WebAC Authorization Delegate

WebAC authorization Fedora module is an implementation of the still evolving draft by the W3C that proposes a decentralized authorization mechanism. See WebAccessControl specifications at the W3C website.

W3C's definition of WebAccessControl

WebAccessControl is a decentralized system for allowing different users and groups various forms of access to resources where users and groups are identified by HTTP URIs.

The WebAC module will enforce access control based on the Access Control List (ACL) RDF resource associated with the requested resource. In WebAC, an Access Control List (ACL) consists of a set of Authorizations. An Authorization is a single rule for access, such as "users alice and bob may write to resource foo", described with a set of RDF properties. Authorizations have the RDF type `http://www.w3.org/ns/auth/acl#Authorization` (for the remainder of this document, the `http://www.w3.org/ns/auth/acl#` namespace will be abbreviated with the prefix `acl:`).

The properties that may be used on an `acl:Authorization` are:

Property	Meaning
acl:accessTo	the URI of the protected resource
acl:agent	the user
acl:mode	the type of access (WebAC defines several modes: `acl:Read`, `acl:Write`, `acl:Append`, and `acl:Control`)
acl:accessToClass	an RDF class of protected resources *(N.B., not implemented in the first version of this module)*
acl:agentClass	an RDF class of users *(N.B., not implemented in the first version of this module)*

Examples of Authorizations

1. The user userA can Read document foo

```
@prefix acl: <http://www.w3.org/ns/auth/acl#>

<> a acl:Authorization ;
acl:accessTo </foo> ;
acl:mode acl:Read;
acl:agent </agents/userA> .
```

2. Users in NewsEditor group can Write to any resource of type News

```
@prefix acl: <http://www.w3.org/ns/auth/acl#>


<> a acl:Authorization ;
acl:accessToClass ex:News ;
acl:mode acl:Read, acl:Write;
acl:agentClass </agents/NewsEditor> .
```

Storing WebAC ACLs in Fedora 4

In Fedora 4, an ACL is a `ldp::BasicContainer` resource with the additional RDF type of `http://fedora.info/definitions/v4/webac#Acl`. This class is part of the Fedora WebAC ontology. Its children should each be resources of type `acl:Authorization`.

Protecting Resources

A resource specifies the location of its ACL using the `acl:accessControl` property. If a resource itself does not specify an ACL, its parent containers are inspected, and the first specified ACL found is used as the ACL for the requested resource. If no ACLs are found, the default policy is to deny access to the requested resource.

Steps in determining the effective authorization

Finding the ACL:

1. Get the ACL of the requested resource, if exists, else.
2. Get the ACL of the next ancestor recursively (using either ldp:contains or fedora:hasParent), if exists, else.
3. If no more ancestor exist (root node reached) and no ACL is found: **Deny access**.

Finding the effective authorization:

1. Find union of authorizations that specify access for the requesting **user**. This includes:
    a. authorizations that specify *accessTo* to the requested **resource**.
    b. authorizations that specify *accessToClass* of the requested **resource type**.
    c. If authorizations exist for user, go to step 6, else go to next step.
2. Find union of authorizations that specify access for the requesting **user's group**. This includes:
    a. authorizations that specify *accessTo* to the requested **resource**.
    b. authorizations that specify *accessToClass* of the requested **resource type**.
    c. If authorizations exist for group, go to step 6, else go to next step.
3. Find union of authorizations that specify access for the requesting **user**. This includes:
    a. authorizations that specify *accessTo*to the requested **resource's ancestor**.
    b. authorizations that specify *accessToClass* of to the requested **resource's ancestor type**.
    c. If authorizations exist for user, go to step 6, else go to next step.
4. Find union of authorizations that specify access for the requesting **user's group**. This includes:
    a. authorizations that specify *accessTo* to the requested **resource's ancestor**.
    b. authorizations that specify *accessToClass* of to the requested **resource's ancestor type**.
    c. If authorizations exist for group, go to step 6, else go to next step.
5. If no authorization exists for user or group: **Deny Access.**
6. Use the most permissive from the set of authorizations found.
    a. if the authorizations permit requested access mode: **Grant access.**
    b. if the authorizations do not permit requested access mode: **Deny access.**

Example Request Authorization Flow

Example Scenarios

1. I want to allow a user with username "smith123" to have **read, write** access to resource http://localhost:8080/rest/webacl_box1.

    Using the two "files" below to create our Authorization and ACL resources.

    Acl.ttl

    ```
    @prefix webac: <http://fedora.info/definitions/v4/webac#> .
    <> a webac:Acl .
    ```

    Authorization.ttl

    ```
    @prefix acl: <http://www.w3.org/ns/auth/acl#> .
    <> a acl:Authorization ;
       acl:agent "smith123" ;
       acl:mode acl:Read, acl:Write ;
       acl:accessTo <http://localhost:8080/rest/webacl_box1> .
    ```

    We would execute the following commands.

```
> curl -X POST -H "Content-type: text/turtle" --data-binary "@Acl.ttl" "http://localhost:8080/rest"

http://localhost:8080/rest/acl

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Authorization.ttl" "http://localhost:8080
/rest/acl/auth1"

http://localhost:8080/rest/acl/auth1

> echo "PREFIX acl: <http://www.w3.org/ns/auth/acl#>
INSERT INTO {
<> acl:accessControl <http://localhost:8080/rest/acl> .
}" | curl -X PATCH -H "Content-type: application/sparql-update" --upload-file - "http://localhost:8080
/rest/webacl_box1"
```

2. I want to let the group "Editors" have **read, write** access on all the items in the collection "http://localhost:8080/rest/box/bag/collection"

   Using the two "files" below to create our Authorization and ACL resources.

Acl.ttl

```
@prefix webac: <http://fedora.info/definitions/v4/webac#> .
<> a webac:Acl .
```

Authorization.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
<> a acl:Authorization ;
   acl:agent "Editors" ;
   acl:mode acl:Read, acl:Write ;
   acl:accessTo <http://localhost:8080/rest/box/bag/collection> .
```

   We would execute the following commands.

```
> curl -X POST -H "Content-type: text/turtle" --data-binary "@Acl.ttl" "http://localhost:8080/rest"

http://localhost:8080/rest/acl

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Authorization.ttl" "http://localhost:8080
/rest/acl/auth1"

http://localhost:8080/rest/acl/auth1

> echo "PREFIX acl: <http://www.w3.org/ns/auth/acl#>
INSERT INTO {
<> acl:accessControl <http://localhost:8080/rest/acl> .
}" | curl -X PATCH -H "Content-type: application/sparql-update" --upload-file - "http://localhost:8080
/rest/box/bag/collection"
```

3. I would like the collection http://localhost:8080/rest/dark/archive to be viewable only by the groupId "Restricted", but I would like to allow **anyone** to view the resource http://localhost:8080/rest/dark/archive/sunshine.

   Using the three "files" below to create our Authorization and ACL resources.

Acl.ttl

```
@prefix webac: <http://fedora.info/definitions/v4/webac#> .
<> a webac:Acl .
```

Auth_restricted.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
<> a acl:Authorization ;
   acl:agent "Restricted" ;
   acl:mode acl:Read ;
   acl:accessTo <http://localhost:8080/rest/dark/archive> .
```

Auth_open.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<> a acl:Authorization ;
   acl:agent foaf:Agent ;
   acl:mode acl:Read ;
   acl:accessTo <http://localhost:8080/rest/dark/archive/sunshine> .
```

The I would execute the following commands.

```
> curl -X POST -H "Content-type: text/turtle" --data-binary "@Acl.ttl" "http://localhost:8080/rest"

http://localhost:8080/rest/acl_lock

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Auth_restricted.ttl" "http://localhost:8080
/rest/acl_lock/auth1"

http://localhost:8080/rest/acl_lock/auth1

> echo "PREFIX acl: <http://www.w3.org/ns/auth/acl#>
INSERT INTO {
<> acl:accessControl <http://localhost:8080/rest/acl_lock> .
}" | curl -X PATCH -H "Content-type: application/sparql-update" --upload-file - "http://localhost:8080
/rest/dark/archive"

> curl -X POST -H "Content-type: text/turtle" --data-binary "@Acl.ttl" "http://localhost:8080/rest"

http://localhost:8080/rest/acl_open

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Auth_open.ttl" "http://localhost:8080/rest
/acl_open/auth2"

http://localhost:8080/rest/acl_open/auth2

> echo "PREFIX acl: <http://www.w3.org/ns/auth/acl#>
INSERT INTO {
<> acl:accessControl <http://localhost:8080/rest/acl_open> .
}" | curl -X PATCH -H "Content-type: application/sparql-update" --upload-file - "http://localhost:8080
/rest/dark/archive/sunshine"
```

4. The collection http://localhost:8080/rest/public_collection should be **readable** by anyone but only **editable** by users in the group **Editors**.

   Using the three "files" below to create our Authorization and ACL resources.

Acl.ttl

```
@prefix webac: <http://fedora.info/definitions/v4/webac#> .
<> a webac:Acl .
```

Auth1.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<> a acl:Authorization ;
   acl:agent foaf:Agent ;
   acl:mode acl:Read ;
   acl:accessTo <http://localhost:8080/rest/public_collection> .
```

Auth2.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
<> a acl:Authorization ;
   acl:agent "Editors" ;
   acl:mode acl:Read, acl:Write ;
   acl:accessTo <http://localhost:8080/rest/public_collection> .
```

I would execute the following code:

```
> curl -X POST -H "Content-type: text/turtle" --data-binary "@Acl.ttl" "http://localhost:8080/rest"

http://localhost:8080/rest/acl

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Auth1.ttl" "http://localhost:8080/rest/acl
/auth1"

http://localhost:8080/rest/acl/auth1

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Auth2.ttl" "http://localhost:8080/rest/acl
/auth2"

http://localhost:8080/rest/acl/auth2

> echo "PREFIX acl: <http://www.w3.org/ns/auth/acl#>
INSERT INTO {
<> acl:accessControl <http://localhost:8080/rest/acl> .
}" | curl -X PATCH -H "Content-type: application/sparql-update" --upload-file - "http://localhost:8080
/rest/public_collection"
```

5. Only the *ex:publicImage* type objects in the container http://localhost:8080/rest/mixedCollection are viewable by anyone, all others are only viewable by the group **Admins**.

   Using the three "files" below to create our Authorization and ACL resources.

Acl.ttl

```
@prefix webac: <http://fedora.info/definitions/v4/webac#> .
<> a webac:Acl .
```

Auth_restricted.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
<> a acl:Authorization ;
   acl:agent 'Admins' ;
   acl:mode acl:Read ;
   acl:accessTo <http://localhost:8080/rest/mixedCollection> .
```

Auth_open.ttl

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<> a acl:Authorization ;
   acl:agent foaf:Agent ;
   acl:mode acl:Read ;
   acl:accessToClass ex:publicImage .
```

I would execute the following commands:

```
> curl -X POST -H "Content-type: text/turtle" --data-binary "@Acl.ttl" "http://localhost:8080/rest"

http://localhost:8080/rest/acl

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Auth_restricted.ttl" "http://localhost:8080
/rest/acl/auth1"

http://localhost:8080/rest/acl/auth1

> curl -X PUT -H "Content-type: text/turtle" --data-binary "@Auth_open.ttl" "http://localhost:8080/rest
/acl/auth2"

http://localhost:8080/rest/acl/auth2

> echo "PREFIX acl: <http://www.w3.org/ns/auth/acl#>
INSERT INTO {
<> acl:accessControl <http://localhost:8080/rest/acl> .
}" | curl -X PATCH -H "Content-type: application/sparql-update" --upload-file - "http://localhost:8080
/rest/mixedCollection"
```