Table of Contents

NOTE: Before we start. We use address dspacecris.university.edu as an example for this guide. It MUST be changed with FQDN address of your server.

0) Install Debian 11 (minimal version) and prerequisite tools

We will need two DNS records for backends and frontend. În this guide we will use the same server for both.

And check if the sudo is working in your install. If not, use root for the action below:

```
apt install sudo -y
apt install net-tools -y
apt install git wget curl -y
```

1) Create user for dspace

useradd -m -s /bin/bash/dspace

passwd dspace

usermod -a -G sudo dspace

Edit sudoers file:

In /etc/sudoers find row below:

```
root    ALL=(ALL:ALL) ALL
```

And add after the row with root user:

```
dspace ALL=(ALL:ALL) ALL
```

Next steps will work with user dspace

2) Update your System

sudo apt update
sudo apt upgrade -y

3) Install Java OpenJDK 11

sudo apt install openjdk-11-jdk

java -version

4) Install Apache Maven and Ant.

DSpace 7 requires maven 3.3.x or above and and 1.8.x or above.

sudo apt-get install ant ant-optional maven -y

mvn -v
ant -version

5) Install PostgreSQL, one of the version: 11.x, 12.x or 13.x (with pgcrypto installed)

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'

sudo apt install gnupg -y

wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -


sudo apt-get update

sudo apt-get install postgresql-12 postgresql-contrib-12 libpostgresql-jdbc-java -y
```

```
sudo nano /etc/postgresql/12/main/pg_hba.conf
```

Add the following line to the bottom of the file then save and close.

```
host    dspace       dspace      127.0.0.1/32         md5
```

Next, execute the following 3 commands in succession to change database user permissions to “trust” only.

```
sudo sed -i 's/ident/trust/' /etc/postgresql/12/main/pg_hba.conf
sudo sed -i 's/md5/trust/' /etc/postgresql/12/main/pg_hba.conf
sudo sed -i 's/peer/trust/' /etc/postgresql/12/main/pg_hba.conf
```

Restart PostgreSQL database for these changes to take effect.

```
sudo systemctl restart postgresql
sudo su postgres
cd ~
createuser dspace
createdb dspace -E UNICODE
```

```
# Next we need to grant permissions to the database user called dspace to the database we just created
```

```
psql -d dspace
```

```
# Then start by creating the pgcrypto extension to the dspace database
```

```
CREATE EXTENSION pgcrypto;
```

```
# Create the password
```

```
ALTER ROLE dspace WITH PASSWORD 'your-db-password-here';
```

```
# Replace the text in quotes above to a secret password of your choice on a production system !
```

Then give the **dspace** database user ownership of the **dspace** database as shown here.

```
ALTER DATABASE dspace OWNER TO dspace;
```

Then give all privileges to **dspace** database user on the **dspace** database with the command below

```
    GRANT ALL PRIVILEGES ON DATABASE dspace TO dspace;

# To view the list of databases existing on the system together with other details like database ownership, issue the following command

    \l
# Exit the PostgreSQK shell by the following command

    \q

# Exit the postgres user session with the following command

    exit

# Restart PostgreSQL with the following command

    sudo systemctl restart postgresql
```

6) Install Apache Solr

 # Solr 8.11.1 or above is recommended

Create a directory to download and install solr. In this guide, we shall use /opt/solr-8.11 to depict the version of solr we shall use. DSpace 7 requires Solr 8.11.1 or later.

```
    sudo mkdir /opt/solr-8.11
```

Change to the directory we just created

```
    cd /opt/solr-8.11
```

Change the directory owner to the dspace user

```
    sudo chown -R dspace:dspace /opt/solr-8.11
```

Then download solr-8.11.1 from there with the following command

```
    wget -c https://downloads.apache.org/lucene/solr/8.11.1/solr-8.11.1.tgz
```

```
# Extract the downloaded archive with this command
   tar xvf solr-8.11.1.tgz
# Move the contents to /opt/solr-8.11 directory
   cp -rf /opt/solr-8.11/solr-8.11.1/* /opt/solr-8.11/
# Remove redundant directory
   rm -rf solr-8.11.1
# Install the solr service
   ./bin/install_solr_service.sh /opt/solr-8.11/solr-8.11.1.tgz
```

Confirm solr is running with

```
   systemctl status solr
```

If you need to start solr:

```
   systemctl start solr
# If you access http://server-url:8983/solr in your browser, solr admin dashboard should load. If it doesn't, review the steps followed again
```

7) Install Tomcat 9

Only Tomcat 9 is supported at this time. Tomcat 10 results in a display issue in the backend's Hal Browser.

See https://github.com/DSpace/DSpace/issues/8173 for more details

```
   sudo apt install tomcat9 -y
# Edit /etc/default/tomcat9 and define JAVA_HOME
```

```
sudo nano /etc/default/tomcat9
```

The value JAVA_HOME basically indicates to the system where your java is installed and this will depend on how you installed java and your operating system. An example of JAVA_HOME is shown below

```
JAVA_HOME=/usr/lib/jvm/java-1.11.0-openjdk-amd64
```

Locate the setting JAVA_OPTS and adjust maximum and minimum memory settings accordingly depending on how much RAM your system has. Tomcat8 requires at least 1GB memory to function normally. The following setting is good for a system with 4GB RAM

```
JAVA_OPTS="-Djava.awt.headless=true -Xmx2048m -Xms1024m -XX:MaxPermSize=1024m"
```

Save and close the file.

Alter Tomcat's default configuration to support searching and browsing of multi-byte UTF-8 correctly. Edit the file ***server.xml*** by editing as shown.

```
sudo nano /etc/tomcat9/server.xml
```

Ensure that the active **connector** element in the file is similar to the one below. You can comment out the existing element and paste in the one below:

```
# For mod_proxy_http:

<Connector port="8080"
               minSpareThreads="25"
               enableLookups="false"
               redirectPort="8443"
               connectionTimeout="20000"
               disableUploadTimeout="true"
               URIEncoding="UTF-8"/>

# For mod_proxy_ajp:

<Connector protocol="AJP/1.3" port="8009" redirectPort="8443" URIEncoding="UTF-8" />
```

```
# Save and close the file, then restart tomcat9 as shown below
```

```
sudo systemctl restart tomcat9
```

Now, when you access the ip address of your system via the browser at port 8080 i.e., http://ip-address:8080, you should see the default tomcat page, **It works !** . If not, you need to review the process

8) Git and install dspace cris

```
# I have more space in /home. It is separate partition with free space. We will deploy dspace in /opt/dspace and will work with sources from /home/src
```

```
sudo mkdir -p /home/dspace-cris-7
sudo ln -s /home/dspace-cris-7 /opt/dspace-cris-7
sudo ln -s /opt/dspace-cris-7 /opt/dspace
sudo mkdir -p /home/src && cd /home/src
sudo chown -R dspace:dspace /home/dspace-cris*
```

```
#(Optional) IP to City Database for Location-based Statistics
```

```
# I prefer to use/install DB-IP's City Lite database (in MMDB format)
```

```
Open in the browser the link below:
```

```
https://db-ip.com/db/download/ip-to-city-lite
```

```
Then get the link to db, like example below:
```

```
curl -OL https://download.db-ip.com/free/dbip-city-lite-XX_XX-CC.mmdb.gz
```

```
# download latest release from here https://github.com/4Science/DSpace/releases
```

```
curl -OL https://github.com/4Science/DSpace/archive/refs/tags/dspace-cris-2022.01.01.tar.gz
```

```
# untar it and rename extracted folder to dspace-parent:
```

```
rm -rdf ./src-dspace-cris-backend && rm -rdf `ls -1 | grep -i DSpace-dspace-cris`
```

```
tar zxvf dspace-cris-2022.01.01.tar.gz
cp -r `ls -1 | grep -i DSpace-dspace-cris` src-dspace-cris-backend
cd ./src-dspace-cris-backend
cp dspace/config/local.cfg.EXAMPLE  dspace/config/local.cfg


nano dspace/config/local.cfg
```

Edit local.cfg and modify to your correct values. More variables in source [1].

```
dspace.dir=/opt/dspace
```

Do not end with '/'

```
dspace.server.url = https://api.dspacecris.university.edu/server
```

Do not end with '/'

```
dspace.ui.url = https://dspacecris.university.edu
db.password =  your-db-password-here
```

Compile/build dspace package

```
sudo mvn package
```

After this we will start install of dspace in directory specified in local.cfg

```
cd dspace/target/dspace-installer
ant fresh_install
```

9) Create an Administrator Account:

Create an initial administrator account from the command line (change [dspace] with your dspace path, in my case is /opt/dspace):

```
[dspace]/bin/dspace create-administrator
```

Ex:

```
/opt/dspace/bin/dspace create-administrator
```

10) Initialize your Database

```
# While this step is optional (as the DSpace database should auto-initialize itself on first startup), it's always good to verify one last time that your database connection is working properly. To initialize the database run (change [dspace] with your dspace path, in my case is /opt/dspace):
```

```
[dspace]/bin/dspace database migrate
```

Ex:

```
/opt/dspace/bin/dspace database migrate
```

11) Configure dspace and solr

~~sudo cp -R /opt/dspace/solr/* /var/solr/data/~~

~~sudo chown -hR solr:solr /var/solr/data~~[1]

```
# NOTE: On Debian systems the configsets may be under /var/solr/data/configsets
```

```
sudo cp -r /opt/dspace/solr/* /opt/solr/server/solr/configsets
```

```
# Make sure everything is owned by the system user who owns Solr
# Usually this is a 'solr' user account
```

```
sudo chown -R solr:solr /opt/solr/server/solr/configsets
sudo systemctl restart solr
systemctl status solr
```

```
# OR
```

```
sudo -u solr /opt/solr/bin/solr status
```

1 Hm... I don't think this is needed, but...

12) Deploy Server web application

```
sudo mkdir /opt/dspace/assetstore
sudo chown -LR tomcat:tomcat /opt/dspace
```

On Debian systems, you may also need to modify or override the "tomcat.service" file to specify the DSpace installation directory in the list of ReadWritePaths. For example:

```
sudo mkdir -p /etc/systemd/system/tomcat9.service.d
sudo nano /etc/systemd/system/tomcat9.service.d/override.conf
```

```
# Replace [dspace] with the full path of your DSpace install

[Service]
ReadWritePaths=[dspace]

# Ex.

[Service]
ReadWritePaths=/opt/dspace
```

After this we should create a link for dspace applications in tomcat.

```
sudo ln -s /opt/dspace/webapps/server /var/lib/tomcat9/webapps/server
sudo systemctl restart tomcat9
```

In this case using /opt/dspace/bin/dspace **IS RECOMMENDED (from this step)** as is shown in the command example below:

```
sudo -u tomcat -s /opt/dspace/bin/dspace healthcheck
```

13) Install Apache HTTPD as reverse proxy (for production)

Install Apache HTTPD, e.g.

```
sudo apt install apache2
```

Install the mod_proxy and mod_headers modules, e.g.

```
    sudo a2enmod proxy
    sudo a2enmod headers
```

If you plan to use mod_proxy_ajp:

```
    sudo a2enmod proxy_ajp
```

Alternatively, you can choose to use mod_proxy_http to create an http proxy. A separate example is commented out below.

In this case:

```
    sudo a2enmod proxy_http
```

I'll plan to use mod_proxy_http.

1 we will disconect default conf:

```
    sudo rm /etc/apache2/sites-enabled/000-default.conf
```

2 we will create virtualhost for dspace backend:

```
    sudo nano /etc/apache2/sites-available/backend.conf
```

3 Enter virtualhost content for dspace backend. We will configure in first steps only HTTP (port 80). And in the latest step will configure certbot and HTTPS (port 443) for both backend and frontend.

```
    <VirtualHost _default_:80>
       ServerName api.dspacecris.university.edu

       ServerAdmin webmaster@localhost
       DocumentRoot /var/www/html

       # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
       # error, crit, alert, emerg.
       # It is also possible to configure the loglevel for particular
       # modules, e.g.
```

```
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/dspace_backend_error.log
    CustomLog ${APACHE_LOG_DIR}/dspace_backend_access.log combined

    # Proxy all HTTP/HTTPS requests to "/server" from Apache to Tomcat via AJP connector
    #ProxyPass /server ajp://localhost:8009/server
    #ProxyPassReverse /server ajp://localhost:8009/server

    # If you would rather use mod_proxy_http as an http proxy to port 8080
    # then use these settings instead
    ProxyPass /server http://localhost:8080/server
    ProxyPassReverse /server http://localhost:8080/server
    # When using mod_proxy_http, you need to also ensure the X-Forwarded-Proto header is sent
    # to tell DSpace it is behind HTTPS, otherwise some URLs may continue to use HTTP
    # (requires installing/enabling mod_headers)
    #RequestHeader set X-Forwarded-Proto https
</VirtualHost>
```

```
# Activate backend site (http)

    sudo a2ensite backend

# Reload apache2

    sudo systemctl reload apache2

# Test api address in the browser for your backend (http port), like in ex:
http://api.dspacecris.university.edu/server

# You should see HAL Browser.
```

14) Working with frontend. Installing requirements.

(1) Installing nodejs. At this moment active LTS version is 16. Check source [8] for latest active LTS and source [9] for install steps.

NOTE: Be attentive at source [10]. For version 7.2 frontend maximum recommended nodejs is version 16.

To install nodejs I used the following commands for installation:

```
sudo curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo bash -
sudo apt-get install gcc g++ make
sudo apt-get install -y nodejs
```

In source [10] you can see that we need yarn v.1 for our needs.

Yarn v1.x is available at https://classic.yarnpkg.com/. It can usually be install via NPM (or through your Linux distribution's package manager). *We do NOT currently support Yarn v2.*

 # You may need to run this command using "sudo" if you don't have proper privileges

```
sudo npm install --global yarn
```

 # Installing PM2 (or another Process Manager for Node.js apps) (optional, but recommended for Production)

 # You may need to run this command using "sudo" if you don't have proper privileges

```
sudo npm install --global pm2
```

15) Creating directories for frontend.

Note:

[dspace-angular] will be in this guide in ***/home/src-angular/src-dspace-cris-angular***
[dspace-ui-deploy] will be in this guide in ***/opt/dspace-angular***

Be attentive all actions here are taken with user dspace!

I have more space in /home. It is separate partition with free space.

```
sudo mkdir -p /home/dspace-cris-angular-7
sudo ln -s /home/dspace-cris-angular-7 /opt/dspace-cris-angular-7
sudo ln -s /opt/dspace-cris-angular-7 /opt/dspace-angular
sudo mkdir -p /home/src-angular && cd /home/src-angular
sudo chown -R dspace:dspace /home/dspace-cris*
sudo chown -R dspace:dspace /home/src*
```

download latest release from here https://github.com/4Science/dspace-angular/releases

```
curl -OL https://github.com/4Science/dspace-angular/archive/refs/tags/dspace-cris-2022.01.01.tar.gz
```

untar it and rename extracted folder to dspace-parent:

```
rm -rdf ./src-dspace-cris-angular && rm -rdf `ls -1 | grep -i dspace-angular-dspace-cris`
tar zxvf dspace-cris-2022.01.01.tar.gz
cp -r `ls -1 | grep -i dspace-angular-dspace-cris` src-dspace-cris-angular
cd ./src-dspace-cris-angular
```

Install the local dependencies:

```
yarn install
```

Build/Compile: Build the User Interface for Production. This builds source code (under [dspace-angular]/src/) to create a compiled version of the User Interface in the [dspace-angular]/dist folder. This /dist folder is what we will deploy & run to start the UI.

```
yarn build:prod
```

You only need to rebuild the UI application if you change source code (under [dspace-angular]/src/). Simply changing the configurations (e.g. config.prod.yml, see below) do not require a rebuild, but only require restarting the UI.

Deployment (to [dspace-ui-deploy]): Choose/Create a directory on your server where you wish to run the compiled User Interface. We'll call this [dspace-ui-deploy] . See *note* above about path.

Copy the entire [dspace-angular]/dist/ folder to this location. For example:

```
cp -r [dspace-angular]/dist [dspace-ui-deploy]
```

or like in this case:

```
cp -r /home/src-angular/src-dspace-cris-angular/dist /opt/dspace-angular
```

WARNING: At this time, you MUST copy the entire "dist" folder and make sure NOT to rename it.

Therefore, the directory structure should look like this:

Contents of [dspace-ui-deploy] folder

[dspace-ui-deploy]
/dist
/browser (compiled client-side code)
/server (compiled server-side code, including "main.js")
/config (Optionally created in the "Configuration" step below)
/config.prod.yml (Optionally created in the "Configuration" step below)

NOTE: the OS account which runs the UI via Node.js (see below) MUST have write privileges to the [dspace-ui-deploy] directory (because on startup, the runtime configuration is written to [dspace-ui-deploy]/dist/browser/assets/config.json)

```
sudo chown -LR dspace:dspace /opt/dspace-angular
```

Creating config directory and config.prod.yaml

```
mkdir -p /opt/dspace-angular/config
nano /opt/dspace-angular/config/config.prod.yaml
```

Example of config.prod.yaml is below. More info see in source [10]

```
# The "ui" section defines where you want Node.js to run/respond.
# It often is a *localhost* (non-public) URL, especially if you are using a Proxy.
# In this example, we are setting up our UI to just use localhost, port 4000.
# This is a common setup for when you want to use Apache or Nginx to handle HTTPS
# and proxy requests to Node on port 4000
ui:
  ssl: false
  host: localhost
  port: 4000
  nameSpace: /

# This example is valid if your Backend is publicly available at https://api.dspacecris.university.edu/server/
# The REST settings MUST correspond to the primary/public URL of the backend.
# Usually, this means they must be kept in sync
# with the value of "dspace.server.url" in the backend's local.cfg
rest:
  ssl: true
  host: api.dspacecris.university.edu
  port: 443
  nameSpace: /server
```

Adding virtualhost to Apache.

```
sudo nano /etc/apache2/sites-available/frontend.conf
```

Add contents:

```
<VirtualHost _default_:80>
  ServerName dspacecris.university.edu

  ServerAdmin webmaster@localhost
  DocumentRoot /var/www/html

  # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
  # error, crit, alert, emerg.
  # It is also possible to configure the loglevel for particular
  # modules, e.g.
  #LogLevel info ssl:warn

  ErrorLog ${APACHE_LOG_DIR}/dspace_frontend_error.log
  CustomLog ${APACHE_LOG_DIR}/dspace_frontend_access.log combined

  # If you would rather use mod_proxy_http as an http proxy to port 8080
  # then use these settings instead
  # Be attentive to the last character on the address below: /
  ProxyPass / http://localhost:4000/
  ProxyPassReverse / http://localhost:4000/
  # When using mod_proxy_http, you may need to also ensure the X-Forwarded-Proto header is sent
  # to tell DSpace it is behind HTTPS, otherwise some URLs may continue to use HTTP
  # (requires installing/enabling mod_headers)
  #RequestHeader set X-Forwarded-Proto https
</VirtualHost>
```

Activate frontend site (http)

```
sudo a2ensite frontend
```

Reload apache2

```
sudo systemctl reload apache2
```

Start up the User Interface. For test run you can check source [10]. In our case we use PM2 (should be installed already).

First you need to create a PM2 JSON configuration file which will run the User Interface.

This file can be named anything & placed where ever you like, but We RECOMMEND save it to your deployment directory (e.g. [dspace-ui-deploy]/dspace-ui.json).

```
nano /opt/dspace-angular/dspace-ui.json
```

And add content like in example below:

```
{
    "apps": [
        {
           "name": "dspace-ui",
           "cwd": "/opt/dspace-angular",
           "script": "dist/server/main.js",
           "env": {
              "NODE_ENV": "production"
           }
        }
    ]
}
```

NOTE: The "cwd" setting MUST correspond to your [dspace-ui-deploy] folder path. And configuration we did already in dist/config.

Now, start the application using PM2 using the configuration file you created in the previous step.

We should start application using PM2 from `/opt/dspace-angular`:

```
cd /opt/dspace-angular
```

In this example, we are assuming the config is named "dspace-ui.json". To start it, you'd run:

```
pm2 start dspace-ui.json
```

To see the logs, you'd run:

```
pm2 logs
```

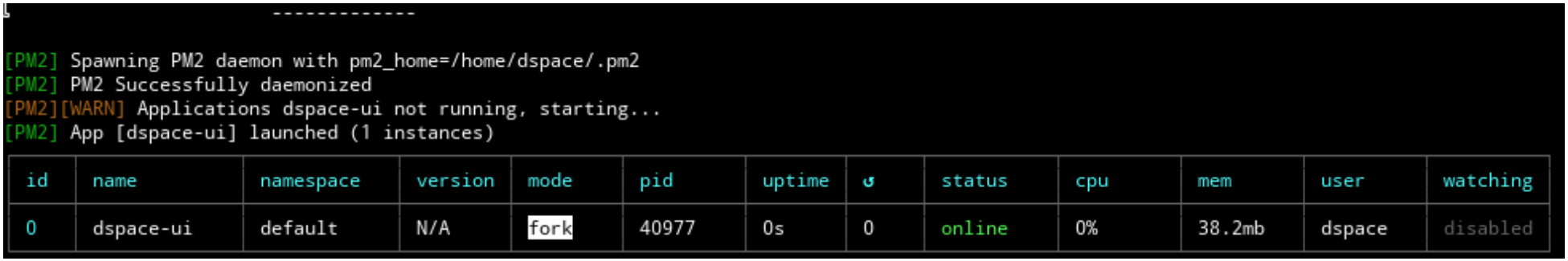
To stop it, you'd run:

```
pm2 stop dspace-ui.json
```

If you need to change your PM2 configs, delete the old config. Create new config file and and restart.

```
pm2 delete dspace-ui.json
```

When application is started with PM2 you 'lll see something like in example below:



Install logrotare for PM2

```
cd /opt/dspace-angular
pm2 install pm2-logrotate
```

If you want to see logs for frontend, I can suggest to start in directory /opt/dspace-angular with command below:

```
pm2 logs
```

16) Install Certbot and generate certificates

Installing snap on Debian

sudo apt update
sudo apt install snapd
sudo snap install core; sudo snap refresh core

sudo apt-get remove certbot
sudo snap install --classic certbot
sudo ln -s /snap/bin/certbot /usr/bin/certbot

```
# In the command below change [email@address] with your or administrator email address.
```

```
sudo certbot certonly --apache --agree-tos -m [email@address] --dry-run -d dspacecris.university.edu -d api.dspacecris.university.edu
```

If test is success , remove **certonly** and **--dry-run** to start saving the certificates to disk.

```
sudo certbot --apache --agree-tos -m webmaster@university.edu -d dspacecris.university.edu -d api.dspacecris.university.edu
```

 # Note: you have limits for generate certificates. Check documentation before starting saving the certificates to disk.

If all was success, you'll see in sites-available two new files with -le-ssl.conf:

```
ls -al /etc/apache2/sites-available/ | grep le-ssl
```

As result you should see in sites-available:

dspace@dspacecris:/opt/dspace-angular$ ls -al /etc/apache2/sites-available/ | grep le-ssl
-rw-r--r-- 1 root root 1430 Eee xx 12:08 backend-le-ssl.conf
-rw-r--r-- 1 root root 1217 Eee xx 12:07 frontend-le-ssl.conf

And this sites should be activated:

ls -al /etc/apache2/sites-enabled/ | grep le-ssl

As result you should see in sites-available:

dspace@dspacecris:/opt/dspace-angular$ ls -al /etc/apache2/sites-enabled/ | grep le-ssl
lrwxrwxrwx 1 root root 48 Eee xx 12:06 backend-le-ssl.conf -> /etc/apache2/sites-available/backend-le-ssl.conf
lrwxrwxrwx 1 root root 49 Eee xx 12:06 frontend-le-ssl.conf -> /etc/apache2/sites-available/frontend-le-ssl.conf

Enable X-Forwarded-Proto https in backend and in frontend.

sudo nano /etc/apache2/sites-available/frontend-le-ssl.conf

Uncomment the line below:

RequestHeader set X-Forwarded-Proto https

Now change in backend:

sudo nano /etc/apache2/sites-available/frontend-le-ssl.conf

Uncomment the line below:

RequestHeader set X-Forwarded-Proto https

Reload apache2

```
sudo systemctl reload apache2
```

At this moment we should have an instance of Dspace Cris running. Let's test if all is ok.

Open in your browser:

- https://api.dspacecris.university.edu
- https://dspacecris.university.edu

{Re}Modify permissions for backend:

```
sudo chown -LR tomcat:tomcat /opt/dspace-angular
```

Hm. And If you have questions, like I can't create collections, or similar. Please see PDF file from source [2]. And also recommendations from source [11].

To initialize the default entity types, just run command below.

```
sudo -u tomcat -s /opt/dspace/bin/dspace dsrun org.dspace.app.util.InitializeEntityTypesOnly -d
```

Sources:

1. https://wiki.lyrasis.org/display/DSDOC7x/Installing+DSpace
2. https://wiki.lyrasis.org/display/DSPACECRIS/Technical+and+User+documentation
3. https://hyperlink.co.ke/2020/08/09/install-dspace-7-on-ubuntu-18-04/?utm_source=pocket_mylist
4. https://mohammedh.io/2021/07/05/install-dspace-cris-7-along-with-free-ssl-and-nginx/?utm_source=pocket_mylist
5. https://github.com/4Science/DSpace/releases
6. https://github.com/4Science/dspace-angular/releases
7. https://www.serverlab.ca/tutorials/linux/web-servers-linux/how-to-reverse-proxy-websockets-with-apache-2-4/
8. https://nodejs.org/en/about/releases/
9. https://github.com/nodesource/distributions/blob/master/README.md
10. https://wiki.lyrasis.org/display/DSDOC7x/Installing+DSpace#InstallingDSpace-BackendInstallation
11. https://groups.google.com/g/dspace-tech/c/L8OUgPPzZIo