General

Atollon Directory

Access Rights in Atollon server are based on tree-hierarchical structure. See the attached document for the Atollon Directory scheme.

Roles

Role is **Access Rights** entity that can be assigned to users in connection to **Folder (Account)** or **Project**. **User** must be allowed to fit into specific role. See **User Settings** to add user into role. By adding user into role, you specify that anyone with R (Rights) permissions can assign the **User** into the **Role**. Nothing else. **User** does not get any rights only by adding the user into the **Role** (this is different compared to **Access Group** rights). **User** gets permission specified by **Role** rights only after *Account Manager*, *Project Manager* or any other authorized **User** adds the user into **Role** on **Project** or **Folder** (Account).

Role "Creator"

Creator role is specific system-generated **Role**, which helps assign User into **Creator** role automatically after creating the **Folder** (Account) or **Project** based on **Folder** / **Project Template** .

Define Project Role Rights

Project Template (Folder Template) may define specific access rights that any **User** assigned to **Project Role** on specific **Project** may receive. To amend **Project Role** rights, go to **Project Template Settings** and change **Custom Rights** options. These rights are pre-definable only to new projects. If you want to change rights to existing projects, you should check function of mass-project rights change in **Reporting** (only Win client).

If the **Project Role** rights are not specified by **Project Template**, but are allowed by **Project Type**, system automatically assigns full rights to the user on the project.

Conditions of adding User to Role on Project

1. Check whether **User** may be assigned to **Role** (see User Settings)
2. Check whether **Project Type** contains the specific **Role**
3. Check whether user assigning other users has **R** (Rights) permissions on the particular **Project**.

Please be aware that usually projects are visible (editable / approvable) also by **Access Group** rights to large number of users (P*roject Managers, Administrators, Everyone*, etc.) depending on **Project** default access rights settings (based on **Project Template**).

Access Rights Properties

ACL (Access Control List)

Atollon system access rights utilize per-object access rights. That means each Container or Leaf Node may have it's own Access Rights definition.

Access Rights Definition

Users, Groups and Roles may be assigned to have authorization to **List**, **View**, **Create**, **Edit**, **Authorize** or amend **Rights** of each individual object that is associated to the ACL. Special rights include **Admin** (this right can be editable by root user only and disallows changing this permission to any other users) and **Finalize** (this right means that the permission is set for the current object only and can not be inherited).

Access Rights Inheritance

Access Rights to one object (ACL) may be automatically taken from another ACL (or multiple ACLs). This is used when setting-up rights for more records (messages, documents, etc.) at the same time. It is enought to set rights in parent node / folder / container and the objects linking to this container will get the same rights as the container itself.

Example: Set that the group Everyone will see the project "Company Party". Any message or document created/uploaded under project "Company Party" gets the same rights as the "Company Party" project, because the new message/document has ACL that links to it's parent (the project folder).

How is the inheritance ensured?

Rights are automatically inherited, because they are (ussually) created based on Template ACL. The Template ACL is the object's property that holds the definition of new (child object) ACL to be created. This Template ACL, by default has set that the newly created (child) objects will link to it's parent (current object).

Multi-link ACL

Some records, such as Folders, Projects, Activities or Invoices are multi-linked. That means their ACL is inherited from several parent nodes, incl. for example **Folder Type and Parent Folder** (in case of Folder). When linking rights, **filters** are applied. That means that in order to get L, V rights to the Folder, you have to have L rights to it's parent, etc.

Are there any exceptions?

Yes, in Project or Folder Templates, administrators may set-up different behavior for creating new ACLs for newly created Folders & Projects. They may change the Template ACL to link to different container and in that way change default rights of various Folders & Projects, based on selected Folder or Project Template.

How to check what righs are inherited?

You just open the ACL detail and click on "Show linked rights". The condition is that the ACL links to other ACL.

How do I avoid inheritance

You can either change this in Template ACL (remove the linking) or you can change it on already created object (again, remove the linking). You can not remove rights that were set as Admin. Those can be removed only by super user.

Proxy Rights

Proxy Rights are used to temporarily or permanently give rights of one user to another. To set-up Proxy Rights, you must open User's details (the one that give the Power of Attorney) and add the other user (the one who will get those rights).

Example: Person leaves for vacation and you want other user to take over the responsibility in the time of absence. Go to the absent user's details and add full Power of Attorney to another user. Please note that this change is global, therefore it affects also the user's personal messages and documents.

Enable:

Edit events in other users calendar: Need access on other users timesheets

Conflict: If enabled editing callendars also editing timesheets is enabled and vice versa

Disable:

Found proxy connection

there is script what found you all proxy use on virtual server instances:

listProxtThroughDatabases.sh

move him to /tmp/, su postgress and run:

```
sh /tmp/listProxtThroughDatabases.sh
```

Output looks like:

```
 database  |     id     |  username  |  proxyid  |  proxyname
-----------+------------+------------+-----------+-------------
  harfonie |  190086000 | snadova    |  67385000 | rytova
  harfonie |  190099000 | marsala    |  67385000 | rytova
  harfonie | 1099228000 | hamalova   | 190099000 | marsala
```

Revision #1
Created 23 January 2020 14:03:45 by Jan Safka
Updated 21 March 2020 06:31:24 by Jan Safka