CS443 – Authentication & Access Control

Note Title 1/16/2013

Announcements

- HW2 due next Tuesday
- Next lab – up Friday

Recap of reading

Access Control

The prevention of unauthorized use of a resource, including the prevention of use in an unauthorized manner.

Probably the central element of computer security.

Access Control incorporates:

1. Authentication
2. Authorization
3. Audit (later)

1) Authentication

4 basic strategies:

1) Something you know : password
2) Something you possess
3) Something you are
4) Something you do

Which is most common?

Passwords : Common Attacks

- Brute force
  (dictionary attacks)
- password sniffing
  (key logger)
- shoulder surfing
- social engineering
- sophisticated attacks
-

Defenses against password attacks

- Guard against bad passwords
- Limit guesses
- Pause unnecessarily
- Capture - recognize text
- Recognizable pictures / passcodes

Hashed Passwords

In general, only hashed versions of passwords are saved.

Why?
- Access to file compromise entire system
- Extra layer of protection

Unix Implementation

- User password of 8 digits
  → 56-bit value
- 12-bit salt value, usually based on account creation time
- Hash function (based on DES) is run ~25 times.
- Resulting 64-bit value is converted to 11-character sequence

Sounds impressive...

In 2003, a supercomputer managed over 500 million password guesses in 80 minutes.

(Back then, a regular machine could have done the same in a month or so.)

Stronger variants of password verification essentially use stronger & slower hashing algorithm.

(One even just runs a dummy for loop!)

Single Most Important Defense:

User education!

- choose secure passwords, since dictionary attacks are first effort.
  - random password
  - recommended techniques:
    common base
    incorporate elements from web page

Password checkers

Algorithms that allow or reject passwords based on how likely they are to be cracked.

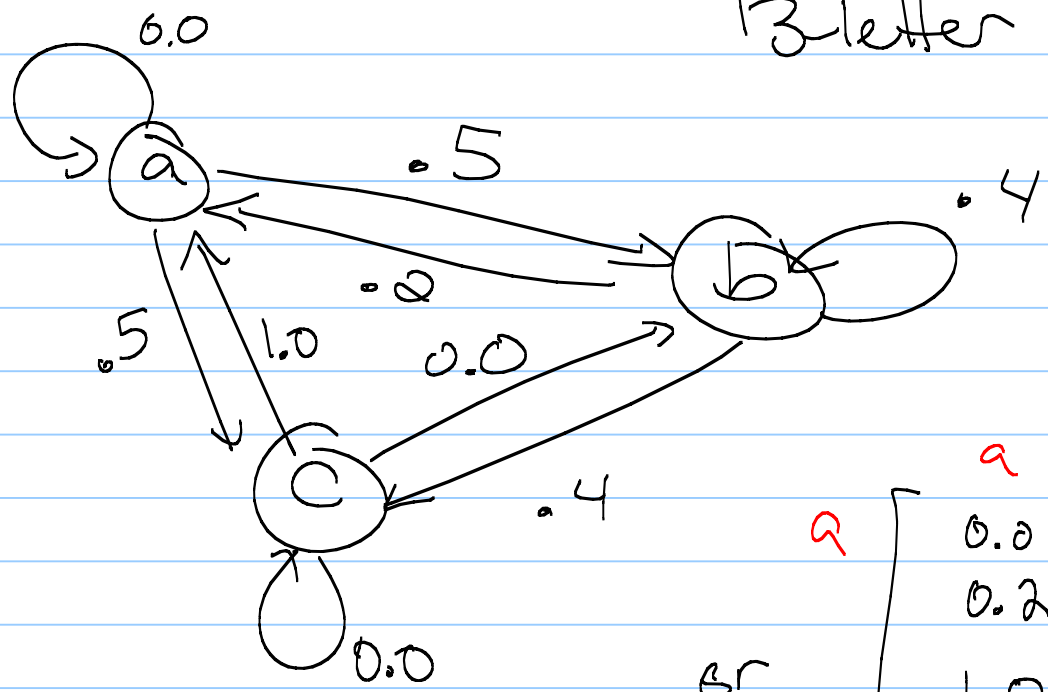
① Rule enforcement:

- correct # of characters
- at least one number

⋮

✱ - add some dictionary checking ⟶

② Markov model: simple version with 3-letter alphabet



$$\underline{or} \quad \begin{array}{c|ccc} & a & b & c \\ \hline a & 0.0 & 0.5 & 0.5 \\ & 0.2 & 0.4 & 0.4 \\ & 1.0 & 0.0 & 0.0 \end{array} \quad \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

② (cont)

For English, they start with a dictionary of passwords

Transitions are based on how common small letter sequences are.

Prev ex: $\frac{\#\text{strings with 'a'}}{\#\text{strings with "ab"}} = .5$

first order model

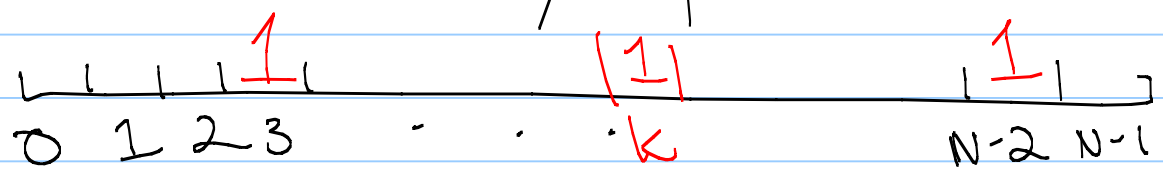
Model catches most dictionary passwords, but still user friendly.

③ Bloom filters

Start with dichonary of passwords to avoid.

Take k independent hash functions.

Hash all dichonary passwords:



$H_1(\text{"secret"}) = 3$

$H_2(\text{"secret"}) = N-2$

$H_3(\text{"secret"}) = k$

$\vdots$

3 (cont)

When a new password is given, its k hash values are all computed. If all = 1 in hash table, it is rejected.

Note:

If rejected, doesn't mean it is in dictionary.

③ (cont)

Math is beyond this class, but with "good" hash functions,

$$P[\text{false positive}] \approx \left(1 - e^{\frac{kD}{N}}\right)^k$$

rejected but should not have been

k = # hash functions
N = # bits in hash table
D = # words in dictionary

Why use Bloom filters?

Simple example: dictionary of 1 million words, so takes ~ 8 MB.

Suppose we want a .01 probability of rejecting a password not in the dictionary.

If we want 6 hash functions, then need $\frac{N}{D} = 9.6$

$\Rightarrow$ Hash table of 9.6×10^6 bits, or 1.2 MB.

Saves space <u>and</u> <u>time</u>.

2 Token-Based Authentication

(something you possess)

Examples:

- cell account (to get message)
- badges
- access cards
- RSA tokens
- ATM cards ...

Attacks:

- Theft
- Duplication

Biometric Authentication

(Something you are or do)

- Hard to steal
- Expensive
- People change → hard to make effective
- Possible (if not easy) to fool

A Note About Remote Authentication

<u>Goal</u>: Give eavesdroppers as little info as possible.

Sample (& simple) protocol:

1) user transmits identity
2) host sends a nonce (random #, r) and specifies 2 functions f and h
3) user sends: $f(r, h(\text{password}))$

avoid repeat attack

2 Authorization: Access Control Policies

Major policy structures:

A) Discretionary Access Control

B) Mandatory Access Control

C) Role-Based Access Control

(These aren't necessarily mutually exclusive, either.)

Terminology

- Subject: a process or user
  3 classes:
  - owner
  - group
  - world

- Object: a resource

Dfn: Access rights describe ways which subjects may interact with objects.

Discretionary Access Control (DAC)

- Most common in modern OS
- Based on subject's identity combined with access rights stating what each subject is allowed to do.

Note: An entity may be given access rights which allow it to give another subject access rights.

Access Control Matrix:
DAC model developed by Lampson in '71:

		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

Image taken from course text, with permission

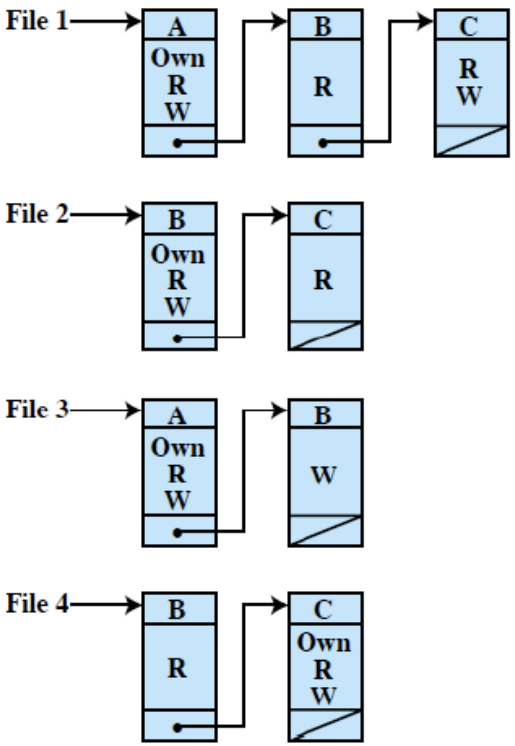
How to implement?

In practice, this matrix tends to be <u>very</u> sparse.

(Think of the number of files & users on our linux systems, much less in larger labs.)

So saving it as a matrix is a waste of memory.

Windows: Access Control Lists



Good:

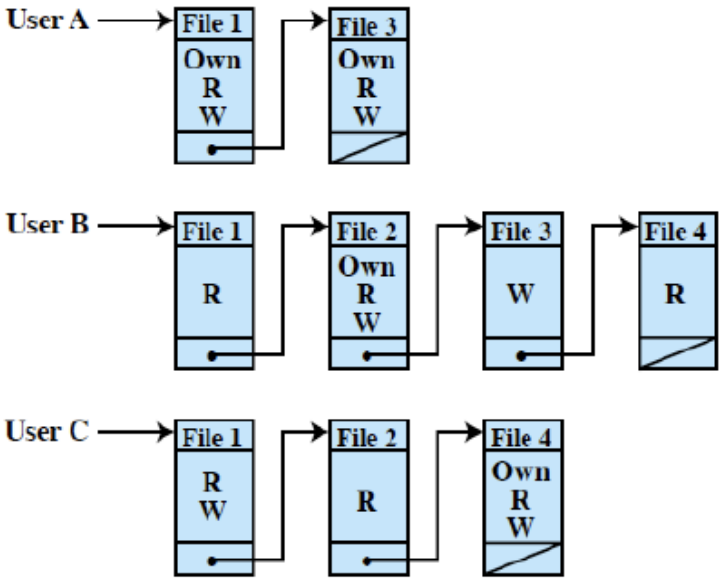
Space efficient.

Given a file, easy to check if user is in list.

Bad:

Hard (& slow) to list all files a user can access.

Capability lists: reverse the previous implementation



Good:

Bad:

Reverse of previous situation

Mandatory Access Control (MAC)

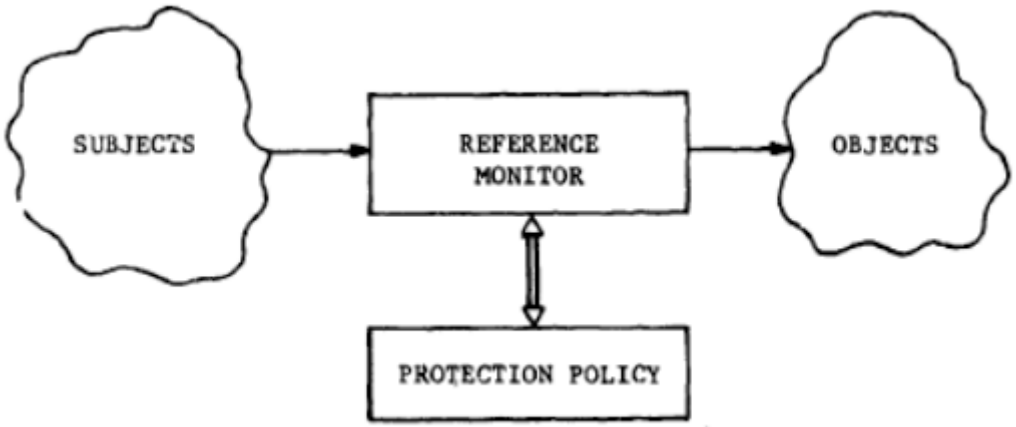
Based on comparing security labels with security clearances.

Mandatory: a subject with access to some resource may not share access with another subject

General use: government & military

Since the 1960's, DoD (& other agencies) have been employing people to develop MAC policies

Ex: Biba ↴



(We'll see more of these later)

Role-Based Access Control (RBAC)

Access rights are based on what roles the user assumes in the system, rather than the user's identity.

Roles may own or control other roles, as well as files or directories.

RBAC is the "hot new thing":

> RBAC is the newest category of access control; it enjoys "widespread commercial use and remains an area of **active research**"
>
> -- Stallings & Brown

Example of RBAC: Medical practitioners