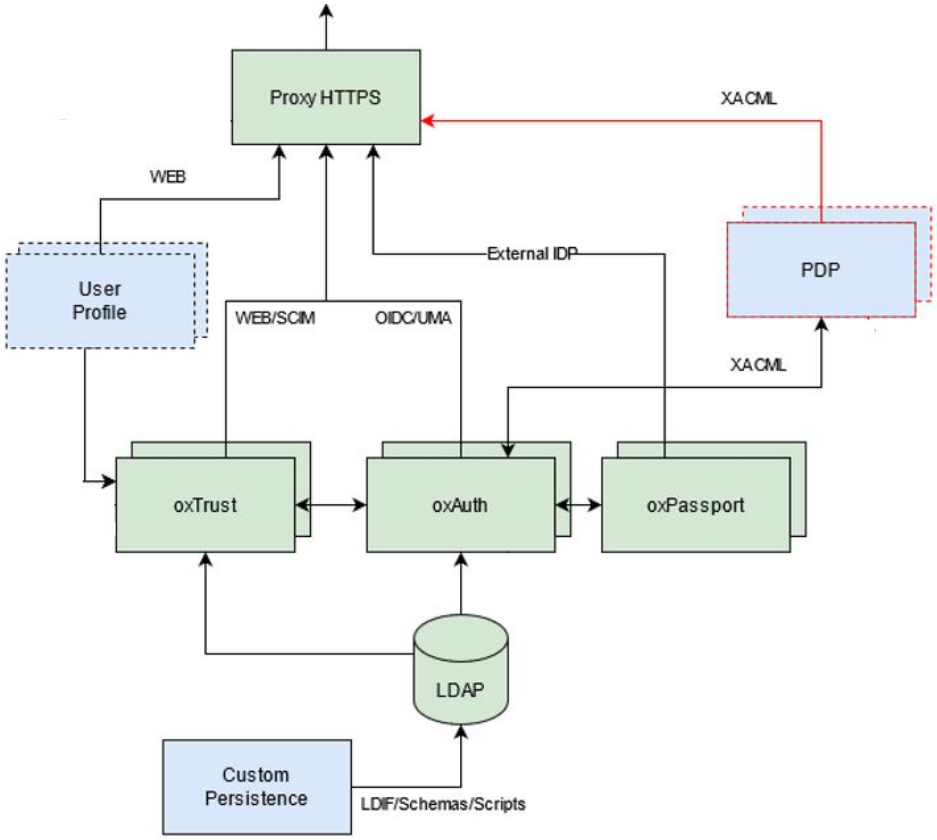
User Management

EOEPCA_UM Operator Training Preparation

Alvaro Villanueva

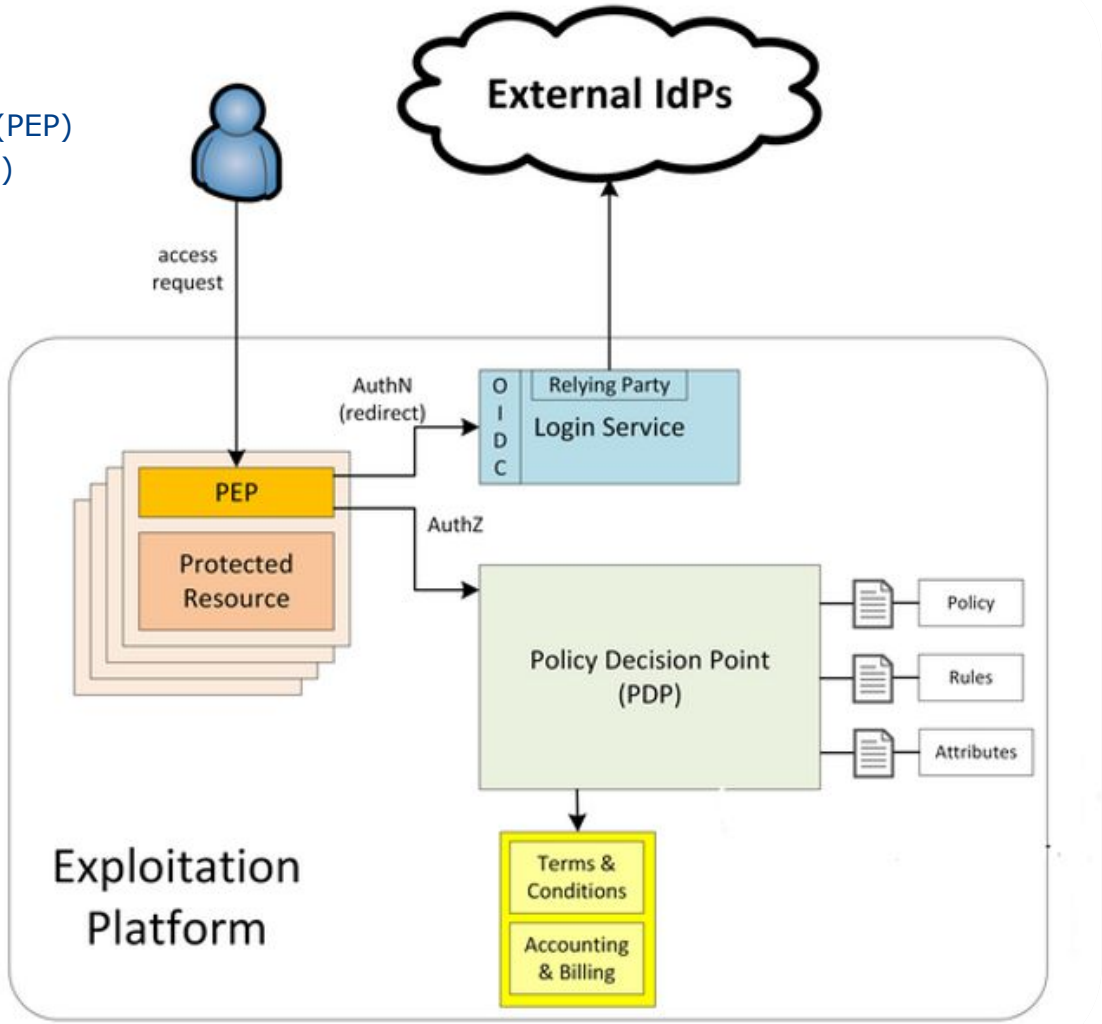
UM Components

1. Authentication and Authorization:
   - Login-Service (Gluu)
   - Policy Enforcement Point (PEP)
   - Policy Decision Point (PDP)
2. User Management:
   - User Profile
   - Login service



Authentication and Authorization

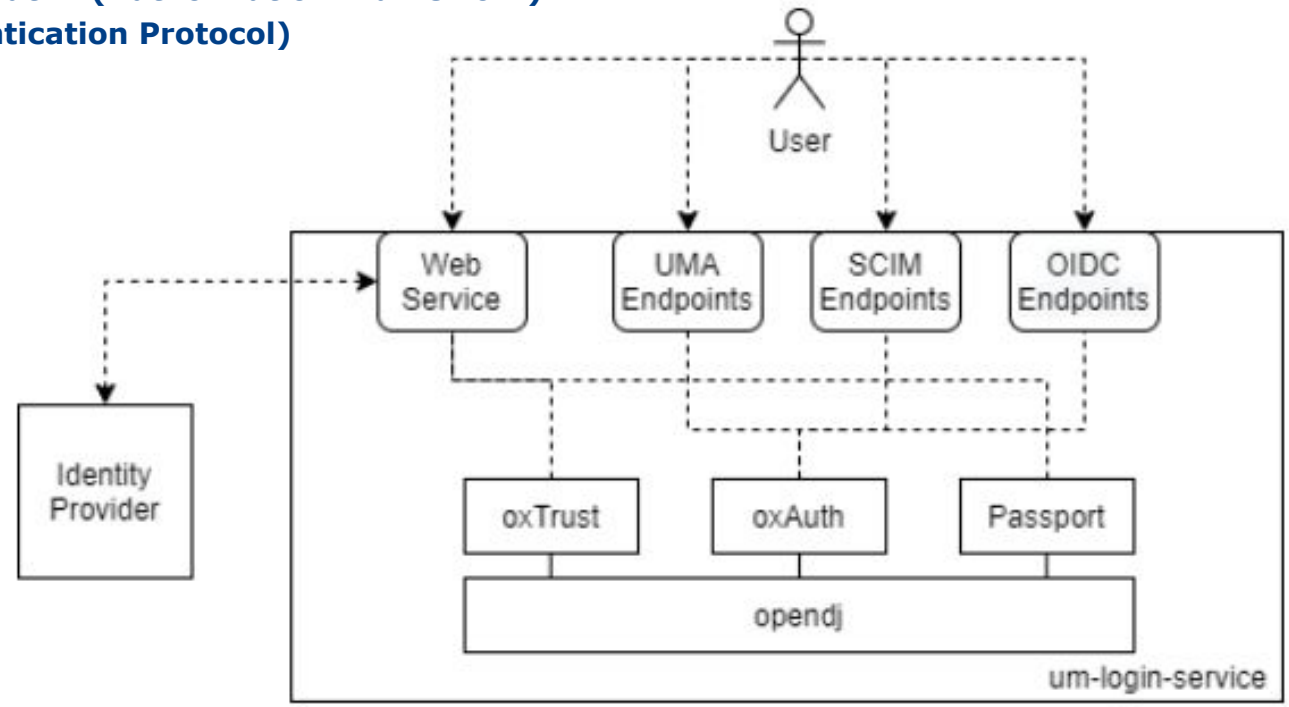
- Login-Service (Gluu)
- Policy Enforcement Point (PEP)
- Policy Decision Point (PDP)



Authentication and Authorization

Login Service:

- Powered by Gluu
- Modular component
- Support of OAuth2 (Authorization Framework)
- OIDC (Authentication Protocol)
- SCIM



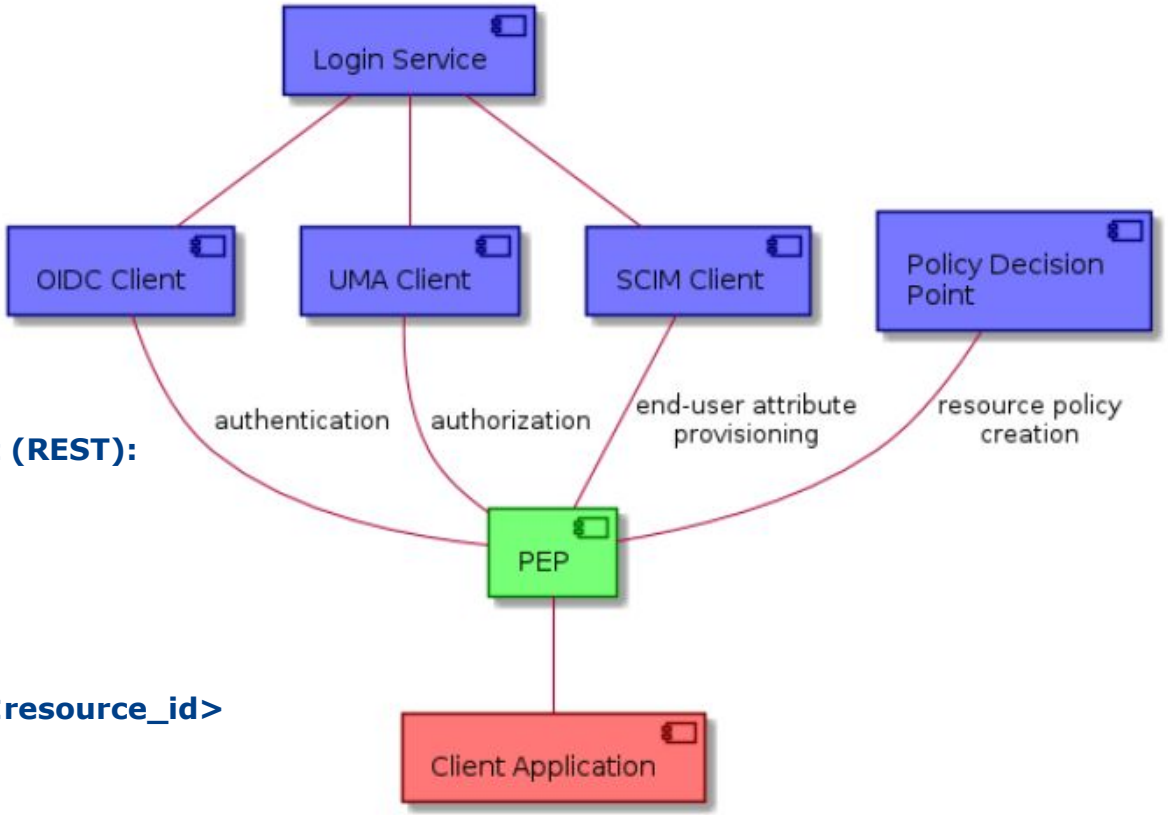
Link to doc:

https://eoepca.github.io/master-system-design/current/#mainUserManagement
https://support.gluu.org/docs/

Authentication and Authorization

Policy Enforcement Point (PEP):

- Python Flask Application
- Stands between a client and the client's desired resource.
- Resource Database
- Proxy API (Auth)
- Resource API



Endpoints:

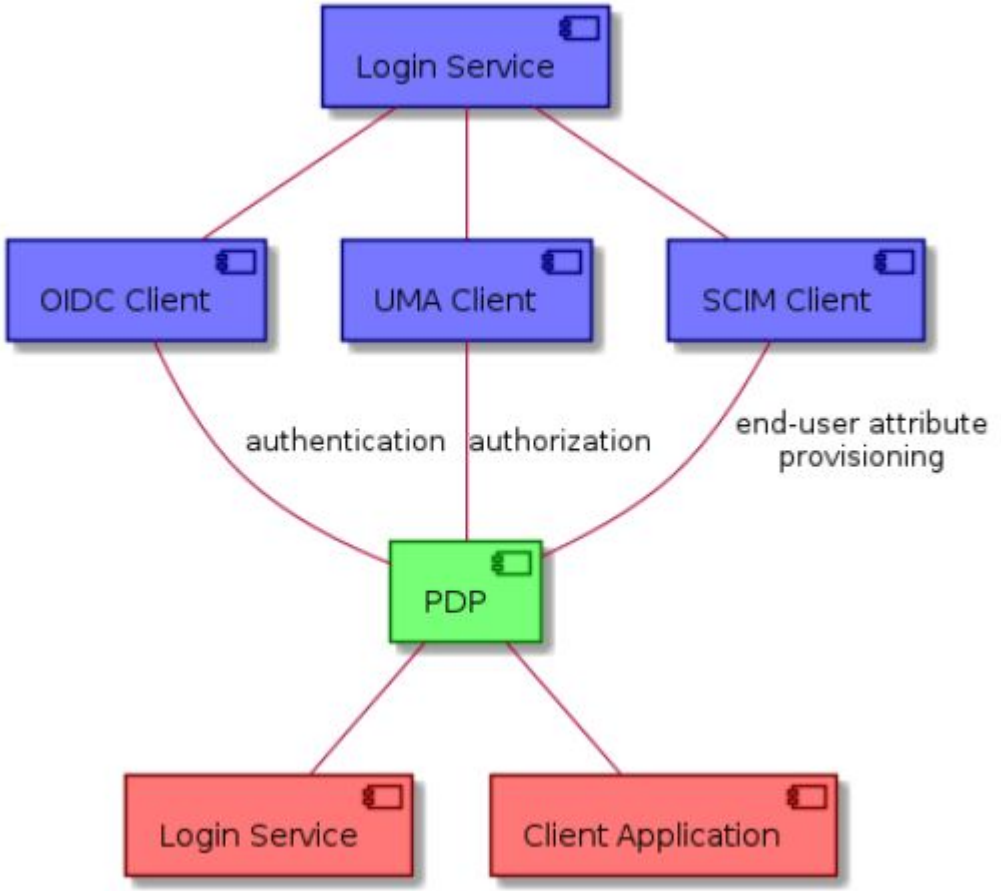
- **Proxy and UMA Flows Endpoint (REST):**
  /
- **Authorize Endpoint (REST):**
  /authorize
- **Resources Endpoint (REST):**
  /resources
  /resources/<resource_id>

Link to doc: https://eoepca.github.io/um-pep-engine/SDD/current/

Authentication and Authorization

Policy Decision Point (PDP):

- Python Flask Application
- UMA Flow 2.0
- Policy Database
- Policy API



Endpoints:

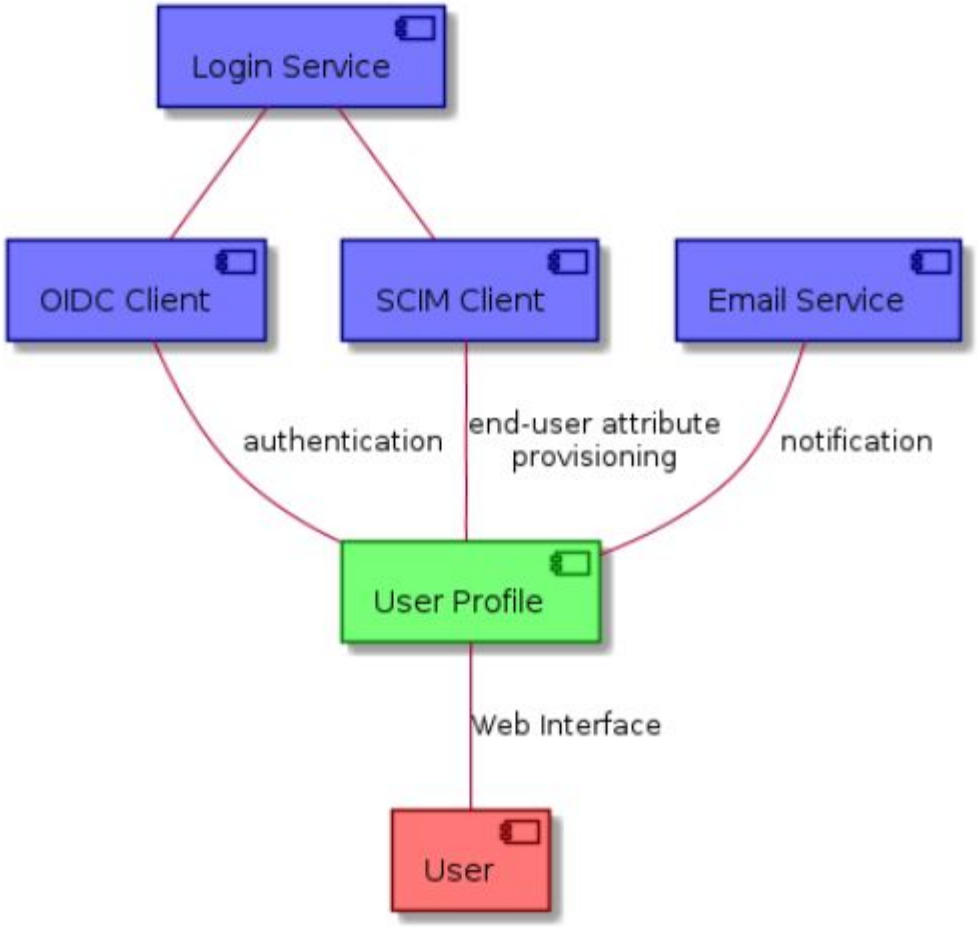
- Platform Policy API (REST):
  - /policy/
  - /policy/<policy_id>
- XACML Policy Check (REST):
  - /policy/validate

Link to doc: https://eoepca.github.io/um-pdp-engine/SDD/current/

User Management

User Profile:

- **Python Flask Application**
- **Web Application**
- **Attribute Control**
- resource visualization



Endpoints:

- **Front Application:**

/web_ui

Link to doc: https://eoepca.github.io/um-user-profile/current/

Distribution and requirements

UM Building Blocks supports the following distributions:

- Docker Containers

Compatible with Kubernetes and Helm Charts

https://github.com/EOEPCA/helm-charts-dev/tree/develop/charts

Repositories:

- Github: https://github.com/EOEPCA
- Dockerhub:
  - Docker Containers: https://hub.docker.com/orgs/eoepca/repositories

BB Configuration

Each Building Block has its own configuration file, which is the *values.yaml*, before installation read the meaning of each value you fill.

The following links will provide the latest changes in terms of configuration values:

- Login Service: https://github.com/EOEPCA/um-login-service/wiki/Chart-Configuration#values
- PDP: https://github.com/EOEPCA/um-pdp-engine/wiki/Chart-Configuration#values
- PEP: https://github.com/EOEPCA/um-pep-engine/wiki/Chart-Configuration#values
- User Profile: https://github.com/EOEPCA/um-user-profile/wiki/Chart-Configuration#values

Tips and Use Cases

Deployment

- The login service is by far the most complicated component. All other components depends on it, so it's the first thing to install.
- If the Login Service's deployment needs a clean restart, its important to remove ***gluu*** secret and configmap manually, but also remove the *config-init* path stored in the persistent volume.
- The PDP component is the second to be installed since PEP interacts with it.

Usage

- Gluu's user interface is the main resource for the UM operator, most questions are documented on their support page
- Administrator password can be set in the login service's config file.
- In order to know the endpoints where the services are being published, the best way is to describe the Ingress of each service (*kubectl describe ingress <BBingress> -n <namespace>)*
- Expiration date of certificates and keys can be found in each instance from the login service under */etc/certs*
- *Login-service well-known endpoints:*
    - /oxauth/.well-known/openid-configuration
    - /.well-known/uma2-configuration

Custom Attributes

In order to add new custom attributes to users, the best way to do it is through a file found in the helm definition of the login service (77-customAttributes.ldif) before the installation.

By editing this file, the LDAP database will allow the attribute to be inserted via Gluu interface

If you want to insert a new attribute in a running instance, follow these steps:

- Enter Opendj instance and edit, based on other attributes there, the file */opt/opendj/config/schema/77-customAttributes.ldif*
- Restart opendj service + oxauth service + oxtrust service
- Insert in Gluu Interface the attribute matching the same values given

Users now can add the new attribute to their profiles, but if they want to retrieve it involving a client, this attribute will have to be introduced in a scope and in turn specify it in the client used.

Gluu doc: https://gluu.org/docs/gluu-server/4.1/admin-guide/attribute/

Customization of the front pages of Gluu

The structure within is very sensible so is recommended to follow carefully the same directory design as the example here:
https://github.com/EOEPCA/eoepca/tree/develop/system/clusters/creodias/user-management/customizathttps://github.com/EOEPCA/eoepca/tree/develop/system/clusters/creodias/user-management/customizationsions

The application of the front layer is done after the login service is installed.
the steps to apply it can be found in the deploy_customizations.sh file. In case of encountering problems, the manual way to do it would be inserting their respective pages into the oxauth or oxtrust containers under the path */opt/gluu/jetty/<oxauth or identity>/custom/*

Gluu doc: https://gluu.org/docs/gluu-server/4.1/operation/custom-design/

Add an external Identity Provider

In all integration of an authentication method through an external IDP there are always some similar steps that need to be followed.

- First step is to create a Identity Provider in Gluu Interface:
    - Go to Passport>Providers>Add New Provider
    - Fill the mandatory options and create the Provider
- A client must be created within the Identity Provider (i.e. GitHub, Google..) as an OAuth Application
    - Redirect Uri must be the callback URL that the Gluu Provider just generated
    - Copy client_id, client_secret and add them to the Providers Options in your Gluu Provider
- Passport functionality must be enabled too.

Gluu Doc: https://gluu.org/docs/gluu-server/4.1/authn-guide/passport/#inbound-identity

Demo

The demo can be found here:
https://github.com/EOEPCA/demo/blob/main/demoroot/notebooks/01%20User%20Management.ipynb

The demo summarizes almost all the use cases required for authentication and authorization. It has examples of use of the PEP and PDP APIs as well as an example of integration with a web application.

To summarize:

- Usage of PEP API and UMA Proxy
    - Insertion of resources
    - Complete UMA Flow
- Usage of PDP API
    - Update of policies
    - Policy Decision RULE standard example
- Usage of Login Service APIs
    - SCIM endpoint for client registration example
    - ID_TOKEN retrieval and inspection
    - External IDP auth example