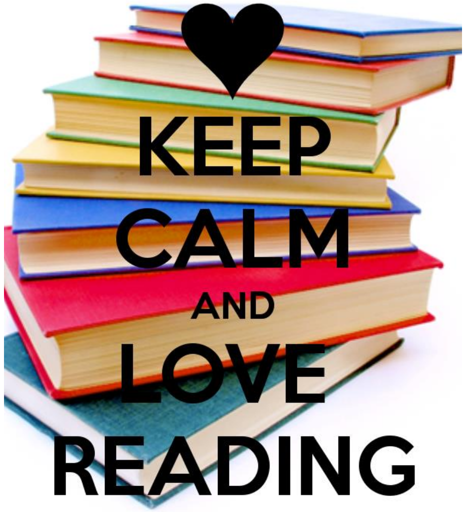
Role and Attribute Based Access Control

Information Security
CS 526

Omar Chowdhury

Reading 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, Feb 1996.



Access Control

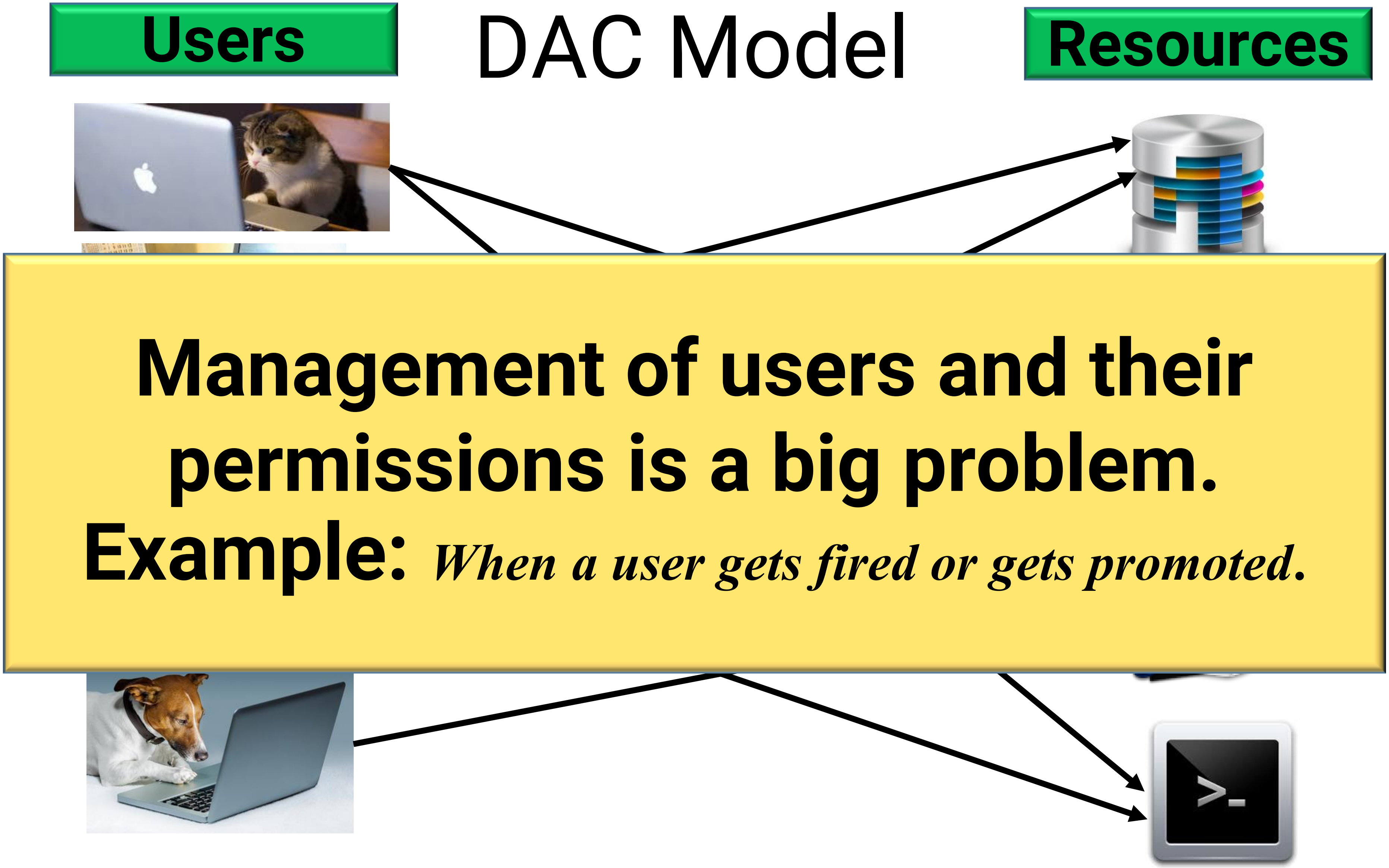
- Access control asserts **who** can access **which resource** with **what capability** under what condition

Examples

Discretionary Access Control (DAC)

Mandatory Access Control (DAC)

DAC Model



Management of users and their permissions is a big problem.

Example: ***When a user gets fired or gets promoted.***

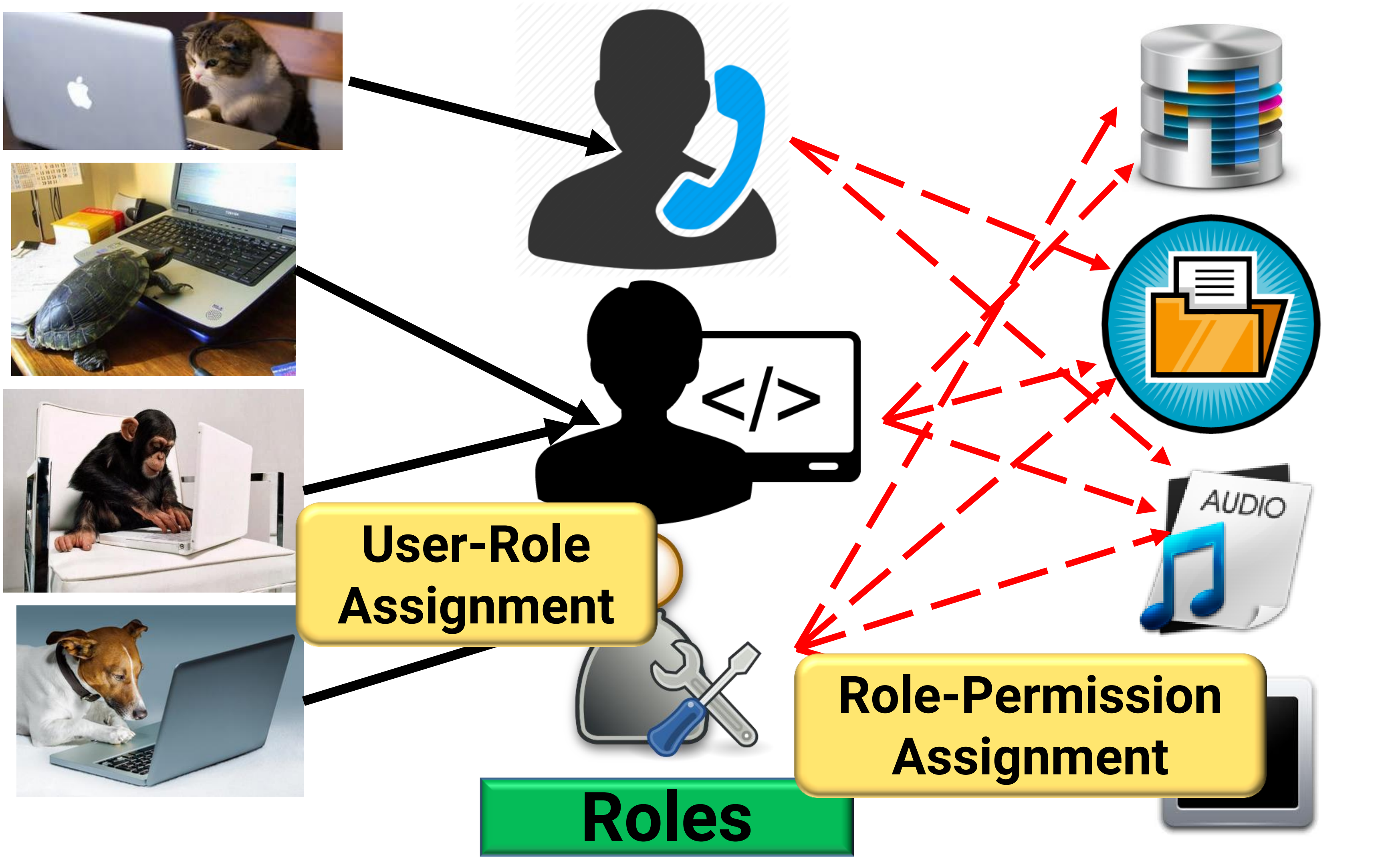
Indirection

Butler Lampson or David Whee

All problem
Science
can
nother level of
ndirection

Roles

RBAC Model



Users

Resources

User-Role Assignment

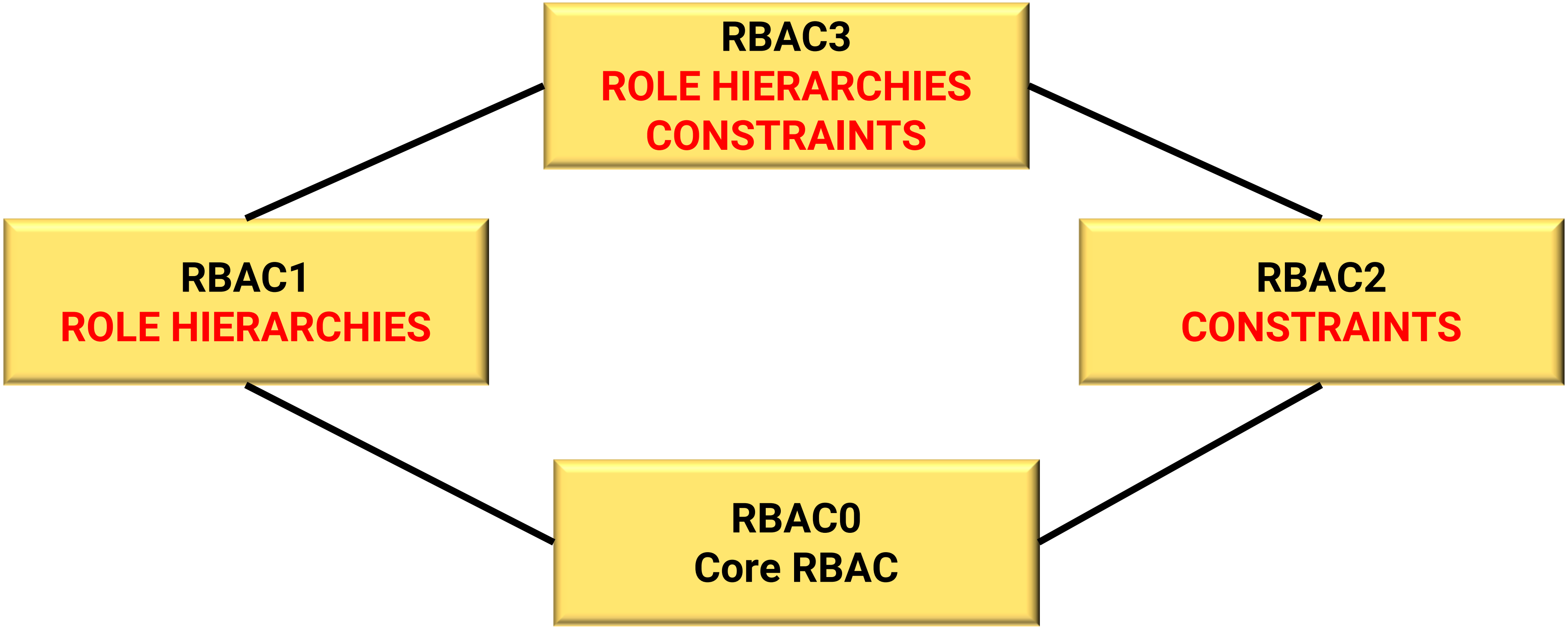
Role-Permission Assignment

Roles

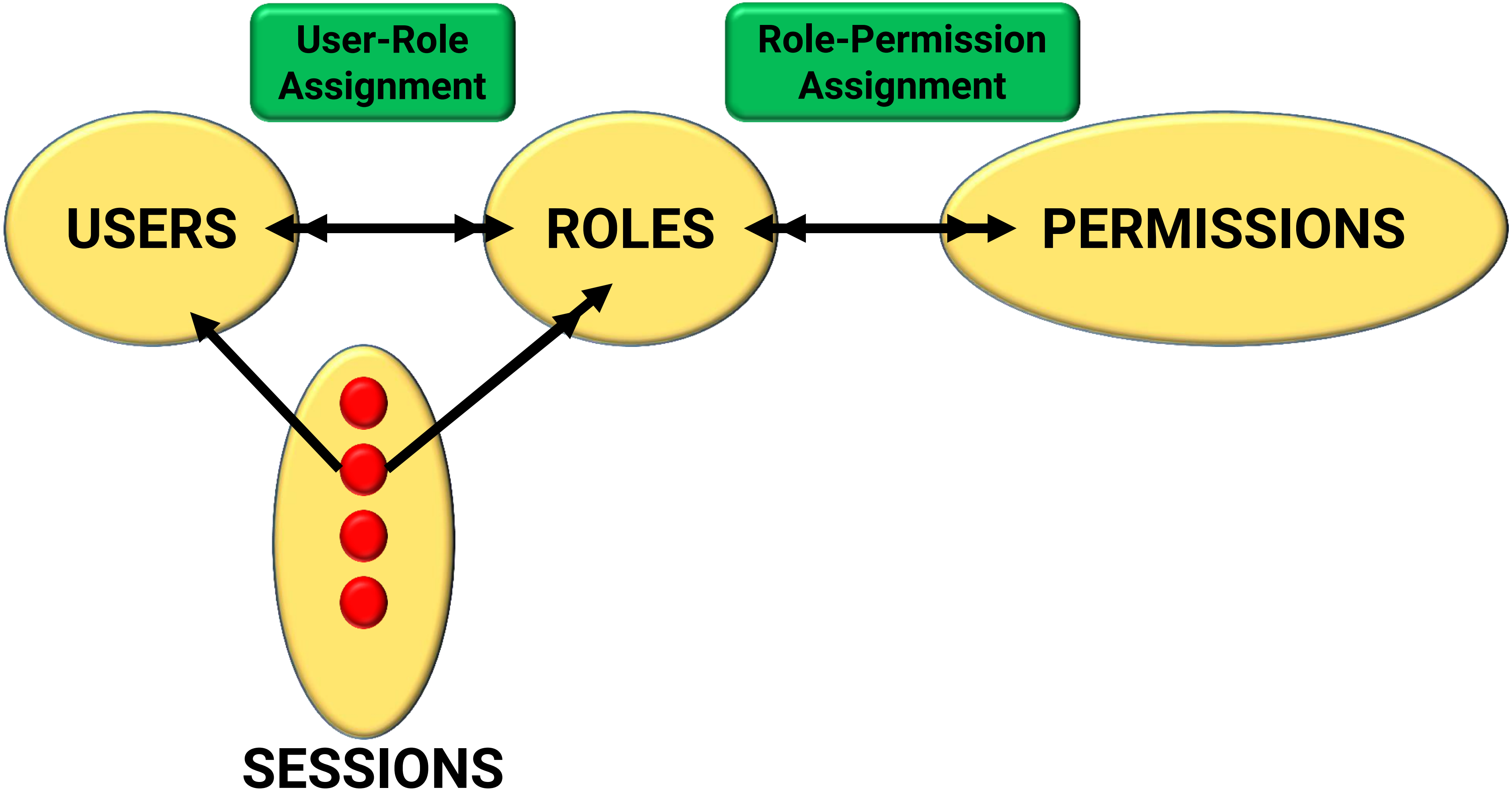
Why Role is the right level of indirection?

- Potentially, fewer relationship to manage
  - O(mn) to O(m+n), m = #users, n = #permissions
- Organizations operate based on roles
- Roles can give a semantic meaning to why someone needs a specific permission
- A role may be more stable than
  - The collection of users and the collection of permissions that are associated with them
- Revocation, granting, or changing of permissions become much easier

RBAC96 Family of Models (Sandhu et. al.)



RBAC0 – Core RBAC



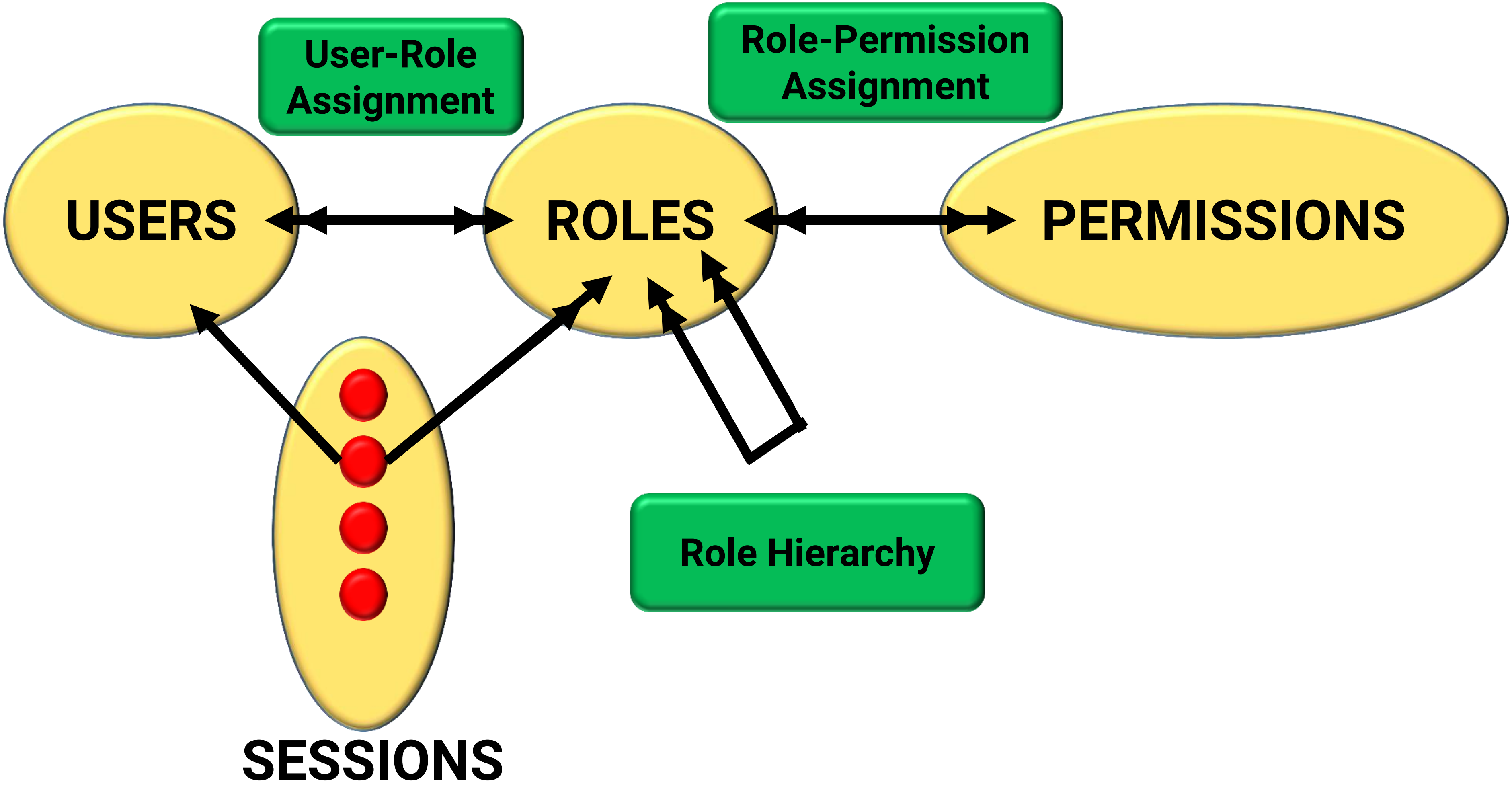
Permissions

- Left intentionally abstract in the RBAC96 model
- Permissions are only **positive**
- No negative permissions or denials
  - **Closed policy**
  - All access are denied unless explicitly authorized by the policy
- No obligations or future requirements
  - Example: If a nurse accesses a patient’s psychotherapy notes, then she must notify patient within 30 days of access

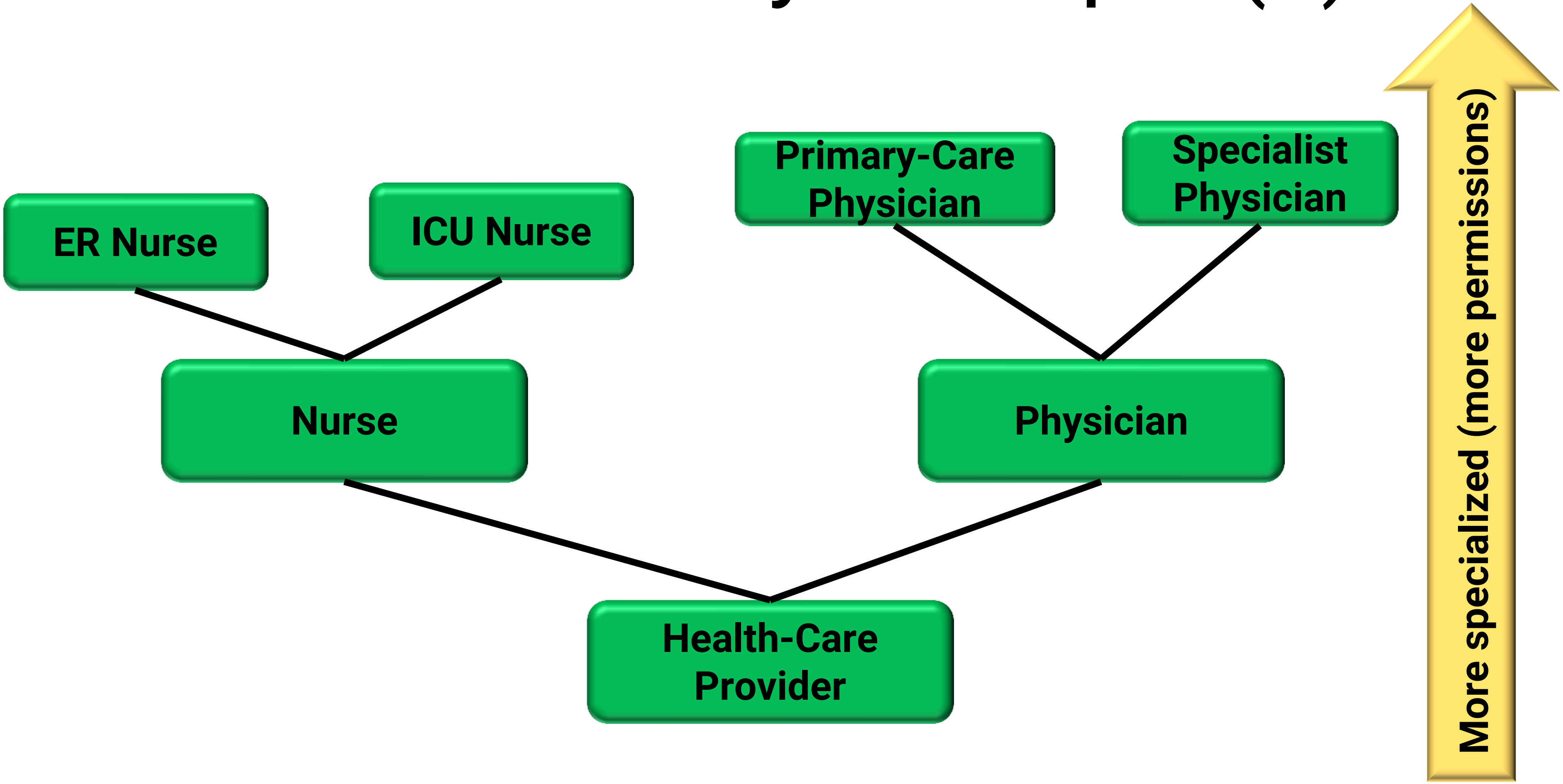
RBAC0- Formal Model

- Vocabulary:
  - U (users), R (roles), P (permissions), S (sessions)
- Static Relations:
  - Permission assignment, PA $\subseteq$ P X R
  - User assignment, UA $\subseteq$ U X R
- Dynamic Relations:
  - **user**: S → U each session has one user
  - **roles**: S → 2^{R} and some activated roles
    - Requires: roles(s) $\subseteq \{r \mid \langle user(s), r\rangle \in \mathbf{UA}\}$
  - Session s has permissions
    - $\bigcup_{r \in roles(s)} \{ p \mid \langle p, r\rangle \in \mathbf{PA} \}$

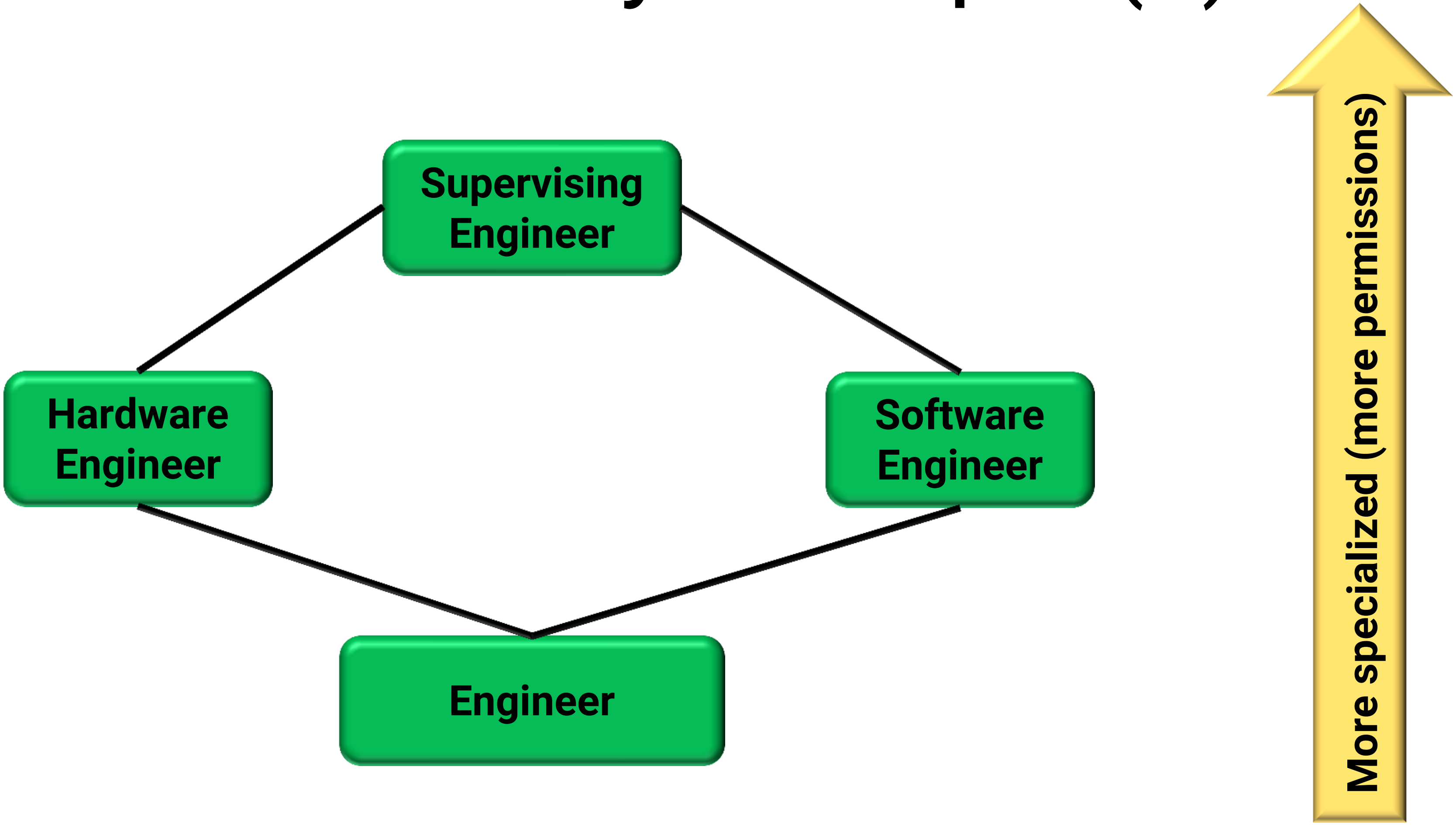
RBAC1– Role Hiearchies



Role Hierarchy Example (1)



Role Hierarchy Example (2)



Supervising Engineer

Hardware Engineer

Software Engineer

Engineer

More specialized (more permissions)

Semantics of Role Hiearchies

- User inheritance
  - r1≥ r2 means every user who has r1 also has r2
- Permission inheritance
  - r1 ≥ r2 means every permission that belongs to r2 also belongs to 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

- From RBAC0: U, R, P, S, PA, UA
- RH ⊆ R X R : a partial order on R, written as $\geq$
  - When r1 $\geq$ r2: r1 is a **senior role** than r2; r2 is a **junior role** than r1
- roles: S → 2^{R}
  - Requires roles(s) ⊆ $\{r \mid \exists r'[(r \geq r') \wedge (users(s), r') \in UA]\}$
- Session s includes permissions $\cup\, r \in roles(s).\{p \mid \exists r''[(r \geq r'') \wedge (p, r'') \in PA]\}$

RBAC2 – RBAC0 + Constraints

- No formal model specified
- Example constraints
  - Mutual exclusion
    - Can be assigned (static) or can activate (dynamic) only one role from the set
    - Enforces separation of duty
  - Pre-condition
    - Can be assigned a role if the user possesses some other pre-condition role
    - Can be used to enforce least privilege principle
  - Cardinality
    - Maximum users that can be assigned a role
    - Maximum roles any user can possess (possibly, in a session)
    - Maximum roles having a certain permission

Mutual Exclusion Constraints

- Mutually exclusion roles
  - Static exclusion – No user can hold both roles
    - Static separation of duty
    - Preventing a user from having too much privilege
  - Dynamic exclusion – No user can activate both roles in the same session
    - Dynamic separation of duty
    - Interact with role hierarchy interpretation

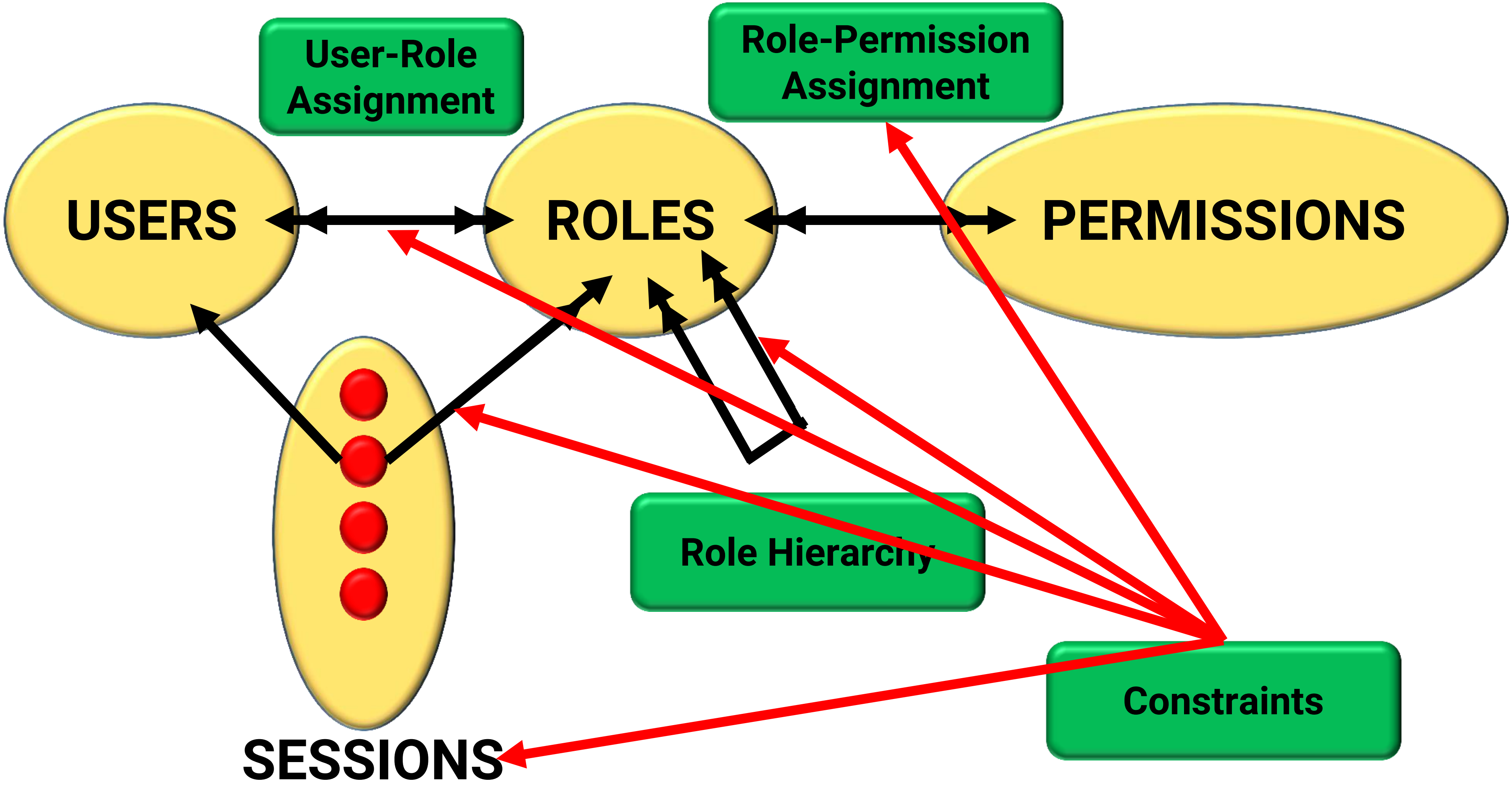
Cardinality Constraints

- 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 role activation
  - At most K users can activate a role
  - ……

Why Constraints?

- For laying out higher level organizational policy
    - When the administrator is centralized it is a sanity checking tool
        - Not essential for a vigilant administrator as he can check all organizational policies are met when making any changes to the RBAC policies
        - Assertion checking in programming languages
- A tool to enforce high-level policies when the administrator is decentralized

RBAC3



User-Role Assignment

Role-Permission Assignment

USERS

ROLES

PERMISSIONS

SESSIONS

Role Hierarchy

Constraints

Products Using RBAC

- Database Management Systems (DBMS)
  - Oracle, PostgreSQL
- Enterprise Security Management
  - IBM Tivoli Identity Manager (central administration and provisioning of account, resources)
- Many operating systems claim to use roles
  - Windows Server 2003, Solaris

NIST Standard for RBAC

- 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.

- The model has number of flaws including typos, errors in mathematical definitions, and other high-level design choices.

Overview of the NIST Standard for RBAC

- Core RBAC and with the following extensions
    - Hierarchical RBAC
    - Static separation of duties
    - Dynamic separation of duties

Advantages of RBAC

- Allows efficient security management
- Principle of least privilege
- Separation of duty to prevent fraud
- Allows grouping of objects/ users
- Policy-neutral – provides generality

Advantages of RBAC (contd.)

TASK	RBAC	NON-RBAC	DIFFERENCE
Assign existing privileges to new users	6.14	11.39	5.25
Change existing users' privileges	9.29	10.24	0.95
Establish new privileges for existing users	8.26	9.26	0.40
Termination of privileges	0.81	1.32	0.51

Estimated time in minutes

Cost Benefit of RBAC

- Saves about **7 minutes** per employee, per year in administrative functions
  - Average IT administrator salary -- **$59.27 per hour**
  - The annual cost saving is
    - **$6,924 / 1,000**;
    - **$692,471/100,000**

Research Challenges in RBAC

- Role engineering
  - Design roles for an access control scenario
  - **Top down approach**: start from analyzing business requirements
  - **Bottom up approach**:
    - *Role mining* – mine existing access control data for roles
- Effective administration of RBAC systems
- Effective usage of constraints

Administrative RBAC

- Administrative roles assigned to administrators
- Sub-models: URA (User-Role assignment), PRA (Permission-Role assignment), RRA (Role-Role assignment)
- Can_assign(ar, Φ, G)
  - Can_assign(Administrator, Physician, Specialist-Physician)
- Can_revoke(ar, G)
  - Can_revoke(Administrator, Physician)
- PRA and RRA are out-of-scope of this class

Attribute-Based Access Control Model

- An access control model where subjects' requests to perform operations on objects are granted or denied based on –
  - attributes of the subject,
    - Job, role, clearance, division/unit, location
  - attributes of the object,
    - Sensitivity level, type
  - contextual or environmental condition,
    - Location, time, state of emergency
  - And a set of policies defined based on the attributes and those conditions
    - A list of rules, firewall rules

Questions to Ponder on

- Can you use RBAC to express DAC?
- Can you use RBAC to express MAC?
- Can you use DAC to express RBAC?
- Can you use MAC to express RBAC?
- In which contexts, DAC makes more sense than RBAC?

Coming Attraction

Network Security:
DNS Cache Poisoning

Acknowledgement

Some of the slide materials are inspired by slides from Ninghui Li and James Joshi.