CUE Content Store

Installation Guide

7.5.12-2

Table of Contents

1 Introduction 4
1.1 Installation Components 4
1.1.1 Escenic Components 4
1.1.2 Third-Party Components 4
1.2 Installation Types 5
1.3 Conventions Used in This Manual 6
1.3.1 Identifying Host Machines 6
1.3.2 Shell Commands 7
1.3.3 Standard File Hierarchy 7
2 Product Requirements 9
2.1 Required Supporting Software 9
2.2 Other Requirements 9
2.2.1 Shared Filesystem 9
2.2.2 SSH/FTP Server 10
2.2.3 Apache Ant 10
3 Installation Procedure 11
3.1 Preparation 12
3.1.1 Install Java SE Development Kit (JDK) 12
3.1.2 Install Ant 12
3.1.3 Install Various Utilities 13
3.1.4 Create CUE Users 13
3.1.5 Create Shared File System 13
3.1.6 Install Database 15
3.1.7 Install Application Server 16
3.1.8 Logging 19
3.1.9 Install Solr 19
3.2 Installation 22
3.2.1 Install CUE Packages 22
3.2.2 Initialize the Assembly Tool 23
3.2.3 Initialize the Bootstrap Layer 23
3.2.4 Edit the Common Configuration Layer 24
3.2.5 Create Host Configuration Layers 25
3.2.6 Configure the ece Script 26
3.3 Finishing Up 27

3.3.1 Assemble and Deploy 27
3.3.2 Verify the Installation 28
3.3.3 Install a Daemon Script 29
3.3.4 Create a Publication 30
3.3.5 Test Web Studio 32
4 Recommended Configurations 34
4.1 A Development/Test Configuration (HTTP) 35
4.1.1 Content Engine / Tomcat Configuration (HTTP) 35
4.1.2 CUE Configuration (HTTP) 35
4.1.3 SSE Proxy Configuration (HTTP) 36
4.2 A Production Configuration (HTTPS) 37
4.2.1 Content Engine / Tomcat Configuration (HTTPS) 37
4.2.2 CUE Configuration (HTTPS) 37
4.2.3 SSE Proxy Configuration (HTTPS) 38
4.3 CORS Configuration 39
5 Recommended Modifications 41
5.1 Search Engine Deployment and Configuration 41
5.2 Password Protect escenic-admin 43
6 Upgrading 45
6.1 Database Upgrades 45

1 Introduction

This **Installation Guide** is intended to be read by the person responsible for installing the CUE Content Store on one or more hosts. It should be read together with the Server Administration Guide, which contains descriptions of the periodic administration tasks a system administrator needs to carry out once the Content Store is installed and in operation, since there is considerable overlap between installation and maintenance of the Content Store.

Both this manual and the **Server Administration Guide** make the following assumptions about the CUE installation and you, the reader:

- The Content Store is to be installed on one or more host computers running an operating system listed in section 1.1.2.
- The supporting software components (database, web server, application server and so on) listed in section 1.1.2 are either already installed on these computers, or available for installation.
- You are a suitably qualified system administrator with a working knowledge of both the operating system on which the Content Store is to be installed and of the components in the supporting software stack.

1.1 Installation Components

A working CUE Content Store installation depends on many components. Some of these components are delivered by Stibo DX as part of the product, others are third-party products on which the Content Store depends.

1.1.1 Escenic Components

The components supplied by Stibo DX are:

CUE Content Store
The Content Store itself, a content management system.

escenic-admin
A server administration web application.

Escenic Web Studio
A publication administration web application.

Publications
The installation includes a few small demo publications for test, demonstration and instructional purposes.

Assembly tool
The assembly tool is an ant build file that automates the application assembly and deployment process.

1.1.2 Third-Party Components

The Content Store depends on the following third-party components:

Copyright © 2008-2021 Stibo DX A/S

Java Development Kit (JDK)
This provides the Java runtime and development environment required by J2EE application servers.

Database
Used to store publication content, structure and so on.

J2EE Java application server
The Content Store is a JEE web application, and therefore runs inside a JEE application server. `escenic-admin`, Web Studio and all CUE publications are also JEE applications and therefore also run within a JEE application server.

Web server (optional)
J2EE applications have built-in web servers that can be used for test and development purposes. For large-scale production installations, however, a separate web server may be advisable.

JDBC (Java DataBase Connectivity) driver
This is required to connect the Content Store to the database.

Apache Ant
This is a Java-based build tool that is used by the Content Store assembly tool.

Apache Solr
This is a Java-based search engine. It is used to provide search functionality both in web publications and in CUE.

Caching Server (optional)
A caching server such as Squid or Varnish can greatly improve the performance of heavily-loaded sites by caching frequently requested pages and sending the cached copies on request unless the original pages have been updated.

Load Balancing Server (optional)
If your site has multiple web hosts, then you may also want to install a load-balancing server to distribute requests evenly between them.

memcached (optional)
If your site has multiple web hosts, then you can improve caching efficiency by using this open source distributed cache manager to provide a shared cache for all your hosts.

In principle, the Content Store can work with many different third-party components. In section 2.1, however, you will find the list of officially supported software. The current version of the Content Store has been tested with these components and is known to work satisfactorily.

1.2 Installation Types

The Content Store can be installed in a number of different ways to satisfy different needs. Common installation types include:

Single computer
Everything (database, application server, Content Store) is installed on a single computer. This is usually the case for demonstration and developer installations.

Multiple application servers, single database
The application server and Content Store is installed on multiple computers, but all share a single database server. This kind of installation is common in large development environments.

Copyright © 2008-2021 Stibo DX A/S

Single web server, single application server and Content Store, single database
This is a common production installation structure for small sites.

Multiple web servers, multiple application servers/Content Stores, database rack
This is a common production installation structure for large sites, and is easily scalable.

Other configurations are, of course, possible.

In large organizations the Content Store often needs to be installed in several different configurations in order to meet the requirements of different environments:

Development
In a development environment there are typically several updates of several files several times a day. Depending on the number of developers working on a system and the potential for conflicting changes, developers may either:

- Share a single development installation, or
- Each use a personal Content Store installation but share a single database.

Test
A test environment is a complete Content Store installation on which 'current stable versions' can be tested. Code updates in this environment are typically carried out at longer intervals and in 'batches' rather than on the file level. In some organizations, the test environment may be merged with the staging environment.

Staging
A staging environment should be as similar to the production environment as practically possible: it should have the same software, the same configuration, and if possible the same hardware, firewall and network setup. This is the last stage before actual production, and is intended to catch "works on the test server but not in production" problems.

Production
A production environment is heavily optimized for high workloads and security.

1.3 Conventions Used in This Manual

This section describes various conventions used in this manual.

1.3.1 Identifying Host Machines

The Content Store and the software it depends on may need to be installed on one or several host machines depending on the type of installation required (see section 1.2). In order to unambiguously identify the machines on which various installation actions must be carried out, the following special host names are used throughout this manual:

assembly-host
The machine used to assemble the various Content Store components into a enterprise archive or .EAR file.

database-host
The machine used to host the database server.

engine-host
The machine(s) used to host application servers and Content Store instances.

Copyright © 2008-2021 Stibo DX A/S

editorial-host
engine-host(s) that are used solely for (internal) editorial purposes. All your CUE clients communicate with these hosts.

presentation-host
engine-host(s) that are used solely for (public) presentation purposes. Readers' browsers communicate with these hosts.

web-host
The machine(s) used to host web servers (if needed).

share-host
The machine used to host the NFS server (if needed).

These host names always appear in a **bold typeface**. If you are installing everything on one host you can, of course, ignore them: you will be doing everything on the same machine. If you are creating a larger multi-host installation, then they should help ensure that you do things in the right places.

1.3.2 Shell Commands

All shell command examples given in the manual are tested on Debian Linux servers: they may need minor modifications to be used on other Linux or UNIX platforms, and it is assumed that you are able to make the necessary "conversions" to your own platform. Some of the commands should be executed as the owner of the Content Store installation. This is signalled by use of the **$** command prompt. For example:

```
$ ls
```

Other commands must be executed as **root**. This is signalled by the use of the **#** command prompt:

```
# /etc/init.d/slapd restart
```

Many of the code and command line examples in the manual contain placeholders: words printed in an *italic* typeface, which you are expected to replace with an appropriate value. For example:

```
$ cat filename
```

The accompanying text usually specifies what kind of value any placeholder(s) represent: *filename* in this case is the name of a file you want to see the contents of.

1.3.3 Standard File Hierarchy

All file paths and URLs shown in the manual are based on the following standard folder structure.

Standard location	Component
/usr/share/escenic	CUE
/usr/share/escenic/escenic-content-engine-7.5.12-2	CUE Content Store
/usr/share/escenic/escenic-assemblytool	CUE assembly tool
/usr/share/escenic/ece-scripts	CUE scripts

Copyright © 2008-2021 Stibo DX A/S

Standard location	Component
`/etc/escenic`	CUE configuration
`/etc/escenic/engine`	CUE Content Store configuration
`/opt/java/jdk`	Java
`/opt/java/ant`	Ant
`/opt/tomcat-engine1`	Tomcat

If your system is organized differently, then adjust the paths you use accordingly.

Copyright © 2008-2021 Stibo DX A/S

2 Product Requirements

The CUE Content Store is written in Java, runs on an application server, and uses a database to store content and user data. One of each of these supporting products must be installed before the Content Store can be installed. The specific supporting products with which the Content Store is known to work are listed in section 2.1. Note, however, that these products have their own hardware and software requirements. Please make sure that your environment meets the requirements stipulated by the relevant product suppliers.

Some additional, more general requirements and recommendations are listed in section 2.2.

2.1 Required Supporting Software

The Content Store requires a combination of the following software components:

Operating System
One of the following:

- Ubuntu Server 18.04 LTS (Long Term Support)
- Red Hat Enterprise Linux 7.4

Application Server
Apache Tomcat 9.0.10

Database Server
MariaDB 10.1

Java Development Kit
One of the following:

- JDK 11 (recommended) - either OpenJDK 11 or Oracle Java SE Development Kit 11
- JDK 8 - either OpenJDK 8 or Oracle Java SE Development Kit 8

Web Server
One of the following:

- Nginx 1.14.0
- Apache 2.4.29

Search Engine
Solr 6.6.5

2.2 Other Requirements

2.2.1 Shared Filesystem

In large installations, the web server may well run on a different host to the application server. However, the web server and application server need access to a common file system, since the

Copyright © 2008-2021 Stibo DX A/S

Content Store writes and updates files that are served by the web server (multimedia files, for example). On Linux systems, NFS can be used to set up the required shared folders. For Further information, see section 3.1.5.

2.2.2 SSH/FTP Server

The Content Store does not need an SSH or FTP Server. Setting up an SSH or FTP server is, however, highly recommended. An SSH/FTP Server can be used to upload new and updated files to the server. If it is not possible to install an SSH or FTP server on the **engine-host** (for security reasons, perhaps), then some other way of uploading files must be provided.

2.2.3 Apache Ant

Ant is used to package an Enterprise Archive (EAR file) containing the Content Store and all the web applications required to run it.

Ant is only required on the **assembly-host**. It is not necessary to install ant on all of the hosts in a cluster.

Copyright © 2008-2021 Stibo DX A/S

3 Installation Procedure

This chapter contains step-by-step instructions for installing the Content Store on a single host computer or on a cluster of several host computers. The instructions use the host names listed in section 1.3.1 to indicate where you should carry out various steps. If some of your hosts are "multi-purpose" (if for example, your database server is installed on the same host as your editorial Content Store installation), then carry out all appropriate steps on that machine (for example, all **database-host** steps and all **editorial-host** steps).

Some of the steps are not required for single-host deployment. These steps are clearly marked.

The following placeholders are used in paths specified in the instructions:

Placeholder	Path
engine-installation	`/usr/share/escenic/escenic-content-engine-7.5`
assemblytool_installation	`/usr/share/escenic/escenic-assemblytool`

There are a number of preparatory tasks that you need to carry out, described in section 3.1. The main installation procedure is described in section 3.2. Post-installation tasks (assembling and deploying an EAR file containing the supplied CUE web applications, checking that everything is installed correctly and so on) are described in section 3.3.

A note about version code names

The CUE Content Store and all related applications (plug-ins, the CUE editor, CUE Print, CUE Front and so on) are released on a synchronized schedule where all product versions in a given release are known to work well together. Only these approved version combinations are supported. Each set of compatible product versions is identified by a code name, and during installation you can use this code name instead of the individual product's version number, thereby simplifying the installation process.

In the case of the Content Store and other Linux applications installed on Ubuntu using **`apt`**, the code name is actually the name of a repository containing compatible versions of all products. This means that in order to ensure version compatibility, all you need to do is add the name of the required repository to your **`/etc/apt/sources.list.d/escenic.list`** file. Once you have done this you do not need to specify any version numbers when installing individual packages - **`apt`** will just install the latest maintenance release from that repository.

> Note that code names cannot be used in this way on CentOS/Red Hat installations, where the application packages to be installed must still be identified by their version numbers.

The code name for Content Store 7.5 is **`nelson`**.

Copyright © 2008-2021 Stibo DX A/S

3.1 Preparation

First of all, it is necessary to install and configure the various software packages used by a CUE system, and carry out a few other preparatory tasks.

3.1.1 Install Java SE Development Kit (JDK)

On your **assembly-host** and **engine-host**(s), while logged in as **root**, download and install Java SE development kit (JDK), version 8.

You can download the JDK from http://www.oracle.com/technetwork/java/javase/downloads/index.html. Follow the installation instructions supplied on the Oracle site. When you have installed it you need to make sure that it is installed as the default Java installation. You can do this by entering the following commands while logged in as **root**:

```
# update-alternatives --install "/usr/bin/java" "java" "java-install-path" 1
# update-alternatives --set java java-install-path
```

where *java-install-path* is the path of the location in which you installed Java (for example, **/usr/lib/jvm/java-8-oracle/jre/bin/java**).

Verify that Java is correctly installed by entering:

```
# java -version
```

This should result in output something like this:

```
java version "1.8.0_11"
Java(TM) SE Runtime Environment (build 1.8.0_11-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.11-b03, mixed mode)
```

3.1.2 Install Ant

On your **assembly-host**, while logged in as **root**:

Download a binary distribution of **ant** from http://ant.apache.org/bindownload.cgi. Unpack the distribution to an appropriate place and follow the installation instructions given here:

http://ant.apache.org/manual/install.html#installing.

You also need to install the additional library **ant-contrib** (see http://ant-contrib.sourceforge.net/#install).

It may also be possible to download **ant** and **ant-contrib** from one of your Linux distribution's software repositories. On Debian-based Linux machines, for example, you can install it by entering:

```
# apt-get install ant-contrib
```

Verify that **ant** is correctly installed by entering:

```
# ant -version
```

This should produce a response something like this:

```
Apache Ant version 1.7.1 compiled on September 8 2010
```

Copyright © 2008-2021 Stibo DX A/S

3.1.3 Install Various Utilities

Make sure the following utilities are available on all your hosts:

- **`unzip`**
- **`telnet`**

On Debian-based Linux machines, you can easily install these by entering the following commands as **`root`**:

```
# apt-get install unzip
# apt-get install telnet
```

You may also find it useful to install an SSH server on all your hosts so that you can easily switch sessions and copy files between them. On Debian-based Linux machines, you can do this as follows:

```
# apt-get install openssh-server
```

3.1.4 Create CUE Users

All the Content Store components should run under the same user, and this user should have the same UID on all hosts. So create a user (preferably called **`escenic`**) on each host machine as follows:

```
# adduser escenic
```

and make sure they all have the same UID. If you are installing on clean machines, and **`escenic`** is the first user you create, then they will all have the same UID by default.

On your **assembly-host** and **engine-host**(s), make sure that Java is in the path of the **`escenic`** users:

```
# su - escenic
$ echo 'export JAVA_HOME=/usr/lib/jvm/java-8-oracle' >> .bashrc
$ echo 'export PATH=$JAVA_HOME/bin:$PATH' >> .bashrc
$ source ~/.bashrc
```

3.1.5 Create Shared File System

> If you are installing everything on a single host, you can skip this section.

If you are installing the Content Store on multiple hosts then it is usually a good idea to create some shared folders that can be accessed from all the hosts. If your **engine-host** is not the same machine as your **web-host**, then it is more or less a requirement, since the Content Store creates and modifies multimedia files that need to be accessed by the web server. (If you cannot create shared folders for some reason, then you need to find some other means of synchronizing files between the hosts.)

Shared folders can also be useful for other reasons, such as access to downloads or shared configuration files, and these instructions assume that you will use shared folders for these purposes.

The following instructions show how to set up three shared folders: **`ece-conf`**, **`multimedia`** and **`download`**.

On your **share-host**, while logged in as **`root`**:

Copyright © 2008-2021 Stibo DX A/S

1. Download and install the NFS server software. For example, on a Debian-based Linux distribution:

```
# apt-get install nfs-kernel-server
```

2. Create the folders to be shared and set **escenic** as their owner:

```
# mkdir -p /exports/ece-conf
# mkdir -p /exports/multimedia
# mkdir -p /exports/download
# chown escenic:escenic /exports/ece-conf /exports/multimedia /exports/download
```

3. Open the NFS exports file **/etc/exports** and add an entry for each folder that you want to share:

```
/exports/ece-conf   subnet-specification(rw,sync)
/exports/multimedia subnet-specification(rw,sync)
/exports/download   subnet-specification(rw,sync)
```

   where *subnet-specification* identifies the IP address range within which the shared folders are to be accessible. If, for example, all your hosts have IP addresses in the **192.168.71.***nnn* subnet, then you might enter the subnet specification **192.168.71.0/255.255.255.0**.

4. Restart all NFS-related services:

```
# /etc/init.d/portmap restart
# /etc/init.d/nfs-kernel-server restart
# /etc/init.d/nfs-common restart
```

5. Create symbolic links to the shared folders as follows:

```
# ln -s /exports/ece-conf /mnt/ece-conf
# ln -s /exports/multimedia /mnt/multimedia
# ln -s /exports/download /mnt/download
```

   This ensures that the shared folders will be accessible via the same paths on your **share-host** as on all the other hosts.

6. Change the owner of the links to **escenic**:

```
# chown escenic:escenic /mnt/ece-conf
# chown escenic:escenic /mnt/multimedia
# chown escenic:escenic /mnt/download
```

On all your other hosts, while logged in as **root**:

1. Download and install the NFS client software. For example, on a Debian-based Linux distribution:

```
# apt-get install nfs-common
```

2. Create mount points for the shared folders:

```
# mkdir /mnt/multimedia
# mkdir /mnt/download
```

3. Open **/etc/fstab** and add an entry for each shared folder:

```
share-host:/exports/multimedia /mnt/multimedia nfs defaults 0 0
share-host:/exports/download /mnt/download nfs defaults 0 0
```

   where *share-host* is the host name or IP address of your **share-host**.

4. Mount the shared folders:

```
# mount -a
```

Copyright © 2008-2021 Stibo DX A/S

5. Create a symbolic link to the multimedia shared folder from the **`filePublicationRoot`** path you will specify later in **`ServerConfig.properties`** (see section 3.2.4:

```
# ln -s /mnt/multimedia /var/lib/escenic/engine
```

3.1.6 Install Database

The following instructions describe how to install and set up MariaDB for use by the Content Store.

On your **database-host**, while logged in as **`root`**:

1. Install the MariaDB server and client packages. For example, on a Debian-based Linux distribution:

```
# apt-get install mariadb-server mariadb-client
```

2. Log in to the system and create a database for the Content Store:

```
# mysql -p
mysql> create database db-name character set utf8 collate utf8_general_ci;
Query OK, 1 row affected (0.00 sec)

mysql> grant all on db-name.* to user@'%' identified by 'password';
Query OK, 0 rows affected (0.00 sec)

mysql> grant all on db-name.* to user@'localhost' identified by 'password';
Query OK, 0 rows affected (0.00 sec)
mysql> exit
```

   Replace *db-name*, *user* and *password* in the above commands with a database name, user name and password of your choice.

3. If you are installing everything on a single host or if you created a shared file system as described in section 3.1.5, then you should have direct access to the Content Store package you downloaded earlier. If not, you will need to copy it to some temporary location on the **database-host**.

4. Change user to **`escenic`** and unpack the Content Store package to a temporary location:

```
# su - escenic
$ cd /tmp
$ unzip /mnt/download/engine-7.5.12-2.zip
```

5. Still as the **`escenic`** user, run the Content Store's database scripts:

```
$ cd /tmp/engine-7.5.12-2/database/mysql/
$ for el in tables.sql indexes.sql constants.sql constraints.sql; do \
    mysql -u user -ppassword db-name < $el
    done;
```

   Replace *db-name*, *user* and *password* in the above commands with the names you chose in step 2. Your *password* must **immediately follow** the **`-p`** switch, with no intervening space (as shown above).

6. On some platforms, external access to the MariaDB server is disabled by default. To enable external access, you need to bind the **`mysql`** process to the **database-host**'s IP address. To do this, you need to open **`/etc/mysql/my.cnf`** for editing (as **`root`** again) and set the **`bind-address`** parameter:

```
bind-address = database-host-ip-address
```

You should then verify that the MariaDB server is running and accessible by trying to connect to port 3306 from each of your other hosts using **`telnet`**:

Copyright © 2008-2021 Stibo DX A/S

```
$ telnet database-host 3306
```

where *database-host* is the host name or IP address of the **database-host**. If a connection is opened, then the database server is running and accessible.

On a single-host installation, you can check that MariaDB is running by entering the following command:

```
$ mysqladmin -u root -p status
```

You should get a response something like this:

```
Uptime: 605 Threads: 1 Questions: 615 Slow queries: 0 Opens: 842 Flush tables: 1 Open
 tables: 40 Queries per second avg: 1.16
```

If the server is not running then you will see an error reporting that it was not possible to connect to the server.

3.1.7 Install Application Server

The following instructions describe how to install and set up the Apache Tomcat application server for use by the Content Store.

On your **engine-host**(s), while logged in as **`escenic`**:

1. Download the latest Tomcat package from http://tomcat.apache.org/. If you are installing on more than one machine and have created the shared folders described in section 3.1.5, then it is a good idea to download these packages to the **`/mnt/download`** folder you created. Otherwise, download them to some temporary location of your choice.
2. Change user to **`root`** and unpack the Tomcat package to **`/opt`**:

   ```
   $ su
   # cd /opt
   # tar -zxvf /mnt/download/apache-tomcat-version-number.tar.gz
   ```

   where *version-number* is the version number of the package you have downloaded.
3. Create a symbolic link from **`/opt/tomcat-engine1`** to the Tomcat folder, so that it will be easier to upgrade to new versions when necessary, and change the owner of the Tomcat folder to **`escenic`**.

   ```
   # ln -s /opt/apache-tomcat-version-number /opt/tomcat-engine1
   # chown -R escenic:escenic /opt/apache-tomcat-version-number
   ```
4. Open Tomcat's configuration file file (**`/opt/tomcat-engine1/conf/catalina.properties`**) for editing and add the following characters:

   ```
   ,${catalina.base}/escenic/lib/*.jar
   ```

   to the end of the **`common.loader`** property setting.
5. Download the latest version of **Connector/J** (the MariaDB JDBC driver) from https://downloads.mariadb.org/connector-java/.
6. Change user back to **`escenic`** and install the driver by copying it to the **`/opt/tomcat-engine1/lib`** folder. If you downloaded the driver to **`/mnt/download`**, for example:

   ```
   # su - escenic
   $ cp /mnt/download/mariadb-java-client-version-number.jar /opt/tomcat-engine1/lib/
   ```

Copyright © 2008-2021 Stibo DX A/S

7. Create the **`/opt/tomcat-engine1/escenic/lib`** folder you added to the **`common.loader`** path in step 4:

```
$ mkdir -p /opt/tomcat-engine1/escenic/lib/
```

8. Set up database pooling by opening **`/opt/tomcat-engine1/conf/context.xml`** for editing and inserting the following resource definition as a child of the root **`Context`** element:

```
<Resource
  name="jdbc/ECE_DS"
  auth="Container"
  type="javax.sql.DataSource"
  username="user"
  password="password"
  driverClassName="org.mariadb.jdbc.Driver"
  maxTotal="200"
  maxIdle="10"
  maxWaitMillis="10000"
  removeAbandonedOnBorrow="true"
  removeAbandonedOnMaintenance="true"
  removeAbandonedTimeout="60"
  url="jdbc:mysql://database-host/db-name?
autoReconnect=true&amp;useUnicode=true&amp;characterEncoding=UTF-8"
/>
```

Replace *database-host*, *db-name*, *user* and *password* in the above definitions with the names you have defined earlier (see section 3.1.6).

Note that the **`maxTotal`** and **`maxWaitMillis`** attribute names above are only correct if:

- You are running Tomcat 8, **and**
- Your Tomcat installation uses the default Apache Commons DBCP (see http://commons.apache.org/proper/commons-dbcp/configuration.html for details).

If you are running Tomcat 7, then you will need to change **`maxTotal`** to **`maxActive`** and **`maxWaitMillis`** to **`maxWait`**. For general information about the differences between Tomcat 7 and Tomcat 8, see https://tomcat.apache.org/migration-8.html.

If you are running a Tomcat 8 that has been set up to use Tomcat's own DBCP, then you will also need to change **`maxTotal`** to **`maxActive`** and **`maxWaitMillis`** to **`maxWait`**. For information about Tomcat's DBPC, see http://tomcat.apache.org/tomcat-8.0-doc/jdbc-pool.html#Introduction.

The parameter settings shown above are provided only as recommended starting values. They may need to be adjusted to provide optimum performance.

> If you copy and paste the above resource definitions, make sure to remove any line breaks in the **`url`** values (caused by page width restrictions).

9. Set up indexing by inserting the following elements in **`/opt/tomcat-engine1/conf/context.xml`** as children of the root **`Context`** element:

```
<Environment
    name="escenic/indexer-webservice"
    value="http://indexer-web-service-host:8080/indexer-webservice/web-service-
name/"
    type="java.lang.String"
    override="false"/>

<Environment
    name="escenic/index-update-uri"
    value="http://localhost:8983/solr/solr-core/update/"
```

Copyright © 2008-2021 Stibo DX A/S

```
    type="java.lang.String"
    override="false"/>

<Environment
    name="escenic/solr-base-uri"
    value="http://localhost:8983/solr"
    type="java.lang.String"
    override="false"/>

<Environment
    name="escenic/head-tail-storage-file"
    value="/var/lib/escenic/head-tail.index"
    type="java.lang.String"
    override="false"/>

<Environment
    name="escenic/failing-documents-storage-file"
    value="/var/lib/escenic/failures.index"
    type="java.lang.String"
    override="false"/>
```

where:

- On an **editorial-host**, *indexer-web-service-host* is the host name or IP address of the **editorial-host** on which the Content Store's internal indexer web service is to run and *web-service-name* is **`index`**.
- On a **presentation-host**, *indexer-web-service-host* is the host name or IP address of the **presentation-host** on which the Content Store's external indexer web service is to run and *web-service-name* is **`presentation-index`**.
- The *solr-core* is usually either **`editorial`** or **`presentation`**.

If you are creating a single-host installation, then you can use the host name **`localhost`** and the indexer web service name **`index`**.

> Make sure that your **`escenic/solr-base-uri`** setting in **`/opt/tomcat-engine1/conf/context.xml`** is as shown, and **does not** include a trailing slash (**`/`**). If it does, then search will not work.

10. Open Tomcat's **`web.xml`** file (**`/opt/tomcat-engine1/conf/web.xml`**) for editing and insert links to the resources you have defined:

```
<resource-ref>
  <description>Escenic link</description>
  <res-ref-name>jdbc/ECE_DS</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>
```

These elements must be inserted as children of the root **`web-app`** element: otherwise, position is irrelevant.

11. Open **`/opt/tomcat-engine1/conf/server.xml`** for editing. Somewhere in this file you will find a **`Connector`** element that configures connections on port 8080. Make sure that this element contains a **`URIEncoding`** attribute, and that it is set to **`UTF-8`**. For example:

```
<Connector port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
```

Copyright © 2008-2021 Stibo DX A/S

```
            URIEncoding="UTF-8"/>
```

This ensures that Content Store search functionality works for non-Latin characters.

3.1.8 Logging

On one of your **engine-host**s, while logged in as **`root`**:

1. Create the folder **`/etc/escenic/engine/common`**:

   ```
   # mkdir -p /etc/escenic/engine/common
   ```

2. Create a logging configuration file called **`trace.properties`** in the new folder. This file should have the following content:

   ```
   log4j.rootCategory=ERROR
   log4j.category.com.escenic=ERROR, ECELOG
   log4j.category.neo=ERROR, ECELOG

   log4j.appender.ECELOG=org.apache.log4j.DailyRollingFileAppender
   log4j.appender.ECELOG.File=/var/log/escenic/engine/ece-messages.log
   log4j.appender.ECELOG.layout=org.apache.log4j.PatternLayout
   log4j.appender.ECELOG.layout.ConversionPattern=%d %5p [%t] %x (%c) %m%n
   ```

On all your **engine-host**s, while logged in as **`escenic`**:

1. Create the output folder specified in **`trace.properties`**:

   ```
   $ mkdir /var/log/escenic/engine
   ```

2. In order to make Tomcat use the **`trace.properties`** file, create a link to it from **`/opt/tomcat-engine1/lib`**:

   ```
   $ cd /opt/tomcat-engine1/lib
   $ ln -s /etc/escenic/engine/common/trace.properties
   ```

3. In order to make Tomcat output its log files to a standard Unix/Linux location, open **`/opt/tomcat-engine1/conf/logging.properties`** for editing and set **all** the log folder properties (they have names ending with **`FileHandler.directory`**) to **`/var/log/tomcat`**.

3.1.9 Install Solr

The Content Store distribution used to include a **`solr.war`** file that could be deployed together with the Content Store search. This is, however, not possible with more recent versions of Solr. Solr must now be installed as a standalone server on all your **engine** hosts.

To install Solr, log in as **`root`** on each **engine** host, and:

1. Download the latest stable release of Solr 6 from a suitable Apache mirror site, to a temporary location. For example:

   ```
   # cd /tmp
   # wget http://apache.uib.no/lucene/solr/solr-version/solr-solr-version.zip
   ```

   where *solr-version* is the latest stable Solr 6 release number.

2. Install the contents of the download package in **`/opt`**:

   ```
   # unzip /tmp/solr-solr-version.zip -d /opt
   ```

Copyright © 2008-2021 Stibo DX A/S

3. Create a symlink **/opt/solr** for the new installation folder:

```
# cd /opt
# ln -s solr-solr-version solr
```

4. Create the configuration and start-up files required to manage the Solr service, as described in section 3.1.9.1.
5. Create a Solr **core** by copying the Solr configuration files supplied with the Content Store to a suitable location on the host, as described in section 3.1.9.2.

> In Solr terminology, a **core** is the set of configuration files that define a specific application of Solr – a schema, an index and all related configuration files. A typical CUE production installation makes use of two Solr cores – one for editorial purposes and one for presentation purposes. For further information about this, see Customizing the Index Schema..

3.1.9.1 Create Solr Startup Files

To create the necessary Solr startup files:

1. Create a Solr core folder and **core.properties** file as follows:

```
#  mkdir -p /var/lib/escenic/solr/solr-core
#  cat > /var/lib/escenic/solr/solr-core/core.properties <<EOF
name=solr-core
config=solrconfig.xml
schema=schema.xml
dataDir=data
EOF
```

   where *solr-core* is the name of your Solr core. Typically you might choose to use either **editorial** or **presentation** as the name of your core depending on its purpose (that is, what kind of host you are installing it on).
2. Change ownership of the Solr folder and all its contents, so that it can be used by the **escenic** user:

```
# chown -R escenic:escenic /var/lib/escenic/solr
```

3. Create a configuration file for the **/opt/solr/bin/solr** startup script. Do this by **moving** (not copying) the file as follows:

```
# mv /opt/solr/bin/solr.in.sh /etc/default/
```

4. Append settings for **SOLR_HOME**, **SOLR_LOGS_DIR** and **SOLR_PID_DIR** to the moved script as shown below:

```
# cat >> /etc/default/solr.in.sh <<EOF
SOLR_HOME=/var/lib/escenic
SOLR_LOGS_DIR=/var/log/escenic
SOLR_PID_DIR=/var/run/escenic
EOF
```

   This will ensure that Solr looks for the core in the right place (where you created it in step 1) and writes log files and runtime files to the same locations as corresponding Content Store files.
5. Make sure that all the folders you specified in step 4 actually exist and that the **escenic** user is either the owner or has read/write permissions.
6. Add an **init.d** script for starting Solr automatically on boot:

```
# cp /opt/solr/bin/init.d/solr /etc/init.d/
# sed -i "s#RUNAS=\"solr\"#RUNAS=\"escenic\"#" /etc/init.d/solr
```

Copyright © 2008-2021 Stibo DX A/S

7. Add the **`init.d`** script to the desired run levels. For Debian- and Ubuntu-based systems, you can do this as follows:

   ```
   # update-rc.d solr defaults 35
   ```

   For other systems, please consult the appropriate documentation.

3.1.9.2 Copy Solr Configuration

You now need to copy the following Solr configuration data:

- The main Solr configuration file supplied with Solr itself
- Solr core configuration files supplied with the Content Store

The configuration data should be copied into a folder under **`/etc/escenic`** and then linked to from the **`/var/lib/escenic/solr/`** tree as described below.

For a single host installation:

1. Log in as **`root`** and create a **`solr`** folder under **`/etc/escenic`**, then copy the **`solr.xml`** file supplied with Solr to **`/etc/escenic`**:

   ```
   # mkdir -p /etc/escenic/solr
   # cp /opt/solr-solr-version/server/solr/solr.xml /etc/escenic
   ```

   where *solr-version* is the version number of your Solr installation.

2. Copy the configuration folder supplied with the Content Store:

   ```
   # cp -r engine-installation/solr/conf /etc/escenic/solr/solr-core
   ```

   where *solr-core* is the name of your Solr core (most likely **`editorial`** on a single host installation).

3. Log in as **`escenic`** and create links to the copied file and folder as follows:

   ```
   $ cd /var/lib/escenic/
   $ ln -s /etc/escenic/solr.xml
   $ cd solr/solr-core
   $ ln -s /etc/escenic/solr/solr-core conf
   ```

For a multiple host installation you will need to repeat these steps on each host, and in this case, *solr-core* is likely to vary according to the type of **engine host** – either **`editorial`** or **`presentation`**. Before you start, copy the *engine-installation*`/solr/conf` folder from your **assembly host** to a temporary location (say **`/tmp/conf`**) on each engine host:

```
$ scp -r engine-installation/solr/conf engine-host:/tmp/
```

Then repeat the five steps listed above on each host, replacing step 2 with:

```
# mv /tmp/conf /etc/escenic/solr/solr-core
```

Note that the Solr configuration data supplied with the Content Store is intended for use on an editorial host and is not ideally suited to presentation hosts, so for production installations, it will need to be modified. For more about this, see Customizing the Index Schema..

Copyright © 2008-2021 Stibo DX A/S

3.2 Installation

This section describes the main installation procedure.

A note about version code names

The CUE Content Store and all related applications (plug-ins, the CUE editor and so on) are released on a synchronized schedule where all product versions in a given release are known to work well together. Only these approved version combinations are supported. Each set of compatible product versions is identified by a code name, and during installation you can use this code name instead of the individual product's version number, thereby simplifying the installation process.

In the case of the Content Store and other Linux applications installed using **`apt`**, the code name is actually the name of a repository containing compatible versions of all products. This means that in order to ensure version compatibility, all you need to do is add the name of the required repository to your **`/etc/apt/sources.list.d/escenic.list`** file. Once you have done this you do not need to specify any version numbers when installing individual packages - **`apt`** will just install the latest maintenance release from that repository.

> Note that code names cannot be used in this way on CentOS/Red Hat installations, where the application packages to be installed must still be identified by their version numbers.

The code name for Content Store 7.5 is **`nelson`**.

3.2.1 Install CUE Packages

To install the required CUE packages on your **assembly-host**, log in as **`root`** and perform the following steps:

1. If necessary, add the Escenic repository signing key to your machine's keyring:

   ```
   # curl --silent https://apt.escenic.com/repo.key | apt-key add -
   ```

2. Add the current version repository name to your list of sources.

   ```
   # echo "deb https://user:password@apt.escenic.com nelson 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.

3. Update your package lists and install the Content Store and the CUE scripts:

   ```
   # apt-get update
   # apt-get install escenic-content-engine
   # apt-get install escenic-content-engine-updater
   # apt-get install plug-in-packages
   ```

   where *plug-in-packages* is a list of any CUE plug-in packages that you want to install together with the Content Store.

4. Change the owner of the assembly tool that has been installed (as part of the escenic-content-engine-updater package) to **`escenic`**:

   ```
   # chown escenic /usr/share/escenic/escenic-assemblytool-at-version/
   ```

   where *at-version* is the version number of the installed assembly tool.

Copyright © 2008-2021 Stibo DX A/S

At multi-host installations you need to repeat this installation on all your **engine-hosts**. To do this, log in as **root** on each host and enter the following commands:

```
# curl --silent https://apt.escenic.com/repo.key | apt-key add -
# echo "deb https://user:password@apt.escenic.com nelson main non-free" >> /etc/apt/
sources.list.d/escenic.list
# apt-get update
# apt-get install escenic-content-engine
# apt-get install escenic-content-engine-updater
# apt-get install plug-in-packages
# chown escenic /usr/share/escenic/escenic-assemblytool-at-version/
```

The commands you need to use to install these packages on RedHat / CentOS systems are:

```
# rpm -Uvh https://user:password:yum.escenic.com/rpm/escenic-content-
engine-7.5-7.5.12-2.x86_64.rpm
# rpm -Uvh https://user:password:yum.escenic.com/rpm/escenic-content-engine-
updater-7.5-7.5.12-2.x86_64.rpm
```

where *version* is the correct version number of the package.

3.2.2 Initialize the Assembly Tool

On your **assembly-host**, while logged in as **escenic**:

1. **cd** to the *assemblytool_installation* folder:

   ```
   $ cd assemblytool_installation
   ```

2. Enter the following command:

   ```
   $ ant initialize
   ```

   This command creates the **.properties** files the assembly tool needs.

3. Open **assemble.properties** for editing.
4. Uncomment the **engine.root** setting near the top of the file, and set it as follows:

   ```
   engine.root = engine-installation
   ```

5. If you want the **escenic-admin** web application to be password protected, uncomment the **escenic-admin-authentication** property, and set it to **true**:

   ```
   escenic-admin-authentication = true
   ```

   You are strongly recommended to set this property when installing a production system. If you do so, then you must also define the required administrator username and password in Tomcat. For details of how to do this, see section 5.2.

3.2.3 Initialize the Bootstrap Layer

On your **assembly-host**, while logged in as **escenic**:

1. **cd** to the *assemblytool_installation* folder:

   ```
   $ cd assemblytool_installation
   ```

Copyright © 2008-2021 Stibo DX A/S

2. Enter the following command:

```
$ ant -q ear
```

This command (which takes a few minutes to complete) creates the **bootstrap layer**. The bootstrap layer is a set of configuration files that "configure the configuration process". Once you have run the above command, you will find the bootstrap layer located in *assemblytool_installation*`/conf`. This bootstrap layer is set up to look for three configuration layers. The first two layers (`default` and `addon`) are part of the delivered system, and do not concern you. The third layer is called `common`, and the bootstrap layer is set up to look for it in `/etc/escenic/engine/common`. You will therefore need to create a common configuration layer there.

3.2.4 Edit the Common Configuration Layer

The Content Store package installs a **skeleton configuration layer** in `/etc/escenic/engine/common`. This folder tree contains configuration files that you can use as a basis for your host configurations. All the property settings in these files are commented out, and you need to uncomment and set values in some of the files.

On your **assembly-host**, while logged in as `root`:

1. Open `/etc/escenic/engine/common/ServerConfig.properties` for editing, and make sure the following properties are present, uncommented and set correctly for your site:

   `databaseProductName`
   The name of the database you are using. Currently allowed values are:

   - `MySQL`

   `filePublicationRoot`
   The path of the folder in which all binary files served by the Content Store are stored (multimedia files - video, audio, images, Word documents, PDF files and so on). You should set this to:

   ```
   filePublicationRoot=/var/lib/escenic/engine/
   ```

   The path you specify **must** include a trailing slash.

   `webPublicationRoot`
   The URL root you want to be used for all your publications. For example `http://`*my-company*`.com/`. The URLs of your publications are formed by appending the publication name to this string. This property is in general used only for development purposes. In production, it is often the case that each publication has its own root URL.

   `customerId`
   The name of your organization, or some other name that will clearly identify your installation to Stibo DX support staff, plus information about the type of this Content Store

Copyright © 2008-2021 Stibo DX A/S

instance. This identifier is used for error and usage reporting. For production installations you **must** use the following format:

```
customerId=prod@customer-id
```

where *customer-id* is an identifier for your organization - your domain name for example:

```
customerId=prod@mycompany.com
```

You can optionally prefix the customer ID with a qualifier of some kind if you have multiple installations and separate reporting is considered useful. You might, for example, replace **`prod@mycompany.com`** with **`prod@dailynews.mycompany.com`** on some hosts and **`prod@weeklymag.mycompany.com`** on others.

For non-production instances, you can replace the **`prod`** instance type identifier with another identifier of your choice, such as **`test`** or **`staging`**.

Reporter components that are installed with the Content Store automatically send reports about the Stibo DX software running at your site. The reported information includes:

- Which versions of the Content Store and its plug-ins are running at your site.
- The number of active CUE users (reported once an hour)

This information is very valuable to Stibo DX support engineers and helps with the diagnosis of reported errors and performance problems. It is also used in the calculation of support and maintenance charges. For this reason your are **required** to set this property on all Content Store installations and to configure your firewall so that reports can be delivered (see below).

> The reporter components sends their reports to Google Analytics. In order for this to work, your firewalls must allow outgoing traffic on port 80 to **`http://www.google-analytics.com`**.

2. Open **`/etc/escenic/engine/common/neo/io/managers/ContentManager.properties`** for editing, and make sure the following properties are uncommented and set correctly:

   `dataConnector`
   Must be set as follows:

   ```
   dataConnector=/connector/DataConnector
   ```

3.2.5 Create Host Configuration Layers

> If you are installing everything on one host, then skip this section.

In a multi-host installation, a small number of configuration properties may need to be set differently on different hosts. This can be achieved by creating individual configuration layers for each host. The Content Store instance on each host reads its host configuration layer **after** the common configuration layer, and any settings it finds there override both system defaults and any values set in the common configuration layer.

On your **assembly-host**, while logged in as **`root`**, create a folder called **`/etc/escenic/engine/host`**. Under this folder, create a folder for each machine that is to host a Content Store instance. If, for example, your installation has one **editorial host** (called **`editorial1`**) and one **presentation host** (called **`presentation1`**), then you should enter:

```
# mkdir -p /etc/escenic/engine/host/{editorial1,presentation1}
```

Copyright © 2008-2021 Stibo DX A/S

If there are any configuration settings that you know need to be set differently on a specific host, you can set them as follows:

1. Copy the appropriate **.properties** file from the skeleton configuration layer to the same relative location in the appropriate host configuration layer.
2. Open the copied file for editing.
3. Uncomment the required property and set it to the required value.

For further information about the configuration layers, see Configuring The Content Engine.

3.2.6 Configure the ece Script

To configure the **ece** shell script:

On your **assembly-host**, while logged in as **root**:

1. Create an empty file in the **/etc/escenic/** folder **called ece-engine1.conf**:

   ```
   # touch /etc/escenic/ece-engine1.conf
   ```

2. Open **/etc/escenic/ece-engine1.conf**, and copy into it the parameter settings listed below from **/etc/escenic/ece.conf**. Edit the copied parameter settings. If you have followed all the instructions in this guide exactly, then the following settings should work:

   java_home
   Make sure that this is set to the path of the Java virtual machine included in the JDK you installed (see section 3.1.1).

   ece_home
   Set this to *engine-installation*.

   escenic.server
   Set this to the host name or IP address.

   ece_security_configuration_dir
   Set this to **/etc/escenic/engine/common/security**.

   appserver
   Currently, the only valid setting for this is **tomcat**.

   tomcat_home
   Set this to **/opt/tomcat-engine1**.

   assemblytool_home
   Set this to *assemblytool_installation*.

3. Create the standard installation directory structure. Enter the following commands:

   ```
   # mkdir -p /var/{crash,lib,log,run,cache,spool}/escenic
   # chown escenic:escenic /var/{crash,lib,log,run,cache,spool}/escenic -R
   ```

   These commands will create the following folders and assign them to the **escenic** user:

   /var/crash/escenic
   /var/lib/escenic
   /var/log/escenic
   /var/run/escenic
   /var/cache/escenic

Copyright © 2008-2021 Stibo DX A/S

/var/spool/escenic

> If, for any reason, you do not want to install Content Store files in standard locations and you have modified any of the following settings in the **ece.conf** file:
>
> **cache_dir**
> **log_dir**
> **pid_dir**
> **heap_dump_dir**
>
> then you must make sure that all the referenced folders exist and that the **escenic** user has write permission to them. You are strongly advised, however, not to do so. All instructions in this manual and the Server Administration Guide assume files are installed in standard locations.

3.3 Finishing Up

You may now need to assemble and deploy an EAR file containing the supplied CUE web applications, check that everything is installed correctly and so on. These tasks are described in the following sections.

3.3.1 Assemble and Deploy

You **may** now need to assemble and deploy your web applications. You only need to do this if your installation includes one or more old-style JSP-based Escenic publications. If all your publications are rendered via CUE Front, then you do not need to assemble and deploy.

Assembly and deployment consists of the following steps:

1. On your **assembly-host**, while logged in as **escenic**, run the **ece** script to assemble a set of Content Store applications:

   ```
   $ ece assemble
   ```

   This generates an **enterprise archive** (EAR file) which you can deploy on all your **engine-host**s. The EAR file is called **engine.ear**, and is created in the **/var/cache/escenic/** folder.

2. > If you are installing everything on one host, then skip this step.

   On each **engine-host**, copy **/var/cache/escenic/engine.ear** from the **assembly-host**. If you have installed an SSH server on the **assembly-host** and SSH clients on your **engine-host**s, then you can do this as follows:

   ```
   $ scp -r escenic@assembly-host-ip-address:/var/cache/escenic/engine.ear /var/
   cache/escenic/
   ```

   where *assembly-host-ip-address* is the host name or IP address of your **assembly-host**.

3. On each **engine-host**, deploy the EAR file by entering:

   ```
   $ ece deploy
   ```

Copyright © 2008-2021 Stibo DX A/S

3.3.2 Verify the Installation

It should now be possible to start the Content Store and verify that the major components are working. To do so, carry out the following procedure on each **engine-host**, while logged in as **escenic**:

1. Start the Content Store by entering:

   ```
   $ ece start
   ```

2. Verify that the Content Store is running by entering:

   ```
   $ ece status
   ```

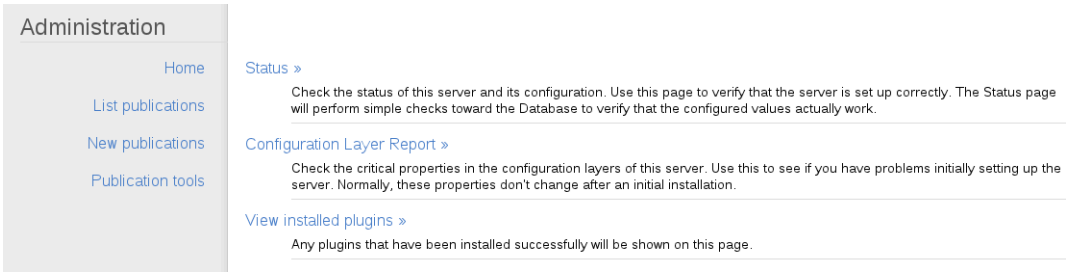
   This should produce something like the following output:

   ```
   [ece] Escenic Content Engine IS running (PID 1936) up 0d 1h 33m 51s
   ```

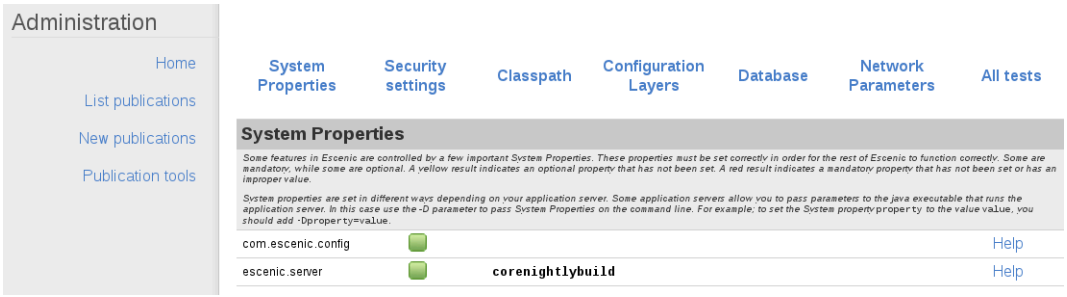
3. Open a browser, and try accessing the **escenic-admin** web application on each of the **engine-host**s by entering:

   ```
   http://engine-host-ip-address:8080/escenic-admin/
   ```

   in the browser address field. There may be a delay while the Content Store initializes, but eventually, you should see the following page:



4. Click on the **Status** link to display a set of status pages containing Content Store test results.



5. Across the top of these pages is a menu containing links to each of the status pages. Click on each of these links in turn and verify that all the tests pass (indicated by a green icon).

6. The red icon indicates that something is not correctly configured. If you see one of these icons, then you need to check over all relevant settings. In most cases there is a help link next to the test result indicating the most likely causes of failure. Make any necessary corrections, then restart the Content Store by entering:

   ```
   $ ece restart
   ```

7. Wait a minute or so, then refresh the browser window to re-execute the tests that failed.

Verify that the Content Store is correctly configured on **all** your **engine-host**s before proceeding any further.

Copyright © 2008-2021 Stibo DX A/S

On some of the test pages you will see ! warning icons. In most cases these are simply an indication that no publications have yet been created, and they can be ignored.

3.3.3 Install a Daemon Script

At this point you have successfully installed the Content Store and managed its instances using the **ece** script. Now you may want to install a daemon script that will automatically run the Content Store on system start-up and shut it down on system shutdown. If you have the database server on the same machine then you have to make sure that it also starts automatically on system start-up.

The following instructions tell you how to install the daemon script.

Log in as **root** on your **assembly-host** and on each of your **engine-host**s and do the following:

1. Copy the daemon script **ece** to **/etc/init.d/**, and the script's default settings (**ece-scripts/etc/default/ece**) to a file called **ece** in the **/etc/default** folder. On the **assembly-host** you can do this as follows:

   ```
   # cp engine-installation/ece-scripts/etc/init.d/ece /etc/init.d/
   # cp engine-installation/ece-scripts/etc/default/ece /etc/default/
   ```

   On the engine-hosts, however, you will have to copy the files from the **assembly-host**. If you have installed an SSH server on the **assembly-host** and SSH clients on your **engine-host**s, then you can do it as follows:

   ```
   # scp escenic@assembly-host-ip-address:engine-installation/ece-scripts/etc/init.d/
   ece /etc/init.d/
   # scp escenic@assembly-host-ip-address:engine-installation/ece-scripts/etc/
   default/ece /etc/default/
   ```

2. Make the daemon script executable:

   ```
   # chmod +x /etc/init.d/ece
   ```

3. The next step is to run the daemon script, **ece**. But before that you may need to modify the settings for the daemon script. Open **/etc/default/ece** in an editor and modify the settings to meet your requirements.

   If you have not installed the Content Store files in standard locations, then you must modify the **dir_list** setting according to the **/etc/escenic/ece.conf**.

   Now, if you have started the Content Store, stop it.

   ```
   # ece stop
   ```

   Then enter the following commands:

   ```
   # sudo /etc/init.d/ece start
   ```

   If everything is ok at this point, then your are ready to add this script as a daemon. Other wise you need to fix the problem and then try again.

4. On your **engine-host**s only, enter the following commands:

   ```
   # update-rc.d ece defaults
   ```

   This command actually installs the script as a daemon, ensuring that it will be run on system startup. Note that this command is specific to Debian-based Linux distributions: the procedure for installing daemon scripts is different on other distributions.

Copyright © 2008-2021 Stibo DX A/S

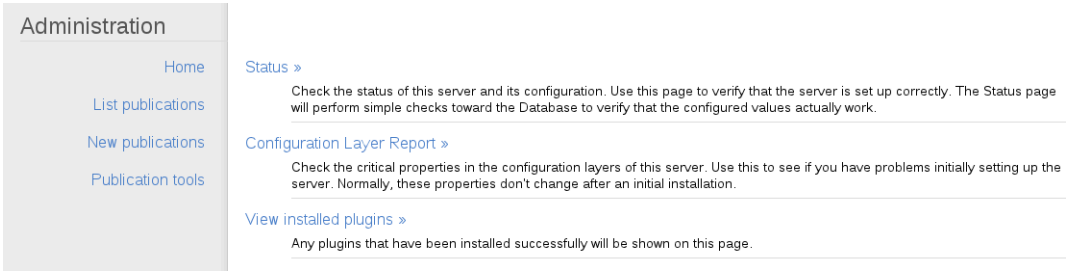
3.3.4 Create a Publication

To really verify that everything is working correctly, you need a publication in the database. The quickest way to achieve this is to add one of the demo publications included with the installation. To add the **`demo-temp-dev`** publication (which is a very simple, stripped down publication with almost no content):

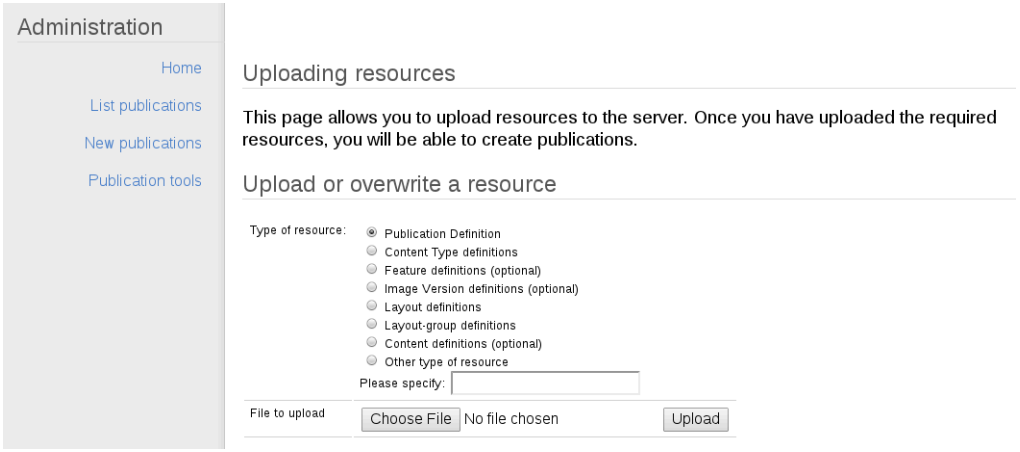
1. Download the publication WAR file from one of your **engine-host**s to the machine on which you are working. You will find **`demo-temp-dev.war`** in the *engine-installation*`/contrib/wars` folder.
2. Open a browser, and access the **`escenic-admin`** web application on one of your **editorial-host**s by entering:

   ```
   http://editorial-host-ip-address:8080/escenic-admin/
   ```

   in the browser address field. The following page is displayed:



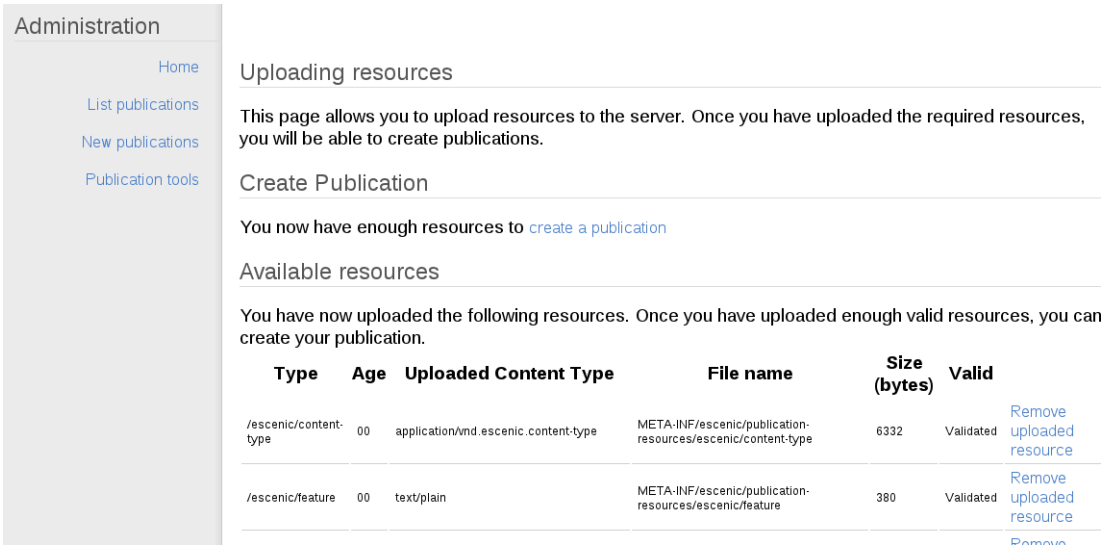
3. Click on **New publications**. The following page is displayed:



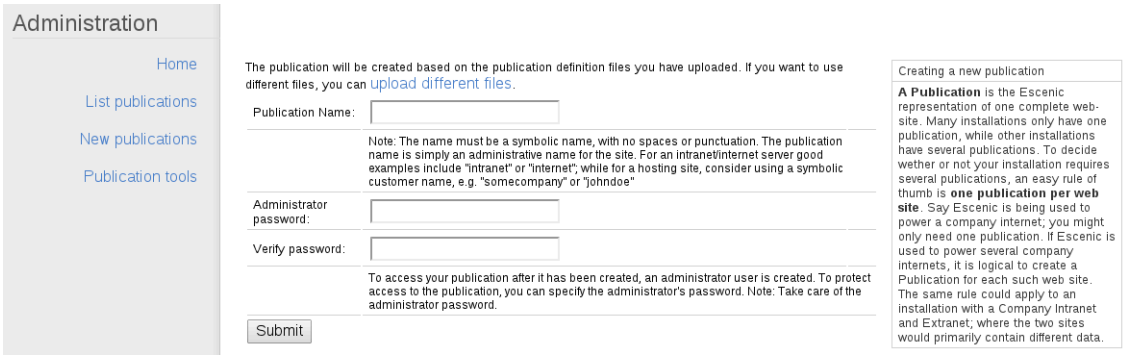
4. Click on the **Choose File** button and locate the **`demo-temp-dev.war`** file you have copied to your local machine.

Copyright © 2008-2021 Stibo DX A/S

5. Click on **Upload**. The following page is displayed:

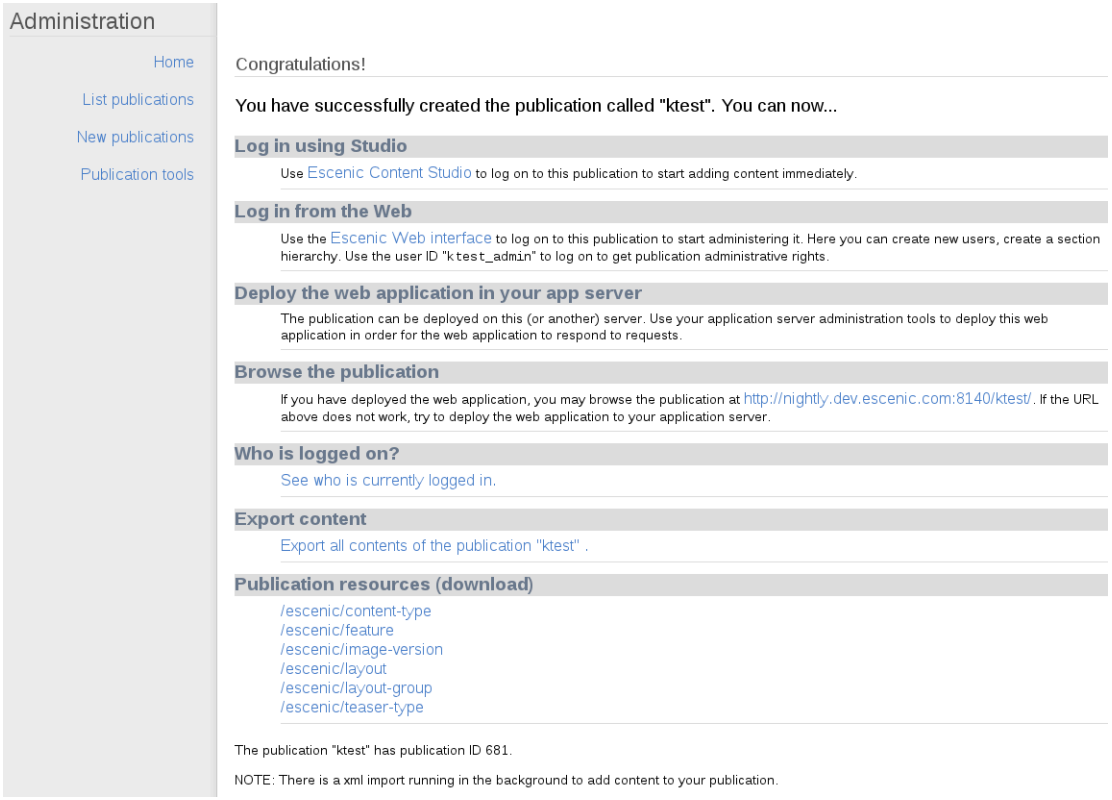


6. Click on the **create a publication** link. The following page is displayed:



Copyright © 2008-2021 Stibo DX A/S

7. Enter a name for the publication and an administrator password in the displayed form, then click on **Submit**. The following page is displayed:



You have now created a publication. If you click on **Home** and then **Status** to redisplay the status pages, you should see that all the ! warnings that were displayed earlier have now been replaced by icons. This should be the case not only on the **editorial-host** where you are currently working, but on all your **engine-host**s.

3.3.5 Test Web Studio

Now you have created a publication, you can use it to check that the Web Studio web application is correctly installed. To do this:

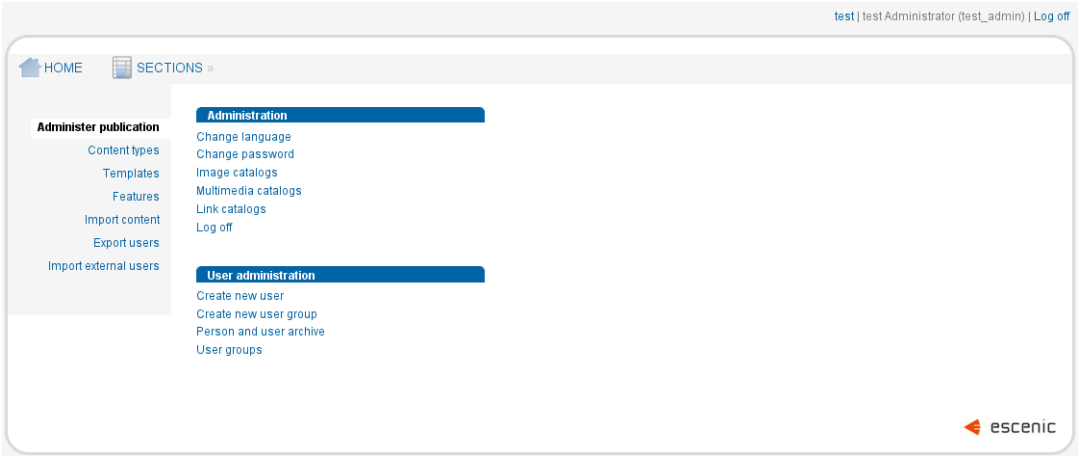
1. Open a browser, and access the Web Studio web application (called **`escenic`**) on one of your **editorial-host**s by entering:

   ```
   http://editorial-host-ip-address:8080/escenic/
   ```

   in the browser address field. A login page should be displayed.
2. Enter the **`demo-temp-dev`** publication's administrator user name (*publication-name*`_admin`) and the administrator password you specified when creating the publication.

Copyright © 2008-2021 Stibo DX A/S

3. The following page should then be displayed:



Copyright © 2008-2021 Stibo DX A/S

4 Recommended Configurations

This chapter contains detailed descriptions of recommended system configurations for the Content Store, CUE and an SSE Proxy, which is required to forward Server-Sent Events from the Content Store to CUE clients. These components can be configured to work together in a variety of ways to meet different requirements, and typical system configurations will often also involve the configuration and use of other components such as web servers, load balancers and so on. This chapter only describes two possible configurations:

- A minimal HTTP-based development/test configuration, where all components are installed on the same host
- A more robust HTTPS-based production configuration where the CUE editor is installed on a different host from the Content Store and the SSE Proxy.

These descriptions should provide a basis for understanding how the components communicate, so that you can make any adjustments needed to meet your specific requirements.

All configurations involve at least the following tasks:

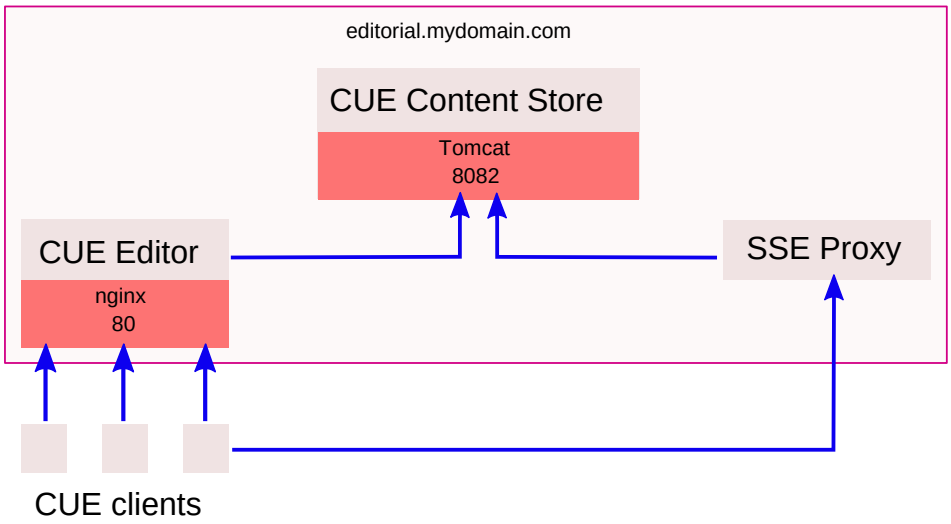
- Installing an SSE Proxy (see the SSE Proxy documentation for details).
- Configuring your Content Store installation's Tomcat server to direct SSE messages to the SSE Proxy.
- Configuring the SSE Proxy to accept messages from the Content Store, and forward them to CUE.
- Configuring CUE to connect to the Content Store.

If your installation includes more than one Content Engine web service host, with a load balancer distributing requests between them, then you must ensure that both CUE and its SSE proxy use sticky sessions and connect to the same back end.

Copyright © 2008-2021 Stibo DX A/S

4.1 A Development/Test Configuration (HTTP)

The three main components in this configuration all belong to the host **`editorial.mydomain.com`**:



All network connections are made using HTTP. The configuration details for the Content Store, CUE and the SSE Proxy are as described in the following sections.

4.1.1 Content Engine / Tomcat Configuration (HTTP)

You need to create a connector in Tomcat for the SSE Proxy It must use a different port from the default connector used by CUE (port 8082 instead of 8080, for example). In this case CUE will connect to the Content Store on **`http://editorial.mydomain.com`** so you can set up the new connector as follows:

```
<Connector port="8082" protocol="HTTP/1.1"
  maxThreads="50" URIEncoding="UTF-8"
  SSLEnabled="false"
  proxyName="editorial.mydomain.com" proxyPort="80"
 scheme="org.apache.coyote.http11.Http11NioProtocol"/>
```

Restart Tomcat for the change to take effect.

> If you want to use Varnish with your installation, then you should set **`proxyPort`** to **`81`** and also use port **`81`** in your CUE configuration (see section 4.1.2).

4.1.2 CUE Configuration (HTTP)

CUE Configuration involves configuring both CUE itself and the **`nginx`** web server used to host it (for more about this, see the CUE documentation). The CUE configuration file **`/etc/escenic/cue-web-`***version*`/config.yml`** must contain the following setting identifying the Content Store web service to connect to:

```
endpoints:
  escenic: "http://editorial.mydomain.com:80/webservice/index.xml"
```

> If you want to use Varnish with your installation, then you should use port **`81`** rather than port **`80`** and also set **`proxyPort`** to **`81`** in your Tomcat configuration (see section 4.1.1).

Copyright © 2008-2021 Stibo DX A/S

The required **nginx** configuration is as follows:

- In **/etc/nginx/sites-available/default**:

```
server {
  listen 80 default;
  include /etc/nginx/default-site/*.conf;
}
```

- In **/etc/nginx/default-site/cue-web.conf**:

```
location /cue-web/ {
        alias /var/www/html/cue-web/;
        expires modified +310s;
}
```

- In **/etc/nginx/default-site/webservice.conf**:

```
location ~ "/(escenic|studio|webservice|webservice-extensions)/(.*)" {
    proxy_set_header Host $http_host;
    proxy_pass http://127.0.0.1:8082;
}
```

- In **/etc/nginx/conf.d/request-entity-size-limit.conf**:

```
client_max_body_size 0;
```

4.1.3 SSE Proxy Configuration (HTTP)

To configure an SSE Proxy, you edit its **sse-proxy.yaml** configuration file. All you need to do is add the URL of the Content Store's change log SSE endpoint to it's list of backends:

```
backends:
  - uri: http://editorial.mydomain.com:8082/webservice/escenic/changelog/sse
```

Even though your SSE Proxy is in this case installed on the same host as the Content Store, you must still use the host name specified in the Tomcat connector definition: do not replace it with **localhost**.

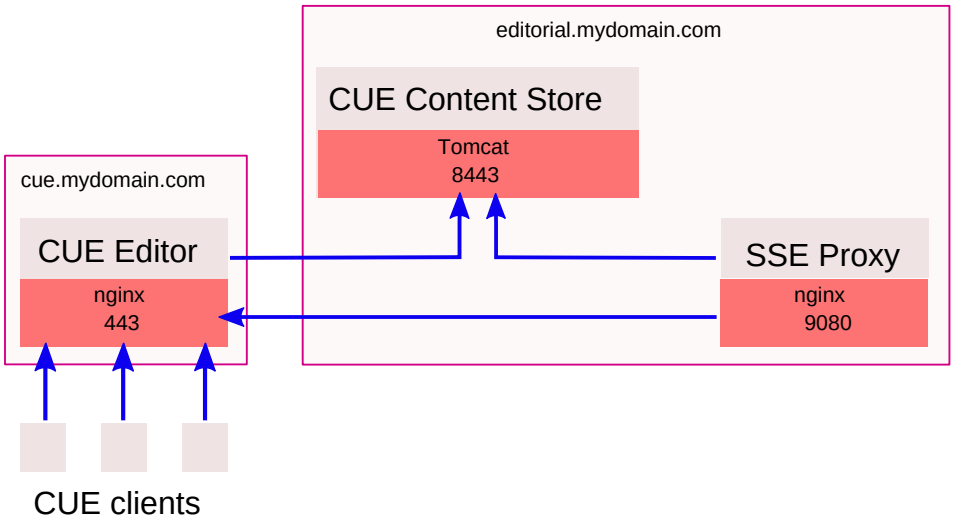
For full information about installing and configuring an SSE Proxy, see the SSE Proxy documentation.

> Once your SSE Proxy is configured, you will also need to reconfigure the change log event stream URI that the Content Store offers to clients. By default the Content Store sends clients the URI of its own SSE endpoint, but now you want clients to get their change log events from the SSE Proxy instead, so you need to set this configuration to:
>
> ```
> sseEndpoint=http://editorial.mydomain.com:9080/
> ```
>
> For detailed instructions on how to do this, see Changing the Content Engine Event Stream URI.

Copyright © 2008-2021 Stibo DX A/S

4.2 A Production Configuration (HTTPS)

In this production configuration, the CUE application is installed on a different host from the Content Store and the SSE Proxy. All network communication is SSL-protected. Since the SSE Proxy does not have built-in SSL support, it is placed behind an **nginx** web server to provide that support.



In order to set up an HTTPS configuration, you need to install SSL certificates, which must be obtained from a certificate authority such as Let's Encrypt, Thawte or Verisign. There is a list of certificate suppliers here. Follow the supplier's instructions for installing the certificates they provide.

4.2.1 Content Engine / Tomcat Configuration (HTTPS)

In this case the connector you set up for the SSE Proxy must be configured to support SSL connections as follows:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
  maxThreads="50" URIEncoding="UTF-8"
  SSLEnabled="false" secure="true"
  proxyName="cue.mydomain.com" proxyPort="443" scheme="https" />
```

Restart Tomcat for the change to take effect.

4.2.2 CUE Configuration (HTTPS)

CUE Configuration involves configuring both CUE itself and the **nginx** web server used to host it (for more about this, see the CUE documentation). The CUE configuration file **/etc/escenic/cue-web-*version*/config.yml** must contain the following setting identifying the Content Store web service to connect to:

```
endpoints:
  escenic: "https://cue.mydomain.com/webservice/index.xml"
```

The required **nginx** configuration is as follows:

- In **/etc/nginx/sites-available/default**:

```
server {
  include /etc/nginx/certs/*.conf;
```

Copyright © 2008-2021 Stibo DX A/S

```
    include /etc/nginx/default-site/*.conf;
}
```

- In **/etc/nginx/default-site/cue-web.conf**:

```
location /cue-web/ {
        alias /var/www/html/cue-web/;
        expires modified +310s;
}
```

- In **/etc/nginx/default-site/webservice.conf**:

```
location ~ "/(escenic|studio|webservice|webservice-extensions)/(.*)" {
    proxy_set_header Host $http_host;
    proxy_pass http://editorial.mydomain.com:8443/;
}
```

- In **/etc/nginx/conf.d/request-entity-size-limit.conf**:

```
client_max_body_size 0;
```

- In **/etc/nginx/certs/cue.mydomain.com.conf**:

```
listen 443 ssl;
server_name cue.mydomain.com
ssl_prefer_server_ciphers On;
ssl_certificate /etc/nginx/certs/cue.mydomain.com.pem;
ssl_certificate_key /etc/nginx/certs/cue.mydomain.com.key;
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
ssl_ciphers ECDH+AESGCM:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH+AES128:DH+AES:RSA
+AESGCM:RSA+AES:!aNULL:!MD5:!DSS;
```

This configuration is based on the assumption that you have obtained and installed a valid SSL certificate. For information on how to do this, please see the documentation of your chosen certificate supplier.

4.2.3 SSE Proxy Configuration (HTTPS)

Add the URL of the Content Store's change log SSE endpoint to the SSE Proxy's list of backends and set the Host header as follows:

```
backends:
  - uri: http://editorial.mydomain.com:8443/webservice/escenic/changelog/sse
      httpHeaders:
        Host: cue.mydomain.com
```

For full information about installing and configuring an SSE Proxy, see the SSE Proxy documentation.

The web server in front of the SSE Proxy must be able to handle an SSE stream without constantly disconnecting clients: It must be configured to:

- Allow long-running connection
- Support chunked encoding
- Not buffer or cache the traffic

The following example shows a suitable **nginx** configuration (**/etc/nginx/default-site/sse-proxy.conf**) for a virtual host fronting an SSE Proxy installed on the same host as the Content Store:

```
location /sse {
```

Copyright © 2008-2021 Stibo DX A/S

```
    proxy_read_timeout     300;
    proxy_connect_timeout  300;

    proxy_pass http://editorial.mydomain.com:9080;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header Connection '';
    proxy_http_version 1.1;
    proxy_buffering off;
    proxy_cache off;
    chunked_transfer_encoding off;
  }
```

Once your SSE Proxy is configured, you will also need to reconfigure the change log event stream URI that the Content Store offers to clients. By default the Content Store sends clients the URI of its own SSE endpoint, but now you want clients to get their change log events from the SSE Proxy instead, so you need to set this configuration to:

```
sseEndpoint=https://cue.mydomain.com/sse
```

For detailed instructions on how to do this, see Changing the Content Engine Event Stream URI.

4.3 CORS Configuration

CORS (Cross-Origin Resource Sharing) is required in situations where the CUE editor is located in a different domain from other resources to which the CUE editor client will require access. Neither of the recommended configurations described in this chapter require use of CORS. Other possible configurations, however, may require some components to be configured to support CORS.

If, for example, the production configuration described in section 4.2 is modified by placing an **`nginx`** proxy in front of the Content Store web services, then a **`/etc/nginx/default-site/cors.conf`** file would need to be included in the proxy server's configuration, with the following content:

```
location ~ "/(escenic|studio|webservice|webservice-extensions)/(.*)" {
    if ($http_origin ~* (https?://[^/]*.mydomain.com(:[0-9]+)?)$) {
        set $cors "true";
    }
    if ($request_method = 'OPTIONS') {
        set $cors "${cors}options";
    }
    if ($request_method = 'GET') {
        set $cors "${cors}get";
    }
    if ($request_method = 'HEAD') {
        set $cors "${cors}get";
    }
    if ($request_method = 'POST') {
        set $cors "${cors}post";
    }
    if ($request_method = 'PUT') {
        set $cors "${cors}post";
    }
    if ($request_method = 'DELETE') {
        set $cors "${cors}post";
    }
    if ($cors = "trueget") {
```

Copyright © 2008-2021 Stibo DX A/S

```
        add_header "Access-Control-Allow-Origin" "$http_origin" always;
        add_header "Access-Control-Allow-Credentials" "true" always;
        add_header "Access-Control-Expose-Headers" "Link,X-ECE-Active-
Connections,Location,ETag,Allow" always;
    }
    if ($cors = "truepost") {
        add_header "Access-Control-Allow-Origin" "$http_origin" always;
        add_header "Access-Control-Allow-Credentials" "true" always;
        add_header "Access-Control-Expose-Headers" "Link,X-ECE-Active-
Connections,Location,ETag" always;
    }
    if ($cors = "trueoptions") {
        add_header 'Access-Control-Allow-Origin' "$http_origin";
        add_header 'Access-Control-Allow-Credentials' 'true';
        add_header 'Access-Control-Max-Age' 1728000;
        add_header 'Access-Control-Allow-Methods' 'GET, POST, HEAD, OPTIONS, PUT,
 DELETE';
        add_header 'Access-Control-Allow-Headers' 'Authorization,Content-
Type,Accept,Origin,User-Agent,DNT,Cache-Control,X-Mx-ReqToken,Keep-Alive,X-Requested-
With,If-Modified-Since,If-Match,If-None-Match,X-Escenic-Locks,X-Escenic-media-
filename,X-Escenic-home-section-uri';
        add_header 'Content-Length' 0;
        add_header 'Content-Type' 'text/plain charset=UTF-8';
        return 204;
    }
}
```

Copyright © 2008-2021 Stibo DX A/S

5 Recommended Modifications

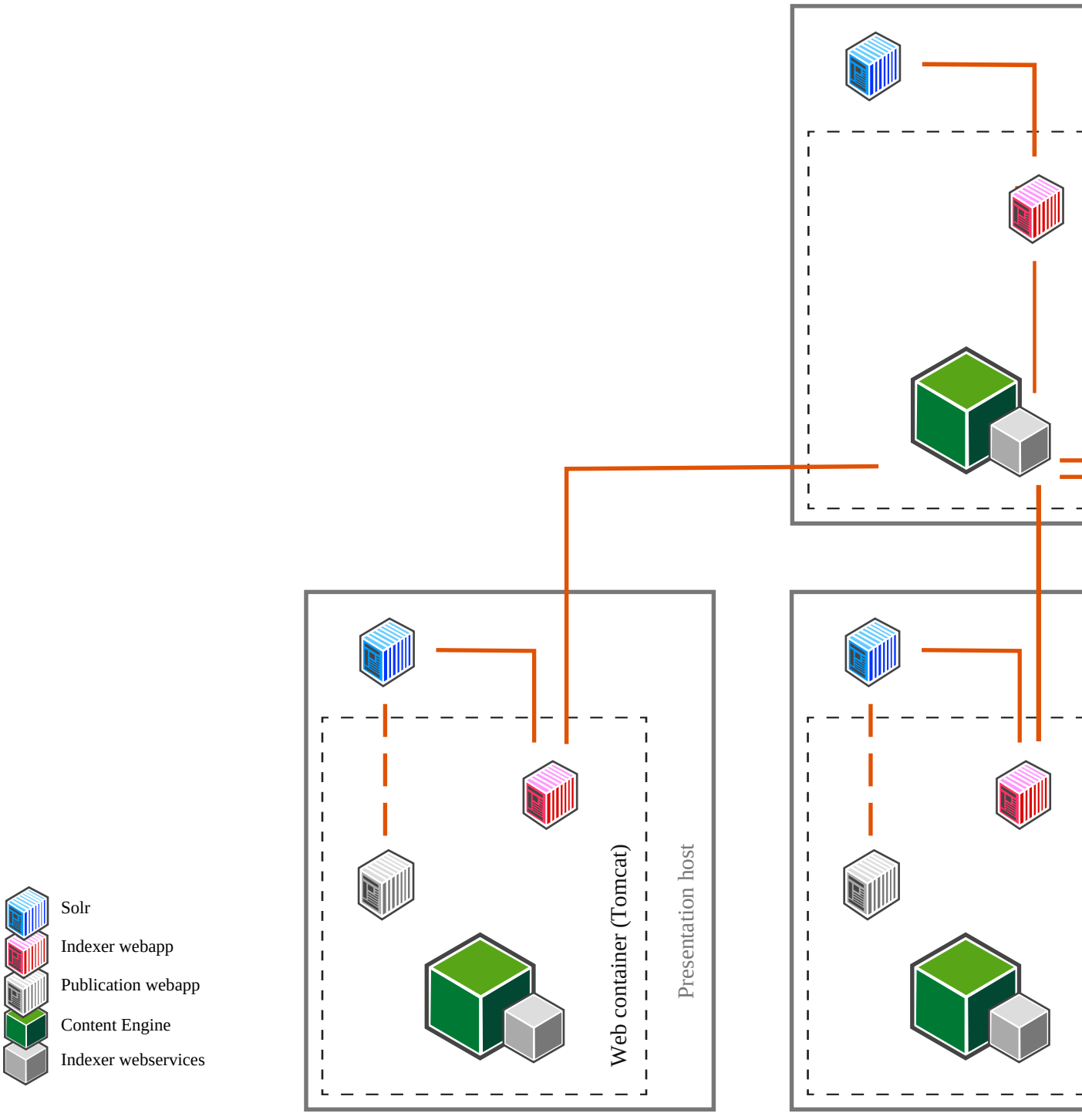
This section contains instructions for various additions and modifications you can make to the basic installation described in chapter 3. These changes are recommended for production installations.

5.1 Search Engine Deployment and Configuration

The Content Store's search functionality is provided by the Java search engine Apache Solr, which runs as a web application. The standard Content Store installation includes a **`solr`** web application and an associated **`indexer`** web application that indexes the content of all CUE publications. These two applications are deployed along with the Content Store by the **`ece`** script's **`deploy`** action.

Copyright © 2008-2021 Stibo DX A/S

The result of following the basic installation procedure described in chapter 3, therefore, is that a **`solr`** instance and **`indexer`** web application is deployed on every **engine host** in your installation, all with identical configurations.



This set up will work, but it is relatively inefficient and is unlikely to work well in a production environment. There are two main reasons for this:

Copyright © 2008-2021 Stibo DX A/S

Solr memory usage

solr and the **indexer** can at times consume large amounts of memory and trigger large garbage collection operations in the JVM, which has severe effects on Content Store performance. They should therefore not be run in the same JVM as the Content Store on production systems. **solr** already runs in its own webapp container (and therefore in a different JVM), but the indexer is deployed to the same Tomcat instance as the Content Store. The simplest way to achieve this separation on a single-host installation is to move the **indexer** webapp to a separate Tomcat instance. For more about this, see Isolating The Search Engine.

Solr stemming

In the default **solr** configuration, English stemming is enabled by default. This means that searching non-English content might give unexpected results.

If your content is in a language other than English, you should either disable stemming or modify the configuration to suit your language.

To disable stemming, remove the following line from **schema.xml**:

```
<filter class="solr.SnowballPorterFilterFactory" protected="protwords.txt"
 language="English"/>
```

Disabling stemming will improve **solr**'s performance.

For information about how to configure stemming for other languages, see the Solr documentation on http://lucene.apache.org/solr/.

Solr optimization issues

The default **solr** configuration is optimized for editorial purposes: it indexes all the fields needed to support the search functionality provided by CUE, resulting in very large indexes. This is acceptable in the editorial context, since the number of concurrent CUE users, even in a very large organisation, is not likely to be very large. The **presentation host**s in a large CUE installation, however, can be required to serve many thousands of concurrent users, and the default **solr** configuration may perform poorly in this context.

The default configuration, therefore, is fine for the **editorial host**s in a production system, but for the **presentation host**s you are recommended create a custom indexer configuration that only indexes the fields actually needed to support the kinds of search required in your publications.

To do this, open **/var/lib/escenic/***solr-core***/schema.xml** for editing on each of your **presentation host**s, and modify the index schema to meet your requirements. Editing this file is outside the scope of this manual. In order to tune the search engine you need to take account of the both the contents of your publications, your users' needs with regards to search and the limitations imposed by your particular hardware configuration. For further information and advice on tuning, see the Solr documentation on http://lucene.apache.org/solr/.

There are many more changes you can make to your search engine set-up in order to optimize it for your particular needs. For a discussion of the general principles involved, see Search Engine Configuration and Management.

5.2 Password Protect escenic-admin

escenic-admin is a web application for administration of the Content Store. In a default installation it is installed without password protection. **escenic-admin** is, however, a powerful tool and

Copyright © 2008-2021 Stibo DX A/S

should definitely not be openly accessible in a production environment. You are therefore strongly recommended to limit access to a small group of competent users.

To password protect **escenic-admin**:

1. Set **escenic-admin-authentication = true** in the **assemblytool** properties file before running the **assemblytool**.
2. Define one or more user log-ins for **escenic-admin** in Tomcat's **tomcat-users.xml** file.

To define an **escenic-admin** user in Tomcat, you need to add an entry like this to **tomcat-users.xml**:

```
<user username="admin-user" password="admin-password" roles="ECEAdmin"/>
```

where:

- *admin-user* is the user name you want to use for logging in to **escenic-admin**
- *admin-password* is the password you want to use for logging in to **escenic-admin**

> It is obviously **not** a good idea to enter the password in this file as plain text. For a description of how to encrypt the password, see http://www.jdev.it/encrypting-passwords-in-tomcat/.

Copyright © 2008-2021 Stibo DX A/S

6 Upgrading

All you need to do to upgrade your Content Store (including any plug-ins that you have installed from **`apt`** / **`rpm`** packages) is:

1. Read the release notes for your planned upgrades (both Content Store and plug-ins). Make a note of any special tasks that need to be carried out in connection with the upgrades. Some changes, for example, require you to run a database upgrade script after the upgrade.
2. Log in as **`root`** on your **assembly-host** and on each of your **engine-hosts**, and enter the following commands:

   ```
   # apt-get update
   # apt-get upgrade
   ```

3. Carry out any required upgrade tasks. For general information about how to carry out a database upgrade should that be necessary, see: section 6.1.

6.1 Database Upgrades

If the upgrade tasks for the upgrade you are carrying out include upgrading the database, then a script for performing the upgrade will be included in the distribution. All you need to do is log in as the **`escenic`** user on your **database-host** and run the appropriate script for your particular database. The scripts are located in the *engine-installation*`/contrib/database/sql/` folder. Here you will find a subfolder for each supported database - **`MySQL`** and so on (**`MySQL`** = MariaDB). In each subfolder is an **`upgrade`** folder and a script for upgrading to the current version (plus scripts for upgrading to a number of previous versions. To upgrade a MariaDB database, for example, you would need to run:

```
engine-installation/contrib/database/sql/MySQL/upgrade/to-version
```

where *version* is the current Content Store version number.

Copyright © 2008-2021 Stibo DX A/S