dictionary with the words spelled backwards
- A list of first names (best obtained from some mailing list). Last names, street names, and city names also work well.
- The above with initial upper-case letters.
- All valid license plate numbers in your state. (About 5 hours work in 1979 for New Jersey.)
- Room numbers, social security numbers, telephone numbers, and the like.

# What makes a good password?

---

- Password Length
  - 64 bits of randomness is hard to crack
  - 64 bits is roughly 20 “common” ASCII characters
  - But... People can’t remember random strings
  - Longer not necessarily better: people write the passwords down
- Pass phrases
  - English Text has roughly 1.3 random bits/char.
  - Thus about 50 letters of English text
  - Hard to type without making mistakes!
- In practice
  - Non-dictionary, mixed case, mixed alphanumeric
  - Not too short (or too long) 8 - 12 characters
  - Tools that check password strength
    - <http://www.microsoft.com/protect/yourself/password/checker.msp>
    - <http://www.fastcrack.com/pwcheck.html>

# Hacks on plaintext password file

---

- Is the password file readable by the OS?
  - Then if I break the OS
- Can privileged users see the file?
  - ... and make copies
- Is the file backed up somewhere
  - ... insecure?
- Is the file/password in plaintext somewhere in memory?
  - Core dump
- Fool the user
  - A program that masquerades as the authentication program

# Counter-hacks

---

- Control-Alt-Del for logging in
  - Establishes a "trusted path" in hardware
  - Prevents trojan horses from intercepting passwords
- Slow down / restrict number of tries
  - Make guessing take too long
  - e.g. 3 tries and you're blocked for 30 seconds
- Encrypt the password file
  - "Salt" - to prevent duplicates
  - Use one way hashes or encryptions on the passwords

# Add Salt

---

- “Salt” the passwords by adding random bits.
  - Decreases the likelihood that two identical passwords will appear as identical entries in the password file.
- 12 bit salt results in 4,096 versions of each password.
- Unix: /etc/passwd entry:

user_id	salt <sub>u</sub>	Hash(salt <sub>u</sub> + passwd <sub>u</sub> )	...
---------	-------------------	--	-----

- Modern implementations use so-called *shadow* password files /etc/shadow that aren't world readable.

# Conclusion

---

- Passwords
  - Best practices, attacks