Authentication in OA4MP

Authentication – the means by which users are identified as valid[1] -- is not as simple as one would like in OA4MP. Since it is to be “open authorization for many people”, it does not, in fact, have a native concept of a user, but relies on some added mechanism. This permits institutions to use their existing infrastructure with OA4MP rather than having to give every user yet another set of credentials on an OA4MP server. There are 4 main ways this integration can be done.

1. *Proxy case.* Use a proxy to a trusted OAuth service that has all of your users. Their login there is accepted as a valid authentication.
2. *Header case.* Configure OA4MP to use an HTTP header from a web server. The standard is the REMOTE_USER header, but you may use any other header of your choice). Most commonly if Tomcat is hosting OA4MP, then Tomcat users can be accepted as valid OA4MP users. This also works with reverse proxies such as Apache or Nginx. Note that since it is easy to forge headers, you must manage the trust relation with the authenticating service.
3. *Extension case.* Extend OA4MP (in Java) directly to use your existing user management system. In this case there is already a system in place. Note that the default “out of the box” behavior for OA4MP is to reject any requests.
4. *Detached case.* Write your own Authentication Service (AS) which is wholly independent of OA4MP then use callouts to OA4MP to notify it of progress. This lets you put deploy your service along with OA4MP. CILogon is a famous example of this.
5. *Dedicated issuer case.* Write your own service that handles all requests up until token issuance. OA4MP then will issue them for flows.

1 In OA4MP, authorization – what a user is allowed to do – is set by policies.

6.