# Avail Node - Session Keys

## Introduction

BEFORE YOU START This chapter continues from the 'Simple Deployment' page, so be sure to read that one before proceeding with this one. NOTE Before trying anything, make sure that you fully read the chapter first before doing any actual work. With our infrastructure set up and our node running, we are ready to register ourselves as potential future validators. If the need for on-chain registration to become a validator seems foreign to you, then this would be a perfect opportunity to do your own research about how Proof of Stake and Nominated Proof Of Stake works.

In short, to prevent people from running faulty validators and earning tokens, all validators need to bond a certain amount of tokens to register and be eligible for the position of producing blocks. With that mechanism in place, all validators that are in some way faulty will either be deregistered or, in the worst case, slashed.

This explanation only covers the surface of how the whole system works, and before continuing, make sure that you do your own research and learn how the system actually operates. A good place to start is to readPolkadot's explanation(opens in a new tab) on how it works.

## Session Keys

Before we can bond our tokens we need to generate our session keys.

In simpler terms, session keys in Substrate are like special secret codes that uniquely identify who's in charge of creating and validating blocks in a blockchain. Each participant, known as a validator, has their own secret key. They use these keys to sign off on the blocks they create, sort of like a personalized signature. This helps make sure everyone knows who's responsible for what, and it adds a layer of security to the whole blockchain process. So, we can think of session keys as the blockchain's way of keeping track of who did what in a trustworthy and secure manner.

To generate it, we can the following command inside the server that is running our node:

**Bare Metal:**

# Generate session keys

curl

-H

"Content-Type: application/json"

-d

'{"id":1, "jsonrpc":"2.0", "method": "author_rotateKeys", "params":[]}'

http://localhost:9944

# Restart our Node

sudo

systemctl

restart

avail.service

**Docker/ Podman**

# First let's ssh into our container

# Option 1: Docker non-root

sudo

docker

exec

-it

( docker

ps

-lq )

/bin/bash

# Option 2: Podman

podman

exec

-lit

/bin/bash

# Generate the Session keys

curl

-H

"Content-Type: application/json"

-d

'{"id":1, "jsonrpc":"2.0", "method": "author_rotateKeys", "params":[]}'

http://localhost:9944

# Exit and restart containe

exit

# Option 1: Docker non-root

sudo

docker

restart

( docker

ps

-lq )

# Option 2: Podman

podman

restart

-l NOTE If you encounter the following error inside the container:bash: curl: command not found , make sure to first installcurl inside the container and then retry it. For Ubuntu, you can runapt install curl -y for root orsudo apt install curl -y for non-root user to have it installed

### Output

Running the curl command either bare metal or inside a container should yield the same result. The command generated the session keys for us, stored them inside our node-related-data folder, and displayed it for us. This key is unique to your own node, so it's important to make sure that it is stored safely and never shared with anyone.

Example Output

{ "jsonrpc" :

"2.0" , "result" :

"0x0c26b9ba4b8fb5f3051a24663ce491663d8e8e81e8f1450ab23e4edd7d3b4f4c7e5e1047ae1edAb5bde2dad7915629b8aec892c550515ed0b1cb7e056e8a33d16c9c7faba50abd75438d4a2bbdbb6b2c3
, "id" :

1 }

Simple deployment Stake your validator