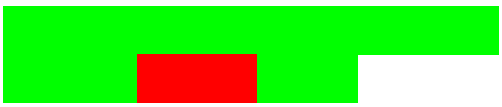

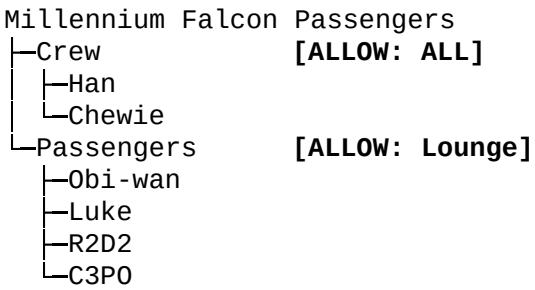
Millennium Falcon Passengers
Crew
[ALLOW: ALL]
Han
Chewie
Passengers
[ALLOW: Lounge]
Obi-wan
Luke
R2D2
C3PO

Move on to "Passengers", which explicitly says that "Passengers" have Lounge access, so change the internal result to "ALLOW".

Move to the "Jedi" node, which doesn' t mention the Lounge at all.

Finally move to Luke' s node, and again there' s nothing there about the Lounge.

There' s nowhere left to go, so the result returned is the current value of the internal result: "ALLOW"

Example 2: We ask: "Does Chewie have access to the Engines?"

Set the default result, "DENY".

Work out a path to Chewie:

Millennium Falcon Passengers → Crew → Chewie

Start at the top of the tree and move towards Chewie. The "Millennium Falcon Passengers" node doesn' t say anything about anywhere, so do nothing here.

Move on to "Crew", which explicitly says that "Crew" have Engine access, so change the internal result to "ALLOW".

This shows how easy it is to grant new people access. If we used the original matrix scheme, we' d have to set permissions for each room for both Lando and Hontook. Instead, we simply add them to their appropriate groups and their access is implicitly and easily defined.

Resolving conflicts

What happens if we add Chewie to the list of Engineers?

```
Default: DENY ALL
Millennium Falcon Passengers
```

Keep in mind AXO' s are optional, if you don' t specify an AXO when calling acl_check() and a mnta chg acADP exists with noXO wif t will be linowed. Howeverf yo

4.

Advanced setup

Reusing an already existing ADOdb installation

If you already have ADOdb installed you can get phpGACL to use this copy of ADOdb.

1. Edit phpgacl/gacl.class.php so that ADODB_DIR reflects the location of the ADOdb library in your path.

2. Rename the phpgacl/adodb folder to something else like adodb_x and reload the phpgacl/admin/acl_admin.php page to ensure it still works.

3. Erase the adodb directory installed with phpGACL.

Reusing an already existing Smarty installation

If you already have ADOdb installed you can get phpGACL to use this copy of ADOdb.

1. Edit phpgacl/admin/gacl_admin.inc.php so that the variables $smarty_dir and $smarty_compile_dir reflect the location of the Smarty library in your path and the template_c directory you already use.

   Move the templates directory that came with phpGACL to another directory (e.g. one level up). Adjust the $smarty_template_dir so it points to the new location. If you like you can move those templates to your existing templates folder, of course.

2. Rename the phpgacl/smarty folder to something else like smarty_x and reload the phpgacl/admin/acl_admin.php page to ensure it still works.

3. Erase the smarty directory installed with phpGACL.

How do I move the phpGACL files out of my website tree while leaving a link in the tree for administration?

1. Go to your website root.

2. Move the phpGACL directory to your includes directory and create a symlink to the admin directory where you want the admin tool to go. For example:

   mv phpgacl/ /www/includes_directory
   ln -s /www/includes_directory/phpgacl/admin/ gacl

3. Now surfing to http://yoursite.net/gacl/acl_admin.php will take you to the admin page. If it doesn' t work, make sure your Webserver allows symbolic links in the website tree.

Using phpGACL in your application

Basic usage

This example shows a basic example of using phpGACL in your code. It uses the ADOdb

U

API

edit_group()

Edits a group.

edit_group (

int GROUP_ID,

string NAME,

int GROUP_PARENT_ID,

string GROUP_TYPE) The Access Object type ("aco", "aro" or "axo").

Returns:

int TRUE on success, FALSE on failure.

del_group()

Deletes a group, re-parenting or deleting children if specified.

del_group (

int GROUP_ID,

bool REPARENT_CHILDREN,

string GROUP_TYPE) The Access Object type ("aco", "aro" or "axo").

Returns:

int TRUE on success, FALSE on failure.

Access Objects (ARO/ACO/AXO)

This section of the API manages Access Objects like AROs, ACOs and AXOs.

get_object()

add_object (

string SECTION_VALUE,

string NAME,

string VALUE,

int ORDER,

bool HIDDEN,

string GROUP_TYPE) The Access Object type ("aco", "aro" or "axo").

Returns:

array OBJECT_ID on success, FALSE on failure.

edit_object()

Edits an object.

edit_object (

string SECTION_VALUE,

string NAME,

string VALUE,

int ORDER,

bool HIDDEN,

string GROUP_TYPE) The Access Object type ("aco", "aro" or "axo").

Returns:

array OBJECT_ID on success, FALSE on failure.

del_object()

Deletes an object.

del_object (

int OBJECT_ID,

string GROUP_TYPE, The Access Object type ("aco", "aro" or "axo").

bool ERASE)

Returns:

int TRUE on success, FALSE on failure.

Access Object Sections

This part of the API manages the Sections that comprise part of the unique name of an Access Object. See "Sections" for more information.

get_object_section_section_id()

Returns the ID of an existing Section. You must specify either the Section' s name, value, or both.

If only the name is specified, there may be multiple results (since the NAME does not have to be unique). In this case, an error is returned.

get_object_section_section_id (

string NAME, A short description of what this Section is for. (e.g. "Levels in building").

string VALUE, The name of the Section (e.g. "Floor").

string GROUP_TYPE) The Access Object type ("aco", "aro" or "axo").

Returns:

8 TmoegECTION_ of success, FALSE on failure.

FAQ