UMA Spec in my own words

Terms

authorizing user: An UMA-defined variant of an [OAuth20] resource owner; a web user who configures an authorization manager with policies that control how it makes access decisions when a requester attempts to access a protected resource at a host.

authorization manager (AM): An UMA-defined variant of an [OAuth20] authorization server that carries out an authorizing user's policies governing access to a protected resource.

protected resource: An access-restricted resource at a host.

host: An UMA-defined variant of an [OAuth20] resource server that enforces access to the protected resources it hosts, as decided by an authorization manager.

token validation URL: The URL at an authorization manager that a host can use to validate an access token.

claim: A statement (in the sense of [IDCclaim]). Claims are conveyed by a requester on behalf of a requesting party to an authorization manager in an attempt to satisfy an authorizing user's policy.

requester: An UMA-defined variant of [OAuth20] client that seeks access to a protected resource.

requesting party: A web user, or a corporation (or other legal person), that uses a requester to seek access to a protected resource.

Requirements

Requirements for the resource owner

- The resource owner MUST be able to choose a different AM for each Host
- (The resource owner MUST be able to choose a different AM for each protected resource)

Requirements for the protected resource

Overview

The protocol is divided into three parts:

1. The introduction of the Host and Authorization Manager
2. The retrieval of an Access Token for a Host by a Requester
3. The access to the Protected Resource on the Host by the Requester

The first step is only needed once per Host and Authorization Manager. In this step the resource owner decides which AM is handling access to the protected resource on that Host. The resource owner can either choose
The Resource Owner which has a protected resource on Host decides that the access to this resource is being managed by the AM he chooses.

Step 1: Introduction of Host and AM

In this step the following needs to be achieved:

- The Host needs to get an Access Token from the AM to later be able to verify Access Tokens from Requesters. Lets name them AM-Host Access Tokens or AH-AT for short.
- The Resource Owner needs to define conditions the Requesters need to fulfill before access t the protected resource is granted. This can e.g. be "is from domain twitter.com". This is done in terms of claims the Requester must send. Claims are basically attributes it sends and which can be verified eventually by the AM.
- The AM needs to know some details about the Host, e.g. the title and description. This is important as in communication with the Resource Owner it needs to tell it some details about the which protected resource the Requester actually wants access to. So maybe it even needs to be described on a resource by resource basis? The user might not really know much about "xyz wants access to method POST on http://someapi.com/mrtopf/")

The protocol involves some steps:

1. User provisions host with AM location (out of band)
2. Host obtains AM metadata via hostmeta
3. Host initiates an OAuth 2.0 Web Server flow to obtain an access token from the AM

The User provisions the Host with an AM location

How this is done is out of scope. Examples can be that the URL is entered or that the Host looks up the User's webfinger profile and finds the default AM in there. The AM is only given as domain name.

The Host obtains the AM metadata via a hostmeta lookup

The AM provides the following information in it's XRD:

- the authorization endpoint URL for the Host to initiate an Access Token authorization (Host authorization endpoint)
- the token endpoint URL for the Host to exchange the temporary token with the access token (Host Token endpoint)
- the authorization endpoint URL for the Requester to initiate an Access Token authorization (Requester Authorization endpoint)
- the token endpoint URL for the Requester to exchange the temporary token with the access token (Requester token endpoint)
- (some endpoint for verifying access to protected resources. This will become clearer when thinking about Step 3)

(example XRD here)

The Host initiates an OAuth 2.0 Web Server flow to obtain an access token for the AM

The normal OAuth flow is started here with the Host Authorization Endpoint found in the metadata above. The result is the Host having an Access Token from the AM.

During this flow the user will be redirected to the AM and needs to configure the policies for this host.

Questions

- How does the AM learn details about the Host? It needs to know
  - it's name, description, maybe TOS
  - which resources are available and to be configured
- How do AM and Host work to together? Shouldn't the policies maybe be defined on the Host as it knows much more about the semantics of it's own resources. It might then send profile identifiers to the AM which this can protect (or so..)
- Is there a distinction in claims about the requester and claims about the requesting party? (e.g. requester "allow access to newspaper.com because I just talked to them", requesting party: "needs to be older than 18").

Brainstorming

- Would it also makes sense the other way round in that the AM actually retrieves an Access Token from the Host? That way the AM could serve as a proxy and more or less relay an access token produced on the Host to the Requester while still doing auditing and maybe filtering access according to some policies (but then again: Wouldn't these policies be better handled on the Host side anyway?).

- From the calendaring example: A protected resource might have different views depending on who is asking: Sometimes it's full details, sometimes only free/busy, sometimes it might be in town/out of town information. Can this be solved by the Host defining different "profiles" which mean those access levels and could these be sent to the AM in order to show them to the user in order to select one? (or being automatically chosen)

Step 2: Requester obtains Access Token from AM for Host

Step 3: Requester accesses Protected Resource on Host