181211-12 @ Shibboleth OIDC Extension Tutorial

Agenda

	First Day
12:00-	Lunch
12:30 - 1:15pm	Introduction
1:15 - 2:00pm	Installation
2:00 - 3:00pm	Trust management + OP configuration
3:00 - 3:30pm	Break
3:30 - 4:30pm	Authentication
	Second Day
10:00 - 10:45am	Attribute definitions
10:45 -11:30am	Attribute filtering
11:30 - 12:00am	Subject identifier
12:00 - 12:45	Lunch
12:45 - 2:00pm	Profile configurations

Table of Contents

Introduction

Section Topics

- Project in general
- Project resources (releases, documentation and source code)
- Support channels
- Tutorial logistics

Project in general

OpenID Connect

- https://openid.net/connect/
  - OAuth 2.0 based interfaces have become very popular because they were chosen by the social media providers
  - Relying Party libraries exist for many programming languages
- Specifications
  blocked URL
  - Core: http://openid.net/specs/openid-connect-core-1_0.html
  - Discovery: http://openid.net/specs/openid-connect-discovery-1_0.html
  - Dynamic registration: http://openid.net/specs/openid-connect-registration-1_0.html
- Certification tool + programme
  - https://openid.net/certification/

Shibboleth OIDC plugin overview

Project resources

Releases and Documentation

- https://github.com/CSCfi/shibboleth-idp-oidc-extension/releases
  - https://github.com/CSCfi/shibboleth-idp-oidc-extension/releases/tag/v0.5.0a
  - https://github.com/CSCfi/shibboleth-idp-oidc-extension/releases/tag/v0.6.0a
  - https://github.com/CSCfi/shibboleth-idp-oidc-extension/releases/tag/v0.7.0a
  - https://github.com/CSCfi/shibboleth-idp-oidc-extension/releases/tag/v0.8.0b
- https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki

Source code

- https://github.com/CSCfi/shibboleth-idp-oidc-extension
  - idp-oidc-extension-parent
  - idp-oidc-extension-api
  - idp-oidc-extension-impl

Clone from GitHub

```
          bash-3.2$ git clone https://github.com/CSCfi/shibboleth-idp-oidc-extension
Cloning into 'shibboleth-idp-oidc-extension'...
remote: Enumerating objects: 894, done.
remote: Counting objects: 100% (894/894), done.
remote: Compressing objects: 100% (462/462), done.
remote: Total 16265 (delta 292), reused 783 (delta 204), pack-reused 15371
Receiving objects: 100% (16265/16265), 1.78 MiB | 2.03 MiB/s, done.
Resolving deltas: 100% (5582/5582), done.
bash-3.2$ cd shibboleth-idp-oidc-extension
```

Compile with Maven

```
          bash-3.2$ mvn -f idp-oidc-extension-parent/pom.xml package
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] idp-oidc-extension-parent
[INFO] idp-oidc-extension-api
[INFO] idp-oidc-extension-impl
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building idp-oidc-extension-parent 0.8.0-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-site-plugin:3.7:attach-descriptor (attach-descriptor) @ idp-oidc-extension-parent ---
[INFO] No site descriptor found: nothing to attach.
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:test-jar (default) @ idp-oidc-extension-parent ---
[INFO] Skipping packaging of the test-jar
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building idp-oidc-extension-api 0.8.0-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ idp-oidc-extension-api ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] ------------------------------------------------------------------------


...

[INFO] Reactor Summary:
[INFO]
[INFO] idp-oidc-extension-parent ......................... SUCCESS [  1.478 s]
[INFO] idp-oidc-extension-api ............................ SUCCESS [  5.131 s]
[INFO] idp-oidc-extension-impl ........................... SUCCESS [ 14.305 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 21.243 s
[INFO] Finished at: 2018-10-04T20:07:01+03:00
[INFO] Final Memory: 47M/517M
[INFO] ------------------------------------------------------------------------
```

Support channels

Support channels
Shibboleth mailing lists: https://www.shibboleth.net/community/lists/ • Currently, before official releases: https://marc.info/?l=shibboleth-dev GitHub issues ▪ https://github.com/CSCfi/shibboleth-idp-oidc-extension/issues

Tutorial logistics

Virtual machines
Everybody should have a paper note containing **IP address** and a **password** for *cloud-user*

The virtual machines are running CentOS 7 with the following software already installed

- OpenJDK 8 JRE
  - JAVA_HOME = /usr/lib/jvm/jre-1.8.0
- Apache 2
  - Running on ports 80 and 8443
  - OICD RP module mod_auth_openidc
- Jetty v9.4.2
  - Running on ports 8080 and 443
  - /opt/jetty
  - /opt/shibboleth-idp/jetty-base
- MariaDB
  - Database name for IdP: 'idp'
  - Username 'idp' password: 'not_set_yet'
- 389 Directory Server
  - Admin 'cn=Directory Manager' and pwd 'testpword'
  - End-user ***teppo***, password ***testaaja***
- Shibboleth IdP 3.4
  - service name ***shibboleth-idp***
  - IDP home directory */opt/shibboleth-idp*

Exercises

Exercise 1.1 - Check VM connection

1. Verify that you can log in to your virtual machine
    a. SSH-connection to the IP address as *cloud-user* with the given password
2. Restart the *shibboleth-idp* service

Hints, Tips and Result

```
          [vagrant@gn43-oidcshibop-devel ~]$ sudo su
[root@gn43-oidcshibop-devel vagrant]# systemctl stop shibboleth-idp
[root@gn43-oidcshibop-devel vagrant]# systemctl start shibboleth-idp
```

3. Verify from the logs that it starts up without errors

Hints, Tips and Result

```
          [root@gn43-oidcshibop-devel vagrant]# tail -f /opt/shibboleth-idp/logs/idp-process.log
2018-09-28 22:35:50,509 - INFO [net.shibboleth.ext.spring.context.FilesystemGenericApplicationContext:
583] - Refreshing shibboleth.ReloadableAccessControlService: startup date [Fri Sep 28 22:35:50 UTC
2018]; parent: Root WebApplicationContext
2018-09-28 22:35:50,562 - INFO [net.shibboleth.ext.spring.service.ReloadableSpringService:421] -
Service 'shibboleth.ReloadableAccessControlService': Completed reload and swapped in latest
configuration for service 'shibboleth.ReloadableAccessControlService'
2018-09-28 22:35:50,562 - INFO [net.shibboleth.ext.spring.service.ReloadableSpringService:428] -
Service 'shibboleth.ReloadableAccessControlService': Reload complete
2018-09-28 22:35:50,576 - INFO [net.shibboleth.utilities.java.support.service.
AbstractReloadableService:199] - Service 'shibboleth.ReloadableAccessControlService': Reload time set
to: 300000, starting refresh thread
2018-09-28 22:35:50,585 - INFO [net.shibboleth.utilities.java.support.service.
AbstractReloadableService:173] - Service 'shibboleth.ReloadableCASServiceRegistry': Performing initial
load
2018-09-28 22:35:50,585 - INFO [net.shibboleth.utilities.java.support.service.
AbstractReloadableService:258] - Service 'shibboleth.ReloadableCASServiceRegistry': Reloading service
configuration
2018-09-28 22:35:50,587 - INFO [net.shibboleth.ext.spring.util.SchemaTypeAwareXMLBeanDefinitionReader:
317] - Loading XML bean definitions from file [/opt/shibboleth-idp/conf/cas-protocol.xml]
2018-09-28 22:35:50,596 - INFO [net.shibboleth.ext.spring.context.FilesystemGenericApplicationContext:
583] - Refreshing shibboleth.ReloadableCASServiceRegistry: startup date [Fri Sep 28 22:35:50 UTC
2018]; parent: Root WebApplicationContext
2018-09-28 22:35:50,647 - INFO [net.shibboleth.ext.spring.service.ReloadableSpringService:421] -
Service 'shibboleth.ReloadableCASServiceRegistry': Completed reload and swapped in latest
configuration for service 'shibboleth.ReloadableCASServiceRegistry'
2018-09-28 22:35:50,660 - INFO [net.shibboleth.ext.spring.service.ReloadableSpringService:428] -
Service 'shibboleth.ReloadableCASServiceRegistry': Reload complete
2018-09-28 22:35:50,672 - INFO [net.shibboleth.utilities.java.support.service.
AbstractReloadableService:199] - Service 'shibboleth.ReloadableCASServiceRegistry': Reload time set
to: 900000, starting refresh thread
2018-09-28 22:35:51,184 - WARN [net.shibboleth.utilities.java.support.net.CookieManager:171] - Use of
secure and httpOnly properties are strongly advisable, currently one or both are false
2018-09-28 22:35:51,911 - INFO [net.shibboleth.ext.spring.context.DelimiterAwareApplicationContext:
583] - Refreshing WebApplicationContext for namespace 'idp-servlet': startup date [Fri Sep 28 22:35:51
UTC 2018]; parent: Root WebApplicationContext
2018-09-28 22:35:51,943 - INFO [net.shibboleth.ext.spring.resource.ConditionalResource:87] -
ConditionalResource conditional:/opt/shibboleth-idp/conf/mvc-beans.xml: getInputStream failed on
wrapped resource
2018-09-28 22:35:51,944 - INFO [net.shibboleth.ext.spring.resource.ConditionalResource:87] -
ConditionalResource conditional:/opt/shibboleth-idp/conf/mvc-beans.xml: getInputStream failed on
wrapped resource
2018-09-28 22:35:52,832 - INFO [net.shibboleth.idp.authn.impl.RemoteUserAuthServlet:215] -
RemoteUserAuthServlet will process REMOTE_USER, along with attributes [] and headers []
```

Installation

Section Topics

- Building the 0.8.0 release
- Installing the extension from archive.

Exercises

Exercise 2.1 - Build

Build the beta release.

1. Install maven

```
sudo yum install maven
```

2. Checkout beta release

```
git clone https://github.com/CSCfi/shibboleth-idp-oidc-extension
cd shibboleth-idp-oidc-extension
git checkout tags/v0.8.0b
```

3. Build

```
cd idp-oidc-extension-parent
mvn package
```

Exercise 2.2 - Installation

Install the extension.

1. Locate the tar

```
/home/cloud-user/shibboleth-idp-oidc-extension/idp-oidc-extension-distribution/target/idp-oidc-extension-distribution-0.8.0-bin.tar.gz
```

2. Study the release notes: https://github.com/CSCfi/shibboleth-idp-oidc-extension/releases
3. Install extension from archive, instructions on https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki/Installing-from-archive. Apply following training specific further instructions.
   - "**Profile Configurations**" is the last step completed. "**Session Storage"** and further will not be done.
   - Before starting the process, become super user. Set environment variables to execute java.

```
[/opt/shibboleth-idp] sudo su -
[/opt/shibboleth-idp] source /etc/default/shibboleth-idp
```

   - Set log level for extension

```
nano +40 /opt/shibboleth-idp/conf/logback.xml

    <!-- ========================================================== -->
    <!-- ============== Logging Categories and Levels ============== -->
    <!-- ========================================================== -->

    <logger name="org.geant" level="ALL"/>
```

   - In "**Install the archive**" step after extracting the tar you want to set the ownership of the files to "**root:jetty**".

```
[/opt/shibboleth-idp] chown -R root:jetty *
```

   - In the phase you are requested to set your issuer identifier, set it to https://your_public_ip_address
   - In the phase you are requested to create keyset.jwk, set it as file /opt/shibboleth-idp/static/oidc/keyset.jwk
   - Activate provided oidc example attribute resolver and filter. These are needed in exercises.

```
[/opt/shibboleth-idp] cp /opt/shibboleth-idp/conf/attribute-filter-oidc-training.xml /opt/shibboleth-idp/conf/attribute-filter.xml
[/opt/shibboleth-idp] cp /opt/shibboleth-idp/conf/attribute-resolver-oidc-training.xml /opt/shibboleth-idp/conf/attribute-resolver.xml
```

Trust Management & OP configuration

Section Topics

- SAML IdP Metadata vs. OpenID Provider Configuration
- OIDC Client Information
- Trust relationship establishment between Shibboleth OP and test RP

SAML IdP Metadata vs. OpenID Provider Configuration

SAML IdP Metadata

- Example IdP metadata: https://idp.csc.fi/idp/shibboleth
  - By default, the contents are fetched from /opt/shibboleth-idp/metadata/idp-metadata.xml
  - The file is generated by the installation script and is **NOT** updated afterwards automatically
- Example federation metadata: https://haka.funet.fi/metadata/haka-metadata.xml
  - Signed by the federation operator (trusted 3rd party)

OpenID Provider Configuration

- OpenID Connect Discovery spec - responds to two questions
  1. How the OP configuration is found?
     - https://openid.net/specs/openid-connect-discovery-1_0.html#ProviderConfig
       - Webfinger protocol exploited for finding out the issuer name from the resource identifier
       - URL format: **<issuer>/.well-known/openid-configuration**
  2. What are the contents of the OP configuration?
     - https://openid.net/specs/openid-connect-discovery-1_0.html#ProviderMetadata
- Two ways to publish OP configuration in Shibboleth OP: https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki/DiscoveryAndOPConfiguration#openid-provider-metadata
  1. Static file - example found at */opt/shibboleth-idp/static/.well-known/openid-configuration*
  2. Static file with dynamic claims - example found at */opt/shibboleth-idp/flows/oidc/discovery* (URL-endpoint *../idp/profile/oidc/discovery*)

Exercises

Exercise 2.1 - Google's OP configuration

Google OP issuer name is 'https://accounts.google.com'

1. What is the endpoint URL for the openid-configuration?

   Hints, Tips and Result

   ```
   https://accounts.google.com/.well-known/openid-configuration
   ```

2. Check the contents of Google's openid-configuration

   Hints, Tips and Result


```
{
 "issuer": "https://accounts.google.com",
 "authorization_endpoint": "https://accounts.google.com/o/oauth2/v2/auth",
 "token_endpoint": "https://oauth2.googleapis.com/token",
 "userinfo_endpoint": "https://www.googleapis.com/oauth2/v3/userinfo",
 "revocation_endpoint": "https://oauth2.googleapis.com/revoke",
 "jwks_uri": "https://www.googleapis.com/oauth2/v3/certs",
 "response_types_supported": [
  "code",
  "token",
  "id_token",
  "code token",
  "code id_token",
  "token id_token",
  "code token id_token",
  "none"
 ],
 "subject_types_supported": [
  "public"
 ],
 "id_token_signing_alg_values_supported": [
  "RS256"
 ],
 "scopes_supported": [
  "openid",
  "email",
  "profile"
 ],
 "token_endpoint_auth_methods_supported": [
  "client_secret_post",
  "client_secret_basic"
 ],
 "claims_supported": [
  "aud",
  "email",
  "email_verified",
  "exp",
  "family_name",
  "given_name",
  "iat",
  "iss",
  "locale",
  "name",
  "picture",
  "sub"
 ],
 "code_challenge_methods_supported": [
  "plain",
  "S256"
 ]
}
```

Exercise 2.2 - Shibboleth OP Configuration

Everybody has a Shibboleth OP instance running on a virtual machine with public IP. The OP issuer name is https://IP_ADDRESS

- What is the endpoint URL for the openid-configuration?

  Hints, Tips and Result

  ```
  https://IP_ADDRESS/.well-known/openid-configuration
  ```

- What are the contents of the well-known endpoint?

  Hints, Tips and Result


```
{
"issuer":"https://192.168.0.150",
"authorization_endpoint":"https://192.168.0.150/idp/profile/oidc/authorize",
"registration_endpoint":"https://192.168.0.150/idp/profile/oidc/register",
"token_endpoint":"https://192.168.0.150/idp/profile/oidc/token",
"userinfo_endpoint":"https://192.168.0.150/idp/profile/oidc/userinfo",
"jwks_uri":"https://192.168.0.150/oidc/keyset.jwk",
"response_types_supported":[
   "code",
   "id_token",
   "token id_token",
   "code id_token",
   "code token",
   "code token id_token"
],
"subject_types_supported":[
   "public",
   "pairwise"
],
"grant_types_supported":[
   "authorization_code",
   "implicit",
   "refresh_token"
],
"id_token_encryption_alg_values_supported":[
   "RSA1_5"
],
"id_token_encryption_enc_values_supported":[
   "A128CBC-HS256"
],
"id_token_signing_alg_values_supported":[
   "RS256",
   "RS384",
   "RS512",
   "HS256",
   "HS384",
   "HS512",
   "ES256"
],
"userinfo_encryption_alg_values_supported":[
   "RSA1_5"
],
"userinfo_encryption_enc_values_supported":[
   "A128CBC-HS256"
],
"userinfo_signing_alg_values_supported":[
   "RS256",
   "RS384",
   "RS512",
   "HS256",
   "HS384",
   "HS512",
   "ES256"
],
```

```
   "request_object_signing_alg_values_supported":[
      "none",
      "RS256",
      "RS384",
      "RS512",
      "HS256",
      "HS384",
      "HS512",
      "ES256",
      "ES384",
      "ES512"
   ],
   "token_endpoint_auth_methods_supported":[
      "client_secret_basic",
      "client_secret_post",
      "client_secret_jwt",
      "private_key_jwt"
   ],
   "claims_parameter_supported":true,
   "request_parameter_supported":true,
   "request_uri_parameter_supported":false,
   "require_request_uri_registration":false,
   "display_values_supported":[
      "page"
   ],
   "scopes_supported":[
      "openid",
      "profile",
      "email",
      "address",
      "phone",
      "offline_access"
   ],
   "response_modes_supported":[
      "query",
      "fragment",
      "form_post"
   ],
   "claims_supported":[
      "aud",
      "iss",
      "sub",
      "iat",
      "exp",
      "acr",
      "auth_time",
      "email",
      "email_verified",
      "address",
      "phone",
      "phone_number_verified",
      "name",
      "family_name",
      "given_name",
      "middle_name",
      "nickname",
      "preferred_username",
      "profile",
      "picture",
      "website",
      "gender",
      "birthdate",
      "zoneinfo",
      "locale",
      "updated_at"
   ]
}
```

Exercise 2.3 - (Advanced) Dynamic Shibboleth OP Configuration

- Enable *OIDC.Configuration* in the *shibboleth.UnverifiedRelyingParty* profile configurations (relying-party.xml)

  Hints, Tips and Result

```
            <bean id="shibboleth.UnverifiedRelyingParty" parent="RelyingParty">
      <property name="profileConfigurations">
          <list>
              ...
              <ref bean="OIDC.Configuration" />
          </list>
      </property>
  </bean>
```

- What are the contents of the /idp/profile/oidc/discovery -endpoint? HINT: Copy-paste it to jsolint.com service for making it "human-readable".

  Hints, Tips and Result


```
         {"issuer":"https:\/\/192.168.0.150","jwks_uri":"https:\/\/192.168.0.150\/oidc\/keyset.jwk","
authorization_endpoint":"https:\/\/192.168.0.150\/idp\/profile\/oidc\/authorize","token_endpoint":"
https:\/\/192.168.0.150\/idp\/profile\/oidc\/token","registration_endpoint":"https:\/\/192.168.0.150\
/idp\/profile\/oidc\/register","scopes_supported":["openid","profile","email","address","phone","
offline_access"],"response_types_supported":["code","id_token","id_token token","code id_token","code
token","code id_token token"],"response_modes_supported":["query","fragment","form_post"],"
grant_types_supported":["authorization_code","implicit","refresh_token"],"
token_endpoint_auth_methods_supported":["client_secret_basic","client_secret_post","
client_secret_jwt","private_key_jwt"],"request_object_signing_alg_values_supported":["none","RS256","
RS384","RS512","HS256","HS384","HS512","ES256","ES384","ES512"],"request_parameter_supported":true,"
request_uri_parameter_supported":false,"require_request_uri_registration":false,"
tls_client_certificate_bound_access_tokens":false,"signing-keys":[{"kty":"RSA","e":"AQAB","use":"sig","
kid":"testkeyRS","n":"pNf03ghVzMAw5sWrwDAMAZdSYNY2q7OVlxMInljMgz8XB5mf8XKH3EtP7AKrb8IAf7rGhfuH3T1N1C7F-
jwIeYjXxMm2nIAZ0hXApgbccvBpf4n2H7IZflMjt4A3tt587QQSxQ069drCP4sYevxhTcLplJy6RWA0cLj-
5CHyWy94zPeeA4GRd6xgHFLz0RNiSF0pF0kE4rmRgQVZ-
b4_BmD9SsWnIpwhms5Ihciw36WyAGQUeZqULGsfwAMwlNLIaTCBLAoRgv370p-XsLrgz86pTkNBJqXP5GwI-
ZfgiLmJuHjQ9l85KqHM87f-QdsqiV8KoRcslgXPqb6VOTJBVw"},{"kty":"EC","crv":"P-256","x":"
2uzfE1oK0cf1_c11SFc9vFdGLnJoH3e0AKTrGPAmUis","y":"14410NGKqwLM58b26ZcvGOruFixpHt_SJTw8I5wwgLQ","alg":"
ES256"}],"subject_types_supported":["public","pairwise"],"userinfo_endpoint":"https:\/\/192.168.0.150\
/idp\/profile\/oidc\/userinfo","id_token_signing_alg_values_supported":["RS256","RS384","RS512","
ES256","HS256","HS384","HS512"],"id_token_encryption_alg_values_supported":["A128CBC-HS256"],"
id_token_encryption_enc_values_supported":["A128CBC-HS256"],"userinfo_signing_alg_values_supported":
["RS256","RS384","RS512","ES256","HS256","HS384","HS512"],"userinfo_encryption_alg_values_supported":
["A128CBC-HS256"],"userinfo_encryption_enc_values_supported":["A128CBC-HS256"],"
display_values_supported":["page"],"claims_supported":["aud","iss","sub","iat","exp","acr","
auth_time","email","email_verified","address","phone","phone_number_verified","name","family_name","
given_name","middle_name","nickname","preferred_username","profile","picture","website","gender","
birthdate","zoneinfo","locale","updated_at"],"claims_parameter_supported":true,"
frontchannel_logout_supported":false,"backchannel_logout_supported":false}
```


- Why they are different? Check */opt/shibboleth-idp/flows/oidc/discovery/discovery-beans.xml*
  - **Hints, Tips and Result**

```
          ...

    <!-- The 5 entries below in the dynamicValueResolvers map (see entry key="...") are dynamically
built. -->


    <bean id="shibboleth.oidc.OpenIdConfigurationResolver"
        class="org.geant.idpextension.oidc.metadata.impl.DynamicFilesystemProviderMetadataResolver">
        <property name="dynamicValueResolvers">
            <map value-type="org.geant.idpextension.oidc.metadata.resolver.MetadataValueResolver">
                <entry key="signing-keys" value-ref="CredentialResolver" />
                <entry key="id_token_signing_alg_values_supported" value-ref="
SignatureAlgorithmInfoResolver" />
                <entry key="userinfo_signing_alg_values_supported" value-ref="
SignatureAlgorithmInfoResolver" />
                <entry key="id_token_encryption_alg_values_supported" value-ref="
EncryptionAlgorithmInfoResolver" />
                <entry key="userinfo_encryption_alg_values_supported" value-ref="
EncryptionAlgorithmInfoResolver" />
            </map>
        </property>
        <constructor-arg type="java.io.File"
            value="/opt/shibboleth-idp/static/.well-known/openid-configuration" />
    </bean>
...
```

OIDC Client Metadata

OIDC Client Information

Static configuration example from Google:

- https://developers.google.com/identity/protocols/OpenIDConnect

Dynamic registration:

- https://openid.net/specs/openid-connect-registration-1_0.html#ClientMetadata

There's no direct equivalent to SAML SP Metadata (or EntityDescriptor/EntitiesDescriptor) on OIDC

- Similar federation metadata signing logic cannot be directly applied to OIDC
- See OIDC Federation draft
  - https://openid.net/specs/openid-connect-federation-1_0.html
  - Applies to OP discovery + dynamic client registration phases - different from the SAML R&E model

Trust relationship establishment between Shibboleth OP and test RP

OIDC Client Information

https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki/MetadataConfiguration

Exercises

Exercise 2.3 - Add a trusted RP

- Try OIDC flow with an RP that is not (yet) trusted
  - The OIDC sequence can be started with https://IP_ADDRESS:8443/protected/ endpoint - You should end up into an error screen at Shibboleth OP. The logs should show that the client is not trusted.

Hints, Tips and Result

```
            [vagrant@gn43-oidcshibop-devel ~]$ tail -f /opt/shibboleth-idp/logs/idp-process.log
2018-12-05 11:08:32,258 - 81.197.147.73 - WARN [org.geant.idpextension.oidc.profile.impl.
OIDCMetadataLookupHandler:122] - Message Handler:  No client information returned for test_rp
2018-12-05 11:08:32,264 - 81.197.147.73 - DEBUG [org.geant.idpextension.oidc.profile.impl.
InitializeRelyingPartyContext:170] - Attaching RelyingPartyContext for rp test_rp
2018-12-05 11:08:32,275 - 81.197.147.73 - WARN [net.shibboleth.idp.profile.impl.
SelectProfileConfiguration:117] - Profile Action SelectProfileConfiguration: Profile http://csc.
fi/ns/profiles/oidc/sso/browser is not available for RP configuration shibboleth.
UnverifiedRelyingParty (RPID test_rp)
2018-12-05 11:08:32,302 - 81.197.147.73 - DEBUG [org.geant.idpextension.oidc.profile.impl.
AbstractBuildErrorResponseFromEvent:123] - Profile Action
BuildAuthenticationErrorResponseFromEvent: No mapped event found for
InvalidProfileConfiguration, creating general invalid_request
2018-12-05 11:08:32,305 - 81.197.147.73 - DEBUG [org.geant.idpextension.oidc.profile.impl.
AbstractBuildErrorResponseFromEvent:133] - Profile Action
BuildAuthenticationErrorResponseFromEvent: Error response not formed
```

- Statically add an RP as trusted (RP config already done)

- Add a file metadaresolver by enabling the commented out example from ***/opt/shibboleth-idp/conf/oidc-metadata-providers.xml***

Hints, Tips and Result

```
                <util:list id="shibboleth.oidc.ClientInformationResolvers"
        value-type="org.geant.idpextension.oidc.metadata.resolver.ClientInformationResolver">
        <ref bean="ExampleFileResolver" />
        <ref bean="ExampleStorageClientInformationResolver" />
    </util:list>

    <bean id="ExampleFileResolver"
        class="org.geant.idpextension.oidc.metadata.impl.FilesystemClientInformationResolver"
        p:id="ExampleFileResolver1"
        p:remoteJwkSetCache-ref="shibboleth.oidc.RemoteJwkSetCache">
        <constructor-arg>
            <bean class="java.io.File" id="ExampleFile">
                <constructor-arg type="String" value="/opt/shibboleth-idp/metadata/oidc-client.
json" />
            </bean>
        </constructor-arg>
    </bean>
```

- Add client metadata file having one dummy client, ***/opt/shibboleth-idp/metadata/oidc-client.json***

```
            [
  {
    "scope":"openid phone",
    "redirect_uris":["https://demorp.example.com/redirect_uri"],
    "client_id":"demo_rp",
    "client_secret":"topsecret",
    "response_types":["code"],
    "grant_types":["authorization_code","implicit","refresh_token"]
  }
]
```

- Check the RP-side configuration first from ***/etc/httpd/conf.d/auth_openidc.conf***. RP has been preconfigured for these training exercises.

Hints, Tips and Result

```
            OIDCClientID test_rp
OIDCClientSecret testSecret1234
OIDCProviderMetadataURL https://IP_ADDRESS/.well-known/openid-configuration
OIDCProviderIssuer https://IP_ADDRESS
OIDCOAuthSSLValidateServer Off
OIDCSSLValidateServer Off
OIDCRedirectURI https://IP_ADDRESS:8443/protected/redirect_uri
OIDCCryptoPassphrase secret
OIDCResponseType "code"
OIDCScope "openid profile email address phone"
```

- Pick up the settings for *client_id*, *client_secret*, *redirect_uris* and *scope* from the RP configuration. Also add "*response_types": ["id_token","code"]* and *"grant_types":["authorization_code","implicit","refresh_token"] claims and* apply them to ***/opt/shibboleth-idp /metadata/oidc-client.json***. Either replace the existing configuration for client *demo_rp*, or add an additional configuration, as instructed by our wiki (MetadataConfiguration).

  Hints, Tips and Result

  ```
  [root@gn43-oidcshibop-devel vagrant]# cat /opt/shibboleth-idp/metadata/oidc-client.
  json
  [
    {
      "scope":"openid info profile email address phone",
      "redirect_uris":["https://IP_ADDRESS:8443/protected/redirect_uri"],
      "client_id":"test_rp",
      "client_secret":"testSecret1234",
      "response_types":["id_token","code"],
      "grant_types":["authorization_code","implicit","refresh_token"]
    }
  ]
  ```

- Reload the OIDC client metadata by reloading service **shibboleth.ClientInformationResolverService**.

  Hints, Tips and Result

  ```
  [root@gn43-oidcshibop-devel shibboleth-idp]# /opt/shibboleth-idp/bin/reload-service.
  sh -id shibboleth.ClientInformationResolverService
  ```

- You should now get the login screen when starting the login sequence at: https://IP_ADDRESS:8443/protected/. Use username **teppo** with password **testaaja**. Check the logs during the login sequence.

Configuring Authentication

Section Topics

- OIDC Authentication Request
- Authentication flow selection for OIDC authentication request

OIDC Authentication Request

Authentication Request

http://openid.net/specs/openid-connect-core-1_0.html#AuthRequest

http://openid.net/specs/openid-connect-core-1_0.html#IndividualClaimsRequests

- Requested Authentication Context Class Reference values are defined by acr_values request parameter and acr claims request.
  - Request may be essential or non-essential.
- prompt has options
  - none, The Authorization Server MUST NOT display any authentication or consent user interface pages. Translates to isPassive.
  - login, The Authorization Server SHOULD prompt the End-User for reauthentication. Translates to forceAuthn.
  - consent, The Authorization Server SHOULD prompt the End-User for consent before returning information to the Client
  - select_account, The Authorization Server SHOULD prompt the End-User to select a user account. Not supported by current implementation.
- max_age, specifies the allowable elapsed time in seconds since the last time the End-User was actively authenticated by the OP.

Authentication flow selection for OIDC authentication request

OIDC Extension Authentication Configuration

https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki/AuthenticationConfiguration

Exercises

Exercise 3.1 - ACR value

ACR value

1. RP does not request for ACR by default. Modify the RP to request for password authentication.

```
        nano +418 /etc/httpd/conf.d/auth_openidc.conf

OIDCAuthRequestParams acr_values=password

service httpd restart
```

2. Run authentication sequence and verify from the logs that password authentication is being requested

Hints, Tips and Result

```
        Parameters:
   scope:openid profile email address phone
   acr_values:password
   response_type:code
   state:UeWkLZsv4qfSh5kmzN21TsHqj0E
   redirect_uri:https://195.148.31.24:8443/protected/redirect_uri
   nonce:dC3KenQHX-8R1eSXrkNQajljL0NuxpafDOiAZ-5vHnk
   client_id:test_rp
```

3. OP does not seem to respond with ACR claim value - there is no [OIDC_CLAIM_acr] on your landing page. The authentication method principals are set in ***/opt/shibboleth-idp/conf/authn/general-authn.xml***. See https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki/AuthenticationConfiguration on how to add "password" authentication method principal for password flow.

4. Run authentication sequence and verify the use of ACR claim.
    a. Verify the ACR claim value from your landing page.

Hints, Tips and Result

```
            [OIDC_CLAIM_acr] => password
```

    b. The value is set in action AddAcrToIdToken. Verify from the logs the action has taken place.

Hints, Tips and Result

```
            grep AddAcrToIDToken /opt/shibboleth-idp/logs/idp-process.log

2018-09-10 04:52:49,068 - DEBUG [org.geant.idpextension.oidc.profile.impl.AddAcrToIDToken:58] -
Profile Action AddAcrToIDToken: Setting acr to id token
2018-09-10 04:52:49,079 - DEBUG [org.geant.idpextension.oidc.profile.impl.AddAcrToIDToken:60] -
Profile Action AddAcrToIDToken: Updated token {"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","aud":
["_443085776b9c4370eeb8b7481b99dbe3"],"acr":"password","auth_time":1536555148,"iss":"https:\/\
/192.168.0.150","exp":1536558769,"iat":1536555169}
```

    c. Client receives the ACR claim in ID Token. Locate ID Token from Token Response (field "id_token") in the logs. Decode the ID Token and verify the existence and value of ACR claim. One option for you to decode the ID Token is http://calebb.net/

Hints, Tips and Result

```
            grep id_token /opt/shibboleth-idp/logs/idp-process.log

Content:{"access_token":"AAdzZWNyZXQxV9G7fr...tJ9","id_token":"
eyJraWQiOiJ0ZXN0a2V5UlMiLCJhbGciOiJSUzI1NiJ9.
eyJhdF9oYXNoIjoiaW9wTFlpbjNSYmVmUmxJamdjWkNQQSIsInN1YiI6IlZVRzQ3NzdZUDNOTVU1S1JGRVNYNlNLUkFQWExF
NE1JIiwiYXVkIjoiXzQ0MzA4NTc3NmI5YzQzNzBlZWI4Yjc0ODFiOTlkYmUzIiwiYWNyIjoicGFzc3dvcmQiLCJhdXRoX3Rp
bWUiOjE1MzY1NTUxNDgsImlzcyI6Imh0dHBzOlwvXC8xOTIuMTY4LjAuMTUwIiwiZXhwIjoxNTM2NTU4NzY5LCJpYXQiOjE1
MzY1NTUxNjksIm5vbmNlIjoiUmZPNjhnSmVHbEoyb0xaOU9XamVCWWJidlNiNVI1U3Mza3ZwM1daSTVOQSJ9.
kIF8PKmVVEULbas6gsaiOchO51W_614V6WvXMfIraW2RosTAmSoyFVjCT_RxOp5RRHQVbpHVVb9Q9lQcwE0ZjnyTU4VCP2ZQ
N7EqLHP8gTLQ_SIhooQ8cB5hP_w-ijd4ZPw2tSSDk44X2mdfOJv0W0TftwxRFkprp8pleYqz1YIMh0DH3lpvU8AbsmZ-
K7ehJYegln35rWzz9Ve7tqFTBlAB0Gj4sYGCcR6oAKPrMmo9LiYrpe92viFfAGiRZFthz9Mjd1THZOLWNd62tpJJLgm6RqDD
PMolM2NrPahcYvDHakErM9S5Xp0hHoyn6J4ihArDGqWNrgI3y9_6GJu35g","token_type":"Bearer","expires_in":
600}

{
 kid: "testkeyRS",
 alg: "RS256"
}.{
 at_hash: "iopLYin3RbefRlIjgcZCPA",
 sub: "VUG4777YP3NMU5KRFESX6SKRAPXLE4MI",
 aud: "_443085776b9c4370eeb8b7481b99dbe3",
 acr: "password",
 auth_time: 1536555148,
 iss: "https:\/\/192.168.0.150",
 exp: 1536558769,
 iat: 1536555169,
 nonce: "RfO68gJeGlJ2oLZ9OWjeBYbbvSb5R5Ss3kvp3WZI5NA"
}.[signature]
```

Exercise 3.2 - Flow selection

Select flow by ACR value

1. Modify client to request for ACR value "ipaddress".

```
          nano +418 /etc/httpd/conf.d/auth_openidc.conf

OIDCAuthRequestParams acr_values=ipaddress

service httpd restart
```

2. Run authentication sequence.
3. Follow log entries, identify requested ACR and what is actually sent back as ACR in the response. Can you explain why they do not match?

Hints, Tips and Result

```
          Request:
Parameters:
    scope:openid email
    acr_values:ipaddress
    response_type:code
    state:PunYlKqqJA6b57C9DPKbt5cvdq8
    redirect_uri:https://192.168.0.150:8443/protected/redirect_uri
    nonce:4kcflIwolApMDMMfKxYte9IvU5aIULIWc38dW_XG5lc
    client_id:_443085776b9c4370eeb8b7481b99dbe3

2018-09-10 07:24:54,250 - DEBUG [org.geant.idpextension.oidc.profile.impl.ProcessRequestedAuthnContext:
184] - Profile Action ProcessRequestedAuthnContext: Created preferred principal context

2018-09-10 07:25:00,905 - DEBUG [org.geant.idpextension.oidc.profile.impl.AddAcrToIDToken:60] -
Profile Action AddAcrToIDToken: Updated token {"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","aud":
["_443085776b9c4370eeb8b7481b99dbe3"],"acr":"password","auth_time":1536564300,"iss":"https:\/\/192.
168.0.150","exp":1536567900,"iat":1536564300}
```

4. Activate IPAddress authentication, configure it for acr "ipaddress" and verify it is actually used as preferred authentication flow.

Hints, Tips and Result

```
          nano /opt/shibboleth-idp/conf/authn/ipaddress-authn-config.xml

#Modify the existing IPAddress authentication mapping
<util:map id="shibboleth.authn.IPAddress.Mappings">
    <entry key="teppo">
        <list>
            <value>0.0.0.0/0</value>
            <value>::1/128</value>
        </list>
    </entry>
</util:map>


nano /opt/shibboleth-idp/conf/authn/general-authn.xml

#Modify the existing IPAddress authentication
<bean id="authn/IPAddress" parent="shibboleth.AuthenticationFlow"
        p:passiveAuthenticationSupported="true"
        p:lifetime="PT60S" p:inactivityTimeout="PT60S">
    <property name="supportedPrincipals">
        <list>
            <bean parent="shibboleth.SAML2AuthnContextClassRef"
                c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocol" />
            <bean parent="shibboleth.OIDCAuthnContextClassReference"
                c:classRef="ipaddress" />
        </list>
    </property>
</bean>


nano +121 /opt/shibboleth-idp/conf/idp.properties

#Add IPAddress as active authentication option
idp.authn.flows=IPAddress|Password

After restart and performing new authentication sequence you should see:

2018-09-10 15:15:01,007 - DEBUG [org.geant.idpextension.oidc.profile.impl.AddAcrToIDToken:60] -
Profile Action AddAcrToIDToken: Updated token {"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","aud":
["_443085776b9c4370eeb8b7481b99dbe3"],"acr":"ipaddress","auth_time":1536592497,"iss":"https:\/\/192.
168.0.150","exp":1536596100,"iat":1536592500}
```

5. Create essential claims request for ACR values urn:mace:incommon:iap:silver and urn:mace:incommon:iap:bronze. This is now request that must (but will not) be met.

```
         nano +418 /etc/httpd/conf.d/auth_openidc.conf

OIDCAuthRequestParams claims=%7B%22id_token%22%3A%7B%22acr%22%3A%20%7B%22essential%22%3A%20true%2C%
22values%22%3A%20%5B%22urn%3Amace%3Aincommon%3Aiap%3Asilver%22%2C%22urn%3Amace%3Aincommon%3Aiap%
3Abronze%22%5D%7D%7D%7D

service httpd restart
```

Start a new authentication sequence and see the result. This behavior should be something you are used to in SAML2 world.

Hints, Tips and Result

```
         2018-09-10 15:30:58,586 - DEBUG [net.shibboleth.idp.authn.impl.SelectAuthenticationFlow:395]
- Profile Action SelectAuthenticationFlow: Specific principals requested with 'exact' operator:
[AuthenticationContextClassReferencePrincipal{authnContextClassReference=urn:mace:incommon:iap:
silver}, AuthenticationContextClassReferencePrincipal{authnContextClassReference=urn:mace:incommon:iap:
bronze}]
2018-09-10 15:30:58,586 - DEBUG [net.shibboleth.idp.authn.impl.SelectAuthenticationFlow:411] - Profile
Action SelectAuthenticationFlow: No active results available, selecting an inactive flow
2018-09-10 15:30:58,586 - DEBUG [net.shibboleth.idp.authn.impl.SelectAuthenticationFlow:432] - Profile
Action SelectAuthenticationFlow: Checking for an inactive flow compatible with operator 'exact' and
principal 'urn:mace:incommon:iap:silver'
2018-09-10 15:30:58,587 - DEBUG [net.shibboleth.idp.authn.impl.SelectAuthenticationFlow:432] - Profile
Action SelectAuthenticationFlow: Checking for an inactive flow compatible with operator 'exact' and
principal 'urn:mace:incommon:iap:bronze'
2018-09-10 15:30:58,588 - INFO [net.shibboleth.idp.authn.impl.SelectAuthenticationFlow:453] - Profile
Action SelectAuthenticationFlow: None of the potential authentication flows can satisfy the request
2018-09-10 15:30:58,650 - DEBUG [org.geant.idpextension.oidc.profile.impl.
AbstractBuildErrorResponseFromEvent:123] - Profile Action BuildAuthenticationErrorResponseFromEvent:
No mapped event found for RequestUnsupported, creating general invalid_request
2018-09-10 15:30:58,651 - DEBUG [org.geant.idpextension.oidc.profile.impl.
AbstractBuildErrorResponseFromEvent:131] - Profile Action BuildAuthenticationErrorResponseFromEvent:
ErrorResponse successfully set as the outbound message
2018-09-10 15:30:58,652 - DEBUG [org.geant.idpextension.oidc.encoding.impl.NimbusResponseEncoder:148]
- Outbound response
Headers:
    Location:https://192.168.0.150:8443/protected/redirect_uri?error_description=Invalid+request%
3A+RequestUnsupported&state=T_2Ez6onqjVzJFT8T9d3zjzkpns&error=invalid_request
```

6. Let's return to the original state

```
         nano +418 /etc/httpd/conf.d/auth_openidc.conf

#OIDCAuthRequestParams claims=%7B%22id_token%22%3A%7B%22acr%22%3A%20%7B%22essential%22%3A%20true%2C%
22values%22%3A%20%5B%22urn%3Amace%3Aincommon%3Aiap%3Asilver%22%2C%22urn%3Amace%3Aincommon%3Aiap%
3Abronze%22%5D%7D%7D%7D

service httpd restart

nano +121 /opt/shibboleth-idp/conf/idp.properties

#Remove IPAddress as active authentication option
idp.authn.flows=Password

systemctl restart shibboleth-idp
```

Exercise 3.3 - Prompt

prompt=consent and other prompt parameters

1. Set the consent to be requested

```
        nano +417 /etc/httpd/conf.d/auth_openidc.conf

OIDCAuthRequestParams prompt=consent

service httpd restart
```

2. Authenticate the user few times and verify this feature actually works as expected.
3. Try different combinations of the parameter

Attribute Definitions

Section Topics

- OIDC user attributes possible locations
- OIDC Attribute resolving principle, authorization, token and userinfo endpoints
- OIDC attribute encoders

OIDC user attributes possible locations

User attributes may be returned either in ID Token http://openid.net/specs/openid-connect-core-1_0.html#IDToken or in UserInfo response http://openid.net/specs/openid-connect-core-1_0.html#UserInfoResponse.

ID Token - Depending on response type is returned from Authentication or from Token endpoint.

```
    {
 kid: "1e9gdk7",
 alg: "RS256"
}.
{
 iss: "http://server.example.com",
 sub: "248289761001",
 aud: "s6BhdRkqt3",
 nonce: "n-0S6_WzA2Mj",
 exp: 1311281970,
 iat: 1311280970,
 name: "Jane Doe",
 given_name: "Jane",
 family_name: "Doe",
 gender: "female",
 birthdate: "0000-10-31",
 email: "janedoe@example.com",
 picture: "http://example.com/janedoe/me.jpg"
}.
[signature]
```

UserInfo response - Always from UserInfo endpoint.

```
   {
  "sub": "248289761001",
  "name": "Jane Doe",
  "given_name": "Jane",
  "family_name": "Doe",
  "preferred_username": "j.doe",
  "email": "janedoe@example.com",
  "picture": "http://example.com/janedoe/me.jpg"
 }
```

Whether the attribute ends up to the ID Token or UserInfo response depends on authentication request.

- If claim is requested by standard scope parameter, attribute will be set to userinfo response unless response type is "id_token" (i.e. UserInfo endpoint cannot be accessed).
- Client may specifically request claim to be returned either in ID Token or in UserInfo response
- Third option is of course some agreement between RP and OP about non standard scope or some out of band agreement.

> How clients request for claims is handled in more detail in section about attribute filtering

OIDC Attribute resolving principle, authorization, token and userinfo endpoints.

- Authentication endpoint. Front-channel endpoint responsible of authenticating the user are asking for consent. May return depending on response type ID Token, authorization code or Access Token or any combination of the three items.
- Token endpoint. Back-channel endpoint that is responsible of authenticating the client RP. Swaps authorization code(or Refresh Token) to Access Token, ID Token and to Refresh Token.
- Userinfo endpoint. Back-channel endpoint that can be accessed with Access Token returning UserInfo response.

> **Attribute Resolving principle**
>
> Each of the endpoints perform attribute resolving!
>
> Back-channel endpoints do not have all session/context information available!
>
> Attributes that are based on session/context information must be instructed to be carried in tokens (i.e. encoded directly to authorization code and Access Token) unless implicit flow is used.
>
> Attribute resolvers that use session/context information must check the existence for the source information and fail gracefully if checking fails

Resolver example

```
          <AttributeDefinition id="password" xsi:type="ScriptedAttribute" dependencyOnly="true"
language="nashorn">
        <Script><![CDATA[
            logger = Java.type("org.slf4j.LoggerFactory").getLogger("net.shibboleth.idp.script.
password");
            <!-- Subject does not have credentials populated in Token and UserInfo endpoints, have to
give up in such case -->
            subjectCtx = profileContext.getSubcontext("net.shibboleth.idp.authn.context.SubjectContext");
            subject = subjectCtx.getSubjects()[0];
            if (!subject.getPrivateCredentials().isEmpty()){
                password.addValue(subject.getPrivateCredentials().toArray()[0].getName());
            }
    ]]></Script>
    </AttributeDefinition>
```

OIDC attribute encoders

- oidcext:OIDCString, for encoding an IdPAttribute with simple string values as JSON Object.
- oidcext:OIDCScopedString, for encoding an IdPAttribute with scoped string values as JSON Object.
- oidcext:OIDCByte, for encoding an IdPAttribute with binary values as JSON Object.

Encoder formatting options

Default

```
    <AttributeEncoder xsi:type="oidcext:OIDCString" name="affiliation" />

Input: IdPAttribute["member", "staff"]
Output: "affiliation": "member staff"
```

asArray

```
    <AttributeEncoder xsi:type="oidcext:OIDCString" asArray="true" name="affiliation" />

Input: IdPAttribute["member", "staff"]
Output: "affiliation":["member","staff"]
```

asInt

```
    <AttributeEncoder xsi:type="oidcext:OIDCString" asInt="true" name="updated_at" />

Input: IdPAttribute["1536143427"]
Output: "updated_at": 1536143427
```

asBoolean

```
    <AttributeEncoder xsi:type="oidcext:OIDCString" asBoolean="true" name="email_verified" />

Input: IdPAttribute["true"]
Output: "email_verified": true
```

asObject

```
    <AttributeDefinition id="address" xsi:type="ScriptedAttribute">
    <Dependency ref="staticAttributes" />
    <Script><![CDATA[address.addValue("{\"street_address\":\""+street_address.getValues().get(0) + "\","
        +"\"locality\":\""+locality.getValues().get(0) + "\","
        +"\"region\":\""+region.getValues().get(0) + "\","
        +"\"postal_code\":\""+postal_code.getValues().get(0) + "\","
        +"\"country\":\""+country.getValues().get(0) + "\"}");]]></Script>
    <AttributeEncoder xsi:type="oidcext:OIDCString" asObject="true" name="address" />
</AttributeDefinition>


Output: "address":{"street_address":"234 Hollywood Blvd.","country":"US","locality":"Los Angeles","region":"
CA","postal_code":"90210"}
```

Encoder delivery options

placeToIDToken & denyUserinfo

```
    <!-- This demonstrates a claim that is placed always to id token -->
<AttributeDefinition id="email_idtoken" xsi:type="Simple" sourceAttributeID="email">
    <Dependency ref="email" />
    <AttributeEncoder xsi:type="oidcext:OIDCString" placeToIDToken="true" denyUserinfo="true" name="email" />
</AttributeDefinition>
```

setToToken

Attribute not resolvable in token and UserInfo endpoints must be carried in tokens if necessary.

Resolver example

```
         <AttributeDefinition id="password" xsi:type="ScriptedAttribute" dependencyOnly="true" language="
nashorn">
        <Script><![CDATA[
            logger = Java.type("org.slf4j.LoggerFactory").getLogger("net.shibboleth.idp.script.password");
            subjectCtx = profileContext.getSubcontext("net.shibboleth.idp.authn.context.SubjectContext");
            <!-- Subject does not have credentials populated in Token and UserInfo endpoints, have to give up
in such case -->
            if (subjectCtx!=null){
                subject = subjectCtx.getSubjects()[0];
                password.addValue(subject.getPrivateCredentials().toArray()[0].getName());
            }
        ]]></Script>
     <AttributeEncoder xsi:type="oidcext:OIDCString" setToToken="true" name="password"/>
    </AttributeDefinition>
```

Exercises

Exercise 4.1

Available contexts in Endpoints

1. attribute-resolver.xml has a scripted attribute "scriptedAuthenticationFlowId" commented out. Activate the attribute resolving.
2. Authenticate user using with this new configuration. We are not creating release rules for the attribute, yet. See log for any errors related to the attribute.

Hints, Tips and Result

```
          2018-09-20 09:50:48,070 - ERROR [net.shibboleth.idp.profile.impl.ResolveAttributes:314] -
Profile Action ResolveAttributes: Error resolving attributes
net.shibboleth.idp.attribute.resolver.ResolutionException: Attribute Definition
'scriptedAuthenticationFlowId':Script did not run successfully
    at net.shibboleth.idp.attribute.resolver.ad.impl.
ScriptedAttributeDefinition$AttributeDefinitionScriptEvaluator.execute(ScriptedAttributeDefinition.
java:228)
Caused by: java.lang.RuntimeException: javax.script.ScriptException: TypeError: null has no such
function "getAuthenticationResult" in <eval> at line number 3
    at net.shibboleth.utilities.java.support.scripting.AbstractScriptEvaluator.evaluate
(AbstractScriptEvaluator.java:193)
Caused by: javax.script.ScriptException: TypeError: null has no such function
"getAuthenticationResult" in <eval> at line number 3
    at jdk.nashorn.api.scripting.NashornScriptEngine.throwAsScriptException(NashornScriptEngine.java:
470)
Caused by: jdk.nashorn.internal.runtime.ECMAException: TypeError: null has no such function
"getAuthenticationResult"
```

3. Change the response type to "id_token". It means we are using only authentication endpoint.

```
          nano +632 /etc/httpd/conf.d/auth_openidc.conf

OIDCResponseType "id_token"

service httpd restart
```

4. Authenticate user using with this new configuration. Notice there are not errors related to resolving the attribute. Can you explain why?
5. Change the response type back to "code". Correct the attribute to not cause errors. Correct also the attribute to be carried in tokens.

Hints, Tips and Result

```
          <AttributeDefinition id="scriptedAuthenticationFlowId" xsi:type="ScriptedAttribute"
language="nashorn">
    <Script><![CDATA[
    logger = Java.type("org.slf4j.LoggerFactory").getLogger("net.shibboleth.idp.script.password");
    authnCtx = profileContext.getSubcontext("net.shibboleth.idp.authn.context.AuthenticationContext");
    if (authnCtx != null){
        scriptedAuthenticationFlowId.addValue(authnCtx.getAuthenticationResult().
getAuthenticationFlowId());
    }
    ]]></Script>
    <AttributeEncoder xsi:type="oidcext:OIDCString" setToToken="true" name="flow_id"/>
</AttributeDefinition>
```

Exercise 4.2

Formatting attribute

1. Authenticate the user. Locate "manipe" claim from the UserInfo response.

   Hints, Tips and Result

   ```
            Headers:
       Content-Type:application/json; charset=UTF-8
   Content:{"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","zoneinfo":"America\/Los_Angeles","website":"https:\
   /\/www.facebook.com\/officialtomcruise\/","birthdate":"1962","address":{"street_address":"234
   Hollywood Blvd.","country":"US","locality":"Los Angeles","region":"CA","postal_code":"90210"},"
   email_verified":false,"gender":"male","profile":"https:\/\/fi.wikipedia.org\/wiki\/Tom_Cruise","
   phone_number_verified":true,"preferred_username":"ttester","locale":"en-US","given_name":"Teppo
   Matias","middle_name":"Matias","manipe":"zero 1 3 two","picture":"https:\/\/pixabay.com\/fi\/pentu-
   kissa-kukka-potin-tabby-pentu-2766820\/","updated_at":1509450347,"nickname":"TT","name":"Mr.Teppo
   Matias Testaaja","phone_number":"+1 (604) 555-1234;ext=5678","family_name":"Testaaja","email":"
   teppo@example.org"}
   ```

2. Instruct "manipe" to be encoded as an array. Verify the result from the UserInfo response.

   Hints, Tips and Result

   ```
            Headers:
       Content-Type:application/json; charset=UTF-8
   Content:{"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","zoneinfo":"America\/Los_Angeles","website":"https:\
   /\/www.facebook.com\/officialtomcruise\/","birthdate":"1962","address":{"street_address":"234
   Hollywood Blvd.","country":"US","locality":"Los Angeles","region":"CA","postal_code":"90210"},"
   email_verified":false,"gender":"male","profile":"https:\/\/fi.wikipedia.org\/wiki\/Tom_Cruise","
   phone_number_verified":true,"preferred_username":"ttester","locale":"en-US","given_name":"Teppo
   Matias","middle_name":"Matias","manipe":["zero","1","3","two"],"picture":"https:\/\/pixabay.com\/fi\
   /pentu-kissa-kukka-potin-tabby-pentu-2766820\/","updated_at":1509450347,"nickname":"TT","name":"Mr.
   Teppo Matias Testaaja","phone_number":"+1 (604) 555-1234;ext=5678","family_name":"Testaaja","email":"
   teppo@example.org"}
   ```

3. Instruct "manipe" to be encoded as an array of integers. Verify the result from the UserInfo response.

   Hints, Tips and Result

   ```
            Headers:
       Content-Type:application/json; charset=UTF-8
   Content:{"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","zoneinfo":"America\/Los_Angeles","website":"https:\
   /\/www.facebook.com\/officialtomcruise\/","birthdate":"1962","address":{"street_address":"234
   Hollywood Blvd.","country":"US","locality":"Los Angeles","region":"CA","postal_code":"90210"},"
   email_verified":false,"gender":"male","profile":"https:\/\/fi.wikipedia.org\/wiki\/Tom_Cruise","
   phone_number_verified":true,"preferred_username":"ttester","locale":"en-US","given_name":"Teppo
   Matias","middle_name":"Matias","manipe":[1,3],"picture":"https:\/\/pixabay.com\/fi\/pentu-kissa-kukka-
   potin-tabby-pentu-2766820\/","updated_at":1509450347,"nickname":"TT","name":"Mr.Teppo Matias
   Testaaja","phone_number":"+1 (604) 555-1234;ext=5678","family_name":"Testaaja","email":"teppo@example.
   org"}
   ```

Attribute Filtering

Section Topics

- Requesting Attributes in OIDC
- Filtering attributes for OIDC RPs

Requesting Attributes in OIDC

There are two main mechanisms for RP to ask for attributes,

- Authentication request scope - parameter
  - There is a set of standard scope values profile, email, address and phone.
  - The claims requested by the standard scope values are expected to be returned from UserInfo endpoint unless response type is "id_token"
  - The claims requested by the standard scope values are treated as voluntary
  - You may define your own scope values and rules for them. This has to be agreed by both parties of course.
- Authentication request claims - parameter
  - You must declare the target for the claim, ID Token or UserInfo response
  - You may list the claim as essential claim
  - You may list a expected value for the claim

Claims Request

```
 {
"userinfo":
 {
  "given_name": {"essential": true},
  "nickname": null,
  "picture": null,
  "http://example.info/claims/groups": null
 },
"id_token":
 {
  "email": {"essential": true},
  "email_verified": {"essential": true},
  "sub": {"value": "248289761001"}
  "auth_time": {"essential": true},
  "acr": {"essential": true,
          "values": ["urn:mace:incommon:iap:silver",
                     "urn:mace:incommon:iap:bronze"]}
 }
}
```

Filtering attributes for OIDC RPs

It is your job to resolve and filter attributes to match the requests. OIDC extension has reserved the right to control some claims you should not try to resolve or filter.

oidcext:OIDCScope

PolicyRule which returns true if any of the scope values in the authentication request matches a supplied string.

OIDCScope

```
    <!-- This demonstrates a rule that releases email claims in response to all oidc authentication requests
having scope
        email. The requester needs to have scope email as a registered scope. -->

<AttributeFilterPolicy id="OPENID_SCOPE_EMAIL">
    <PolicyRequirementRule xsi:type="oidcext:OIDCScope" value="email" />
    <AttributeRule attributeID="email">
        <PermitValueRule xsi:type="ANY" />
    </AttributeRule>
    <AttributeRule attributeID="email_verified">
        <PermitValueRule xsi:type="ANY" />
    </AttributeRule>
</AttributeFilterPolicy>
```

oidcext:AttributeInOIDCRequestedClaims

Matcher which returns attribute values after comparing them to requested claims parameter of oidc authentication request.

AttributeInOIDCRequestedClaims

```
    <!-- This demonstrates a rule that releases email in id token if specifically asked to be released as
essential for id token -->

<AttributeFilterPolicy id="REQUESTED_CLAIMS">
    <PolicyRequirementRule xsi:type="ANY" />
    <AttributeRule attributeID="email_idtoken">
        <PermitValueRule xsi:type="oidcext:AttributeInOIDCRequestedClaims" matchOnlyIDToken="true"
onlyIfEssential="true" />
    </AttributeRule>
</AttributeFilterPolicy>
```

Exercises

Exercise 5.1

Define a new scope "campus". Target is to have scope "campus" that includes a claim "campus_id" to ID Token.

1. Locate "campusId" attribute from attribute resolver. Add a new filtering rules that releases campusId when scope "campus" is requested.

    Hints, Tips and Result

    ```
            <AttributeFilterPolicy id="OPENID_SCOPE_CAMPUS">
        <PolicyRequirementRule xsi:type="oidcext:OIDCScope" value="campus" />
        <AttributeRule attributeID="campusId">
            <PermitValueRule xsi:type="ANY" />
        </AttributeRule>
    </AttributeFilterPolicy>
    ```

2. Modify client RP to request for scope "campus".

```
          nano +633 /etc/httpd/conf.d/auth_openidc.conf

OIDCScope "openid campus"

service httpd restart
```

3. Authenticate the user. Verify from the logs that scope "campus" is being requested. Find out from the logs why it is not being released.

Hints, Tips and Result

```
          [root@gn43-oidcshibop-devel conf]# grep campus  /opt/shibboleth-idp/logs/idp-process.log
2018-09-24 05:27:27,048 - INFO [net.shibboleth.idp.attribute.resolver.spring.BaseResolverPluginParser:
63] - Parsing configuration for AttributeDefinition plugin with id: campusId
    scope:openid campus
2018-09-24 05:40:41,506 - DEBUG [org.geant.idpextension.oidc.decoding.impl.
OIDCAuthenticationRequestDecoder:69] - Decoded inbound request query string login_hint=teppo%
40192.168.0.150&scope=openid+campus&claims=%7B%22id_token%22%3A%7B%22acr%22%3A%7B%22values%22%3A%5B%
22urn%3Amace%3Aincommon%3Aiap%3Asilver%22%2C%22urn%3Amace%3Aincommon%3Aiap%3Abronze%22%5D%2C%
22essential%22%3Atrue%7D%7D%7D&response_type=code&redirect_uri=https%3A%2F%2F192.168.0.150%3A8443%
2Fprotected%
2Fredirect_uri&state=GfF7nJQf2lEQZIvFOj9UZMCxLYY&nonce=7XWKLlplp2TNt5gHCHS4QGg9XgglPzXGSMNcXyRT1IM&clie
nt_id=_6abd8ce1be06caf6c65b79934dfd5a01
2018-09-24 05:40:41,768 - WARN [org.geant.idpextension.oidc.profile.impl.ValidateScope:93] - Profile
Action ValidateScope: removing requested scope campus for rp _6abd8ce1be06caf6c65b79934dfd5a01 as it
is not a registered one
```

4. Add scope "campus" for the RP as a valid scope value.

5. Authenticate the user. Verify from the logs that scope "campus" is being requested. Find out from the logs is it being released to ID Token or to UserInfo response.

Hints, Tips and Result

```
          grep id_token /opt/shibboleth-idp/logs/idp-process.log

Content:{"access_token":"AAdzZWNyZXQx6kaWFa5m2rBYllmrWmypRG6o8acq7EsttE-kkEqpF_llEOeZ-RWze1B-
p18f3rVWUH4dC_DQP8OHkoYbJuCBHpKdEOSNsmjCvO7MElb_p5VeiiyUqtVAcYnjWe0CAUF0w_lGHKndleYiWjzO_EhGuvH-zwui-
HgvS65A3VNUYtfbUt6fxm-
yC8KoVBfN7TjWhIu6wV_lhUFZuNWpI4HMhwLhV4vqcpAyVwqPmUJzcDqOT2ltktMGImeZmDl3a_rAvfZhF2VZc6K9xdEcu6NEMbasbQ
gCutWJkvrTfkBgMspdGn2_YYapxOWImkMS4s0fYEEdcwMB8nxQRKMpP_JmiV5hIMMFiZSg4qa5J9N2t5oI_F-
vO2nG5Q7bBMyPmvLCcBcXPOVbP2fqZyu_hCcxBwYjnwURYlp1jN-
V9dtG_vdRa5Ddn3P0QcrJQDTQDrAsnkn51NB24b56CiQjj4kVFrc4qNpeDhQ7dgG3M19BKD1rdoFMaNTStN0_Br3YD2G0B4GDJX3I6U
JOM4ksmbqDb5aknDgSMxnJHDt96LQqoNmDNI-RugE3Ombit9ZAUaRuAXMAKvR6gXK5gfhr8LzqflF8k2DTVLPyvI1q","
refresh_token":"AAdzZWNyZXQxvJQv2UDhK1QGie4smtxaYDD9uwFMk09syFcROFj851zrNb1a7UXBSmp6VYJjYHsXyyIBe0-
BQ4_vA7WL-Yuc1u-rJ951w3uKytERn87s0ZGTv45mCVpvlisLpL_XAu6x6Jh7DpOCP7b1FZYH2ctWIYx-
Vo9QLya1b1Zys3fohU6Rg5O24ZvtTuUJugFPyDOdaCjO5tTP2eSVSEGrm8-
NFz_z0VLWdqwCCjhPKq3FKSXmDo6UX1wyxLtH0DVLoBrAJrZtHMrjsNI1jJQY6IeNl_yxN6Hl_0FNmhmbEgD8zMT0N9-
zlrno4w7NPHBQK5ytFCXixmXEK9xgjlf3uqoLQMVMuTuMvee6hQNgRzawcYU3R5jN9KtCNN_NR0v5n8-
R3QTlHlYn7Agdhm1UpEuKomXFCy9fo9Psq2haOj588Yt0YaHqp877bofP5SjaNVSixC7TECwAjc11yg2JhRSPGUHQwlRzFuVhUdpVgd
NZfSD70nrdAMUv_--XVEdzg9vnCNjPk7XE3jjVijvugaVZvUV2HO6RnhQRWCg12lfOROmBq0bUlaMPaC87zwBlEI9GjQOG_e1fw-
K3BfcC7WdQ9DVEtPuOrRh0qA0","id_token":"eyJraWQiOiJ0ZXN0a2V5UlMiLCJhbGciOiJSUzI1NiJ9.
eyJhdF9oYXNoIjoib1RBc3dNX2pxR0pncHo4WDkzRWNydyIsInN1YiI6IlZVRzQ3NzdZUDNOTVU1S1JGRVNYNlNLUkFQWExFNE1JIiw
iYXVkIjoiX2ZhYmJjNmY5MGExMmZmYjNmZWUwMDEwZDRjMmFiZGVlIiwiYWNyIjoidXJuOm1hY2U6aW5jb21tb246aWFwOnNpbHZlci
IsImF1dGhfdGltZSI6MTUzNzc3Mjc4NiwiaXNzIjoiaHR0cHM6XC9cLzE5Mi4xNjguMC4xNTAiLCJleHAiOjE1Mzc3NzYzODcsImlhd
CI6MTUzNzc3Mjc4Nywibm9uY2UiOiJ3NnJzdkh2d05UX0FuSVZ3OFh1UTlucG5iLXFBZ18yenEySTdtUEJMNkdNIn0.
Iqraic8ugywq2_saSoIGIBCZFJIArgUiVvnRQxyjemmZdUG7hxU7lJ06GogBGNmnKDwhOqiA-
Ck1SB2iVkvA1YbK3xGgCBO4kkYQSWGjQ1Mq1R-
3J0_OPZ5SU5pjcpo4YUBGqYpOjv2WDAftfpBgGr0HmXRjTd_CLxcm8AM5XMwokCJufqoP7l6fjAhxcKweQNPnGEASCG200wTNHIEA82
wvOHxZ9BrjB65kyM2lODL40T9NJ5UVvAGHzMzrcw2PUHrhnBGKLmGe89oSO0ww3KfxDiU7vg2jg36xnoMUVwN8RFad7-
fNzH3JwNEMzekQBuusDxrxAHlaVHntTZndrw","token_type":"Bearer","expires_in":600}

Decode id token, you will get something like:

{
 kid: "testkeyRS",
 alg: "RS256"
}.{
 at_hash: "oTAswM_jqGJgpz8X93Ecrw",
 sub: "VUG4777YP3NMU5KRFESX6SKRAPXLE4MI",
 aud: "_fabbc6f90a12ffb3fee0010d4c2abdee",
 acr: "password",
 auth_time: 1537772786,
 iss: "https:\/\/192.168.0.150",
 exp: 1537776387,
 iat: 1537772787,
 nonce: "w6rsvHvwNT_AnIVw8XuQ9npnb-qAg_2zq2I7mPBL6GM"
}.[signature]

Check now userinfo response. It is one of the last lines in the log now

tail /opt/shibboleth-idp/logs/idp-process.log

Content:{"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI","campus_id":"Y2FtcHVzSWQ="}

You notice the campus_id is only in the userinfo response.
```

6. Modify "campusId" attribute resolver to encode the claim always and only to ID Token.

Hints, Tips and Result

```
          <AttributeDefinition id="campusId" xsi:type="Simple" sourceAttributeID="campusId">
    <Dependency ref="staticAttributes" />
    <AttributeEncoder xsi:type="oidcext:OIDCString" name="campus_id" placeToIDToken="true"
denyUserinfo="true"/>
</AttributeDefinition>
```

7. Authenticate the user. Verify from the logs that scope "campus" is being requested. Verify from the logs it is being released only to ID Token.

Hints, Tips and Result

```
          grep id_token /opt/shibboleth-idp/logs/idp-process.log

Content:{"access_token":"AAdzZWNyZXQxX6FZjPLkXdPUi2UBqUNnkcNwayEbDbrsL_LHumwkbTpLM2W8QVEOGhHRsh-
sAtKeaIFx1568_AAESKeHT7qFSJYKOavGxj17wVBJ_S8mQGW7YcIIfL2hAC-LnHtvCgY8n_r-
WsD_HVJN3UqQ0GijbPeX4NHEENfBgG18oEO1bbSMHR0XhBCY0sLcubfaXl7N0E17COObm992Su5jFsvZptfgmYWgkV3OYzX4684bNL2
F_8-6w1TblGpNgXIi0ORci7FSXP0oMQfxb_AeAfADe3n-tl_4dUeJ-rrKP-oG-rUsnkEQQqHUgtBZEIV-
YmIYhA3UP1Ux35XpFYwrbMYSb7yZ2zeH-
fiukZGiStMxoifVmldzeKTZ8euO69lcZqv1WZGWueMmZ3ltfCWH3V5XI26KjYJgnxaX9FGVG4wRBm_yCclgtuplscs8Ixvmrr4HyaSQ
ZOGOS0d8Ri4q6KnKqaob4zCAX2GGkBCWCtYQpTKfotLWWwvNorxhPlyVoorOLDKe2eTl3en4H9RGl_xvaYZTIMyu0bB-
eUJgeXyskz_TEjDpnnBoSSBSO2daBNZPnsQUaLnz5E0RK_C7it73hc2TWrNXWkA7pA","refresh_token":"
AAdzZWNyZXQxjINhPCFGNntDfeWtxJhGEToRHe3CwPT3WTQJO-
7GQBmNHSGiK3dzGPILulNKHI8Akhf7CZiYePeB5AEyV2mFvF5142QjmG0hao_pyH08WyCI9m9XSrGR7QxhzYPdKgShAAoJUyLkNBjsw
V2B0HsW2dg6tys4WZW7bMPOSmJ_o6mIFuuG6umzWUJkKSyqDcuvfUBQdNKZ73UHYMDheuTeZ8VrF9-
6CcUsBkG7EEEwIogpJ4tXGUcstZ-3lKeEf3RR68Ag7Okoy7G-
QqBL28KtCOKTery5iqB5ZbKGqxoGDfHEe1c5TqVbRp7pvvISTMVx8EgDuFoUnUC0IGAoFoFMf4WH0fCwN4kQL2WRfsYiyoHliC0aw2n
fI2gV-rciO9K26kj6mmxBf71urV2HKmjdJg8cR-V3m5T80D94e14jJFewyFs1_3XYgbkDu5CeeptesgObLBYy0oYCLEqxt4ht_-
fxNqBmsrnganoIn4tYRzXDyTkUXJv8w-
foFfkiC484crWlutAEbXzAKipDn2t43k9HOHFdM3nofw4CnY4YHeRMsOphIFEHXSYOgBa73zr4dzdR5If9ivLcuuRNe7lfmaEEhwzcq
llG-FDU9s0GQA","id_token":"eyJraWQiOiJ0ZXN0a2V5UlMiLCJhbGciOiJSUzI1NiJ9.
eyJhdF9oYXNoIjoiWjlPNjRkYVFUbXhKeE9obmZtdkl2USIsInN1YiI6IlZVRzQ3NzdZUDNOTVU1S1JGRVNYNlNLUkFQWExFNE1JIiw
iYXVkIjoiX2ZhYmJjNmY5MGExMmZmYjNmZWUwMDEwZDRjMmFiZGVlIiwiYWNyIjoidXJuOm1hY2U6aW5jb21tb246aWFwOnNpbHZlci
IsImF1dGhfdGltZSI6MTUzNzc3MzIyMCwiaXNzIjoiaHR0cHM6XC9cLzE5Mi4xNjguMC4xNTAiLCJleHAiOjE1Mzc3NzY4MjQsImlhd
CI6MTUzNzc3MzIyNCwibm9uY2UiOiJfcDdCX3lWeDlsQUhEelJQcGhnWkNRa2JYdmNnMC1UR0JqTm9ONVZvdTlrIiwiY2FtcHVzX2lk
IjoiWTJGdGNIVnpTV1E9In0.
ejxpf9N3LW4UqMlbB92F2_DUSNyVwHUckCPYNGO11WxkGRFd9mOKDC1m217f6ezQmZceyTT7rckkmWhWGDDQeJoxKJGzji0tmSHzZU6
2FAm32zxRL49rlmRT-cBpfH7cpAMjmhi8ORjci8pER7M205IL2ANXzM_q8RnLgtS9r4ztm9Xpmu83bPea-
y4Qqm1phpsejPbIanDNZ7WmrChD26_lrSid8dOLu_VTta3Mgh_BcF0sQbnvxSpz5VQIJBHp7vlfxuipfHPMrc96Pqj_6pksQHQjvBOG
r9yLIXF6lO6ja7-8dTRMlOhvTae2tbec62ptepLK5_OSgOFT_GwhEQ","token_type":"Bearer","expires_in":600}

Decode id token, you will get something like:

{
 kid: "testkeyRS",
 alg: "RS256"
}.{
 at_hash: "Z9O64daQTmxJxOhnfmvIvQ",
 sub: "VUG4777YP3NMU5KRFESX6SKRAPXLE4MI",
 aud: "_fabbc6f90a12ffb3fee0010d4c2abdee",
 acr: "password",
 auth_time: 1537773220,
 iss: "https:\/\/192.168.0.150",
 exp: 1537776824,
 iat: 1537773224,
 nonce: "_p7B_yVx9lAHDzRPphgZCQkbXvcg0-TGBjNoN5Vou9k",
 campus_id: "Y2FtcHVzSWQ="
}.[signature]

Check now userinfo response. It is one of the last lines in the log now

tail /opt/shibboleth-idp/logs/idp-process.log

Content:{"sub":"VUG4777YP3NMU5KRFESX6SKRAPXLE4MI"}

You notice the campus_id is only in the id token.
```

Exercise 5.2

Define attribute release rules to release "campusId" attribute to be released if asked to be released for ID Token as essential claim.

1. Make sure "campusId" is not requested anymore by scope.

```
        nano +633 /etc/httpd/conf.d/auth_openidc.conf

OIDCScope "openid"
```

2. Modify RP to ask "campusId" as essential ID Token claim.

```
        nano +417 /etc/httpd/conf.d/auth_openidc.conf

OIDCAuthRequestParams claims=%7B%22id_token%22%3A%7B%22campus_id%22%3A%20%7B%22essential%22%3A%20true%7D%7D%7D

service httpd restart
```

3. Add a new filtering rule that will release "campusId" only if requested to be released as essential ID Token claim

Hints, Tips and Result

```
        <AttributeFilterPolicy id="REQUESTED_CAMPUS_CLAIMS">
    <PolicyRequirementRule xsi:type="ANY" />
    <AttributeRule attributeID="campusId">
        <PermitValueRule xsi:type="oidcext:AttributeInOIDCRequestedClaims" matchOnlyIDToken="true"
onlyIfEssential="true" />
    </AttributeRule>
</AttributeFilterPolicy>
```

4. Authenticate the user and verify from the logs the attribute is released. At this point you should be able to do it without hints and tips.

Subject Identifier

Section Topics

- Subject Identifier
- Subject Identifier Generation

Subject Identifier

Locally unique and never reassigned identifier within the Issuer for the End-User, which is intended to be consumed by the Client. It MUST NOT exceed 255 ASCII characters in length. The value is a case sensitive string.

Two types of Subject Identifiers, public and pairwise, http://openid.net/specs/openid-connect-core-1_0.html#SubjectIDTypes

Different RPs may belong to same pairwise group by sharing the sector_identifier_uri - value

Placed always in ID Token and UserInfo response, whether requested or not.

Subject Identifier Generation

Extension provides default resolver and a configuration for it.

oidc-subject.properties

```
    # The source attribute used in generating the subject
idp.oidc.subject.sourceAttribute = uid

# The digest algorithm used in generating the subject
#idp.oidc.subject.algorithm = SHA

# The encoding used in generating the subject
#idp.oidc.subject.encoding = BASE32

# The salt used in generating the subject
# Do *NOT* share the salt with other people, it's like divulging your private key.
idp.oidc.subject.salt = this_too_should_be_ch4ng3dExercises
```

attribute-resolver.xml

```
        <!-- Subject Identifier is a attribute that must always be resolved.
    There has to be exactly one resolved and filtered attribute that would be encoded as 'sub'.
    This example attribute (the data connector actually ) will generate public or pairwise 'sub' depending on
client registration data.  -->

    <AttributeDefinition id="subject" xsi:type="Simple" activationConditionRef="SubjectRequired">
        <InputDataConnector ref="computedSubjectId" attributeNames="subjectId"/>
        <AttributeEncoder xsi:type="oidcext:OIDCString" name="sub" />
    </AttributeDefinition>

    <!--
    Subject Identifier is a attribute that must always be resolved.
    There has to be exactly one resolved and filtered attribute that would be encoded as 'sub'.

    Use activation conditions and filters to ensure the requirement is met if you have need for several
different kind of formats for 'sub'.

    <AttributeDefinition id="subject-public" xsi:type="Simple" sourceAttributeID="uid" activationConditionRef="
PublicRequired">
        <Dependency ref="uid" />
        <AttributeEncoder xsi:type="oidcext:OIDCString" name="sub" />
    </AttributeDefinition>

    <AttributeDefinition id="subject-pairwise" xsi:type="Simple" activationConditionRef="PairwiseRequired">
        <InputDataConnector ref="computedSubjectId" attributeNames="subjectId"/>
        <AttributeEncoder xsi:type="oidcext:OIDCString" name="sub" />
    </AttributeDefinition>
    -->


   <!-- Data Connector for generating 'sub' claim.
         The connector may be used to generate both public and pairwise subject values -->
    <DataConnector id="computedSubjectId" xsi:type="ComputedId"
            generatedAttributeID="subjectId"
            sourceAttributeID="%{idp.oidc.subject.sourceAttribute}"
            salt="%{idp.oidc.subject.salt}"
            algorithm="%{idp.oidc.subject.algorithm:SHA}"
            encoding="%{idp.oidc.subject.encoding:BASE32}">
            <Dependency ref="%{idp.oidc.subject.sourceAttribute}"/>
    </DataConnector>
```

attribute-filter.xml

```
    <AttributeFilterPolicy id="OPENID_SCOPE">
    <PolicyRequirementRule xsi:type="oidcext:OIDCScope" value="openid" />
    <AttributeRule attributeID="subject">
        <PermitValueRule xsi:type="ANY" />
    </AttributeRule>
</AttributeFilterPolicy>
```

Exercise 6.1

Configuring Subject claim

1. Modify the registration data of the RP to list subject of type pairwise.

```
          nano /opt/shibboleth-idp/metadata/oidc-client.json

{
  "scope":"openid info profile email address phone",
  "redirect_uris":["https://192.168.0.150:8443/protected/redirect_uri"],
  "client_id":"test_rp",
  "client_secret":"testSecret1234",
  "response_types":["id_token","code"],
  "grant_types":["authorization_code","implicit","refresh_token"],
  "subject_type":"pairwise"
}
```

2. Authenticate the user. Verify from landing page or from logs that the value of subject is different from previously used public value.

Hints, Tips and Result

```
          [OIDC_CLAIM_sub] => DQ3YFEXBF65XMAULUJHBAI34IVRR3GT5
```

3. Activate the attributes "subject-public" and "subject-pairwise". Ensure the filtering rules are such that these attributes are not filtered out but the attribute "subject" is filtered out for your RP.
4. Modify the registration data of the rp to request subject of type public.

```
          nano /opt/shibboleth-idp/metadata/oidc-client.json

{
  "scope":"openid info profile email address phone",
  "redirect_uris":["https://192.168.0.150:8443/protected/redirect_uri"],
  "client_id":"test_rp",
  "client_secret":"testSecret1234",
  "response_types":["id_token","code"],
  "grant_types":["authorization_code","implicit","refresh_token"],
  "subject_type":"public"
}
```

5. Authenticate the user. Verify from landing page or from logs that the value of subject is now this plain text value.

Hints, Tips and Result

```
          [OIDC_CLAIM_sub] => teppo
```

Profile Configurations

Section Topics

- SAML and OIDC profile configurations
- Profile configuration options
- Default vs. RP-specific profile configurations

SAML and OIDC profile configurations

Default (SAML) profile configurations

- The profile configuration file is */opt/shibboleth-idp/conf/relying-party.xml*

```
          <beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:util="http://www.springframework.org/schema/util"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:c="http://www.springframework.org/schema/c"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org
/schema/beans/spring-beans.xsd
                           http://www.springframework.org/schema/context http://www.springframework.org
/schema/context/spring-context.xsd
                           http://www.springframework.org/schema/util http://www.springframework.org
/schema/util/spring-util.xsd"
       default-init-method="initialize"
       default-destroy-method="destroy">

    <!--
    Unverified RP configuration, defaults to no support for any profiles. Add <ref> elements to the
list
    to enable specific default profile settings (as below), or create new beans inline to override
defaults.

    "Unverified" typically means the IdP has no metadata, or equivalent way of assuring the identity
and
    legitimacy of a requesting system. To run an "open" IdP, you can enable profiles here.
    -->
    <bean id="shibboleth.UnverifiedRelyingParty" parent="RelyingParty">
        <property name="profileConfigurations">
            <list>
            <!-- <bean parent="SAML2.SSO" p:encryptAssertions="false" /> -->
            </list>
        </property>
    </bean>

    <!--
    Default configuration, with default settings applied for all profiles, and enables
    the attribute-release consent flow.
    -->
    <bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">
        <property name="profileConfigurations">
            <list>
                <bean parent="Shibboleth.SSO" p:postAuthenticationFlows="attribute-release" />
                <ref bean="SAML1.AttributeQuery" />
                <ref bean="SAML1.ArtifactResolution" />
                <bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" />
                <ref bean="SAML2.ECP" />
                <ref bean="SAML2.Logout" />
                <ref bean="SAML2.AttributeQuery" />
                <ref bean="SAML2.ArtifactResolution" />
                <ref bean="Liberty.SSOS" />
            </list>
        </property>
    </bean>


...
```

Default OIDC profile configurations

- The OIDC profile configuration file is */opt/shibboleth-idp/conf/oidc-relying-party.xml*

```
          ...

    <!-- OIDC Profile Configurations. -->
    <bean id="OIDC.SSO" class="org.geant.idpextension.oidc.config.OIDCCoreProtocolConfiguration"
        p:securityConfiguration-ref="%{idp.security.oidc.config:shibboleth.oidc.
DefaultSecurityConfiguration}"
        p:iDTokenLifetime="%{idp.oidc.idToken.defaultLifetime:PT1H}"
        p:accessTokenLifetime="%{idp.oidc.accessToken.defaultLifetime:PT10M}"
        p:authorizeCodeLifetime="%{idp.oidc.authorizeCode.defaultLifetime:PT5M}"
        p:refreshTokenLifetime="%{idp.oidc.refreshToken.defaultLifetime:PT2H}"
        p:servletRequest-ref="shibboleth.HttpServletRequest"
        p:tokenEndpointAuthMethods="%{idp.oidc.tokenEndpointAuthMethods:client_secret_basic,
client_secret_post,client_secret_jwt,private_key_jwt}" />

    <bean id="OIDC.UserInfo" class="org.geant.idpextension.oidc.config.OIDCUserInfoConfiguration"
        p:securityConfiguration-ref="%{idp.security.oidc.config:shibboleth.oidc.
DefaultSecurityConfiguration}"
        p:servletRequest-ref="shibboleth.HttpServletRequest" />

    <bean id="OIDC.Registration" class="org.geant.idpextension.oidc.config.
OIDCDynamicRegistrationConfiguration"
        p:securityConfiguration-ref="%{idp.security.oidc.config:shibboleth.oidc.
DefaultSecurityConfiguration}"
        p:servletRequest-ref="shibboleth.HttpServletRequest"
        p:tokenEndpointAuthMethods="%{idp.oidc.dynreg.tokenEndpointAuthMethods:client_secret_basic,
client_secret_post,client_secret_jwt,private_key_jwt}" />

    <bean id="OIDC.Configuration" class="org.geant.idpextension.oidc.config.
OIDCProviderInformationConfiguration"
        p:securityConfiguration-ref="%{idp.security.oidc.config:shibboleth.oidc.
DefaultSecurityConfiguration}"
        p:servletRequest-ref="shibboleth.HttpServletRequest"/>

    <bean id="OAUTH2.Revocation" class="org.geant.idpextension.oauth2.config.
OAuth2TokenRevocationConfiguration"
        p:securityConfiguration-ref="%{idp.security.oidc.config:shibboleth.oidc.
DefaultSecurityConfiguration}"
        p:servletRequest-ref="shibboleth.HttpServletRequest"/>

...
```

Relying party configuration

- The main configuration file is */opt/shibboleth-idp/conf/relying-party.xml*

```
         <beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:util="http://www.springframework.org/schema/util"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:c="http://www.springframework.org/schema/c"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org
/schema/beans/spring-beans.xsd
                           http://www.springframework.org/schema/context http://www.springframework.org
/schema/context/spring-context.xsd
                           http://www.springframework.org/schema/util http://www.springframework.org
/schema/util/spring-util.xsd"
       default-init-method="initialize"
       default-destroy-method="destroy">

    <import resource="oidc-relying-party.xml"/>

    <bean id="shibboleth.UnverifiedRelyingParty"  p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" parent="RelyingParty">
        <property name="profileConfigurations">
            <list>
                <bean parent="OIDC.Registration" />
                <bean parent="OIDC.Configuration" />
            </list>
        </property>
    </bean>

    <bean id="shibboleth.DefaultRelyingParty" p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" parent="RelyingParty">
        <property name="profileConfigurations">
            <list>
                <bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" />
                <ref bean="SAML2.Logout" />
                <bean parent="OIDC.SSO" p:postAuthenticationFlows="attribute-release" />
                <bean parent="OIDC.UserInfo"/>
                <bean parent="OAUTH2.Revocation"/>
            </list>
        </property>
    </bean>

...
```

Profile configuration options

Profile configuration options

- https://github.com/CSCfi/shibboleth-idp-oidc-extension/wiki/ProfileConfigurations
- https://wiki.shibboleth.net/confluence/display/IDP30/RelyingPartyConfiguration
    - Shared options with all configurations
        - Standard security configuration and our extensions.
            - Used in various ways, depending on the context
                - OIDC.Configuration: signing + encryption configuration (credentials, algorithms) for openid-configuration
                - OIDC.Registration: which signing + encryption configuration details are supported
                - OIDC.SSO: which signing + encryption configuration is enabled
        - Inbound interceptor flows
        - Outbound interceptor flows
    - Client authenticable configuration options for OIDC.SSO, OIDC.Registration and OAUTH2.Revocation
        - Endpoint authentication methods
    - Flow-aware configuration options for OIDC.SSO and OIDC.Registration
        - Flags to enable implicit, hybrid and authorization code flows
        - Flag to enable refresh tokens
    - Flow-specific options
        - Multiple options especially for OIDC.SSO and OIDC.Registration (lifetimes, etc)
        - Post authentication flows, default authentication methods for OIDC.SSO

Default vs. RP-specific profile configuration

Profile configuration vs client metadata - overlapping configurations

- Default profile configurations enable wide set of features
- Standard *shibboleth.RelyingPartyOverrides* mechanism can be used with OIDC RPs too

Snippet of /opt/shibboleth-idp/conf/relying-party.xml

```
         ...

    <util:list id="shibboleth.RelyingPartyOverrides">
        <bean parent="RelyingPartyByName"  p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" c:relyingPartyIds="test_rp">
            <property name="profileConfigurations">
                <list>
                    <bean parent="OIDC.SSO" />
                </list>
            </property>
        </bean>
    </util:list>

...
```

- Some of the profile configuration options have overlapping claims in the client metadata
    - E.g. token endpoint authentication methods

Exercises

Exercise 7.1 - Modifying default profile configuration

1. Add additional audience ***test_api*** for all authenticated relying parties

   Snippet of /opt/shibboleth-idp/conf/relying-party.xml

```
          ...

    <bean id="shibboleth.DefaultRelyingParty" p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" parent="RelyingParty">
        <property name="profileConfigurations">
            <list>
                <bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" />
                <ref bean="SAML2.Logout" />
                <bean parent="OIDC.SSO" p:postAuthenticationFlows="attribute-release" p:
additionalAudiencesForIdToken="test_api" />
                <bean parent="OIDC.UserInfo"/>
                <bean parent="OAUTH2.Revocation"/>
            </list>
        </property>
    </bean>

...
```

2. Verify that the additional audience is visible in the *id_token.*

   Hints, Tips and Result

```
          [OIDC_CLAIM_aud] => test_rp,test_api
```

Exercise 7.2 - Modifying RP-specific profile configuration

1. Remove *postAuthenticationFlows* and *additionalAudiencesForIdToken* settings for *test_rp*.

   Snippet of /opt/shibboleth-idp/conf/relying-party.xml

```
          ...

    <util:list id="shibboleth.RelyingPartyOverrides">
        <bean parent="RelyingPartyByName"  p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" c:relyingPartyIds="test_rp">
            <property name="profileConfigurations">
                <list>
                    <bean parent="OIDC.SSO" />
                </list>
            </property>
        </bean>
    </util:list>

...
```

2. Are the additional audiences now visible for *test_rp* as they are defined in **shibboleth.DefaultRelyingParty**? Why?

Hints, Tips and Result

```
          [OIDC_CLAIM_aud] => test_rp


They are not, because the settings from OICD.SSO defined in oidc-relying-party.xml are inherited, not
OIDC.SSO settings from shibboleth.DefaultRelyingParty.
```

3. What happens if you configure that only *private_key_jwt* is accepted as the token endpoint authentication method for *test_rp*?

Snippet of /opt/shibboleth-idp/conf/relying-party.xml

```
          ...

    <util:list id="shibboleth.RelyingPartyOverrides">
        <bean parent="RelyingPartyByName"  p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" c:relyingPartyIds="test_rp">
            <property name="profileConfigurations">
                <list>
                    <bean parent="OIDC.SSO" p:tokenEndpointAuthMethods="private_key_jwt" />
                </list>
            </property>
        </bean>
    </util:list>

...
```

Snippet of /opt/shibboleth-idp/logs/idp-process.log

```
          ...
2018-10-04 12:45:49,839 - DEBUG [org.geant.idpextension.oidc.profile.impl.
InitializeRelyingPartyContext:170] - Attaching RelyingPartyContext for rp test_rp
2018-10-04 12:45:49,839 - DEBUG [org.geant.idpextension.oidc.profile.impl.
InitializeRelyingPartyContext:175] - Profile Action InitializeRelyingPartyContext: Setting the rp
context verified
2018-10-04 12:45:49,840 - DEBUG [net.shibboleth.idp.profile.impl.SelectRelyingPartyConfiguration:136]
- Profile Action SelectRelyingPartyConfiguration: Found relying party configuration EntityNames
[test_rp,] for request
2018-10-04 12:45:49,843 - WARN [org.geant.idpextension.oidc.profile.impl.
ValidateEndpointAuthentication:250] - Profile Action ValidateEndpointAuthentication: The requested
method client_secret_basic is not enabled
2018-10-04 12:45:49,843 - WARN [org.geant.idpextension.oidc.profile.impl.
ValidateEndpointAuthentication:230] - Profile Action ValidateEndpointAuthentication: Unsupported
client authentication method client_secret_basic
2018-10-04 12:45:49,853 - WARN [org.opensaml.profile.action.impl.LogEvent:105] - A non-proceed event
occurred while processing the request: AccessDenied
...
```

Exercise 7.3 - Advanced access-control configuration with context-check

The goal of this exercise is to configure the **test_rp** application to be only accessible for **teppo2** user. Shibboleth IdP provides context-check interceptor for this purpose.

1. Add *context-check* post authentication flow to the relying party configuration

Snippet of /opt/shibboleth-idp/conf/relying-party.xml

```
          ...

    <util:list id="shibboleth.RelyingPartyOverrides">
        <bean parent="RelyingPartyByName"  p:responderIdLookupStrategy-ref="
profileResponderIdLookupFunction" c:relyingPartyIds="test_rp">
            <property name="profileConfigurations">
                <list>
                    <bean parent="OIDC.SSO" p:postAuthenticationFlows="context-check"/>
                </list>
            </property>
        </bean>
    </util:list>

...
```

2. Edit */opt/shibboleth-idp/conf/intercept/context-check-intercept-config.xml* for your needs. HINT! The existing file contains good basis, find out from attribute-resolver which is the username in your configuration.

Hints, Tips and Result

```
          ...


    <bean id="shibboleth.context-check.Condition" parent="shibboleth.Conditions.AND">
        <constructor-arg>
            <list>
                <bean parent="shibboleth.Conditions.RelyingPartyId" c:candidates="#{{'test_rp'}}" />
                <bean class="net.shibboleth.idp.profile.logic.SimpleAttributePredicate"
                        p:useUnfilteredAttributes="true">
                    <property name="attributeValueMap">
                        <map>
                            <entry key="uid">
                                <list>
                                    <value>teppo2</value>
                                </list>
                            </entry>
                        </map>
                    </property>
                </bean>
            </list>
        </constructor-arg>
    </bean>


...
```

3. Restart IDP service and try to access the test RP with *teppo* and *teppo2* (same password). You can logout the user via */idp/profile/Logout* -endpoint.

Snippet of /opt/shibboleth-idp/logs/idp-process.log

```
          ...
2018-10-05 01:20:37,187 - INFO [Shibboleth-Audit.SSO:276] -
20181005T012037Z|AuthenticationRequest||test_rp|http://csc.fi/ns/profiles/oidc/sso/browser|https://192.
168.0.150|||teppo|||||
...
```