Authentication

Amit Konar
Math and Computer Sc., UMSL

What is Authentication?

Authentication is the binding of an identity to a subject.

Subjects act on behalf of some other, external entity. The identity of that entity controls the actions that its associated subjects may take. Hence, the subjects must bind to the identity of that external entity.

How does an authentication system work?

The authentication process consists of obtaining the authentication information from an entity, analyze the data, and determine if it is associated with that entity.

This means that the computer must store some information about the entity.

Formalization of an Authentication System

An authentication system consists of **5 components:**

1) **Authentication information A**= set of specific information with which entities prove their identities,
2) **Complementary information C**= set of information that the system stores and uses to validate the the authentication information

Authentication: Formalization (Contd.)

3) **Complementation Functions F** that generate the complementary information from the authentic information, i.e.,

f: A $\rightarrow$C.

4) **Set L of authentication functions** that verify identity. That is for *l* ε L,

l: A X C $\rightarrow$**{true, false}.**

5) The set **S of selection functions** that enable an entity **to create or alter the authentication and complementary information**.

Example illustrating the parameters

Suppose, a user authenticates himself by entering his password, which the system compares with the cleartext passwords stored online. Here,

A= set of strings making up acceptable passwords,

C=stored information =A

F= {I}, the identity function as f: A$\rightarrow$A.

L= A X A $\rightarrow$ {true}; so L ={eq}.

Passwords

A **password** is information associated with an entity that confirms the entity's identity.

Example: One installation requires each user to choose a sequence of 10 digits as a password. Then A has 10^{10} elements (0000 0000 00 to 9999 9999 99)

What happens if Complimentary functions and Authentication functions are identical?

Morris and Thompson recount an amusing example.

A Multics system editor swapped pointers to the temporary files being used to edit the password file and the message of the day file. The result is that **whenever a user logged in, the cleartext password file was printed**.

Soln: The solution is to use a one-way-Hash function to hash the passwords into a complement.

Reviewing UNIX password Maintenance Sytem

1) A UNIX password is composed of 8 ASCII characters (NUL is disallowed). Hence, A is the set of strings of 8 characters, each chosen from 127 possible characters. So, A contains 127^8 or 6.9×10^{16} passwords.
2) UNIX hashes the passwords (using one of 4096 functions) into an 11 character string, and two characters that identify the functions used are pre-pended, thus making 13-character strings fore each password. So, C contains 128^{13} or 3.0×10^{23}. strings.

UNIX Password System

3) The subset of C corresponding to selected passwords may/may not be readable. Many UNIX systems store these strings in the file ***/etc/psswd***, which all users can read. Many other versions of the UNIX system, however, store these strings in shadow password files that only the superuser can read.

UNIX password System- A Review

4) The UNIX hashing functions $f \in F$ consists of 4096 hash functions.

5) The UNIX **authentication functions** are **login, su** and other programs **that confirm a user's password during execution**.

6) Some of these functions are accessible over networks through **telnet/ftp protocols**.

Goal of an Authentication System

The goal of authentication system is to ensure that the entities are correctly identified. If one entity guesses another's password, then the guesser can impersonate the other.

The goal here is to find $a \in A$, such that for $f \in F$, $f(a) = c \in C$, and c is associated with a particular entity.

How to achieve this goal?

We can check whether f(a) = c for $a \in A$ and $c \in C$ by **two** approaches: **by computing f(a) or by authenticating by l(a, c).**

So, to **protect passwords** we have two approaches.

1) Hide enough information so that one of a, c and f cannot be found.

2) Prevent access to the authentication functions L.

Attacking a Password System

The simplest attack is to guess passwords.

Dictionary Attack: It is the guessing the passwords by repeated trial and error. The name of this attack came from the list of words used selected from a dictionary for guesses.

Types of Dictionary Attack

Dictionary Attack of Type 1: The dictionary attack takes each guess g, computes f(g) for each $f \in F$, and checks whether f(g) = c.

Dictionary Attack of Type 2: If either of c or f not available, then authentication function I may be used. If the guess g results in I returning true, g is the correct password. This is called dictionary attack of type 2.

Countering Password Guessing

The goal of defenders is to maximize the time needed to guess the password. **Anderson's formula** provides a basis to this.

Let

P= the probability that an attacker guesses a password in a specified period of time,

G= no. of guesses that can be tested in one time unit,

T= number of time units during which guessing occurs,

N= no. of possible passwords.

Then,

P >= TG/N

Remodeling Anderson's Formula

Let

R= no. of bytes per minute that can be sent over a communication line,

E= no. of characters exchanged when logged in,

S= length of a password,

A= no. of characters in the alphabet from which the characters of the passwords are drawn.

Then no. of possible passwords N = A^S.

no. of guesses per minute is G= R/E.

If the period of guessing extends M months (or 4.32×10^4 M minutes), then

$$P >= 4.32 \times 10^4 M (R / E) / A^S,$$

$$\text{Or, } A^S >= 4.32 \times 10^4 M. R/P. E$$

Minimum Length of Password to have a successful guess with Probability 0.5

Suppose, passwords are composed from an alphabet set of 96 characters. Assume that 10^4 guesses can be tested each second. We wish the probability of a successful guess to be 0.5 over a 365-day period. What is the minimum password length that will give us this probability?

Here, N >= T.G/P

$=(365 \times 24 \times 60 \times 60)\ 10^4 / 0.5$

$= 6.31 \times 10^{11}$.

We should choose an integer S such that

$$\sum_{i=1}^{S} 96^i >= N = 6.31 \times 10^{11},$$

Which yields **S>=6**. So, the length of the passwords should be at least 6 characters.

Biometrics in System Authentication

Biometrics is the automated measurement of biological or behavioral features that identify a person.

When a user is given an account, the system administration takes a set of measurement that identify that user to an acceptable degree of error.

Types of Biometrics

Fingerprints

Voices

Iris and retina

Faces

DNA samples