SETTING UP SOVRIN

https://stackoverflow.com/questions/65931922/cant-install-indy-sdk-in-ubuntu-20-04

https://www.linkedin.com/pulse/hands-on-decentralized-identityssi-setting-up-indy-node-sripathi/ ?

This page is a step-by-step guide on how to set up Hyperledger indy network and connecting to it with python code

The official guides are mostly done for ubuntu 16.04 and 18.04 → [github]

However, I used ubuntu 22.04 and found some work arounds to get it to work → this guide will be for 22.04 ubuntu distribution

Step 1 – install Docker:

[link]

(1) remove any docker installs for a fresh install

```
sudo apt-get remove docker docker-engine docker.io containerd runc
```

(2) update and install required packages

```
sudo apt-get update
```

```
sudo apt-get install \ ca-certificates \ curl \ gnupg
#TODO -- CHANGE TO sudo apt-get install ca-certificates individually each
#TODO -- ADD COMMANDS HERE
```

(3) set up a repository

```
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com
/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

(4) install docker

```
sudo apt-get update
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

(5) verify installation → should return "hello from docker"

```
sudo docker run hello-world
#TODO -- install libssl1.1
#TODO -- install npm
```

Step 2 – install indy-sdk :

Sovrin is built on hyperledger indy, the indy sdk is what allows you to interact with it through code, the base sdk was created for c++ (I believe), but there are many wrappers for python, java ...

The official guide on how to install it covers only ubuntu 16.04 and ubuntu 18.04 [link]

The commands to install it are pretty straight forward:

sudo apt-get install ca-certificates -y
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys CE7709D068DB5E88
sudo add-apt-repository "deb https://repo.sovrin.org/sdk/deb (xenial|bionic) {release channel}"
sudo apt-get update
sudo apt-get install -y {library}.

→ {library} must be replaced with libindy, libnullpay, libvcx or indy-cli

→ (xenial|bionic) xenial for 16.04 Ubuntu and bionic for 18.04 Ubuntu.

→ {release channel} must be replaced with master, rc or stable to define corresponded release channel. Please See the section "Release channels" above for more details.

I found a workaround for ubuntu 20.04 that works for 22.04 as well [link]:

sudo apt-get update -y && apt-get install -y \ gnupg \ ca-certificates

curl -sSL 'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xCE7709D068DB5E88' > CE7709D068DB5E88.gpg

sudo gpg --no-default-keyring --keyring gnupg-ring:/etc/apt/trusted.gpg.d/sovrin.gpg --import CE7709D068DB5E88.gpg

sudo chmod 644 /etc/apt/trusted.gpg.d/sovrin.gpg

rm CE7709D068DB5E88.gpg

echo "deb https://repo.sovrin.org/sdk/deb bionic stable" | sudo tee -a /etc/apt/sources.list

sudo apt-get update -y

sudo apt-get install -y libindy

sudo npm install --save indy-sdk

Step 3 – build and run docker container :

Now we need to clone from github

→ git clone https://github.com/hyperledger/indy-sdk#indy-sdk

#TODO – wrong link

got into the folder → cd indy-sdk

build docker → sudo `docker build -f ci/indy-pool.dockerfile -t indy_pool .`

run docker → sudo `docker run -itd -p 9701-9708:9701-9708 indy_pool`

You only need to build it once, and run at each computer restart

#TODO – install python3-pip – pip install python3-indy

Step 4 – first interactions with node :

link to video part 1

link to video part 2

STEP1:

connect to the pool

STEP2:

configure the stewards

STEP3:

register the DID for government - university - company → these will be trust anchors, and will be enrolled to the ledger through the steward configured in S2

STEP4:

for each different credential, a schema has to be defined – fields that the credential must contain – this makes the credential standardized

this transcript schema will be issued by the government

transcripts can't be deleted, as they are registered to the ledger, however they may be updated through the version field

this operation returns a transcript id – so any one trying to issue a credential will be able to find its definitions

STEP5:

for each (here university) a credential definition must be created, the definition tells verifiers how they can verify the signatures – might also have public keys

the university loads the definition set by the government, and then add some more fields, like algorithm used, or even it it can be revoked or not

STEP6:

Alice gets transcript definition from university and creates request

university first creates a transcript credential offer, and sends it to Alice

based on the offer, Alice will be able to create a request for this specific credential

to do this Alice must create a master secret, to make sure that that credential is only valid when combined with the master secrete

STEP7:

university issues credential

university will be able to issue the credential, and send it back to Alice

at which point Alice must store it in her own wallet

STEP8:

company defines job application – Alice applies

first the company creates a job application proof request, which has some defined properties, some of them are attributes that an applicant must fill, name, age which have no restrictions

however, some attributes have restrictions, like degree – this proof request is sent to Alice

To apply, Alice must get the credentials for each field defined in the proof request

Alice creates proof of job application, and sends it to the company

STEP9:

company validates application claims

link to video at important time