# Spocp and selected use cases



### Course Deadline Extended (B2) (B2)

*A student in Dr. Schonfeld's Ordinary Differential Equations course is unable to attend the final exam due to an authorized absence (a death in her family). Professor Schonfeld has removed access in the LMS to her class notes for the prior semester's students, since the semester is at an end, but she makes an exception for the student at the request of the Dean, and grants her access to the course space in the LMS for an additional week in order to complete studying for the make-up exam. One week later, the student's access is automatically removed by the system.*

Let's assume the following

- course id for the "Ordinary Differential Equations course" is ODE1
- The students userid is abc001
- Identifier for the system is LMS
- Today's data is October 3rd 2010, a week from now is October 10th

A plausible access rule would then be:

(LMS (resource course ODE01)(action read)(subject student abc001))

If one wanted one could add the endtime

(LMS (resource ODE01)(action read)(subject student abc001)(time (* range le "2010-10-11T00:00:00Z")))

In which case the rule engine would handle the stop date and the rule could be
removed sometime later.

(* range le "2010-10-11T00:00:00Z") means that the rule will be valid upon till "2010-10-11T00:00:00Z".

A query from the LMS system would then be of the form:

(LMS (resource ODE01)(action read)(subject student abc001)(time "2010-10-03T10:31:23Z")))

where time is the time of the query for authorization. If the time limit is handled by Spocp.

AND

(LMS (resource ODE01)(action read)(subject student abc001))

if the time limit is handled by something else.

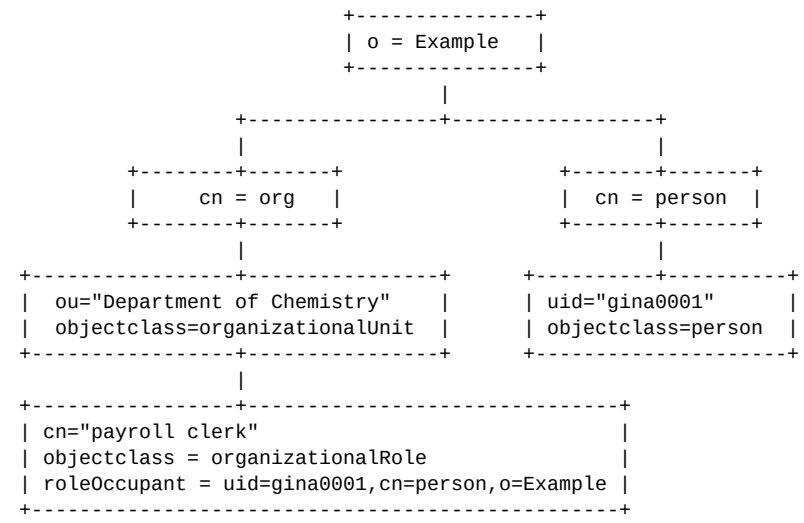
### Old and New Payroll Clerks (A8) (A8)

*Gina, an administrative assistant in the Department of Chemistry, vacates her position in the department to take a new position in the Office of the Comptroller. Gina has been the department's payroll clerk for a number of years. The department chair chooses his executive assistant, Marcus, to take over as payroll clerk for the department. As payroll clerk, Marcus will need access to sensitive payroll information about non-exempt employees in the department, but will not need access to faculty salary information or student records. The department chair logs into an access management system and designates Marcus as the new payroll clerk for the Department of Chemistry. In so doing, he grants Marcus a collection of rights within various financial applications appropriate for a departmental payroll clerk in his department, and Gina (who is still employed by the university and still recognized by the authorization system as a user) has her payroll clerk privileges for the Chemistry department revoked.*

A couple of asssumptions:

- There is a role "payroll clerk" in the LDAP directory and it is expressed as an organizationalRole object immediately below a organizational unit object.
- The following DIT structure is assume. The solution can easily be adapted to other structures.

**DIT**




The privileges for payroll clerks is predefined and the same disregarding ou.
This set of rules are therefor entered into the SPOCP ruleset by the authority that decides on what payroll clerks can do, sometime early in the process.

Say that you have the following permission that you want to express:

- The permission to read payroll information about non-exempt
- The permission to maintain (add, delete, modify groups) the group organization
- The permission to maintain (add, delete, modify) the FYI documents

You also realize that if you don't want to make specific rules per person, you have to enter a check if the person matches a condition.
This is done by using a boundary condition.

Possible rule formats:

(FA (payroll non-exempt)(domain)(action read)(subject)) => (ref payroll_clerk)
(FA (organization group)(domain)(action)(subject)) => (ref payroll_clerk)
(FA (document fyi)(domain)(action)(subject)) => (ref payroll_clerk)

A query would them have the format:

(FA (payroll non-exempt)(domain OU)(action A)(subject X))

OU is the name of the organizational unit
X is the uid of the person who wants access.
A is the action ( read, write, add, delete, ...)

The last element of the set starting with "domain" is always the name of the organizational unit. Which means that in this case that is the domain for which authorization can be issued.

Next you construct the boundary condition. Which checks if a person has the role payroll clerk at a specific organizational unit. This works for all payroll clerks since the boundary condition also checks the ou.
Given that the LDAP server DIT follows the above mentioned pattern you would get the following condition.

**ldapset condition**

```
payroll_clerk := 'ldapset:{//domain[1]}{//subject[last]}:ldap.example.edu;cn=person,o=example;cn=org,o=example;
{{\\1%ou & ${0}}%cn & "payrole clerk"}\roleOccupant & {\\0$uid & ${1}}}'
```

A bit cryptic but spelled out:
This boundary condition, uses the 'ldapset' backend and does the following thing in sequence (all using ldap.example.edu as the LDAP server):
1. from the query picks the second element in the set starting with "domain", this becomes argument 0 (${0})

2. from the query picks the last element in the set that starts with "subject", this becomes argument 1 (${1})

3. Does a LDAP one-level search with the filter (ou=<argument 0>) with cn=org, o=example as baseDN.

This will return a DN for the organizational unit.
4. Find the object with cn="payroll clerk" immediately below the ou from (3).
This ought to be an object of the organizationRole type. You can check for that, but I've left that out here.

5. Does a LDAP one-level search with the filter (uid=<argument 1>) with cn=person,o=example as baseDN.
This will return the DN of the person

6. Checks if the DN of the person exists as value on roleOccupant of the object from (4).
Baselevel search with the filter (roleOccupant=<personDN>).

If all is OK then this boundary condition will return True.

The financial applications then has to be trained to pose the right kind of questions to the Spocp server.

Once this is setup, giving someone the "payroll clerk" permissions, at an organizational unit, in the financial applications amounts to, in the LDAP server, adding the DN of that person as a value on roleOccupant in the organizationalRole object with the name "payroll clerk" below the relevant ou object.

Removing the permissions is done by removing the persons DN from the roleOccupant attribute in the object mentioned above.

(provided by Roland Hedberg)