ENEE 459-C
Computer Security

Authentication and passwords

UNIVERSITY OF
MARYLAND

Select a Password

- Choose a case-sensitive alphanumeric password
- That is, your password should use the following characters
  - 0123456789
  - abcdefghijklmnopqrstuvwxyz
  - ABCDEFGHIJKLMNOPQRSTUVWXYZ
- Let’s try to crack it!
- http://ophcrack.sourceforge.net/

Why do we need Passwords?

- We need a password to authenticate our identity (who we are) to a system
- Authentication is the act of confirming the truth of an attribute of a datum or entity
- There are three authentication factors:
  - Knowledge : Something the user knows (e.g., a password, or PIN, challenge/response (the user must answer a question), pattern)
  - Ownership: Something the user has (e.g., phone number, ID card, security token, etc.)
  - Inherence: Something the user is or does (e.g., fingerprint, retinal pattern, DNA sequence, voice, etc.).

Storing Passwords

User

Password file

GOD

cryptographic hash function

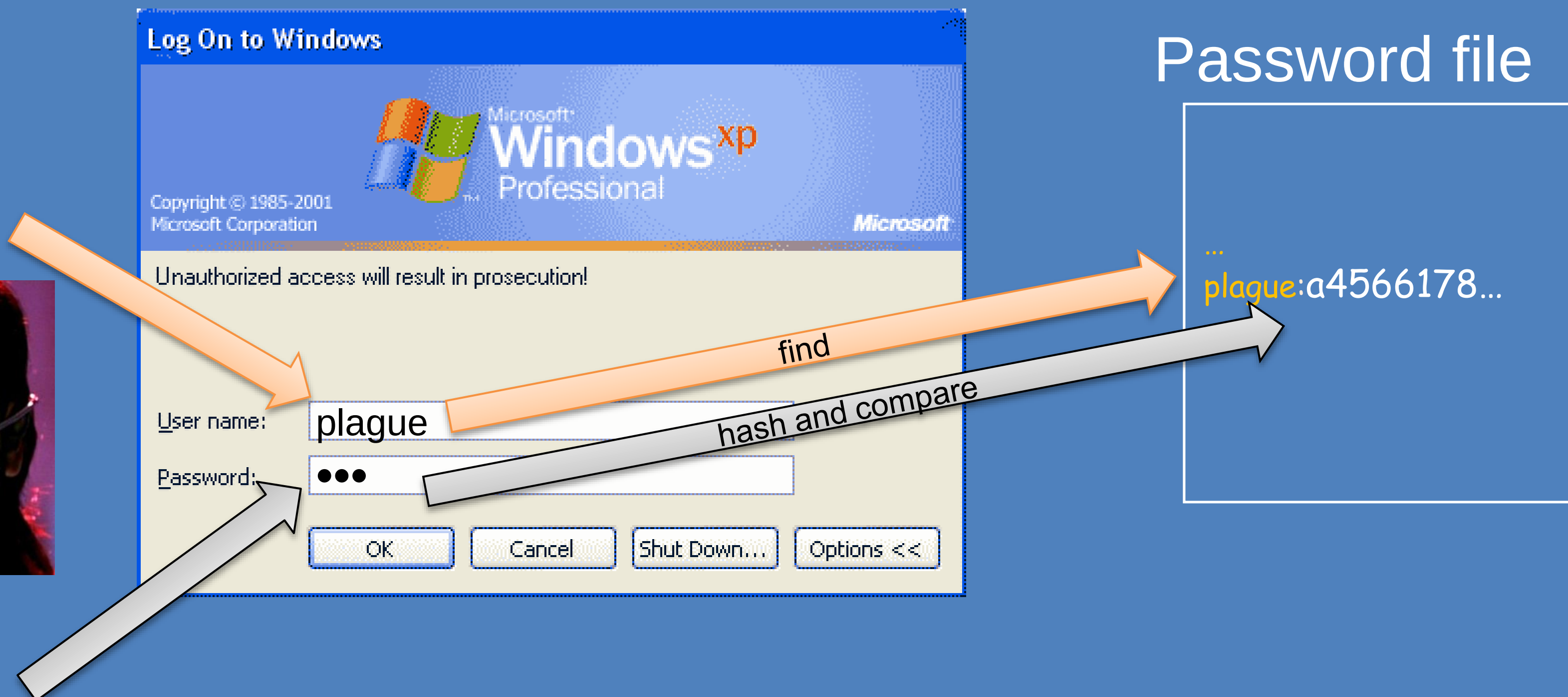
...
plague:a4566178ab4f56a3f90968e210b95bca
zero_cool:2d0f1a836407eb204c0a9186687b713e
...

Password Verification

Logon prompt

plague

GOD



Password file

User inserts username and password

System hashes the password entered by the user and compares it with the stored hash

Where is the Password File?

- Windows (32 bit): C:\WINDOWS\system32\config\SAM
- Linux: /etc/passwd
- Mac OS X: /var/db/shadow/hash/

Strong Passwords

- Long passwords preferred
- Use all available characters
  - UPPER/lower case characters
  - Digits
  - Special characters: &, %, $, £, “, |, ^, §, …
- Which of the following passwords are strong?
  - cpap1
  - john
  - P@$$w0rd
  - TD2k5s@}ecV87^R:@DKlksj298RLO<j;-*h

Password Complexity

- Consider a password with six characters
  - Digits (10)
    10^6 = 1,000,000
  - Lower case characters (26)
    26^6 = 308,915,776
  - UPPER and lower case characters (52)
    52^6 = 19,770,609,664
  - Special characters: &, %, $, @, “, |, ^, <, … (32)
    32^6 = 1,073,741,824
  - Standard keyboard characters (94)
    94^6 = 689,869,781,056
  - All 7-bit ASCII characters
    128^6 = 4,398,046,511,104

Password Length

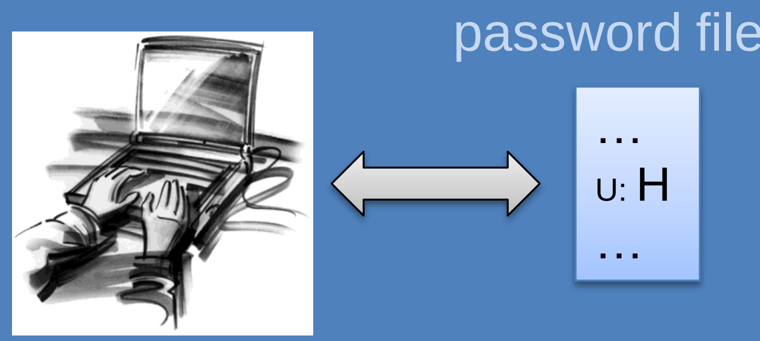
- Assume a standard keyboard with 94 characters

Password length	Number of passwords
5	94^5 = 7,339,040,224
6	94^6 = 689,869,781,056
7	94^7 = 64,847,759,419,264
8	94^8 = 6,095,689,385,410,816
9	94^9 = 572,994,802,228,616,704

Password Salt

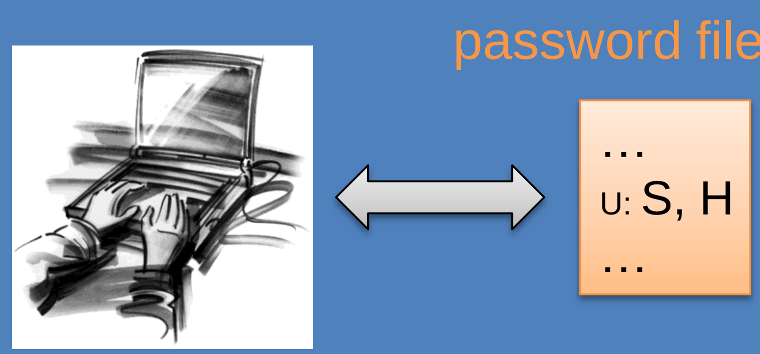
Without salt

1. User types userid, U, and password, P
2. System looks up H, the stored hash of U's password
3. System tests whether h(P) = H



With salt

1. User types userid, U, and password, P
2. System looks up S and H, where S is the random salt for userid U and H is the stored hash of S and U's password
3. System tests whether h(S, P) = H



Random Salting

- Alice and Bob selected same password
  - If not random salted hashes will be the same
  - probably they come from a common word
- Too easy for an hacker to start an attack from equal hash

Alice:root:a483b303c23af34761de02be038fde08

Bobby:root:3282abd0308323ef0349dc7232c349ac

Cecil:root:a483b303c23af34761de02be038fde08

Same Password Same Hash No Salted

Alice:root:b4ef21:3ba4303ce24a83fe0317608de02bf38d

Bobby:root:a9c4fa:3282abd0308323ef0349dc7232c349ac

Cecil:root:209be1:a56456546ffggfkhjrejklhrehytey8

Same Password Different Hash Salted

- Command in Linux to build a password hash:
  - $ mkpasswd --method=SHA-256 –S 12345678

Simple Cracking Methods

- Brute force
  - Try all passwords in a given space
  - Online vs. offline attack
  - Eventually succeeds given enough time and CPU power
- Dictionary
  - Precompute hashes of all passwords in a given space
  - Create search structure indexed by hash value
  - Search for password in the “dictionary”
  - Vast, and possibly unfeasible, space usage and preprocessing time

Brute Force Cracking

- The attacker has 60 days
- How many hash computations per second?
  - 5 characters: 1,415
  - 6 characters: 133,076
  - 7 characters: 12,509,214
  - 8 characters: 1,175,866,008
  - 9 characters: 110,531,404,750