Design for integrating LDAP with RTC using external plugins

# Introduction

RTC currently provides LDAP sync to allow customers to have users pulled from LDAP for authentication as well as define which users are Jazz Admin or Jazz User. However, it does not support any external group definitions or the ability for LDAP to determine the team membership, roles for individuals and allocated licenses. The purpose of this design document is to provide a services solution to allow group membership to be defined in LDAP along with the appropriate roles and these will then be automatically synced to RTC. The sync will be both in adding users and removing users and updating licenses.

Major Design Considerations

The major design points as part of the solution are as follows:

* Customer assigns membership to LDAP groups using existing LDAP directory browsers
* The solution must be able to run on a defined schedule or kicked off manually when required
* The solution will assign users to project areas and/or team areas in RTC
* The solution will assign roles to users in project areas and team areas
* The solution will assign client access licenses to users
* The solution must sync users in groups adding and removing as required
* The solution will update user name and email address

# Overview

This solution is intended to provide a temporary work around until the RTC product provides the capability within the product. The solution should be built in a customer agnostic way so it can be used by any customers who need this function until the product function is available.

The Jazz foundation requirement to provide the product capability is:

This function allows users to have a single point of defining user access to capability through LDAP. No user administration in RTC should be required with this sync function available. The LDAP sync needs to work with LDAP and when LDAP is backed by RACF.

The following picture visually describes how the function would work. The items specified in Yellow are the parts the customer must update by updating the LDAP groups and the mapping file. The blue indicates RTC that is updated only via the custom code and the grey box represents this solution.



# LDAP Entries for RTC

This section describes the design for the LDAP entries that will be used by the synchronizer to update RTC server configurations. The design is described and demonstrated through a concrete example based on the Money That Matters sample CLM application, as configured in an example.com LDAP server. Screen captures are from the Apache Directory Studio plugin in eclipse. Other LDAP directory browsers may also be used.

The LDAP server has an LDAP organizationalUnit for all LDAP users and one for managing RTC. The LDAP users could also be organized into LDAP groups.../../../../../../../temp/LDAP-RTCSy



The RTC organizationalUnit contain a nested organizationalUnit for each server



Each RTC server organizes its users (JazzAdmins, JazzUsers, etc.), client access licenses, and project areas.



JazzAdmins and JazzUsers are LDAP groupsOfNames whose members play the corresponding role in the RTC server. Adding an LDAP group as a member will automatically add all the users in that group to the JazzAdmins or JazzUsers role in RTC. Grouping is simply a convenience for reusing collections of related users that should be treated the same way.



The Licenses organizationalUnit contains an LDAP groupOfNames for each client access license key installed for that server.



Each client access license contains members who have been assigned that license.



An RTC server also contains project areas. A project area is an LDAP organizationalUnit that contains LDAP groupOfNames for project area administrators and users.



The Administrators and Members groups contain the users that have been added to the project area.



An RTC project area also contains an organizationalRole for process roles, each of which contains an LDAP groupOfNames for each process role:





Project areas also contain LDAP organizationalUnits for RTC team areas, organized in hierarchies. Team areas also have Administrators, Members and Process Roles since team members may play different roles in different team areas.



# Implementation

The implementation consists of an LDAP 🡪 RTC synchronizer Java application called LDAP2RTCSync. This application can be invoked from the command line or scheduled to run periodically using typical operating system application scheduling techniques such as UNIX cron. For example, consider the following command:

LSAP2RTCSync --directory ldap://9.32.253.195:10389 –admin uid=admin,ou=system --password password

This command starts the synchronizer, reading the RTC information from the given LDAP server using the provided credentials. This command will:

1. Iterate over all the RTC servers in the ou=RTC organizational unit
2. For each server, it will:
   1. Update all the server users given the JazzAdmins, JazzUsers, etc. roles, adding and archiving users as needed.
   2. Allocate client access licenses for the users as specified in the Licenses organizational unit
   3. For each project area organizational unit in the server it will:
      1. Add Members and Administrators to the project area
      2. Assign the users the roles given in Process Roles
      3. For each team area organizational unit in the project area it will (recursively):
         1. Add Members and Administrators to the project area
         2. Assign the users the roles given in Process Roles

Additional rules:

1. RTC administrator creates the project areas, team areas, and add client access license keys to the server using the RTC admin UI.
2. LDAP users that are removed from the LDAP server, or any of the RTC servers result in archived users in the RTC server, not removed users.
3. Mismatch between ou or cn names in LDAP with item names in RTC will result in errors. The synchronizer does not create, update or archive any project or team area names, and cannot change project area process descriptions (or process roles).
4. The synchronizer will not support any kind of undo function.
5. Deleting servers, project areas, team areas, roles and/or client access license keys in LDAP has no impact on the corresponding items in RTC. RTC admin functions are required to make these changes.
6. Renaming servers, project areas, team areas, roles and/or client access license keys in LDAP has no impact on the corresponding items in RTC and will result in synchronizer errors until the corresponding updates are made in RTC. RTC admin functions are required to make these changes.

# Open Questions

1. Is a mapping between LDAP common names and RTC item names required, or can they be the same?

Currently the LDAP / RTC userSync function requires the LDAP uid and RTC user ids to be the same in order to detect users that should be added or removed from RTC. This design would like to extend that correlation to apply to all project area, team area, role and client access license names. This eliminates the need to create and manage mapping files between LDAP and RTC.

1. Does the LDAP interface to RACF support the LDAP schema and naming conventions used in the design? We can use LDAP federation to point to another LDAP to do this mapping.

The z/OS LDAP server supports multiple backends including RACF (using SDBM) and TMDB (using DB2). A TMDB backend is required to support the current repotools-userSync application, and this can be used to add the additional configuration information needed to meet the requirements.

1. Do all update operations performed by the synchronizer need to be logged, or only errors and warnings? Should only log errors and warnings to keep the log file from blowing up.
2. Is there a need for a –simulate option so that LDAP2RTCSync can log what it will do without changing anything in RTC? This may be useful for verifying and updating LDAP configurations while minimizing unintended changes in RTC.
3. Is an LDAP schema for RTC required or desirable? The current design uses OOTB LDAP schemas (which may not be available in RACF). However, an LDAP schema specific for RTC may be useful and might simplify and improve the usability of administering this information using an LDAP directory editor/browser. Treat this as optional or lower priority.

# RACF to RTC Mapping File

If a mapping file should be needed to address z/OS IBM Tivoli Directory Server limitations, the file below provides an example that has the same configuration as the LDAP example above. In this case, the LDAP group names are translated to RTC servers, project areas, team areas, client access license keys and process roles. Users who are members of those groups in LDAP would be configured appropriately in the corresponding RTC items.

{

"LDAPConnection": {

"URI": "ldap://9.32.253.195:10389/",

"adminid": "admin",

"adminPassword": "password"

},

"RTC01": {

"serverURI": "https://ce4iot.rtp.raleigh.ibm.com:9443/ccm",

"Licenses": [

{"RDMFLT": "Rational Rhapsody Design Management Contributor-Floating"},

{"CLMFLT": "Rational Solution for Collaborative Lifecycle Management Contributor-Floating (Unlocked"}

],

"Project Areas": [

{

"ou":"PA01",

"name": "JKE Banking (Change Management",

"Process Roles": [

{"R01": "Product Owner"},

{"R02": "Scrum Master"},

{"R03": "Stakeholder"},

{"R04": "Team Membr"}

],

"TA01": {

"name": "Business Recovery Matters",

"Process Roles": [

{"R01": "Product Owner"},

{"R02": "Scrum Master"},

{"R03": "Stakeholder"},

{"R04": "Team Membr"}

]

},

"TA02": {

"name": "Energy Efficiency Matters"

},

"TA03": {

"name": "Release Engineering"

}

},

{

"ou": "PA02",

"name": "pet Store"

}

]

}

}