OpenID Connect Quick Start

- Introduction
- Ingredients
- Step 1: Prerequisites and Assumptions
- Step 2: Configure OAuth2/OpenID Connect Service (OpenAM)
- Step 3: Configure OAuth2/OpenID Client Configuration (OpenAM)
- Step 4: Setup Simple Client
- Step 5: Loading OpenAM IDP Metadata into Sample
- Step 6: Click on link and login

Introduction

The goal of this guide is to describe a recipe of how to build a rapid model to test OpenID Connect with a simple as possible browser-based client. This sample client can then be extended based upon any web language and/or framework of choice.

Ingredients

- JDK 1.8 or higher: [http://www.oracle.com/technetwork/java/javase/downloads/index.html](http://www.oracle.com/technetwork/java/javase/downloads/index.html)
- Tomcat 7.0 for OpenAM (other containers could be used): [http://tomcat.apache.org/download-70.cgi](http://tomcat.apache.org/download-70.cgi)
- OpenAM [https://backstage.forgerock.com/#!/downloads/enterprise/OpenAM](https://backstage.forgerock.com/#!/downloads/enterprise/OpenAM)
- Sample browser-based client: [https://github.com/sjarosz/oidc-simple-client](https://github.com/sjarosz/oidc-simple-client)

Step 1: Prerequisites and Assumptions

Install JDK 1.8, Tomcat per their recommendations

It is assumed the FQDN for both instances is forgerock.example.com. These would not need to be the same in real world just used to make easy test environment.

OpenAM Installation video: [https://www.youtube.com/watch?v=5X1cWnMDtH0](https://www.youtube.com/watch?v=5X1cWnMDtH0)

Step 2: Configure OAuth2/OpenID Connect Service (OpenAM)

First step is to configure the OpenAM to support OpenID Connect and thereby OAuth2.

Login into OpenAM Admin console as AmAdmin select the default realm and select Configure OAuth Provider link from Common Tasks

![Common Tasks > Configure OAuth Provider](image)

Next choose Configure OpenID Connect

For the most part choose defaults. It may be desired, at least until debugging is completed that the Lifetime settings are made rather high.
Towards the bottom of this screen select *Always Return Claims in ID Tokens.*

**Step 3: Configure OAuth2/OpenID Client Configuration (OpenAM)**

Create a new Agent of type OAuth2.0/OpenID Connect Client, call it *MyClientID* and password of *password*. This ID name can be different and unique for your deployment, but per this example stick with this name.

Edit the MyClientID settings and go with defaults, however a valid *Redirect URIs* needs to be available.

In this example with the sample application should be, *http://localhost:8000* can be added.
Step 4: Setup Simple Client

```bash
mkdir ~/web
create index.html
```
<html>
<head>
<script>
function gup( name, url ) {
    if (!url) url = location.href;
    name = name.replace(/\[/,\"\[\").replace(/\]/,\"\]\")
    var regexS = "\([^\&]*\)\+name\+==\([^\&]*\)\+";
    var regex = new RegExp( regexS );
    var results = regex.exec( url );
    return results == null ? null : results[1];
}

var outString="";
var idToken = gup(\'id_token\');
if (idToken!=null) outString = (JSON.parse(atob(idToken.split(\./)[1])));
if (idToken!=null) console.log(outString);
if (idToken!=null) alert("Hello "+outString.sub);
</script>

</head>
<body>
</body>
</html>

Start Web Server

python -m SimpleHTTPServer 8000

Step 5: Loading OpenAM IDP Metadata into Sample
Step 6: Click on link and login

Afterwards redirect back to client application and watch the Hello Alert box display the user.

Also, if choosing the developer console in the browser the entire unencoded JSON object is displayed.

References

https://wikis.forgerock.org/confluence/display/openam/OpenID+Connect+-+Curl+Commands
http://openid.net
https://github.com/ForgeRock/openid