hueniverse

Protocol Workflow

OAuth is best explained with real-life examples. The specification introduction (http://tools.ietf.org/html/rfc5849#section-1.2) includes a similar example but focuses on the HTTP calls syntax. This walk-through demonstrates a typical OAuth session and includes the perspectives of the resource owner, client, and server. The websites and people mentioned are fictional. The Scottish references are real. And so our story begins…

Jane is back from her Scotland vacation. She spent 2 weeks on the island of Islay sampling Scotch. When she gets back home, Jane wants to share some of her vacation photos with her friends. Jane uses Faji, a photo sharing site, for sharing journey photos. She signs into her faji.com account, and uploads two photos which she marks private.

Using OAuth terminology, Jane is the resource owner and Faji the server. The 2 photos Jane uploaded are the protected resources.
After sharing her photos with a few of her online friends, Jane wants to also share them with her grandmother. She doesn’t want to share her rare bottle of Scotch with anyone. But grandma doesn’t have an internet connection so Jane plans to order prints and have them mailed to grandma. Being a responsible person, Jane uses Beppa, an environmentally friendly photo printing service.

Using OAuth terminology, Beppa is the client. Since Jane marked the photos as private, Beppa must use OAuth to gain access to the photos in order to print them.

Jane visits beppa.com and begins to order prints. Beppa supports importing images from many photo sharing sites, including Faji. Jane selects the photos source and clicks Continue.
When Beppa added support for Faji photo import, a Beppa developer known in OAuth as a client developer obtained a set of client credentials (client identifier and secret) from Faji to be used with Faji’s OAuth-enabled API.

After Jane clicks Continue, something important happens in the background between Beppa and Faji. Beppa requests from Faji a set of temporary credentials. At this point, the temporary credentials are not resource-owner-specific, and can be used by Beppa to gain resource owner approval from Jane to access her private photos.
Jane clicked Continue and is now waiting for her screen to change. She sips from her prized Black Bowmore while waiting for the next page to load.

When Beppa receives the temporary credentials, it redirects Jane to the Faji OAuth User Authorization URL with the temporary credentials and asks Faji to redirect Jane back once approval has been granted to http://beppa.com/order.

Jane has been redirected to Faji and is requested to sign into the site. OAuth requires that servers first authenticate the resource owner, and then ask them to grant access to the client.

Jane notices she is now at a Faji page by looking at the browser URL, and enters her username and password.

OAuth allows Jane to keep her username and password private and not share them with Beppa or any other site. At no time does Jane enters her credentials into beppa.com.

After successfully logging into Faji, Jane is asked to grant access to Beppa, the client. Faji informs Jane of who is requesting access (in this case Beppa) and the type of access being granted. Jane can approve or deny access.

Jane makes sure Beppa is getting the limited access it needs. She does not want to allow Beppa to change her photos or do anything else to them. She also notes this is a onetime access good for one hour which should be enough time for Beppa to fetch her photos.
Once Jane approves the request, Faji marks the temporary credentials as resource-owner-authorized by Jane. Jane’s browser is redirected back to Beppa, to the URL previously provided http://beppa.com/order (http://beppa.com/order) together with the temporary credentials identifier. This allows Beppa to know it can now continue to fetch Jane’s photos.

*Jane waits for Beppa to present her with her photos fetched from her Faji account.*
While Jane waits, Beppa uses the authorized Request Token and exchanges it for an Access Token. Request Tokens are only good for obtaining User approval, while Access Tokens are used to access Protected Resources, in this case Jane’s photos. In the first request, Beppa exchanges the Request Token for an Access Token and in the second (can be multiple requests, one for a list of photos, and a few more to get each photo) request gets the photos.

(https://hueniversedotcom.files.wordpress.com/2009/12/flow3grfc.png)

When Beppa is done, Jane’s browser refreshes to complete the order.
Beppa successfully fetched Jane’s photo. They are presented as thumbnails for her to pick and place her order.

Jane is very impressed how Beppa grabbed her photos without asking for her username and password. She likes what she sees and places the print order.

Continue to Security Framework (http://hueniverse.com/oauth/guide/security/)

14 thoughts on “Protocol Workflow”

1. Giordano says:
   August 10, 2011 at 2:58 am
   Very nice and clear explanation, thanks!

2. Karl B, Minn, MN says:
   September 15, 2011 at 12:40 pm
   Thanks for the explanation – well illustrated and to the point.

3. mani says:
November 11, 2011 at 2:42 am
excellent work flow shown.

4. **Eddie says:**
   January 21, 2012 at 2:09 am
   Great demonstration. But what happens if she revisits Beppa again? Does she have to go through the same process?

   - **Eran Hammer says:**
     January 23, 2012 at 11:39 pm
     That depends on how the client chooses to implement it. They can keep the access token and look it up using a session cookie. Also, the provider can also automatically redirect back without prompting the user to do it again.

   - **Franck says:**
     February 26, 2012 at 9:11 pm
     Keeping the access token would not help much since it’s valid for just 1 hour. (correct?) When you say the provider (the server, Faji) can also automatically redirect back…, do you mean it can use it’s own cookie (or other ways) to automatically authenticate the resource owner and redirect back to the Beppa site without logging in the Faji web-site? In that case, Jane would not know (I mean not explicitly like the first time) the second time that she gave access to the Beppa site, right?

   - **Eran Hammer says:**
     February 28, 2012 at 12:41 am
     Access tokens can last longer than 1 hour. It all depends on the server’s security policy. As for the blind redirect, yes, Jane will just see her photos appear without granting access again.

   - **Franck says:**
     February 28, 2012 at 10:03 am
     which brings me to the following question. Let’s say I’ve been using the Faji site for some time and it remembers me each time I connect to it (I don’t need to provide my login/password before it’s keeping all needed on my computer). Would that mean, using the Beppa site to print my photos that even the first time, it would not ask me for my credential and I would in that case not be aware that I’m actually providing access to my resources to a third party? I did some test with different site and I think I saw that behavior. Off course the server could be implemented in a way that the first time it would ask for the credentials, but it doesn’t sounds like it’s what happening in the real world.

5. **Oleg Derid says:**
   January 27, 2012 at 12:07 pm
   Great example.
From privacy point of view I would not like Beepa to fetch all my photos from Faji. We can see that Beepa fetches all photos, so from client perspective there is a concern that Beepa could fetch and catch on server side photos I wouldn’t like them to show.

I think there is an architectural way to solve this problem:
a. explicitly restrict some photos on Faji side (make them private, so that external party like Beepa can’t fetch them).
b. when granting authorization on Faji side choose which photos to share.

6. Rich says:
   June 21, 2012 at 7:22 am
   hah... what are the chances...? I just returned from a two-week trip to Scotland sampling high-end single malt scotch whiskies. I return to the US and begin researching federated security systems, including OAuth, and the first site I find is yours. Probably the first tech tutorial ever to include references to a Scotland vacation, let alone collector-quality whiskies. I’m guessing you must have a single malt enthusiast on staff...?
   
   By the way, no one “sips from” a Black Bowmore...

7. Dominic says:
   August 1, 2012 at 1:47 am
   Excellent illustration!

8. Naina says:
   October 6, 2012 at 5:19 pm
   Excellent (and entertaining) example! Thanks, you just made my life a little easier.

9. vinay says:
   October 12, 2012 at 12:22 am
   You made it very easy to understand..thank you..

10. Vishy Ranganath says:
    November 17, 2012 at 7:40 am
    Thank you so much for this lucid explanation of the OAuth flow. I created a diagram to help understand the flow. Here is the link. It will be great if people can review it and point out any errors. Here is a link to the diagram:
    
    Thank you,
    Vishy

   Comments are closed.

Follow “hueniverse”

Build a website with WordPress.com