Security

Security in the real world

- Security decisions based on:
  - Value, Locks, Police
- Some observations:
  - Not all locks are the same
  - People pay for security they need
  - Police are critical to the picture
  - Security is only as good as the weakest link

Security in Computer Systems

- In computer systems, this translates to:
  - **Au**thorization
  - **Au**thentication
  - **Au**dit
- This is the Gold Standard for Security (Lampson)
- Some security goals:
  - Data confidentiality: secret data remains secret
  - Data integrity: no tampering of data
  - System availability: unable to make system unusable
  - Privacy: protecting from misuse of user’s information

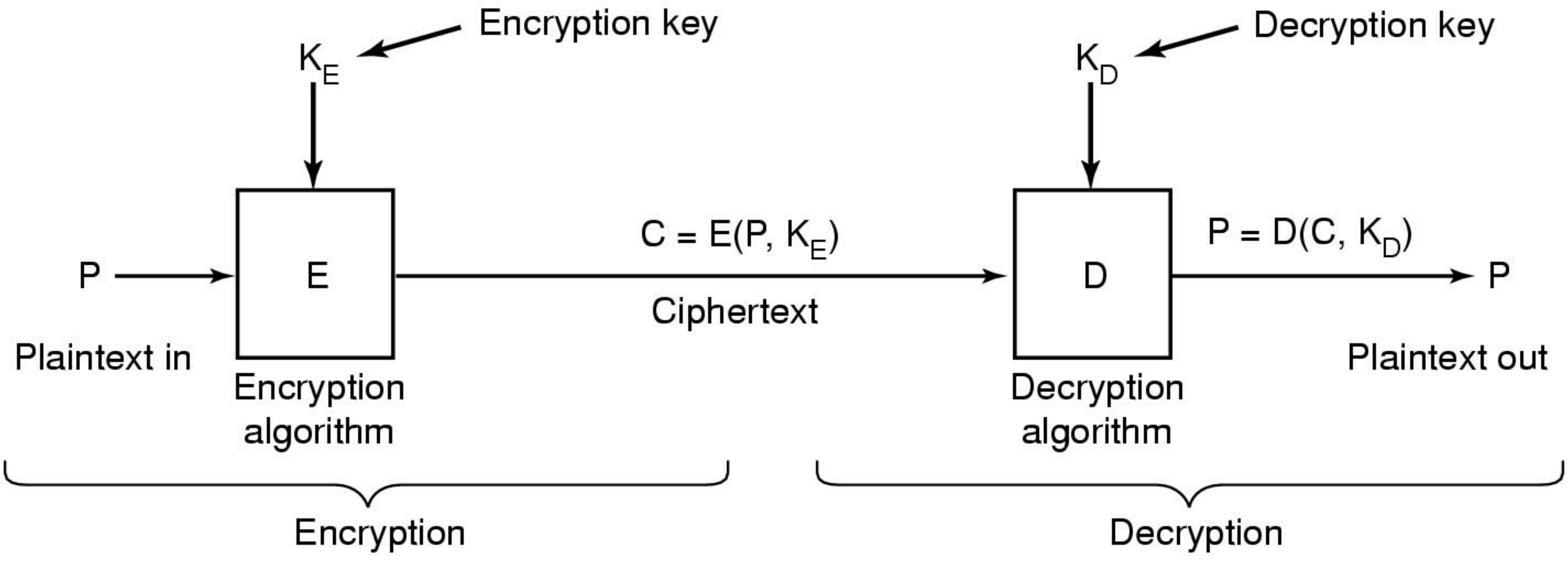
Security Threats

Identified by Defense Science Board:

- Incomplete, inquisitive and unintentional blunders.
- Hackers driven by technical challenges.
- Disgruntled employees or customers seeking revenge.
- Criminals interested in personal financial gain or stealing services.
- Organized crime with the intent of hiding something or financial gain.
- Organized terrorist groups attempting to influence U.S. policy by isolated attacks.
- Foreign espionage agents seeking to exploit information for economic, political, or military purposes.
- Tactical countermeasures intended to disrupt specific weapons or command structures.
- Multifaceted tactical information warfare applied in a broad orchestrated manner to disrupt a major U.S. military mission.
- Large organized groups or nation-states intent on overthrowing the US

Cryptography Overview

- Encrypt data so it only makes sense to authorized users
  - Input data is a message or file called plaintext
  - Encrypted data is called ciphertext



Secret-Key Cryptography

- Also called *symmetric cryptography*
  - Encryption algorithm is publicly known
  - E(message, key) = ciphertext   D(ciphertext, key) = message
- Naïve scheme: monoalphabetic substitution
  - Plaintext   : ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - Ciphertext: QWERTYUIOPASDFGHJKLZXCVBNM
  - So, *attack* is encrypted to: *qzzqea*
  - 26! possible keys ~ $4x10^{26}$ possibilities
    - 1 µs per permutation $\Rightarrow$ 10 trillion years to break
  - easy to break this scheme! How?
    - ‘e’ occurs 14%, ‘t’ 9.85%, ‘q’ 0.26%

Symmetric Key Cryptography

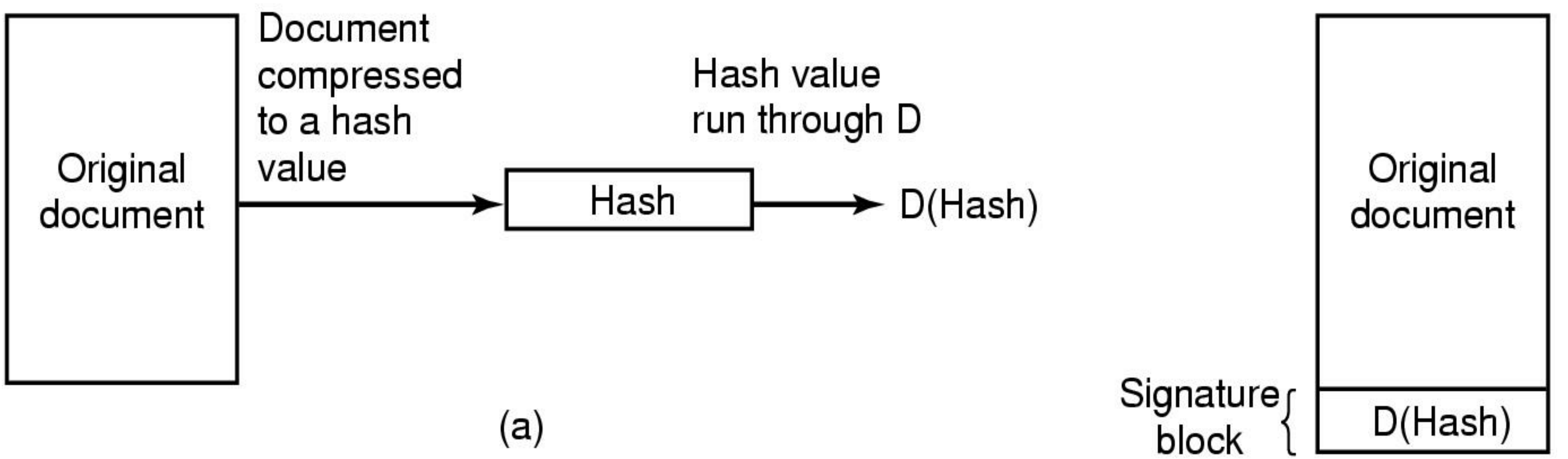
- Which encryption algorithm is good?
  - DES was proposed in the 1970s
    - Encrypts 64 bits of data with 56 bit key to give 64-bit ciphertext
    - Uses 16 rounds of substitution and permutation
    - EFF invested $250000 to break DES message in 56 hours
    - DES made powerful by encrypting message 3 times (DES3)
  - Current standard is AES
    - A result of 3-year competition with entries from 12 countries
    - Winning entry was from Belgium, called ‘Rijndael’
  - Strong algorithms, such as DES3, RC4 are used
    - WEP uses RC4

Public Key Cryptography

- Diffie and Hellman, 1976
- All users get a public key and a private key
  - Public key is published
  - Private key is not known to anyone else
- If Alice has a packet to send to Bob,
  - She encrypts the packet with Bob's public key
  - Bob uses his private key to decrypt Alice's packet
- Private key linked mathematically to public key
  - Difficult to derive by making it computationally infeasible (RSA)
- Pros: more security, convenient, digital signatures
- Cons: slower

Digital Signatures

- Hashing function hard to invert, e.g. MD5, SHA
- Apply private key to hash (decrypt hash)
  - Called signature block
- Receiver uses sender’s public key on signature block
  - *E(D(x))* = *x* should work (works for RSA)



Authentication

- Establish the identity of user/machine by
  - Something you know (password, secret)
  - Something you have (credit card, smart card)
  - Something you are (retinal scan, fingerprint)
- In the case of an OS this is done during login
  - OS wants to know who the user is
- Passwords: secret known only to the subject
  - Simplest OS implementation keeps (login, password) pair
  - Authenticates user on login by checking the password
  - Try to make this scheme as secure as possible!
    - Display the password when being typed? (Windows, UNIX)

Online passwords attacks

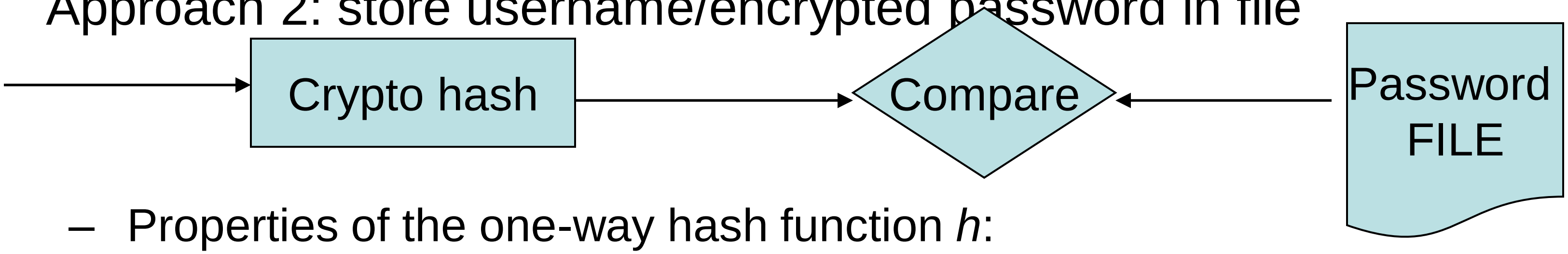
- Online attacks: system used to verify the guesses
  - How someone broke into LBL

```
LBL> telnet elxsi
ELXSI AT LBL
LOGIN: root
PASSWORD: root
INCORRECT PASSWORD, TRY AGAIN
LOGIN: guest
PASSWORD: guest
INCORRECT PASSWORD, TRY AGAIN
LOGIN: uucp
PASSWORD: uucp
WELCOME TO THE ELXSI COMPUTER AT LBL
```

  - Thwart these attacks:
    - limit the number of guesses
    - better passwords

Offline password attacks

- Depends on how passwords are stored
- Approach 1: store username/password in a file
  - Attacker only needs to read the password file
  - Security of system now depends on protection of this file!
- Approach 2: store username/encrypted password in file



  - Properties of the one-way hash function *h*:
    - *h* is not invertible: *h(m)* easy to compute, $h^{-1}(m)$ difficult
    - It is hard to find *m* and m' s.t. *h(m) = h(m')*
  - Should use standard functions, such as SHA, MD5, etc.

More offline attacks

- Previous scheme can be attacked: Dictionary Attack
  - Attacker builds dictionary of likely passwords offline
  - At leisure, builds hash of all the entries
  - Checks file to see if hash matches any entry in password file
  - There will be a match unless passwords are truly random
  - 20-30% of passwords in UNIX are variants of common words
    - Morris, Thompson 1979, Klein 1990, Kabay 1997
- Solutions:
  - Shadow files: move password file to /etc/shadow
    - This is accessible only to users with root permissions
  - Salt: store *(user name, salt, E(password+salt))*
    - Simple dictionary attack will not work. Search space is more.

Salting Example

Bobbie, 4238, e(Dog4238)
Tony, 2918, e(6%%TaeFF2918)
Laura, 6902, e(Shakespeare6902)
Mark, 1694, e(XaB@Bwcz1694)
Deborah, 1092, e(LordByron,1092)

- If the hacker guesses Dog, he has to try Dog0001, …
- UNIX adds 12-bit of salt
- Passwords should be made secure:
  - Length, case, digits, not from dictionary
  - Can be imposed by the OS! This has its own tradeoffs

One time passwords

- Password lasts only once
  - User gets book with passwords
  - Each login uses next password in list

  - UNBREAKABLE..

Challenge Response Scheme

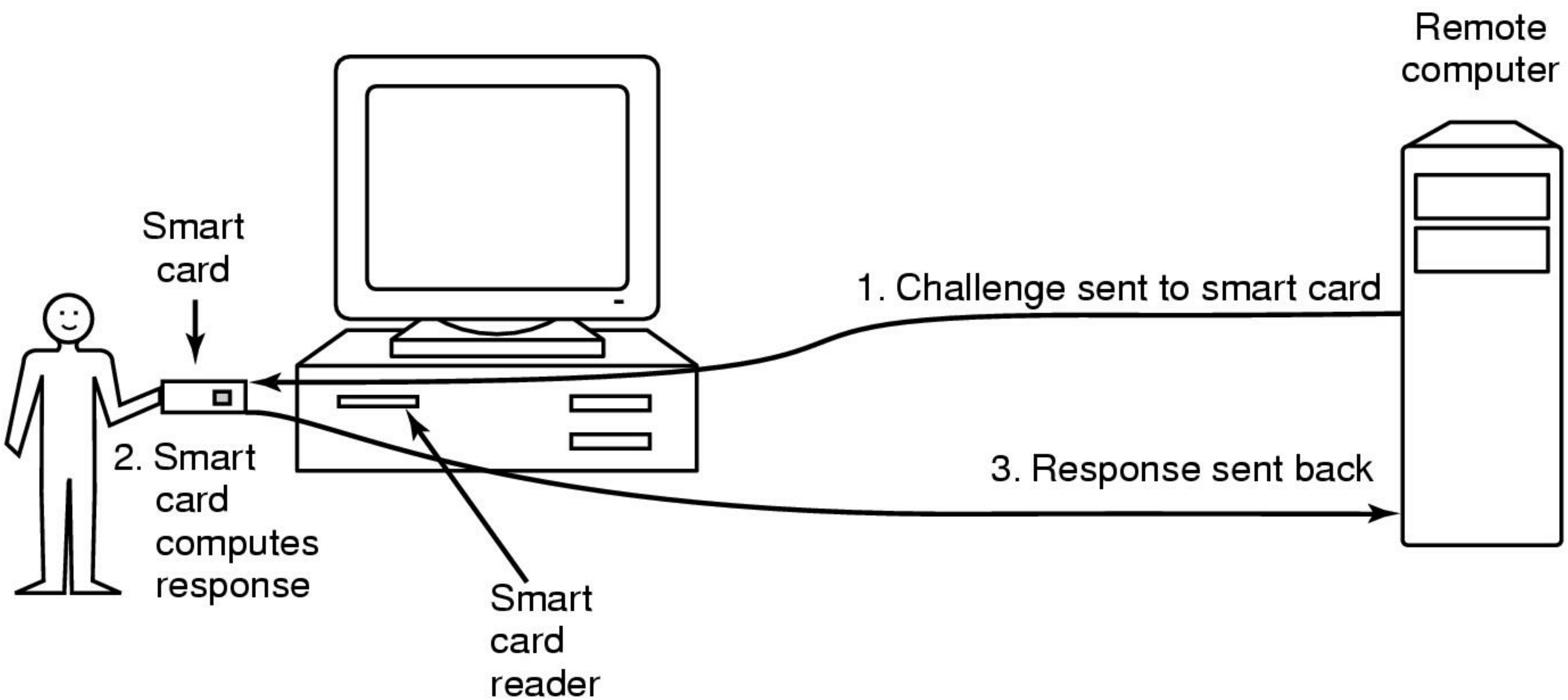
- New user provides server with list of ques/ans pairs
  - Server asks one of them at random
  - Requires a long list of question answer pairs
- Prove identity by computing a secret function
  - User picks an algorithm, e.g. x^2
  - Server picks a challenge, e.g. x=7
  - User sends back 49
  - Should be difficult to deduce function by looking at results
- In practice
  - The algorithm is fixed, e.g. one-way hash, but user selects a key
  - The server's challenge is combined with user's key to provide input to the function

Auth. Using Physical Objects

- Door keys have been around long
- Plastic card inserted into reader associated with comp
  - Also a password known to user, to protect against lost card
- Magnetic stripe cards: about 140 bytes info glued to card
  - Is read by terminal and sent to computer
  - Info contains encrypted user password (only bank knows key)
- Chip cards: have an integrated circuit
  - Stored value cards: have EEPROM memory but no CPU
    - Value on card can only be changed by CPU on another comp
  - Smart cards: 4 MHz 8-bit CPU, 16 KB ROM, 4 KB EEPROM, 512 bytes RAM, 9600 bps comm. channel

Smart Cards

- Better security than stored value cards
  - Card sends a small encrypted msg. to merchant, who can later use it to get money from the bank
  - Pros: no online connection to bank required
- Perform local computations, remember long passwords



Biometrics: something you are

- System has 2 components:
  - Enrollment: measure characteristics and store on comp
  - Identification: match with user supplied values
- What are good characteristics?
  - Finger length, voice, hair color, retinal pattern, voice, blood
- Pros: user carries around a good password
- Cons: difficult to change password, can be subverted

Security: Attacks

Trojan Horse

- Malicious program disguised as an innocent one
  - Could modify/delete user's file, send important info to cracker, etc
- The program has to get to the computer somehow
  - Cracker hides it as a new game, e-card, windows update site, etc.
- When run, Trojan Horse executes with user's privileges
- Examples:
  - Hide program in path directory as a common typo: *la* for *ls*
  - Malicious user puts malicious *ls* in directory, and attracts superuser
    - Malicious *ls* could make user the superuser
    - Denning's paper 1999

Login Spoofing

- Specialized case of Trojan Horse
  - Attacker displays a custom screen that user thinks belong to the system
  - User responds by typing in user name and password

(a) (b)

Logic Bombs

- Piece of code, in the OS or app, which is dormant until a certain time has elapsed or event has occurred
  - Event could be missing employee record from payroll
- Could act as a Trojan Horse/virus once triggered
- Also called “slag code” or “time bomb”
- Recovery options for a firm include:
  - Calling the police
  - Rehiring the programmer

Trap Doors

- Code in system inserted by programmer to bypass normal check
- Ken Thompson “Reflections on Trusting Trust”
  - Hole in UNIX system utility; enforced by C compiler

```
while (TRUE) {
    printf("login: ");
    get_string(name);
    disable_echoing( );
    printf("password: ");
    get_string(password);
    enable_echoing( );
    v = check_validity(name, password);
    if (v) break;
}
execute_shell(name);
```

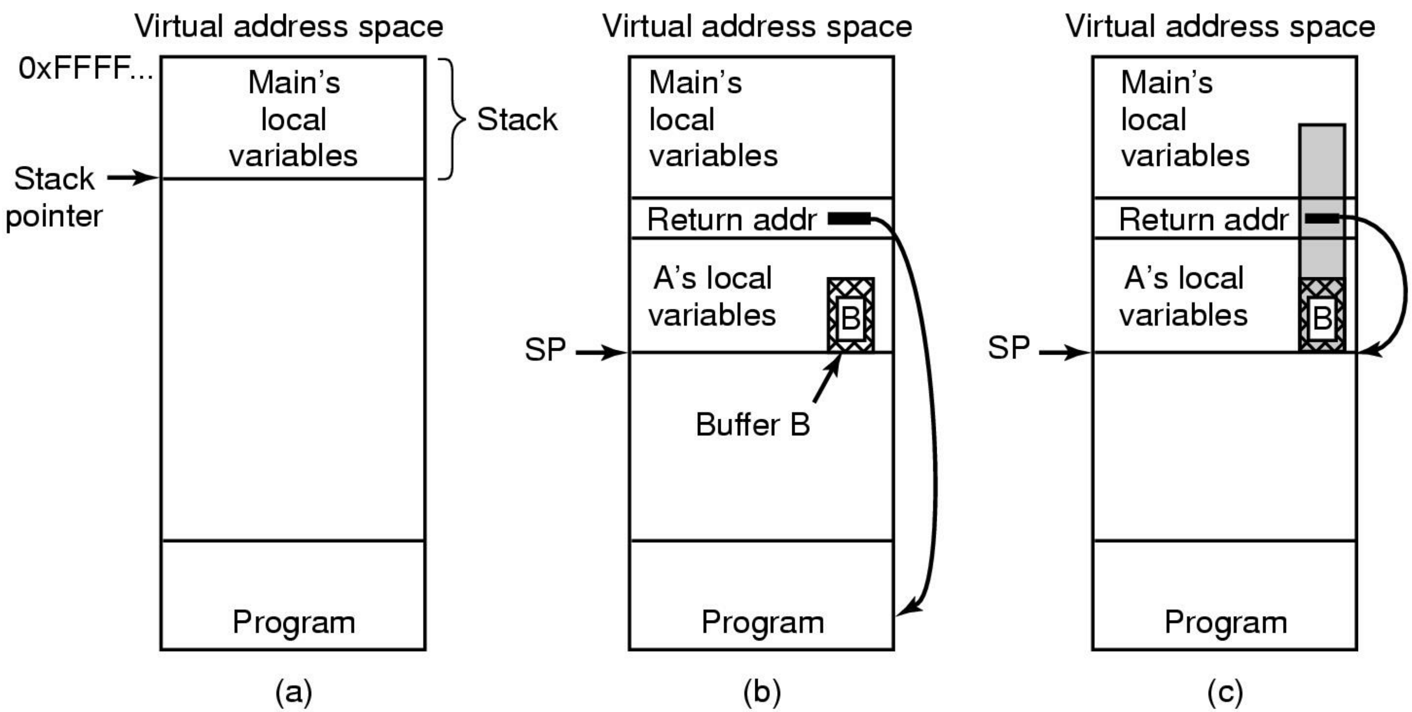
(a)

```
while (TRUE) {
    printf("login: ");
    get_string(name);
    disable_echoing( );
    printf("password: ");
    get_string(password);
    enable_echoing( );
    v = check_validity(name, password);
    if (v || strcmp(name, "zzzzz") == 0) break;
}
execute_shell(name);
```

(b)

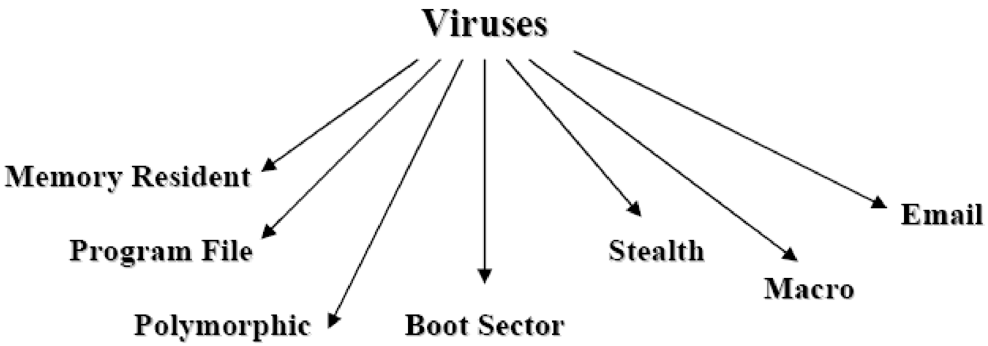
Buffer Overflow

- C compiler does no array bounds checking
  - A number of programs a written in C
  - Cracker can force his routine to run by violating array bounds



Viruses and Worms

- Virus is a program that reproduces itself by attaching its code to another program
  - They require human intervention to spread
  - Melissa, I LOVE YOU spread by e-mail
- Worms actively replicate without a helper program
  - Is a subclass of virus, but does not require user intervention
  - Sasser and Blaster targeted machines with out of date software



Denial of Service

- Client sends a legitimate-looking request for service to a service provider
- Service provider commits the necessary resources to provide the service
  - Ports, buffer space, bandwidth
- The resources are wasted, legitimate users get diminished service
  - Usually launched from many computers controlled by attackers
- Possible whenever the cost to ask for service is far cheaper than the cost of providing it
  - Challenge-response mechanism, selective packet tagging

Other Network Attacks

- Protocol attacks:
  - E.g. IEEE 802.11 WEP
- Brute force attacks
- Use Network Firewalls to reduce security risk

Protection: ACLs & Capabilities

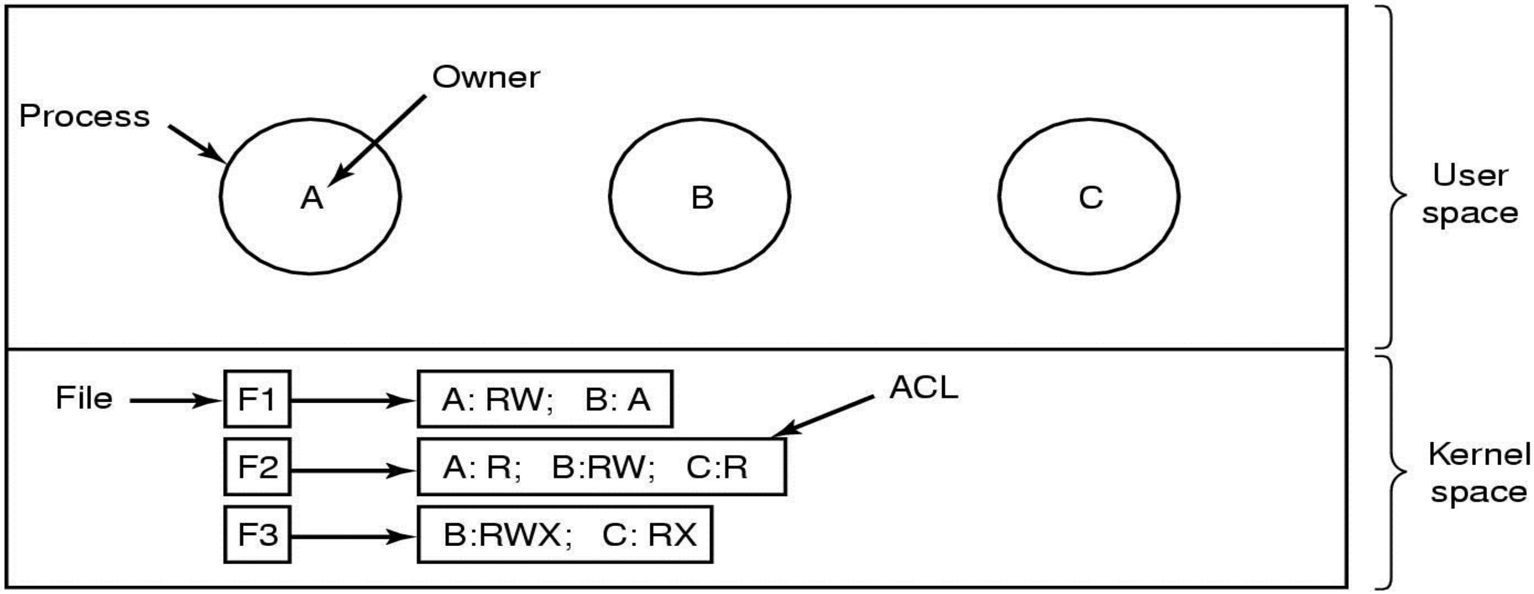
Encoding Security

- Depends on how a system represents the Matrix
  - Not much sense in storing entire matrix!
  - ACL: column for each object stored as a list for the object
  - Capabilities: row for each subject stored as list for the subject

	Cs414 grades	**Cs415 grades**	**Emacs**
Ranveer	r/w	r/w	Kill/resume
Tom	r	r/w	None
Mohamed	r	r	None

Access Control Lists

- Example: to control file access



File	Access control list
Password	tana, sysadm: RW
Pigeon_data	bill, pigfan: RW; tana, pigfan: RW; ...

Access Control Lists Examples

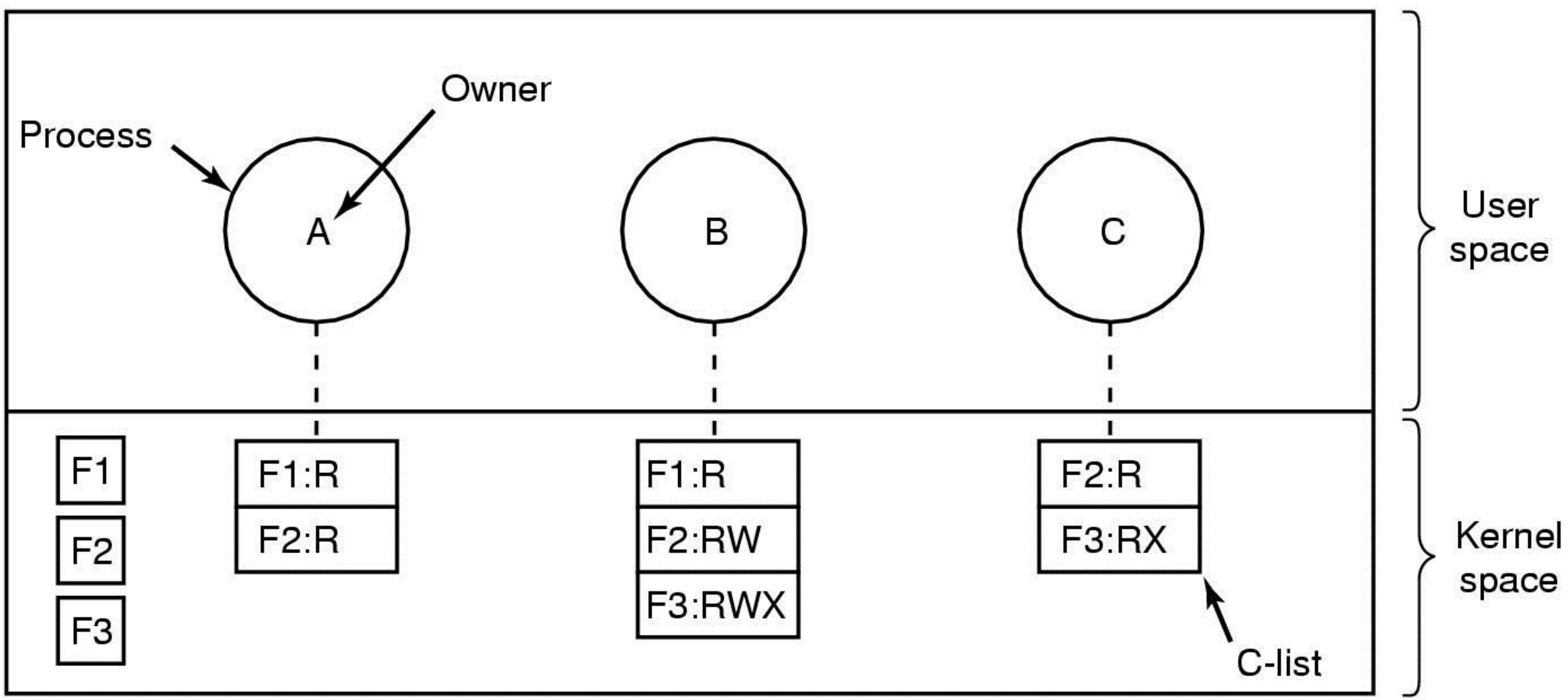
- UNIX: has *uid* and *gid*
  - Each i-node has 12 mode bits for user, group and others
  - What does x without r mean for a directory?
    - Can access file if you know the name, but cannot list names
  - What does r without x mean?
    - Can list files, but cannot access them
  - Only the owner can change these rights with chmod command
  - Last 3 mode bits allow process to change across domains
- In NTFS: each file has a set of properties (ACL is one)
  - Richer set than UNIX: RWX P(permission) O(owner) D(delete)
  - Further packaging: read (RX), change (RWXO), full control (RWXOPD)

ACLs Discussion

- Need good data structures
- User will need to have multiple identities
- Need defaults for new objects
- Good security metaphors to users are needed!

Capabilities

- Store information by rows
  - For each subject, there is list of objects that it can access
  - Called a capability list of c-list; individual items are



Capabilities

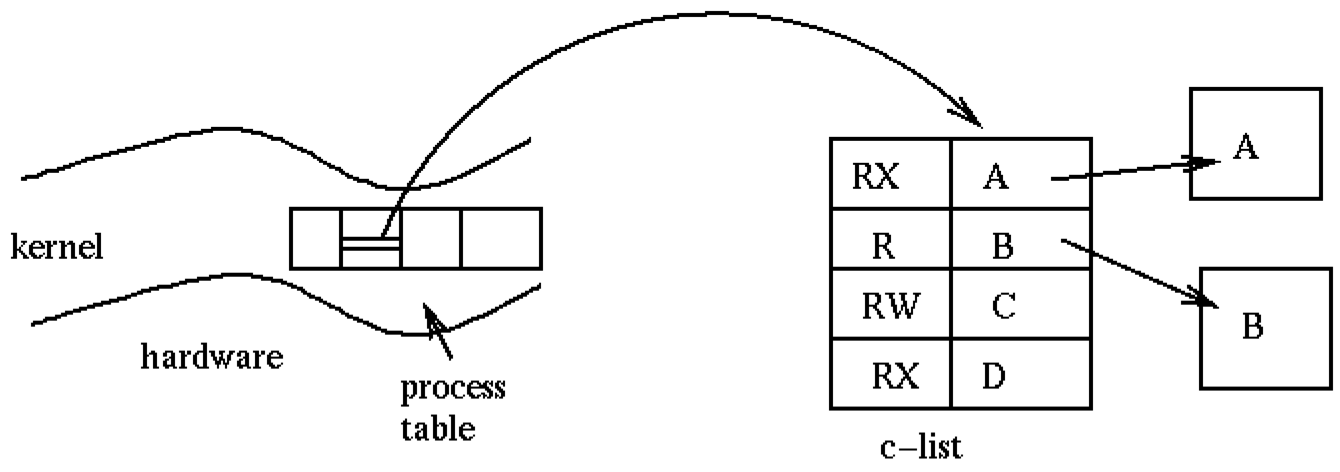
- To access an object, subject presents the capability
  - ‘capability’ word coined by Dennis and Van Horn in 1966
  - Capability is (x, r) list. x is object and r is set of rights
  - Capabilities are transferable
- How to name an object?
  - Is start address sufficient?
    - Array and first element of array have same address
  - Is start address + length of object sufficient?
    - What if start address changes?
  - Random bit string: use hash table to translate from name to bits
- Need to protect capabilities from being forged by others
  - ACLs were inherently unforgeable

Protecting Capabilities

- Prevent users from tampering with capabilities
- Tagged Architecture
  - Each memory word has extra bit indicating that it is a capability
  - These bits can only be modified in kernel mode
  - Cannot be used for arithmetic, etc.
- Sparse name space implementation
  - Kernel stores capability as *object+rights+random number*
  - Give copy of capability to the user; user can transfer rights
  - Relies on inability of user to guess the random number
    - Need a good random number generator

Protecting Capabilities

- Kernel capabilities: per-process capability information
  - Store the C-list in kernel memory
  - Process access capabilities by offset into the C-list
  - Indirection used to make capabilities unforgeable
  - Meta instructions to add/delete/modify capabilities



Protecting Capabilities

- Cryptographically protected capabilities
  - Store capabilities in user space; useful for distributed systems
  - Store <server, object, rights, f(object, rights, check)> tuple
  - The check is a *nonce*,
    - unique number generated when capability is created;
    - kept with object on the server; never sent on the network
- Language-protected capabilities
  - SPIN operating system (Mesa, Java, etc.)

Capability Revocation

- Kernel based implementation
  - Kernel keeps track of all capabilities; invalidates on revocation
- Object keeps track of revocation list
  - Difficult to implement
- Timeout the capabilities
  - How long should the expiration timer be?
- Revocation by indirection
  - Grant access to object by creating alias; give capability to alias
  - Difficult to review all capabilities
- Revocation with conditional capabilities
  - Object has state called “big bag”
  - Access only if capability’s little bag has sth. in object’s big bag

Comparing ACLs & Capabilities

- Number of comparisons on opening a file?
  - Capability: just one   ACLs: linear with number of subjects
- Implementing when no groups are supported:
  - Capabilities: easier   ACLs: Need to enumerate all the subjects
- Finding out who has access to an object?
  - Capabilities: difficult
- Is it possible to control propagation of rights?
  - Capabilities: some counter can be used
- Selective revocation of rights:
  - Easy for ACLs (no immediate effect); difficult for capabilities
- Easier propagation of rights for capabilities

Trusted Systems

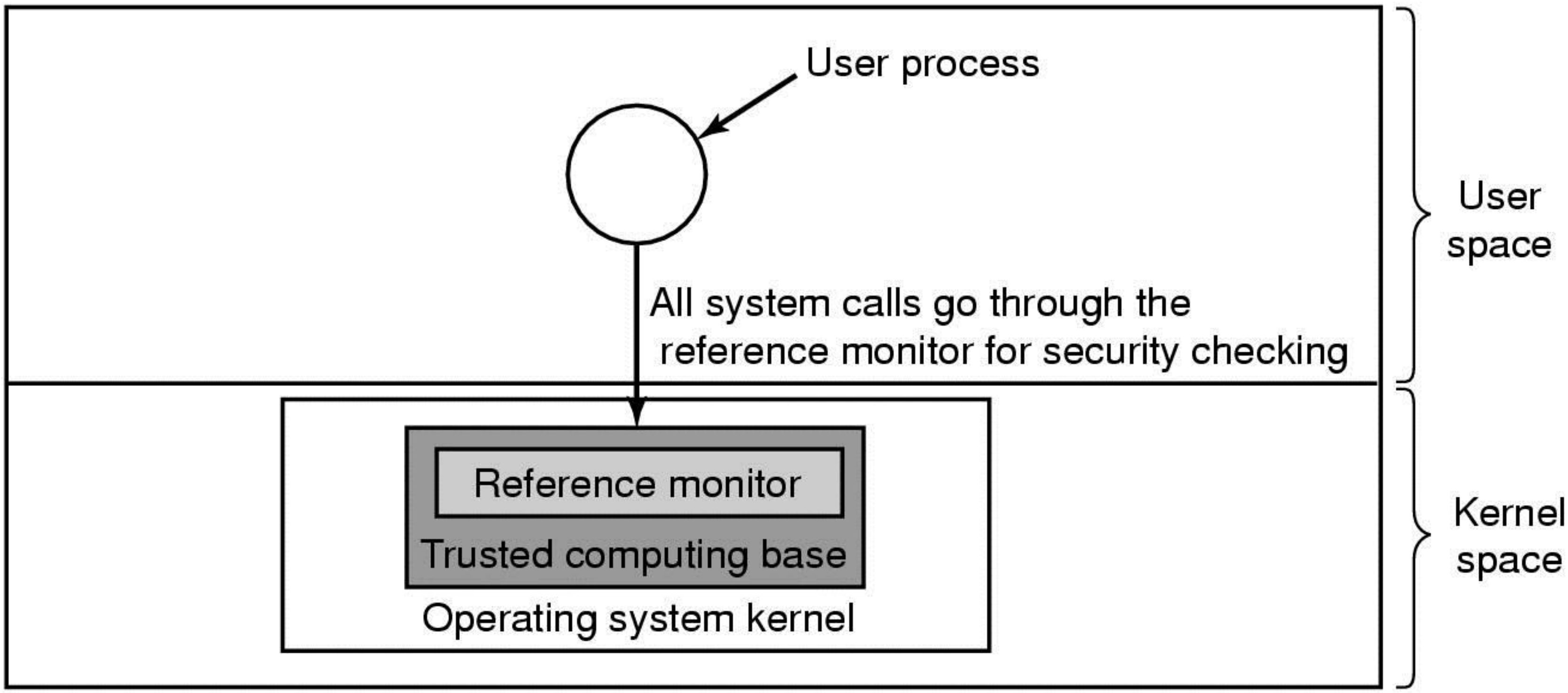
- The computer world right now is full of security problems
- Can we build a secure computer system?
  - Yes!
- Then why has it not been built yet?
  - Users unwilling to throw out existing systems
  - New systems have more features, so:
    - more complexity, code, bugs and security errors
- Examples: e-mail (from ASCII to Word), web (applets)
- Trusted Systems: formally stated security requirements, and how they are met

Trusted Computing Base

- Heart of every trusted system has a small TCB
  - Hardware and software necessary for enforcing all security rules
  - Typically has:
    - most hardware,
    - Portion of OS kernel, and
    - most or all programs with superuser power
- Desirable features include:
  - Should be small
  - Should be separable and well defined
  - Easy to audit independently

Reference Monitor

- Critical component of the TCB
  - All sensitive operations go through the reference monitor
  - Monitor decides if the operation should proceed



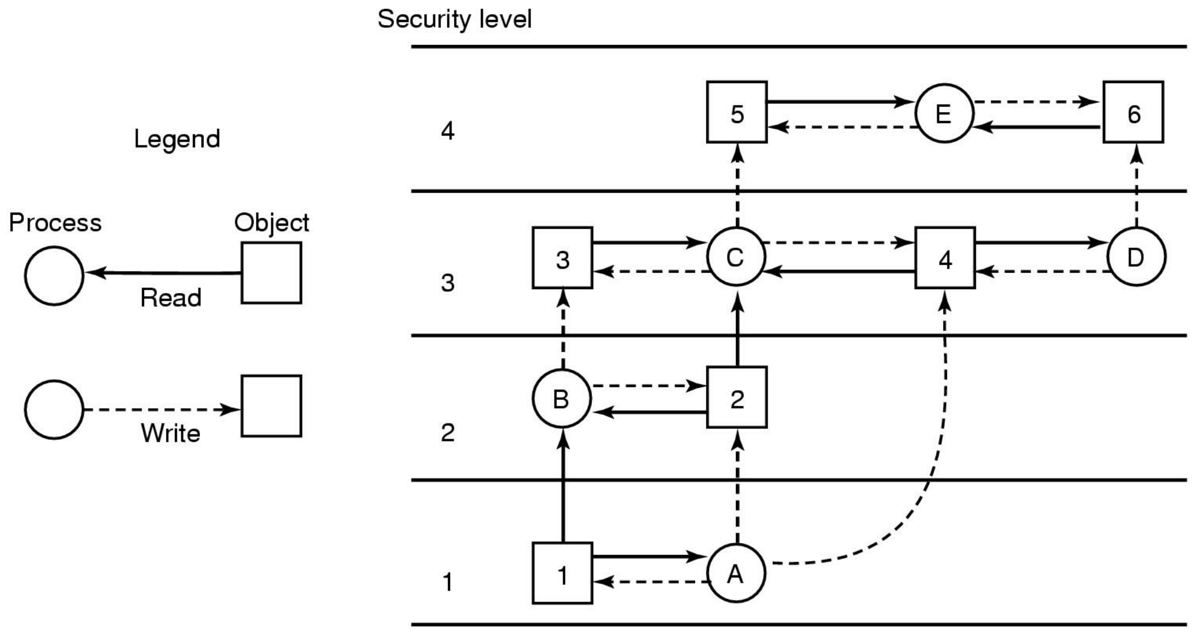
Access Control

- Discretionary Access Control (DAC)
  - Subjects can determine who has access to their objects
  - Commonly used, for example in Unix File System
  - Is flawed for tighter security, since program might be buggy
- Mandatory Access Control (MAC)
  - System imposes access control policy that object owner's cannot change
  - Multi-level Security as an example of MAC
    - MLS is environment where there are various security levels
      - Eg. Classify info as unclassified, confidential, secret, top secret
      - General sees all documents, lieutenant can only see below confidential
    - Restrict information flow in environments where various levels interact

Bell-La Padula Model

"no read up, no write down"

- Properties to satisfy for information flow
  - Security property: user at level 'k' can read objects at level 'j'
    - j <= k
  - * property: user can write objects at level j >= k



Biba Model

"No write up, no read down"

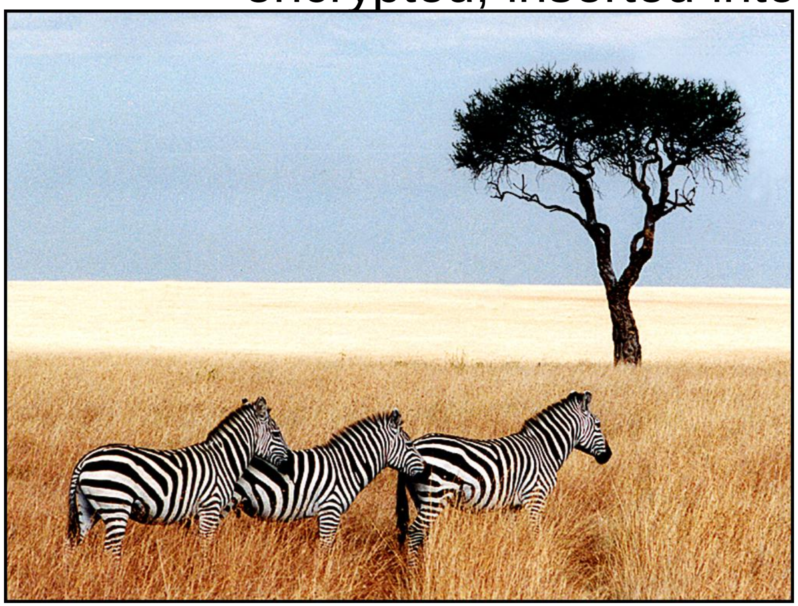
- Integrity property: A user at security level k can write only objects at level j, j <= k
- The integrity * property: A user at level k can read only objects at level j, j >= k
- Want Bell-La Padula and Biba in the same system, for different types of objects
  - But Bell-La Padula and Biba are in direct conflict (Confidentiality vs. Data Integrity)
- In practice, a mix of DAC and MAC

Covert Channels

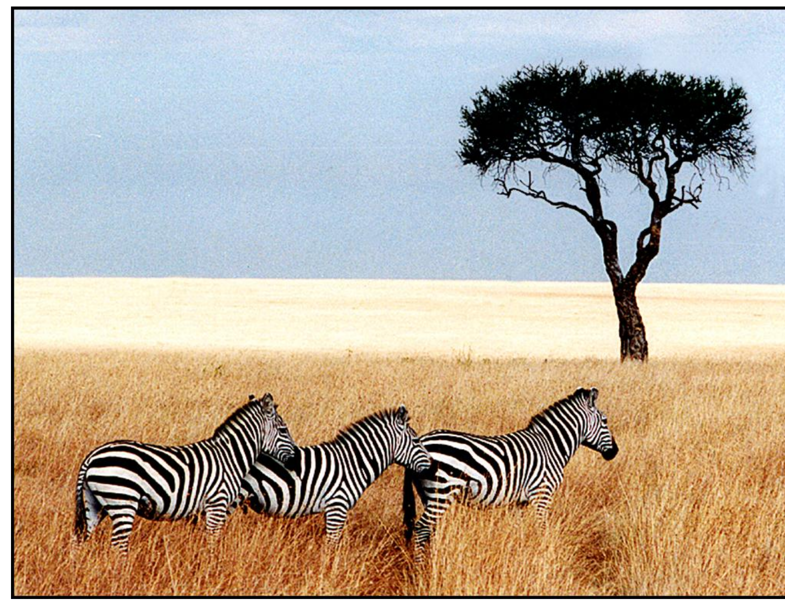
- Do these ideas make our system completely secure?
  - No. Security leaks possible even in a system proved secure mathematically. Lampson 1973
- Model: 3 processes. The client, server and collaborator
  - Server and collaborator collude
  - Goal: design a system where it is impossible for server to leak to the collaborator info received from the client (Confinement)
- Solution: Access Matrix prevents server to write to a file that collaborator has read access; no IPC either
- BUT: Covert Channel => compute hard for 1, sleep for a 0
- Others: paging, file locking with ACKs, pass secret info even though there is censor

Steganography

- Pictures appear the same
- Picture on right has text of 5 Shakespeare plays
  - encrypted, inserted into low order bits of color values



Zebras



Hamlet, Macbeth, Julius Caesar
Merchant of Venice, King Lear

Orange Book

- Dept. of Defense Standards DoD 5200.28 in 1985
  - Known as Orange Book for the color of its cover
- Divides OSes into categories based on security property
  - **D** – Minimal security.
  - **C** – Provides discretionary protection through auditing. Divided into **C1** and **C2**. **C1** identifies cooperating users with the same level of protection. **C2** allows user-level access control.
  - **B** – All the properties of **C**, however each object may have unique sensitivity labels. Divided into **B1**, **B2**, and **B3**.
  - **A** – Uses formal design and verification techniques to ensure security.