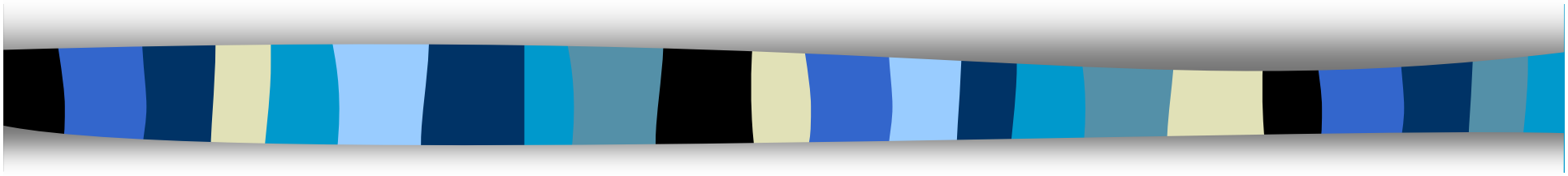
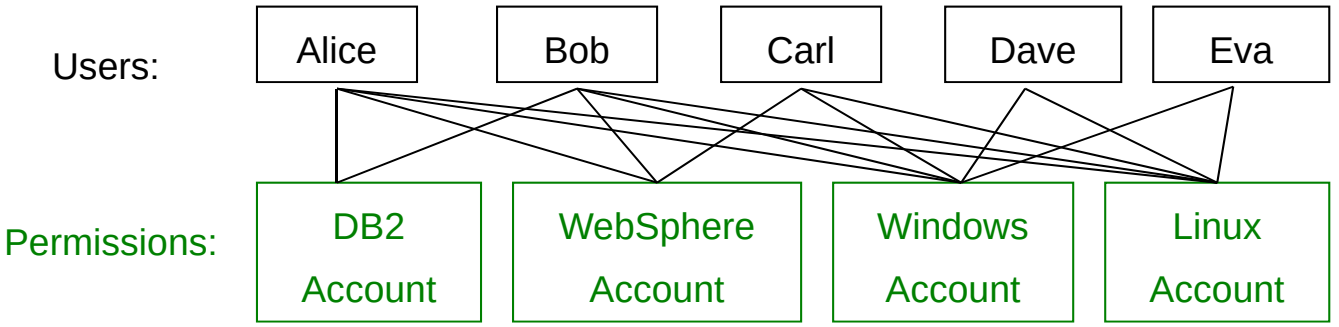
CS 426 (Fall 2010)



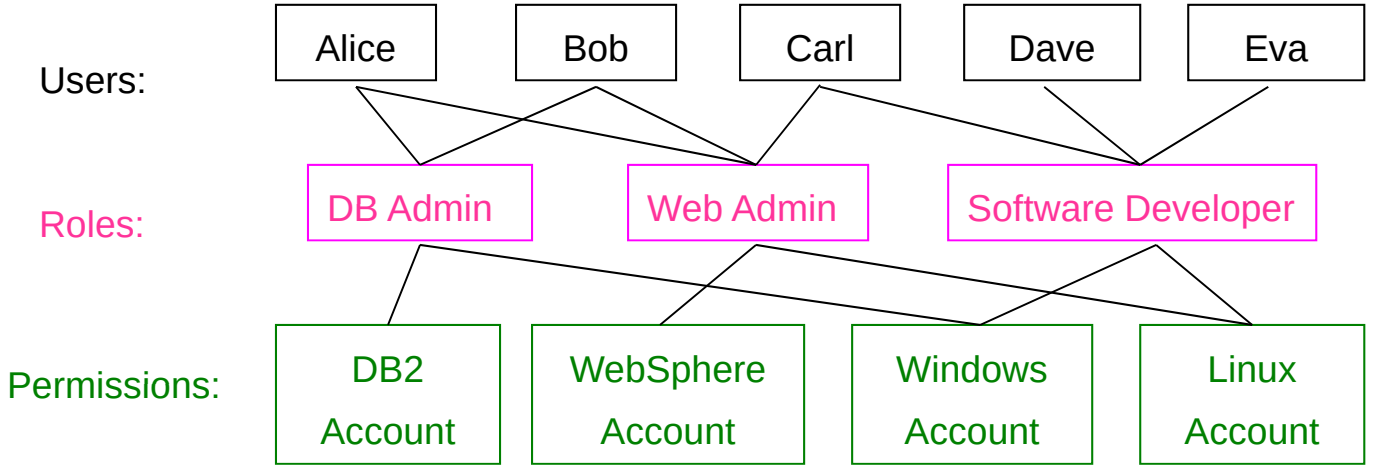
Role Based Access Control

Background: Role Based Access Control

- Non-role-based systems



- Role-Based Access Control Systems (RBAC)



ROLE-BASED ACCESS CONTROL (RBAC)

- Motivating Problem: how to administer user-permission relation
  - Different from DAC and MAC, which deal with processes in operating systems
- Roles as a level of indirection
  - Butler Lampson: "all problems in Computer Science can be solved by another level of indirection"
- RBAC is multi-faceted and open ended
  - Extensions: ARBAC (administrative), CBRAC (constraint), dRBAC (dynamic), ERBAC (enterprise), fRBAC (flexible), GRBAC (generalized), HRBAC (hierarchical), IRBAC (interoperability), JRBAC (Java), LRBAC (Location), MRBAC (Management), PRBAC (privacy), QRBAC (QoS), RRBAC(Rule), SRBAC(Spatial), TRBAC (temporal), V, W, x.
  - Non extension: OrBAC

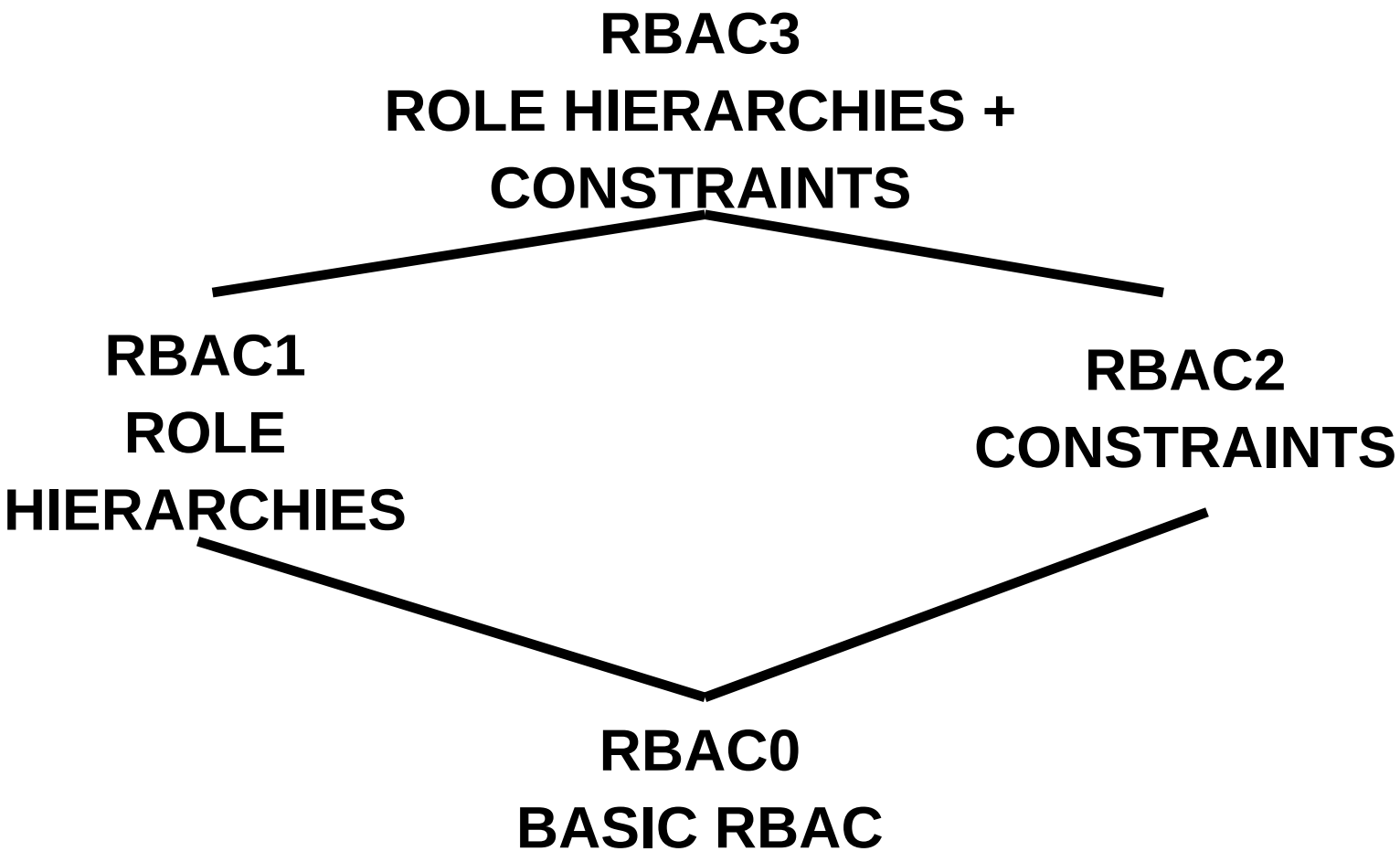
Why Roles?

- Fewer relationships to manage
  - possibly from O(mn) to O(m+n), where m is the number of users and n is the number of permissions
- Roles add a useful level of abstraction
- Organizations operate based on roles
- A role may be more stable than
  - the collection of users and the collection of permissions that are associated with it

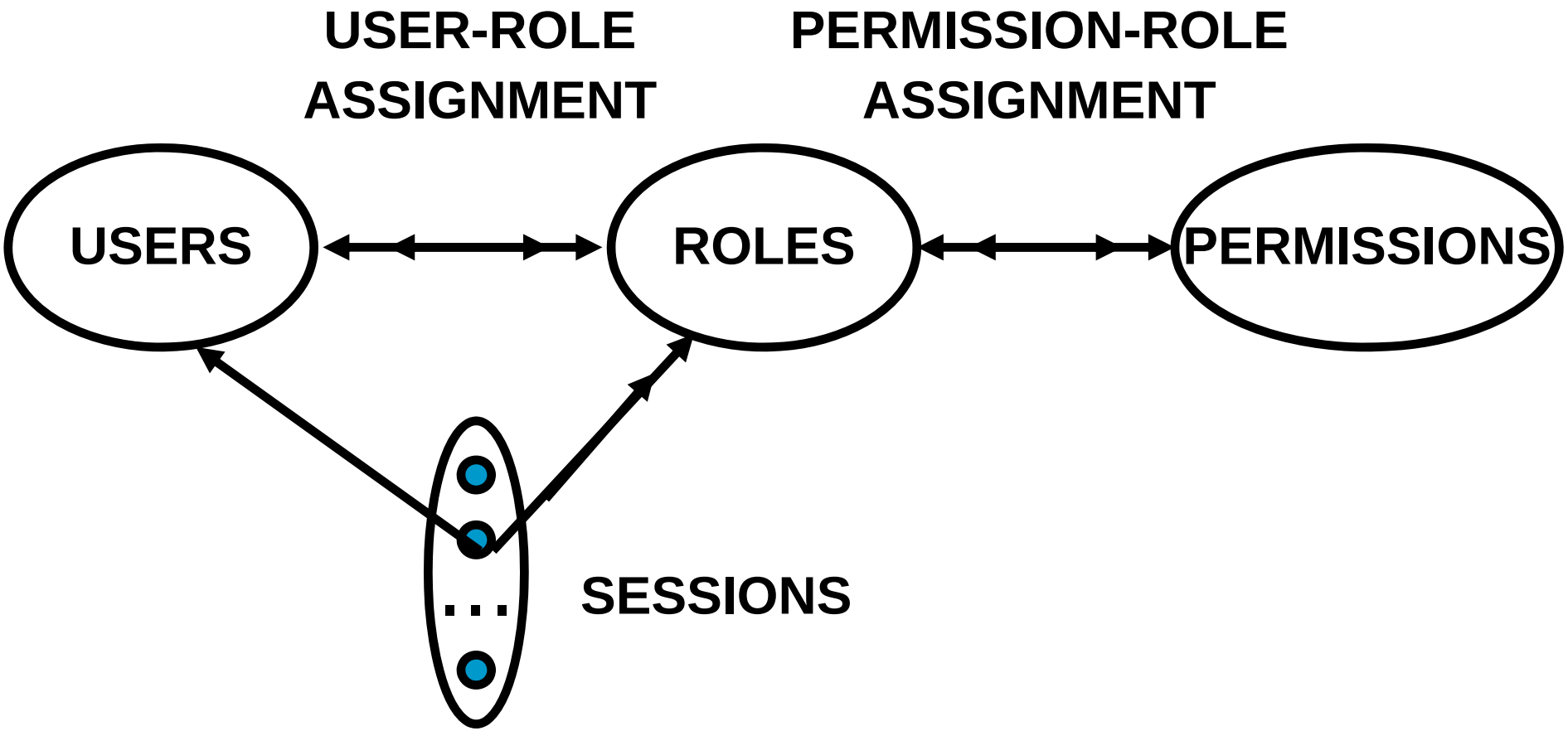
Groups vs. Roles

- Depending on the precise definition, can be the same or different.
- Some differences that may or may not be important, depending on the situation
  - Answer 1: sets of users vs. sets of users as well as permissions
  - Answer 2: roles can be activated and deactivated, groups cannot
    - Groups can be used to prevent access with negative authorization.
    - Roles can be deactivated for least privilege
  - Answer 3: can easily enumerate permissions that a role has, but not for groups

RBAC96 FAMILY OF MODELS (Sandhu et al.)



RBAC0


USER-ROLE
ASSIGNMENT
PERMISSION-ROLE
ASSIGNMENT
USERS
ROLES
PERMISSIONS
. . .
SESSIONS

PERMISSIONS

- Left abstract in the RBAC96 model

- Permissions are positive
- No negative permissions or denials
  - RBAC defines a closed policy, i.e., all accesses are denied unless they are explicitly authorized

- No duties or obligations
  - Example obligation: can access patient document, but must notify patient, or must delete after 30 days

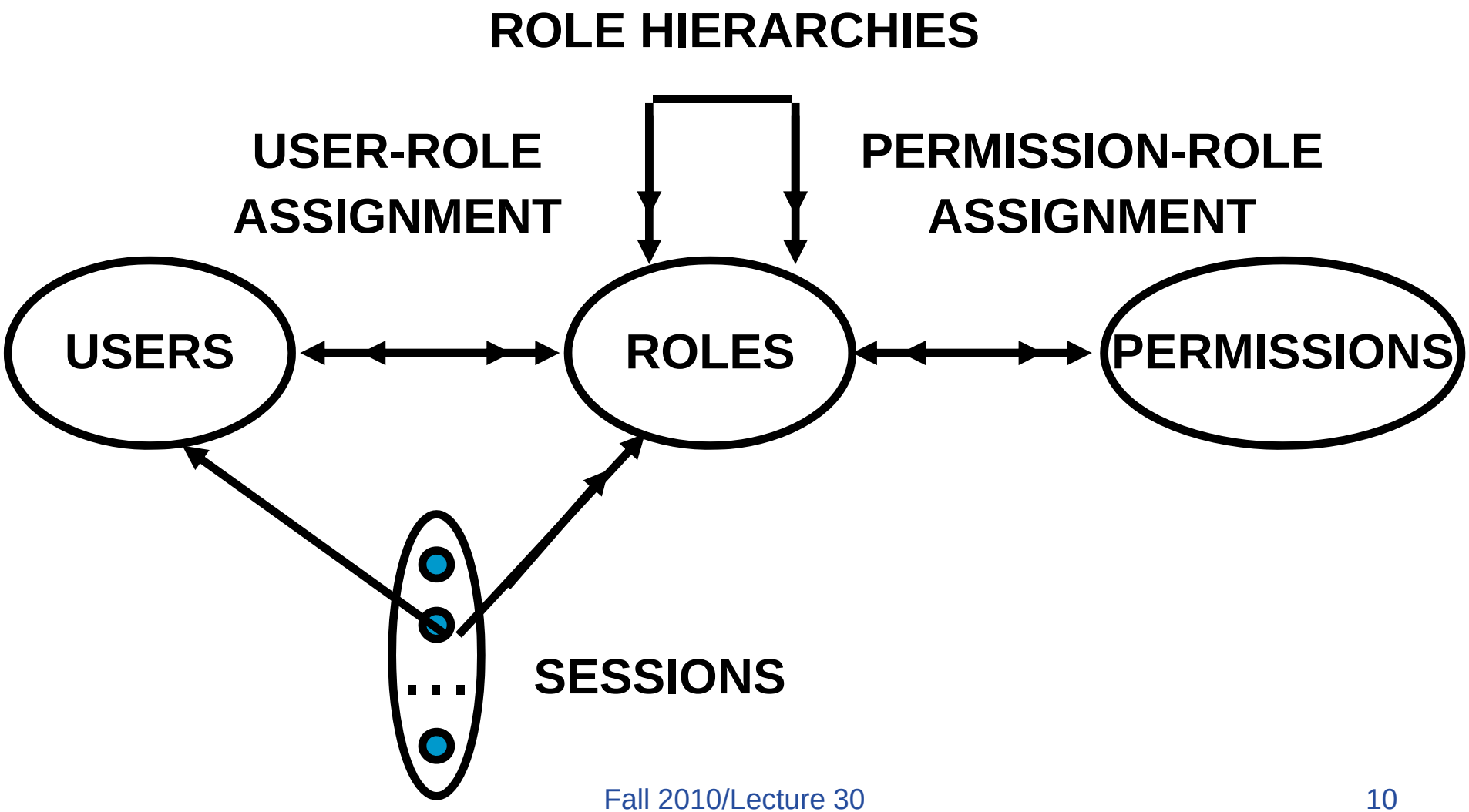
RBAC0: Formal Model

- Vocabulary: U, R, P, S (users, roles, permissions, and sessions)
- Static relations:
  - $PA \subseteq P \times R$ (permission assignment)
  - $UA \subseteq U \times R$ (user assignment)
- Dynamic relations:
  - user: $S \rightarrow U$ each session has one user
  - roles: $S \rightarrow 2^R$ and some activated roles
    - requires $roles(s) \subseteq \{ r \mid (user(s), r) \in UA \}$

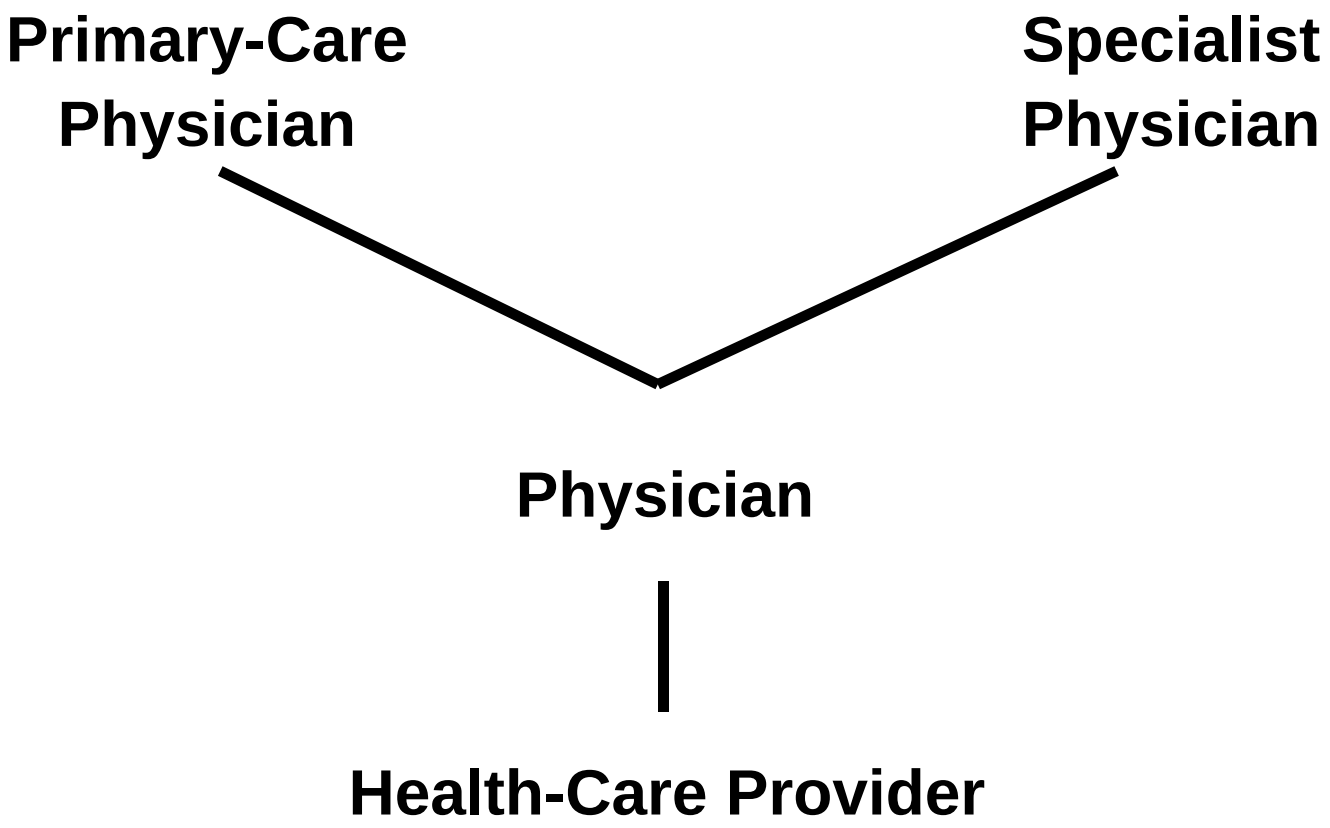
Session s has permissions

$$\bigcup_{r \in roles(s)} \{ p \mid (p, r) \in PA \}$$

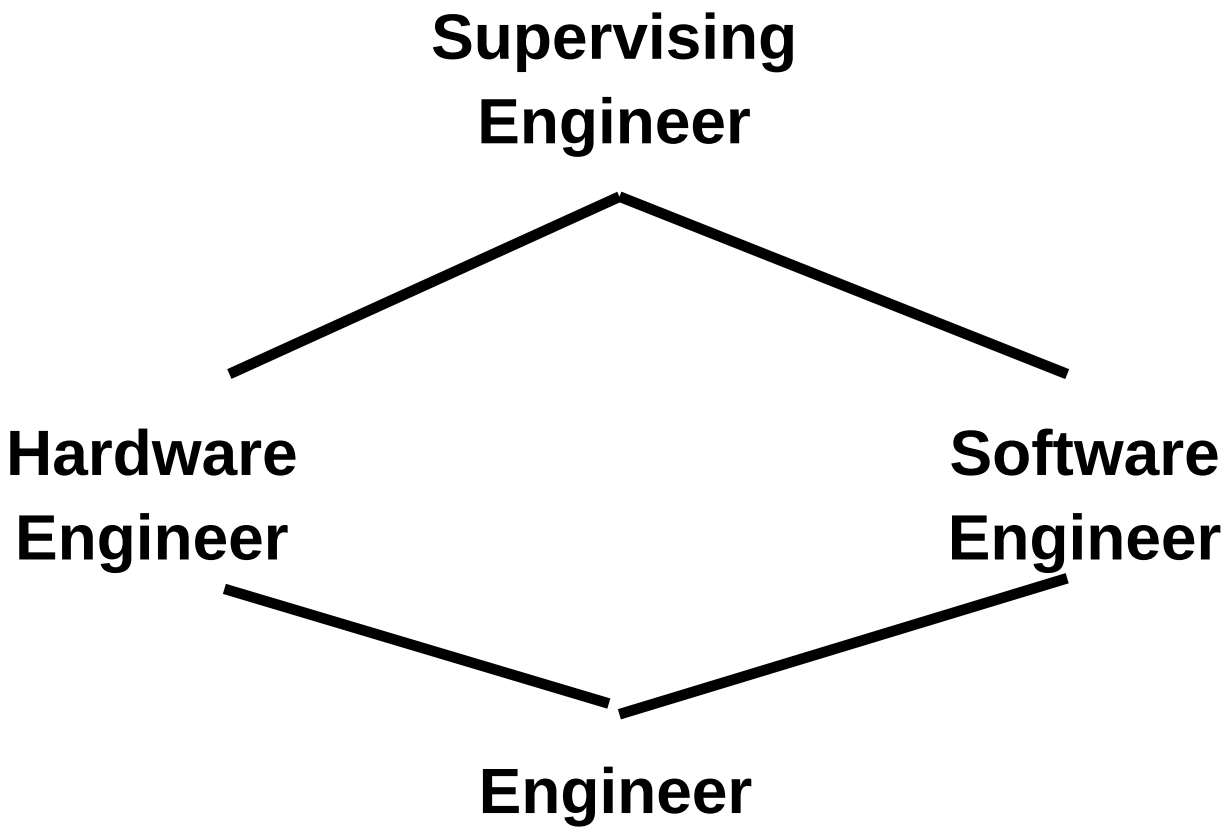
RBAC1


ROLE HIERARCHIES
USER-ROLE ASSIGNMENT
PERMISSION-ROLE ASSIGNMENT
USERS
ROLES
PERMISSIONS
SESSIONS

HIERARCHICAL ROLES (ex 1)

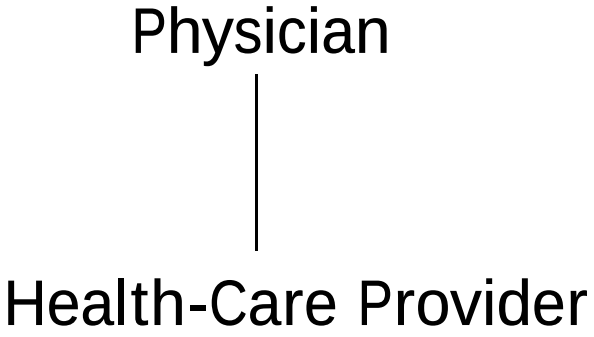


HIERARCHICAL ROLES (ex 2)



Semantics of Role Hierarchies

- User inheritance
  - r1≥r2 means every user that is a member of r1 is also a member of r2
- Permission inheritance
  - r1≥r2 means every permission that is authorized for r2 is also authorized r1
- Activation inheritance
  - r1≥r2 means that activating r1 will also activate r2



Permission and Activation inheritance have different effect when there are constraints about activation.

RBAC1: Formal Model

- U, R, R, S, PA, UA, and user unchanged from RBAC0
- RH $\subseteq$ R × R : a partial order on R, written as $\geq$
  - When r1 $\geq$ r2, we say r1 is a senior than r1, and r2 is a junior than r1
- roles: S $\rightarrow 2^R$
  - requires roles(s) $\subseteq$
    $\{ r \mid \exists r' [(r' \geq r) \& (user(s), r') \in UA] \}$

Session s includes permissions

$$\bigcup_{r \in roles(s)} \{ p \mid \exists r'' [(r \geq r'') \& (p, r'') \in PA] \}$$

RBAC2: RBAC0 + Constraints

- No formal model specified
- Example constraints
  - Mutual exclusion
  - Pre-condition: Must satisfy some condition to be member of some role
    - E.g., a user must be an undergrad student before being assigned the UTA role
  - Cardinality

Mutual Exclusion Constraints

- Mutually Exclusive Roles
  - Static Exclusion: No user can hold both roles
    - often referred to as Static Separation of Duty constraints
    - Preventing a single user from having too much permissions
  - Dynamic Exclusion: No user can activate both roles in one session
    - Often referred to as Dynamic Separation of Duty constraints
    - Interact with role hierarchy interpretation

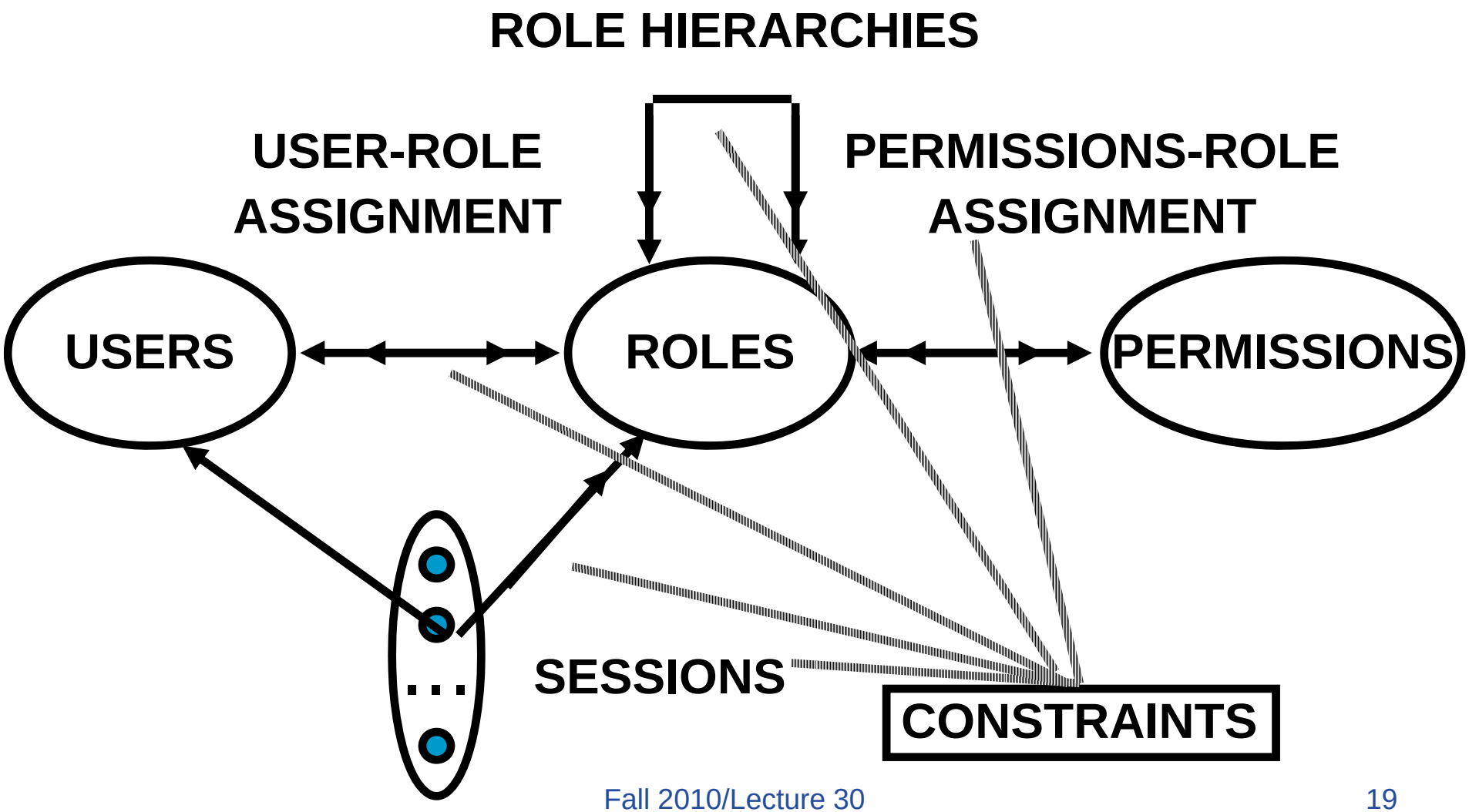
Cardinality Constraints

- On User-Role Assignment
  - at most k users can belong to the role
  - at least k users must belong to the role
  - exactly k users must belong to the role
- On activation
  - at most k users can activate a role
  - …

Why Using Constraints?

- For laying out higher level organization policy
  - Only a tool for convenience and error checking when admin is centralized
    - Not absolutely necessary if admin is always vigilant, as admin can check all organization policies are met when making any changes to RBAC policies
  - A tool to enforce high-level policies when admin is decentralized

RBAC3


ROLE HIERARCHIES
USER-ROLE ASSIGNMENT
PERMISSIONS-ROLE ASSIGNMENT
USERS
ROLES
PERMISSIONS
SESSIONS
CONSTRAINTS

Products Using RBAC

- Data Base Management Systems (DBMS)
- Enterprise Security Management
  - IBM Tivoli Identity Manager (central administration and provisioning of accounts, resources, etc)
- Many operating systems claim to use roles
  - Though only in very limited way

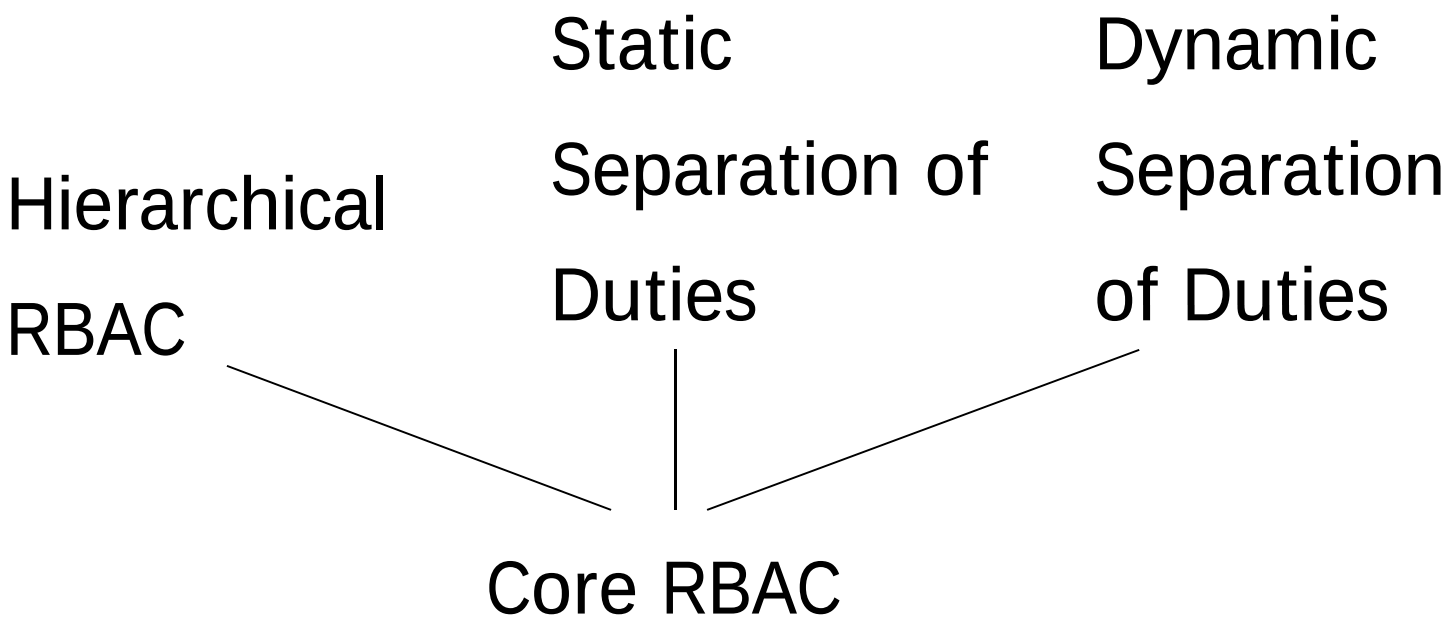
RBAC Economic Impact Study in 2002

- Based on interviews with software developers and companies that integrate RBAC products into their business operations (end users), the Research Triangle Institute (RTI) estimates that by 2006 between 30 and 50 percent of employees in the service sector and between 10 and 25 percent of employees in the non-service sectors will be managed by RBAC systems. RTI also estimates that this degree of market penetration will result in economic benefits to the U.S. economy through 2006 of approximately $671 million in net present value terms. This estimate is conservative because it reflects only the administrative and productivity benefits from RBAC.

The NIST Standard

- Proposed NIST Standard for Role-Based Access Control. David F. Ferraiolo, Ravi S. Sandhu, Serban I. Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. TISSEC, August 2001.
- American National Standards Institute Standard, 2004
  - Has a number of flaws, including with typos, errors in math definitions, and others high-level design choices

Overview of the NIST Standard for RBAC

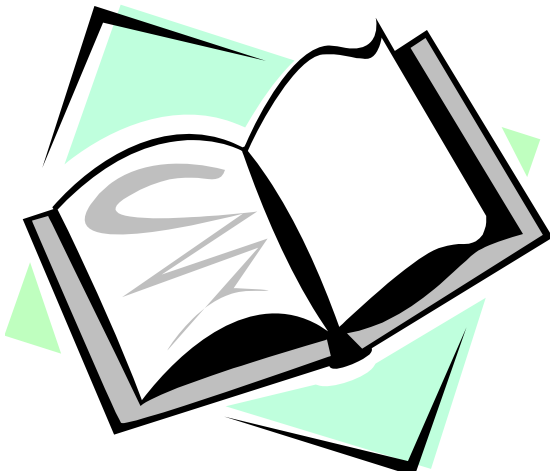


Research Challenges in RBAC

- Role engineering
  - Design roles for an access control scenario.
  - Top down approach: start from analyzing business requirement.
  - Bottom up approach: Role Mining: mine existing access control data for roles
- Effective administration of RBAC systems
  - Especially help ensure updates still lead to useful states
- Effective usage of constraints

Readings for This Lecture

- RBAC96 Family
  - R.S. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman. “Role-Based Access Control Models”. *IEEE Computer*, 29(2):38--47, February 1996.



Coming Attractions …

- Public Key Cryptography