# Hyperledger Indy

Theory and Applications of Blockchain (CS61065) - Tutorial 3

# Indy

**Hyperledger Indy** is an open-source, decentralized identity management platform for individuals and organizations to have complete control over their digital identities. It provides a secure and scalable infrastructure to manage and store decentralized identity information

# Key Features

- Distributed ledger - built for decentralized identity

- DIDs (Decentralized Identifiers) without requiring any centralized resolution authority

- Verifiable Credentials in an interoperable format for exchange of digital identity attributes

- Zero Knowledge Proofs which prove that some or all of the data in a set of Claims is true without revealing any additional information

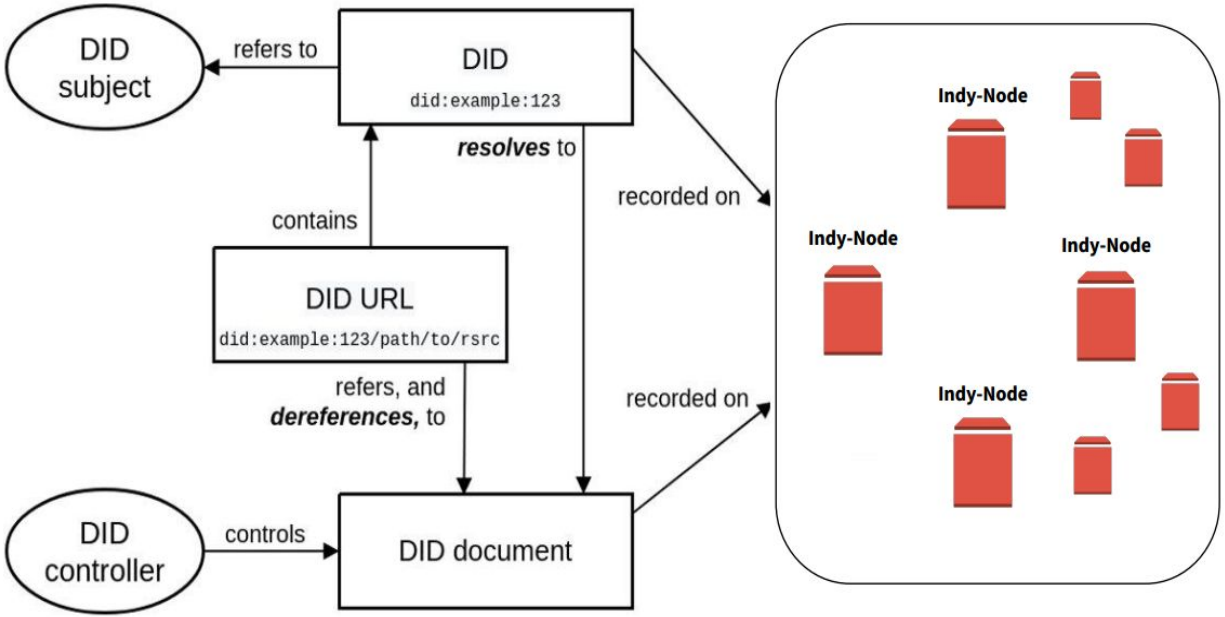
# Indy Work Projects

Indy-Plenum:

- ❏ It is the ledger part
- ❏ Used for consensus in Indy
- ❏ https://github.com/Hyperledger/indy-plenum

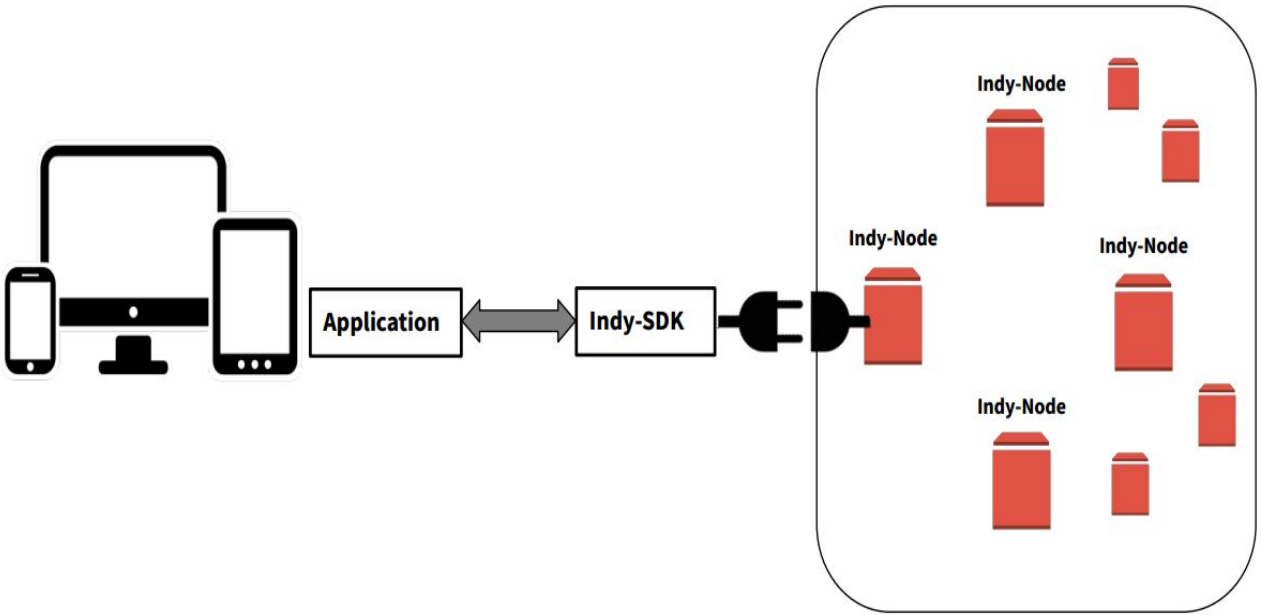
- Indy-Node:
  - ❏ This codebase embodies all the functionality to run nodes
  - ❏ It provide a self-sovereign identity ecosystem on top of the edger
  - ❏ https://github.com/Hyperledger/indy-node
- Indy-SDK
  - ❏ Provides services and interface to applications for interacting with Indy network
  - ❏ Indy- https://github.com/Hyperledger/indy-sdk

# Indy and Decentralized Identities




* Courtesy: NPTEL Lecture Series

# Indy Connectivity




* Courtesy: NPTEL Lecture Series

# Indy Set Up using Indy SDK

indy node

## Start Indy Pool using Indy SDK

**Clone indy-sdk**

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

cd indy-sdk

-

# Indy Set Up

**Build and run indy pool docker image**

```
docker build -f ci/indy-pool.dockerfile -t indy_pool .
docker run -itd -p 9701-9708:9701-9708 indy_pool
```

**Alternatively an easy way to start indy pool**

- docker run -itd -p 9701-9708:9701-9708 mailtisen/indy_pool:latest

# Also Installing Indy from Indy Node

**Clone indy-node**
git clone https://github.com/hyperledger/indy-node.git

Move to the directory indy-node/environment/docker/pool
./pool_start.sh [number of nodes in pool] [IP addresses of nodes] [number of clients] [port for the first node]
Eg.
./pool_start.sh 4 10.0.0.2,10.0.0.3,10.0.0.4,10.0.0.5 10 9701

# Install SDK and Wrappers

Ubuntu based distributions (Ubuntu 18.04)
It is recommended to install the SDK packages with APT. (Not tested yet with Snap)

-

sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys CE7709D068DB5E88
(use other keyservers if ubuntu one is not available)

sudo add-apt-repository "deb https://repo.sovrin.org/sdk/deb (xenial|bionic) {release channel}"

# Install SDK and Wrappers

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 and older Ubuntu and bionic for 18.04 Ubuntu.
- {release channel} must be replaced with master, rc or stable to define a corresponding release channel.

Please see the "Release channels" section in the Hyperledger Indy documentation for more details.

# Install SDK and Wrappers

**Install Python3 Wrapper Library**

pip install python3-indy

**Alternatively**

sudo apt install python3-pip
pip3 install python3-indy

# Connect To Indy Pool → Find path to genesis txn

**Default pool genesis txn:**

{"reqSignature":{},"txn":{"data":{"data":{"alias":"Node1","blskey":"4N8aUNHSgjQVgkpm8nhNEfDf6txHznoYREg9kirmJrkivgL4oSEimFF6nsQ6M41QvhM2Z33nves5vfSn9n1UwNFJBYtWVnHYMATn76vLuL3zU88KyeAYcHfsih3He6UHcXDxcaecHVz6jhCYz1P2UZn2bDVruL5wXpehgBfBaLKm3Ba","blskey_pop":"RahHY¡CvoNCtPTrVtP7nMC5eTYrsUA8WjXbdhNc8debh1agE9bGiJxWBXYNFbnJXoXhWFMvyqhqhRoq737YQemH5ik9oL7R4NTTCz2LEZhkgLJzB3QRQqJyBNyv7acbdHrAT8nQ9UkLbaVL9NBpnWXBTw4LEMePaSHEw66RzPNdAX1","client_ip":"127.0.0.1","client_port":9702,"node_ip":"127.0.0.1","node_port":9701,"services":["VALIDATOR"]},"dest":"Gw6pDLhcBcoQesN72qfotTgFa7cbuqZpkX3Xo6pLhPhv"},"metadata":{"from":"Th7MpTaRZVRYnPiabds81Y"},"type":"0"},"txnMetadata":{"seqNo":1,"txnId":"fea82e10e894419fe2bea7d96296a6d46f50f93f9eeda954ec461b2ed2950b62"},"ver":"1"}

# Connect To Indy Pool

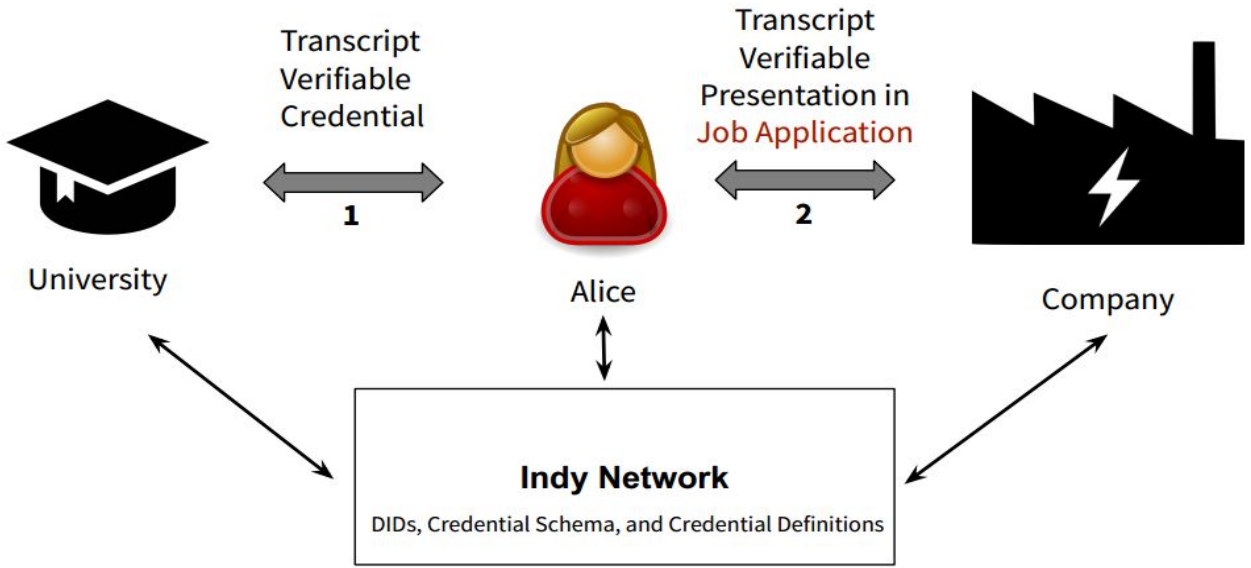
**Default pool genesis txn:**

{"reqSignature":{},"txn":{"data":{"data":{"alias":"Node2","blskey":"37rAPpXVoxzKhz7d9gkUe52XuXryuLXoM6P6LbWDB7LSbG62Lsb33sfG7zqS8TK1MXwuCHj1FKNzVpsnafmqLG1vXN88rt38mNFs9TENzm4QHdBzsvCuoBnPH7rpYYDo9DZNJePaDvRvqJKByCabubJz3XXKbEeshzpz4Ma5QYpJqjk","blskey_pop":"Qr658mWZ2YC8JXGXwMDQTzuZCWF7NK9EwxphGmcBvCh6ybUuLxbG65nsX4JvD4SPNtkJ2w9ug1yLTj6fgmuDg41TgECXjLCij3RMsV8CwewBVgVN67wsA45DFWvqvLtu4rjNnE9JbdFTc1Z4WCPA3Xan44K1HoHAq9EVeaRYs8zoF5","client_ip":"127.0.0.1","client_port":9704,"node_ip":"127.0.0.1","node_port":9703,"services":["VALIDATOR"]},"dest":"8ECVSk179mjsjKRLWiQtssMLgp6EPhWXtaYyStWPSGAb"},"metadata":{"from":"EbP4aYNeTHL6q385GuVpRV"},"type":"0"},"txnMetadata":{"seqNo":2,"txnId":"1ac8aece2a18ced660fef8694b61aac3af08ba875ce3026a160acbc3a3af35fc"},"ver":"1"}

{"reqSignature":{},"txn":{"data":{"data":{"alias":"Node3","blskey":"3WFpdbg7C5cnLYZwFZevJqhubkFALBfCBBok15GdrKMUhUjGsk3jV6QKj6MZgEubF7oqCafxNdkm7eswgA4sdKTRc82tLGzZBd6vNqU8dupzup6uYUf32KTHTPQbuUM8Yk4QFXjEf2Usu2TJcNkdgpyeUSX42u5LqdDDpNSWUK5deC5","blskey_pop":"QwDeb2CkNSx6r8QC8vGQK3GRv7Yndn84TGNijX8YXHPiagXajyfTjoR87rXUu4G4QLk2cF8NNyqWiYMus1623dELWwx57rLCFqGh7N4ZRbGDRP4fnVcaKg1BcUxQ866Ven4gw8y4N56S5HzxXNBZtLYmhGHvDtk6PFkFwCvxYrNYjh","client_ip":"127.0.0.1","client_port":9706,"node_ip":"127.0.0.1","node_port":9705,"services":["VALIDATOR"]},"dest":"DKVxG2fXXTU8yT5N7hGEbXB3dfdAnYv1JczDUHpmDxya"},"metadata":{"from":"4cU41vWW82ArfxJxHkzXPG"},"type":"0"},"txnMetadata":{"seqNo":3,"txnId":"7e9f355dffa78ed24668f0e0e369fd8c224076571c51e2ea8be5f26479edebe4"},"ver":"1"}

{"reqSignature":{},"txn":{"data":{"data":{"alias":"Node4","blskey":"2zN3bHM1m4rLz54MJHYSwvqzPchYp8jkHswveCLAEJVcX6Mm1wHQD1SkPYMzUDTZvWvhuE6VNAkK3KxVeEmsanSmvjVkReDeBEMxeDaayjcZjFGPydyey1qxBHmTvAnBKoPydvuTAqx5f7YNNRAdeLmUi99gERUU7TD8KfAa6MpQ9bw","blskey_pop":"RPLagxaR5xdimFzwmzYnz4ZhWtYQEj8iR5ZU53T2gitPCyCHQneUn2Huc4oeLd2B2HzkGnjAff4hWTJT6C7qHYB1Mv2wU5iHHGFWkhnTX9WsEAbunJCV2qcaXScKj4tTfvdDKfLiVuU2av6hbsMztirRze7LvYBkRHV3tGwyCptsrP","client_ip":"127.0.0.1","client_port":9708,"node_ip":"127.0.0.1","node_port":9707,"services":["VALIDATOR"]},"dest":"4PS3EDQ3dW1tci1Bp6543CfuuebjFrg36kLAUcskGfaA"},"metadata":{"from":"TWwCRQRZ2ZHMJFn9TzLp7W"},"type":"0"},"txnMetadata":{"seqNo":4,"txnId":"aa5e817d7cc626170eca175822029339a444eb0ee8f0bd20d3b0b76e566fb008"},"ver":"1"}

# Demo Scenario



# Indy Roles

**STEWARDS**
Steward can onboard new actors in the system and assigns role to them.

**Trust Anchor(TA)**
TA's are the link between User and Stewards. TA can be banks, universities, hospitals, service providers, insurance companies. TA's are endorsers onboarded by Stewards.

# Indy Credential Workflow

**Indy Credential Workflow**

- Getting the ownership for Steward's DID
- Register DID for Government, University and Company
-
- Register Credential Schema
  - Government creates transcript schema for university.
- Create Credential Definition
  - University creates a credential definition.

# Indy Credential Workflow

**Indy Credential Workflow**

- Issue Credential
  - University issues transcript credential to Alice.

-

- Verifiable Presentation
  - Alice sends Verifiable Presentation to Company.

- Validate Presentation
  - Company validates Alice's claims from the presentation.

# Indy Demo Code Snippet

```
import asyncio
import json
import time

from indy import pool, wallet, did, ledger, anoncreds, blob_storage
from indy.error import ErrorCode, IndyError
from indy.pairwise import get_pairwise

async def run():

    pool_ = {
    'name': 'pool1'
    }
    print("Open Pool Ledger: {}".format(pool_['name']))
    pool_['genesis_txn_path'] = "pool1.txn"
    pool_['config'] = json.dumps({"genesis_txn": str(pool_['genesis_txn_path'])})

    print(pool_)

    # Set protocol version 2 to work with Indy Node 1.4
    await pool.set_protocol_version(2)

    try:
    await pool.create_pool_ledger_config(pool_['name'], pool_['config'])
    except IndyError as ex:
    if ex.error_code == ErrorCode.PoolLedgerConfigAlreadyExistsError:
    pass
    pool_['handle'] = await pool.open_pool_ledger(pool_['name'], None)

    print(pool_['handle'])
    #                   ------------------------------------------------------------------------
    #  Accessing a steward.

    steward = {
    'name': "Sovrin Steward"
```

# References


- Indy Walkthrough:
  https://github.com/hyperledger/indy-sdk/blob/master/docs/getting-started/indy-walkthrough.md

- Indy Walkthrough Python Code:
  https://github.com/hyperledger/indy-sdk/blob/master/samples/python/src/getting_started.py

  Sample code in other languages:
  https://github.com/hyperledger/indy-sdk/tree/master/samples

- Indy-node: https://github.com/hyperledger/indy-node
-
- Indy-sdk: https://github.com/hyperledger/indy-sdk

- Indy-plenum: https://github.com/hyperledger/indy-plenum


*Most of the explanation and codes are taken from previous NPTEL Lectures and slides. Please refer to them for further details.