Access Control Specification

Table of contents

1 Access Control Specification....2
2 Terminology....2
3 We use the following names....2
4 Concept....2
5 Role resolving algorithm....2
6 User interface....2
7 Examples....2

1. Access Control Specification

- Possibility to deny permissions

2. Terminology

- An accreditable is either a user, a group, an IP range, or the world.
- A credential is an assignment of a role to an accreditable.

3. We use the following names.

- n - a node
- parent(n) - the parent node of n
- n.c1, ..., n.ci - the credentials of the node n
- acc(c) - the accreditable which the credential refers to
- role(c) - the role which the credential assignes to the accreditable
- method(c) - the method of the credential

4. Concept

- Credential methods can be grant or deny.
- The order of credentials at a node is important.
- To find out if a certain accreditable a has a role r for the node n, use the following algorithm:
  - Search for the first matching credential, starting with the first credential of the node n.
  - If no credential of n matches the accreditable, continue with the parent node.
  - When a credential is found which assigns the role r to the accreditable a, return the method of the credential.
  - When a credential is found which assigns the role r to the accreditable a, return the method of the credential.

5. Role resolving algorithm

- while not matched:
  - for c : n.ci to n.c1: + if acc(c) = a and role(c) = r, return method(c)
  - n := parent(n)
- return deny

6. User interface

The user interface has to allow the following operations:

- add/remove credentials for users, groups, world
- set the credential method
- change credential order (move up/down)

7. Examples

Like stated above the order of the credential is important. Credentials are builded from policies. Imaginge you are trying to access http://localhost:8888/default/introduction.html

The defined policy (with highest priority) would be:
config/ac/policies/introduction.html/subtree-policy.acml. Imaginge you have defined:

```
<policy xmlns="http://apache.org/cocoon/lenya/ac/1.0">
  <world>
    <role id="visit" method="deny"/>
  </world>
  <group id="editor">
    <role id="edit" method="grant"/>
  </group>
</policy>
```

Then you try to login in with user "lenya" who is in the editor group. However you will not be successful, because everybody always is world. Since the DENY of world is coming first nobody will now be able to see the page. Changing above policy to

```
<policy xmlns="http://apache.org/cocoon/lenya/ac/1.0">
  <group id="editor">
    <role id="edit" method="grant"/>
  </group>
  <world>
    <role id="visit" method="deny"/>
  </world>
</policy>
```

Let all user of the group editor access the page.

Best practise is to deny access early in a node tree of policies for e.g. WORLD. Meaning to define it e.g. in config/ac/policies/authoring/subtree-policy.acml