Sovrin Steward Preparation Guide

Technical Vetting and Set Up

Purpose

The purpose of this document is help you determine if you’re ready to be a Sovrin Steward and, if so, how to set up a validator node on the Sovrin network.

This document is a work-in-progress and will be periodically updated.

If you’ve already been approved by the Sovrin Foundation Board of Trustees to become a Sovrin Steward and simply need to stand up your validator node, please skip to this section: [Performing the Set Up](#_uqphq2k9fmhg).

Table of Contents

[**1. High Level Overview**](#_3dbrl6j1u2hl) **2**

[**2. Adding to the Ecosystem**](#_6m7dtogse8x6) **3**

[**3. Privileges and Responsibilities**](#_uax6hflfs3th) **3**

[**4. Looking Forward: Observer Nodes**](#_etumcguqxacj) **4**

[**5. Hyperledger Indy**](#_gpas0fvfmd0v) **4**

[**6. Required Paperwork**](#_yfp53tdqlhxn) **4**

[**7. Technical Questions**](#_zhsw7q5sv9rm) **4**

[**8. Performing the Set Up**](#_uqphq2k9fmhg) **6**

[8.1. Two Machines](#_m33afyvay3qr) 6

[Ubuntu 16.04 (64-bit)](#_1fjqj5hkbiey) 6

[8.2. Validator Node Machine](#_gjp5a2kbk3c4) 7

[Get the IP Address](#_9pqlbsj57ewy) 7

[Choose Port Numbers](#_cduypi53job) 7

[Choose an Alias](#_gt2lnsg9j5bz) 7

[Note](#_8wjskcoclr9o) 7

[**9. Setup and Configuration**](#_7ln5jfkoaki6) **8**

[9.1. Client Installation](#_2dhehhwi6ub8) 8

[9.1.1. Install the Client](#_yaxbdm5gdb5w) 8

[9.1.2. Generate the Steward Key](#_uorq4gpjyit8) 8

[Generate a Seed](#_hsgdh9ngvuc1) 8

[Option A - Choosing your Own Seed](#_rhkscsejd0f2) 9

[Option B - Let Linux Generate a Strong Random Seed for You (PREFERRED)](#_v114ysmpbysm) 9

[Run the Indy CLI](#_vswq00o97c5k) 9

[Create a New Key From the Seed](#_z1n9frvau8qu) 9

[9.2. Validator Node Installation](#_il4b7xijujxy) 9

[9.2.1. Perform Network Test](#_a2y2mmlyqj2t) 10

[Download and Run the “Listener Script”](#_8ahp7u68lubk) 10

[Run the Network Test](#_z9t5k9cgcqgh) 10

[9.2.2. Install the Validator Node](#_2jxzfb9yqyht) 11

[9.2.3. Create the Key for the Validator Node](#_votw8a9lan5j) 11

[9.3. Run the Security Scans](#_x8bhobvzrf7w) 12

[9.4. Provide Information to Trustee](#_wxj0v2mifgh6) 13

[9.5. Add Node to a Test Pool](#_kd2hsddfkkcs) 13

[9.5.1. Configuration](#_5cm2syi0cvm) 13

[Make Sure Your Version Is Current](#_bewxfh16h7m4) 13

[Add Validator Node to Ledger](#_eg8r1qxkvix7) 14

[9.5.2. Enable the Service](#_jrhulsmrx2me) 15

[9.6. See if the Node Is Working](#_9fp1gdy5j26c) 16

[9.7 Moving to the Provisional (Live) Network](#_gs92cfpxo93r) 17

[9.7.1 Configure the validator](#_225bvrigtosr) 17

[9.7.2 Add node attributes to the live ledger](#_o1bcesrji0ry) 17

[9.7.3 Start the indy-node Service](#_alj5b0zc96ei) 18

[9.8 Verify Operation of the Validator](#_in5ygac0mj5x) 18

# 1. High Level Overview

Sovrin Stewards operate the validator nodes of the Sovrin decentralized identity network. All stewards must be approved by the [Sovrin Foundation Board of Trustees](https://sovrin.org/about/). The qualifications for becoming a steward are described in section 5.2 of the [Sovrin Provisional Trust Framework](https://sovrin.org/trust-framework/). If you believe your organization meets these qualifications and would like to apply to become a Sovrin steward, please send an email describing your organization and your interest in becoming a Sovrin steward to Steve Fulling, Executive Director of the Sovrin Foundation, [steve@sovrin.org](mailto:steve@sovrin.org).

Having a validator node will allow you or your organization to be part of what is called **consensus**. Consensus is a protocol spoken between all of the validator nodes on the network when they come together to agree upon modifying the ledger. In essence, they say, “This is what is currently on the ledger, we agree about the nature and order of these new items that should added, and we agree that this is what the ledger will look like when we’re done.”

By design, as a steward, you are only allowed to operate one validator node. However, you are allowed to operate unlimited trust anchors. To better understand these terms, please refer to Section 3 of the [Sovrin Provisional Trust Framework](https://sovrin.org/trust-framework/).

Once you have the necessary permissions to become a Sovrin Steward, these instructions will show you how to:

* Set up a machine with all of the necessary software
* Create the validator node and start its service
* Configure everything properly

Let’s see if you’re ready to become a Sovrin Steward.

# 2. Adding to the Ecosystem

The Sovrin Foundation is interested in stewards that will enhance the value and robustness of the Sovrin ecosystem. Important characteristics include:

* Diversity in industry, legal jurisdiction, physical location, expertise, and technical stack
* Ability to provide high quality technical services
* Commitment to Sovrin Foundation values and guiding principles
* Financial stability
* Strong relationships with the DLT and blockchain communities, trade organizations, academia, and others in the ecosystem
* Market reputation as a trusted institution

# 3. Privileges and Responsibilities

Being a steward confers both privileges and responsibilities. Anyone can use the Sovrin network without being a steward, but stewards have unique access to their own central node on the ledger, which may improve performance of tools and services they build. Stewards also have a say in how **trust anchors** are created, which in turn simplifies provisioning of identities across the population that a steward serves. In addition, stewards receive acknowledge and gain reputation in the network for their efforts. All of this can furthers a steward’s mission.

Stewards also have a responsibility to be good custodians and champions of the network they support. If their node goes down, it places burden on other nodes; enough down nodes can slow the network or even stop it. Security and privacy in Sovrin is affected by how carefully stewards administer their node.

# 4. Looking Forward: Observer Nodes

Today, stewardship on Sovrin has practical limits; the distributed consensus at the heart of Sovrin slows down as more stewards are added. The first stage of the network, called the **Sovrin Provisional Network**, will be limited to a few dozen steward nodes. A steward that joins the network in this first stage earns the designation of “Founding Steward”.

As Sovrin grows, we plan for hundreds or thousands of nodes to be run by stewards, with some handling writes and many more providing a hot-swappable read cache (like a CDN). These latter nodes are called **observers**. A node’s status as a validator (handling writes) or observer (handling reads) may change over time, either statically or dynamically.

# 5. Hyperledger Indy

All stewards run the same codebase—open source validator node software developed under the [Hyperledger Indy](https://www.hyperledger.org/blog/2017/05/02/hyperledger-welcomes-project-indy) project hosted and managed by the [Linux Foundation](https://www.linuxfoundation.org/). Not only does this enable stewards (and everyone else in the ecosystem) to be confident in the code running the Sovrin network, but in the future it will enable all Sovrin steward nodes (both validators and observers) to monitor each other’s performance and upvote or downvote their peers, dynamically maintaining the health of the network. (Note: this reputation system is not developed yet, and it will not undo Founding Steward status for a steward.)

# 6. Required Paperwork

In the first stage of the Sovrin network, all stewards must execute the [Founding Steward Agreement](http://bit.ly/svrn-fstewardagr) that is an integral part of the [Sovrin Provisional Trust Framework](https://bit.ly/svrn-ptrustfw). A steward must also comply with the technical requirements in section 7.1 of the Trust Framework. Please read the entire document, and particularly that section, before proceeding.

# 7. Technical Questions

The Sovrin Foundation is committed to being open and inclusive in all ways that it can, while still accomplishing its overarching purpose, “identity for all.” Thus, our predisposition is to welcome all steward candidates. However, we need to know a few things to test alignment and fitness first. These questions will help us get to know you, a potential steward, to see if we can find a way to work together. Feel free to ask your own questions of us in return.

1. **What type of technical stack do you propose to provide (for running Indy)?** For example, “We’ll run a dual-core CentOS7 machine with 16GB of RAM in our physical datacenter, on top of ESX” or “We plan to run Windows Server 2016 on Azure”.[[1]](#footnote-0)
2. **Please describe the technical background of at least two people in your org who would be involved in operating your node.** For example, “Sally is an IT guru with 14 years of experience, mainly doing sysadmin work with servers such as Oracle and Apache. She is most comfortable on HP UX, but she also has substantial experience on linux distros. She’s been to AWS bootcamp and is a power user of Chef.”
3. **Please describe any experience you have dealing with software vulnerability reports, CVEs, hacks/intrusions, threat models, or other cybersecurity topics.**
4. **Do you have any team members who are capable of contributing code, or of debugging Indy code in a low-level way?** If so, please describe their technical background, their experience with open source, and their availability.
5. **Do you have any team members who would attend Sovrin working groups, visit Sovrin or Indy chat communities, participate in hackfests or other community conferences, or log (or work on!) Sovrin/Indy tickets in jira?** If so, please describe your intentions.
6. **As a simple test, without searching the web, please have a technical team member describe best practices for securing a web server against hacking.** You need not go into great detail, but please identify at least a handful of critical items you would consider.
7. **What is your ability to respond to a report of a hacked or downed node in the middle of the night?**
8. **What is your escalation path?** Assume your node is damaged by a sudden accident (hard drive failure, earthquake, etc). Who would learn about it first, and how would they be notified? How long would it take to notice? What steps could they take to escalate? How would that escalation happen (phone? Email? In person?) Do they have budget authority to power up another node on short notice? Do they have a backup person in case they are in the hospital or out of cell phone range when this happens? Please provide a paragraph or two of commentary.
9. **Have you read all of the technical requirements in section 7.1 of the Trust Framework? Do you have any concerns about them?**
10. **What type of support (processes, automated mechanisms, information) would you like in running a node?**
11. **What Country/State will your Validator physically be located?**
12. **What legal jurisdiction(s) could subpoena access to your node? Do you have a legal contact who is capable of fielding issues like this?**
13. **Who, besides your own personnel, would have physical access to your node?**
14. **Operating a node is not enormously expensive, but it does involve a modest ongoing cost (on AWS, for example, it might amount to a few hundred dollars/euros a month). How do you feel about this expense?**
15. **What relationships do you have that would make your steward status valuable to the overall network?**
16. **What questions do you have for us?**

Note: A [steward FAQ](https://docs.google.com/document/d/1QSGbkaM88duMqgbBcQWqqPyf5Mm4ykJr0dZtBayTFfU/edit?usp=sharing) is available.

If you need technical support with this guide, or at any future time as a Sovrin steward, you can email [support@sovrin.org](mailto:support@sovrin.org). We also recommend [joining the Sovrin team on Slack](https://sovrin-slack-signup.herokuapp.com/), and subscribing to the **validator-support** chan8. Performing the Set Up

Before you start this process, you’ll need to gather a couple of things and make a few decisions.

## 8.1. Two Machines

You’ll need two machines: one for your validator node and the other for the CLI. They can be actual physical machines, virtual machines, or a combination. The client machine can be turned on and off at your convenience (e.g., it could be a VM on a laptop); only the validator node needs to be public and constantly up.

**Important: for security reasons, you should not use your validator node as a CLI client. If you do, it could expose your steward credentials needlessly.**

#### Ubuntu 16.04 (64-bit)

Both machines need to be running a fresh install of Ubuntu 16.04 (64-bit) as this is the only version we have prebuilt packages for.

## 8.2. Validator Node Machine

#### Get the IP Address

Your validator node machine will be the machine that interacts with the Sovrin network. Your validator node is required to have a static, publicly accessible, world routable IP address. In other words, the IP address does not have to be assigned to the machine your validator node is on, but that machine should be accessible by that IP address.

Obtain an IP address that meets this requirement.

#### Choose Port Numbers

The validator node machine will also be required to have the following:

* **Node Port: TCP** - The validators use this port to communicate with each other.
* **Client Port: TCP** - All other agents use this port to communicate with the validator node to the ledger.

By convention, you’ll need to choose ports between 9700-9799. Ports are not required to be in this range, but this is the current convention.

#### Choose an Alias

The validator node machine will need to have an alias. This will be used later when we create a key for the validator node. It can be any convenient, unique name that you don’t mind the world seeing. It need not reference your company name.

#### Note

As you proceed through these steps, you will be generating data that will be needed later. As you obtain the following, store them for later use:

* Your steward key seed
* Your steward ID
* Your steward verification key (verkey)
* The validator node IP address
* The validator alias
* The validator verkey
* The validator BLS key
* The validator node port
* The validator client port

# 9. Setup and Configuration

You must perform the following instructions in order.

## 9.1. Client Installation

### 9.1.1. Install the Client

On the machine you’ve chosen for the CLI, open a terminal and run the following lines.

In the CLI node:

ubuntu@cli$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 68DB5E88

ubuntu@cli$ sudo apt-get install -y software-properties-common python-software-properties

ubuntu@cli$ sudo add-apt-repository "deb https://repo.sovrin.org/deb xenial stable"

Update it:

ubuntu@cli$ sudo apt update

ubuntu@cli$ sudo apt upgrade

Install the Indy client:

ubuntu@cli$ sudo apt install -y sovrin

### 9.1.2. Generate the Steward Key

Next, we’ll generate a steward key. Keys are secrets, sort of like passwords; however, they are not convenient to type. For this reason, we recommend that you generate a key from a human-friendly “seed”. Knowing and remembering your seed will allow you to regenerate the key on demand. *To keep this secure, you will need to have a* ***very*** *secure seed that is not easy to guess.*

#### Generate a Seed

**WARNING:**

You want to guard your seed well. The seed will be used to generate your public (verification) key as well as your secret private key. It is possible that if your seed falls into the wrong hands, someone could regenerate your private key, and take over your identity on the ledger. Keys can be rotated, which can stop some of the damage, but damage will still have been done. So again, guard your seed well!

##### Option A - Choosing your Own Seed

You may choose your own seed. All seeds must be 32 characters long, with no spaces, and should be very difficult for someone to guess.

If you prefer not to choose your own seed, choose Option B.

##### Option B - Let Linux Generate a Strong Random Seed for You (PREFERRED)

In the terminal run the following:

ubuntu@cli$ sudo apt install pwgen

ubuntu@cli$ pwgen -s 32 1

***In either case, the resulting seed should be saved, but must be protected strongly and guarded very well!***

#### Run the Indy CLI

Run the indy command line client by entering:

ubuntu@cli$ indy

#### Create a New Key From the Seed

In the command line, enter the following with your 32 character string inserted in place of <seed>, without the brackets.

indy> new key with seed <seed>

The result should look something like this:

Key created in wallet Default

DID for key is GWBkCr1MsThC46vjk8eMYi

Verification key is ~Ti6PrzckbD6T5gK1eLZc9G

Current DID set to GWBkCr1MsThC46vjk8eMYi

Save the “DID” and “Verification key” portions of this. They will be used whenever you are prompted to supply your steward verkey and ID.

## 9.2. Validator Node Installation

Go to your validator node machine.

### 9.2.1. Perform Network Test

In your web browser, navigate to this URL: <https://s3.us-east-2.amazonaws.com/evernym-cs/sovrin-ProvisionalNetwork/www/network-test.html>

#### Download and Run the “Listener Script”

Download the “listener script” from this page to your validator machine. You can use the link found on the page or use the curl command below. This script is basically an echo service that will listen on the ports.

ubuntu@validator$ curl https://s3.us-east-2.amazonaws.com/evernym-cs/sovrin-ProvisionalNetwork/www/python\_listener.py -o python\_listener.py

Run the script with your node port number and client port number in place of (node port #) and (client port #):

ubuntu@validator$ chmod 750 python\_listener.py

ubuntu@validator$ ./python\_listener.py (node port #) (client port #)

You should get a message that begins with what you see below. You will notice the entire message is not included here.

Starting client port (port number)

Starting node port (port number)...

…

…

#### Run the Network Test

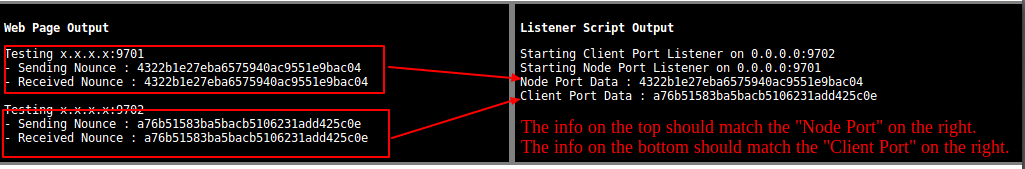
Enter the IP address you initially chose for your validator node machine into the field “IP Address” on the network test URL. Enter each port number you chose for the respective nodes. Click **Test Connectivity**.

The black box to the right of your entered information should show the IP Address, Node Port, and Client Port you entered to the left.

The box on the bottom right should contain information that you’ll see in the command line.

In the box on the bottom left, the “Sending Nonce” and the “Receiving Nonce” should match the “Node Port” on the box on the bottom right.

In the box on the bottom left, the bottom set should match the “Client Port” on the right. See the image for an example.



If these items match, then your network test was successful. Otherwise, this node cannot be used as a Sovrin validator. It will need to be reconfigured until it can pass this test.

### 9.2.2. Install the Validator Node

On the machine you’ve chosen for the validator node, open a Terminal and run the following.

Important: You must use a login user with sudo privileges (**not** root or indy) to run these commands, unless otherwise indicated.

ubuntu@validator$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 68DB5E88

ubuntu@validator$ sudo apt-get install -y software-properties-common

ubuntu@validator$ sudo add-apt-repository "deb https://repo.sovrin.org/deb xenial stable"

Update it:

ubuntu@validator$ sudo apt update

ubuntu@validator$ sudo apt upgrade

Install Indy:

ubuntu@validator$ sudo apt install -y sovrin

Switch to the indy user:

ubuntu@validator$ sudo su - indy

### 9.2.3. Create the Key for the Validator Node

Enter the following where <ALIAS> is the alias you chose for your validator node machine and the <node port #> and <client port #> are your respective port numbers.

indy@validator$ init\_indy\_node <ALIAS> <node port #> <client port#>

You will see something like this (blue highlighting added):

Node-stack name is Node5

Client-stack name is Node5C

Generating keys for random seed b'FA7b1cc42Da11B8F4BC83990cECF63aD'

Public key is e3b5bd1cc81958c5d0fd64c4418b5ff8809e2b116c6de001b7d8e023efbc9437

Verification key is bfede8c4581f03d16eb053450d103477c6e840e5682adc67dc948a177ab8bc9b

BLS Public key is 4kCWXzcEEzdh93rf3zhhDEeybLij7AwcE4NDewTf3LRdn8eoKBwufFcUyyvSJ4GfPpTQLuX6iHjQwnCCQx4sSpfnptCWzvFEdJnhNSt4tJMQ2EzjcL9ewRWi24QxAaCnwbm2BBGJXF7JjqFgMzGfuFXXHhGPX3UtdfAphrojk3A1sgq

**Store the Public Key, the Verification Key and the BLS Public Key.** These are the keys for your validator node (not to be confused with the keys for you in your steward role). The Verification Key and BLS Key for the validator node are public and will be published on the ledger. The random seed should also be saved, but should be protected from disclosure.

You may now exit to your default login user:

indy@validator$ exit

## 9.3. Run the Security Scans

Download [this script](https://raw.githubusercontent.com/sovrin-foundation/steward-tools/master/steward_tech_check.py), upload it to your validator node, and set the execution flag on it:

ubuntu@validator$ curl https://raw.githubusercontent.com/sovrin-foundation/steward-tools/master/steward\_tech\_check.py > steward\_tech\_check.py

ubuntu@validator$ chmod +x steward\_tech\_check.py

Execute it, answering the questions that it asks. There are no wrong answers; please be honest. Questions that can be answered by scripting are automatically completed for you.

ubuntu@validator$ sudo ./steward\_tech\_check.py

After the script completes, copy the output beginning at '== Results for "A Steward MUST" ==', and paste it into an email addressed to support@sovrin.org and susan.bradford@sovrin.org

## 9.4. Provide Information to Trustee

At this point you should have the following data available:

* Your steward verkey and ID
* The validator node IP address
* The validator alias
* The validator verkey
* The validator node port
* The validator client port

Please go to the [Founding Steward Validator Registration](https://docs.google.com/forms/d/e/1FAIpQLSdDqiu2AWpFyF7a0eirwgBKdLXkQfOM0Z8ilJvMOxQTyI7qig/viewform?usp=sf_link) page and provide us with this information (along with your email address, in case there is any follow-up needed).

## 9.5. Add Node to a Test Pool

After your data is submitted via the Founding Steward Validator Registration form, a Sovrin administrator will put your steward credentials into the ledger. You will receive notification when your DID and verification key have been added to the ledger. You should not proceed further with this document until your key is in the ledger.

### 9.5.1. Configuration

After you have been informed that your credentials have been placed onto the ledger of the Sovrin Test Network (STN), you may complete the configuration steps to activate your validator node on that network. Those who are comfortable doing so may do these steps unassisted, or assistance with this process can be requested at [support@sovrin.org](mailto:support@sovrin.org).

#### Make Sure Your Version Is Current

In some cases, some time may have passed before reaching this point. You should ensure that you have the current version of indy software installed before proceeding. On the validator node, execute the following.

In the validator node:

ubuntu@validator$ sudo apt update

ubuntu@validator$ sudo apt upgrade sovrin

On the CLI node, execute the following.

In the CLI node:

ubuntu@cli$ sudo apt update

ubuntu@cli$ sudo apt upgrade sovrin

#### Add Validator Node to Ledger

In order to successfully add the new node to an existing pool, the validator node HEX verification key must be converted to a Base58 form. You’re now going to run what you see below, substituting the HEX version of the node verification key from your output from init\_indy\_node for NODE\_VERKEY. An example, with the NODE\_VERKEY highlighted in blue, will follow to make it more clear.

ubuntu@cli$ python3 -c "from plenum.common.test\_network\_setup import TestNetworkSetup; print(TestNetworkSetup.getNymFromVerkey(str.encode('NODE\_VERKEY')))"

**Example:**

ubuntu@cli$ python3 -c "from plenum.common.test\_network\_setup import TestNetworkSetup; print(TestNetworkSetup.getNymFromVerkey(str.encode('bfede8c4581f03d16eb053450d103477c6e840e5682adc67dc948a177ab8bc9b')))"

When you press Enter​, your Base58 verification key (verkey) will be generated. Below is just an example. We have highlighted it in green so you can see what we’re doing with it in a moment.

4Tn3wZMNCvhSTXPcLinQDnHyj56DTLQtL61ki4jo2Loc

Now return to the CLI prompt on your CLI node by typing indy, and enter the DID that was returned earlier, when you typed ‘new key with seed …’ for your steward user:

ubuntu@cli$ indy

indy> use DID <YourStewardDID>

* *Note: You may need to run "new key with seed <your\_steward\_seed>" instead, if your credentials are not stored in your local wallet.*

Connect the CLI to the Sovrin test network (STN):

indy> connect sandbox

If your client is able to connect to the STN, you will see many lines displayed showing that the client is communicating with the nodes of the STN. You should also see the following highlighted line (in blue, in this example):



If you do not see this, you will need to contact support for remediation before continuing.

If the connection is successful, in the CLI enter the following, substituting the various pieces for the respective correct data. An example will follow to be more clear.

* *Suggestion: Edit this in a text editor first, then copy and paste it onto the Indy CLI. Some editors will insert 'smart quotes' in place of regular ones. This will cause the command to fail.*

indy@sandbox> send NODE dest=<the base58 of the verkey of the node> data={'blskey': '<BLS public key from init\_indy\_node>', 'client\_port': <port at which node listens for clients>, 'client\_ip': '<ip at which node listens for clients>', 'alias': '<some user friendly name for the node>', 'node\_ip': '<ip at which node listens for nodes>', 'node\_port': <port at which node listens for nodes>, 'services': ['VALIDATOR']}

**Example:**

In this example, notice how the green portion is the same green output we received after doing the conversion (above). This is the Base58 key and this is where it goes. Replace it with what you got from the Base58 conversion output.

indy@sandbox> send NODE dest=4Tn3wZMNCvhSTXPcLinQDnHyj56DTLQtL61ki4jo2Loc data={'blskey': '4kCWXzcEEzdh93rf3zhhDEeybLij7AwcE4NDewTf3

LRdn8eoKBwufFcUyyvSJ4GfPpTQLuX6iHjQwnCCQx4sSpfnptCWzvFEdJnhNSt4tJMQ2EzjcL9ewRWi24QxAaCnwbm2BBGJXF7JjqFgMzGfuFXXHhGPX3UtdfAphrojk3A1s

gq', 'client\_port': 9702, 'client\_ip': '10.20.30.205', 'alias': 'Node5', 'node\_ip': '10.20.30.205', 'node\_port': 9701, 'services': [

'VALIDATOR']}

* *Suggestion: Save this command. You will use it again when you later connect your node to the provisional network.*

### 9.5.2. Enable the Service

Return to the validator node machine. If your session is still set to use the indy user, exit to get back to our original sudoer user.

In the validator node:

ubuntu@validator$ exit

Start the validator service:

ubuntu@validator$ sudo systemctl start indy-node

Verify the start:

ubuntu@validator$ systemctl status indy-node.service

Enable the service:

ubuntu@validator$ sudo systemctl enable indy-node.service

## 9.6. See if the Node Is Working

If the setup is successful, your validator node is now connected to the validator pool. You can check that in the log file, seeing if connections to the other validator nodes are reported. Replace <ALIAS> with your node’s alias:

In the validator node:

ubuntu@validator$ v

If your node is configured properly, you should see several nodes being selected as the primary or its backups, as in this example:

ubuntu@validator$ grep 'selected primary' /var/log/indy/sandbox/virginia.log

2017-11-14 20:13:08,670 | DISPLAY | primary\_selector.py ( 291) | \_startSelection | PRIMARY SELECTION: virginia:0 selected primary brazil:0 for instance 0 (view 1)

2017-11-14 20:13:08,671 | DISPLAY | primary\_selector.py ( 291) | \_startSelection | PRIMARY SELECTION: virginia:1 selected primary canada:1 for instance 1 (view 1)

2017-11-14 20:13:08,672 | DISPLAY | primary\_selector.py ( 291) | \_startSelection | PRIMARY SELECTION: virginia:2 selected primary england:2 for instance 2 (view 1)

2017-11-14 20:13:08,672 | DISPLAY | primary\_selector.py ( 291) | \_startSelection | PRIMARY SELECTION: virginia:3 selected primary korea:3 for instance 3 (view 1)

2017-11-14 20:13:08,672 | DISPLAY | primary\_selector.py ( 291) | \_startSelection | PRIMARY SELECTION: virginia:4 selected primary singapore:4 for instance 4 (view 1)

## 9.7 Moving to the Provisional (Live) Network

After a period of testing on the STN to assure the correct operation of the validator, a Sovrin trustee will place your steward credentials onto the ledger of the provisional network. **After you are informed that your credentials are on the ledger,** you may proceed with these final steps.

### 9.7.1 Configure the validator

On your validator node, turn off the indy services

ubuntu@validator$ sudo systemctl stop indy-node indy-node-control

Configure your validator node to connect to the live network instead of to the STN by using this command to change a line in the indy\_config.py file:

ubuntu@validator$ sudo sed -i "s/'sandbox'/'live'/" /etc/indy/indy\_config.py

When you ran init\_indy\_node before, it auto-generated a seed that you were told to save securely. Run the node initialization script again, this time adding the seed that was auto-generated when you ran it before:

ubuntu@validator$ sudo -i -u indy init\_indy\_node <ALIAS> <node port #> <client port#> <seed from previous output of init\_indy\_node>

Finally, make sure that the provisional (live) network genesis files are in place on your validator node. You should see "pool\_transactions\_genesis" and "domain\_transactions\_genesis" files in the /var/lib/indy/live/ directory. If they are not there, you will need to stop here and request them from [support@sovrin.org](mailto:support@sovrin.org) before proceeding.

ubuntu@validator$ sudo su - indy

indy@validator$ ls /var/lib/indy/live/\*genesis

indy@validator$ exit

Desired result:

/var/lib/indy/live/domain\_transactions\_genesis /var/lib/indy/live/pool\_transactions\_genesis

### 9.7.2 Add node attributes to the live ledger

In your CLI node, log into the indy CLI, establish your steward credentials, and connect to the **live** network:

ubuntu@cli$ indy

indy> use DID <yourStewardDID>

* *Note: You may need to run "new key with seed <your\_steward\_seed>" instead, if your credentials are not stored in your local wallet.*

Connect to the live Sovrin network:

indy> connect live

As you did for the test network, put your node attributes onto the ledger of the provisional network. Use the same transaction here that you used there. For example:

indy@live> send NODE dest=4Tn3wZMNCvhSTXPcLinQDnHyj56DTLQtL61ki4jo2Loc data={'blskey': '4kCWXzcEEzdh93rf3zhhDEeybLij7AwcE4NDewTf3

LRdn8eoKBwufFcUyyvSJ4GfPpTQLuX6iHjQwnCCQx4sSpfnptCWzvFEdJnhNSt4tJMQ2EzjcL9ewRWi24QxAaCnwbm2BBGJXF7JjqFgMzGfuFXXHhGPX3UtdfAphrojk3A1s

gq', 'client\_port': 9702, 'client\_ip': '10.20.30.205', 'alias': 'Node5', 'node\_ip': '10.20.30.205', 'node\_port': 9701, 'services': [

'VALIDATOR']}

### 9.7.3 Start the indy-node Service

In your validator node:

ubuntu@validator$ sudo systemctl start indy-node

Verify the start:

ubuntu@validator$ systemctl status indy-node.service

## 9.8 Verify Operation of the Validator

Repeat the steps of section 9.6 *for the live network* to determine that the validator is operating properly:

ubuntu@validator$ sudo grep 'selected primary' /var/log/indy/live/<ALIAS>.log

1. Note: as of Aug 2017, the only officially supported OS is Ubuntu 16.04 (64-bit). Support for Windows is planned and partly implemented, and RedHat/CentOS is similarly in progress. Until these distros are officially supported, stewards on these platforms cannot be onboarded. [↑](#footnote-ref-0)