ACL

The Access Control List (ACL) is made available to plugins so access may be restricted and configured under Staff Groups.

> **The plugin ACL system has been replaced since Blesta version 4.10.0**
>
> The method of adding permissions has been replaced as of version v4.10.0. We strongly recommend to no longer use the older method in any of your custom code. The following describes the new plugin ACL system.

Adding to the ACL

To add your plugin to the ACL you must first decide under which permission group your access permissions will go. There are existing permission groups, but you may wish to create your own. Group permissions are now defined inside the **getGroupPermissions()** method.

/plugins/my_plugin/my_plugin_plugin.php

```
<?php
class MyPluginPlugin extends Plugin {

    ...

    public function getPermissionGroups()
    {
        return [
            [
                'name' => 'Permission Group Name',
                'level' => 'staff',
                'alias' => 'my_plugin'
            ]
        ];
    }
}
?>
```

Once you have a permission group to reference, define your permissions inside the **getPermissions()** method with the appropriate parameters to create your access permission, setting 'alias' as the plugin.controller (e.g. MyPlugin plugin FooBar controller becomes my_plugin.foo_bar) and 'action' as the method to control (use * for all methods in a controller).

/plugins/my_plugin/my_plugin_plugin.php

```
<?php
class MyPluginPlugin extends Plugin {
        ...

        public function getPermissions()
        {
                return [
            [
                'group_alias' => 'my_plugin', // Alias of the permission group
                'name' => 'Some Action',
                'alias' => 'my_plugin.foo_bar',
                'action' => '*'
            ]
                ];
        }
}
?>
```

Enforcing the ACL

Every controller that inherits from AppController (either directly or indirectly) can enforce the ACL rules on the requested resource simply by invoking the **requireLogin()** method.

/plugins/my_plugin/controllers/client_main.php

```
<?php
class ClientMain extends MyPluginController {
    public function preAction() {
        parent::preAction();

        // Login required
                $this->requireLogin();
        }

        public function index() {
                // Automatically protected by the ACL
        }
}
?>
```

Adding to the ACL prior to Blesta version 4.10.0

> **The plugin ACL system has been replaced since Blesta version 4.10.0**
>
> The method of adding permissions described below has been replaced as of version v4.10.0. We strongly recommend to no longer use this method in any of your custom code.

To add your plugin to the ACL you must first decide under which permission group your access permissions will go. There are existing permission groups, but you may wish to create your own. To do so, invoke Permissions::addGroup().

/plugins/my_plugin/my_plugin_plugin.php

```
<?php
class MyPluginPlugin extends Plugin {

    ...

    public function install($plugin_id) {
        Loader::loadModels($this, array("Permissions"));

                // Add a new permission group
        $group = array('plugin_id' => $plugin_id, 'name' => "Permission Group Name", 'level' => "staff",
'alias' => "my_plugin");
        $group_id = $this->Permissions->addGroup($group);
    }
}
?>
```

Once you have a permission group to reference, invoke Permissions::add() with the appropriate parameters to create your access permission, setting 'alias' as the **plugin.controller** (e.g. MyPlugin plugin FooBar controller becomes my_plugin.foo_bar) and 'action' as the method to control (use * for all methods in a controller).

/plugins/my_plugin/my_plugin_plugin.php

```
<?php
class MyPluginPlugin extends Plugin {

    ...

    public function install($plugin_id) {
        Loader::loadModels($this, array("Permissions"));

                // Add a new permission group
        $group = array('name' => "Permission Group Name", 'level' => "staff", 'alias' => "my_plugin");
        $group_id = $this->Permissions->addGroup($group);

                // Add a new permission to the group for the FooBar controller of this plugin
        $perm = array('group_id' => $group_id, 'plugin_id' => $plugin_id, 'name' => "Some Action", 'alias' =>
"my_plugin.foo_bar", 'action' => "someMethod");
        $this->Permissions->add($perm);
    }
}
?>
```

Clean up after your plugin

Be sure to remove any permissions or permission groups you've added when your plugin is uninstalled by using the Permissions::deleteGroup() and Permissions::delete() methods when your plugin's uninstall() method is called. Use the Permissions::getGroupByAlias() and Permissions::getByAlias() methods to look up any permission groups or permissions your plugin created.

Once you've added all of your access permissions there is nothing more you need to do. Your plugin will now only be made available according to the access permissions you've defined and have been configured.

Is my plugin required to use the ACL?

Your plugin is not required to use the ACL, but it's a good idea. Using the ACL allows users who install your plugin finer grained control over where your plugin can appear and who can use it.