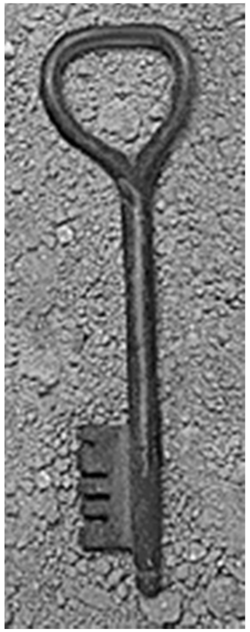
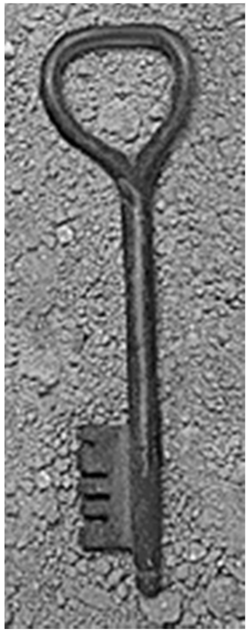


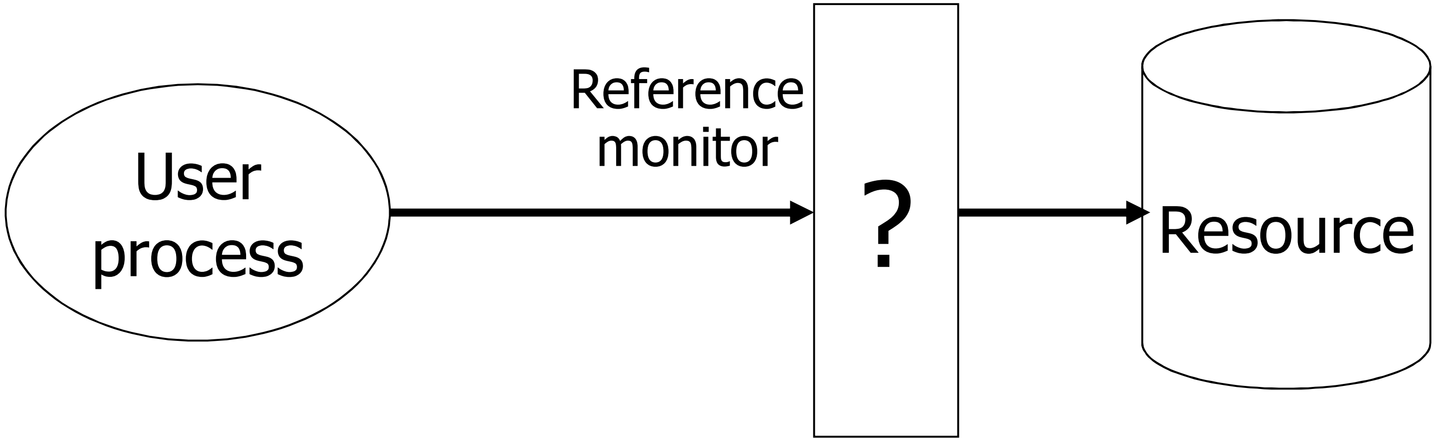
Authorization

- **given who you are, what can you do?**
- **how do we control privileges?**

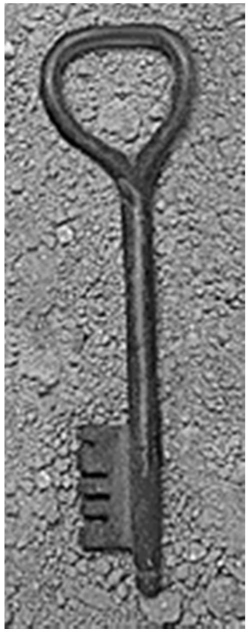


Authorization - Access Control

- Common Assumption
  - System knows who the user is
    - User has passed Identification & Authentication
  - Access requests pass through gatekeeper
    - System must be designed monitor cannot be bypassed

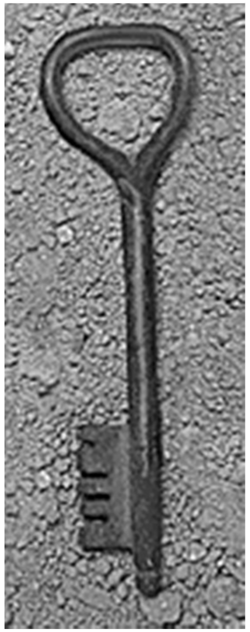


Decide whether user can apply operation to resource



Agenda

- Access matrix model
- Access control lists versus Capabilities
- Discretionary versus mandatory controls
- Bell-LaPadula model
- Advanced: Covert channels
- Advanced: Beyond MAC and DAC



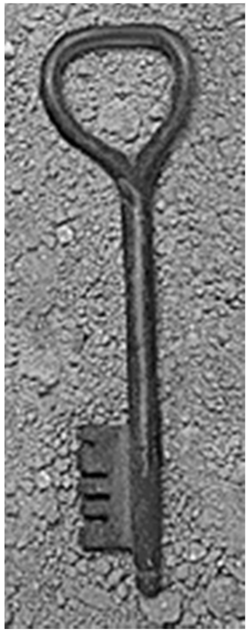
Access Control Matrix [Lampson 1971]

Objects

rights

	File 1	File 2	File 3	…	File n
User 1	read	write	-	-	read
User 2	write	write	write	-	-
User 3	-	-	-	read	read
…					
User m	read	write	read	write	read

Subjects

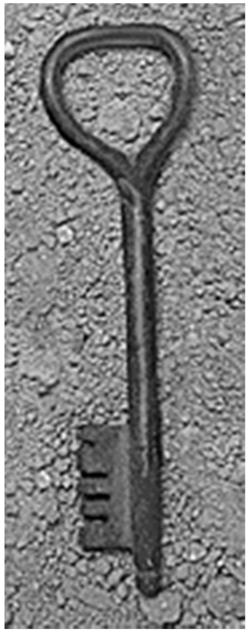


Access Control Matrix

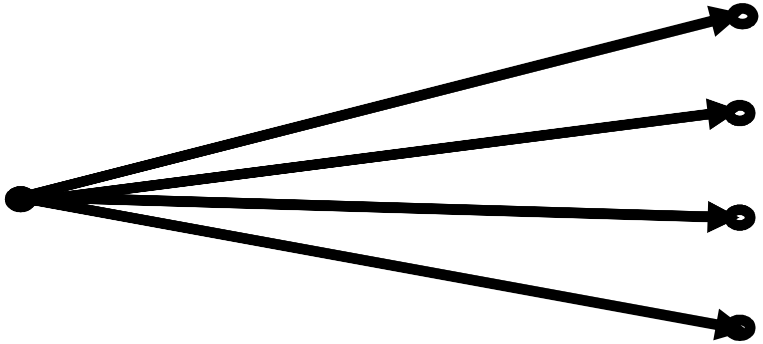
- Basic Abstractions
  - Subjects
  - Objects
  - Rights

	File 1	File 2	File 3	...	File n
User 1	read	write	-	-	read
User 2	write	write	write	-	-
User 3	-	-	-	read	read
...					
User m	read	write	read	write	read

- The rights in a cell specify the access of the subject (row) to the object (column)



Subjects = Users ?



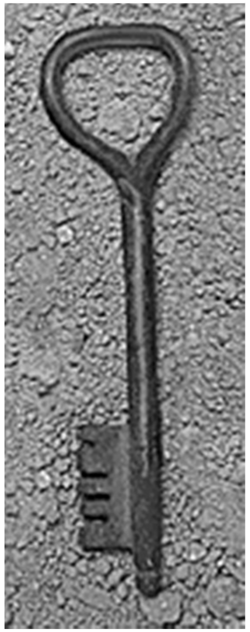
USERS

PRINCIPALS

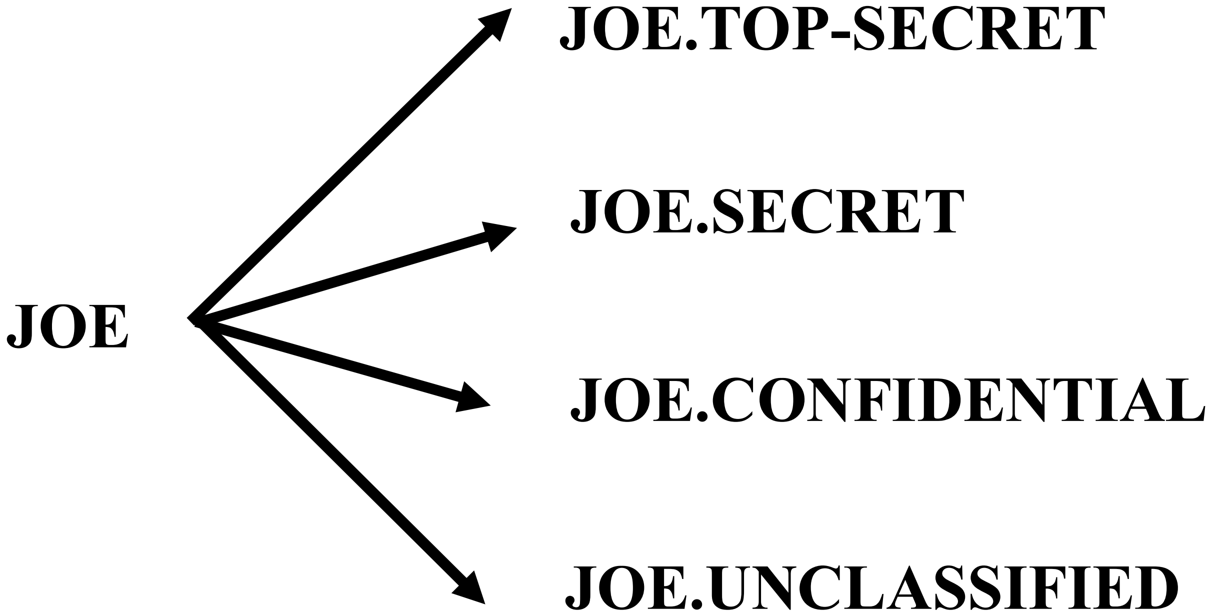
Real World User

Unit of Access Control and Authorization

the system authenticates the user in context of a particular principal



Subjects = Users ?



JOE

JOE.TOP-SECRET

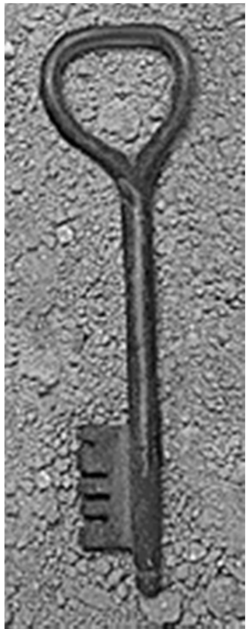
JOE.SECRET

JOE.CONFIDENTIAL

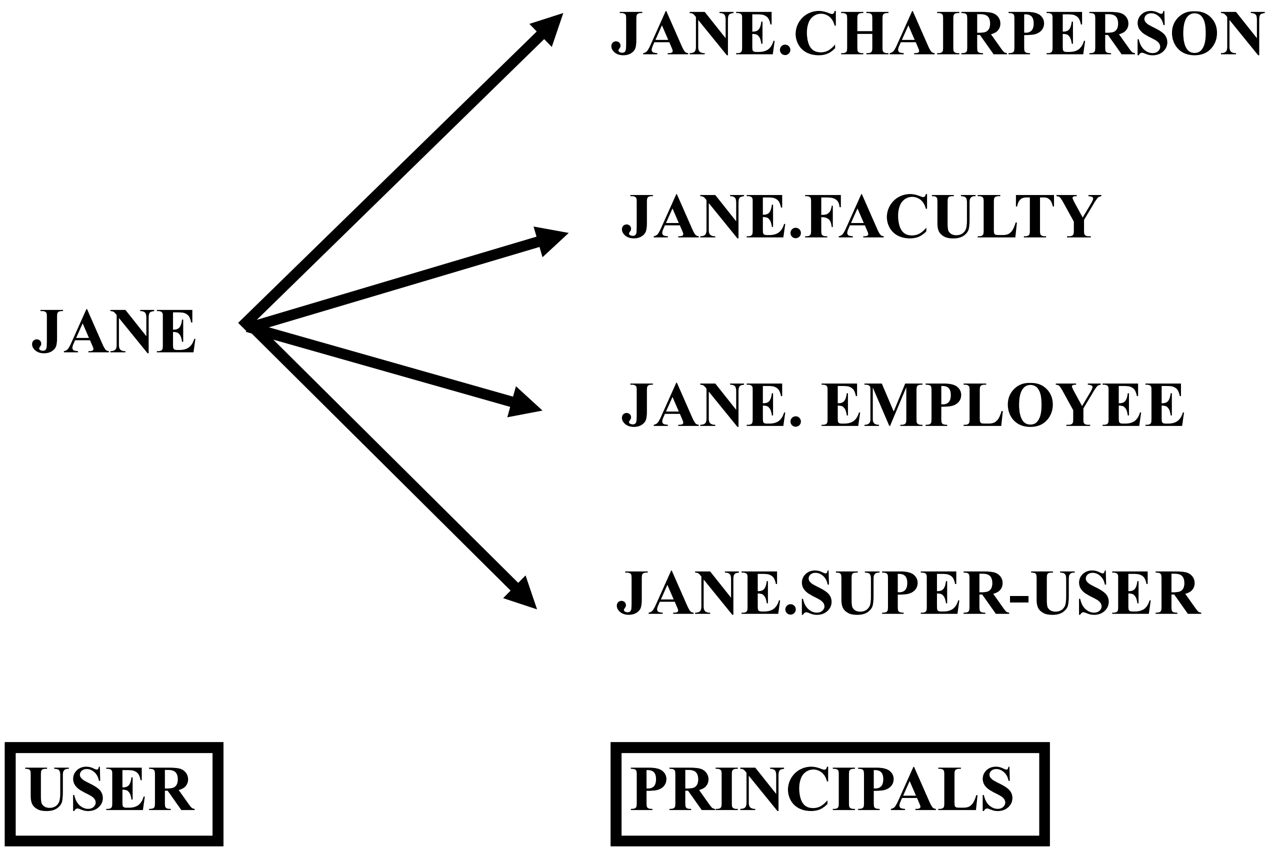
JOE.UNCLASSIFIED

USER

PRINCIPALS



Subjects = Users ?



JANE

JANE.CHAIRPERSON

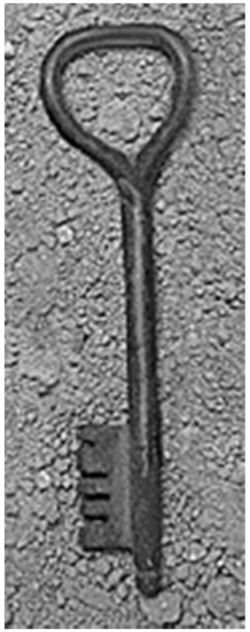
JANE.FACULTY

JANE. EMPLOYEE

JANE.SUPER-USER

USER

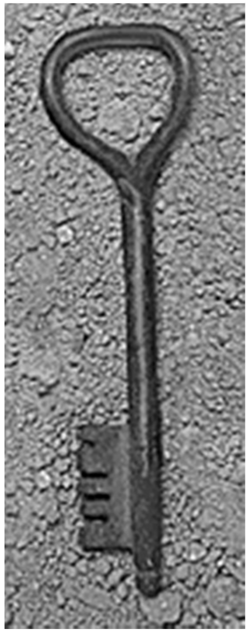
PRINCIPALS



Subjects = Users ?

- There should be a one-to-many mapping from users to principals
  - a user may have many principals, but
  - each principal is associated with an unique user
- This ensures accountability of a user's actions

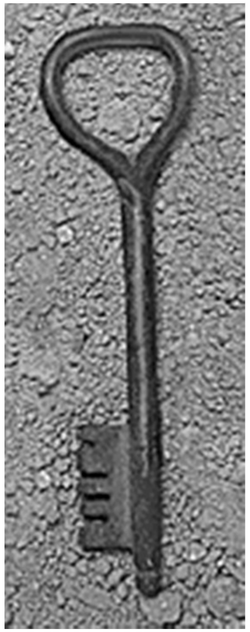
In other words, shared accounts (principals) are bad for accountability

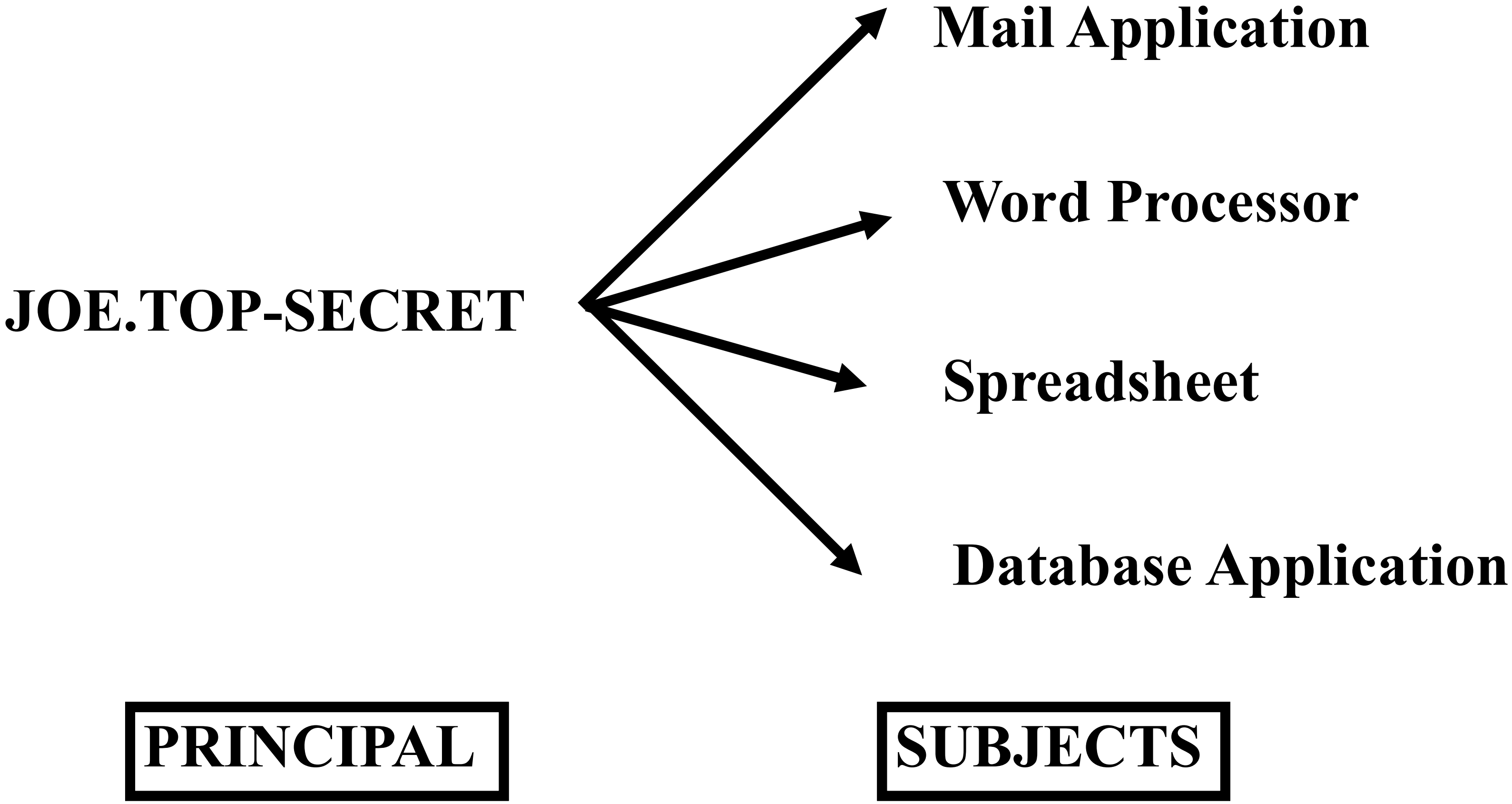


Subjects = Users ?

- A subject is a program (application) executing on behalf of a principal
- A principal may at any time be idle, or have one or more subjects executing on its behalf

Principles and Subjects





JOE.TOP-SECRET

Mail Application

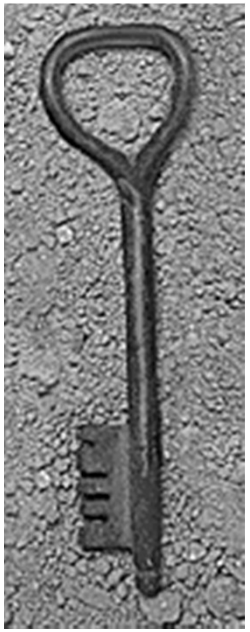
Word Processor

Spreadsheet

Database Application

PRINCIPAL

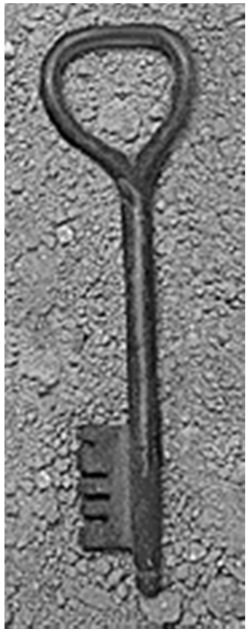
SUBJECTS



Principles and Subjects

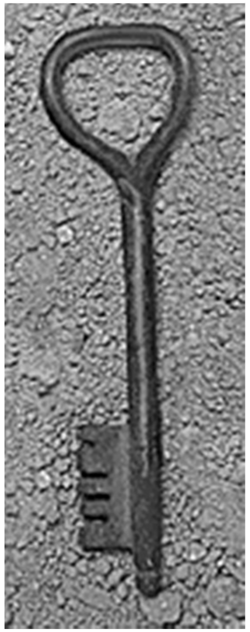
- ♦ Usually (but not always)
  - each subject is associated with a unique principal
  - all subjects of a principal have identical rights (equal to the rights of the invoking principal)
- ♦ This case can be modeled by a one-to-one mapping between subjects and principals

For simplicity, a principal and subject can be treated as identical concepts. On the other hand, a user should always be viewed as multiple principals



Objects

- An object is anything on which a subject can perform operations (mediated by rights)
- Usually objects are passive, for example:
  - File
  - Directory (or Folder)
  - Memory segment
- But, subjects can also be objects, with operations
  - Kill, Suspend, Resume



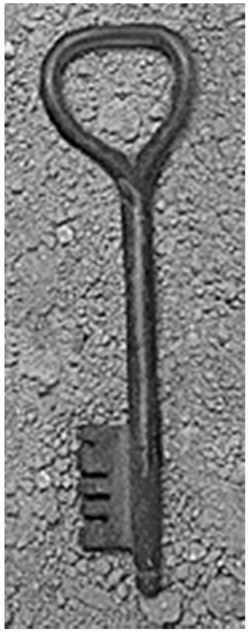
Access Control Matrix [Lampson]

Objects

rights

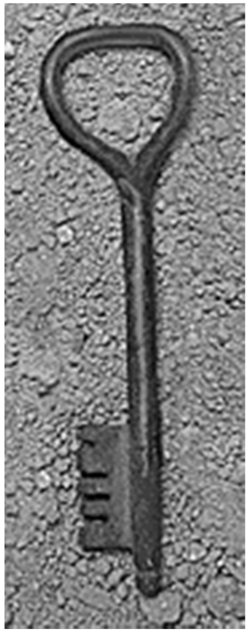
Subjects

	File 1	File 2	File 3	…	File n
User 1	read	write	-	-	read
User 2	write	write	write	-	-
User 3	-	-	-	read	read
…					
User m	read	write	read	write	read



Agenda

- Access matrix model
- Access control lists versus Capabilities
- Discretionary versus mandatory controls
- Bell-LaPadula model
- Covert channels
- Beyond MAC and DAC

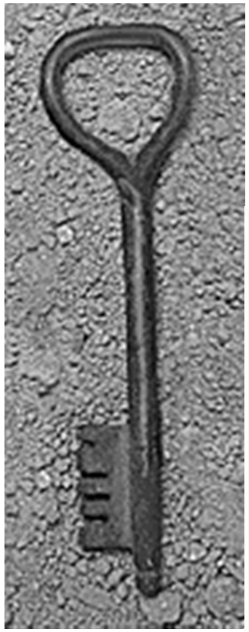


How to Implement ?

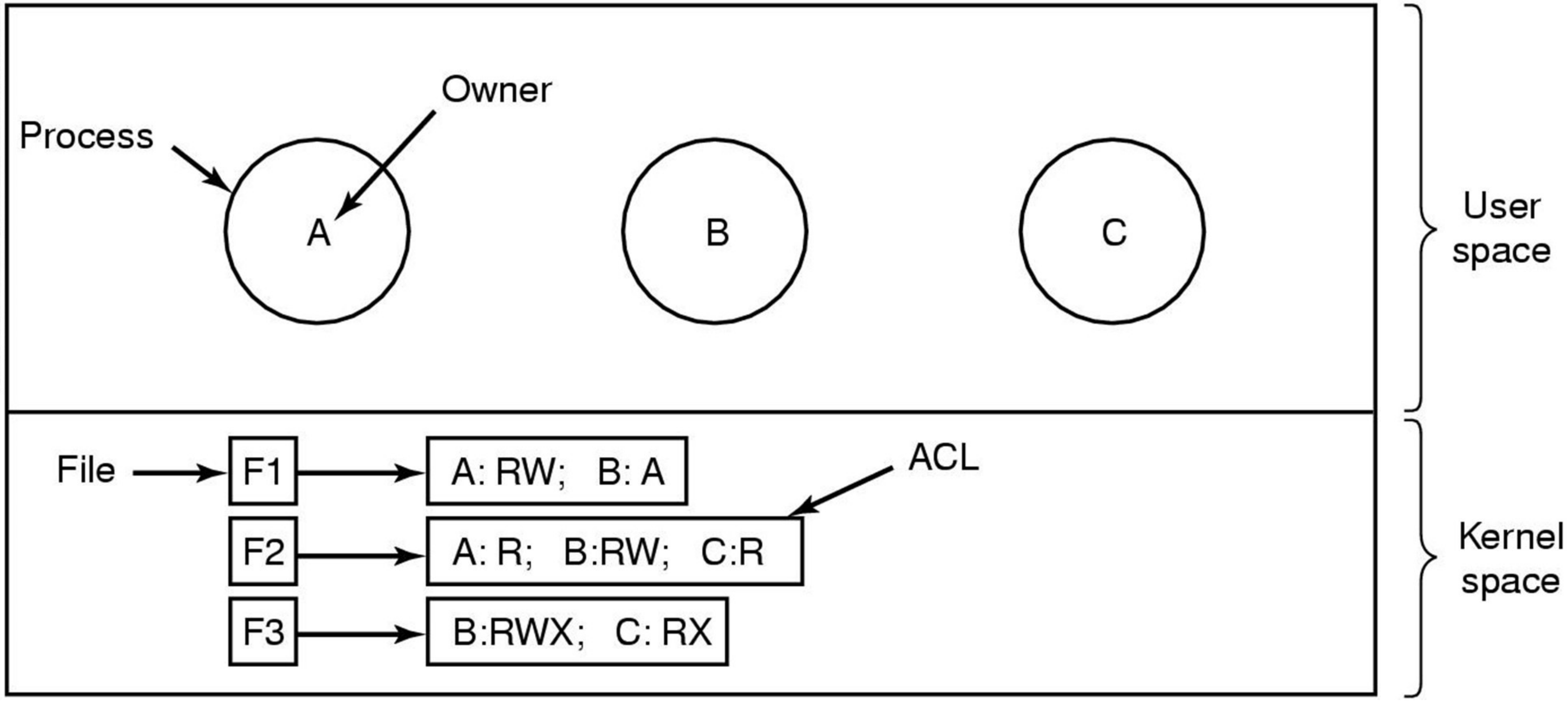
- Access control list (ACL)
  - Store column of matrix with the resource
- Capability
  - User holds a unforgeable “ticket” for each resource

	File 1	File 2	...
User 1	read	write	-
User 2	write	write	-
User 3	-	-	read
...			
User m	read	write	write

访问控制列表，权限跟着每个文件。

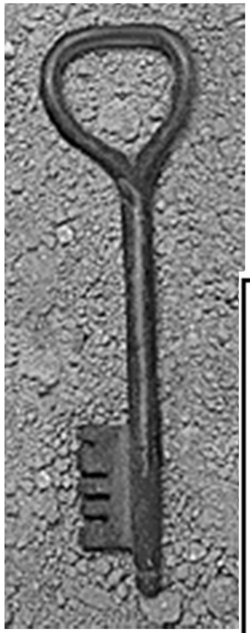


Access Control Lists (1)

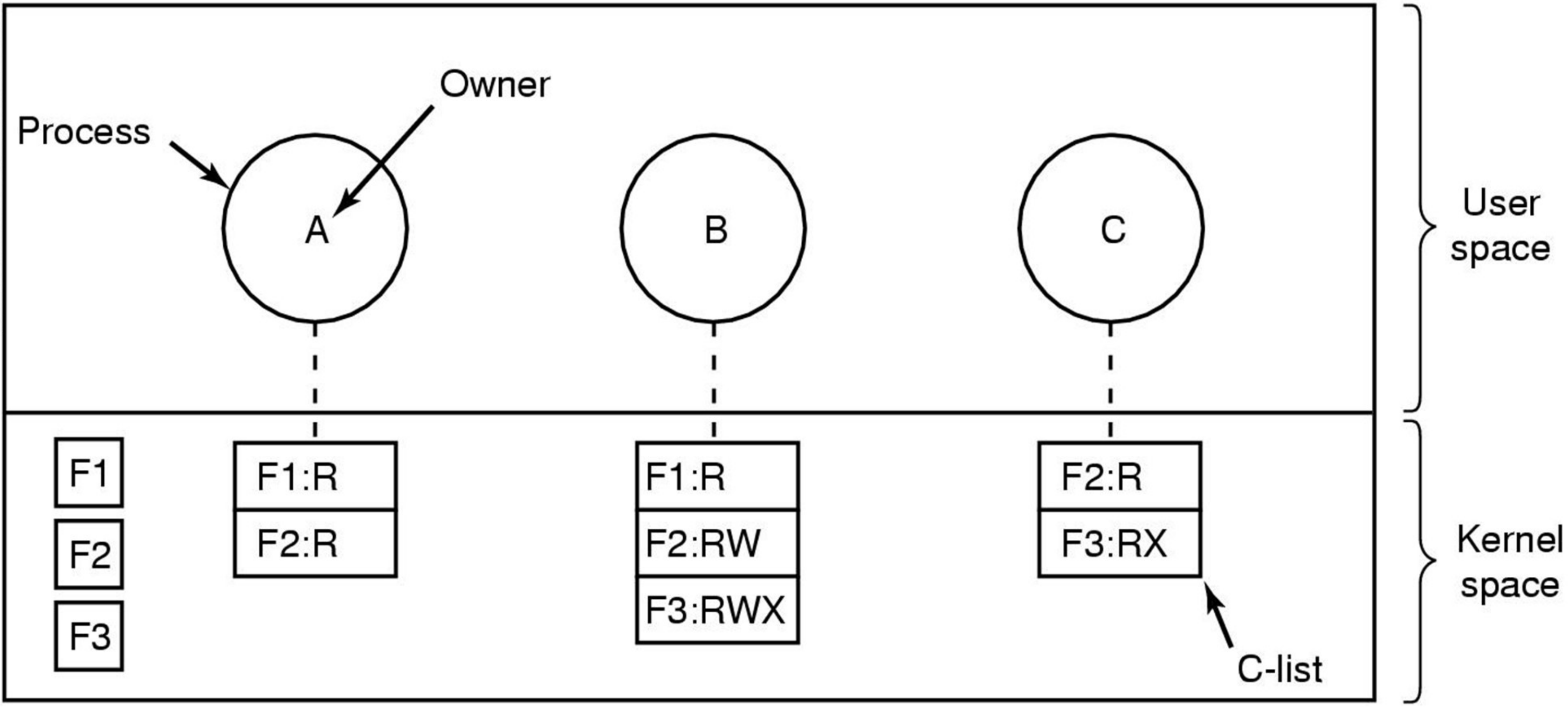


Use of access control lists of manage file access

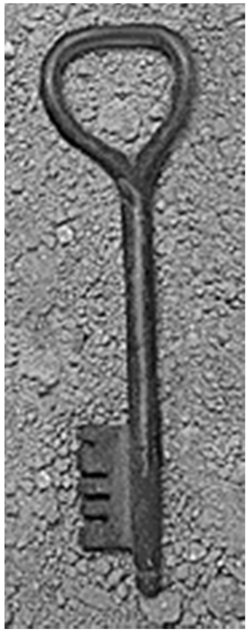
能力列表，人 有 权限表，当回收时，直接把A的删除掉，不用遍历所有客体。



Capabilities (1)

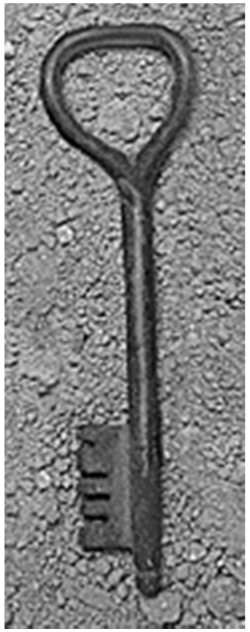


Each process has a capability list



ACL'S vs Capabilities

- ACL's require authentication of subjects
- Capabilities do not require authentication of subjects, but do require unforgeability and control of propagation of capabilities



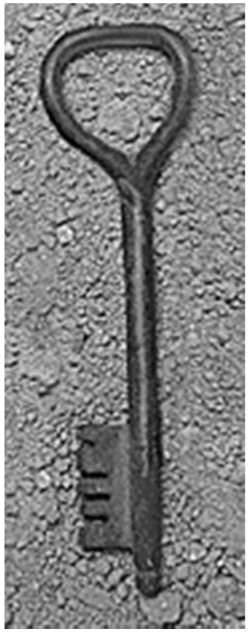
ACL'S vs Capabilities

Access Review

- ACL's provide for superior access review on a per-object basis
- Capabilities provide for superior access review on a per-subject basis

Revocation

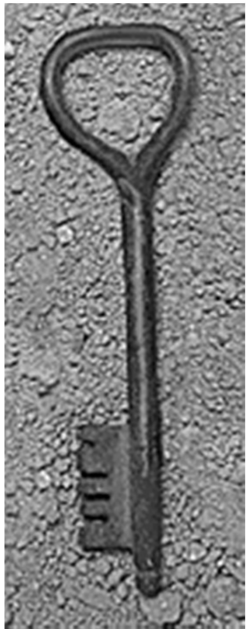
- ACL's provide for superior revocation facilities on a per-object basis
- Capabilities provide for superior revocation facilities on a per-subject basis



ACL usually wins out !

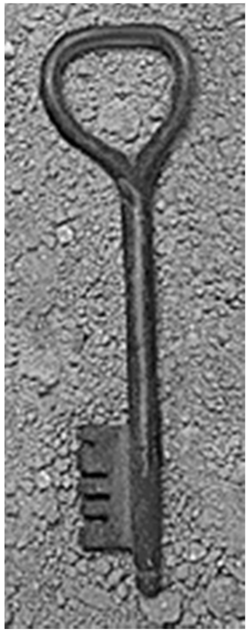
- The per-object basis usually wins out so most Operating Systems protect files by means of ACL's
- Many Operating Systems use an abbreviated form of ACL's with just three entries (Like Unix)
  - Owner, Group, Other
  - rwx-rwx-rwx

每个列表只需要九个bit



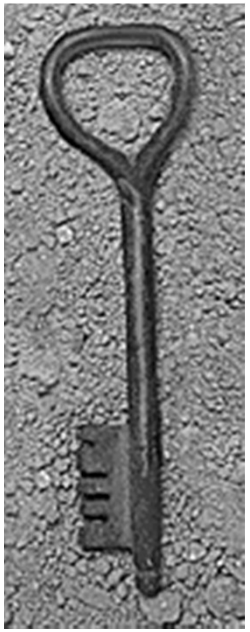
Case Study: UNIX O.S.

- Unix and other operating systems using access lists for file management often “abbreviate” the lists by dividing users into categories based on their relationships
- Typical categories
  - **User** who ***owns*** the file
  - Users in the same ***group*** as the file’s owner
  - All ***other*** users (or everyone else in the ***world***)



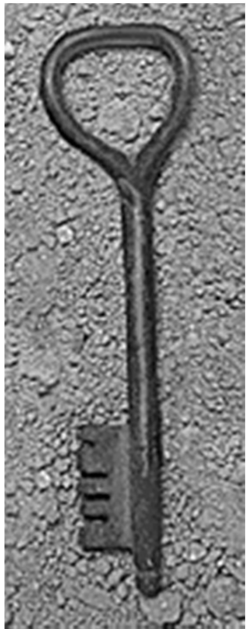
Unix-style representation

- We minimize the privileges in Unix-like systems to: **R**ead, **W**rite, E**x**ecute
- We therefore need nine bits to represent a file’s access list:
  - First three: owner bits
  - Second three: group bits
  - Third three: world/other bits



Equivalent Expressions

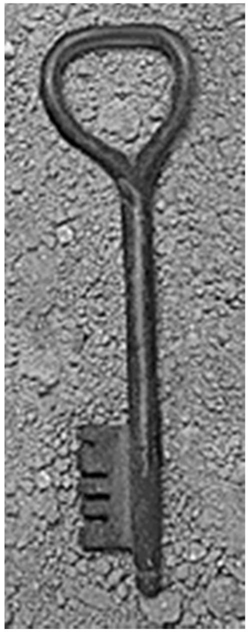
R W X	R X	R
1 1 1	1 0 1	1 0 0
7	5	4



Issues

节省了空间，但是权限表达不是那么精准。

- We have traded *space* reduction and speed of checking for reduced flexibility in expressing an access control policy
- Some operating systems (DEC VMS, some variants of HPUX) allow these abbreviated access lists to be augmented by additional entries - such as the ability to add privileges 特权 for a specific user or more than one group.

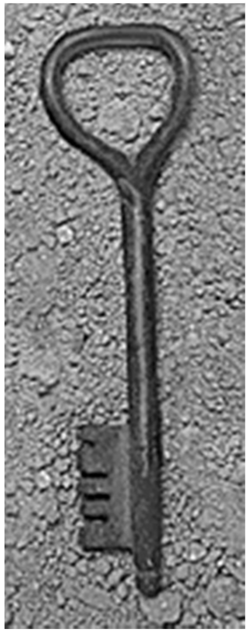


Capabilities ?

Least Privilege

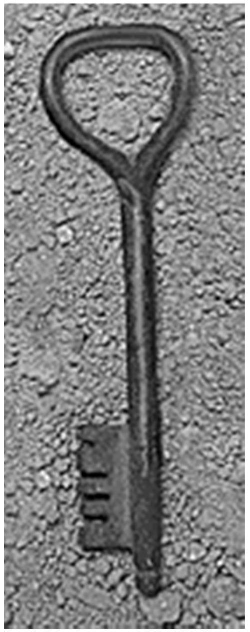
高安全系统，情报系统：最小特权。
完成一个任务需要的最小的权限和知识。

- Capabilities provide for finer grained least privilege control with respect to subjects, especially dynamic short-lived subjects created for specific tasks



Agenda

- Access matrix model
- Access control lists versus Capabilities
- Discretionary versus mandatory controls
  谨慎的，自主的。 强制
- Bell-LaPadula model
- Covert channels
- Beyond MAC and DAC



Discretionary vs Mandatory Access Controls

- Discretionary Access Controls (DAC) allow access rights to be propagated from one subject to another

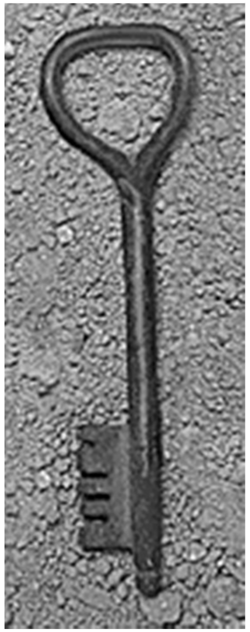
主人可以自主决定让谁看。

Possession of an access right by a subject is sufficient to allow access to the object

- Mandatory Access Controls (MAC) restrict the access of subjects to objects on the basis of security labels

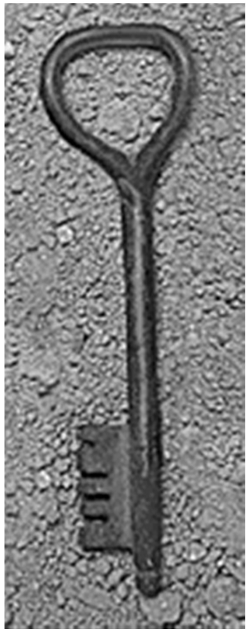
强制访问中文件权限受限于安全等级。

核专家想把核武器资料复制给他老婆，系统会拒绝，因为其老婆安全级别不够。



Inherent Weakness of DAC

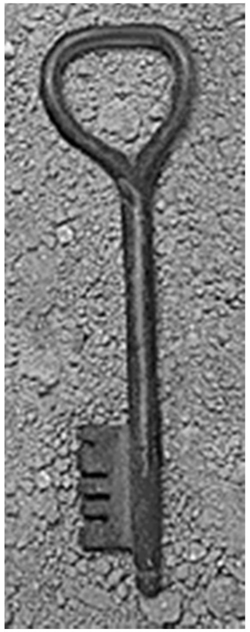
- Unrestricted DAC allows information from an object which can be read to any other object which can be written by a subject
- Suppose our users are trusted not to do this deliberately. It is still possible for Trojan Horses to copy information from one object to another.



Trojan Horses

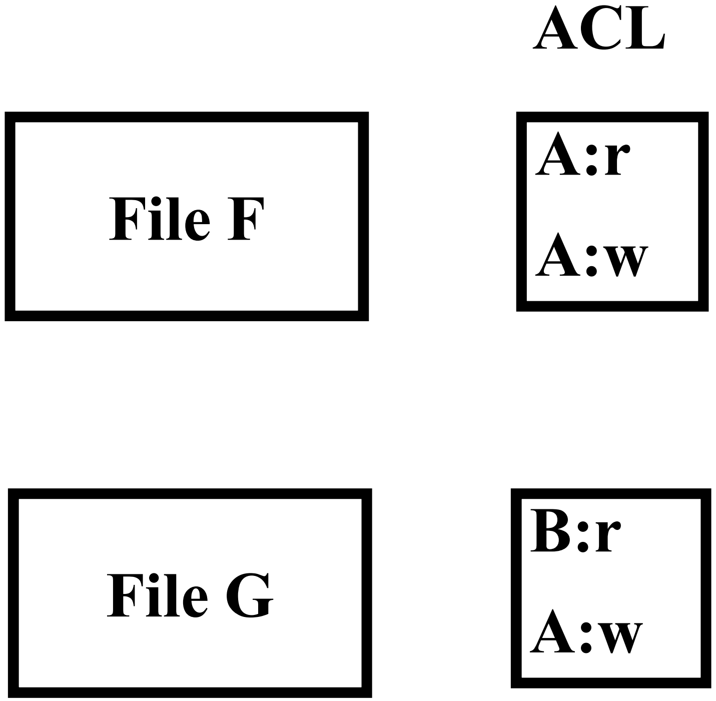
特洛伊木马，把自己伪装成好的东西，诱使主人去做。

- A Trojan Horse is rogue software installed, perhaps unwittingly, by duly authorized users
- A Trojan Horse does what a user expects it to do, but in addition exploits the user's legitimate privileges to cause a security breach



Trojan Horse Example

自主访问系统，B不能读文件F，可以先让A把文件F内容写入文件G，这样B就可以读到F的内容。



ACL

File F

A:r

A:w

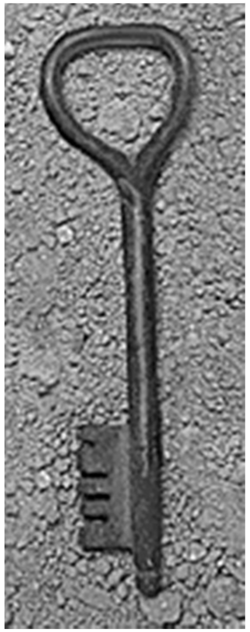
File G

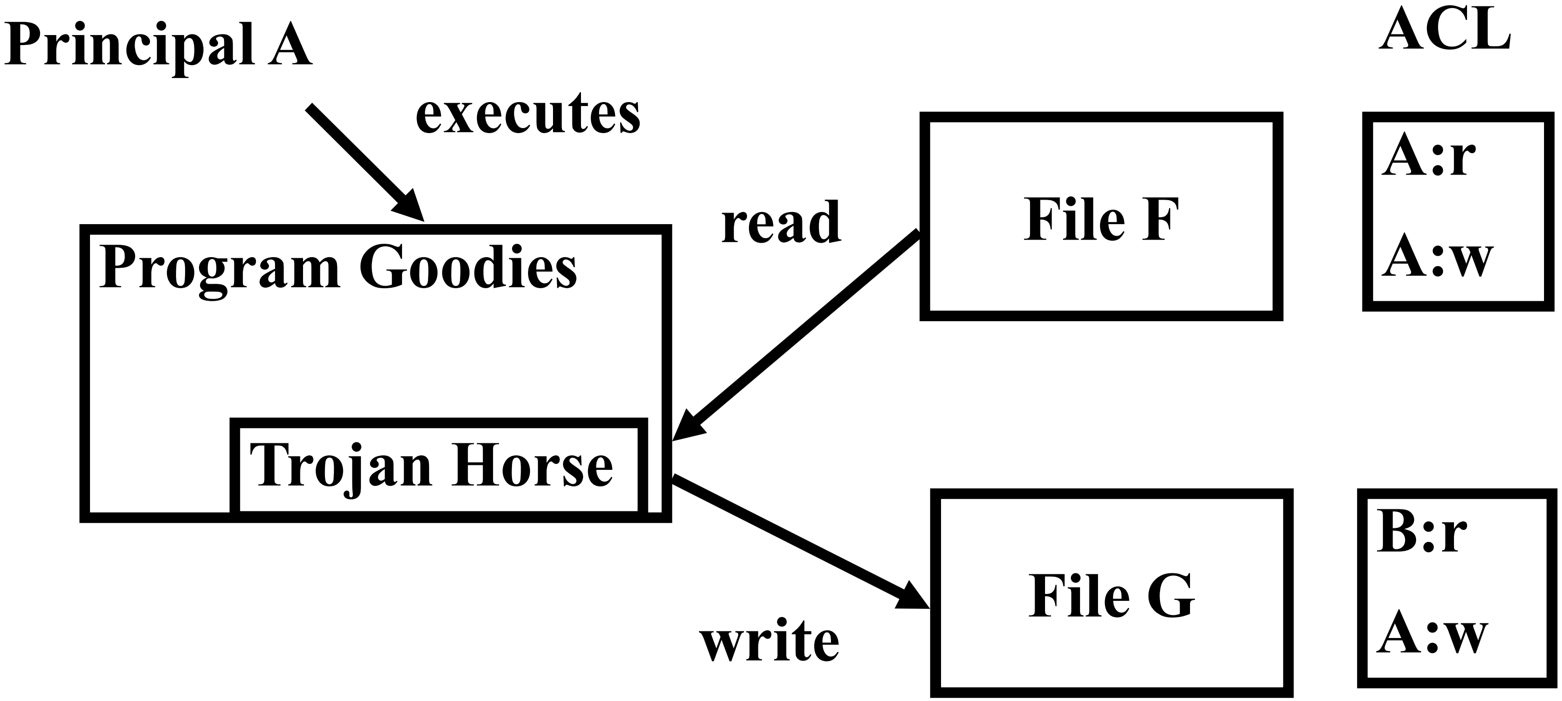
B:r

A:w

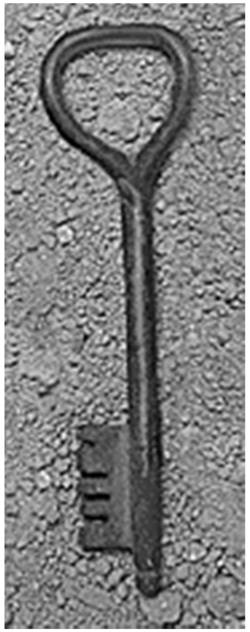
Principal B cannot read file F

Trojan Horse Example



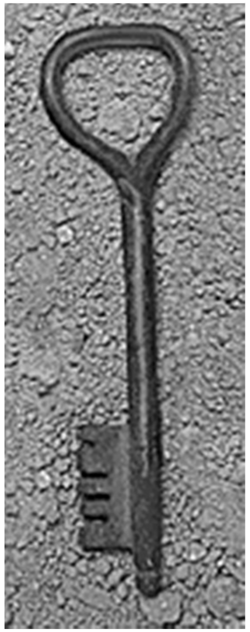


Principal B can read contents of file F copied to file G



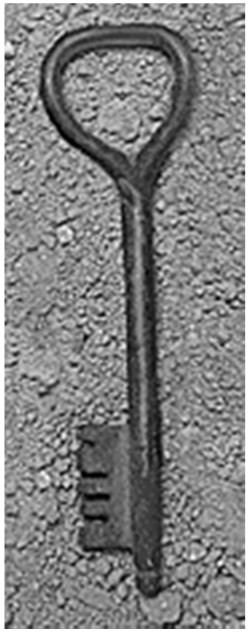
Trojan Horses – very hard to handle

- Trojan Horses are the most insidious threat
- Viruses and logic bombs are examples of Trojan Horses
- It is possible to embed Trojan Horses in hardware and firmware (Ex. Printer Virus in Yugoslavia War)
- It is possible to embed Trojan Horses in critical system software such as compilers and Database Management Systems (Ex. The Thompson’s C Compiler)



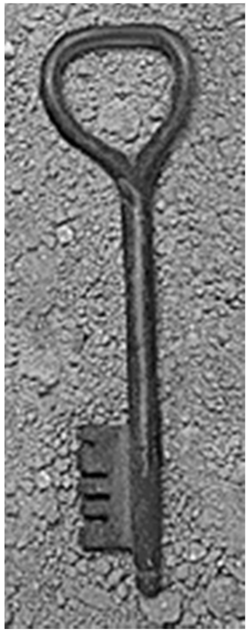
Agenda

- Access matrix model
- Access control lists versus Capabilities
- Discretionary versus mandatory controls
- Bell-LaPadula model 强制访问控制最有名的模型 BLP模型
- Covert channels
- Beyond MAC and DAC



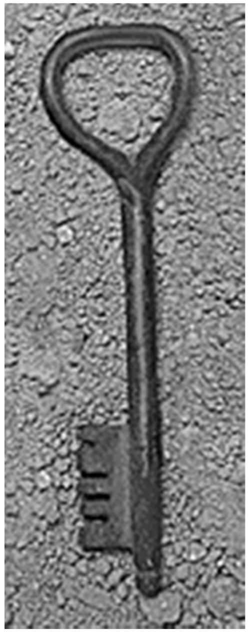
Mandatory Access Control

- Goal: prevent the unauthorized disclosure of information
  - Usually used in Military Applications
  - Deals with the “Hidden” information flow
- Multi-level security models are best-known examples 多级安全的模型
  - Bell-LaPadula Model (BLP) basis for many, or most, of these

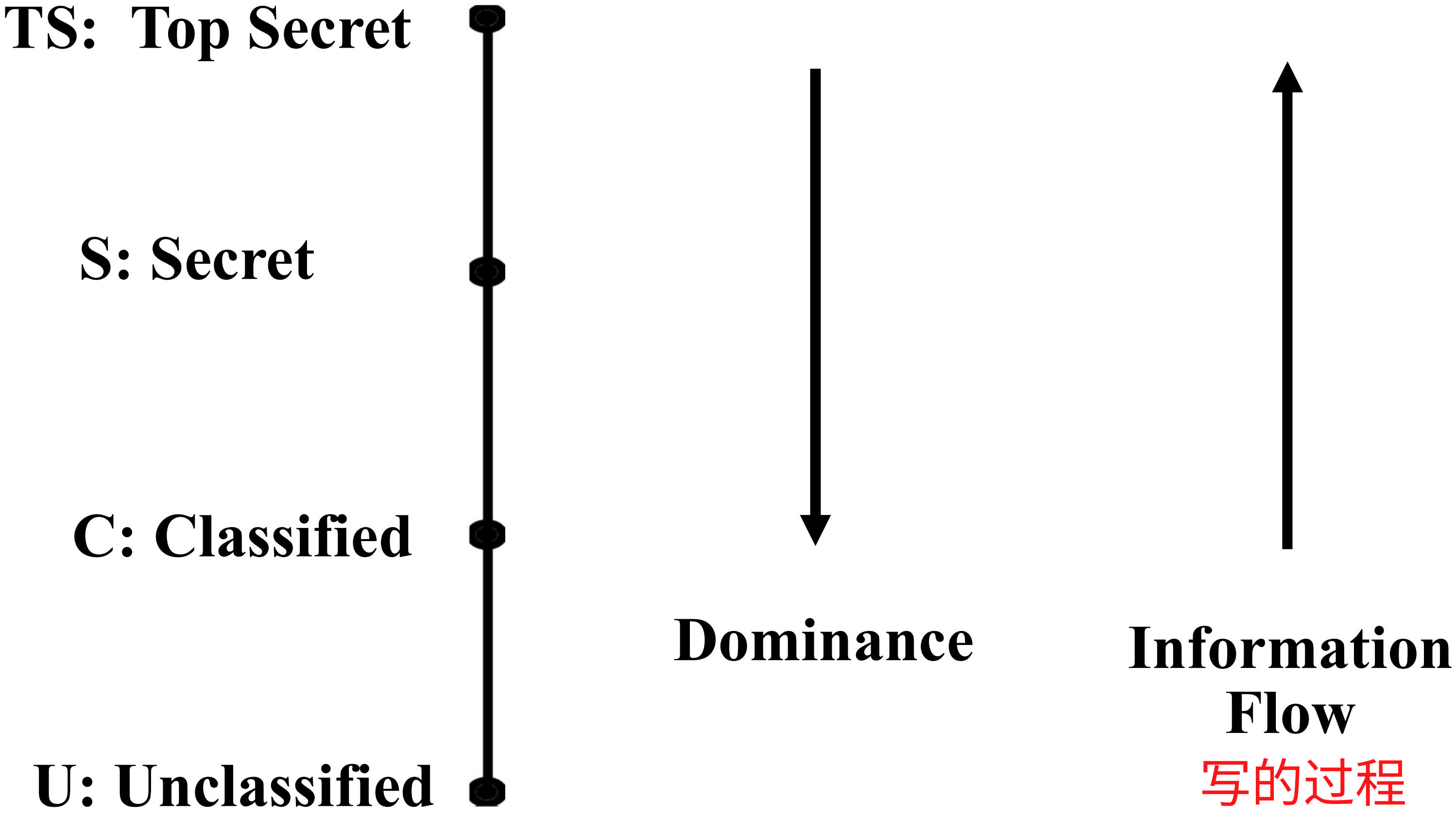


Bell-LaPadula Model, Step 1

- Security levels arranged in linear ordering
  - Top Secret: highest
  - Secret
  - Confidential
  - Unclassified: lowest
- Levels consist of *security clearance* $L(s)$
  - Objects have *security classification* $L(o)$



Multilevel Security



TS: Top Secret

S: Secret

C: Classified

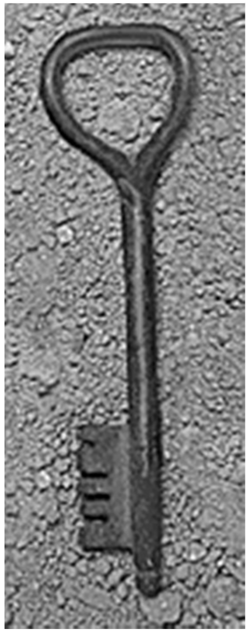
U: Unclassified

Dominance

Information Flow

写的过程

Lattice of security labels

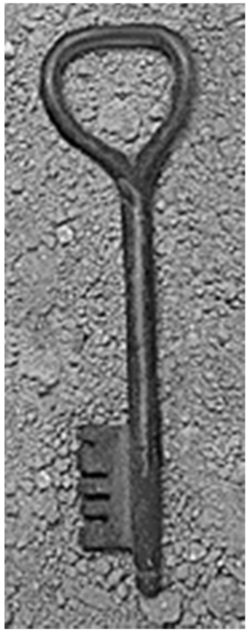


Example

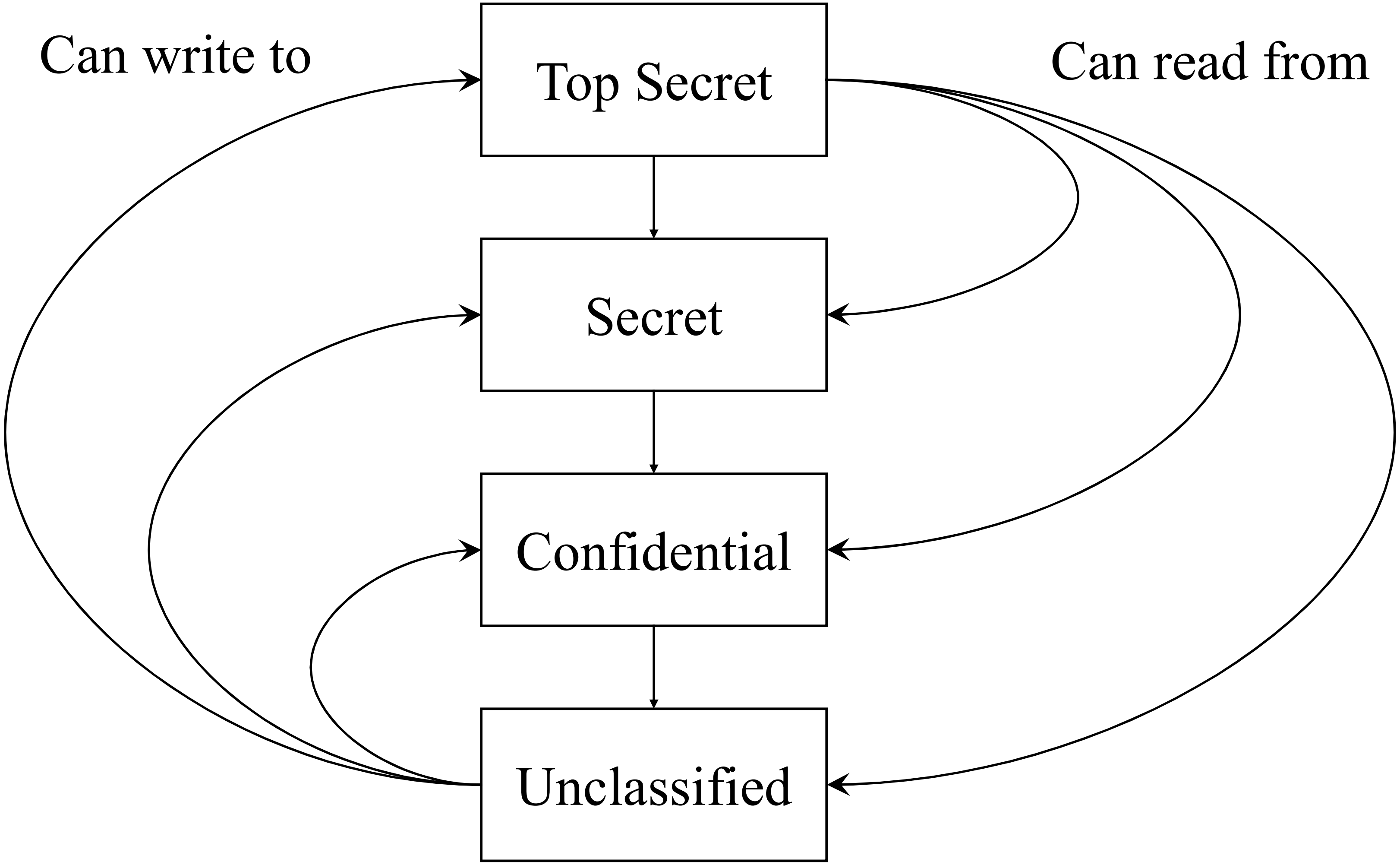
top secret不能写一个文件，所有人都能读。防止特洛伊木马。

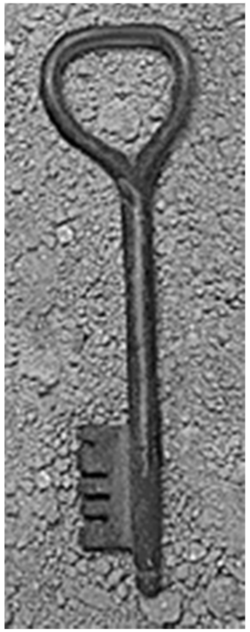
security level	*subject*	*object*
Top Secret	Tamara	Nuclear Files
Secret	Samuel	Battleship Files
Confidential	Claire	Logistics Files
Unclassified	Alice	Telephone Lists

- Tamara can read all files
- Claire cannot read Nuclear or Battleship Files
- Alice can only read Telephone Lists



Linear Access Level Lists

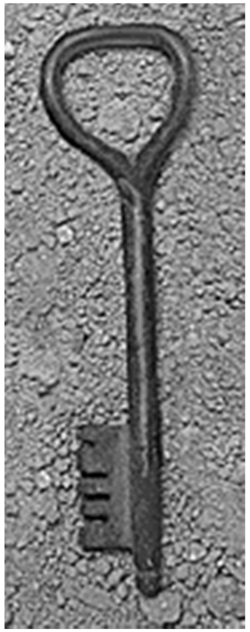




Reading Information

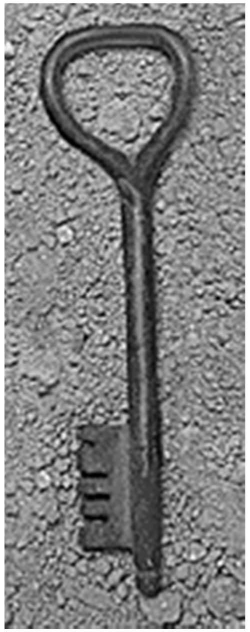
只能往上写，往下读。

- Information flows *up*, not *down*
  - "Reads up" disallowed, "reads down" allowed
- Simple Security Condition (Step 1)
  - Subject *s* can read object *o* iff, $L(o) \le L(s)$ and *s* has permission to read *o*
    - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
  - Sometimes called "no reads up" rule



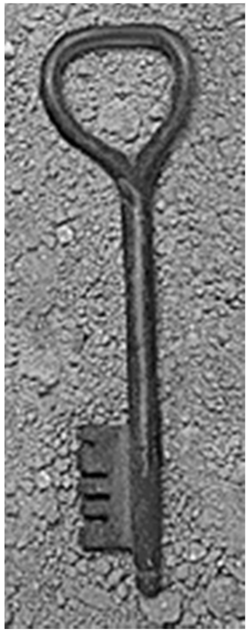
Writing Information

- ♦ Information flows up, not down
  - – "Writes up" allowed, "writes down" disallowed
- ♦ *-Property (Step 1)
  - – Subject *s* can write object *o* iff $L(s) \leq L(o)$ and *s* has permission to write *o*
    - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
  - – Sometimes called "no writes down" rule



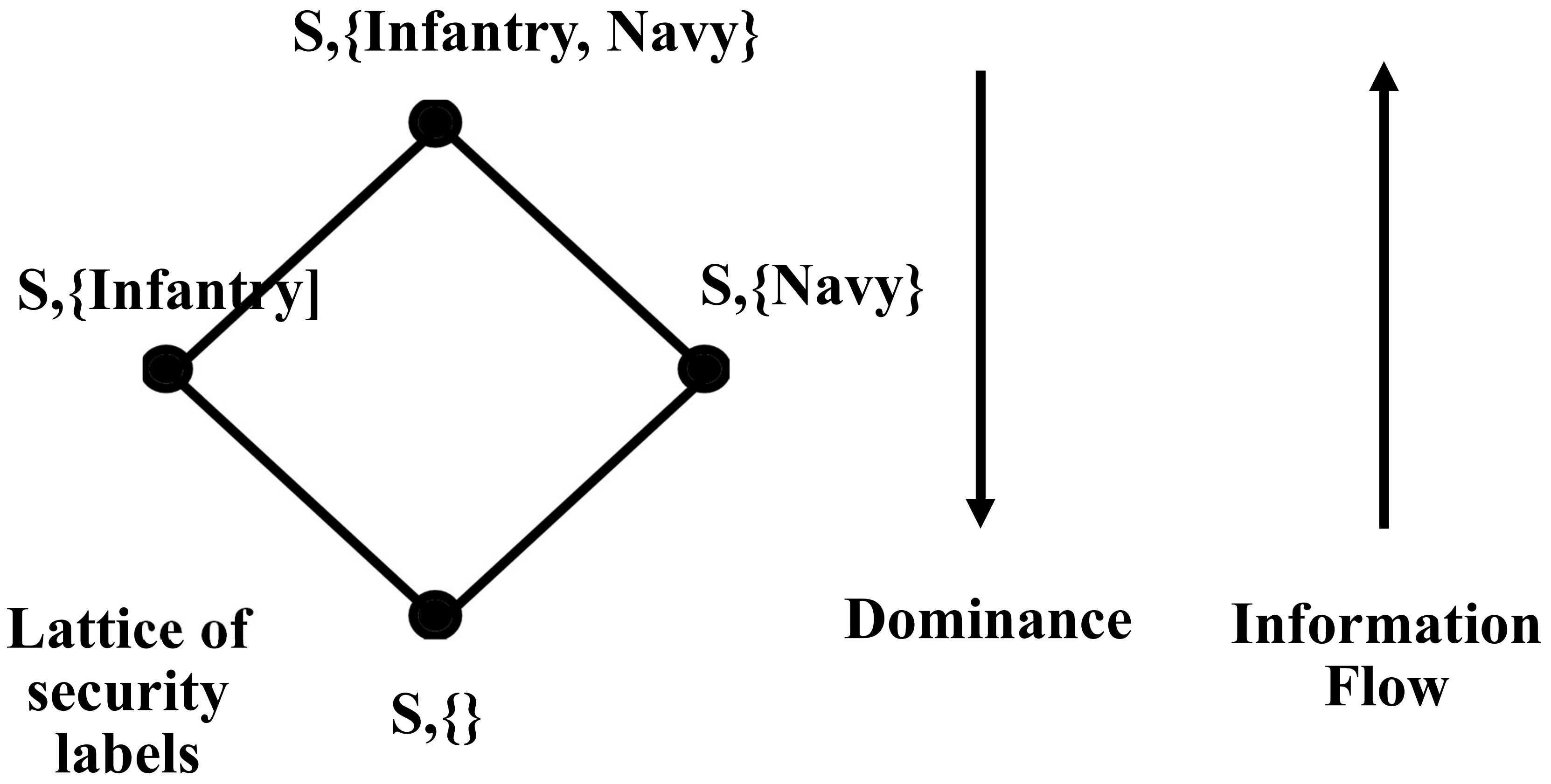
Bell-LaPadula Model, Step 2

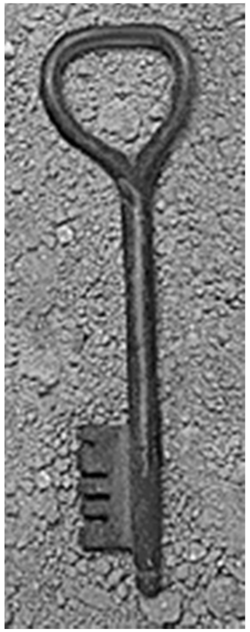
- ♦ Expand notion of security level to include categories 增加了类别。
- ♦ Security level is (*clearance*, *category set*) 类别集合
- ♦ Examples
  - – (Top Secret, { General Staff, Infantry, Navy })
  - – (Confidential, { General Staff, Infantry })
  - – (Secret, { General Staff, Navy })
  - – (Unclassified, { Navy })



MULTILEVEL SECURITY

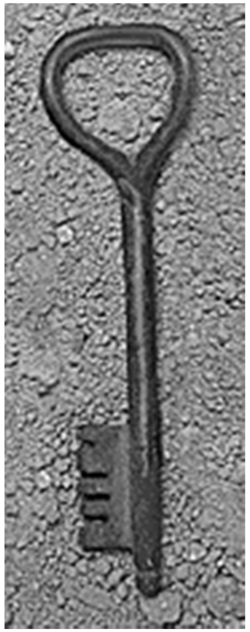
格状，线性有序。





Agenda

- ♦ Access matrix model
- ♦ Access control lists versus Capabilities
- ♦ Discretionary versus mandatory controls
- ♦ Bell-LaPadula model
- ♦ Covert channels
- ♦ Beyond MAC and DAC



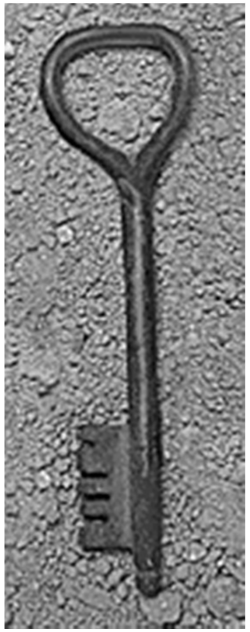
Covert Channels

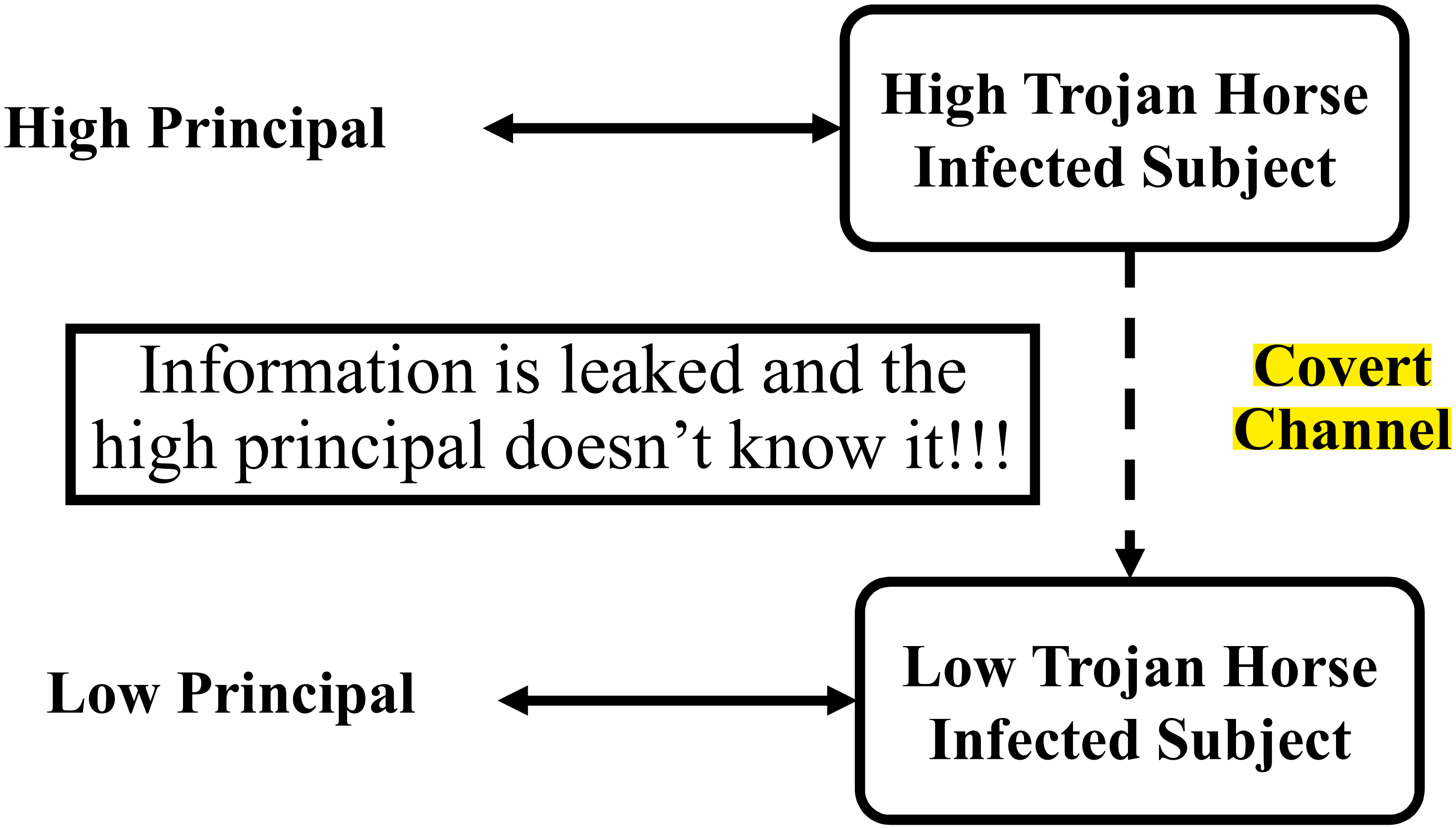
隐通道

- A covert channel is a communication channel based on the use of system resources not normally intended for communication between the subjects (processes) in the system

Covert Channels

当有噪声时，增加校验码。





High Principal

High Trojan Horse Infected Subject

Information is leaked and the high principal doesn’t know it!!!

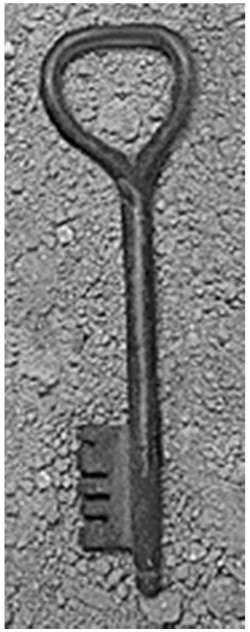
Covert Channel

Low Principal

Low Trojan Horse Infected Subject

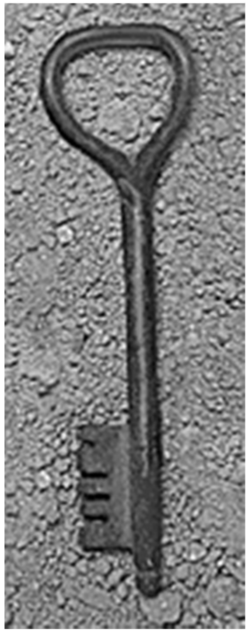
用多个特洛伊木马

隐蔽的通道



Covert Channels

- The concern is with subjects not users
  - users are trusted (must be trusted) not to disclose secret information outside of the computer system
  - subjects are not trusted because they may have Trojan Horses embedded in the code they execute
- *-property prevents overt leakage of information and does not address the covert channel problem



Resource Exhaustion Channel

资源耗尽型的通道

Given 5MB pool of dynamically allocated memory

High-Level Process

bit = 1 ⇒ request 5MB of memory

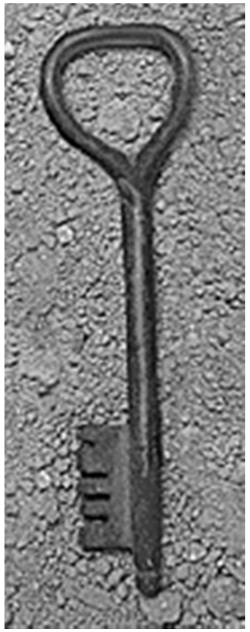
bit = 0 ⇒ request 0MB of memory

Low-Level Process

当资源用光，收到1
当资源为0，收到0。

request 5MB of memory

if allocated then bit = 0 otherwise bit = 1



Load Sensing Channel

负载敏感型通道

High-Level Process

让CPU的负载发生改变。

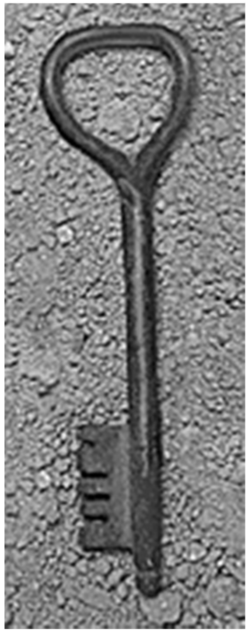
bit = 1 ⇒ enter computation intensive loop

bit = 0 ⇒ go to sleep

Low-Level Process

perform a task with known computational requirements

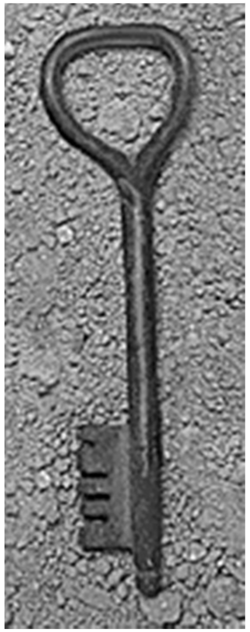
if completed quickly then bit = 0 otherwise bit = 1



Coping with Covert Channels

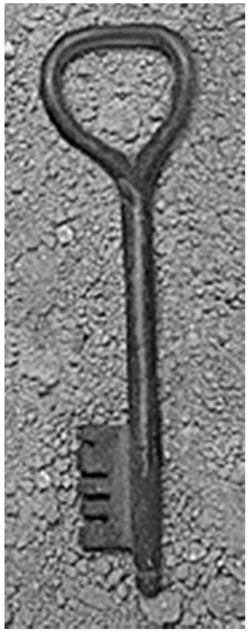
对抗隐通道：带宽，不共享硬件资源。

- After Identification
  - close the channel or slow it down
  - detect attempts to use the channel
  - tolerate its existence



Orange Book Criteria (TCSEC)

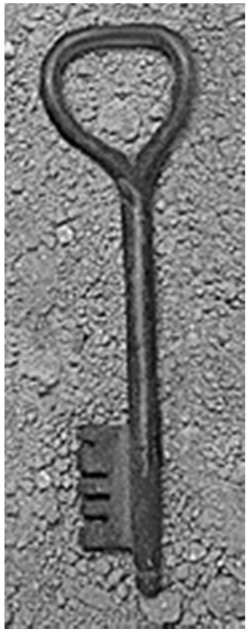
- Level D DOS
  - No security requirements
- Level C
  - Discretionary Access Control (DAC)
- Level B, A
  - Must enforce Bell-LaPadula model (MAC)



Orange Book Criteria (TCSEC)

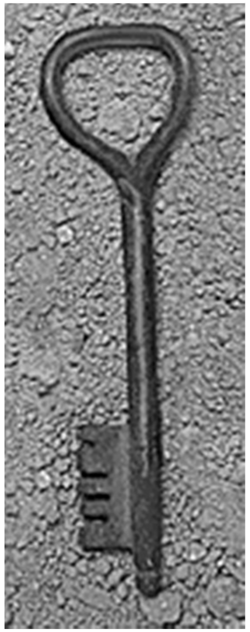
商用系统

- C1
  - Cooperating users at same level of sensitivity
  - Access control; users can protect their own data
  - Discretionary access control
- C2
  - Finer granularity of control
  - Better audit functions; each individual access to each object can be tracked



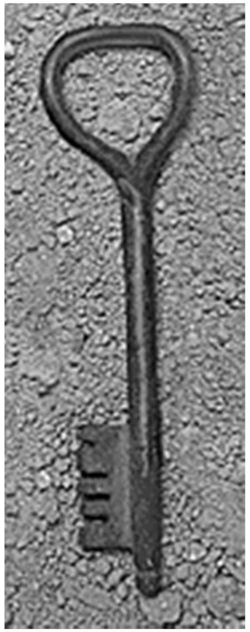
Orange Book Criteria (TCSEC)

- B1
  - Non-discretionary access control; subjects and (most) objects assigned a security level
  - Bell-LaPadula model + DAC to further limit access
- B2
  - Independent modules
  - Design and implementation go through more thorough review/testing based on verifiable top-level design
  - Principle of least privilege 最小特权，能力列表。



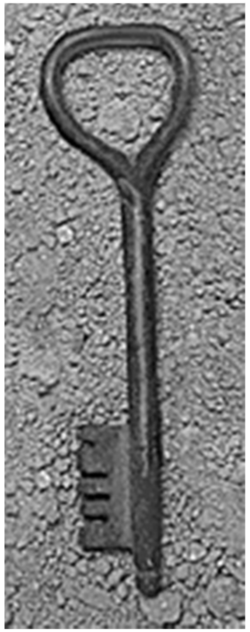
Orange Book Criteria (TCSEC)

- B3
  - Security functions small enough for extensive testing/review and tamperproof
- A1
  - Verifiable design
  - Formal model and proof of consistency



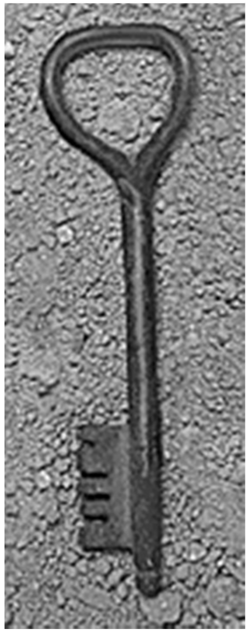
Covert Channels and the Orange Book

C2	No labels
B1	Labels with Bell-LaPadula controls, but no need to address covert channels
B2	Must address storage channels (such as resource exhaustion channel)
B3	Must also address timing channels (such as load sensing channel)
A1	Must use formal techniques (where available)



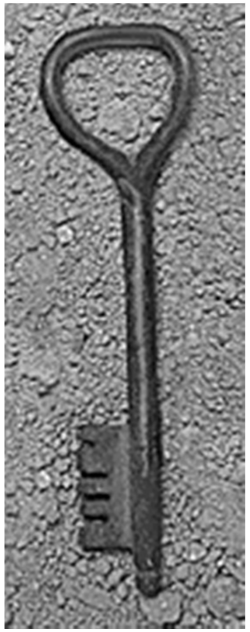
Agenda

- Access matrix model
- Access control lists versus Capabilities
- Discretionary versus mandatory controls
- Bell-LaPadula model
- Covert channels
- Beyond MAC and DAC

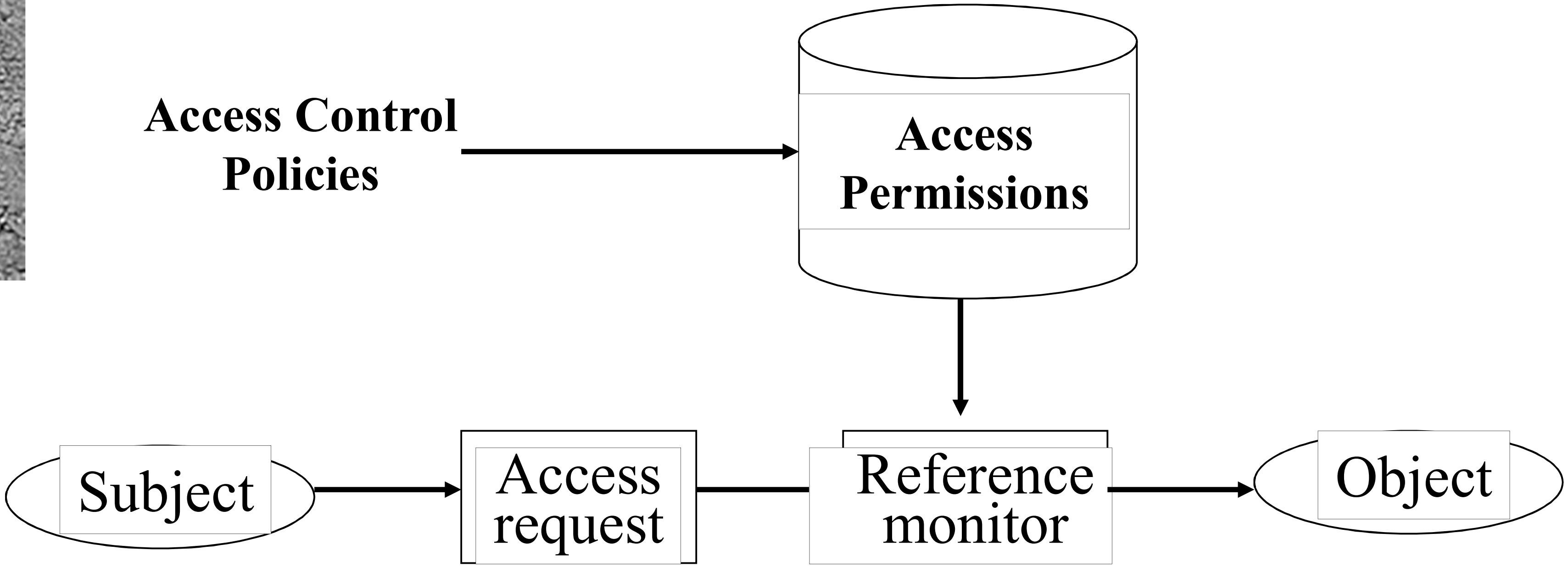


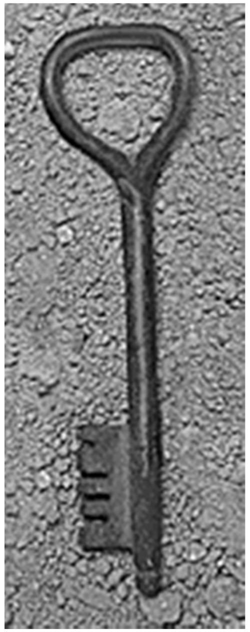
Beyond MAC/DAC

- ♦ DAC and MAC are extreme points of a continuum of access controls
- ♦ There are legitimate policies that fall in between, for example:
  - Document release: a document cannot be released by a scientist without first obtaining approvals from a patent-officer and a security-officer
  - Originator control: information in an object should not be propagated without permission of the owner of the object



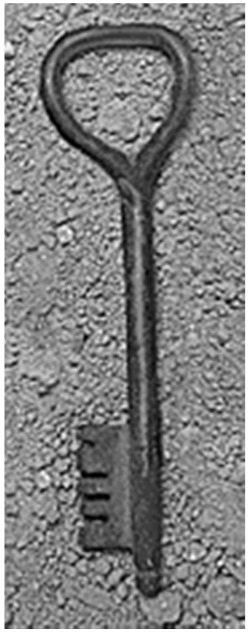
Access Control Policies





Content Dependent Access Control

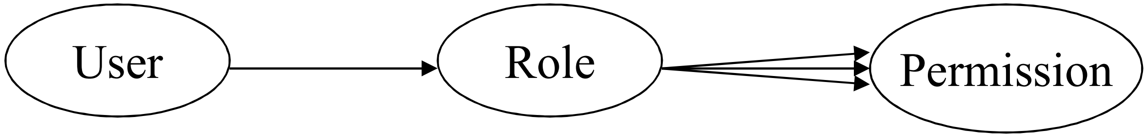
- ♦ Content dependent controls such as
  - you can only see salaries less than 50K, or
  - you can only see salaries of employees who report to you
- ♦ Beyond the scope of Operating Systems and usually are provided by Database Management Systems



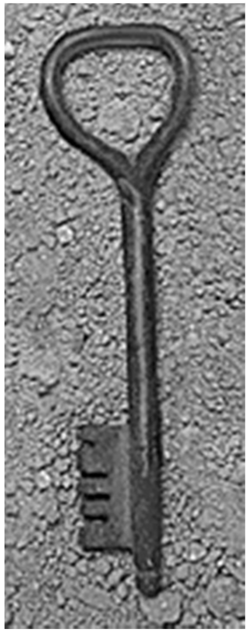
Role-Based Access Control

- Sandu et al. formalized Role-Based Access Control in 1996

基于角色的权限控制

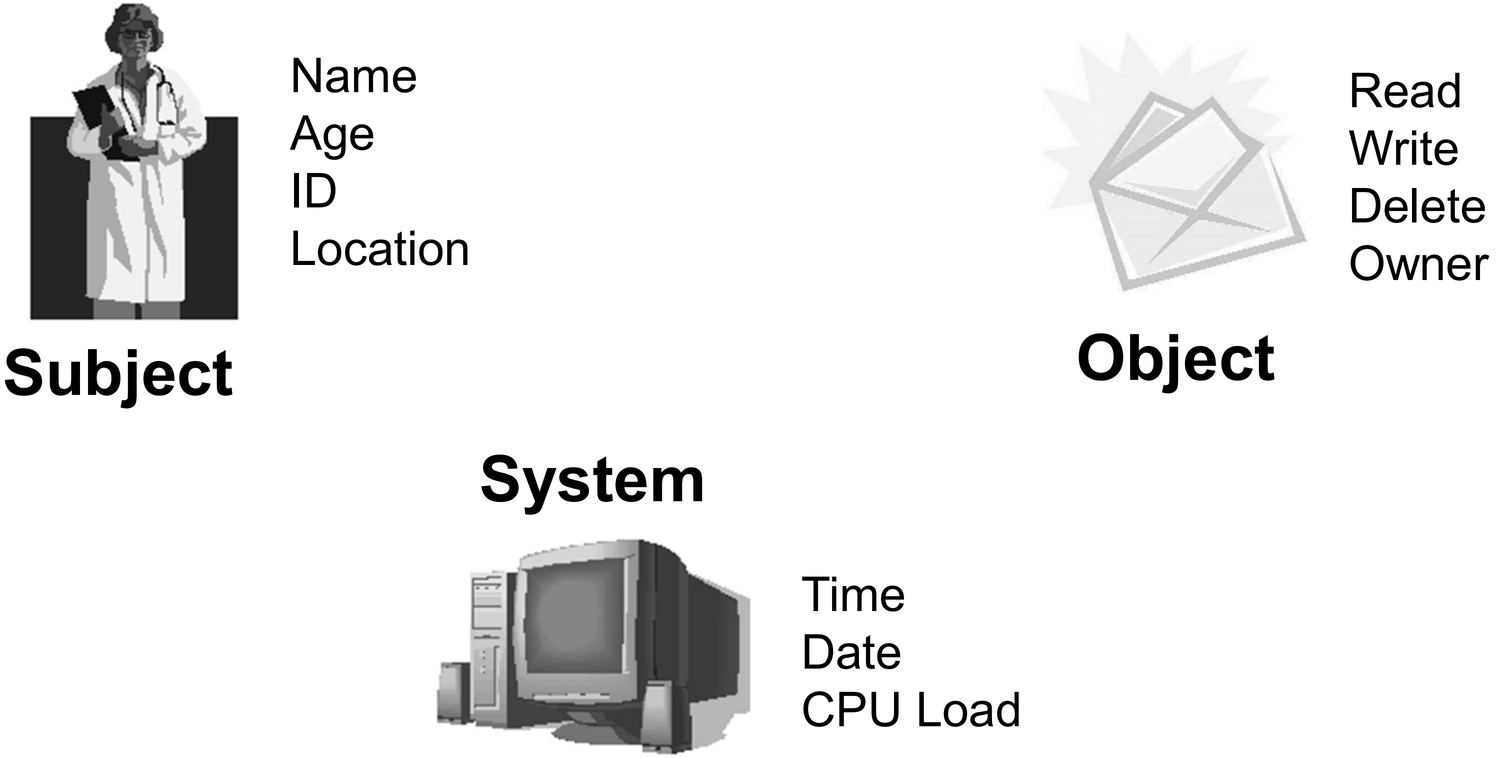


- User U acting in role R is granted permission P
  - Advantage: greatly improved efficiency
  - Disadvantage: cannot specify fine-grained rules



Context-Based Access Control

- What is "context"?
  - Circumstances in which an event occurs



Name
Age
ID
Location

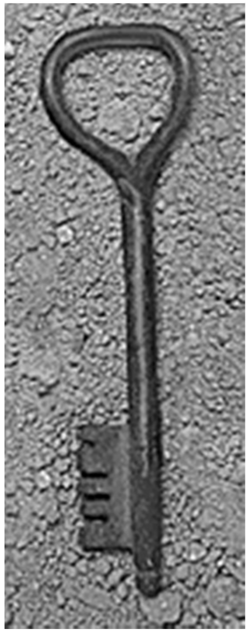
Subject

Read
Write
Delete
Owner

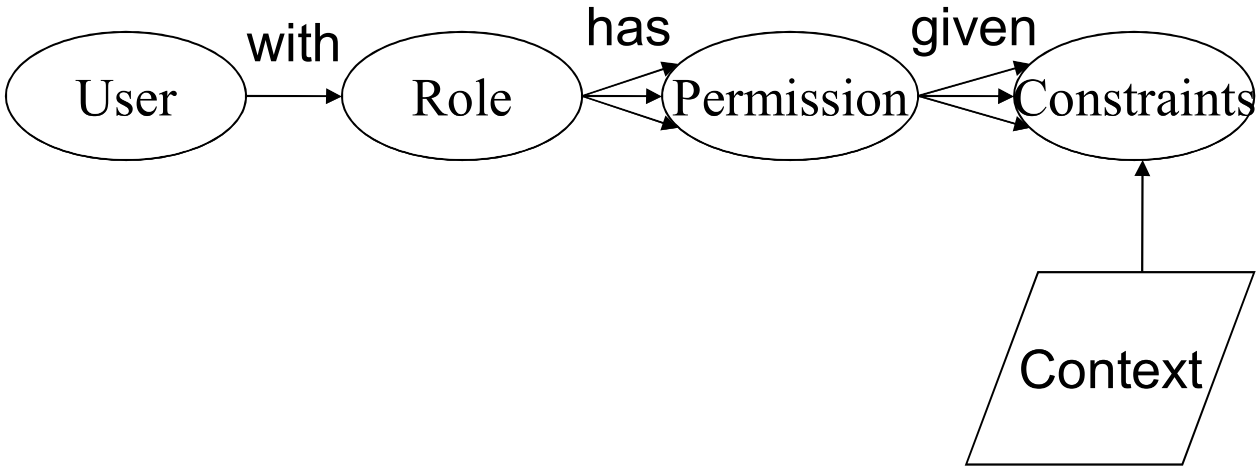
Object

System

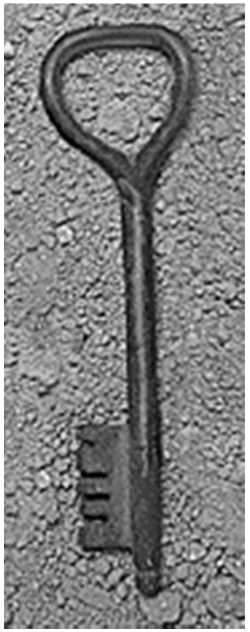
Time
Date
CPU Load



Context-Based Access Control

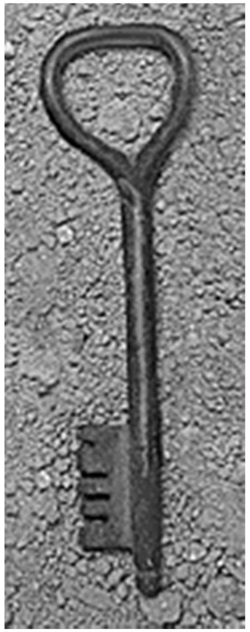


- ♦ Advantage: access control is context-aware
- ♦ Disadvantage: this is still a static model



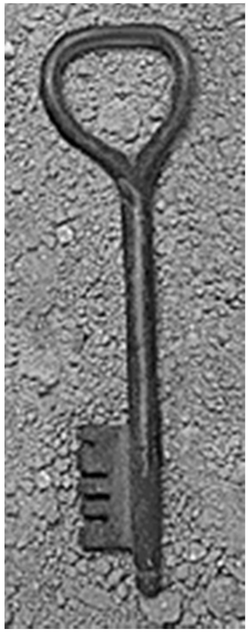
Context-Based Access Control

- Examples 1: Cannot access classified information via a remote login
- Examples 2: Salary information can be updated only at year end
- Examples 3: Company's earnings report is confidential until announced at the stockholders meeting
- can be partially provided by the Operating System and partially by the Database Management System
- more sophisticated context dependent controls such as based on past history of accesses definitely require Database support



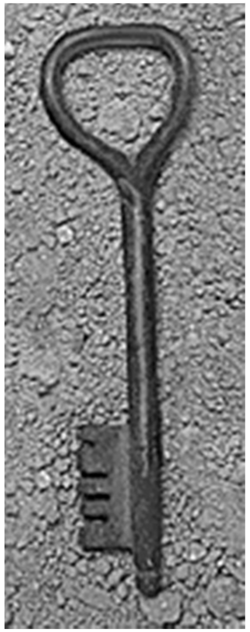
What Else Might We Add?

- ♦ Default Rule (Telephone Sys. Example)
  - – General default: Receive
  - – Object default: Call Internal
- ♦ Time-based access
  - – Allow long distance call after hours?
- ♦ History-based access



Access Control by History

- Example: Statistical Database
  - Allows queries for general statistics
  - But not individual values
- Valid queries: Statistics on 20+ individuals
  - Total salary of all Deans
  - Salary of Computer Science Professors
- See a problem coming?
  - Salary of CS Professors who aren't Deans



Solution: Query Set Overlap Control (Dobkin, Jones & Lipton '79)

- Query valid if intersection of query coverage and each previous query < *r*
- Given *K* minimum query size, r overlap:
  - Need *1 + (K-1)/r* queries to compromise
- Can represent as access control matrix
  - Subjects: entities issuing queries
  - Objects: *Powerset* of records
  - $O_s(i)$: objects referenced by *s* in queries *1..i*
  - $A[s,o]$ = read iff $\forall_{q \in O_s(i-1)} |q \cap o| < r$