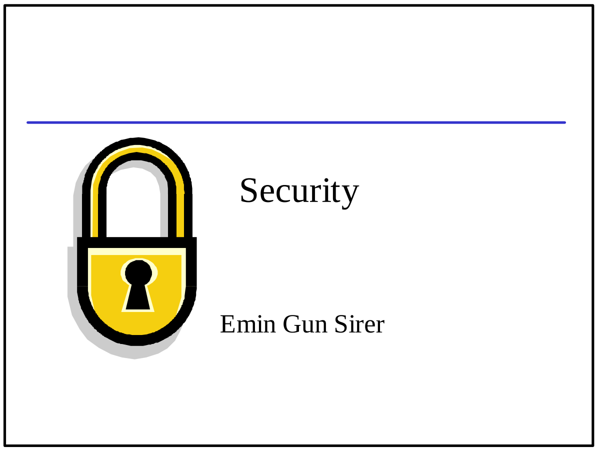
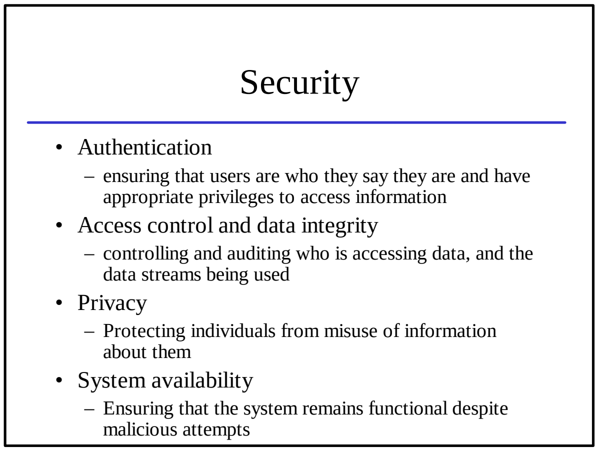


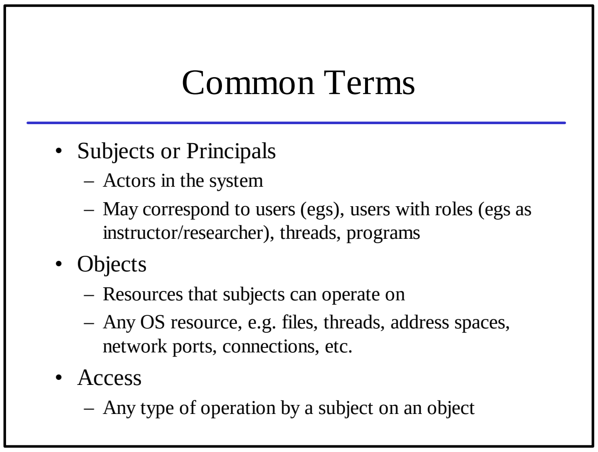
Security

Emin Gun Sirer



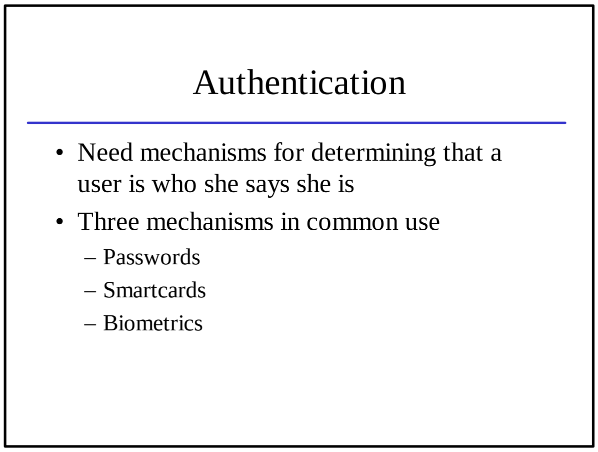
Security

- Authentication
  - ensuring that users are who they say they are and have appropriate privileges to access information
- Access control and data integrity
  - controlling and auditing who is accessing data, and the data streams being used
- Privacy
  - Protecting individuals from misuse of information about them
- System availability
  - Ensuring that the system remains functional despite malicious attempts



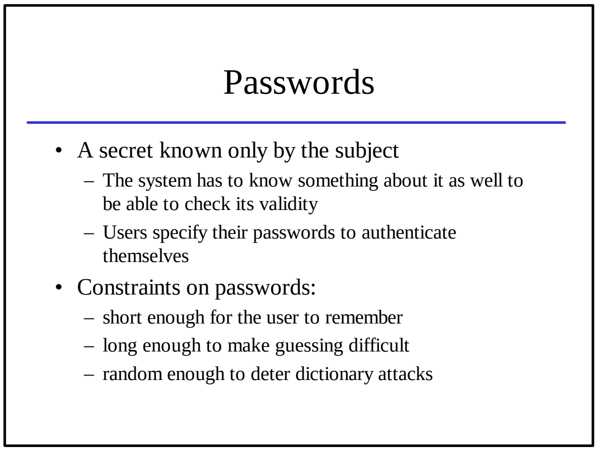
Common Terms

- Subjects or Principals
  - Actors in the system
  - May correspond to users (egs), users with roles (egs as instructor/researcher), threads, programs
- Objects
  - Resources that subjects can operate on
  - Any OS resource, e.g. files, threads, address spaces, network ports, connections, etc.
- Access
  - Any type of operation by a subject on an object



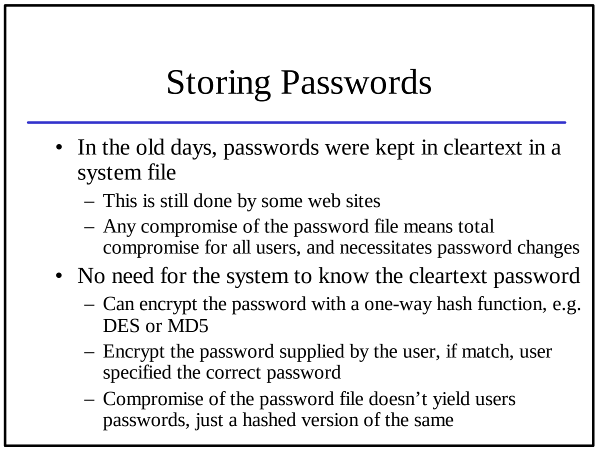
Authentication

- Need mechanisms for determining that a user is who she says she is
- Three mechanisms in common use
  - Passwords
  - Smartcards
  - Biometrics



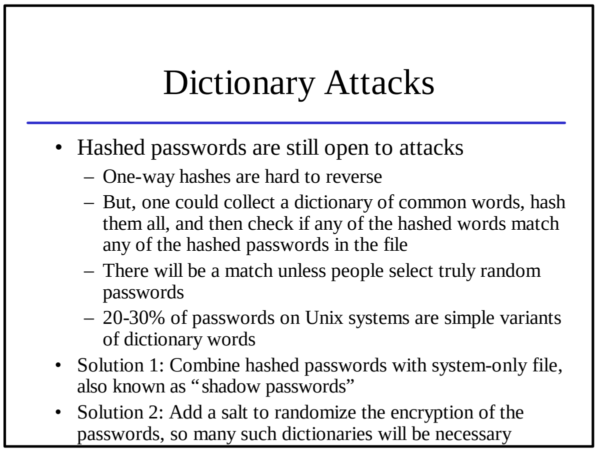
Passwords

- A secret known only by the subject
  - The system has to know something about it as well to be able to check its validity
  - Users specify their passwords to authenticate themselves
- Constraints on passwords:
  - short enough for the user to remember
  - long enough to make guessing difficult
  - random enough to deter dictionary attacks



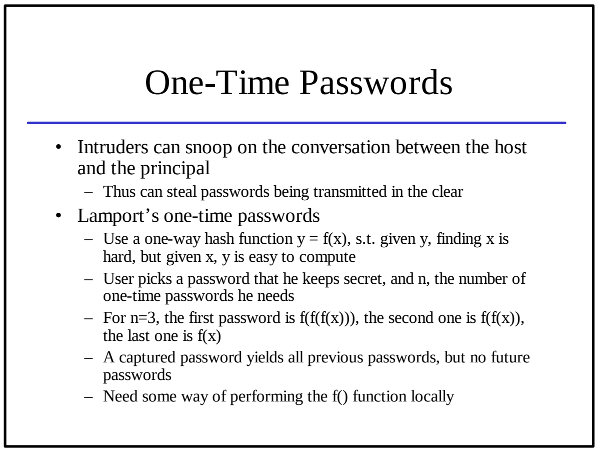
Storing Passwords

- In the old days, passwords were kept in cleartext in a system file
  - This is still done by some web sites
  - Any compromise of the password file means total compromise for all users, and necessitates password changes
- No need for the system to know the cleartext password
  - Can encrypt the password with a one-way hash function, e.g. DES or MD5
  - Encrypt the password supplied by the user, if match, user specified the correct password
  - Compromise of the password file doesn't yield users passwords, just a hashed version of the same



Dictionary Attacks

- Hashed passwords are still open to attacks
  - One-way hashes are hard to reverse
  - But, one could collect a dictionary of common words, hash them all, and then check if any of the hashed words match any of the hashed passwords in the file
  - There will be a match unless people select truly random passwords
  - 20-30% of passwords on Unix systems are simple variants of dictionary words
- Solution 1: Combine hashed passwords with system-only file, also known as "shadow passwords"
- Solution 2: Add a salt to randomize the encryption of the passwords, so many such dictionaries will be necessary



One-Time Passwords

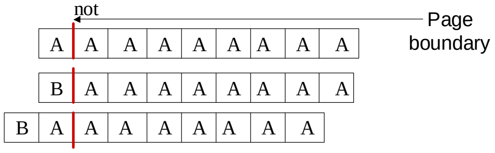
- Intruders can snoop on the conversation between the host and the principal
  - Thus can steal passwords being transmitted in the clear
- Lamport's one-time passwords
  - Use a one-way hash function y = f(x), s.t. given y, finding x is hard, but given x, y is easy to compute
  - User picks a password that he keeps secret, and n, the number of one-time passwords he needs
  - For n=3, the first password is f(f(f(x))), the second one is f(f(x)), the last one is f(x)
  - A captured password yields all previous passwords, but no future passwords
  - Need some way of performing the f() function locally

Challenge-Response Authentication

- Challenge-response scheme requires proving identity by knowing an algorithm:
  - User picks an algorithm, e.g. x^2
  - Server picks a challenge, e.g. x=7
  - User sends back 49
- In practice
  - The algorithm is fixed, e.g. a one-way hash, but the user selects a key
  - The server's challenge is combined with the user's key to provide the input to the function

Other Attacks

- Placement attack:
  - TENEX kept password in memory, compared against plaintext
  - Can align password across an invalid page boundary
  - If page fault, the prefix is valid, if illegal password, it is not

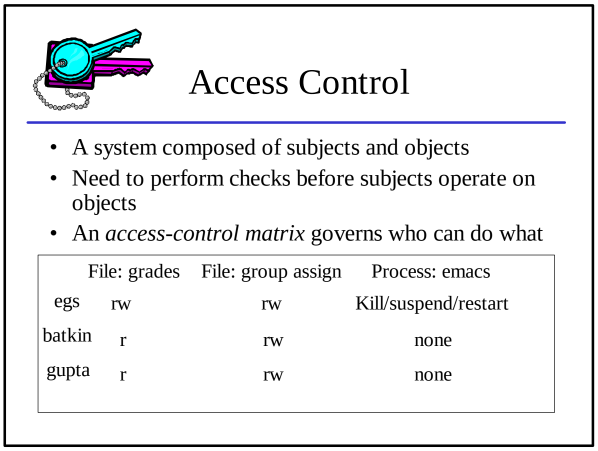


Smartcards

- A smartcard is a credit-card sized device with its own processor and memory
  - Often limited, e.g. 256 bytes
  - Small form factor makes it convenient
- Smartcards enable the computation of complicated local functions
- Smartcards also make it easy to remember long, random passwords and keys

Biometrics

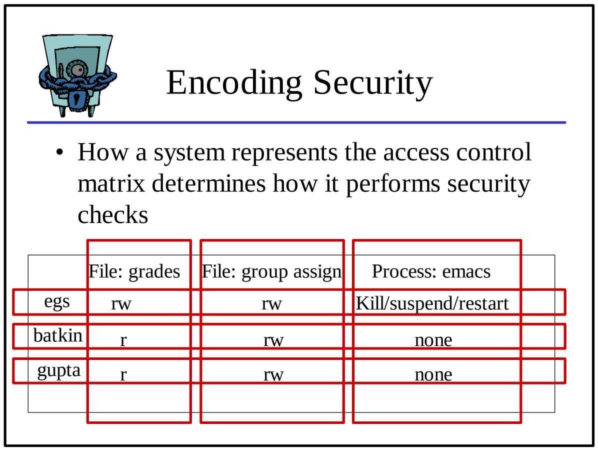
- Can use biological properties to identify people
  - Fingerprint, pattern of veins in the retina
- These properties then take the place of the user's password
  - The user can thus carry around a much longer, better randomized password
- Can be subverted
  - Password hard to change
- The systems ought to guard biometric properties like they guard passwords
  - Not always treated in a sensitive manner



Access Control

- A system composed of subjects and objects
- Need to perform checks before subjects operate on objects
- An *access-control matrix* governs who can do what

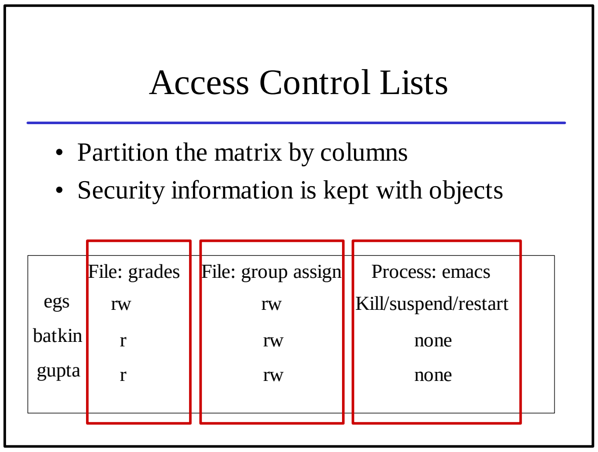
	File: grades	File: group assign	Process: emacs
egs	rw	rw	Kill/suspend/restart
batkin	r	rw	none
gupta	r	rw	none



Encoding Security

- How a system represents the access control matrix determines how it performs security checks

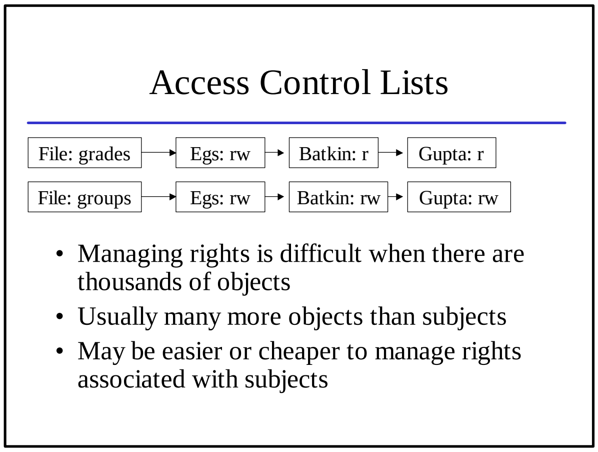
	File: grades	File: group assign	Process: emacs
egs	rw	rw	Kill/suspend/restart
batkin	r	rw	none
gupta	r	rw	none



Access Control Lists

- Partition the matrix by columns
- Security information is kept with objects

	File: grades	File: group assign	Process: emacs
egs	rw	rw	Kill/suspend/restart
batkin	r	rw	none
gupta	r	rw	none



Access Control Lists

File: grades → Egs: rw → Batkin: r → Gupta: r

File: groups → Egs: rw → Batkin: rw → Gupta: rw

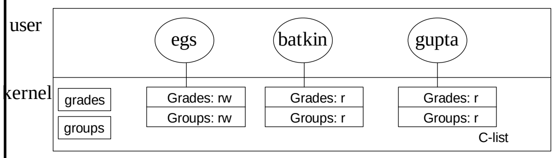
- Managing rights is difficult when there are thousands of objects
- Usually many more objects than subjects
- May be easier or cheaper to manage rights associated with subjects

Capabilities

- Partition the matrix by rows
- Security information kept with principals

	File: grades	File: group assign	Process: emacs
egs	rw	rw	Kill/suspend/restart
batkin	r	rw	none
gupta	r	rw	none

Capabilities



- Each principal has a list of objects it may access, along with a description of permitted operations
  - This list is called a C-list
- Capability lists must be protected from user tampering

Protecting Capabilities (1)

- Tagged Architecture
  - Each memory word has an extra bit indicating that it carries a capability
  - These bits can only be modified in kernel mode, and cannot be used for arithmetic, etc.
- Kernel capabilities
  - Store the C-list in kernel memory
  - Users name capabilities by offset into the C-list
  - Like file descriptors in Unix

Protecting Capabilities (2)

- Cryptographically protected capabilities
  - Store capabilities in user space
  - Store <server, object, rights, f(object, rights, check)> tuple
  - The check is a *nonce*, a unique number generated when the capability is created
- Probabilistically protected capabilities
  - 64 bit address space, each object mapped to a random location
  - Chances of guessing the address of an object are small
- Language-protected capabilities
  - SPIN operating system (Mesa, Java, et al.)