THE UNIVERSITY OF BRITISH COLUMBIA

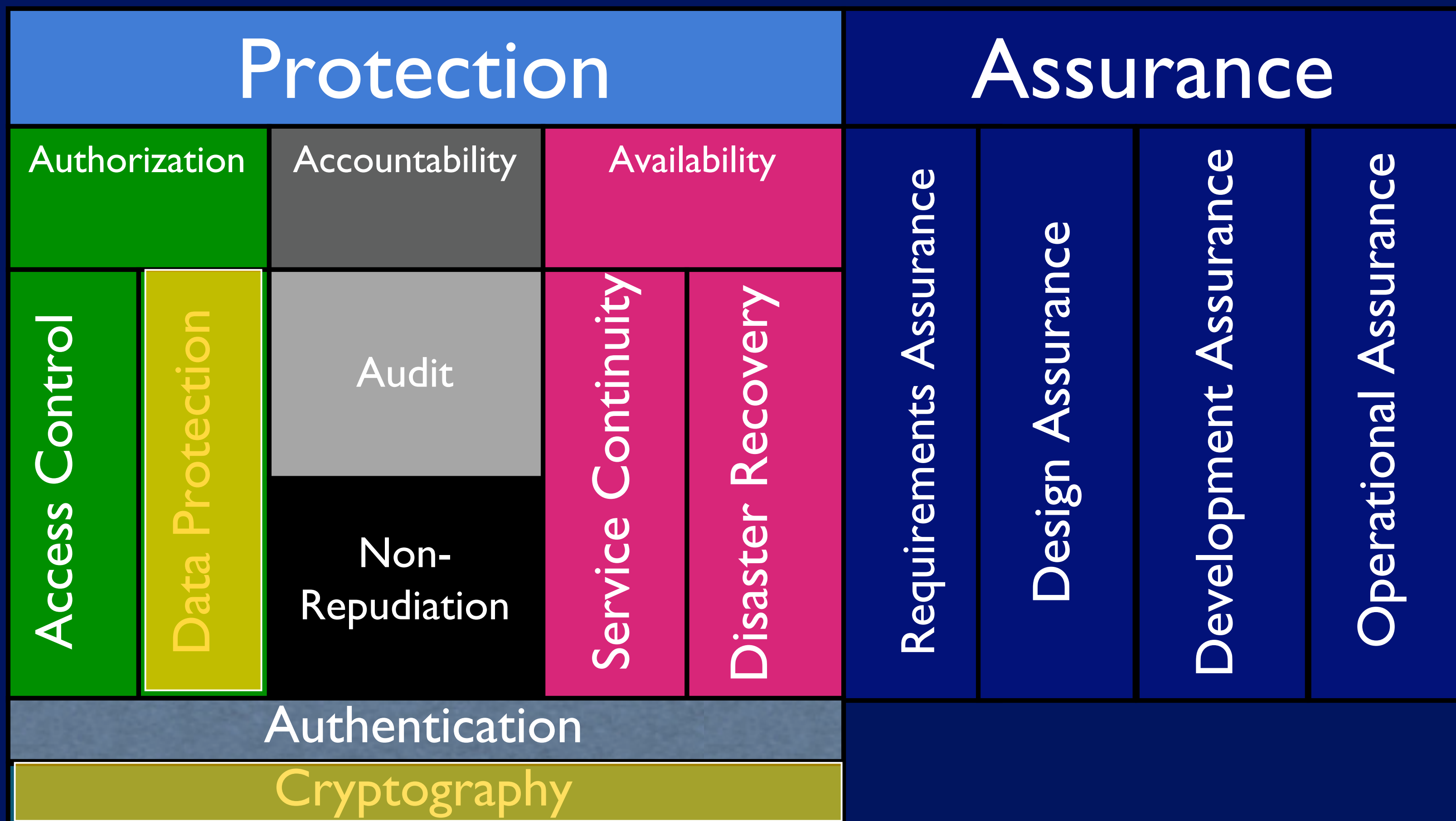
Security Policies

EECE 412

Outline

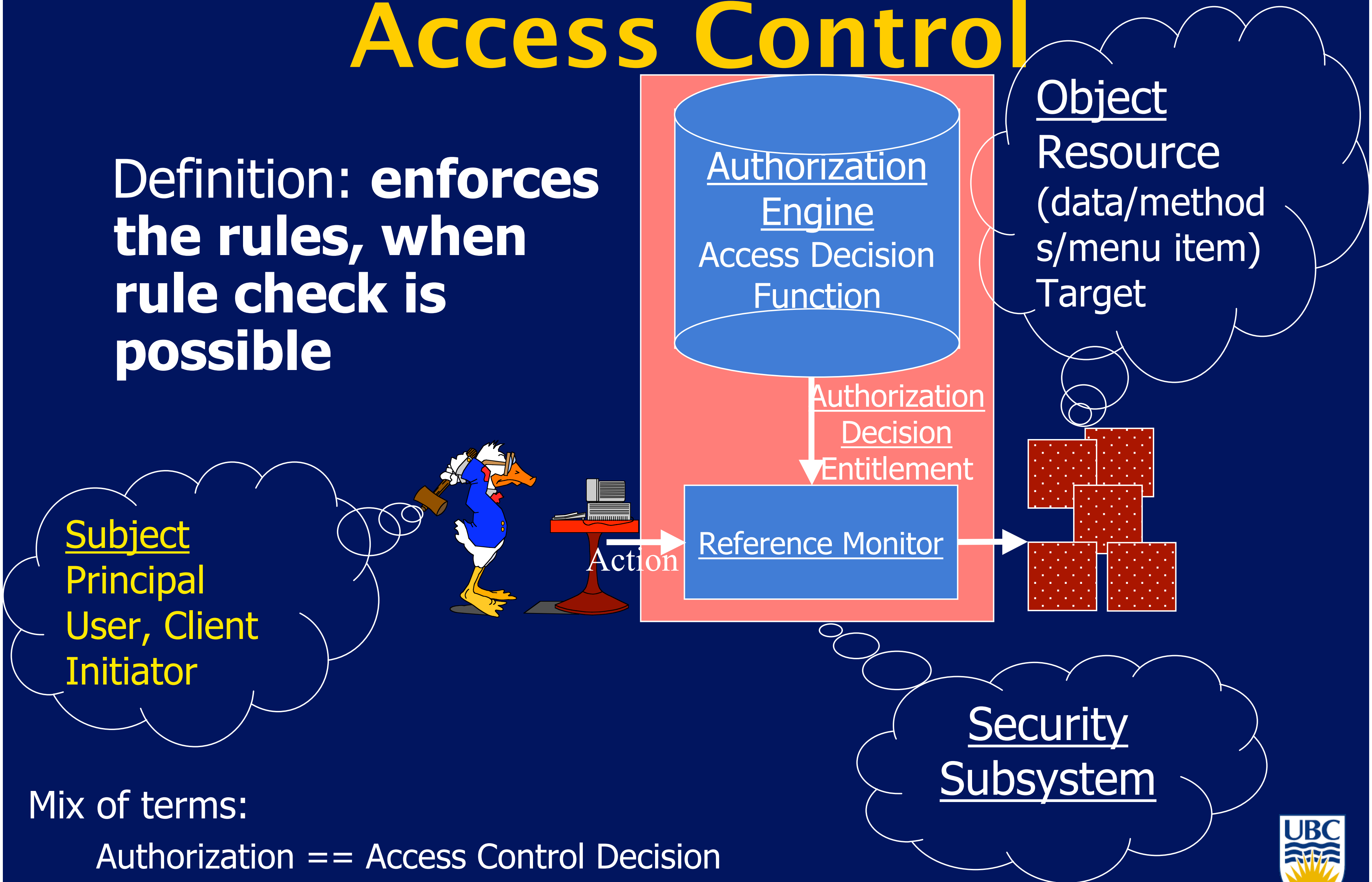
- Access control mechanisms
- Access Matrix (DAC)
- Security policies
  - Confidentiality policies
    - Bell LaPadula confidentiality model
  - Integrity policies
    - Biba integrity model
    - Clark-Wilson Integrity Model
  - Hybrid policies
    - RBAC

Where We Are


Protection
Assurance
Authorization
Accountability
Availability
Access Control
Data Protection
Audit
Non-Repudiation
Service Continuity
Disaster Recovery
Requirements Assurance
Design Assurance
Development Assurance
Operational Assurance
Authentication
Cryptography

Authorization Mechanisms: Access Control

Definition: **enforces the rules, when rule check is possible**



Mix of terms:

Authorization == Access Control Decision

Authorization Engine == Policy Engine

Policies and Mechanisms

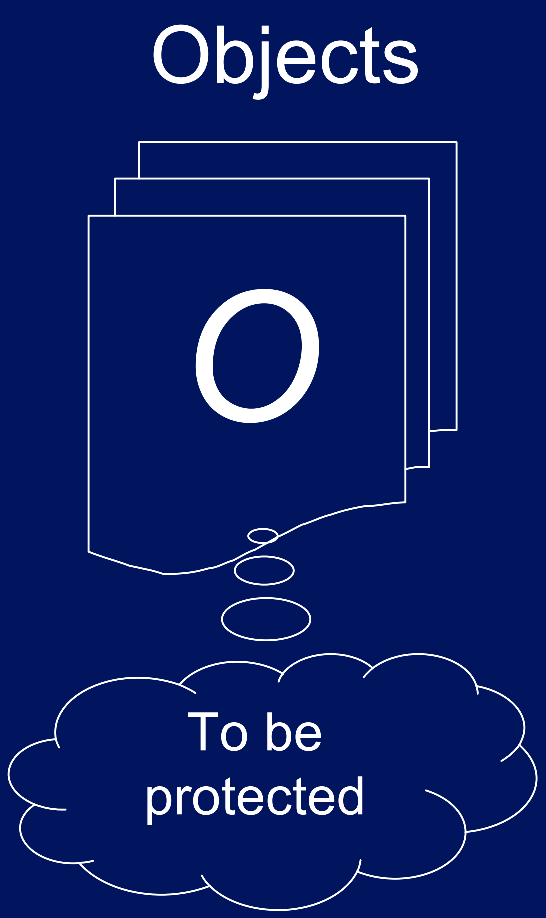
- Policies describe what is allowed
- Mechanisms control how policies are enforced

Access Matrix

Object System

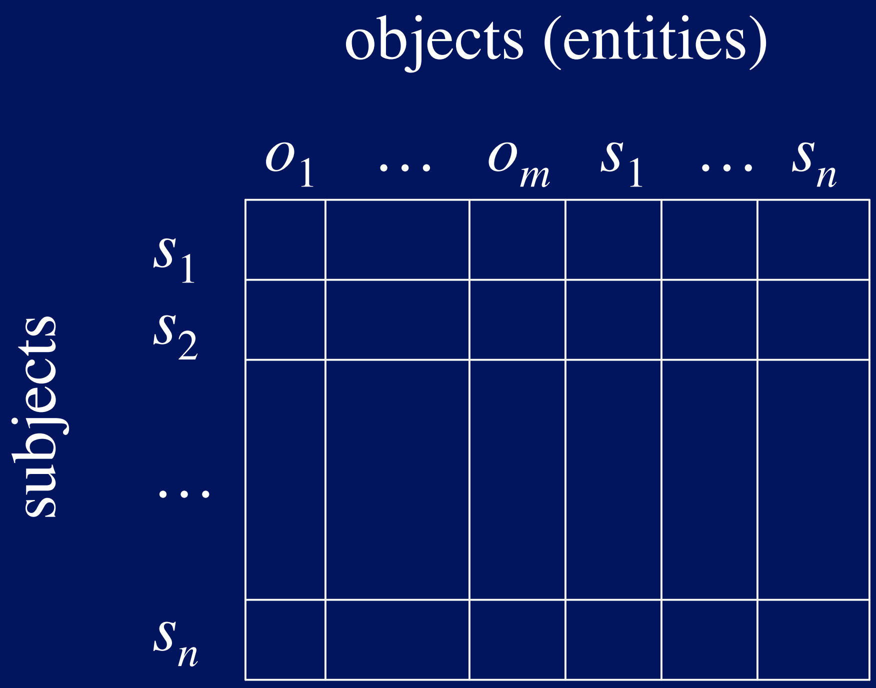
Access Matrix

	Subject 1	Subject 2	Subject 3	File 1	File 2	Process 1
Subject 1	*owner control	*owner control	*call	*owner *read *write		
Subject 2			call	*read	write	wakeup
Subject 3			owner control	read	*owner	



- Subjects are objects
- Objects are not subjects

Access Matrix Structure



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

Example

- Processes *p*, *q*
- Files *f*, *g*
- Rights *r*, *w*, *x*, *a*, *o*

	f	*g*	*p*	*q*
p	*rwo*	*r*	*rwxo*	*w*
q	*a*	*ro*	*r*	*rwxo*

Matrix Implementation Techniques

1. $T = \{<s,o,A_{d,x}>\}$ – impractical
   a) Only relevant parts of A need to be handy
   b) Could be very inefficient for some As (e.g. public files)
   c) List of objects to which d has access
2. Capability = $<o,A_{d,x}>$
   - C-lists
   - Attach C-list to subjects
   - Addresses (a), (c) and potentially (b)
3. attach the protection information to the object: $A_x(d)$
   - Access key – capability used for identification, (credential)
   - {<access key, {access attributes}>} – access control list (ACL)

Access Matrix Summary

- Object System
  - Subjects, objects, access matrix
  - Objects are shared
  - All subjects are objects
    - not all objects are subjects
- Matrix implementation
  - Capability lists
  - Access control lists

THE UNIVERSITY OF BRITISH COLUMBIA

Security Policies

What's Security Policy?

- Policy partitions system states into:
  - Authorized (secure)
    - These are states the system can enter
  - Unauthorized (nonsecure)
    - If the system enters any of these states, it's a security violation
- Secure system
  - Starts in authorized state
  - Never enters unauthorized state
- Authorized state in respect to what?

What's Confidentiality?

- *X* set of entities, *I* information
- *I* has *confidentiality* property with respect to *X* if no $x \in X$ can obtain information from *I*
- *I* can be disclosed to others

- Example:
  - *X* set of students
  - *I* final exam answer key
  - *I* is confidential with respect to *X* if students cannot obtain final exam answer key

What's Integrity?

- X set of entities, I information
- I has *integrity* property with respect to X if all $x \in X$ trust information in I
- *Examples?*

Types of Access Control

- Discretionary Access Control (DAC, IBAC)
  - individual user sets access control mechanism to allow or deny access to an object
- Mandatory Access Control (MAC)
  - system mechanism controls access to object, and individual cannot alter that access
- Originator Controlled Access Control (ORCON)
  - originator (creator) of information controls who can access information

Key Points about Policies and Mechanisms

- Policies describe what is allowed
- Mechanisms control how policies are enforced

THE UNIVERSITY OF BRITISH COLUMBIA

Confidentiality Policies

What's Confidentiality Policy

- Goal: prevent the unauthorized disclosure of information
  - Deals with information flow
  - Integrity incidental
- Multi-level security models are best-known examples
  - Bell-LaPadula Model basis for many, or most, of these

Bell-LaPadula Model, Step 1

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

Example

security level	*subject*	*object*
Top Secret	Alice	Personnel Files
Secret	Bob	E-Mail Files
Confidential	Chiang	Activity Logs
Unclassified	Fred	Telephone Lists

- Alice can read all files
- Chiang cannot read Personnel or E-Mail Files
- Fred can only read Telephone Lists

Reading Information

- Information flows *up*, not *down*
  - "Reads up" disallowed, "reads down" allowed
- Simple Security Property
  - Subject *s* can read object *o* iff, $L(o) \leq L(s)$ 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
  - Subject *s* can write object *o* iff $L(s) \le L(o)$ 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

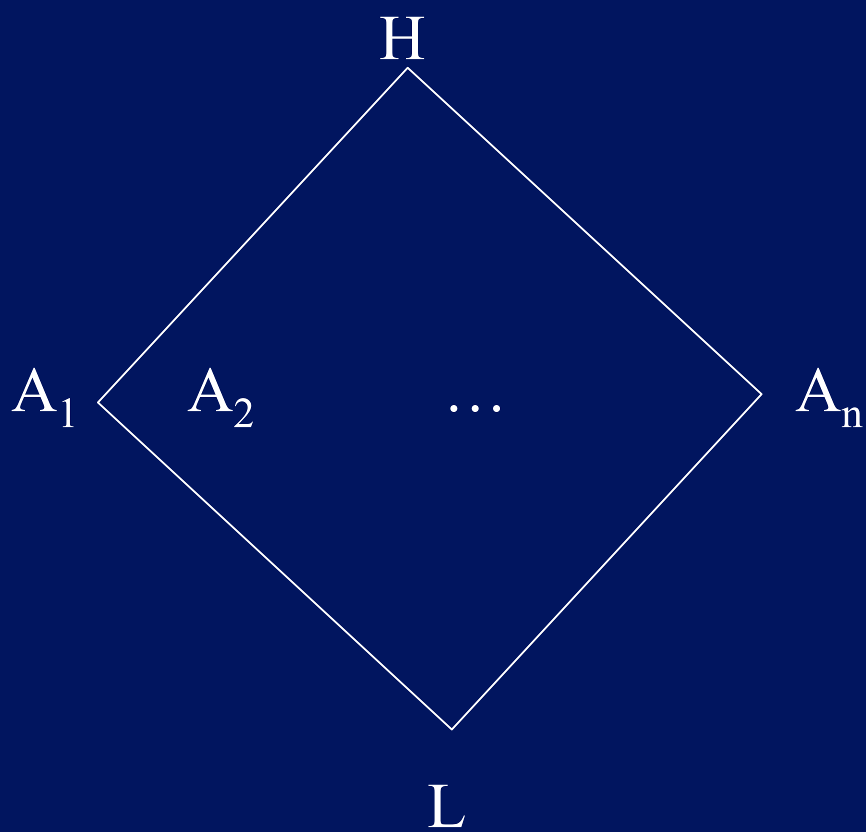
Bell-LaPadula Model, Step 2

- Expand notion of security level to include categories
- Security level is (*clearance*, *category set*)
- Examples
  - (Top Secret, { NUC, EUR, ASI })
  - (Confidential, { EUR, ASI })
  - (Secret, { NUC, ASI })

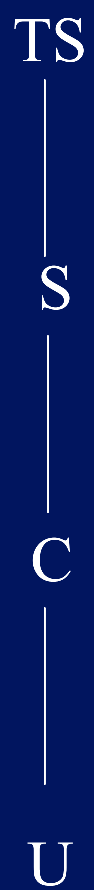
Levels and Lattices

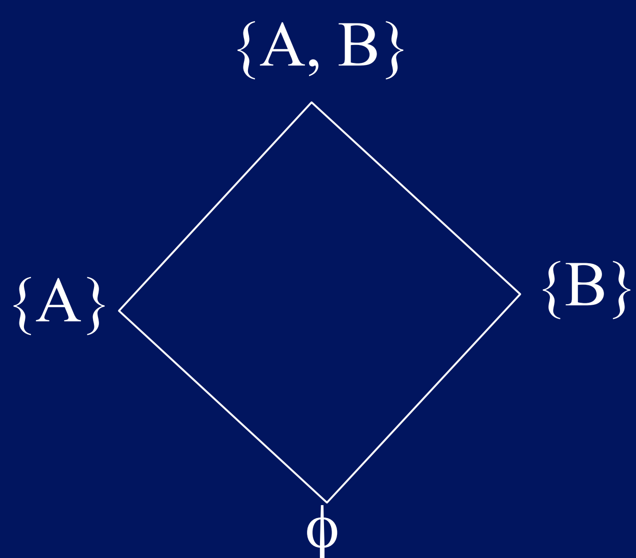
- (A, C) *dominates* (A', C') iff $A' \le A$ and $C' \subseteq C$
- Examples
  - (Top Secret, {NUC, ASI}) *dom* (Secret, {NUC})
  - (Secret, {NUC, EUR}) *dom* (Confidential,{NUC, EUR})
  - (Top Secret, {NUC}) ¬*dom* (Confidential, {EUR})
- Let C be set of classifications, K set of categories. Set of security levels $L = C \times K$, *dom* form lattice

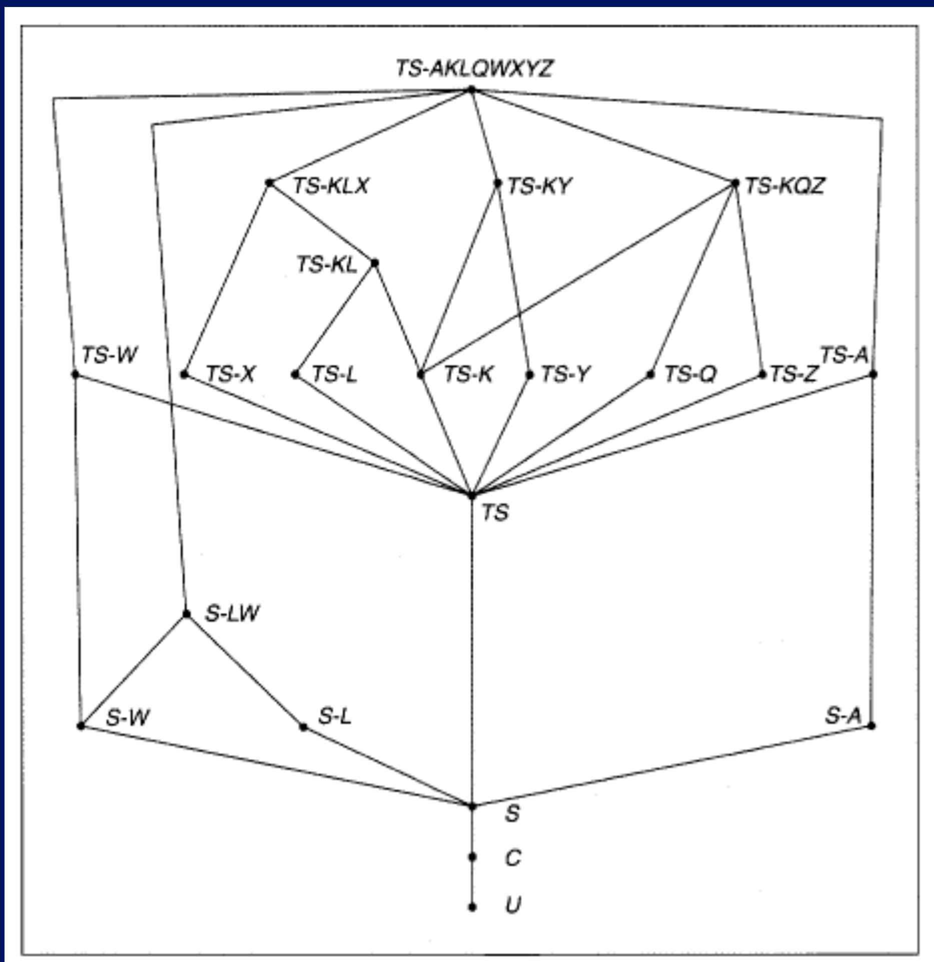
Bounded Isolated Classes



The Military Lattice







Levels and Ordering

- Security levels partially ordered
  - Any pair of security levels may (or may not) be related by *dom* relation
- Note:
  - "dominates" serves the role of "greater than"
  - "greater than" is a total ordering, though

Reading Information

- Information flows *up*, not *down*
  - "Reads up" disallowed, "reads down" allowed
- Simple Security Property (Step 2)
  - Subject *s* can read object *o* iff $L(s)$ *dom* $L(o)$ 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 2)
  - Subject *s* can write object *o* iff *L*(*o*) *dom* *L*(*s*) 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

Problem

- Colonel has (Secret, {NUC, EUR}) clearance
- Major has (Secret, {EUR}) clearance
- Major can talk to colonel (“write up” or “read down”)
- Colonel cannot talk to major (“read up” or “write down”)
- Clearly absurd!

Solution

- Define maximum, current levels for subjects
  - *maxlevel*(*s*) *dom* *curlevel*(*s*)
- Example
  - Treat Major as an object (Colonel is writing to him/her)
  - Colonel has *maxlevel* (Secret, { NUC, EUR })
  - Colonel sets *curlevel* to (Secret, { EUR })
  - Now *L*(Major) *dom* *curlevel*(Colonel)
    - Colonel can write to Major without violating "no writes down"

Key Points Regarding Confidentiality Policies

- Confidentiality policies restrict flow of information
- Bell-LaPadula model supports multilevel security
  - Cornerstone of much work in computer security

THE UNIVERSITY OF BRITISH COLUMBIA

Integrity Policies

Biba Integrity Model (1977)

- Set of subjects S, objects O, integrity levels I, relation $\leq \subseteq I \times I$ holding when second dominates first or same
- *min*: $I \times I \rightarrow I$ returns lesser of integrity levels
- *i*: $S \cup O \rightarrow I$ gives integrity level of entity
- r: $S \times O$ means $s \in S$ can read $o \in O$
- w: $S \times O$ means $s \in S$ can write $o \in O$
- x: $S \times O$ means $s \in S$ can execute $o \in O$

What does a higher integrity level of an object mean?

Intuition for Integrity Levels

- The higher the level, the more confidence
  - That a program will execute correctly
  - That data is accurate and/or reliable
- Note relationship between integrity and trustworthiness
- Important point: *integrity levels are **not** security levels*

Low-Water-Mark Policy

- Idea: when *s* reads *o*, $i'(s) = min(i(s), i(o))$; *s* can only write objects at lower levels
- Rules
  1. $s \in S$ can write to $o \in O$ if and only if (iff) $i(o) \leq i(s)$.
  2. If $s \in S$ reads $o \in O$, then $i'(s) = min(i(s), i(o))$, where $i'(s)$ is the subject's integrity level after the read.
  3. $s_1 \in S$ can execute $s_2 \in S$ if and only if $i(s_2) \leq i(s_1)$.
- When can s read o according to the Low-Water-Mark policy?

Problems

- Subjects' integrity levels decrease as system runs
  - Soon no subject will be able to access objects at high integrity levels
- What could be a solution?
- Alternative: change object levels rather than subject levels
  - Soon all objects will be at the lowest integrity level

Ring Policy

- Idea: subject integrity levels static
- Rules
  1. $s \in S$ can write to $o \in O$ if and only if $i(o) \le i(s)$.
  2. Any subject can read any object.
  3. $s_1 \in S$ can execute $s_2 \in S$ if and only if $i(s_2) \le i(s_1)$.
- Eliminates indirect modification problem

Strict Integrity Policy (a.k.a., "Biba's Model")

- Similar to Bell-LaPadula model
  1. $s \in S$ can read $o \in O$ iff $i(s) \leq i(o)$
  2. $s \in S$ can write to $o \in O$ iff $i(o) \leq i(s)$
  3. $s_1 \in S$ can execute $s_2 \in S$ iff $i(s_2) \leq i(s_1)$
- Add compartments and discretionary controls to get full dual of Bell-LaPadula model

Example: LOCUS and Biba

- Goal: prevent untrusted software from altering data or other software
- Approach: make levels of trust explicit
  - *credibility rating* based on estimate of software's trustworthiness (0 untrusted, *n* highly trusted)
  - *trusted file systems* contain software with a single credibility level
  - Process has *risk level* or highest credibility level at which process can execute
  - Must use *run-untrusted* command to run software at lower credibility level

THE UNIVERSITY OF BRITISH COLUMBIA

Clark-Wilson Integrity Model

Model

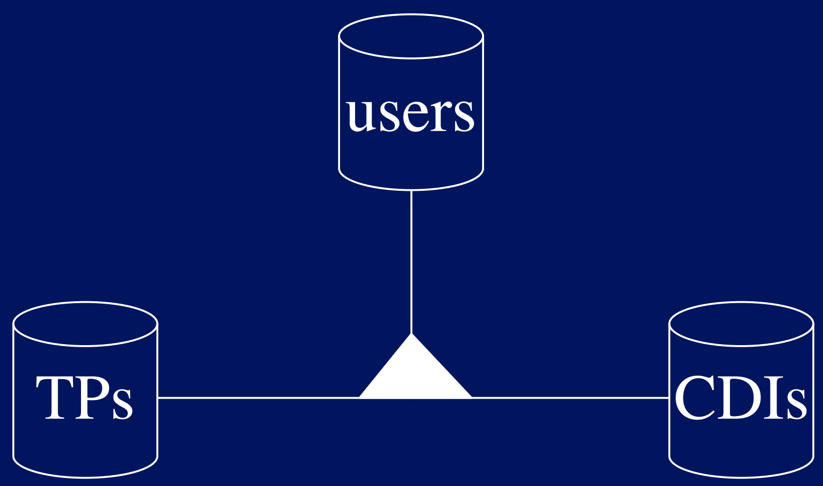
- Integrity defined by a set of constraints
  - Data in a *consistent* or valid state when it satisfies these
- Example: Bank
  - *D* today's deposits, *W* withdrawals, *YB* yesterday's balance, *TB* today's balance
  - Integrity constraint: $YB + D - W = TB$
- *Well-formed transaction* move system from one consistent state to another
- Issue: who examines, certifies transactions done correctly?
  - The principle of separation of duty

Entities in the Model

- CDIs: constrained data items
  - Data subject to integrity controls
- UDIs: unconstrained data items
  - Data not subject to integrity controls
- IVPs: integrity verification procedures
  - Procedures that test the CDIs conform to the integrity constraints
- TPs: transaction procedures
  - Procedures that take the system from one valid state to another

The Idea

Constrain who can do what by defining authorized triples: (user, TP, {CDI})



Chinese Wall Model

What's Chinese Wall Model

Problem:

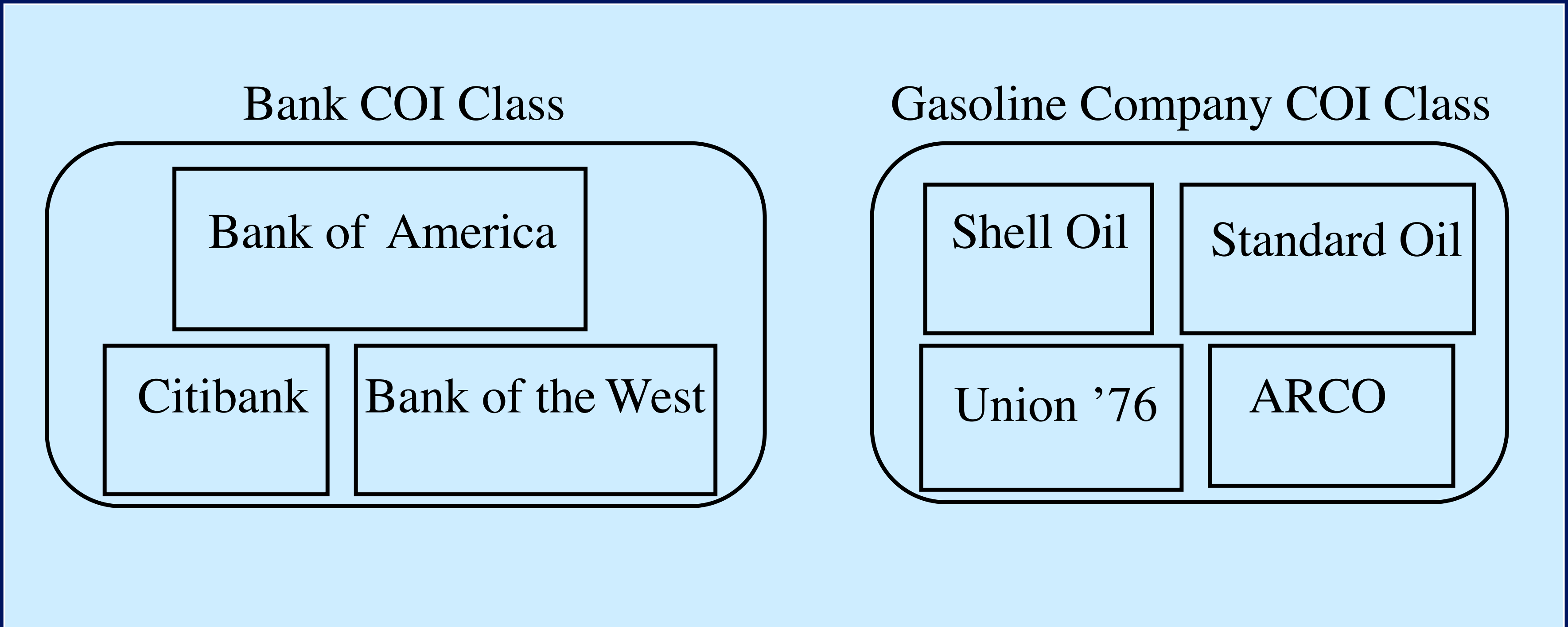
- Tony advises American Bank about investments
- He is asked to advise Toyland Bank about investments

- Conflict of interest to accept, because his advice for either bank would affect his advice to the other bank

Organization

- Organize entities into "conflict of interest" classes
- Control subject accesses to each class
- Control writing to all classes to ensure information is not passed along in violation of rules
- Allow sanitized data to be viewed by everyone

Example



- If Anthony reads any *Company dataset* (CD) in a conflict of interest (COI), he can *never* read another CD in that COI
  - Possible that information learned earlier may allow him to make decisions later

CW-Simple Security Condition

- *s* can read *o* iff either condition holds:
  1. There is an o' such that *s* has accessed o' and $CD(o') = CD(o)$
     - Meaning *s* has read something in *o*'s dataset
  2. For all $o' \in O$, $o' \in PR(s) \Rightarrow COI(o') \neq COI(o)$
     - Meaning *s* has not read any objects in *o*'s conflict of interest class
- Ignores sanitized data (see below)
- Initially, $PR(s) = \varnothing$, so initial read request granted

Writing

- Anthony, Susan work in same trading house
- Anthony can read Bank 1's CD, Gas' CD
- Susan can read Bank 2's CD, Gas' CD
- If Anthony could write to Gas' CD, Susan can read it
  - Hence, indirectly, she can read information from Bank 1's CD, a clear conflict of interest

THE UNIVERSITY OF BRITISH COLUMBIA

ORCON Model

What's the problem ORCON solves?

Problem: organization creating document wants to control its dissemination

- Example: Secretary of Agriculture writes a memo for distribution to her immediate subordinates, and she must give permission for it to be disseminated further. This is "originator controlled" (here, the "originator" is a person).

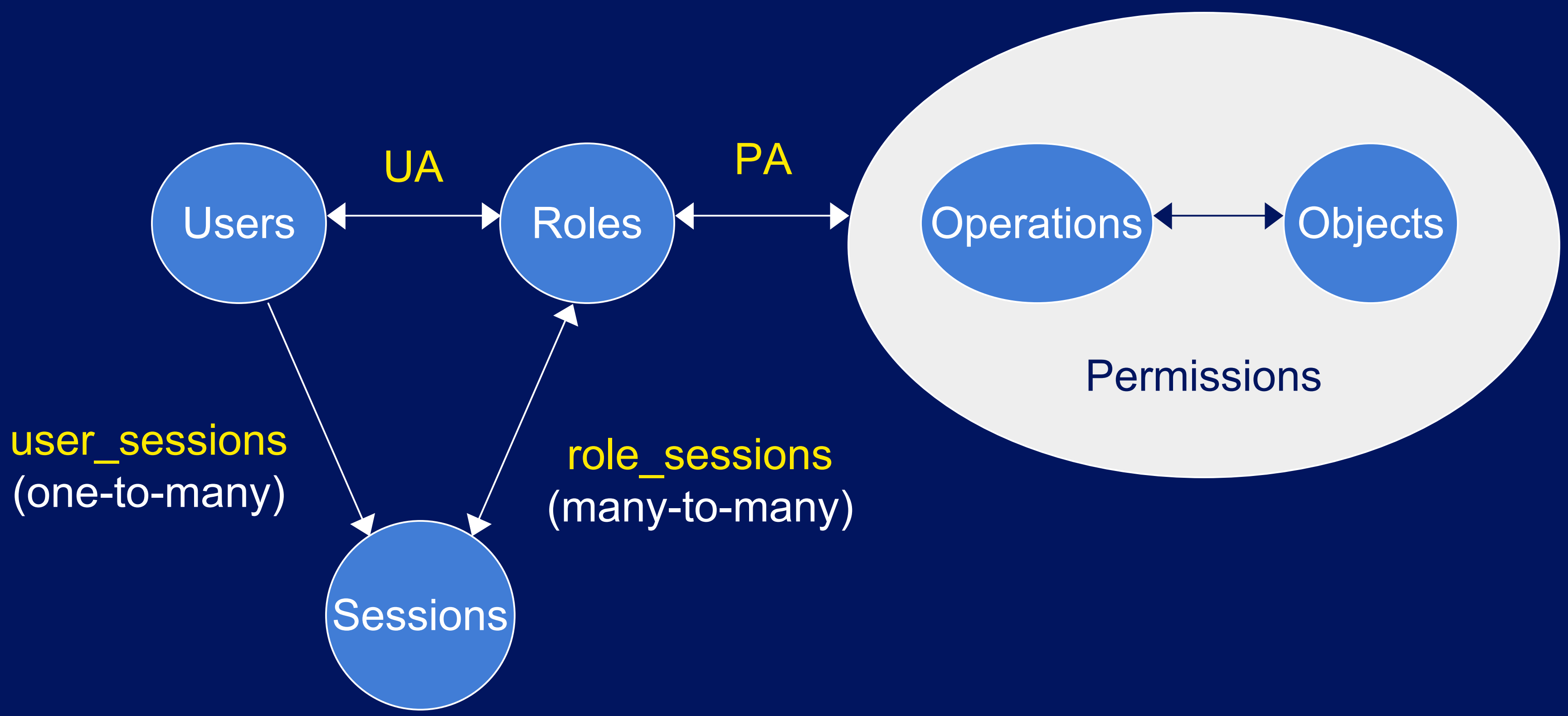
THE UNIVERSITY OF BRITISH COLUMBIA

Role-based Access Control (RBAC)

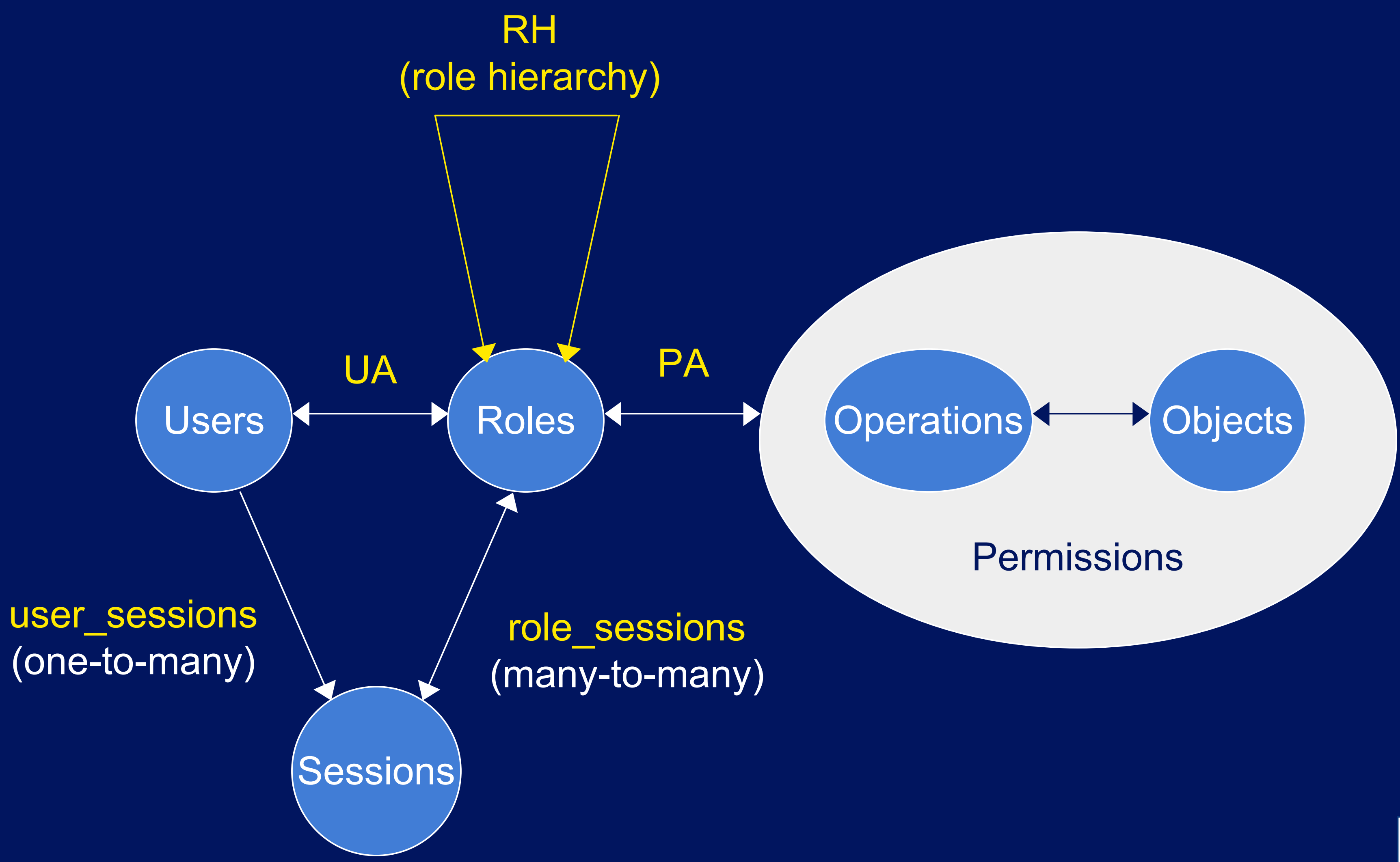
RBAC

- Access depends on role, not identity or label
  - Example:
    - Allison, administrator for a department, has access to financial records.
    - She leaves.
    - Betty hired as the new administrator, so she now has access to those records
  - The role of “administrator” dictates access, not the identity of the individual.

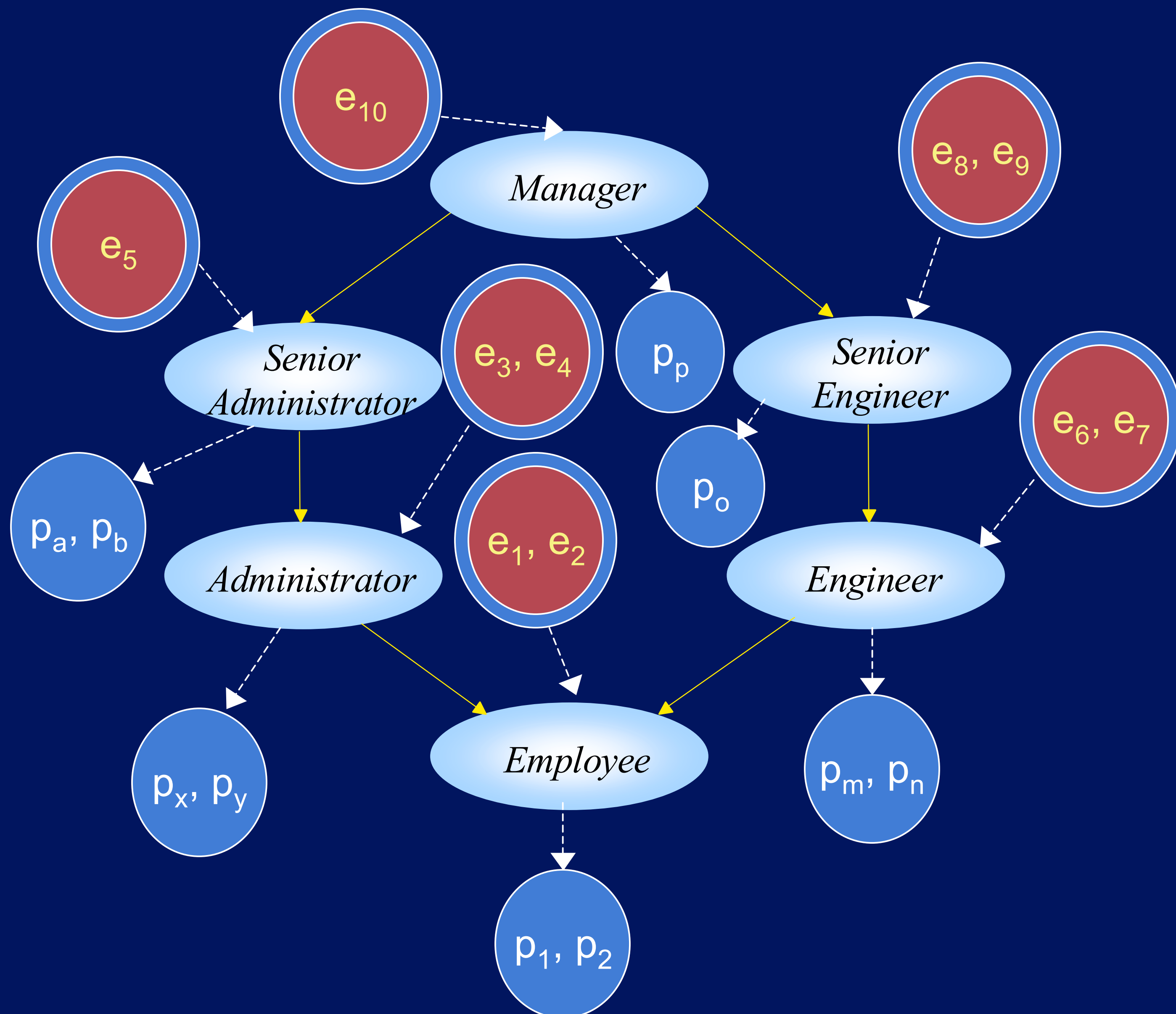
RBAC (NIST Standard)



RBAC with General Role Hierarchy

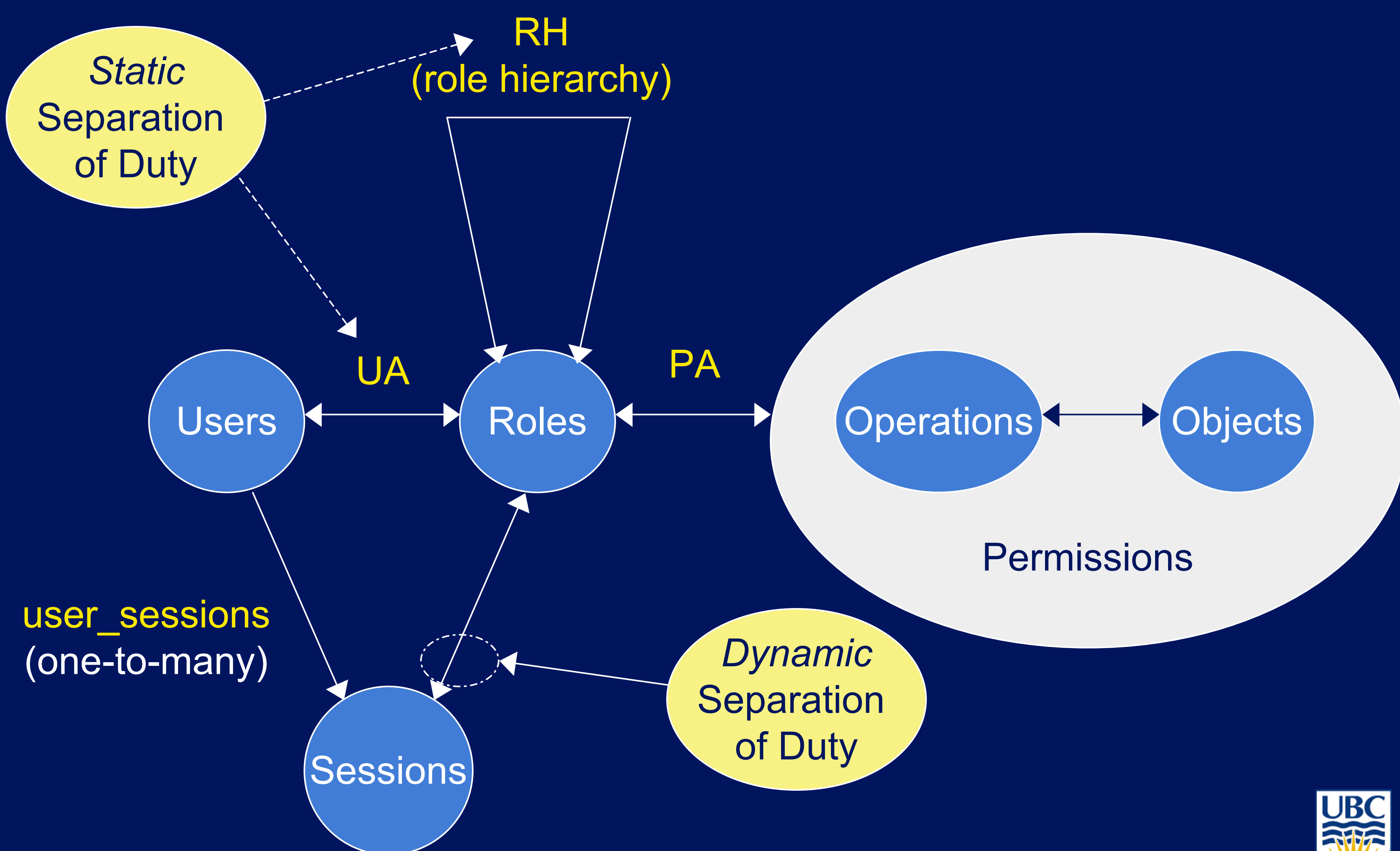


Example



e_{10}
Manager
e_8, e_9
e_5
Senior Administrator
e_3, e_4
p_p
Senior Engineer
e_6, e_7
p_a, p_b
p_o
Administrator
e_1, e_2
Engineer
p_x, p_y
Employee
p_m, p_n
p_1, p_2

Constrained RBAC



Key Points

- Integrity policies
  - deal with trust
    - As trust is hard to quantify, these policies are hard to evaluate completely
    - Look for assumptions and trusted users to find possible weak points in their implementation
  - Biba based on multilevel integrity
  - Clark-Wilson focuses on separation of duty and transactions
- Hybrid policies
  - deal with both confidentiality and integrity
  - Different combinations of these
  - ORCON model neither MAC nor DAC
    - Actually, a combination
  - RBAC model controls access based on subject's role(s)