Access Control with Jess

CS3025, Knowledge-Based Systems
Lecture 19

Yuting Zhao
Yuting.zhao@gmail.com

2017-11-21

Outline

- **Security**
- **Authorization Policies**
  - Access Control
  - Role-based Access control (RBAC)
- **Design RBAC & implementation by Jess**
  - Thanks to Bodo Eggert (University of Hamburg)

  (https://web.sec.uni-passau.de/members/christopher/bodo07bacc.pdf)

What goes into system security?

- Authentication (password/crypto/etc.)
  - Who are you?
- Authorization (Access control)
  - What are you allowed to do.
- Enforcement Mechanism
  - How its policy implemented/enforced

Night Club Example

- Authentication
  - ID Check
- Access Control
  - Over 18 - allowed in
  - Over 21 - allowed to drink
  - On VIP List - allowed to access VIP area
- Enforcement Mechanism
  - Walls, Doors, Locks, Bouncers

Night Club Example: More Interesting Phenomena

- Tickets
  - Name or anonymous?
  - Date

- What if you want to leave and come back
  - Hand stamp or bracelet

Outline

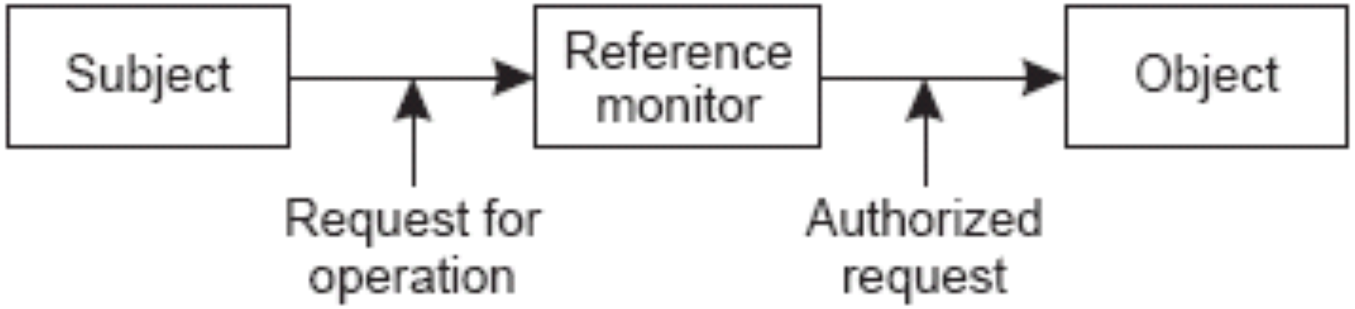
- **Security**
- **Authorization Policies**
  - Access Control
  - Role-based Access control (RBAC)
- **Design RBAC & implementation by Jess**
  - Thanks to Bodo Eggert (University of Hamburg)

(https://web.sec.uni-passau.de/members/christopher/bodo07bacc.pdf)

Access Control

- Access Control is a means by which the ability is explicitly enabled or restricted in some way (usually through physical and system-based controls)
- Computer-based access controls can prescribe not only who or what process may have access to a specific system resource, but also the type of access that is permitted.
- Once a client and a server have established a secure channel, the client can issue requests to the server
- Requests can only be carried out if the client has sufficient ***access rights***
- The verification of access rights is ***access control***, and the granting of access rights is ***authorization***
  - These two terms are often used interchangeably

The Basic Model for Access Control



- The ***subject*** (**user**) issues requests to access the ***object***, and protection is enforced by a ***reference*** *monitor* that knows which subjects are allowed to issue which requests

Access Control Matrix

- The ***access control matrix*** is a matrix with each subject represented by a row, and each object represented by a column
- The entry *M[**s**, **o**]* lists the operations that subject ***s*** may carry out on object ***o***
- Access **Rights** (operations)
  - e.g. Simple:
    - Read,
    - Write
  - e.g. Complex:
    - execute,
    - change ownership

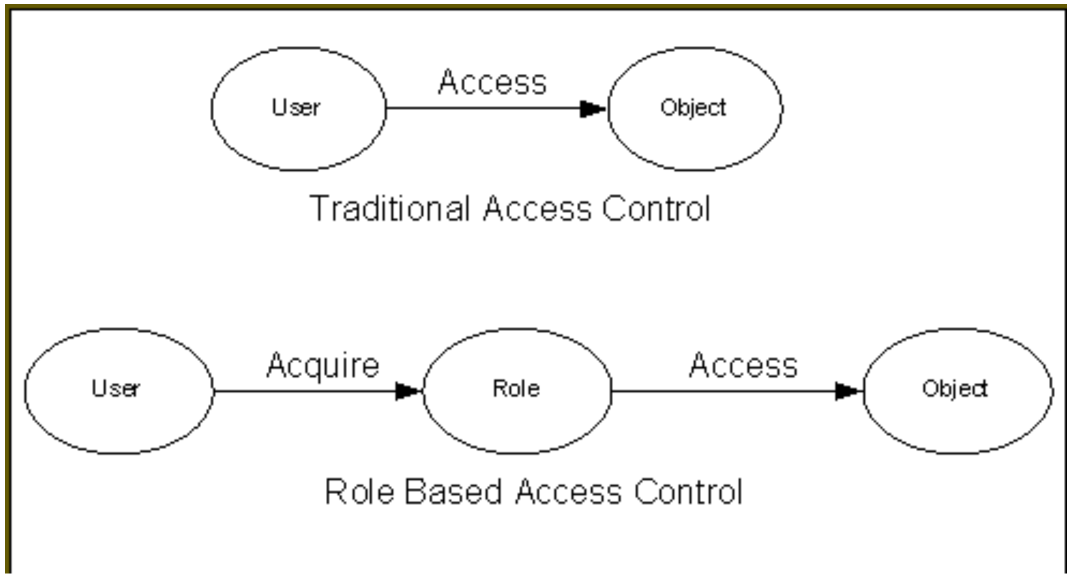
Objects

subjects

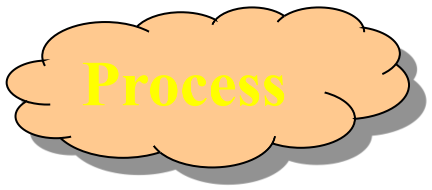
	A	B	C	D
alice	r	r/w	r	-
bob	r	r	-	r/w
charlie	-	-	w	-
dave		r/w	-	w

Role Based Access Control (RBAC)

- With Role-Based Access Control (**RBAC**), access decisions are based on the roles that individual users have as a part of an organization.
- Roles are closely related to the concept of user groups in access controls.
- Role brings together a set of users on one side and a set of permissions on the other whereas user groups are typically defined as a set of users.



USERS

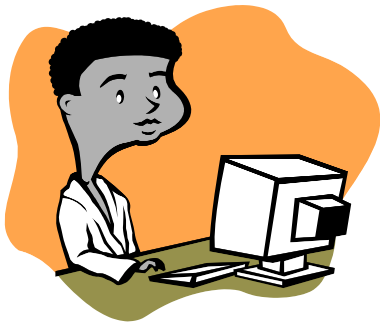


Person

Intelligent Agent

ROLES

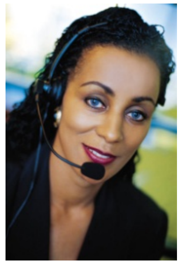
An organizational **job function** with a clear definition of inherent responsibility and authority (permissions).



Developer

Budget Manager

Director



Help Desk Representative

Relation between USERS & PRMS

OPERATIONS

An **execution** of an a program specific function that's invocated by a user.

- ✓ Database – Update Insert Append
- ✓ Delete Locks – Open   Close
- ✓ Reports – Create View   Print
- ✓ Applications - Read Write Execute

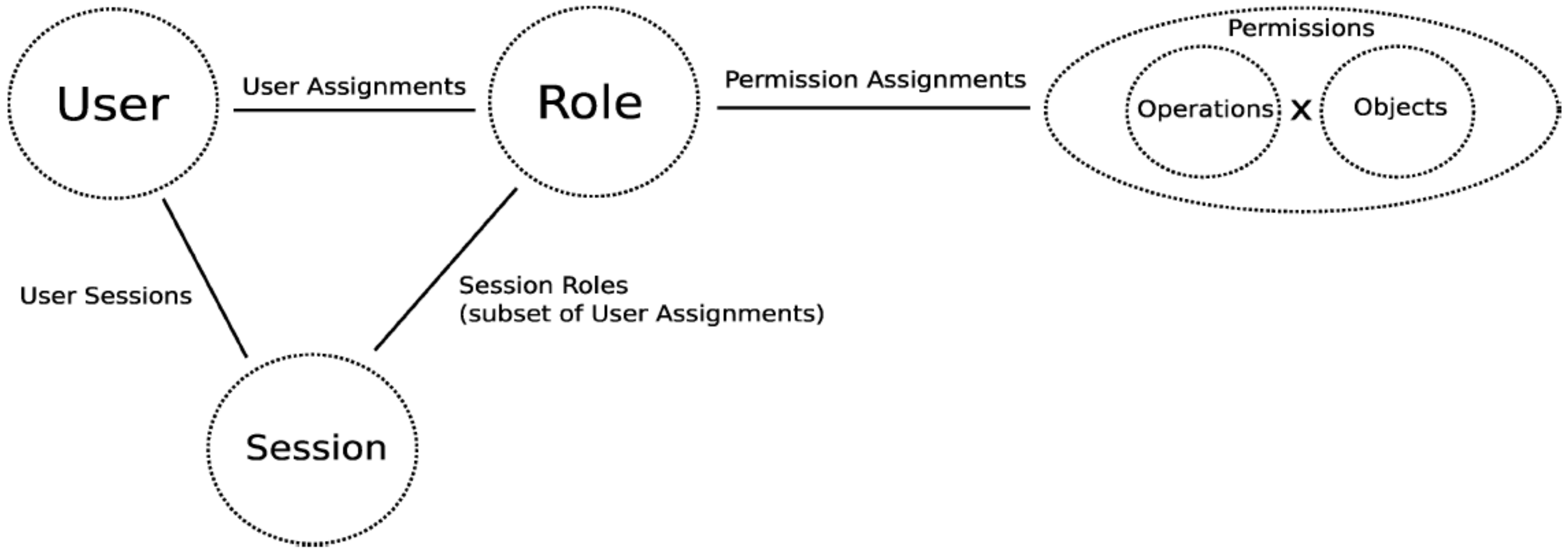
OBJECTS

An **entity** that contains or receives information, or has exhaustible system resources.

- OS Files or Directories
- DB Columns, Rows, Tables, or Views
- Printer
- Disk Space
- Lock Mechanisms

RBAC will deal with all the objects listed in the permissions assigned to roles.

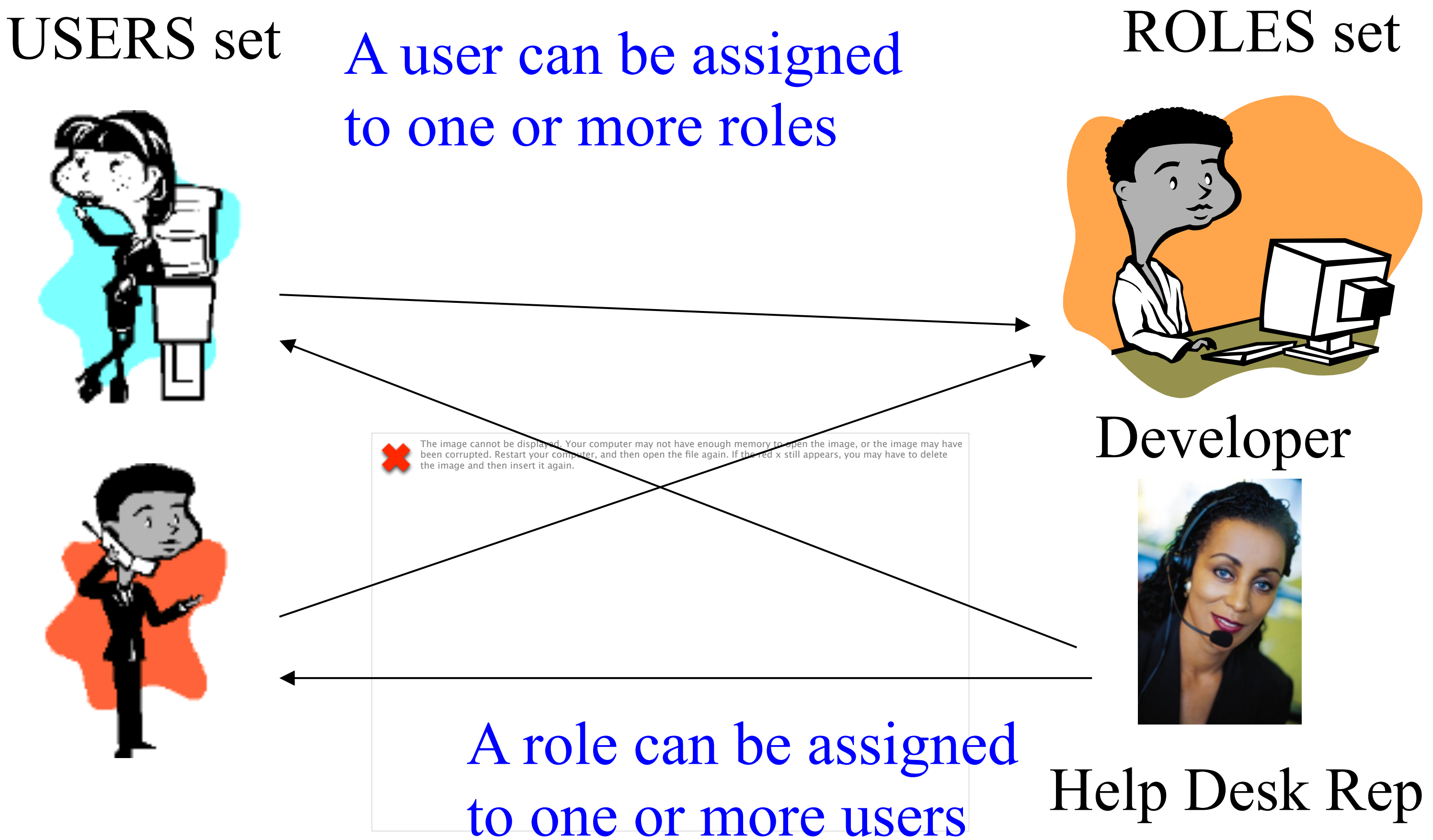
RBAC User and Permission Assignments



RBAC User and Permission Assignments [*The RBAC American National Standard*, 2004]

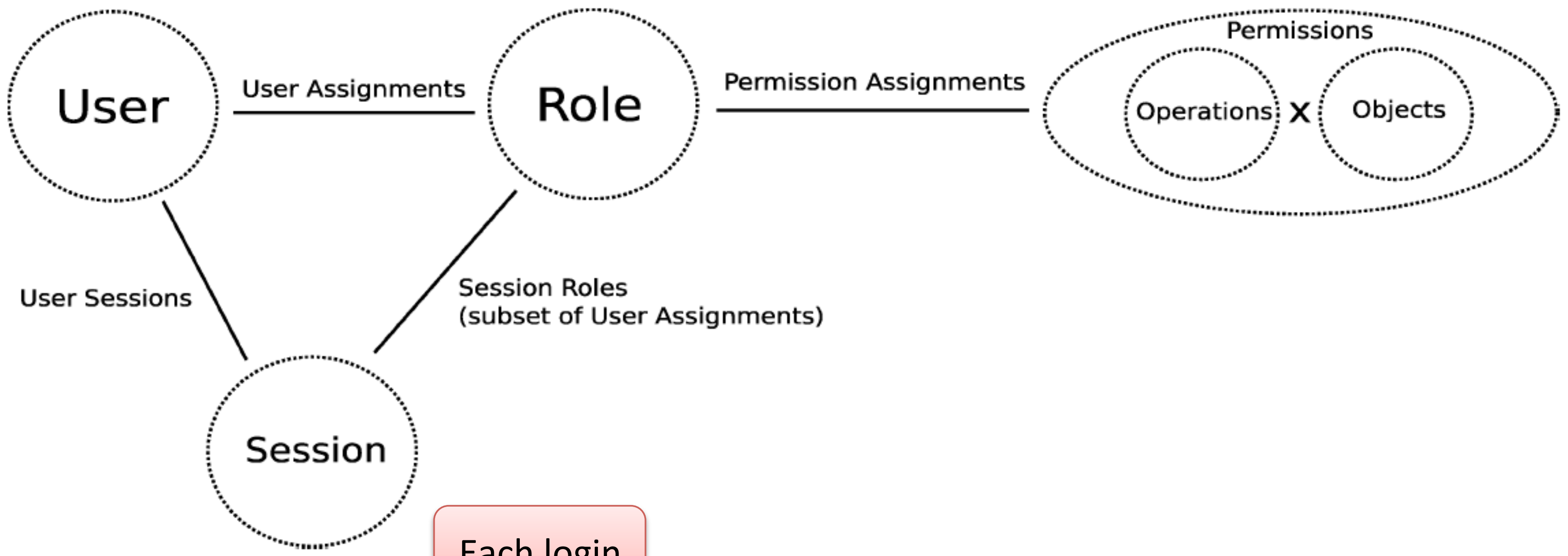
USER Assignment

A user can be assigned
to one or more roles



A role can be assigned
to one or more users

RBAC User and Permission Assignments



Each login

RBAC User and Permission Assignments [*The RBAC American National Standard*, 2004]

Outline

- **Security**
- **Authorization Policies**
  - Access Control
  - Role-based Access control (RBAC)
- **Design RBAC & implementation by Jess**
  - Thanks to **Bodo Eggert** (University of Hamburg)

(https://web.sec.uni-passau.de/members/christopher/bodo07bacc.pdf)

RBAC Data model

- User Assignment and Permission Assignment data

```
(deftemplate UA
"The User Assignment m:n relation"
            (slot uid)
            (slot role))
```

```
(deftemplate grant-by-role-intent
"The Permission Assignment m:n:o relation"
            (slot urole)
            (slot intent)
            (slot objtype))
```

```
e.g.  (assert (UA (uid "Dave Null") (role "programmer")) )
      (grant-by-role-intent (urole "user") (intent "read") (objtype "system"))
```

RBAC Data model

- Sessions

```
(deftemplate CreateSession
"Request for creating a session
id: The desired session id
uid: The identity of the user
roles: The set of requested roles
groles: Internal use, the list of roles granted so far
result: The result of the rule evaluation"
		(slot id)
		(slot uid)
		(multislot roles)
		(multislot groles)
		(slot result))
```

e.g. (assert (CreateSession (id 1) (uid "Dave Null") (roles "admin")))

RBAC Data model

- Access Control

```
(deftemplate Access
"Access request from a session for a file.
session: The user session which requested this access
objtype: The type of the requested object
intent: The intent, e.g. read, write, print ...
result: The result of the rule evaluation"
            (slot session)
            (slot objtype)
            (slot intent)
            (slot result))
```

```
e.g.  (assert (Access (session 1) (objtype "system") (intent "read")) )
```

Operations on the RBAC Data

- Allow Access by Role

```
(defrule access-by-role-r
"This rule allows access to an object, if there is a Permission Assignment granting
access to an object for a member role of this session."

    ?f <- (Access (session ?session) (objtype ?objtype) (result nil))
    (Session (roles $?roles) (id ?session) )
    (grant-by-role (urole ?urole & :(member$ ?urole ?roles)) (objtype ?objtype) )
=>
    (printout t "Access Granted" crlf)
    (modify ?f (result "granted")))
```

Operations on the RBAC Data

- Default Deny Session Creating Rule

```
(defrule default_deny_Session
"This rule will fire if there are no other rules,
providing the default-deny behaviour"
                        ; Don't run unless there are no other rules:
            (declare (salience -100))
                        ; fire only for rules without a result:
            ?f <- (CreateSession (result nil))
=>
            (printout t "I'm sorry, Dave, I can't do that" crlf)
            (modify ?f (result "denied")))
```

Test

- rbac.jess
- policy.jess

```
(deffacts access-by-role-info
(UA (uid "root") (role "programmer"))
(UA (uid "root") (role "user"))
(UA (uid "root") (role "admin"))
(UA (uid "Dave Null") (role "programmer"))
(UA (uid "Dave Null") (role "user"))
(DSD (n 2) (roles "programmer" "admin"))
(grant-by-role
(urole "admin") (objtype "system"))
(grant-by-role
(urole "user") (objtype "userhome"))
(grant-by-role-intent
(urole "user")(intent "read") (objtype "system"))
)
```

test

```
Jess> (batch rbac.jess)
Jess> (batch policy.jess)
Jess> (reset)
Jess> (assert (CreateSession (id 1) (uid "Dave Null") (roles "admin")))
Jess> (run)
I’m sorry, Dave, I can’t do that
Jess> (facts)
...
f-10   (MAIN::CreateSession (id 1) (uid "Dave Null") (roles "admin") (groles ) (result "denied"))
Jess> (assert (CreateSession (id 1) (uid "Dave Null") (roles "user")))
Jess> (assert (Access (session 1) (objtype "system") (intent "read")))
Jess> (run)
Access Granted
Jess> (assert (Access (session 1) (objtype "system") (intent "write")))
Jess> (run)
Permission Denied
```

Special thanks to

- Bodo Eggert (University of Hamburg)
- Dr. Saeed Rajput & Reena Cherukuri
- Trent Jaeger (IBM)
- Tal Garfinkel

… for their slides

Summary

- Access control
  - Securtiy
  - RBAC
- Design RBAC & implementation by Jess

  …
- Question?