**Test Xode V1.0.0 for Runtime Upgrade**

(Adding Collators by Session Keys and Collator Selection)

**Source:**

https://docs.substrate.io/tutorials/build-a-parachain/prepare-a-local-relay-chain/

https://wiki.polkadot.network/docs/maintain-guides-how-to-validate-polkadot

https://doc.deepernetwork.org/tutorials/v3/cumulus/start-relay/#pre-configured-chain-spec-files

**Step 1: Prepare a local relay chain**

1.1 Build the relay chain node. Go to the terminal and follow this command

git clone https://github.com/paritytech/polkadot-sdk.git

cd polkadot-sdk

cargo build –release –package polkadot

Also build the subkey for generating node keys

cargo build –release –package subkey

1.2 Download the raw chainspec of Paseo

wget -O paseo-local.raw.json https://raw.githubusercontent.com/paseo-network/runtimes/main/chain-specs/paseo-local.raw.json

1.3 Start the Relay chain node

First, generate the node key for Alice.

./target/release/subkey generate-node-key --file alice\_secret\_ed25519

After that, run the first validator

./target/release/polkadot \

--alice \

--validator \

--node-key-file alice\_secret\_ed25519 \ //paste the generated node key here

--base-path /tmp/relay/alice \

--chain paseo-local.raw.json \

--port 30333 \

--rpc-port 9944

Open a new terminal and start the second validator using the bob account.

The command like the command used to start the first node with a few important differences.

Generate node key for Bob

./target/release/subkey generate-node-key –file bob\_secret\_ed25519

After that, run the second validator

./target/release/polkadot \

--bob \

--validator \

--node-key-file bob\_secret\_ed25519 \ //paste the generated node key here

--base-path /tmp/relay/bob \

--chain paseo-local.raw.json \

--port 30334 \

--rpc-port 9945

1.4 Reserve a unique identifier

* Open the Polkadot/Substrate Portal in a browser.
* Go to where the relay chain is running.
* Click Network and select Parachains.
* Click **Parathreads**, then click **ParaId**.

Click Submit

Click **Sign and Submit** to authorize the transaction.

After you submit the transaction, click **Network** and select Explorer.

Check the list of recent events for successful registrar.Reserved and click the event to see details about the transaction.

**Step 2: Connect a Local Parachain**

2.1 Obtain the xode-node binary v0.1.0. Open your terminal and follow this command:

**curl -L** "https://drive.usercontent.google.com/download?id=10zStcLL08V3hiCy507CBXMCKCb2VFQsM&confirm=xxx" -o xode-node

**After you obtain it, make it executable.**

chmod +x xode-node

2.2 Modify the default chain specification

* Go to the directory where you obtain the xode-node v0.1.0. Generate the plain text chain specification for the xode node by running the following command:

./xode-node build-spec --disable-default-bootnode > plain-parachain-chainspec.json

* Open the plain text chain specification for the parachain template node in a text editor.

sudo nano plain-parachain-chainspec.json

* Set the para\_id and the parachainId to the parachain identifier that you previously reserved.
* For example, if your reserved identifier is 2000, set the para\_id field and parachainId to 2000:

...

"relay\_chain": "paseo-local",

"para\_id": 2000,

"codeSubstitutes": {},

"genesis": {

...

}

...

"parachainSystem": null,

"parachainInfo": {

"parachainId": 2000

},

...

Save your changes and close the plain text chain specification file.

2.4 Generate a raw chain specification file from the modified chain specification file by running the following command:

./xode-node build-spec --chain plain-parachain-chainspec.json --disable-default-bootnode --raw > raw-parachain-chainspec.json

**Example Output:**

2022-08-30 13:00:50 Building chain spec

2022-08-30 13:00:50 assembling new collators for new session 0 at #0

2022-08-30 13:00:50 assembling new collators for new session 1 at #0

2.5 Prepare the parachain collator

* Export the WebAssembly runtime for the parachain.

./xode-node export-genesis-wasm --chain raw-parachain-chainspec.json para-2000-wasm

* Generate a parachain genesis state.

./xode-node export-genesis-state --chain raw-parachain-chainspec.json para-2000-genesis-state

* Start 2 collator nodes:

First, we must generate node keys for the collators.

Go back to where the polkadot-sdk directory is and follow this command.

**Source:** https://github.com/r0gue-io/pop-docs/blob/main/pop-cli-for-appchains/guides/running-on-paseo-locally.md

**For the 1st Collator (Alice)**

./target/release/subkey generate-node-key --file /home/pc/xode-blockchain/alice\_secret\_ed25519 –chain /home/pc/xode-blockchain/raw-parachain-chainspec.json // --file <where you want to store the key file> --chain <where the chainspec of the parachain is located>

**For the 2nd Collator (Bob)**

./target/release/subkey generate-node-key --file /home/pc/xode-blockchain/bob\_secret\_ed25519 –chain /home/pc/xode-blockchain/raw-parachain-chainspec.json // --file <where you want to store the key file> --chain <where the chainspec of the parachain is located>

**For the 3rd Collator (Charlie)**

./target/release/subkey generate-node-key --file /home/pc/xode-blockchain-v1/charlie\_secret\_ed25519 –chain /home/pc/xode-blockchain/raw-parachain-chainspec.json // --file <where you want to store the key file> --chain <where the chainspec of the parachain is located>

* Now Run the 1st Collator.

Go back to the directory where the xode-node v0.1.0 is stored.

Now run this command:

./xode-node \

--alice \

--collator \

--node-key-file alice\_secret\_ed25519 \ //put the generated node key here for alice

--force-authoring \

--chain raw-parachain-chainspec.json \

--base-path /tmp/parachain/alice \

--port 40333 \

--rpc-port 8844 \

-- \

--execution wasm \

--chain /home/pc/polkadot-sdk/paseo.local.raw.json \ //path where the chainspec of paseo is stored

--port 30343 \

--rpc-port 9977

* 2nd Collator.

Open another terminal for 2nd Collator.

Now run this command:

./xode-node \

--bob \

--collator \

--node- --node-key-file bob\_secret\_ed25519 \ //put the generated node key here for bob

--force-authoring \

--chain raw-parachain-chainspec.json \

--base-path /tmp/parachain/bob \

--port 40334 \

--rpc-port 8845 \

-- \

--execution wasm \

--chain /home/pc/polkadot-sdk/paseo.local.raw.json \ //path where the chainspec of paseo is stored

--port 30344 \

--rpc-port 9978

**Step 3: Register with the local relay chain**

To register the parachain:

3.1 Open the Polkadot/Substrate Portal in a browser. Go to the relay chain.

3.2 Click **Developer** and select **Sudo**.

3.3 Select **paraSudoWrapper**, then select **sudoScheduleParaInitialize(id, genesis)** to initialize the reserved paraID at the start of the next relay chain session.

For the transaction parameters:

1. **id:** Type your reserved ParaId. For this tutorial, the reserved identifier is 2000.
2. **genesisHead:** Click **file upload** and upload the genesis state you exported for the parachain. For this tutorial, select the para-2000-genesis file.
3. **validationCode:** Click **file upload** and upload the WebAssembly runtime you exported for the parachain For this tutorial, select the para-2000-wasm file.
4. **paraKind:** Select **Yes**.

3.4 Click **Submit Sudo**. Review the transaction details, then click **Sign and Submit** to authorize the transaction.

3.5 After you submit the transaction, click **Network** and select **Explorer**. Check the list of recent events for successful sudo.Sudid and paras.PvfCheckAccepted and click the event to see details about the transaction.

Go to the PolkadotJS UI where the xode v0.1.0 is running. Wait for the blocks to appear.

**Step 4. Run the xode-node v1.0.0 (3rd Collator)**

**Source:** https://hackmd.io/@s\_iGZLIITG6WjSgnFX0pcg/the-collator-setup-guide?utm\_source=preview-mode&utm\_medium=rec

https://docs.litentry.com/parachain/get-started/litentry-network/collator

4.1 Build the xode-node v1.0.0. Go to the terminal and follow this command

git clone git@github.com:Xode-DAO/xode-blockchain-v1.git

cd xode-blockchain-v1

cargo build –release –package xode-node

4.2 Copy the raw-parachain-chainspec.json from the other collator and paste in this directory.

mv /home/pc/xode-blockchain/raw-parachain-chainspec.json .

4.3 Start the xode-node v1.0.0 with this command:

./target/release/xode-node \

--charlie \

--collator \

--force-authoring \

--node-key-file charlie\_secret\_ed25519 \ //put the generated node key here for charlie

--chain raw-parachain-chainspec.json \

--base-path /tmp/parachain/charlie \

--port 40335 \

--rpc-port 8846 \

-- \

--execution wasm \

--chain /home/pc/polkadot-sdk/paseo.local.raw.json \ //path where the chainspec of paseo is stored

--port 30345 \

--rpc-port 9979

4.4 Generate Session Key.

The session key needs to be set for a collator to start producing blocks. Generate a session key by sending an RPC call to the http endpoint of the parachain with the author\_rotateKeys method: Go to a new terminal and follow this:

curl http://127.0.0.1:8846 -H \

"Content-Type:application/json;charset=utf-8" -d \

'{

"jsonrpc":"2.0",

"id":1,

"method":"author\_rotateKeys",

"params": []

}'

An exemplary result: {"jsonrpc":"2.0","result":"0x56066a71efc51e4a6f0f838cac959a08b238e22d478bd5dc0cdc2ac5b40d2e66","id":1}

This command will insert a key in /data/chains/local-testnet/keystore (the private one) and respond to you with a public key to associate your account.

Copy the result because you will need to paste that in the session keys.

4.5 Bind your collator account to the generated session key.

Go to the PolkadotJS UI where the new Collator is running. Go to **Developers >Extrinsics > session > setKey** with your associated account. There you will put the pubkey you just obtained, as “proof” you’ll simply put “0x” (0x is prefix for hex content and there is no validation implemented for the proof field, so we simply send empty content)

Click Submit transaction, then Sign and Submit.

To verify that that association is ok, do this: Go to **Developers > chain state > session > nextKeys. Push the + on the right and you must discover your key in the field at the bottom right of the page**

**5.6 Invulnerables & Candidates**

As for cumulus' out-of-the-box implementation, the set of collators that are allowed to build blocks are coming from the pallet collator-selection, more specifically the invulnerables and candidates.

If we want to add a new collator we either add it to invulnerables through root or we can register as a candidate.

In this, I used Invulnerable. Go to **Developers > Sudo> collatorSelection > setInvulnerable(new). Add the Collators.**

After that, Submit Sudo, then Signa and Submit.

Go back to Network > Explorer and check the recent events. Check if it is successful.

Just wait for the new collator to collate in the next session (Maybe about 3,600 blocks).

**Step 5: Runtime Upgrade**

5.1 Go to PolkadotJS UI and go to where the xode-node v0.1.0 is running. Create some transactions. For example, I will transfer some balance.

5.2 Click Submit transaction, then sign and submit and view your recent event in Explorer.

The Balance transfer is successful.

5.3 Now for Runtime Upgrade. To upgrade, upload the WASM file using Sudo. Follow this:

Click Submit Transaction, then sign and submit and then view your recent event in Network > Explorer

**Result:** The xode-node binary v0.1.0 block has halted after the parachainSystem.ValidationFunctionApplied

**Result:** The xode-node binary v1.0.0 has resumed and upgraded. But you need to wait for the second session (which is 3600 blocks) before the new collator can collate.