Setting up DRK

1. Install docker and docker-compose

```
sudo apt update && sudo apt install -y docker.io docker-compose
```

2. Update the system

```
sudo apt update
sudo apt upgrade
```

3. Install DRK (This step is crucial. All submodules must be installed)

```
git clone --recursive https://gitlab.cc-asp.fraunhofer.de/iop-dataspace/drk-edc.git
```

4. The directory `~/drk-edc` should now exist.

5. The connector will be registered on the DAPS. For this, the `client_name` (the name of the client) and the `client_id` (the client's URL) are required.

   If the DAPS is already set up, skip to step 6.

   Example Values:

   ```
   client_name = base_connector_1
   client_id = iop-ds-inst1.dbis.rwth-aachen.de
   ```

   DAPS Setup:

   a. Navigate to the directory:

   ```
   cd ~/omejdn-daps/
   ```

   b. Run the following command:

   ```
   scripts/register_connector_clientid.sh base_connector_1 iop-ds-inst1.dbis.rwth-aachen.de idsc:BASE_SECURITY_PROFILE
   ```

   c. Verify that the following entry is created in `~/omejdn-daps/config/clients.yml`:

```
- client_id: iop-ds-inst1.dbis.rwth-aachen.de
  client_name: base_connector_1
  grant_types: client_credentials
  token_endpoint_auth_method: private_key_jwt
  scope: idsc:IDS_CONNECTOR_ATTRIBUTES_ALL
  attributes:
    - key: idsc
      value: IDS_CONNECTOR_ATTRIBUTES_ALL
    - key: securityProfile
      value: idsc:BASE_SECURITY_PROFILE
    - key: referringConnector
      value: http://base_connector_1.demo
    - key: "@type"
      value: ids:DatPayload
    - key: "@context"
      value: https://w3id.org/idsa/contexts/context.jsonld
    - key: transportCertsSha256
      value:
d4a81f9dcfc96b163394cbbc9712cb9160cfcaeef8ffab97974bd769ae0f8ea0
```

   d. Ensure the following files are created in `~/omejdn-daps/keys`:

```
base_connector_1.cert
base_connector_1.key
```

e. Convert the keys to .p12 format. Run the following command in the `~/omejdn-daps` directory (the password must contain at least six characters):

```
openssl pkcs12 -export -in keys/base_connector_1.cert -inkey
keys/base_connector_1.key -out base_connector_1.p12
```

f. Verify the file `~/omejdn-daps/base_connector_1.p12` exists.

g. Transfer the .p12 file to the DRK in `~/drk-edc/crypto/`:

```
scp base_connector_1.p12 iop@137.226.58.137:~/drk-edc/crypto/
das geht noch übers ssh vielleicht geht da nicht. Sonst was mit http
https etc versuchen
```

6. Generate JKS File

a. Navigate to the directory:
`cd ~/drk-edc/crypto`

b. Ensure `base_connector_1.p12` exists.

c. Install `Keytool` if not already installed:
`sudo apt install openjdk-21-jre-headless`

d. Generate the JKS file:
`./generate_ski_aki.sh base_connector_1.p12 PASSWORD`
(Replace PASSWORD with the password used when creating the .p12 file)

e. If an error (e.g. permission denied) occurs, grant permissions and retry:
`chmod +x generate_ski_aki.sh`

f. Verify that the following entry is created:
`~/drk-edc/crypto/base-connector_1.jks`

7. Configure compose_params.yaml

a. Create or edit the file `~/drk-edc/launchers/compose_params.yaml` with the following content:

```
target_launcher_dir: "base_connector"
conn_type: connector

hostname: "iop-ds-inst1.dbis.rwth-aachen.de" # Adjust this
value to then actual hostname
name: "iop-ds-inst1.dbis.rwth-aachen.de" # This value should
match the hostname
full_title: "Base Connector 1" # Connector title, Adjust this
information
description: "Base Connector 1 for DRK EDC" # Connector
description, Adjust this information
```

```
connector_owner: "DBIS RWTH" # The institution the connector
is used at, Adjust this information

client_id: "iop-ds-inst1.dbis.rwth-aachen.de"  # This value
should match the Client ID registered in DAPS
daps_token_url: "http://137.226.58.139/auth/token" # DAPS
token
endpoint on the DAPS VM (shouldnt be changed)
daps_jwks_url: "http://137.226.58.139/auth/jwks.json" # JWKS
URL on the DAPS VM (shouldnt be changed)
keystore_name: "base_connector_1.jks"  # Name of the JKS file
created in the crypto directory
keystore_passwd: "123456789"   # Password used when creating
the JKS file

apikey_guard_verification_url:
"http://137.226.58.137:8080/introspect"  # Example URL for
introspection, adjust if necessary
apikey_guard_client_id: "apikey-guard"
apikey_guard_client_secret: ""

iam_url: "http://137.226.58.137/iam"  # Example URL for the
IAM server, adjust if necessary
edc_connectors_endpoint: "http://137.226.58.139/connectors" #
Endpoint to retrieve the list of active connectors (shouldnt
be changed)

protocol: "http"  # Protocol for the connection (http or
https)

nginx_ssl_fullchain: "none"
nginx_ssl_privkey: "none"
```

b. The fields `hostname`, `name`, and `client_id` contain the previously defined `client_id: iop-ds-inst1.dbis.rwth-aachen.de`

c. The fields `full_title`, `description`, and `connector_owner` can be defined as needed

d. The fields `daps_token_url`, `daps_jwks_url`, and `edc_connectors_endpoint` contain the IP of the DAPS `http://137.226.58.139`

e. The `keystore_name` field contains the name of the generated .jks file, and `keystore_passwd` contains the associated password

f. The fields `apikey_guard_verification_url` and `iam_url` contain the IP of the connector `http://137.226.58.137`, i.e., the IP of the domain `iop-ds-inst1.dbis.rwth-aachen.de`

g. To make the page secure, change the `protocol` field to `"https"` and add the path to the corresponding certificates to `nginx_ssl_fullchain` and `nginx_ssl_privkey`.

h. This file is available in the directory as
`compose_params.yaml.example_filled_out`

8. Navigate to the directory:
   `cd ~/drk-edc/launchers`

9. Run the following command:
   `python3 create_compose.py compose_params.yaml`

10. Navigate to the directory:
    `cd ~/drk-edc/launchers/base_connector`

11. Run the following command:
    `sudo docker compose up --build`

12. The connector should now be accessible in the browser at the `hostname/client_id` URL.

```
Probleme mit der Synchronisation?
      sudo nano /etc/systemd/timesyncd.conf
und  NTP=ntp1.rwth-aachen.de
```