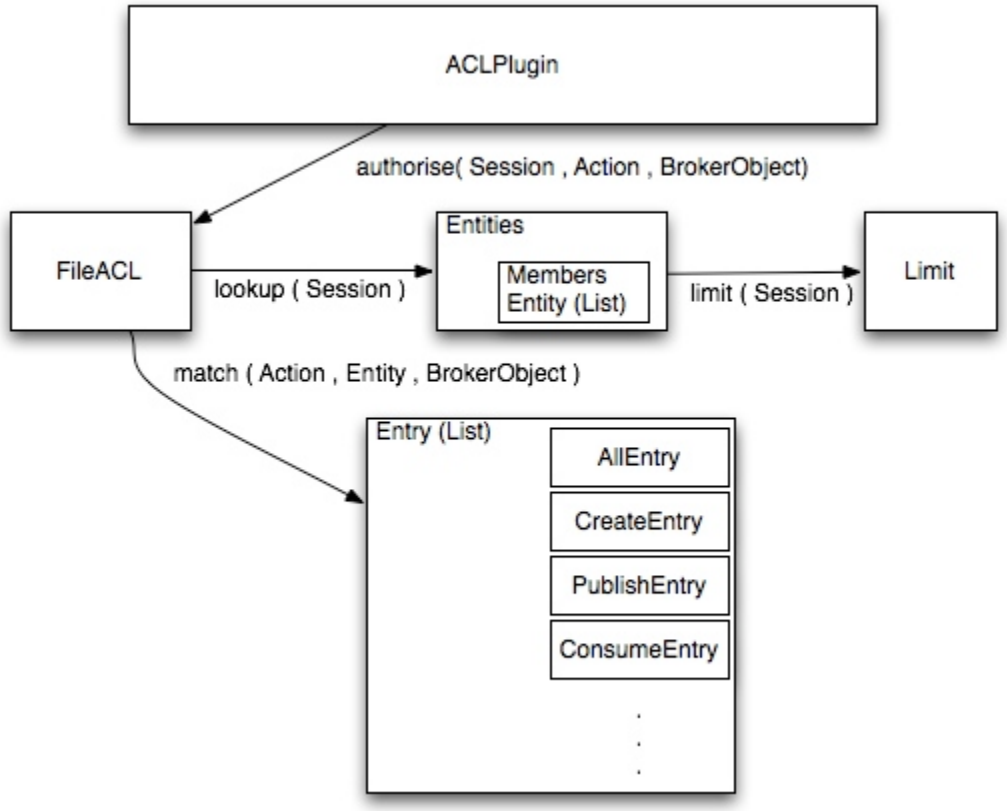
FileACL Design

FileACL : ACL v2 Java design documentation

Desgin



Interface

The current ACLPlugin interface has a single authorise method, however its current format ties it to the Framing layer. The interface has been abstracted to take Actions and a new BrokerObject on which the action should be performed.

```
    authorise( Session, Action , BrokerObject )
```

BrokerObjects

These new objects are taken from the ACLv2 documentation are used to represent the internal broker objects. These objects can be created with the properties so the ACL can be evaluated without needing access to the functional broker components. Providing the actual broker objects is not possible as that would require items such as an AMQQueue to be created before evaluating wither the the User has rights to create the queue.

ACL Entries

Each line from the ACL file is converted into an Entry and added to a list maintaining order for later evaluation. Each ACL Action type maps to an entry which in turn handles the processing. The Entries are much smaller and clearer to understand than the large case statement method that was utilised in the SimpleXML ACL Plugin. There is also scope here to provide an extension point to limit the ability of an entity.

```
[user|group] ... limit-<limit-type>=<value>
```

Examples would be to limit the number of connections a user may create or IP White/Black listing.

Current Development State

Currently the FileACL processing of the file format is complete and unit tested. Each of the entries have been created however they all do not fully take in to consideration all the potential variations of Objects and Properties that can be specified.

Testing has started by modifying the existing SimpleACLTest to allow different configuration and ACLPlugins to be loaded and evaluated against the existing Request/Response application design.

Items to complete

1. Complete implementation and testing of all Object an Property combinations
2. Complete parsing of *user* to understand Realms and Domains.
3. Provide an ACL independent mechanism for testing ACLs that can be performed against both Java & C++ brokers to ensure consistency in implementation. This would also allow future ACL implementations to be tested for consistency with existing implementations.

Note: What do these new properties mean for the Java broker.

Integration with existing ACLPlugin

The development of this plugin has been done to require no changes to the existing broker. For clarity a rename of the Permission class to Action has been carried. The introduction of the above interface needs further discussion as any new interface will require future support.

For the moment the existing ACLPlugin interface has been implemented and maps from the Framing layer to Actions and BrokerObjects.

Future Refinement

Analysing each ACL Entry in turn for a large acl file would be expensive however, it would be possible to perform some load time optimisations.

Examples of such optimisations are:

- Each Action could have it's own list so as to eliminate the lookup and method invocation on Entries that will never succeed.
- Each Entity could also have a list of entries attached to it that would allow quicker evaluation of the entries based on those that pertain to the given Entity
- Both of these optimisations can be applied to allow a much shorter list of Actions to be retrieved for a given Entity.