

# IT 4823

## Information Security Administration

### Access Control



Notice: This session is  
being recorded.



Copyright © 2016 by Bob Brown



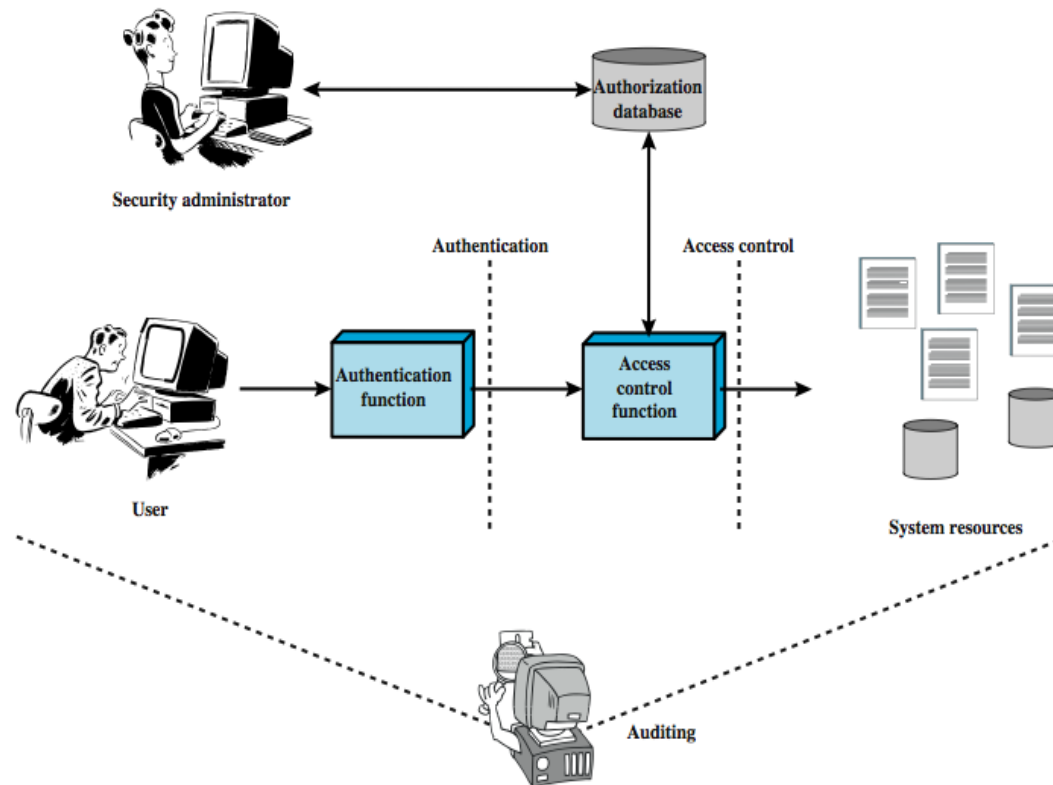
# Basics of Access Control

- *Access control* is a collection of methods and components:
  - Supports confidentiality (protects information from unauthorized disclosure)
  - Supports integrity (protects information from unauthorized modification)
- ***Authorization***: allow only authorized subjects to access objects that they are permitted to access, and only in the permitted ways. (Acceptable use)
- ***Least privilege*** philosophy:  
A subject is granted permissions needed to accomplish the required tasks and nothing more.

## Authentication is a Prerequisite

- Access control mechanisms assume the subject (person) has been authenticated.
- If another can masquerade as the subject, that person will be granted the subject's level of access.

# Authentication and Access Control



## Goals of Access Control

- **Complete mediation:** Check every access.  
(What happens if access is removed while I am using a file? What *should* happen?)
- **Least privilege:** In granting access to an object, do not also grant more rights than needed, nor rights to other objects.
- **Acceptable use:** Permitted operations depend upon the nature of the object and access granted.

## Access Control Policies

- **Discretionary access control:** Access to objects is at the discretion of the object owner.
- **Mandatory access control:** Access to objects is based on externally-enforced policies.
- **Role-based access control:** Access is based upon a role assumed by the subject.
- **Attribute-based access control:** Access is based on attributes of subject and object.
- Not mutually exclusive.

## Vimercati's List

- Reliable input
- Support for fine and coarse specifications
- Least privilege
- Separation of duties
- Dual control
- Default permit and default deny policies
- Combination of policies: conflict resolution
- Administrative mechanisms

# Definitions

- **Object** - access controlled resource
  - *e.g.* files, directories, records, programs etc.
  - number/type depend on environment
- **Subject** - entity that can access objects
  - a process representing user/application
  - often have 3 classes: owner, group, world
- **Access right** - way in which subject accesses an object
  - *e.g.* read, write, execute, delete, create, search



# Access Control Matrix

- Describes the protection state of a system, that is, the conditions under which it is secure.
- Access Control Matrix Model: simplest abstraction mechanism for representing protection state
- Protection state transitions effectively change the access control matrix.

# The Access Control Matrix

		objects (entities)					
		$o_1$	...	$o_m$	$s_1$	...	$s_n$
subjects (actors)	$s_1$						
	$s_2$						
	...						
	$s_n$						

- Subjects  $S = \{ s_1, \dots, s_n \}$
- Objects  $O = \{ o_1, \dots, o_m \}$
- Rights  $R = \{ r_1, \dots, r_k \}$
- Entries  $A[s_i, o_j] \subseteq R$
- $A[s_i, o_j] = \{ r_x, \dots, r_y \}$   
means subject  $s_i$  has  
rights  $r_x, \dots, r_y$  over  
object  $o_j$

## Example: Files

- Processes *Andy*, *Betty*
- Files *f*, *g*
- Rights *r*, *w*, *x*, *a*, *o*

	<i>f</i>	<i>g</i>	<i>Andy</i>	<i>Betty</i>
<i>Andy</i>	<i>rwo</i>	<i>r</i>	<i>rwxo</i>	<i>w</i>
<i>Betty</i>	<i>a</i>	<i>r</i>	<i>r</i>	<i>rwxo</i>

## Protection State

- $P = \{p_1, p_2, \dots p_n\}$  is the set of protection states of a system.
- $Q \subseteq P$  is the set of authorized protection states. (So,  $P - Q$  is the set of unauthorized states.)
- Security policy defines  $Q$
- Security mechanism prevents the system from entering a state in  $P - Q$ .

# State Transitions

- Operations (or commands) change the state of the system.
- Those that affect the protection state are of interest to security mechanism.
- Assume we start with an authorized state. Then, only operations on the starting state that result in a (new) authorized state must be allowed.

# Access Control Lists

Columns of access control matrix

	<i>file1</i>	<i>file2</i>	<i>file3</i>
<i>Andy</i>	rx	r	rwo
<i>Betty</i>	rwxo	r	
<i>Charlie</i>	rx	rwo	w

ACLs:

- file1: { (Andy, rx) (Betty, rwxo) (Charlie, rx) }
- file2: { (Andy, r) (Betty, r) (Charlie, rwo) }
- file3: { (Andy, rwo) (Charlie, w) }

## Default Permissions

- Normal: if subject not named in an ACL, *no* rights over a object. This is the *Principle of Fail-Safe Defaults* (“closed” or “**default deny**” policy)
- If there are many subjects, use groups or wildcards in the ACL

## Privileged Users

- Do ACLs apply to privileged users (*root*)?
- This is a philosophical question, sort-of.
- Solaris: abbreviated lists do not, but full-blown ACL entries do
- Other vendors: varies



# Capability Lists

Rows of access control matrix

	<i>file1</i>	<i>file2</i>	<i>file3</i>
<i>Andy</i>	rx	r	rwo
<i>Betty</i>	rwxo	r	
<i>Charlie</i>	rx	rwo	w

C-Lists:

- Andy: { (file1, rx) (file2, r) (file3, rwo) }
- Betty: { (file1, rwxo) (file2, r) }
- Charlie: { (file1, rx) (file2, rwo) (file3, w) }

## ACLs vs. Capabilities

- Both theoretically equivalent; consider two questions
  1. Given an object, what subjects can access it, and how?
  2. Given a subject, what objects can it access, and how?
- ACLs answer first question easily; C-Lists answer the second.
- Possibly the first question, which in the past has been of most interest, is the reason ACL-based systems more common than capability-based systems
- As second question becomes more important (in incident response, for example), this may change

# PACLs

- Propagated Access Control List
  - Implements Originator Controlled Access Control
- Creator's identity kept with PACL, copies
  - Only owner can change PACL
  - Subject reads object: object's PACL associated with subject
  - Subject writes object: subject's PACL associated with object
- Proof of concept has been shown, but either few or no actual implementations.

# Controls

- Mechanisms put into place to allow or disallow object access
  - Any potential barrier to unauthorized access
- Controls are organized into different categories
- Common categories
  - Administrative (enforce security policy through procedures, rules)
  - Logical/Technical (implement object access restrictions)
  - Physical (limit physical access to hardware)

# Access Control Techniques

- Choose techniques that fit the organization's needs (which are **defined by policy**)
- Considerations include
  - Level of security required
  - User and environmental impact of security measures
- Techniques differ in
  - The way objects and subjects are identified
  - How decisions are made to approve or deny access

# Access Control Designs

- Access control designs define rules for users accessing files or devices
- Four common access control designs
  - Mandatory access control
  - Discretionary access control
  - Non-discretionary (role-based) access control
  - Attribute based access control

# Mandatory Access Control

- Assigns a security level to each subject and object (Example: Top secret, secret, etc.)
- Matches level of subject to level of object to determine when access should be granted
- Often requires a subject to have a need to know in addition to proper security clearance
- Need to know indicates that a subject requires access to object to complete a particular task (least privilege)
- Information flows *up*, not *down*.

## Bell-La Padula Model, Preliminary

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



# Classification Levels

- Common military data classifications
  - Unclassified, Sensitive but Unclassified, Confidential, Secret, Top Secret
- Common commercial data classifications
  - Public, Sensitive, Private, Confidential
- Classification schemes can include the idea of “need to know.”

## Example

<i>security level</i>	<i>subject</i>	<i>security level</i>	<i>object</i>
Top Secret	Tamara	Top Secret	Personnel Files
Secret	Samuel	Secret	E-Mail Files
Confidential	Clarence	Confidential	Activity Logs
Unclassified	Ursula	Unclassified	Phone Lists

- Tamara can read all files
- Clarence cannot read Personnel or E-Mail Files
- Ursula can only read Phone Lists, but Tamara could copy those personnel files to the telephone lists directory. Bad.

# Reading Information

- Information flows *up*, not *down*
  - “Reads up” disallowed, “reads down” allowed
- Simple Security Condition (Preliminary)
  - Subject  $s$  can read object  $o$  iff  $L(o) \leq L(s)$
  - Sometimes called “*no read up*” rule

## Writing Information

- Information flows *up*, not *down*
  - “Writes up” allowed, “writes down” disallowed
- \*-Property (Preliminary)
  - Subject  $s$  can write object  $o$  iff  $L(s) \leq L(o)$
  - Sometimes called “*no write down*” rule; prevents leakage. (This is what keeps Tamara from putting the personnel files in the telephone list directory.)

## Discretionary Access Control

- All access to an object is defined by the object owner
- This is the most common design in commercial operating systems
  - Generally less secure than mandatory control
  - Generally easier to implement and more flexible
- Includes
  - Identity-based access control
  - Access control lists (ACLs)

## Role-Based Access Control

- Uses a subject's role or a task assigned to subject to grant or deny object access
  - Also called *non-discretionary* or task-based access control
- Works well in environments with high turnover of subjects since access is not tied directly to subject

# Attribute-Based Access Control

- A relatively new model
- Evaluates properties of
  - Subject
  - Resource
  - Environment
- Strength: Expressive power
- Concern: Performance
- Interest in application to web services, which already have significant overhead.

## Subjects and Objects

- A *subject* is an active entity, something that causes information flow. Examples: people, processes.
- An *object* is a passive entity, a resource, that can be accessed by a subject. Objects contain or receive information. Example: files.



# Subject Attributes

Attributes define the identity and characteristics of a subject:

- Identifier
- Name
- Job title
- Role

# Object Attributes

Object attributes describe the characteristics of an object.

- Name
- Creation date
- Modification date
- Ownership

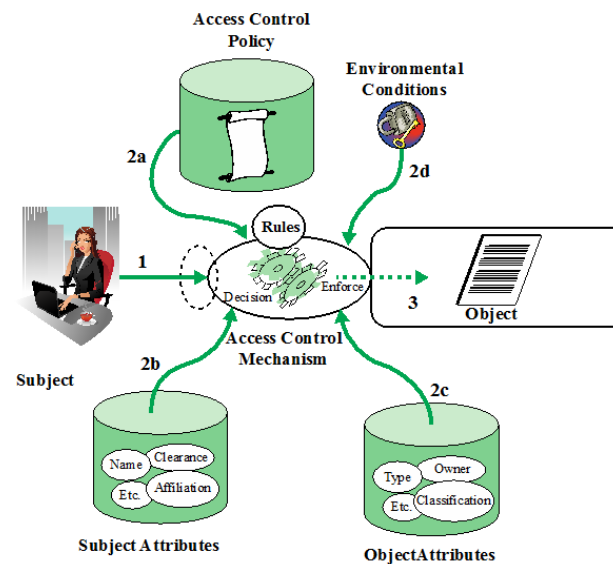
# Environment Attributes

Attributes not associated with a subject or object, but which may be used in making access decisions.

- Time and date
- Day of week
- Network status

# ABAC Access Control Policy

Access rules evaluate the **attributes** of subject, object, and environment to produce a binary decision.



# ABAC Example

## Subject

- Department    accounting
- Role            {accountant, manager}

## Environment

- Time            > 6:00 AM & < 9:00 PM

*can\_access*

## Object

- File owner     accounting
- File use        ! Archive

# Access Control Administration

- Can be implemented as centralized, decentralized, or hybrid
- Centralized access control administration
  - All requests go through a central authority
  - Administration is relatively simple
  - Single point of failure, sometimes performance bottlenecks
  - Common packages include Remote Authentication Dial-In User Service (RADIUS), Challenge Handshake Authentication Protocol (CHAP), Terminal Access Controller Access Control System (TACACS)

## Decentralized Access Control Administration

- Object access is controlled locally rather than centrally
- More difficult administration
  - Objects may need to be secured at multiple locations
- More stable
  - Not a single point of failure
- Usually implemented using security domains

# Accountability

- System auditing used by administrators to monitor
  - Who is using the system
  - What users are doing
- Logs can trace events back to originating users
- Process of auditing can have a negative effect on system performance
  - Must limit data collected in logs
  - Clipping levels set thresholds for when to start collecting data



# File and Data Ownership

- Different layers of responsibility for ensuring security of organization's information
- Data owner
  - Bears ultimate responsibility, sets classification levels (*i.e.* sets policy)
- Data custodian
  - Enforces security policies, often a member of IT department
- Data user
  - Accesses data on a day-to-day basis, responsible for following the organization's security policies

## Key Points

- Use access control to ensure that only authorized users can view/modify information
- Access control designs define rules for accessing objects
  - Mandatory, discretionary, role-based (non-discretionary), attribute-based
- Access control administration defines the mechanisms for access control implementation
  - Centralized, decentralized
- Administrators use system logs to monitor access

## Key Points (continued)

- Access control models: provide a conceptual view of security policies
- Identification and authentication methods
  - Used to identify and validate a user
  - Include passwords, smart cards, and biometrics
- Responsibility for information access is shared
  - Data owners, custodians, users
- Attack types related to access controls include
  - Brute force attacks, dictionary attacks, login spoofing

# Questions

