Srbac v 1.1.1 (r209)

1 What is srbac?

Srbac is a module designed for the Yii framework http://www.yiiframework.com/ .
It is designed to make easier the use of Yii authManager components that implements the use of Role Based Access Control (R.B.A.C.).
The authManager that srbac supports is the CdbAuthManger which uses a database to store the authorization data.
Srbac offers a graphical interface for the most of RBAC actions (create / edit / delete authorization items, assigning authorization items to users etc.)

Srbac 1.1.x needs Yii version 1.1.0 or higher.

2 Downloading srbac

Srbac can be downloaded from :
The Yii Extensions page http://www.yiiframework.com/extension/srbac/
Google projects page : http://code.google.com/p/srbac/downloads/list

Also latest developement code can be checked out using this command

```
svn checkout http://srbac.googlecode.com/svn/trunk/ srbac-read-only
```

3 Installing srbac

To install srbac module first extract the zip file to the modules directory of your Yii application.
Then edit your configuration file according to the following:

Configure the database component:

For SQLite :

```
'db'=>array(
  'class'=>'CDbConnection',
  'connectionString'=>'sqlite:path/to/database/yourDatabase.db',
),
```

For MySQL :

```
'db'=>array(
  'class'=>'CDbConnection',
  'connectionString'=>'mysql:host=localhost;dbname=yourDatabase',
  'username'=>'yourUsername',
  'password'=>'yourPassword',
),
```

Configure AuthManager component

```
'authManager'=>array(
  // Path to SDbAuthManager in srbac module if you want to use case insensitive
   //access checking (or CDbAuthManager for case sensitive access checking)
  'class'=>'modules.srbac.components.SDbAuthManager',
  // The database component used
  'connectionID'=>'db',
  // The itemTable name (default:authitem)
  'itemTable'=>'items',
  // The assignmentTable name (default:authassignment)
  'assignmentTable'=>'assignments',
  // The itemChildTable name (default:authitemchild)
  'itemChildTable'=>'itemchildren',
),
```

Configure srbac module:

```
'srbac' => array(
  'userclass'=>'User', //default: User
  'userid'=>'user_ID', //default: userid
  'username'=>'username', //default:username
  'debug'=>true, //default :false
  'pageSize'=>10, // default : 15
  'superUser' =>'Authority', //default: Authorizer
  'css'=>'srbac.css',  //default: srbac.css
  'layout'=>
    'application.views.layouts.main', //default: application.views.layouts.main,
                                      //must be an existing alias
```

```
     'notAuthorizedView'=> 'srbac.views.authitem.unauthorized', // default:
                       //srbac.views.authitem.unauthorized, must be an existing alias
     'alwaysAllowed'=>array(   //default: array()
        'SiteLogin','SiteLogout','SiteIndex','SiteAdmin',
        'SiteError', 'SiteContact'),
     'userActions'=>array('Show','View','List'), //default: array()
     'listBoxNumberOfLines' => 15,  //default : 10
     'imagesPath' => 'srbac.images', // default: srbac.images
     'imagesPack'=>'noia', //default: noia
     'iconText'=>true, // default : false
     'header'=>'srbac.views.authitem.header', //default : srbac.views.authitem.header,
                                              //must be an existing alias
     'footer'=>'srbac.views.authitem.footer', //default: srbac.views.authitem.footer,
                                              //must be an existing alias
     'showHeader'=>true, // default: false
     'showFooter'=>true, // default: false
     'alwaysAllowedPath'=>'srbac.components', // default: srbac.components
                                              // must be an existing alias
   )
```

Check srbac attributes for detailed information about every attribute.

Import SbaseController (for using the auto checking access feature):

```
  'import'=>array(
    'application.modules.srbac.controllers.SBaseController',
  ),
```

Then point your browser to **/path/to/application/index.php?r=srbac** and you will be redirected to the installation page.
A check is performed and if everything is OK you can proceed to the instalation (There is also a choice to create some demo authorization items).
If srbac is already installed you will be prompt to overwrite the previous installation (**That will drop all tables and delete your currently authorization data**).
An 'Authorizer' role will be created.(You can change the name of the role through srbac configuration). This is the only user that can admin srbac (create, edit, delete roles, tasks, operations and assign them to users).
Notice that until you set srbac debug attribute to false anyone can admin srbac, and
also anyone can admin srbac until you assign the Authorizer role to at least one user.
After assigning the Authorizer role to a user is wise to set srbac debug attribute to false. Also you may remove or rename srbac/views/authitem/install folder.
The srbac main administrator page then is :
path/to/application/index.php?r=srbac/authitem/frontpage.

4 Autocreation and access checking

From version 1.02 and on you can automatically create operations/tasks for your controllers.
The operations are named as
[ModuleId]/[Subdirectory].[ContollerId][Action] :
posts_PostView : module posts , controller Post, action View
posts_admin.PostDelete : module posts, subdirectory admin, controller Post, action delete
SiteIndex : controller Site , action Index.

Also you can create 2 tasks named [ControllerId]Viewing, [ControllerId]Administrating (eg PostViewing, PostAdministrating).
All operations are assigned to the administrating task, and you can select which operations are assigned to viewing task by editing the userActions attribute in srbac configuration.

If you also want srbac to automatically check for access in your controllers, your controllers should extend the SBaseController class in srbac module or any other class that extends this one.

SbaseController overides the beforeAction($action) method and checks if the user has access to the current controller/action.
To decide what should be done when the user has not access you may override the onUnauthorizedAccess method in SBaseController. This way you can display a not Authorized Access message, redirect to the login page etc.

5 Internationalization

If you want to translate srbac texts you should set the target language in your Yii configuration file:
'language'=>'fr',
then create the following files that contain the translations :
srbac/messages/**fr/srbac.php** (you can copy el_gr/srbac.php and translate the messages).
srbac/views/install/**fr/installText.php** (you should translate srbac/views/install/installText.php)

If you want to help with translating srbac contact me through Yii forum :
http://www.yiiframework.com/forum/index.php?/user/1089-spyros/

6 Srbac attributes

$userid	**String**	The user's id attribute. Defaults to **“userid”**.
$username	**String**	The user's name attribute. Defaults to **“username”**.
$userclass	**String**	The users class. Defaults to **“User”**. (You have to create a user class in your applications – it’s not created by srbac.)
$debug	**Boolean**	If srbac is in debug mode. While in debug mode srbac can be installed, every user can administrate srbac and missing translations will be marked with a red star ***.** Defaults to **false. Remove it or set it to false while production.**
$pagesize	**integer**	The number of auth items displayed in each page of the auth items list. Defaults to **15**.
$superuser	**String**	The name of the srbac administrator role. Defaults to **“Authorizer”**.
$css	**String**	The css file to use. Srbac will first look for the file in default applications css directory (webroot.css) and then in srbac css directory (application.modules.srbac.css). Defaults to **”srbac.css”**.
$layout	**String**	The layout to use when rendering srbac views. Defaults to empty string **“”** meaning the applications main layout.
$notAuthorizedView	**String**	The view to display to users that are accessing a page without authorization. Defaults to **"srbac.views.authitem.unauthorized"**.
$alwaysAllowed	**mixed**	The actions that are always allowed even to guests (e.g.SiteIndex, SiteLogin, SiteLogout etc) .Can be an array an alias to a file that returns an array or a coma delimeted string.This array will be merged with rbac/components/allowed.php file generated by the always allowed GUI. Defaults to **array()**.
$userActions	**mixed**	The operations that are assigned to viewing taskby default (e.g. Show, List, View etc). Can be an array or a coma delimeted string .Defaults to **array()**.
$listBoxNumberOfLines	**integer**	The number of lines in the assign view listboxes. Defaults to **10**.
$imagesPath	**String**	Alias to the srbac images directory. Defaults to **“application.modules.srbac.images”**.
$imagesPack	**String**	The images pack to use. Current packs that are available are **“tango”** and **“noia”.** If you want to use your pack you should define $imagesPath and put there your pack (eg “myImagesPath/myPack”). *
$iconText	**Boolean**	Whether to show icon text next to the icons. Defaults to **false**.
$header	**String**	The view to render above srbac GUI. Defaults to **"srbac.views.authitem.header"**
$footer	**String**	The view to render below srbac GUI. Defaults to **"srbac.views.authitem.footer"**
$showHeader	**Boolean**	Whether to show header or not. Defaults to **false**
$showFooter	**Boolean**	Whether to show footer or not. Defaults to **false**
$alwaysAllowedPath	**String**	The path where allowed.php is saved. Defaults to **“srbac.components”**

** The images pack includes the following icons:*

- admin.png – 16x16 pixels : Administration of auth items
- allowed.png – 16x16 pixels : The edit alwaysAllowed icon.
- create.png - 16x16 pixels : Create new auth item
- delete.png - 16x16 pixels : Delete auth items / Delete auto created auth items
- eraser.png - 16x16 pixels : Scan for authItems that no longer belong to a controller
- manageAuth.png – 32x32 pixels :Managing authItems
- preview.png – 16x16 pixels : filter auth items button.

- update.png – 16x16 pixels : Edit an auth item.
- users.png – 32x32 pixels : Show users assignments.
- usersAssign.png – 32x32 pixels : Assign auth items to users
- wizard.png – 16x16 pixels : Auto create auth items.

7 Bug Reporting

Please report any bugs at the Yii Forum srbac thread :
Yii Forum – srbac Extension

Or open an issue at Google code issue tracker :
srbac google code issue tracker

8 Upgrading srbac version

When upgrading to Version 1.1.0.2 (r174) you should change in auth items the '_' character that identifies the modules to '/'.

9 Srbac history

Version 1.1.1 (r209)

Enhancements

- Added srbac.components.SDbAuthManager.php .You can set this as your AuthManager class in main.php configuration file if you want case insensitive access checking.

Bug Fixes

- Fixed bug when allowed.php did not return an array.

Updates

- Added Lithuanian translation
- Added Ukranian translation
- Updated Spanish and Russian translation

Version 1.1.0.2 (r174)

Enhancements

- Always allow access when srbac is in debug mode
- Auth items are auto created only for controllers that extend SBaseController

Bug Fixes

- Fixed IE bug when tab selection was not allowed in assign page
- Fixed bug when layout is set to 'main'
- Change the character that indentifies a module in authItems from _ to /

Version 1.1.0 (r174)

Enhancements

- Added alwaysAllowedPath attribute.

- Added bredcrumbs (hard coded)

Bug Fixes

- Fix alwaysAllowed bug when using srbac as submodule (fixes issue 38)
- Fix AlwaysAllowedFile creation
- Fixed css publishing
- Fixed css for tabview

Updates

- Updated Spanish translation
- Added Portuguese translation

Version 1.1.0 RC (r145)

Enhancements

- Support for table prefix.
- Autocreate scans in subdirectories of controller paths.
- GUI for editing the alwaysAllowed list
- Added onUnauthorizedAccess protected method in SBaseController
- alwaysAllowed attribute can also accept an alias to a file that returns an array.
- alwaysAllowed and userActions can now be an array or a coma delimeted string.
- Add clever assigning of operations to tasks.
- Add configuration checking of attributes **userid, username, css, notAuthorizedView, layout, imagesPath, imagesPack** before installation.

Bug Fixes

- Replaced deprecated function in php 5.3 ereg_replace and split with preg_replace and explode.
- Fixed scanning of modules when url formating is used
- All list boxes in assign tabview are sorted alphabetically.
- Fixed bug when using different databases for the main application and the AuthManager.
- Fixed bugs related to version checking and translations.
- Fixed process-working.gif to be a transparent gif.
- Fixed controller action scanning.
- Fixed using different db connections for CDbAuthManager and main application causes srbac to detect wrong auth database.

Updates

- Update all files for Yii 1.1 compatibility.
- Updated Greek translation.
- imagesPath attribute now accepts alias instead of url.
- Updated Spanish translation.

Version 1.0.3

Enhancements

- Added imagesPack attribute.User can choose which pack of images to use for the srbac icons.
- Hovering over an authItem in authItems manager will show its description.
- If an action is in alwaysAllowed array will be ignored during the autocreation of authItems.
- Added srbac attribute imagesPath to set the path to the srbac icons (create, delete, admin etc).
- Custom srbac css can be placed not only in srbac/css directory but also in default application/css directory.
- Auto create checks if task exist and displays them or not in the create tasks list.
- Auto creating of modules controllers actions.

- listBoxNumberOfLines attributes controls the number of lines in assign tabview listboxes.

Bug Fixes

- Added a default unauthorized view in case the user has not defined one.
- Fixed an IE bug in autocreation view(Thanks to 'idle sign').
- Access is now always denied to guests, except if the current page is in alwaysAllowed array.
- Fixed php short tags in views/AuthItem/userAssignments.php.
- srbac should work now in linux os.
- Changed dataGrid class to srbacDataGrid so it won't conflict with Yii dataGrid class.

Updates

- Added new icons and a control center top bar (Thanks to 'idle sign').
- Added new css styles and new ajax-loader (thanks to 'idle sign').
- Added noia icons pack.
- Changed custom images path relative to webroot .
- Set default css to srbac.css.
- Changed allowedAccess() method in SBaseController to protected so it can be inherited.
- Updated spanish translation (Ricardo Obregon).

Version 1.0.2

Enhancements

- When auto creating/deleting operations a check is performed to see if the items already exist.
- Srbac attribute alwaysAllowed to define in which pages checking is not performed.
- Srbac attribute userActions to define which actions are assigned to using task by default.
- layout attribute accepts path alias now.
- Automatic creation of operations based on controllers' actions.
- Also Creation of two tasks (using, administrating).
- The operations are also assigned to these tasks based on the action's name (all operations assigned to administrating and list, view, show actions to using).
- Mass delete of automatic created operations, tasks.
- Cannot revoke Authority role if there's no other user with that role.
- Custom not authorized page.
- srbac front page.
- srbac isInstalled() method.
- Added an SBaseController that must be extended for the use of automatic created auth items.

Bug Fixes

- Removed safe rule from authItem model.
- Changed view path to authitem instead of authItem due to case sensitivity errors in some operation systems.
- Import SBaseController in modules init method.
- Udefined variable in assignments.php (Thanks to idle sign).
- ShowAssignments ajax call (Thanks to idle sign).

Updates

- Russian translation updated to v1.2 (Thanks to idle sign)

Version 1.0.1

Enhancements

- Custom layout for srbac (layout must be in default application's layout folder).
- Add ajax indicators.

- Can call user assignments directly from your user's controller by
  ```
  $this->renderPartial(
  'application.modules.srbac.views.authItem.assignments',array("id"->$id));
  ```
  ($id is the user's id)
- Spanish translation (Ricardo Obregon)
- Russian translation (idle sign)

Version 1.0

Enhancements

- Srbac can be installed as a child module too.
- View Roles / Tasks / Operations assigned to any user.
- Delete authItems.
- Ajax based web interface for administrating auth items.
- You can select if demo data will be created in install.
- All assignments calls are made in ajax.
- You can set the name of the Authorizer authItem.
- You can update the names of the auth Items.

Bug Fixes

- Fixed undefined variable "message"(thanks to sebi).
- Fixed possible SQL-injections flaw (thanks to Anticon).
- Fixed bug when when a not-Authorizer-user tries to access SRBAC (thanks to Anticon).
- Fixed bug with deleting tables in wrong order (thanks to rabol).
- Fixed bug with wrong column name resulting in errors while working with innoDB engine(thanks to sebi).
- Fixed udefined variable errors.
- Fixed showing all items in the assigned and in the not assigned panels when accessing the assign page for the first time.
- Fixed errors when pressing '<<' or '>>' with a wrong selection of items.
- Wrong tab display after an assignment.

10 Licence

Copyright © 2009 – 2010 by Spyros
All rights reserved.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
* Neither the name of Spyros nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.