

Documentation for Jetspeed-2 PSML Authorization

by David Sean Taylor

Table of contents

1	2
2	2
3	4
4	5
5	5
6	6
7	6

1.

- Security Constraint
- Declarative and Global Constraints
- Folder Constraints
- Page Constraints
- Spring Configuration
- Subsite Security

Security Constraints are applied to pages and folders. Security Constraints either grant or deny access to pages and folders. Constraints can be defined in one of three places:

1. Global: As declarations in the page.security file found in the root of the PSML tree.
2. Folder: In the folder.metadata file optionally located in every directory
3. Page: In PSML files to constraint access to specific pages

Grants are associated with permissions, authorizing, or granting, a principal list access to a page or folder. A granting security constraint is the association of a list of one or more security principals combined with one or permissions. Grant constraints grant access to a page or folder for the associated list of permissions.

A deny security constraint is declared with one or more security principals; with no associated permissions. Deny constraints prohibit access to the page or folder for the given list of principals. Note that deny constraints must be listed before grant constraints.

When working with pages and folder resource constraints, a constraint can be either a declarative constraint or a referential constraint. Declarative constraints are declared and put to use right in place for the particular page or folder resource. Where as referential constraints refer to a constraint declared in a centralized security constraint resource: the page.security file. Each site or subsite can have one page.security resource for declaring constraints to be referenced in any page or folder.

2.

A security constraint is an XML element found in a PSML file, a folder metadata file, or in the global security declarations. A security constraint has one attribute: the name. A security constraint has the following elements:

- roles - a comma-separated list of one or more role principals or * for all roles
- groups - a comma-separated list of one or more group principals or * for all groups
- users - a comma-separated list of one or more user principals or * for all users
- owner - a single user principal
- permissions - a comma-separated list of one or permissions (view,edit,help)

The first four elements (roles, groups, users, owner) all define the principals who will either have a permission granted or denied.

Permissions are the portal modes that are granted by the security constraint. Note that permissions are only granted, not denied. The view permission is similar to the read permission found in operating systems. The edit permission is similar to the write permission found in operating systems. The help permission is similar to the info permission found in some portals.

Constraints can be granted to one or more role principals for a set of permissions on a given resource. Roles are derived from the authorized users list of role principals, i.e. the roles that the user is a member of. If the authorized user is a member of any of the listed roles, the permission to the resource will be granted.

```
<security-constraint>
  <roles>adminstrator, manager</roles>
  <permissions>view, edit</permissions>
</security-constraint>
```

Constraints can also deny role principals access to the entire resource. If the authorized user is a member of any of the listed roles, all access to the resource is denied.

```
<security-constraint>
  <roles>adminstrator, manager</roles>
</security-constraint>
```

Constraints can be granted to one or more group principals for a set of permissions on a given resource. Groups are derived from the authorized users list of group principals, i.e. the groups that the user is a member of. If the authorized user is a member of any of the listed groups, the permission to the resource will be granted.

```
<security-constraint>
  <groups>accounting, development</groups>
  <permissions>view</permissions>
</security-constraint>
```

Constraints can also deny group principals access to the entire resource. If the authorized user is a member of any of the listed groups, all access to the resource is denied.

```
<security-constraint>
  <groups>accounting, development</groups>
</security-constraint>
```

Constraints can be granted to one or more user principals for a set of permissions on a given resource: The current user must be one of the listed principals in the comma-separated list in order to grant permission to the resource.

```
<security-constraint>
```

```
<users>joeey, deedee, johnny</users>
<permissions>view, edit, help</permissions>
</security-constraint>
```

Constraints can also deny user principals access to the entire resource. If the authorized user is in the list, all access to the resource is denied.

```
<security-constraint>
  <users>fred</users>
</security-constraint>
```

Note that you can grant or deny permissions to a collection of one or more principal types. For example, here we grant view and edit permissions to the roles (manager, developer), and to the groups (QA and Research), and to the particular user (dilbert): If the authorized user is a member of any of the listed roles, groups, or users, the permission to the resource will be granted.

```
<security-constraint>
  <roles>hacker, coder, guru</roles>
  <groups>unix, linux, freebsd</groups>
  <users>betty, fred, barney, wilma</users>
  <permissions>view, edit</permissions>
</security-constraint>
```

Constraints can also deny combinations of principals access to the entire resource. If the authorized user is a member of any of the listed groups, roles or users, all access to the resource is denied.

```
<security-constraint>
  <roles>hacker, coder, guru</roles>
  <groups>unix, linux, freebsd</groups>
  <users>betty, fred, barney, wilma</users>
</security-constraint>
```

The * can be applied to roles, groups, users or permissions to imply ALL.

```
<security-constraint>
  <users>*</users>
  <permissions>*</permissions>
</security-constraint>
```

TODO

3.

Declarative constraints are declared in the page.security file of the root of a site or subsite. Declarative constraints are referenced in pages and folders with the security-constraints-ref tag. Global constraints are also declarative constraints. They are also defined and found in the page.security file in the root PSML repository. The difference with global constraints is ...

Note that there can be more than one page.security files in a subsite Jetspeed installation.

```
<security-constraints-def name="admin">
  <security-constraint>
    <roles>admin</roles>
    <permissions>view, edit</permissions>
  </security-constraint>
</security-constraints-def>
<global-security-constraints-ref>admin</global-security-constraints-ref>
```

Several security constraint declarations are made in the default deployment of Jetspeed:

name	grants	permissions	global
admin	roles: admin	view, edit	yes
manager	roles: manager	view	no
users	roles: user, manager	view	no
public-view	users: *	view	no
public-edit	users: *	view, edit	no

4.

Folder Security constraints are placed in a security-constraints list in the folder.metadata file optionally found in each folder in the site. Note that the absence of a folder.metadata or security constraints within that file means that the folder will inherit the constraints of the parent folder, all the way up to the root folder of the site or subsite. Folder constraints do not inherit across subsites. Folder security constraints are made up of declarative security constraints and referential security constraints. Here is an example of both, the first being a referential constraint, the second a declarative constraint:

```
<security-constraints>
  <security-constraints-ref>public-view</security-constraints-ref>
  <security-constraint>
    <groups>engineering</groups>
    <permissions>view</permissions>
  </security-constraint>
</security-constraints>
```

Note that all security constraints must be placed within a security-constraints collection.

5.

Page Security constraints are placed security-constraints list in PSML files and are optional. Note that the absence of a security constraints list within that file means that the folder will

inherit the constraints of the folder in which it resides. Page security constraints are made up of declarative security constraints and referential security constraints. Here is an example of both, the first being a referential constraint, the second a declarative constraint:

```
<security-constraints>
  <security-constraints-ref>global-view</security-constraints-ref>
  <security-constraint>
    <groups>accounting</groups>
    <permissions>view, edit</permissions>
  </security-constraint>
</security-constraints>
```

Note that all security constraints must be placed within a security-constraints collection.

6.

TODO: page-manager.xml enabling

7.

subsite security