Parking Enterprise Management System

PEMS RBAC Authentication and Menus XML Guide





INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH DUNCAN SOLUTIONS, INC. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN DUNCAN’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, DUNCAN ASSUMES NO LIABILITY WHATSOEVER, AND DUNCAN DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Duncan products are not intended for use in medical, lifesaving, or life sustaining applications.

Duncan may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Duncan reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Duncan sales office or your distributor to obtain the latest specifications and before placing your product order.

RxV, MX and the Duncan logo are trademarks or registered trademarks of Duncan Solutions or its subsidiaries in the United States and other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2008, Duncan Solutions, Inc. All rights reserved.

Table of Contents

1 Introduction 5

2 Theory of Operation 5

2.1 The XML 5

2.2 Creating the XML 6

2.3 Editing the XML 6

2.4 Applying the XML 6

3 RBAC Authorization XML 6

3.1 Overview 6

3.2 Rules 6

3.2.1 Groups 6

3.2.2 Applications 7

3.2.3 Authorizations 7

3.2.4 Menus 9

3.3 XSD 11

4 PEMS Administration Interface 11

4.1 Getting Current XML 11

4.2 Setting New XML 12

**Revision** History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Author | Description | Revision Date |
| 1.0 | Ron Howard | Initial Document | 12/6/2013 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Introduction

# Introduction

Menus and user authorizations are controlled by a role-based access control system (RBAC). This system is implemented using NetSqlAzMan 3.6.0.15 (<http://netsqlazman.codeplex.com/>) and the associated database of PEMS RBAC.

RBAC can be configured or changed through three different processes.

* PEMS System – Through interactions with PEMS system, RBAC may be updated to reflect new roles, new users and the assignment of users to a role.
* NetSqlAzMan Console – A console application is provided during the installation of NetSqlAzMan if installed from the web. (<http://netsqlazman.codeplex.com/>) ***Note: This is not recommended and the console is not installed during installation of PEMS.***
* RBAC Authorization XML – XML defines the attributes of the RBAC system and allows administrative users to configure menus, authorizations, groups and roles.

This document describes the use of PEMS Administration tools to capture and edit a customer’s authorization groups (roles), group membership, authorizations and customer menus via RBAC Authorization XML.

PEMS System provides the tools to capture/download a customer’s present RBAC configuration as RBAC Authorization XML and to upload RBAC Authorization XML to modify a customer’s authorizations and menus. This capability provides a powerful tool by which to manage customers.

The XML configuration of a customer’s RBAC is intended only to be undertaken by an administrator who is thoroughly familiar with the RBAC system in PEMS.

# Theory of Operation

## The XML

The XML is comprised of a five sections. Four of these sections may be used to alter RBAC and the fifth is used for informational purposes only. The sections are:

* **Users** – This section list the users that are assigned to a role for this customer. This section is provided for reference only and any changes to this section will be ignored if the XML is used in an upload process. Users may only be added to a customer via the PEMS application.
* **Groups** – This section represents the roles that exist for a customer. These groups also indicate which users are assigned to a given group. Groups can be added, modified and deleted from a customer.
* **Applications** – This section defines the elements of a customer that can be authorized for different roles and menus. These applications represent an MVC Controller that exists for the customer and can have authorizations given or denied. Underneath each of these applications are operations. These operations represent an MVC Controller Action. These actions ultimately represent the individual pages of the PEMS application.
* **AuthTree** – This section defines groups of operations (pages) that can have authorizations granted or denied to various groups (roles). This section also provides a means to label the authorization groups which is subsequently used in PEMS when authorizing a role to have rights to PEMS functionality.
* **MenuTree** – This section defines the left nav menus that are presented to the user. All menu items are defined in this section. The decision to present the menu or to hide it is based on the present user and the authorizations that user has.

Each section that can affect the authorizations or menus of a customer have a built-in safety process where an XML uploaded for processing will not affect the present settings unless the Action attribute of the XML element is set. This is to prevent accidental modifications to the existing customer settings and require a deliberate edit of the XML file to cause a change.

## Creating the XML

Although an XML file can be built from the ***Duncan.Auth.xsd***, it is not the recommended method. The PEMS Administration site provides a page where present settings of a customer can be extracted and downloaded as an XML file. This file would serve as the basis for any edits to a customer’s authorization or menus. See **Getting Current XML** below for URL and description of this functionality.

## Editing the XML

The XML can be edited with any text editor or your favorite XML environment. Each section of the XML and its associated children can be modified. Unless the XML element has the Action attribute defined with an allowable value, the XML element will not affect the present settings of the customer. All sections are optional. See **Rules** for details on the elements and their attributes and allowable values.

## Applying the XML

The PEMS Administration site provides a page where an updated XML file can be selected and applied to a customer. The process that applies the XML will report on changes made as well as any errors encountered. See **Setting New XML** below for URL and description of this functionality.

# RBAC Authorization XML

## Overview

## Rules

### Groups

Groups are analogous to roles in the PEMS system. For instance, if there is a role called “Administrators”, there will be a group named “Administrators”.

Groups may be added, deleted or have their members updated. The group Name attribute is matched against the NetSqlAzMan “store groups”. The allowable Action attribute values, if attribute exists, are:

* **“Add”** – Add this group, if it does not exist. If it does exist already, do nothing. Walk each child Member element and add if child element Action != “Delete”, add the child element name as a member of the group.
* **“Delete”** – Delete the named group, if it exists. First remove all group members and remove any assignment to application operation rights.
* **“Update”** – Walk child Member elements and act on any child element Action attribute that is defined. This would be used to add/delete users to an existing group (role).

#### Element synopsis

*Element*: Groups – Optional, an element that contains a sequence of Group.

*Element*: Group – Contains a sequence of Member

*Attribute*: Name – Required, name of role in PEMS. Used as a “store group” in NetSqlAzMan.

*Attribute*: SID – Optional, can be ignored.

*Attribute*: Action – Optional, action to perform on this group. For allowable values see the XSD.

*Child Element*: Member – Zero or more members of the group. String, matches name of existing user in PEMS/NetSqlAzMan.

*Attribute*: Action – Optional, action to perform on this member. For allowable values see the XSD.

### Applications

Applications represent the MVC Controllers in the implementation of the PEMS system. Applications can be seen as the first part of the URL following the “/pems/” part of a URL. Each application has operations. These operations are the pages of PEMS and can be seen as the second part of the URL following the “/pems/” part of a URL. For example:

http://dev.pemsportal.com/North%20Sydney%20Council/pems/**Events**/**Index**

This URL would indicate that there is an Application named “Events” which has an Operation named “Index”. This also indicates that there is an MVC controller called “EventsController” which has an action method of “Index”.

#### Element synopsis

*Element*: Applications – Optional, an element that contains a sequence of Application.

*Element*: Application – Contains a sequence of Operation

*Attribute*: Name – Required, name of a “store application” in NetSqlAzMan.

*Attribute*: Action – Optional, action to perform on this group. For allowable values see the XSD.

*Child Element*: Operation – Zero or more members of the group. String name of operation in NetSqlAzMan under this application.

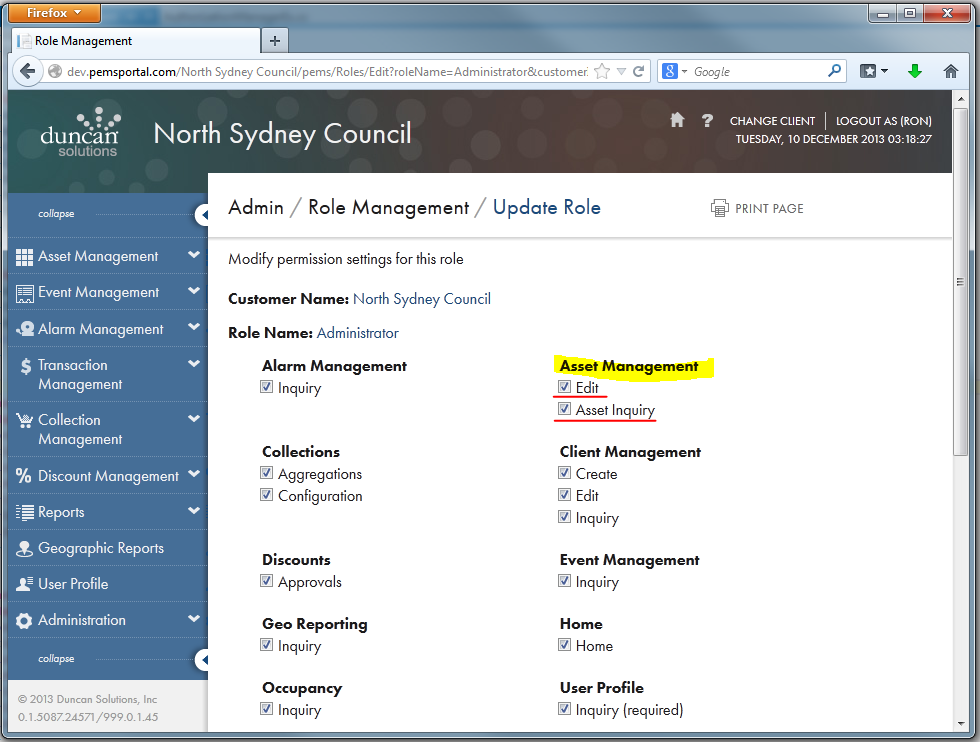
*Attribute*: Action – Optional, action to perform on this operation. For allowable values see the XSD.

### Authorizations

Authorizations are a sequence of AuthItem(s) that group a set of operations from an application under an authorization group. This authorization group is then assigned roles that are authorized to operate on this group of operations. Authorizations are grouped under associated applications, if there is authorizations of the application for the target customer.

The children AuthItem elements under the AuthTree represent the top-level authorization groups that are presented during the administration of roles during rights assignment. The children AuthItem(s) of the first level AuthItem represent groupings of functions that can have rights assigned or denied.

Here is an example screen.



Looking at the example at the highlighted items indicates that the following XML snippet would be generated. Note how the highlighted “authtext” attributes correspond to the highlighted page elements above.

<AuthTree>

…

<AuthItem Name="Assets" authtext="**Asset Management**">

<AuthItem Name="Edit" authtext="**Edit**">

<Operations>

<Operation>Index</Operation>

…

<Operation>ViewTariffConfiguration</Operation>

</Operations>

<Authorizations>

<Authorize>Administrator</Authorize>

<Authorize>Administrators</Authorize>

</Authorizations>

</AuthItem>

<AuthItem Name="Inquiry" authtext="**Asset Inquiry**">

<Operations>

<Operation>Index</Operation>

</Operations>

<Authorizations>

<Authorize>Administrator</Authorize>

<Authorize>Administrators</Authorize>

<Authorize>Support Staff</Authorize>

<Authorize>General Technician</Authorize>

<Authorize>Senior Technician</Authorize>

<Authorize>Collection Team</Authorize>

<Authorize>PTest</Authorize>

</Authorizations>

</AuthItem>

</AuthItem>

#### Element synopsis

*Element*: AuthTree – Optional, an element that contains a sequence of AuthItem elements.

*Element*: AuthItem – Contains a sequence of AuthItem elements representing groups of functionality that can have roles authorized or denied.

*Attribute*: Name – Required, name of a “store application” in NetSqlAzMan.

*Attribute*: authtext – Required, title of the overall functional context. Will be displayed to user as indicated in example above.

*Attribute*: Action – Optional, action to perform on this functional context. For allowable values see the XSD.

*Child Element*: AuthItem – Zero or more authorization functionality groups.

*Attribute*: Name – Required, unique name of item. This must be unique in NetSqlAzMan and must not match the name of any other AuthItem or MenuItem,

*Attribute*: authtext – Required, title of the functional groupt. Will be displayed to user as indicated in example above.

*Attribute*: Action – Optional, action to perform on this operation. For allowable values see the XSD.

*Element*: Operations – Required, sequence of Operation elements.

*Element*: Operation – Required, string, must match the name of an Operation under the associated Application for given “store” (customer).

*Attribute*: Action – Optional, action to perform on this operation. For allowable values see the XSD.

*Element*: Authorizations – Required, sequence of Authorization elements.

*Element*: Authorization – Required, string, must match a group name of the given “store” (customer).

*Attribute*: Action – Optional, action to perform on this operation. For allowable values see the XSD.

### Menus

Menus are presented to a user based on the menu structure defined for a customer and the present user’s rights to access resources that are represented by the menu structure. For example, if a customer has a possible menu selection ‘X’ but the user does not have rights to the resource linked to ‘X’, the menu selection will not be displayed to the user.

The MenuTree element is unique as a top-level element goes since it has an optional Action attribute of “Clear”. If this attribute is set then all existing menu entries for the customer are cleared before the rest of the menu XML is process. This is a powerful feature and should be used with caution.

A MenuTree has a sequence of MenuItem elements. This sequence represents the top-level menus displayed on the left nav menu. Each top-level MenuItem can have zero or more child MenuItem elements representing sub-menus.

Each MenuItem element can have a Link child element which defined the resource available when the user clicks the menu.

When creating a menu, elements are ordered in the system by the order in which they were read from the XML. Use caution when getting the present customer XML and using it to create a new menu. XML read from a customer has an attribute called “menuorder” that is used to indicate the order the menus were originally stored in NetSqlAzMan. When the menu data is read back out, NetSqlAzMan orders it randomly. When the menu data is stored, the order in which the menus are described in the XML is used to create these “menuorder” values. These values are used to order the left nav menu for display.

#### Element synopsis

*Element*: MenuTree – Optional, an element that contains a sequence of MenuItem elements that represent the top-level menu items in the left nav menu.

*Element*: MenuItem – Contains a sequence of MenuItem elements representing child menus under the to-level menu..

*Attribute*: Name – Required, name of a “store application” in NetSqlAzMan. Each top level menu can represent one “store application” and must match one of the application for the customer.

*Attribute*: menutext – Required, the menu text presented to the user.

*Attribute*: menuorder – Optional, used to show order of menu items. This is only valid when read from a customer. This setting will not have any effect on ordering when creating a menu.

*Attribute*: menuicon – Optional, name of a file that contains the icon used for the menu. This file is, by default, found at the root of the web site under /Images/Menu folder.

*Attribute*: tooltip – Optional, text used as a tool tip for the menu item.

*Attribute*: Action – Optional, action to perform on this functional context. For allowable values see the XSD.

*Child Element*: Link – Optional – Defines the resource link for the top-level menu item.

*Attribute*: operation – Optional, name of an operation that, in conjunction with application” will be used to create a hyper-link to a page in PEMS.

*Attribute*: application – Optional, name of an application that, in conjunction with “operation”, will be used to create a hyper-link to a page in PEMS.

*Attribute*: url – Optional, URL link to a resource presented to the user when they click on the menu item.

*Attribute*: newwindow – Optional, In conjunction with “url”, used in hyper-link as the target.

*Attribute*: Action – Optional, action to perform on this operation. For allowable values see the XSD.

*Elements*: MenuItem – Contains a sequence of MenuItem elements representing child menus under the to-level menu..

*Attribute*: Name – Required, name of a “store application” in NetSqlAzMan. Each top level menu can represent one “store application” and must match one of the application for the customer.

*Attribute*: menutext – Required, the menu text presented to the user.

*Attribute*: menuorder – Optional, used to show order of menu items. This is only valid when read from a customer. This setting will not have any effect on ordering when creating a menu.

*Attribute*: menuicon – Optional, name of a file that contains the icon used for the menu. This file is, by default, found at the root of the web site under /Images/Menu folder.

*Attribute*: tooltip – Optional, text used as a tool tip for the menu item.

*Attribute*: Action – Optional, action to perform on this functional context. For allowable values see the XSD.

*Child Element*: Link – Optional – Defines the resource link for the top-level menu item.

*Attribute*: operation – Optional, name of an operation that, in conjunction with application” will be used to create a hyper-link to a page in PEMS.

*Attribute*: application – Optional, name of an application that, in conjunction with “operation”, will be used to create a hyper-link to a page in PEMS.

*Attribute*: url – Optional, URL link to a resource presented to the user when they click on the menu item.

*Attribute*: newwindow – Optional, In conjunction with “url”, used in hyper-link as the target.

*Attribute*: Action – Optional, action to perform on this operation. For allowable values see the XSD.

## XSD

[TBD]

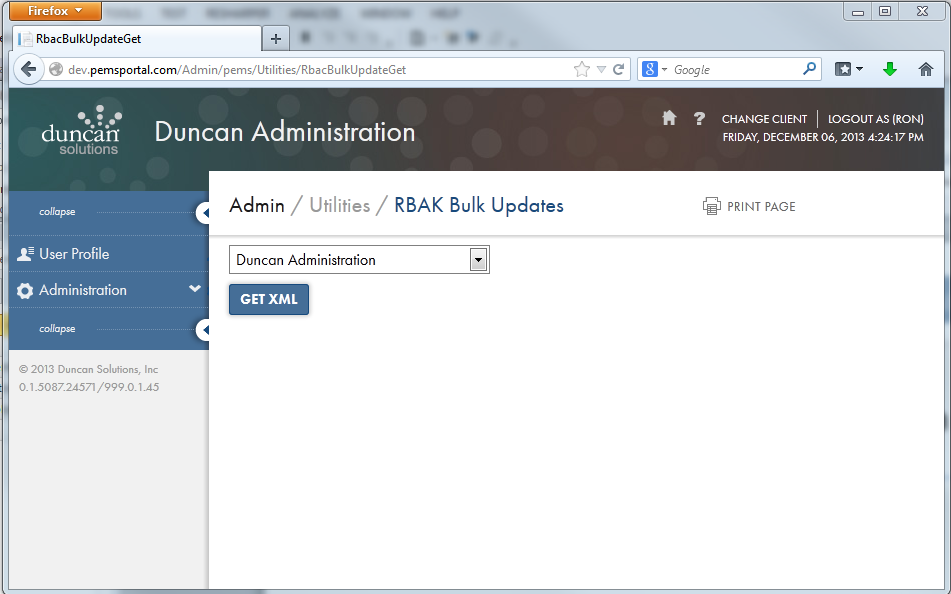
# PEMS Administration Interface

## Getting Current XML

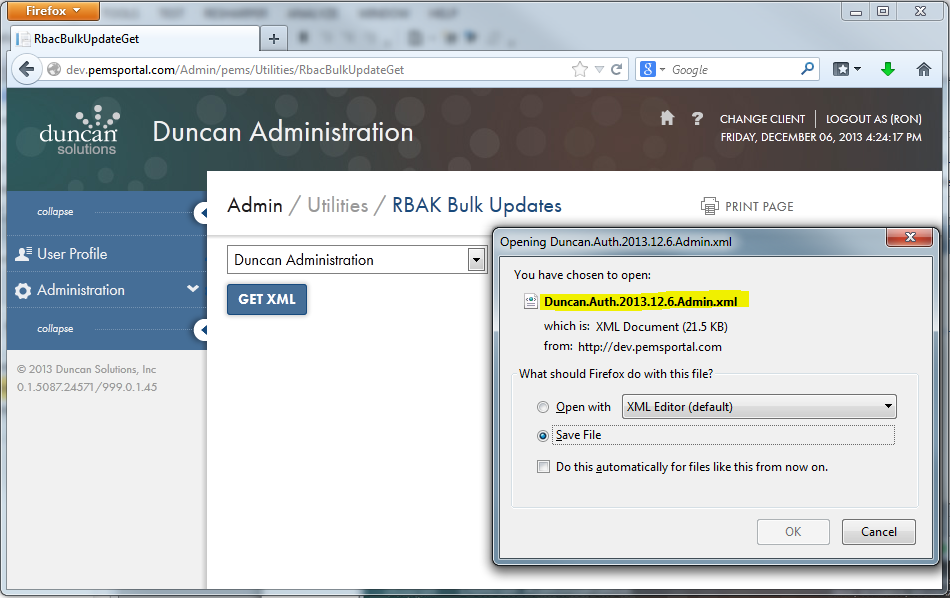
RBAC Authorization XML describing the current settings of the Users, Applications, Groups, AuthTree and MenuTree can be retrieved for any customer by using PEMS Admin page **RbacBulkUpdateGet**. This page is not available as a normal menu link, it is only available by direct address entry for a given PEMS hosting. The page is available at:

***[host]/Admin/pems/Utilities/RbacBulkUpdateGet***

For example, navigating to ***http://dev.pemsportal.com/Admin/pems/Utilities/RbacBulkUpdateGet*** will bring up the following page. Using the drop-down, select the desired customer and click “GET XML” to retrieve the current authorization and menu RBAC Authorization XML.



By default the XML will be named “Duncan.Auth.YYYY.MM.DD.[customer].xml. The XML will contain all five sections as described above. The Customer element’s Name attribute will be set appropriately for the customer you selected.



Save this file to a location on your computer. Make a copy of this file as a backup in case there are errors during the update process.

## Setting New XML

Submitting a new RBAC Authorization XML describing the new settings of the Applications, Groups, AuthTree and MenuTree can be achieved by using PEMS Admin page **RbacBulkUpdateSet**. This page is not available as a normal menu link, it is only available by direct address entry for a given PEMS hosting. The page is available at:

***[host]/Admin/pems/Utilities/RbacBulkUpdateSet***

For example, navigating to ***http://dev.pemsportal.com/Admin/pems/Utilities/RbacBulkUpdateSet*** will bring up the following page. Using the “Select…” button, navigate to the new RBAC Authorization XML file to be uploaded to PEMS.

Note that the RBAC Authorization XML will be targeted at the customer named in the Customer element’s attribute called Name. If this name is not correct, nothing will be processed.

Once you have selected the XML file, click “SUBMIT XML FILE”. The file will be uploaded and the authorization and menu settings will be processed.

After a short period of time, you will see lines displayed in the Results section under Log and possibly Errors if errors were encountered. The Log section will display the name of the file processed (as it appears on the web server) as well as notes on what was changed, added or deleted. Use this log to verify your work.

If any errors were encountered, you will see lines displayed under the Errors section. These errors attempt to guide you as to what has happened. Most often it is the case where an attribute name was given for something that should already have existed in NetSqlAzMan but was misspelled. This document does not attempt to describe all of the possible errors. If you encounter errors, use the error text to guide you to associated section/entry in your XML.

Errors can be fixed and XML resubmitted as the process is idempotent.

