



# CESS Network

The Decentralized Data Infrastructure

## Episode 5

## Demo: Running a Consensus Node



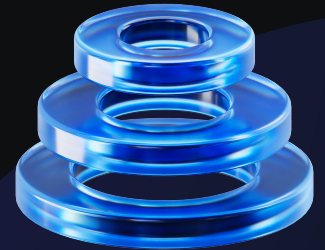
<https://www.cess.network>



# Course Logistics

Course Website: <https://course.cess.network/>

- Episode 1** - . - . - . - . - . - . CESS Network Introduction
- Episode 2** - . - . - . - . - . - . CESS Architecture & Key Technologies
- Episode 3** - . - . - . - . - . - . CESS Ecosystem and Applications
- Episode 4** - . - . - . - . - . - . CESS Nodes & CESS Account Setup
- Episode 5** - . - . - . - . - . - . Demo: Running a Consensus Node
- Episode 6** - . - . - . - . - . - . Demo: Running a Storage Node
- Episode 7** - . - . - . - . - . - . CESS DeOSS and DeOSS REST API
- Episode 8** - . - . - . - . - . - . dApp Development using ink! Smart Contract
- Episode 9** - . - . - . - . - . - . dApp Development using Solidity Smart Contract
- Episode 10** - . - . - . - . - . - . Building Custom Pallet



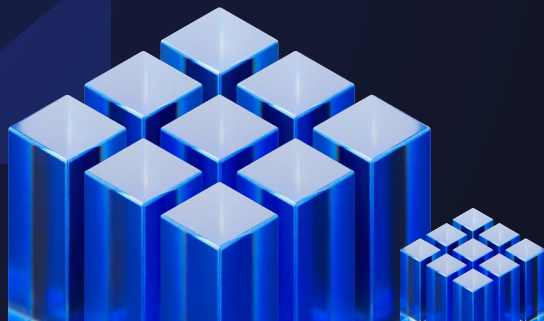
# System Requirements



## Resource

## Specification

- |                                 |                            |
|---------------------------------|----------------------------|
| • Recommended OS                | Ubuntu_x64 20.04 or higher |
| • # CPU Processor               | $\geq 4$                   |
| • Intel SGX Enabled             | Required with FLC          |
| • Memory (SGX encrypted memory) | $\geq 16$ GB               |
| • Bandwidth                     | $\geq 5$ Mbps              |
| • Public Network IP             | Required                   |
| • Linux Kernel Version          | 5.11 or higher             |



# Prerequisites



## Intel SGX with FLC

- Enable from BIOS
- Recommended CPU: Intel E, E3, Celeron
- Preferred CPU: Intel Core i5 10500
- Recommended Motherboard: Supermicro

## Static Public IP

- `curl -4 ifconfig.co`

## CESS Wallet Accounts

- Stash Account: 3Mil TCESS
- Controller Account: 100 TCESS



# CESS Consensus Node Operational Capacity



## 1. Full Node

- Fully-capable consensus node with all necessary functions.
- Generates random challenges, verifies data, computes tags.
- Generates and replaces space holder data.
- Participates in network consensus.
- Requires binding to consensus nodes for registration.

## 2. Verifier Node

- Participates in network consensus.
- Handles random challenges for idle and service data.
- Requires binding to consensus nodes for registration.

## Requirements

Full and Verifier Operational Capacities

### Stash Account

- Keeps all the funds you want to stake.
- Requires at least 3,000,000 TCESS for staking.
- Can be funded by the node owner or delegated by other users.
- Bonds/unbond funds and designates the Controller Account.

### Controller Account

- Pays gas fees for staking-related transactions and registration.
- Takes actions on behalf of the bonded funds in the Stash Account.
- Cannot move bonded funds out of the Stash Account.

# CESS Consensus Node Operational Capacity



## 3. Marker Node

- Computes tags for user's service data.
- Creates, verifies, and replaces idle data segments.
- Serves a designated storage node cluster.
- Can be registered independently.
- Operating as a Marker does not increase reputation points.

## Requirements

### Marker Operational Capacity

- Requires only one account.
- Does not require Binding Funds operation.

**Additional Notes:** Using existing stash accounts with Bounded tokens or using another user's stash account bypasses the need for the Binding Funds operation.

---

# Binding Funds



**Step 1:** Open [CESS Explorer](#) and Select Network > Staking > Accounts > Stash

cess-devnet  
cess-node/102  
#134,734

Accounts ▾ Network ▾ Governance ▾ Developer ▾ Settings ⓘ

GitHub Wiki

CESS Node v0.7.6  
api v10.11.2  
apps v0.133.2-37

Staking Overview Accounts Payouts Targets Bags Slashes Validator stats

✓ All stashes Nominators Validators Inactive

+ Nominator + Validator + Stash

stashes	controller	rewards	bonded
STASH1 (EXTENSION)	STASH1 (EXTENSION)	Staked	3,0000 MTCESS
STASH2 (EXTENSION)	STASH2 (EXTENSION)	Staked	3,0000 MTCESS
6,0000 MTCESS			

Validate Nominate

Validate Nominate

You are connected to the development instance of the UI. For a fully decentralized experience, you are encouraged to use the IPFS deployed version as the canonical URL: [dotapps.io](#)


# Binding Funds



## Step 2:

1. Select appropriate “stash account”
2. Enter at least 3,000,000 TCESS in “value bounded”
3. Select “do not increase the amount at stake” from “payment destination”
4. Click [Bond > Sign and Submit](#) to link the Stash Account.

bonding preferences



stash account

STASH4 (EXTENSION)

cXik7GNf8qY... ▼

value bonded

3000000

balance 5.0000 MTCESS  
TCESS

on-chain bonding duration

28 days

payment destination


Stash account (do not increase the amount at stake) ▼

The stash should be treated as a cold wallet.  
As such it is recommended that you setup a proxy to control operations via the stash.

The amount placed at-stake should not be your full available amount to allow for transaction fees.

Once bonded, it will need to be unlocked/withdrawn and will be locked for at least the bonding duration.

Rewards (once paid) can be deposited to either the stash or controller, with different effects.

 Bond



# Binding Funds Success



cess-devnet  
cess-node/102  
#154,921

Accounts

Network

Governance

Developer

Settings

GitHub

Wiki

Staking

Overview

Accounts

Payouts

Targets

Bags

Slashes

Validator stats

All stashes

Nominators

Validators

Inactive

stashes

controller

rewards

bonded

STASH1 (EXTENSION)

STASH1 (EXTENSION)

Staked

3,0000 MTCESS

Validate

Nominate

STASH2 (EXTENSION)

STASH2 (EXTENSION)

Staked

3,0000 MTCESS

Validate

Nominate

STASH4 (EXTENSION)

STASH4 (EXTENSION)

Stash

3,0000 MTCESS

Session Key

Nominate

9,0000 MTCESS

staking.bond  
inblock

system.ExtrinsicSuccess  
balances.Withdraw  
staking.Bonded  
balances.Locked  
balances.Deposit (x2)  
treasury.Deposit  
transactionPayment.TransactionFeePaid  
extrinsic event

You are connected to the development instance of the UI. For a fully decentralized experience, you are encouraged to use the IPFS deployed version as the canonical URL: dotapps.io

# Consensus Node Installation and Configuration



## Install CESS Client: nodeadm

```
wget https://github.com/CESSProject/cess-nodeadm/archive/refs/tags/v0.5.5.tar.gz
tar -xvf v0.5.5.tar.gz
cd cess-nodeadm-0.5.5
sudo ./install.sh
```

## Client Configuration for a Full node

```
sudo cess config set
```

Enter cess node mode from 'authority/storage/rpcnode' (current: authority, press enter to skip): authority

Enter cess node name (current: cess, press enter to skip):

Enter cess chain ws url (default: ws://cess-chain:9944):

Enter the public port for TEE worker (current: 19999, press enter to skip):

Enter the TEE worker endpoint (current: http://xx.xxx.xx.xx:19999, press enter to skip)

Enter cess validator stash account (current: null, press enter to skip):

Enter what kind of tee worker would you want to be [Full/Verifier]: Full

Enter cess validator controller phrase: xxxxxxxxxxxxxx

# Consensus Node Set Up



## Setting up Consensus Node to Become a Validator

1. Start Consensus Node ---- ccess start
2. Generate a session key ---- ccess tools rotate-keys

```
root@ubuntu:~/cess-nodeadm-0.1.0#  
root@ubuntu:~/cess-nodeadm-0.1.0# curl -H "Content-Type: application/json" -d '{"id":1, "jsonrpc":"2.0", "method": "author_rotatekeys", "params":[]}' http://localhost:9933  
{ "jsonrpc": "2.0", "result": "0x70f0f4ef909c730e0057cfd6d6011f52d1106eaa8ad5ad8a1c20b98f0e568e63a1310b95a210a619b7c5e29a477072c9ad811107eb944f3974070ac0433f1416e6ad41fd9f6208f089b96434062743eae316a664426defed4e256b47b0405d8a064ac3b572e327f77d5a1dfe86f1200c1c19207643f117ebf1a", "id": 1 }
```

3. Setup a Session Key ---- Navigate to [CESS Explorer](#), choose Network > Staking > Accounts > Session Key

The screenshot shows the CESS Explorer interface with the 'Account actions' tab selected. The 'Session Key' button is highlighted with a red box.

stashes		controller	rewards	bonded
TEST-1	TEST-2	Staked	999,0000 TCES	

# Consensus Node Set Up



## 4. Fill in the Session key

set session key

stash account  
CESS 1 (EXTENSION)  
cXfK5CMB8mxtgzDJUA5cok9SCUPjzEPLGFHAdAc...

controller account  
CESS 8 (EXTENSION)  
cXjeHRyckTKxpg5sqZKEcpCNX5THuRQ7g7TYcve...

Keys from rotateKeys  
2804a0387bb25f33c02933ca1d5d5f7010944ee28ec184d4c5a1e46758f8120d0ac96246e287814f

The stash and controller pair. This transaction, setting the session keys, will be sent from the controller.

The hex output from `author_rotateKeys`, as executed on the validator node. The keys will show as pending until applied at the start of a new session.

Set Session Key

## 5. Sign and Submit the transaction

# Consensus Node Set Up



## Becoming a Validator

**Step 1.** Navigate to [CESS Explorer](#), click Network > Staking > Accounts > Validate

The screenshot shows the CESS Explorer interface. The top navigation bar includes links for Accounts, Network, Governance, Developer, and Settings. The 'Network' dropdown menu is open, showing options like Explorer, Staking, Assets, Scheduler, and Event calendar. The 'Staking' option is selected, and the 'Accounts' sub-menu is also open, showing 'Validate' as the active option. The main content area displays a table of stashes with columns for staker, validator, role, and bonded amount. The 'Validate' button for the staker cXf9q2SYcq85... is highlighted with a red box.

stashes	rewards	bonded
SGX-TEST-ACC (EXT-...)	Staked	200,000,000 TCESS
SHUIGE-MINER1 (EXT-...)	Controller	300,000,000 TCESS
cXiHpiCFn6x...	Staked	3,4353 MTCESS
cXf9q2SYcq85...	Staked	3,0000 MTCESS
cXic3WhctsJ...	Staked	3,0000 MTCESS
cXh9naT6CP...	Staked	1,0000 MTCESS
cXh9q6D97j...	Staked	100,000,000 TCESS


# Consensus Node Set Up




## Becoming a Validator

**Step 2.** Enter Reward Commission Percentage as “100” and Select Nominations as “No, block all nominations”

set validator preferences

stash account  
CESS 1 (EXTENSION)  
cXfK5CMB8mxtgzDJUA5cok9SCUPjzEPLGFHAdAc...

controller account  
CESS 8 (EXTENSION)  
cXjeHRycKeTKxpg5sqZKEcpCNX5THuRQ7g7TYcve...

reward commission percentage ?  
100

allows new nominations ?  
No, block all nominations

The stash and controller pair. This transaction, managing preferences, will be sent from the controller.

The commission is deducted from all rewards before the remainder is split with nominators.

The validator can block any new nominations. By default it is set to allow all nominations.

Validate

Click on [Validate](#), and [Sign and Submit](#) the transaction

# Consensus Node Set Up



## Becoming a Validator

**Step 3.** Navigate to [CESS Explorer](#), click Network > Staking > Waiting

The screenshot shows the CESS Explorer interface. The top navigation bar is orange and contains links for Accounts, Network, Governance, Developer, and Settings. The main content area is titled 'Staking' and has sub-tabs for Overview, Accounts, Payouts, Targets, Bags, Slashes, and Validator stats. The 'Overview' tab is active, displaying various staking metrics: 3/3 validators, 1 waiting, 0/0 active/nominators, 50.0% ideal staked, 0.6% staked, and 2.6% inflation. There are also circular progress indicators for epoch (1 hr 7 mins 36 s, 87%) and era (6 hrs 5 mins 7 mins, 14%). Below the metrics, there are buttons for 'Own validators', 'All validators', 'Active', and 'Waiting'. The 'Waiting' button is highlighted with a red box. Below the buttons, there is a filter bar with a search input and a list of nodes. The 'intentions' section is highlighted with a red box, showing a node named 'STASH2 (EXTENSION)' with a 100.00% commission.

validators	waiting	active / nominators	ideal staked	staked	inflation	epoch	era
3 / 3	1	0 / 0	50.0%	0.6%	2.6%	1 hr 7 mins 36 s (87%)	6 hrs 5 mins 7 mins (14%)

Buttons: Own validators, All validators, Active, **Waiting**

Filter: filter by name, address or index

intentions	nominators	commission
★ STASH2 (EXTENSION)		100.00%

Node appear in Candidate Node List

# Redeeming Rewards



Navigate to [CESS Explorer](#): Network > Staking > Payouts > Payout

The screenshot shows the CESS Explorer interface. The top navigation bar includes 'Accounts', 'Network', 'Developer', and 'Settings'. The 'Network' menu is open, showing 'Accounts', 'Address book', and 'Transfer'. The 'Payouts' tab is selected. The 'Own stashes' option is checked under the payout frequency. The 'Payout' button is highlighted in the bottom right corner of the table.

	eras	own	remaining
<b>payout/stash</b>			
CESS 5 (EXTENSION)	50-69, 71-73	902,886.6790 TCESS	15 days 4 hrs
		902,886.6790 TCESS	
<b>payout/validator</b>			
CESS 5 (EXTENSION)	50-69, 71-73	902,886.6790 TCESS	15 days 4 hrs
		902,886.6790 TCESS	

Select Desired Accounts and Click “Payout” then Sign and Submit the Transaction



# Exiting Consensus Node from Validation



## 1. Stop the Consensus

From [CESS Explorer](#), navigate to: Network > Staking > Account Actions > Stop

The screenshot shows the CESS Explorer web interface. At the top, there's a navigation bar with tabs like 'Staking', 'Overview', 'Account actions' (highlighted with a red box), 'Payouts', 'Targets', 'Waiting', 'Slashes', and 'Validator stats'. Below this, there's a sub-navigation bar with buttons for 'All stashes', 'Nominators', 'Validators', and 'Inactive'. The main content area displays a table of stashes. The first row is for 'STASH1' and 'CONTROLLER1'. The 'rewards' column shows '1,000 MTCESS' and the 'bonded' column shows '1,000 MTCESS'. On the right side of the row, there's a 'Stop' button (highlighted with a red box) and a session key '0xe2d0cc...5ef963'. The commission is listed as '100.00%'.

stashes		controller	rewards	bonded	session keys	commission
STASH1	CONTROLLER1	Stash	1,000 MTCESS	1,000 MTCESS	0xe2d0cc...5ef963	100.00%

# Exiting Consensus Node from Validation



## 2. Clear Session Keys

From [CESS Explorer](#), navigate to: Developer > Extrinsics

The screenshot shows the CESS Explorer interface. The top navigation bar has a 'Developer' dropdown highlighted with a red box. Below it, the 'Extrinsics' tab is active. The main content area shows a transaction form. The 'using the selected account' dropdown is set to 'TEST 1'. The 'submit the following extrinsic' dropdown is highlighted with a red box, and the 'purgeKeys()' option is selected and highlighted with a red box. The 'encoded call data' is '0x0b01' and the 'encoded call hash' is '0x486fcd417375f89a45299b604dd8d805e3d2e0c54e44024260b1ebd5acbc346b'. At the bottom, there are buttons for 'Submit Unsigned' and 'Submit Transaction'.

Select Appropriate [Controller Account](#), then Select [session](#) from “submit the following extrinsic” and select `purgeKeys()`

Sign and Submit the transaction

# Redeeming Stake



## Unbound found and Withdraw

From [CESS Explorer](#), navigate to: Network > Staking > Account Actions > Unbond Funds

The screenshot shows the CESS Explorer interface. The top navigation bar includes 'Network', 'Developer', and 'Settings'. The 'Network' dropdown is highlighted with a red box. Below the navigation bar, the 'Account actions' tab is selected and highlighted with a red box. The main content area displays a table of stashes. The first row shows a stash with controller 'CONTROLLER1' and a bonded amount of 1,0000 MTCESS. To the right of the table, a dropdown menu is open, showing options: 'Bond more funds', 'Unbond funds' (highlighted with a red box), 'Withdraw unbonded funds' (highlighted with a red box), 'Change controller account', 'Change reward destination', 'Change session keys', and 'Inject session keys (advanced)'.

stashes	controller	rewards	bonded
STASH1	CONTROLLER1	Stash	1,0000 MTCESS

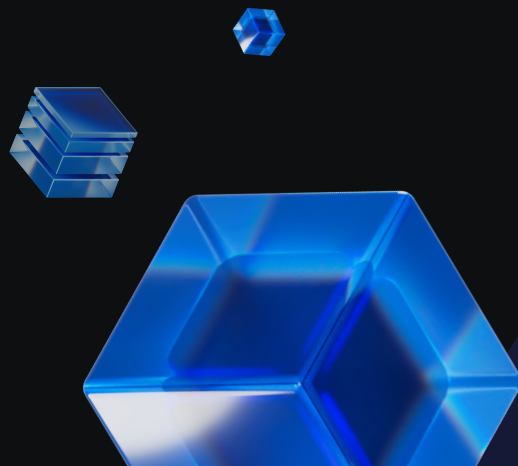
After submitting Unbound funds transaction we can withdraw unbonded funds after 28 eras (Each era is of 6 hours in testnet)

Stop the CESS Client - `cess stop`



# Demo

## Running a Consensus Node





# Thank you for watching

Please Join Our Community





# CESS Network - Episode 5

Demo: Running a Consensus Node



<https://www.cess.network>

