Trusted Systems

Presentation by:
V. Balasubramanian
SSN College of Engineering

Introduction

- Trusted Syatem Technology: the ability of a system to defend against intruders and malicious programs.
- Computer security means protecting our computer systems and the information they contain against unwanted access, damage, destruction or modification
- Computer and network security are built on three pillars, commonly referred to by the C-I-A acronym

Three Basic security Triad

- Confidentiality: the concealment of information or resources.
- Integrity: the trustworthiness of data and resources
- Availability: the ability to use the information or resources desired.

Data Access Control

- After successful logon, the user has been granted access to one or a set of hosts and applications.
- It is not sufficient for a system that includes sensitive data in its database

Contd...

- Associated with each user, there can be a profile that specifies permissible operations and file accesses. The operating system can then enforce rules based on the user profile.

Dilbert on Physical Security

I LOST MY I.D. BADGE.
REPORT TO THE SECURITY OFFICE AND GET A NEW ONE.
HOLD IT! WHERE DO YOU THINK YOU'RE GOING?
TO THE SECURITY OFFICE?
NO ONE IS ALLOWED PAST THIS DESK WITHOUT AN I.D. BADGE.
OKAY...HOW DO I GO TO THE SECURITY OFFICE IF I CAN'T GO TO THE SECURITY OFFICE?
GOOD QUESTION. I GUESS YOU'LL HAVE TO SNEAK PAST ME.
LOOK OVER THERE! IT'S AN ANGEL, AND SHE'S GIVING AWAY FREE BACON!
WELL, WELL; IT SEEMS I'VE FOUND A WORTHY ADVERSARY.
E-mail: SCOTTADAMS@AOL.COM
© 2004 Scott Adams, Inc./Dist. by UFS, Inc.
www.dilbert.com

Protection State

- The state of a system is the collection of the current values of all memory locations, all secondary storage, and all registers and other components of the system. The subset of this collection that deals with protection is the protection state of the system.

Access Control List

- An access control matrix is one tool that can describe the current protection state.
- Consider the set of possible protection states P. Some subset Q of P consists of exactly those states in which the system is authorized to reside. So, whenever the system state is in Q, the system is secure. When the current state is in P – Q, the system is not secure.

Contd…

- Access Control Matrix is the most precise model used to describe a protection state.
- It characterizes the rights of each *subject (active entity, such as a process) with respect to* every other entity.

Contd…

- Operations are reading, loading, altering etc.
- program that changes a variable to 0 does not (usually) alter the protection state.
- variable altered is one that affects the privileges of a process, then protection state might be changed.

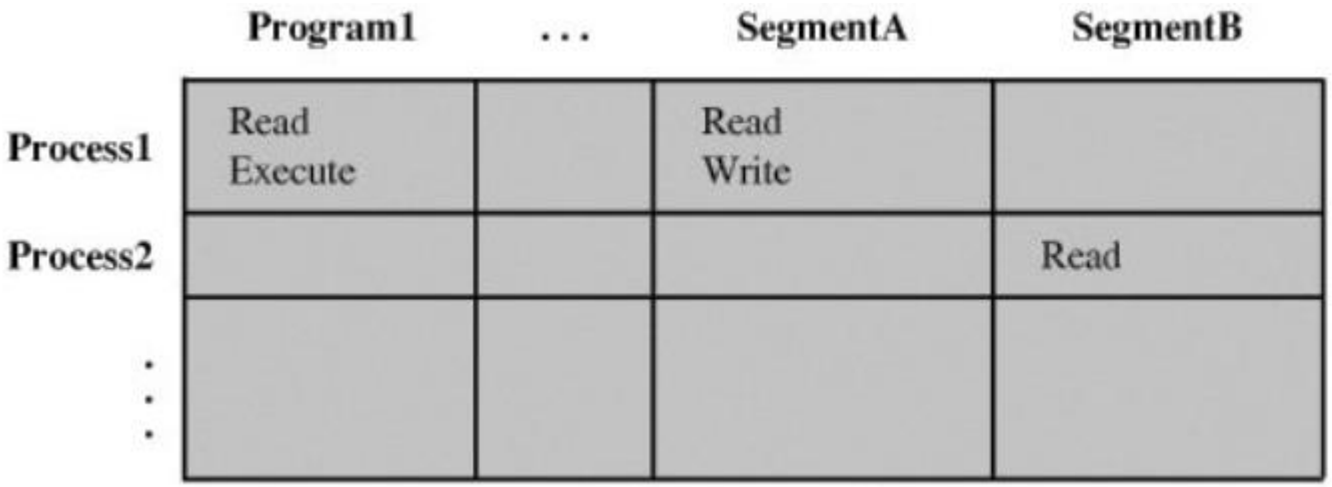
Access Control Matrix

- Butler Lampson -1971
- Graham and Denning refined it.
- The set of all protected entities is called the set of *objects O.*
- The set of *subjects S is the set of active objects, such as* processes and users
- Access Rights: R, W, X

Contd…

- access control matrix model, the relationship between these entities is captured by a matrix *A with rights drawn from a set of rights R in each entry a[s, o]*

	file 1	file 2	process 1	process 2
process 1	read, write, own	read	read, write, execute, own	write
process 2	append	read, own	read	read, write, execute, own



	Program1	. . .	SegmentA	SegmentB
Process1	Read Execute		Read Write	
Process2				Read
. . .				

(a) Access matrix

- The UNIX system defines the rights "read," "write," and "execute."
- "read" means to be able to list the contents of the directory;
- "write" means to be able to create, rename, or delete files or subdirectories in that directory; and
- "execute" means to be able to access files or subdirectories in that directory.

Contd…

- When a process accesses another process, "read" means to be able to
- receive signals, "write" means to be able to send signals, and "execute" means to be able to execute the process as a subprocess.

- the "objects" involved in the access control matrix are normally thought of as files, devices, and processes, they could just as easily be messages sent between processes, or indeed systems themselves

Access Control Matrix

host names	*telegraph*	*nob*	*toadflax*
telegraph	own	ftp	ftp
nob		ftp, nfs, mail, own	ftp, nfs, mail
toadflax		ftp, mail	ftp, nfs, mail, own

	OS	Accounting Program	Accounting Data	Insurance Data	Payroll Data
Bob	rx	rx	r	-	-
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Acct. program	rx	rx	rw	rw	rw

Access Control List (ACL)

In practical, the system should better not to assign numerous numbers of objects and subjects in a large access control matrix. With a large amount of matrix entries and extreme scattering of data [2], the large access control matrix would waste too much memory space (e.g. 10,000 subjects x 1,000,000 objects = 1,000,000,000 matrix entries). It also takes quite a while to check and pair between any subject and any object.

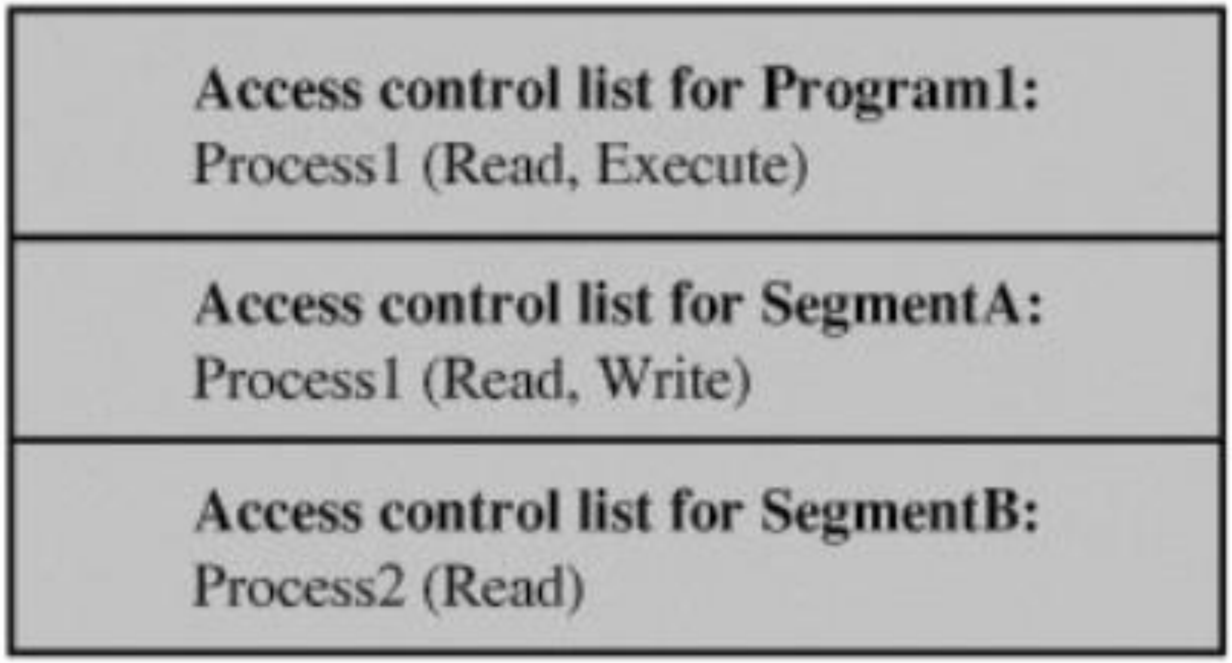
For better performance of authorization operation, the access control matrix is split into two doable options or derivatives. The first is access control list and the later is capability. The matrix is split into columns which represents objects. These columns are called "access control lists" (ACLs). An ACL acts for a column in the access control matrix. ACL is attached to an object and specifies its related subjects.

Access Control List

- an access matrix is usually sparse and is implemented by decomposition in one of two ways.
- The matrix may be decomposed by columns, yielding **access control lists**

- The matrix is split into columns which represents objects. These columns are called “access control lists” (ACLs).

Control



Access control list for Program1:
Process1 (Read, Execute)

Access control list for SegmentA:
Process1 (Read, Write)

Access control list for SegmentB:
Process2 (Read)

(b) Access control list

Access Control Lists (ACLs) cont.

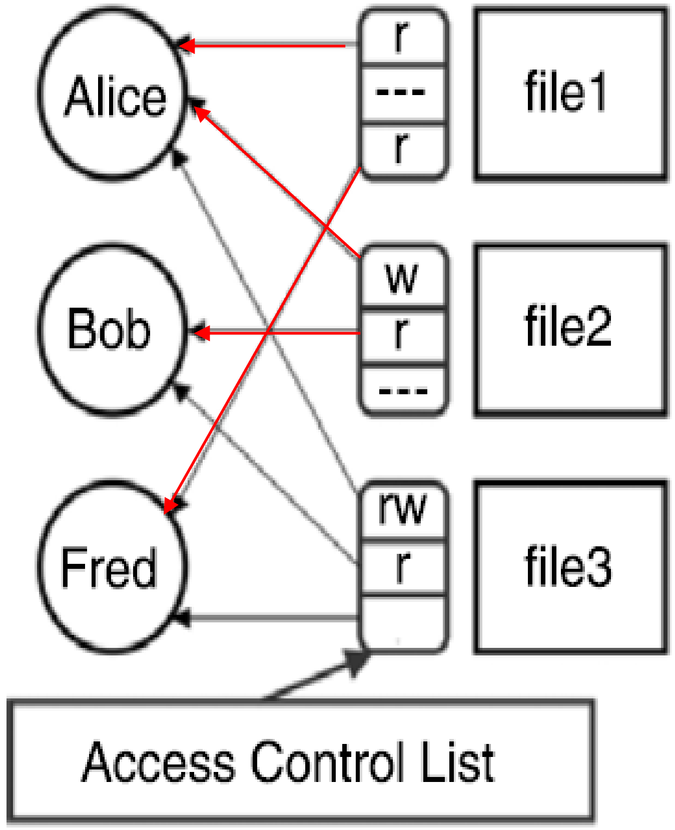


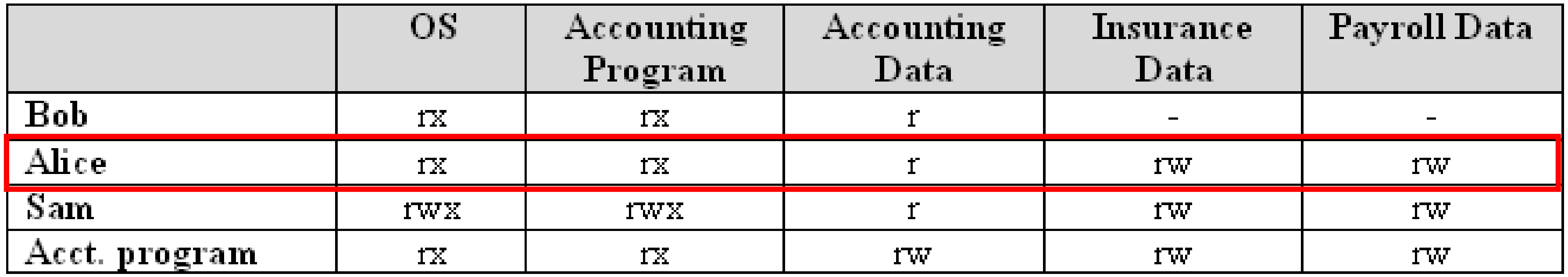
Figure 1: Access Control Lists (ACLs)

From the figure 1, there are three subjects (Alice, Bob and Fred) and three objects (file1, file 2, and file 3). Each object is set for particular subject permissions. According to figure 1, each file or object has its own ACL. File 1 links to two subjects that are Alice and Fred. File 1 allows a permission to read for both subjects. For ACL of file 2, it allows a permission to write file 2 for Alice and a permission to read file 2 for Bob.

Capability List (C-List)

The second derivative of access control matrix is "capability list or C-list". In this case, the access control matrix is spilt into rows, each row represent one subject. A capability list is attached to a subject and specifies its related objects. Each entry in the list is a capability which is a pair of object and a set of access operations. Permissions to access objects for each subject are listed in each C-list.

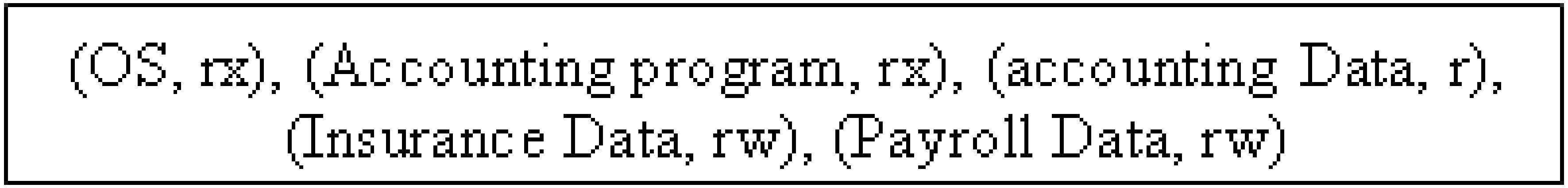
Capability List (C-List) cont.6



	OS	Accounting Program	Accounting Data	Insurance Data	Payroll Data
Bob	rx	rx	r	-	-
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Acct. program	rx	rx	rw	rw	rw

Table 1: Access Control Matrix [3]

From access control matrix, if we look for Alice's C-list we can write:



(OS, rx), (Accounting program, rx), (accounting Data, r), (Insurance Data, rw), (Payroll Data, rw)

Capability (C-List) cont.

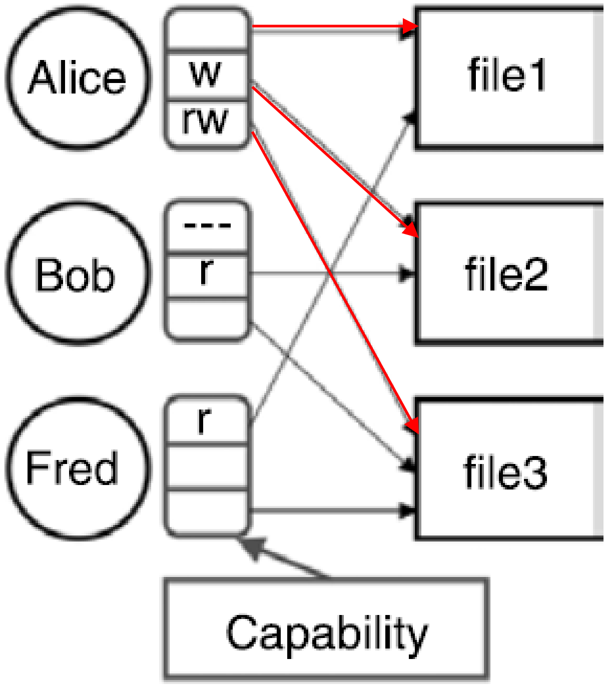
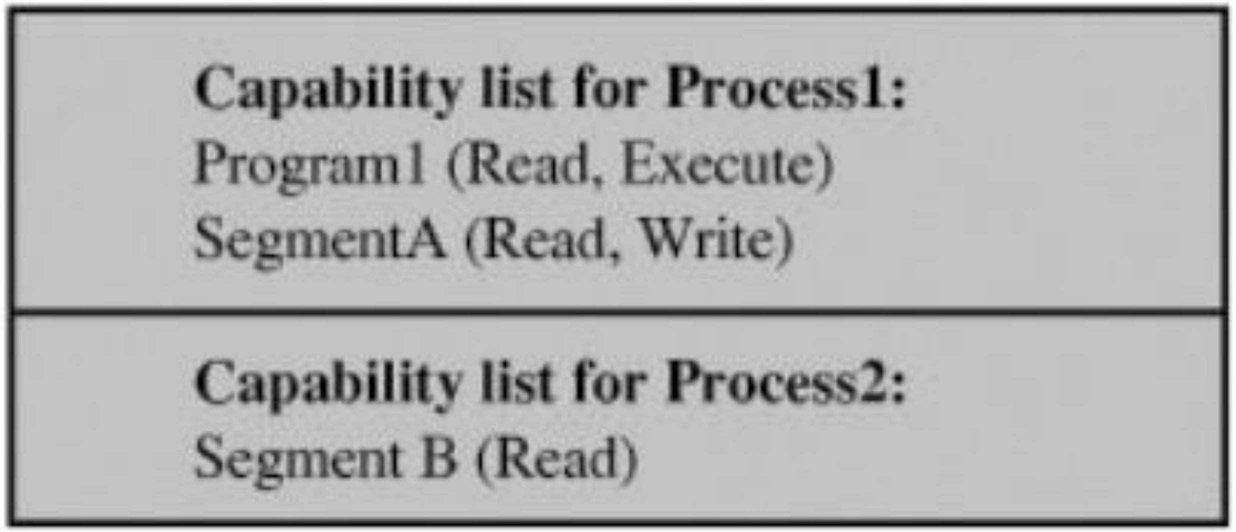


Figure 2: Capability (C-List)

From figure 2, there are three subjects (Alice, Bob and Fred) and three objects (file 1, file 2, and file 3). Each subject is assigned permission for operate on each object. For example, Alice has a permission to write on file 2, read and write on file 3.

Capability List



Capability list for Process1:
Program1 (Read, Execute)
SegmentA (Read, Write)

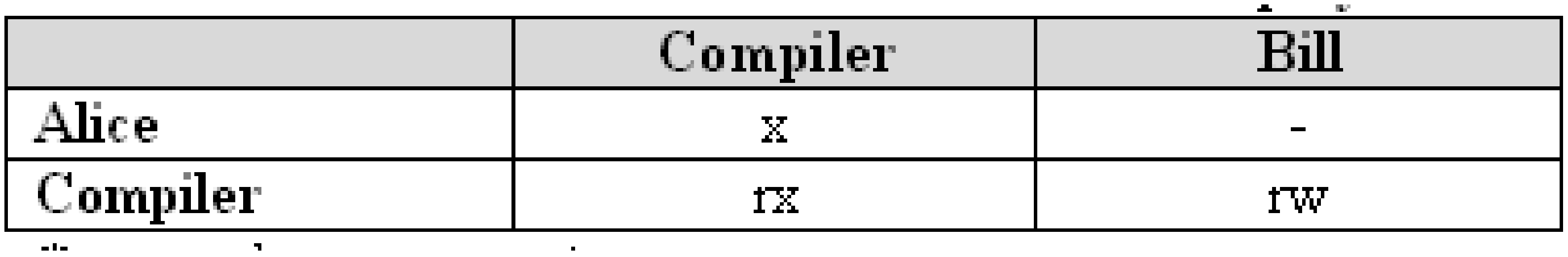
Capability list for Process2:
Segment B (Read)

(c) Capability list

Confused Deputy

A deputy is a program that acts on behalf of users or subjects. One of the known deputies is “compiler”. Compiler, a program that transforms source code into a binary form, must act as a deputy for many users. This act causes a classical security problem which is called “confused deputy”. A confused deputy is a deputy that is inappropriately manipulated. This “confused deputy” problem is commonly found in computer systems.

Confused Deputy cont.



	Compiler	Bill
Alice	x	-
Compiler	rx	rw

Table 3: Access Control Matrix for confused Deputy [3]

**r = read w = write x = execute - = not allowed*

From table 3, the compiler is granted a permission to write anything into a file named “Bill”. The file “Bill” contains critical information for resources. There is a user named “Alice”. Alice can invoke the compiler and give it a file name to get debugging output. If Alice invokes to compiler and provide “Bill” as the name of debugging file, the compiler will get confused. Although, Alice does not have a permission to write anything on file “Bill”, the compiler which is the deputy of Alice will overwrite file “Bill” with debugging information.

The confused deputy cont.

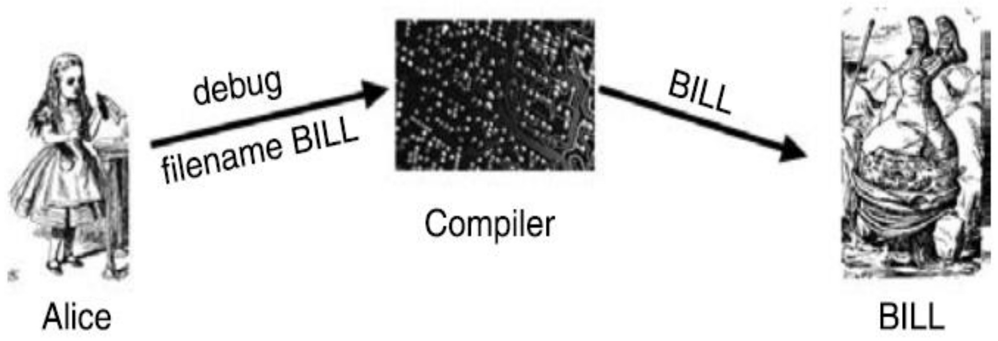


Figure 3: The confused deputy [3]

When the "confused deputy" problem occurs, C-list can prevent it but ACLs have difficulties to avoid this problem. The confusion prevention of C-list is providing C-list to the compiler shortly before starting debugging process. Alice must give her C-List to compiler if she wants to invoke the compiler. Once receiving C-list, the compiler checks all permissions related to the target file. The complier will know that Alice does not have the permission to overwrite file "Bill". On the other hand, ACLs do not have similar protection mechanism to avoid the confusion.

Goals of Confidentiality Policies

- Confidentiality Policies emphasize the protection of confidentiality.
- Confidentiality policy also called information flow policy, prevents unauthorized disclosure of information.
- Example: Privacy Act requires that certain personal data be kept confidential. E.g., income tax return info only available to IRS and legal authority with court order. It limits the distribution of documents/info.

Discretionary Access Control (DAC)

- Definition 4-13: Mechanism where a user can set access control to allow or deny access to an object
- Also called Identity-based access control (IBAC) Section 4.4.
- It is a traditional access control techniques implemented by traditional operating system such as Unix.
  - Based on user identity and ownership
  - Programs run by a user inherits all privileges granted to the user.
  - Programs is free to change access to the user's objects
  - Support only two major categories of users:
    - Completely trusted admins
    - Completely untrusted ordinary users

Problems with DAC

- Each users has complete discretion over his objects.
  - What is wrong with that?
  - Difficult to enforce a system-wide security policy, e.g.
    - A user can leak classified documents to a unclassified users.
    - Other examples?
- Only based user's identity and ownership, Ignoring security relevant info such as
  - User's role
  - Function of the program
  - Trustworthiness of the program
    - Compromised program can change access to the user's objects
    - Compromised program inherit all the permissions granted to the users (especially the root user)
  - Sensitivity of the data
  - Integrity of the data
- Only support coarse-grained privileges
- Unbounded privilege escalation
- Too simple classification of users (How about more than two categories of users?)

Mandatory Access Control (MAC)

- Definition 4-14: Mechanism where system control access to an object and a user cannot alter that access.
- Occasionally called rule-based access control?
- Defined by three major properties:
  - Administratively-defined security policy
  - Control over all subjects (process) and objects (files, sockets, network interfaces)
  - Decisions based on all security-relevant info
- MAC access decisions are based on labels that contains security-relevant info.

What Can MAC Offer?

- Supports a wide variety of categories of users in system.
  - For example, Users with labels: (secret, {EUR, US}) (top secret, {NUC, US}).
  - Here security level is specified by the two-tuple: (clearance, category)
- Strong separation of security domains
- System, application, and data integrity
- Ability to limit program privileges
  - Confine the damage caused by flowed or malicious software
- Processing pipeline guarantees
- Authorization limits for legitimate users

Mandatory and Discretionary Access Control

- Bell-LaPadula model combines Mandatory and Discretionary Access Controls.
- "S has discretionary read (write) access to O"

means that the access control matrix entry for S and O corresponding to the discretionary access control component contains a read (write) right.

	A	B	C	D	O
Q					
S					read(D)
T					

- If the mandatory controls not present, S would be able to read (write) O.

Bell-LaPadula Model

- Also called the multi-level model,
- Was proposed by Bell and LaPadula of MITRE for enforcing access control in government and military applications.
- It corresponds to military-style classifications.
- In such applications, subjects and objects are often partitioned into different security levels.
- A subject can only access objects at certain levels determined by his security level.
- For instance, the following are two typical access specifications: ``Unclassified personnel cannot read data at confidential levels'' and ``Top-Secret data cannot be written into the files at unclassified levels''

Informal Description

- Simplest type of confidentiality classification is a set of security clearances arranged in a linear (total) ordering.
- Clearances represent the security levels.
- The higher the clearance, the more sensitive the info.
- Basic confidential classification system:

	individuals	*documents*
Top Secret (TS)	Tamara, Thomas	Personnel Files
Secret (S)	Sally, Samuel	Electronic Mails
Confidential (C)	Claire, Clarence	Activity Log Files
Unclassified (UC)	Ulaley, Ursula	Telephone Lists

Star Property (Preliminary Version)

- Let L(S)=ls be the security clearance of subject S.
- Let L(O)=lo be the security classification of object).
- For all security classification li, i=0,…, k-1, li<li+1
- Simple Security Condition:
  S can read O if and only if lo<=ls and
  S has discretionary read access to O.
- *-Property (Star property):
  S can write O if and only if ls<=lo and
  S has discretionary write access to O.
- TS guy can not write documents lower than TS. →
  Prevent classified information leak.
- But how can different groups communicate?

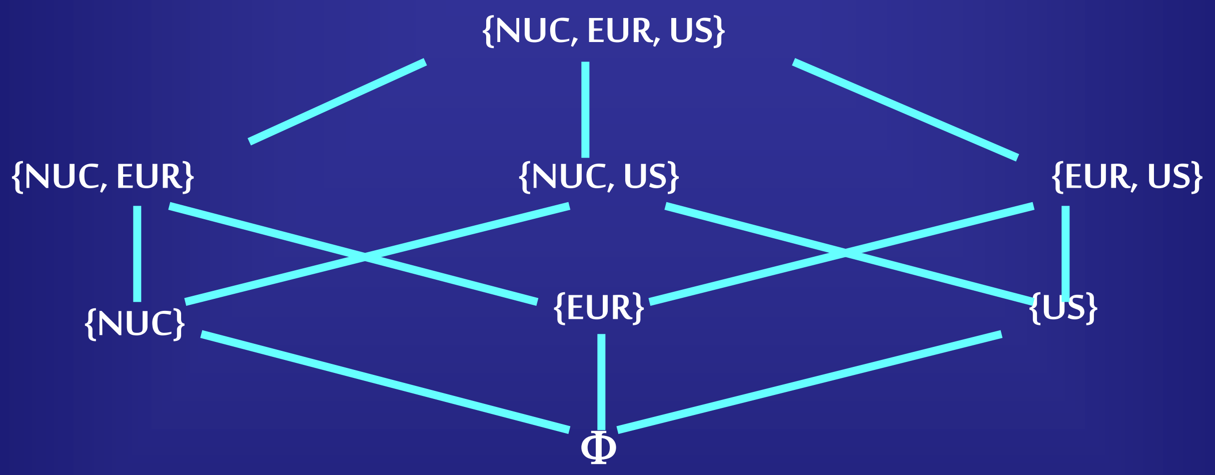
Basic Security Theorem

- Let Σ be a system with secure initial state σ_0
- Let T be the set of state transformations.
- If every element of T preserves the simple security condition, preliminary version, and the *-property, preliminary version,
  Then every state σ_i, $i \geq 0$, is secure.

Categories and Need to Know Principle

- Expand the model by adding a set of categories.
- Each category describe a kind of information.
- These category arise from the "need to know" principle → no subject should be able to read objects unless reading them is necessary for that subject to perform its function.
- Example: three categories: NUC, EUR, US.
- Each security level and category form a security level or compartment.
- Subjects *have clearance at* (are cleared into, or are in) a security level.
- Objects are *at the level of* (or are in) a security level.

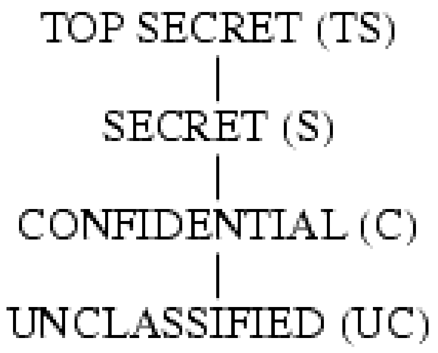
Security Lattice

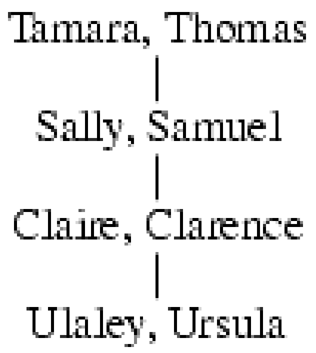


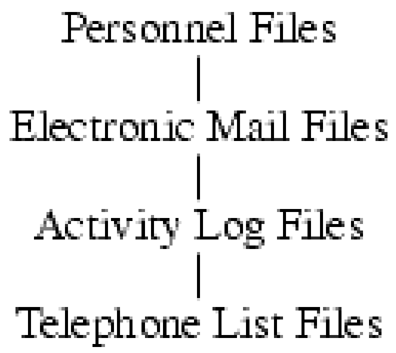
- William may be cleared into level (SECRET, {EUR})
- George into level (TS, {NUC, US}).
- A document may be classified as (C, {EUR})
- Someone with clearance at (TS, {NUC, US}) will be denied access to document with category EUR.

- **No read up: A subject can only read an object of less or equal security level. This is referred to** in the literature as the **Simple Security Property.**
- **No write down: A subject can only write into an object of greater or equal security level. This is** ***-Property**

Example


TOP SECRET (TS)
SECRET (S)
CONFIDENTIAL (C)
UNCLASSIFIED (UC)


Tamara, Thomas
Sally, Samuel
Claire, Clarence
Ulaley, Ursula


Personnel Files
Electronic Mail Files
Activity Log Files
Telephone List Files

- S can read O if and only if lo <= ls and S has discretionary read access to O.
- Claire and Clarence cannot read personnel files, but Tamara and Sally can read the activity log files.

- S can write O if and only if ls <= lo and S has discretionary write access to O.
- activity log files are classified C and Tamara has a clearance of TS, she cannot write to the activity log files.

Trusted System

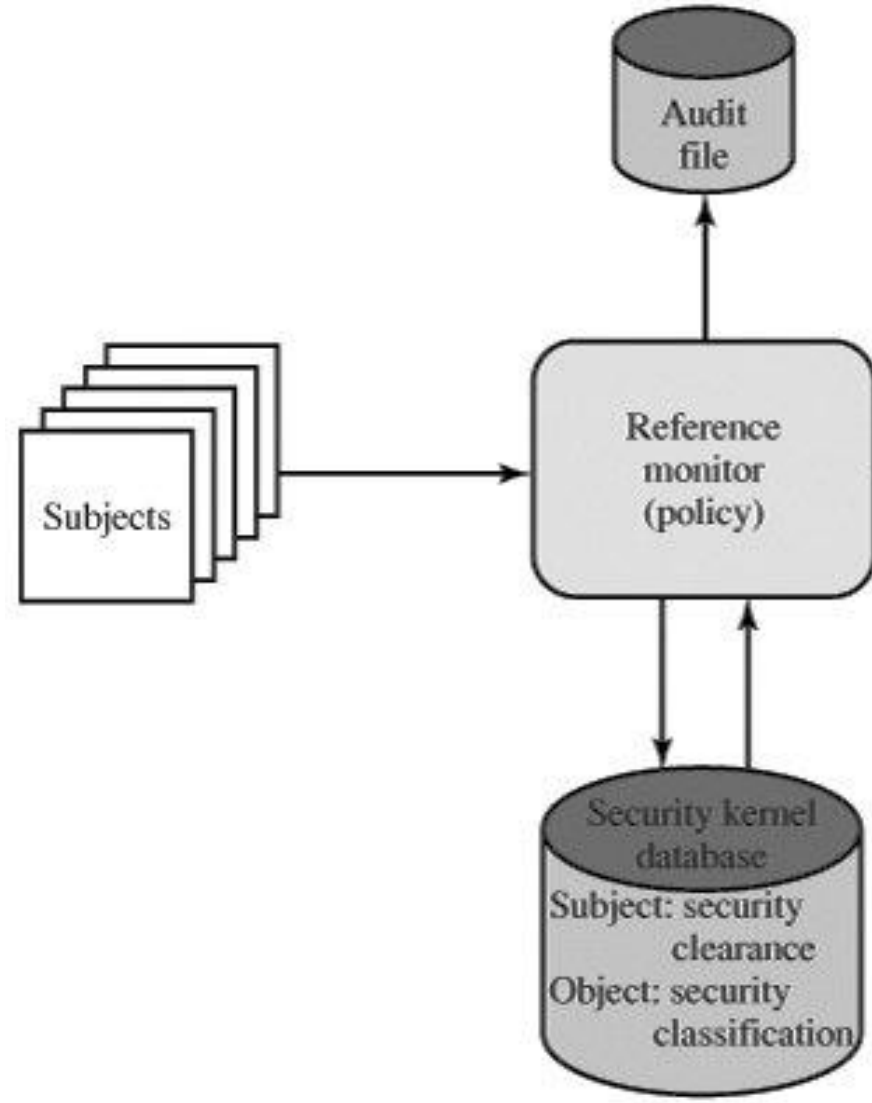
- A computer and operating system that can be verified to implement a given security policy.
- The approach is depicted below using reference monitor.

Reference Monitor

- The reference monitor has access to a file, known as the *security kernel database, that lists the access privileges (security* clearance) of each subject and the protection attributes (classification level) of each object.
- The reference monitor enforces the security rules (no read up, no write down)

Properties

- Complete mediation: The security rules are enforced on every access, not just, for example, when a file is opened.
- Isolation: The reference monitor and database are protected from unauthorized modification.
- Verifiability: The reference monitor's correctness must be provable.


Audit file
Subjects
Reference monitor (policy)
Objects
Security kernel database
Subject: security clearance
Object: security classification

- every access to data within main memory and on disk and tape must be mediated.
- The requirement for isolation means that it must not be possible for an attacker, no matter how clever, to change the logic of the reference monitor or the contents of the security kernel database

- **rusted Computer System Evaluation Criteria** (**TCSEC**) – orange book
- D — Minimal protection
- C — Discretionary protection
- B — Mandatory protection
- A — Verified protection

- Windows sql server
- Windows 7, 8