CUE User Manager

User Guide

3.1.0-11

Table of Contents

1 Introduction

CUE User Manager is CCI / Escenic's new solution for managing user access to CUE. It provides a standard interface to a centralized **Identity and Access Management (IAM)** system that can be shared by all CUE-related back-end systems. Instead of the CUE Content Store having its own integrated user management functionality, it can delegate responsibility for access control to CUE User Manager.

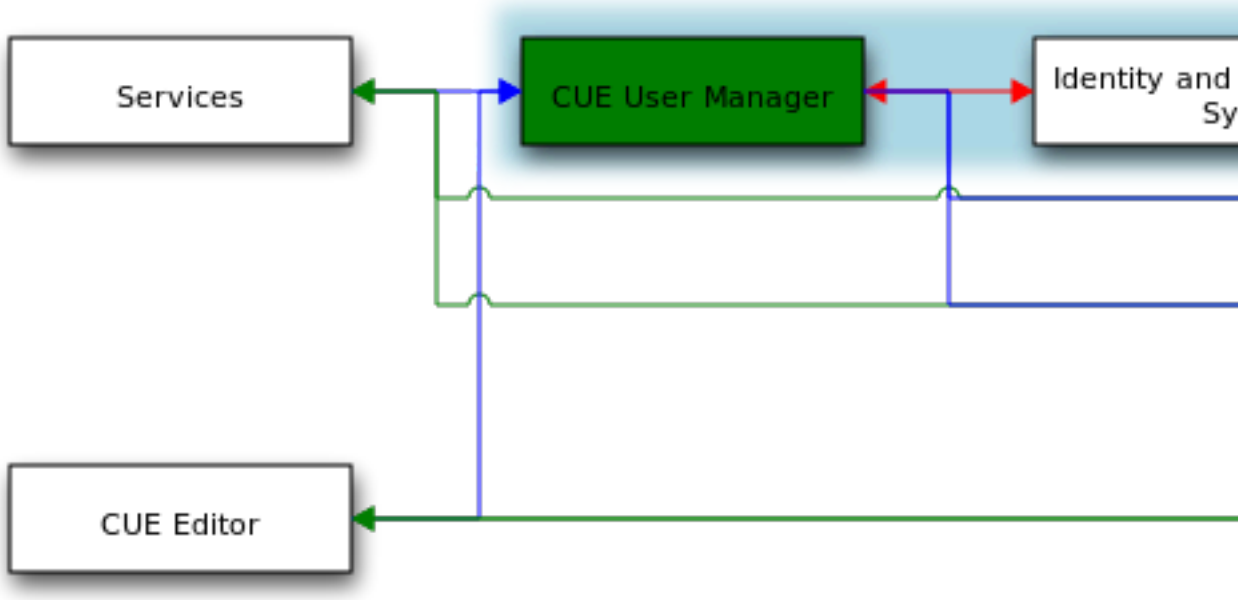
The use of such a centralized IAM system has a number of advantages:

- Single sign-on for users
- Simpler user management: no problems keeping identities consistent between systems, no redundancies or duplication
- Simpler auditing: all user activities are reported by one system in a single, consistent format
- Reduced resource usage
- Deal with upgrades to the IAM system: any changes in the IAM system's API will only affect CUE User Manager rather than requiring changes in all back-end systems.

CUE User Manager needs an IAM system that provides the following user management tasks:

- **Authentication**: is the user who he says he is?
- **Authorization**: what is this user allowed to do?
- **Identity management**: what information should be stored for each user (mail address, profile and so on).

IAM systems can be configured either to act as a complete, standalone IAM system or to co-operate with other authentication systems such as Active Directory, Google and Facebook.



CUE Content Store and CUE Print are already able to make use of a common IAM system (Active Directory) to enable single sign-on for users, so what benefit does CUE User Manager offer? It can best be seen as an insulating layer between the CUE back-end systems and the IAM system, making it easier to deal with upgrades to the IAM system: any changes in the IAM system's API will only affect CUE User Manager rather than requiring changes in all back-end systems.

This guide assumes that CUE User Manager is installed on **`um.example.com`** and the IAM backend system is installed on **`iam.example.com`**.

1.1 Supported IAM Systems

CUE User Manager supports the following IAM systems:

- Azure Active Directory
- Keycloak, either standalone or together with Active Directory

These IAM systems are the only systems supported by Stibo DX.

In theory, CUE User Manager will also work with other IAM systems, as long as they support:

- OpenID Connect Core 1.0 for authentication.
- OpenID Connect Discovery 1.0 for service discovery.
- System for Cross-domain Identity Management 2 (SCIM 2) for identity management.

- User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization for web resource authorization.

> If you choose to use an unsupported IAM system together with CUE User Manager, thorough testing is recommended.

2 Getting Started

This chapter contains step-by-step instructions for installing, configuring and running CUE User Manager.

2.1 Repository Code Names

Stibo DX is in the process of introducing **repository code names** to help with the installation of CUE software. A repository code name is an easy to remember word used to name an APT repository. A repository that has been given a code name contains packages for a set of product versions that are guaranteed to be inter-operable. Instead of having to remember a lot of version numbers and check them for compatibility with one another, you can just use the same code name when installing all the components of your system, and be sure that they will work together.

An alphabetic sequence is followed when choosing code names, so a repository with the code name **`eyre`** can be assumed to contain later versions of products than a repository with the code name **`dedman`**.

If you have been given a code name for installing CUE software, then you should use it as follows when adding CUE repositories to your list of sources:

```
# echo "deb https://user:password@apt.escenic.com code-name main non-free" >> /etc/
apt/sources.list.d/escenic.list
```

If your code name is **`eyre`**, for example, then you should enter:

```
# echo "deb https://user:password@apt.escenic.com eyre main non-free" >> /etc/apt/
sources.list.d/escenic.list
```

This ensures that all products installed using the **`apt`** command will be installed from the **`eyre`** repository.

If you haven't been given a code name, then use the default name **`stable`**. This will add the most recent stable CUE APT repository to your list of sources.

2.2 Installation Procedure

To install the CUE User Manager on an Ubuntu or other Debian-based system, do the following:

1. Log in as **`root`**.

2. If necessary, add the CUE APT repository to your list of sources:

```
# echo "deb https://user:password@apt.escenic.com code-name main non-free" >> /
etc/apt/sources.list.d/escenic.list
```

   where:

   - *user* and *password* are your CUE download credentials (the same ones you use to access the CUE Maven repository). If you do not have any download credentials, please contact CUE support.
   - *code-name* is either a repository code name supplied to you by Stibo DX or, if no code name has been supplied, the default name **`stable`**. For further information, see section 2.1.

3. Enter the following commands:

```
# apt-get update
# apt-get install cue-user-manager
```

On RedHat systems, enter the following command as **`root`**:

```
# rpm -Uvh https://user:password:yum.escenic.com/rpm/cue-user-
manager-3.1.0-11.x86_64.rpm
```

You should now have CUE User Manager installed. The next step is to secure it with TLS, see section 7.7 and then to integrate it with your IAM system, see the chapter 7. Finally, you must tell the other CUE components to use CUE User Manager instead of their default authentication procedures, see section 4.3 and section 4.4

2.3 Configuration

To configure CUE User Manager you must go to the IAM system and create an OIDC client. After selecting your secret passphrase, it will give you an OIDC client ID. CUE User Manager will use this client ID and secret for all authentication services against the IAM system. You can now add the client ID and secret, together with the OIDC discovery endpoint and SCIM endpoint to **`/etc/escenic/user-manager/user-manager.yaml`**:

```
provider:
  oidcEndpoint: https://iam.example.com/.well-known/openid-configuration
  clientId: my-um-oidc-id
  clientSecret: foo-bar-baz
  scimEndpoint: https://iam.example.com/scim/v2/
```

The endpoint URIs should be a part of the IAM system documentation.

The configuration file has many more options, but the ones above should be enough to get you started. The configuration file has lots of helpful comments and examples. See the section on user-manager.yaml (section 4.1.1) for further documentation.

2.3.1 IAMs without SCIM support

If the IAM backend doesn't have SCIM support, it's possible to add an additional field **`type: keycloak`** to the **`provider`** block and extend the OIDC client configuration in the IAM backend to include the user groups in a custom OIDC claim. This claim must be named **`groups`** and be included in the OIDC UserInfo endpoint.

It's important that this **`group`** claim is **not** added to the access token, or the generated URIs will be too long. (If, for example, the group contains around 50 groups then the resulting redirect URI will contain more than 3,000 characters. A redirect this long may not be followed by many browsers.)

For doing this with the Keycloak IAM provider, see the documentation on how to add an additional group mapper.

2.4 Starting and Stopping

To start CUE User Manager, enter:

```
# /etc/init.d/user-manager start
```

In addition to **`start`**, the **`init.d`** script has these options for controlling CUE User Manager

`stop`
: To stop CUE User Manager.

`restart`
: To restart CUE User Manager.

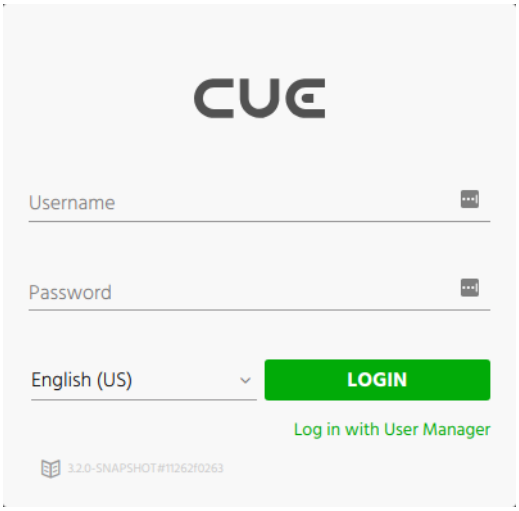
`status`
: To list status information about CUE User Manager.

3 Daily Use

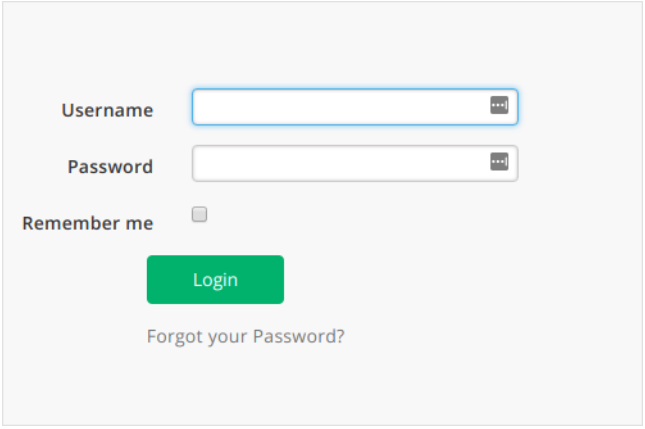
3.1 Logging in to CUE

Test & staging environments

When CUE User Manager is available, CUE Editor will add a link to the login screen that says **Log in with User Manager**:



This takes you to the IAM system's login page, where you then enter your login credentials (Active Directory credentials, for example):



After logging in here, you will be returned to CUE.

Production environments

A link on the login screen is fine for test environments or when you want to have the possibility to login using users that only exist in CUE Content Store, however for production environments, we would

normally recommend that the user is taken directly to the IAM system's login screen, rather than first having to click on a link in the CUE Editor login screen.

To make CUE Editor go directly to the IAM login screen, set the CUE Editor setting **`redirectToLoginPage`** to **`true`**, see section 4.4.

4 Configuration files

4.1 Configuring CUE User Manager

4.1.1 user-manager.yaml

The User Manager Java application itself is configured with **`/etc/escenic/user-manager/user-manager.yaml`**. This section will describe the most important settings, for a full list of available options, you can read the file and the comments therein.

Configuration items under the **`provider`** YAML block:

`type`
The provider type. If the IAM backend doesn't support SCIM, the IAM backend **`type`** can be set to **`keycloak`**.

To use **`type: keycloak`**, you must set up the OIDC client to include user groups in a custom OIDC claim and that this claim must be named **`groups`**. See the Keycloak documentation for more on configuring the OIDC client to include such a group mapper.

`oidcEndpoint`
The OIDC discovery endpoint of your IAM, e.g.: **`https://iam.example.com/.well-known/openid-configuration`**

`clientId`
The id of the OIDC client you've created in the IAM system. CUE User Manager uses this when authenticating with the IAM system.

`clientSecret`
The secret of the OIDC client you've created in the IAM system.

`pkce`
To make the communication with the IAM backend even more secure, CUE User Manager can use PKCE instead of a client secret when communicating with the IAM system.

PKCE must first be configured for the OIDC client in the IAM backend before CUE User Manager can use it. Please check the documentation for your IAM on the details on how to do this. Then, you can enable PKCE in CUE User Manager and optionally remove the **`clientId`** since it will not be used when PKCE is enabled.

```
pkce:
  enabled: true
```

Note, currently PKCE doesn't work in Azure AD for **`Web`** applications (only single page apps). This means it cannot be used with CUE User Manager.

`redirectURI`
Redirection URL after login through OIDC. This needs to be same as you configured in the IAM system when creating the OIDC client and should reference where CUE User Manager's **`/callback`** resource.

Example: **`https://um.example.com/callback`**

adminClient

Keycloak needs this configuration to be able to expose an endpoint for listing users in the IAM backend with **GET https://um.example.com/user?q=john**. For this to work, CUE User Manager needs access to the IAM's admin API. This is a different OIDC client from the regular one UM uses to log users in.

Example:

```
adminClient:
  userName: admin
  password: d58_9QMg8wpb$Wj
```

- **userName** the user name (required)
- **password** the password (required)
- **oidcEndpoint** The OIDC endpoint of the admin client. The default is **http://localhost:8080/auth/realms/master/.well-known/openid-configuration** which is the default Keycloak admin. If you use Keycloak and it runs on the same machine as CUE User Manager, you don't need to change this, otherwise, it's required.
- **clientId** the admin OIDC client. The default is **admin-cli**, which is the default on Keycloak (optional)
- **grantType** the grant type for the admin client. The default is **password**, which is the default on Keycloak (optional)

Currently, listing all users works only with Keycloak and Azure AD.

allowedOrigins

Which systems are allowed to access CUE User Manager? You will typically have one entry for each of CUE Editor, CUE DAM and CUE Print

Example:

```
allowedOrigins:
- https://editor.example.com
- https://dam.example.com
- https://print.example.com
```

scopes

Configurable OIDC scopes for various types of IAM requests. By default:

- The **authorizationCode** OIDC endpoint contains scopes called **openid**, **profile** and **email**. Any scopes you specify here will be merged with existing scopes.
- The **userInfo** OIDC endpoint contains scopes called **profile**, **email**. So, for example, if you want to be able to get user groups from the IAM system, then the **userInfo** endpoint would need to have an additional scope called **groups**.
- The **refresh** token endpoint contains a scope called **openid**. Some IAM systems such as AzureAD require the scope **offline_access** to be supplied in **refresh** token requests.

userNameFilter

A regular expression used to extract the required portion of the user name string supplied by the IAM system. The expression **(.*)@domain.com**, for example would extract only **john.smith** from the supplied user name **john.smith@domain.com**.

Some IAMs (Azure AD, for example) return user names that include a domain name suffix, whereas others (Keycloak, for example) do not. usernameFilter makes it possible to remove this difference, thereby reducing migration issuesg.

scimEndpoint
The SCIM2 endpoint of your IAM, e.g.: **https://iam.example.com/scim/v2/**

imdg
Running an IMDG (in-memory data grid) allows you to run multiple CUE User Manager with the IMDG acting as a shared level three cache. If one of the CUE User Manager instances cannot find a cache entry locally, it will look it up in the distributed memory cache.

Currently, the following IMDGs are supported:

- Hazelcast

Example configuration:

```
imdg:
  type: hazelcast
  baseURI: https://cache.example.com/hazelcast/rest
```

acl
When enabling file based access control lists (section 7.12), you have the following options:

- **hotReloadEnabled**: If set to **true**, CUE User Manager will read all the ACLs on each request. On systems with many files or entries, this will have an impact on performance.
- **emailDomains**: A list of email domains for which CUE User Manager will look for ACLs.

  You must either configure **emailDomains** or **sources** for ACLs to work.
- **sources**: A list of sources for which CUE User Manager will look for ACLs. By default, the **source** field on the user object will be that of the IAM provider, e.g. **keycloak**, so if this hasn't been overriden, then this will make CUE User Manager look up ACLs for **all** users, if you haven't also configured **emailDomains**.

  You must either configure **emailDomains** or **sources** for ACLs to work.
- **users**: Inline user access control list where you can provide the same entries as what the **acl.d** files have.

  Mostly suited for development and test systems, were you want to configure a couple of freelancers or for some other reason don't want dedicated ACL files.

Example configuration:

```
acl:
    enabled: true
    hotReloadEnabled: true
    emailDomains:
      - designers.example.com
      - gmail.com
    sources:
      - keycloak
```

groupSync
If you've set up your IAM system to sync from AD and it doesn't sync groups too, you can set this to **true** to have CUE User Manager create these users in the IAM system for you. The pre-requisite for this to work is that the groups are present in a string field called **userGroups** on the SCIM user object.

Newsroom publication mapping

CUE User Manager uses pattern matching to find the name of the newsroom and then utilizes predefined mappings between newsroom and publication to find out which roles to assign to the user in what publication.

Valid identifiers for groups are **${publication}**, **${newsroom}**, **${name}** and **${ignore}**. If the group identifier is **${ignore}**, the SCIM user objects must have the field **homePublication** set.

If the AD group name is **newroomx_journalist** where the name of the newsroom is **newsroomx** and the role the user should have in this newsroom is journalist, then the template should be **${newsroom}_${name}**.

```
newsroom:
  # publicationMapping:
    # sportsdesk:
        # Corresponding publication names
        # - football.com
        # - cricket.com
        # - golf.com
    # tabloid:
        # - beats.com
  ...
```

If using **publicationMapping**, set the template to:

```
${newsroom}_${name}
```

Default template is:

```
${ignore}_${name}
```

The mapping between the **name** fragment of the group and the roles this group should get in CUE Content Store is defined in the **roleMapping** block:

```
newsroom:
  ...
  roleMapping:
    reader:
      - reader
      - articleWithContentTypeReader
    journalist:
      - journalist
      - reader
      - articleWithContentTypeWriter
      - articleWithContentTypeReader
    editor:
      - editor
      - journalist
      - reader
      - articleWithContentTypeWriter
      - articleWithContentTypeReader
    admin:
      - publicationadmin
      - useradmin
      - administrator
      - editor
      - journalist
      - reader
      - articleWithContentTypeWriter
```

```
        - articleWithContentTypeReader
```

When AD user group naming is consistent and corresponding CUE User Manager mapping is correct, users will get correct permissions in CUE Content Store automatically, making AD not only the master source of users but also of authorization/access control to the different publications inside CUE Content Store.

It is also possible to define the home publication to which all newly-created users, including SSO users, will be assigned by setting the **`homePublication`** property:

```
newsroom:
  ...
  homePublication: beats.com
```

All content created by a user will by default be created in his/her home publication. A user can, however, override this initial default behavior by defining a personal profile containing default OU (organizational unit) and publication settings. Users cannot, however, change their home publication.

4.1.2 user-manager.conf

This file configures the **`/usr/bin/user-manager`** command. You will probably never need to edit this file, but if in case you want to, you edit this in the same way as user-manager.yaml (section 4.1.1).

4.2 Configuring web server

You need a web server/reverse-proxy in front of CUE User Manager. It needs to take care of the following:

- Terminate the TLS connection
- Forward requests to CUE User Manager

  When forwarding a request, the original request headers must also be forwarded so that CUE User Manager can generate correct URIs corresponding to the public facing web server. The headers **`Host`** and **`X-Forwarded-Proto`** are especially important.
- CORS configuration allowing all clients of CUE User Manager, such as CUE Editor, to communicate with it.

The URIs may be very long, as they contain OpenID Connect tokens. It is therefore important to configure the web server with a large enough buffer to hold such long HTTP request headers.

Here is an example based on the use of an nginx web server:

```
server {
  server_name  um.example.com;

  # TLS termination
  listen 443 ssl http2;
  ssl_certificate /etc/letsencrypt/live/um.example.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/um.example.com/privkey.pem;

  # CORS
  # enabling CUE served from editor.example.com to access UM
  if ($http_origin ~ '^https?://editor.example.com) {
```

```
    add_header "Access-Control-Allow-Origin" "$http_origin";
    add_header 'Access-Control-Allow-Methods' 'GET, POST, HEAD';
    add_header 'Access-Control-Allow-Headers' 'Content-Type';
  }

  # Reverse proxy, sending requests to UM
  proxy_set_header Host $host;
  proxy_set_header X-Real-IP $remote_addr;
  proxy_set_header X-Forwarded-Proto https;
  proxy_pass http://localhost:8680;
  proxy_buffer_size 128k;
  proxy_buffers 4 256k;
}
```

4.3 Configuring CUE Content Store to use CUE User Manager

Enable Content Store to use User Manager by editing **`/etc/escenic/engine/instance/engine1/com/escenic/auth/um/UMConfiguration.properties`**. The file itself is well commented and you should normally not have to configure more than these lines:

```
serviceEnabled=true
apiURI=https://um.example.com/openapi.json
```

If you have followed the installation and configuration instructions in section 7.7, this will point to where CUE User Manager listens for incoming connections. Note that this is different from **`https://iam.example.com`** which is where the IAM backend is.

4.4 Configuring CUE editor to use CUE User Manager

Edit **`/etc/escenic/cue-web/oauth.yml`** and comment out any **`oauth`** block that has been previously configured (e.g. to log on using Google or Facebook SSO).

With this change, CUE Editor will now display a link to CUE user manager in the login dialog. This allows user to choose to **either** log on using a local Content Store user **or** using the IAM backend.

Once the testing of User Manager and IAM backend is done, though, most users would want to be taken directly to the IAM backend's login page when needed and not first be prompted by the CUE login dialog. To get this behaviour, add the following to **`oauth.yml`**:

```
redirectToLoginPage: true
```

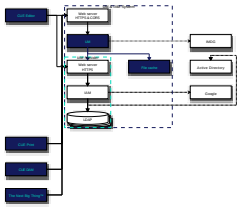
After editing **`/etc/escenic/cue-web/oauth.yml`**, run the following command to apply the configuration:

```
# dpkg-reconfigure cue-web
```

CUE Editor should now redirect the user to the IAM backend for authentication whenever needed.

See the CUE User Guide for further information on how you configure CUE Editor.

5 Architecture



The blue dotted line defines what's the full CUE User Manager solution, whereas the green dotted line defines what's a part of the IAM backend which UM uses.

The green boxes are the CUE User Manager components.

`CUE Editor`
The CUE Editor running in a web browser.

`CUE Print`
Editorial system for print.

`CUE DAM`
Digital asset management system.

`Web server`
Web server listening on port **`80`** and **`443`** (**`https`**). The web server in front of UM must have CORS set up, see section 7.6.

`UM`
CUE User Manager. A Java micro service. Provides a REST interface which CUE Editor, CUE Content Store, CUE DAM and CUE Print use to authenticate users and get user information.

`IAM`
Server having endpoints for OpenID Connect Core 1.0, OpenID Connect Discovery 1.0 and SCIM 2 (optional)

`LDAP`
Internal user/group storage for the IAM system. This can be used in cases where you don't have an external AD or where you want to have certain users, for instance freelancers, in a separate user database from that of company editorial users.

`AD` (optional)
Active Directory. Many customers choose to have the actual users and groups in an external identity provider like AD. This can of course be any other LDAP backend as well. It's the IAM that knows how to delegate password challenges and user and group lookup to AD.

`IMDG` (optional)
In-memory data grid. Running an IMDG allows CUE User Manager to use it as a level three cache. A CUE User Manager is thereby not dependent on finding its user session data in its local file storage.

`Google` (optional)
Using Google as an identity provider.

6 Diagnostics

CUE User Manager provides diagnostic information about itself that you can use to monitor the application's status. The diagnostic information is available in the form of JSON data that can be retrieved from two web application endpoints.

6.1 Health Check

CUE User Manager provides a "health check" at the following URL:

```
https://umhost:8681/healthcheck
```

where *umhost* is the host name or IP address of the machine on which you installed CUE User Manager. If you are logged in on that machine, then you can get nicely formatted output on the command line as follows:

```
$ curl https://localhost:8681/healthcheck | jq .
```

The health check provides information about the current state of both the CUE User Manager itself and the associated IAM manager. For example:

```
{
 "deadlocks": {
   "healthy": true
 },
 "provider-1-gluu": {
   "healthy": true,
   "message": "Connected"
 },
 "provider-1-stateoriginmappercache": {
   "healthy": true,
   "message": "CacheState(size=1)"
 }
}
```

6.2 Metrics

CUE User Manager provides a range of application metrics at the following URL:

```
https://localhost:8681/metrics
```

where *umhost* is the host name or IP address of the machine on which you installed CUE User Manager. If you are logged in on that machine, then you can get nicely formatted output on the command line as follows:

```
$ curl https://localhost:8681/metrics | jq .
```

The information provided at the **`metrics`** endpoint includes:

- System memory: total and free system memory, in KB

- System uptime: both server uptime and application context uptime
- JVM memory usage: initial, used and maximum available heap and non-heap sizes
- Thread statistics: total thread count, number of threads started and number of daemon threads
- Classloader: total classes loaded, number currently loaded and number unloaded
- Garbage collector: garbage collection algorithm, number of times run and time taken on last garbage collection run
- HTTP connections: total number of connections and total active connections
- Data source
- Timers: total, maximum and minimum duration of HTTP requests (all types)

Most of the information in the response is fairly easy to understand and provides a detailed view of the CUE User Manager's current performance.

7 IAM Integration Guide

This chapter will outline the different components of an IAM system that CUE User Manager needs to function properly: OpenID Connect Core 1.0 (authentication), OpenID Connect Discovery 1.0 (service discovery), System for Cross-domain Identity Management 2 (SCIM 2) (identity management) and User-Managed Access 2.0 (UMA) for web resource authorization.

The chapter will also describe the typical scenario of having users in an LDAP server, typically AD and using these users in the IAM, CUE User Manager and ultimately CUE Content Store, CUE Print and CUE Editor.

7.1 OpenID Connect

OpenID Connect (OIDC) is ubiquitous these days and must be supported in the IAM system you want to use. Note that OpenID Connect is not the same as OpenID 1.0 or OpenID 2.0. The IAM system must also support OpenID Connect Discovery 1.0 which gives you a URL which lets a client discover all the OIDC related services, including where to request an access token and where to refresh it.

Granted that the IAM system supports these two standards, CUE User Manager should be able to use it by entering the service discovery URI in **`/etc/escenic/user-manager/user-manager.yaml`**:

```
provider:
  oidcEndpoint: https://iam.example.com/oidc-discovery
```

For further details on the options in **`/etc/escenic/user-manager/user-manager.yaml`**, see section 4.1.1.

7.2 System for Cross-domain Identity Management 2 (SCIM)

System for Cross-domain Identity Management 2 (SCIM) is a standard for identify management. It describes the various resources (user, group++) and a REST interface to on these. CUE User Manager requires the IAM system to support version 2 of SCIM.

CUE User Manager needs the following extensions must be present in the SCIM user schema:

- userGroups (multi value **`string`**)
- homePublication (**`string`**)

Granted that the IAM system supports SCIM and that a valid OIDC access token passed in the request grants access to the REST API, CUE User Manager should be able to use it by entering the SCIM2 REST endpoint URI in **`/etc/escenic/user-manager/user-manager.yaml`**:

```
provider:
  scimEndpoint: https://iam.example.com/scim/v2/
```

7.3 User-Managed Access 2.0 (UMA)

By default, CUE User Manager assumes the the SCIM web resources are UMA protected:

```
provider:
  umaProtectedSCIM: yes
```

When enabled, CUE User Manager will perform the UMA authorization flow before making the actual SCIM HTTP request.

It's up to the IAM provider to validate that the access token is valid.

If the SCIM endpoint is **not** UMA protected, you must set this to **`false`**. The SCIM endpoint will then receive requests from with CUE User Manager with **`Authorization`** headers like:

```
Authorization: Bearer <access-token>
```

Again, it's up to the IAM system to validate that the access token is valid before allowing the client to perform the requested SCIM operation.

7.4 AD synchronisation

A nice feature of CUE User Manager is that you can instantly login into CUE Editor using the users you have in AD. For this to work, the users and their groups must be synced from AD to the IAM system. CUE User Manager will then in turn on demand, create these users and groups inside CUE Content Store if they don't already exist.

How the IAM systems copies users and groups from AD is system dependent and you need to look into the appropriate documentation on this. Typically, you want the IAM system to pull the data from AD, leaving AD oblivious to the existence of the IAM system.

If the IAM doesn't sync groups

Some IAMs, like Gluu, will not sync the groups from AD, in which case you must ensure the group IDs are present as a list of strings in the SCIM user object's **`userGroups`** field. This field is a custom extension that you must add to your SCIM user model, see section 7.2.

You must be sure to set up the AD synchronisation to map from AD's memberOf attribute to the IAM LDAP user's **`userGroups`** field and that this field is of a multi value string type.

CUE User Manager can now create native IAM user groups using this **`userGroups`**field, see the groupSync configuration in user-manager.yaml (section 4.1.1).

7.5 AD Authorization

Authorization through AD, i.e. that the user can use his AD password to login to CUE Editor, is in the domain of the IAM system and its configuration. All such systems will all have a way of forwarding the authentication challenges to AD. But again, this configuration is system specific and you must read the IAM vendor's documentation on this topic.

7.6 Web server CORS setup

A web server configured to provide CORS (Cross-Origin Resource Sharing) must be placed in front of CUE User Manager to enable communication with CUE Editor.

Please refer to the documentation of your web server of choice for general information on how to enable CORS.

In order for CUE User Manager to be able to communicate with CUE, the web server's CORS configuration must ensure that:

- POST requests include the response header Access-Control-Expose-Headers and this header includes **`X-CUE-UM-Access-Token`**
- OPTIONS requests include the response header Access-Control-Allow-Headers and this header includes **`X-CUE-UM-Access-Token`**

7.7 Secure all endpoints

All LDAP and HTTP endpoints that make up the CUE User Manager & IAM system should be protected with TLS. It's worth creating a good strategy for how to create and maintain such certificates for all environments, including development, test and staging. This way you can ensure that all environments are as production-like as possible. Turning off CUE User Manager in non-production systems is not advisable as problems with login, single sign on and authorization will then first be discovered in production.

As a starting point the following endpoints should be secured with TLS:

- The ODIC discovery document and all URIs it lists: **`https://iam.example.com/.well-known/openid-configuration`**
- The SCIM endpoint and all its URIs: **`https://iam.example.com/scim/v2/`**
- CUE User Manager itself: **`https://um.example.com`**
- The LDAP server holding the master source of users and groups. In many cases, this will mean Active Directory.
- The IAM system will have its own storage of users, and the communication between the OIDC and SCIM services and that storage should also go over TLS.

  For instance, Gluu's internal storage for users and application configuration is an LDAP server. This server has its own TLS certificate to ensure that communication between the SCIM component (called **`oxTrust`** or **`identity`**) and the OIDC component (called **`oxAuth`**) and this storage backend can be performed securely.

7.8 Two factor authentication (2FA)

Two factor and multi factor authentication are both in the sole domain of the IAM system and has nothing to do with the CUE User Manager integration. If the IAM system is set up to use 2FA, it will do so regardless of what CUE User Manager tells it.

7.9 Google SSO

Single sign on using the user's Google identification is in the sole domain of the IAM system and has nothing to do with the CUE User Manager integration. If the IAM system is set up to use Google OIDC login , it will do so regardless of what CUE User Manager tells it.

That said, many IAMs, like Gluu, will support users logging in using **either** an AD user **or** a Google user. This means it's possible to separate users, for instance, having in-house editorial users in Active Directory, while freelancers use their Google login.

When the user logs in through Google SSO, a (shadow) user is created inside the IAM system with a link to the Google identity, e.g. in Gluu, the LDAP entry for the user logged in over Google OIDC contains an attribute like **`oxExternalId=gplus:4112343241234`**. Before this user can do anything in CUE User Manager it must be assigned groups matching the newsroom publication mapping configuration that you've configured in **`/etc/escenic/user-manager/user-manager.yaml`**. CUE User Manager uses this to create the appropriate user in CUE Content Store with roles granting it permissions to **do** something. How these groups are added is in the realm of the IAM system. Preferably, the IAM system allows the integrator to hook onto the IAM system's user creation process and add the necessary groups immediately after the user logs in through Google, ensuring a smooth single sign on experience.

7.9.1 Google SSO security considerations

Turning on Google SSO in the IAM system allows any Google user to log into the IAM. If a default home publication has been set in user-manager.yaml (section 4.1.1), then Google users are also allowed to log into the CUE Editor, but they are not allowed to read or write any content, since they have no roles or permissions in the CUE Content Store.

```
  newsroom:
    # default publication
    homePublication: news
```

If the default home publication has **not** been set in user-manager.yaml (section 4.1.1) and no groups have been assigned to the SSO user in the IAM, SSO users will not be able to log in to the CUE Editor. An error stating **The user does not exist on the backend** is displayed in the CUE Editor and the following message is written to the log:

```
Caused by: neo.xredsys.api.IllegalOperationException: No home publication set for user
 john@example.com', can't auto create it
```

7.9.1.1 Limiting unwanted users in database

Although new SSO users cannot read or write content in the CUE Editor by default, they may still result in the creation of many unwanted users in the IAM system. The IAM backend doesn't normally have a way to specify **which** Google users to allow into the system, it's an all-or-nothing approach. To avoid the creation of unwanted SSO users, you can limit Google SSO access by domain name (for Google Workspace users), IP or geo-location. Such filtering would typically be implemented in the web layer before the request hits the IAM. How to set such filtering up is outside the scope of this document.

These SSO users will also be created in the CUE Content Store database the first time they try to log in. This automatic creation of SSO users in the database can be turned off by adding the following setting:

```
autoCreateUser=false
```

to **/etc/escenic/engine/instance/engine1/com/escenic/auth/um/UMConfiguration.properties**.

If **autoCreateUser** has been turned off, ten all Google SSO users must be explicitly created in the CUE Content Store before they can log in using Google SSO. This will still provide a single sign on solution, albeit not single **sign up**.

7.9.1.2 Creating groups for SSO users

As mentioned in an earlier section, SSO users are not assigned any groups by default. Without groups in the IAM, a user will not be assigned any roles or permissions in the CUE Content Store and can therefore not read or write any content.

As an IAM admin, you can assign these groups manually after a user has logged in for the first time. The IAM group names must follow the pattern described under the heading **Newsroom Publication Mapping** in user-manager.yaml (section 4.1.1). Alternatively (assuming your IAM provides the necessary extension points), you can create a custom hook to automatically assign groups to new SSO users. Please check your IAM documentation to determine whether and how such a hook may be added.

A Google SSO user cannot log into the CUE Editor and create editorial content until assigned to a group in this way.

7.9.1.3 Revoking access

Revoking an SSO user's access is simply a matter of removing the user's groups in the IAM admin interface and invalidating its IAM session.

The next time the user tries to log in, or the CUE Editor asks for a new access token to keep the session alive, the user will be re-directed the IAM login screen since the session has been invalidated. After successfully logging into the IAM (via the SSO provider), the user will no longer have access to the CUE Editor since the user no longer has any groups granting it editorial rights.

7.9.1.4 Two factor authentication

CUE User Manager is not directly involved in setting up two factor authentication – it is the users' responsibility to do this in his/her Google account. The IAM system only sets up Google SSO and the user is redirected to Google's login page on which the configured authentication scheme is followed.

If the Google user is a company account governed by Google Workspace, then the company administrator can enforce two factor authentication using the Google Workspace admin interface.

7.10 Migrating existing Escenic users

In cases where you have an existing user base in the CUE Content Store database and these don't exist anywhere else (i.e. the CUE Content Store database is the master source for these users), you can migrate these to the IAM system using CUE User Manager's REST interface. This will typically be a one time job to bootstrap the system.

The CUE User Manager is documented "live" using the OpenAPI Specification and is accessible on your CUE User Manager system under the URI **`https://um.example.com/openapi.json`**. There's also an interactive web interface through which you can experiment with the API under **`https://um.example.com/swagger-ui/`**

Here's an example of creating a user using CUE User Manager's REST interface:

```
$ curl \
  --request POST \
  --header 'Content-type: application/json' \
  --header 'Authorization: Bearer <access-token>' \
  --data '{
     "userName": "lisa",
     "givenName": "Lisa",
     "familyName": "Doe",
     "displayName": "Lisa Doe",
     "email":"lisa@example.com"
    }' \
  https://um.example.com/user
```

The user data can be exported from the CUE Content Store in several ways. Check out the documentation for the version you have installed at: cuedocs.escenic.com. Once you have them on a machine readable format, you can write a script to create these users in the IAM system by using the mentioned CUE User Manager REST interface. These users will be created in the IAM system through the SCIM2 interface. If these users then are to be further exported to Active Directory, the appropriate LDAP synchronisation must be set up. Again, this is in the realm of the IAM system.

The code you write to migrate the users will need a long lived access token from the IAM system (since the code is not a person that can authorise requests in a web browser). You thus need to configure the IAM system to issue access tokens with a long TTL, e.g. 30 minutes and then obtain one on the behalf of your integration script by pointing your web browser at: **`https://um.mycompany/auth`** . At the end of the OIDC authentication exchange, you will see the access token in the URI in your browser. This token can then be used in requests to the **`/user`** endpoint of CUE User Manager to create the users and groups you need.

7.11 Identifying users by source and source-id

It is possible to have users in CUE Content Store identified by a unique source and source-id from the IAM system instead of a username. This makes it possible to change a user's username (login name), for example if the person changes name in real life, while still being the same person in CUE Content Store.

By default, CUE User Manager will generate source and source-id based on the name of the IAM provider and the subject identifier included in the id token returned from the IAM.

The source and source-id will be included in the JSON response from the **`https://um.example.com/user/me`** webservice like so:

```
{
  ..
  "source": "keycloak",
  "sourceId": "00u12efkyrMnlKKlL5d7"
  ..
```

```
}
```

It is possible to override the default values by extending the OIDC client configuration in the IAM backend to include a source and source-id in custom OIDC claims. The claims must be named **`source`** and **`source_id`**, and they must be included in the OIDC UserInfo endpoint.

7.12 Managing Freelancer Access

User groups control the user's access to the CUE system. These groups normally reside in the IdP backing the IAM, e.g. Active Directory. However, CUE User Manager also supports controlling user access by defining one or more YAML files in **`/var/lib/escenic/user-manager/acl.d`**. With this approach, the IAM still manages the **authentication** of the user, but the **authorization** is handled by the **`acl.d`** files.

If file based ACLs are used for a given user, then any groups that the user may have in the IAM/IdP, included in a custom groups claim, will be ignored.

The YAML files look like this:

```
 # Access control entry for a specific user
- email: john.doe@gmail.com
  groups:
    - living_journalist
    - sports_reader

# All users from design.agency.com
- email: @design.agency.com
  groups:
    - fashion_reader
    - clothes_journalist
```

There can be multiple YAML files allowing grouping of users that make sense for the system admin:

```
/var/lib/escenic/user-manager/acl.d/freelancers.yaml
/var/lib/escenic/user-manager/acl.d/summer-interns.yaml
/var/lib/escenic/user-manager/acl.d/youtubers.yaml
```

See user-manager.yaml (section 4.1.1) on how to enable file based access control lists.

8 Security Considerations

This section outlines some of the security issues to consider when deploying an IAM/CUE User Manager solution. If you are using Google SSO, then you should also read section 7.9.1.

8.1 IAM Firewall

If you are hosting your own IAM, then place a firewall in front of it. Only CUE User Manager needs to make OIDC requests to the IAM. All UM clients, including CUE Content Store, CUE Editor and CUE Print make authentication, authorization and user identity requests to CUE User Manager only. The firewall must therefore be configured to:

- Block incoming requests to all the OIDC endpoints listed in your OpenID service discovery documents that originate from any client other than CUE User Manager.
- Block end user access to every part of your IAM except the login pages.

8.2 OAuth2 Best Practices

Read OAuth 2.0 Security Best Current Practice.

The most important items to consider are:

- If you customize your IAM landing page, be very careful to ensure that it does not leak access tokens.
- Aim for long-lived refresh tokens and short-lived access tokens. It is also a good idea to make the IAM invalidate the old refresh token and create a new one when the refresh endpoint is requested.
- Even though CUE Editor receives access tokens as URI fragments during logins, XSS exploits cannot use them to create a new token since the client id and client passwords needed to request new access and refresh tokens are only stored on the CUE User Manager server.

  However, an XSS exploit can use the access token to access the IAM system on behalf of the logged in user. It's therefore recommended to tighten the section 8.1.

8.3 Logging and Developer Mode

When running CUE User Manager in production, **`developerMode` must not** be set to **`true`**. If you run CUE User Manager in developer mode with **`DEBUG`** logging selected, CUE User Manager logs **`curl`** commands to represent what it is doing behind the scenes. These **`curl`** statements contain complete access tokens and anyone with read access to the server logs can therefore make requests to the IAM system on behalf of any user.