

Zero-fee, p2p computing

A data-backed, app token platform on
top of the Storecoin zero-fee public blockchain



Feb 2019



v0.5

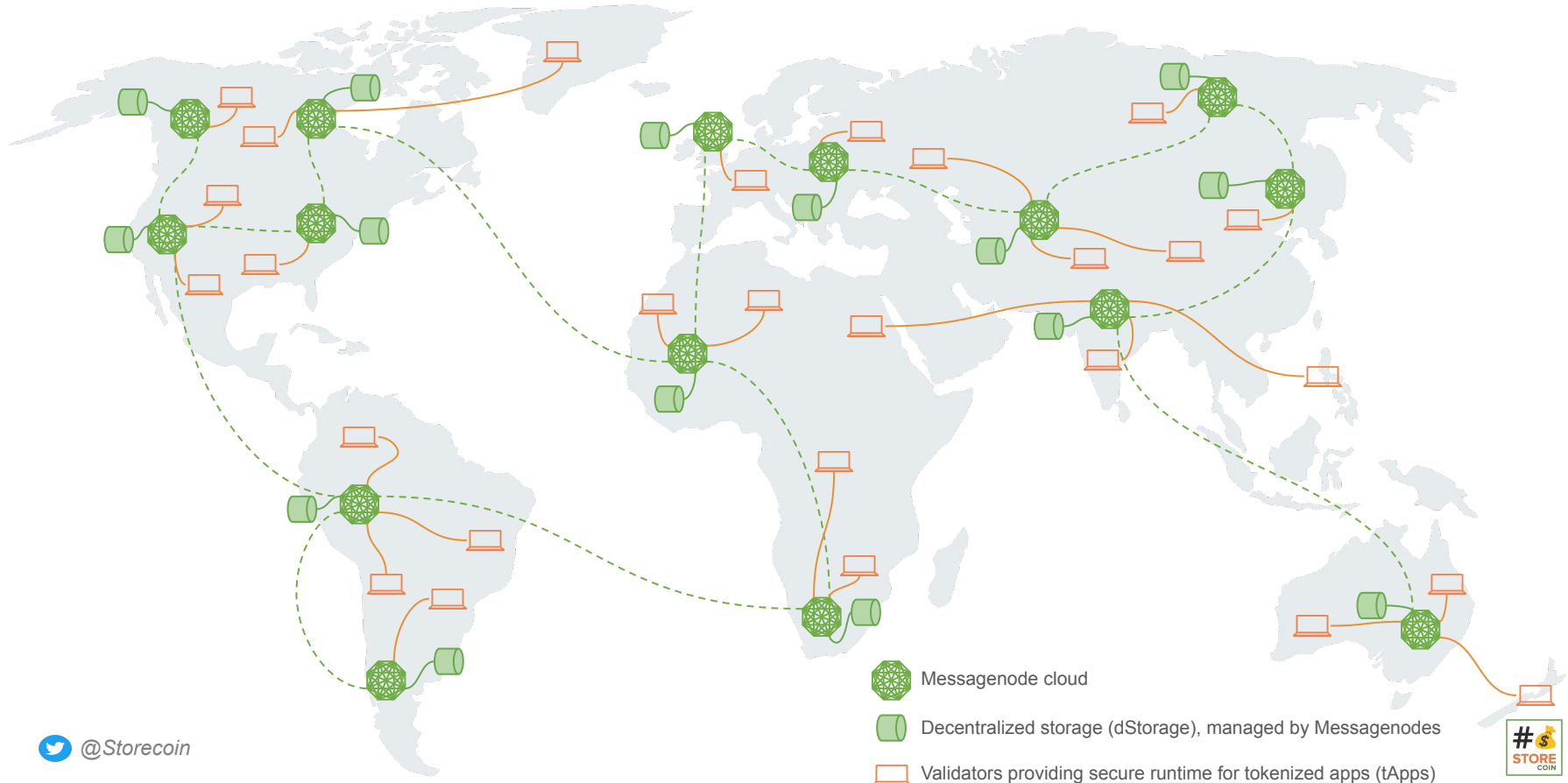
Data as a (zero-fee) tokenized asset is the next logical evolution of "assets" recorded into a p2p ledger. The first asset of course is the currency that incentivizes the miners to invest the resources required to verify the p2p ledger. We think the next asset is data. This is the premise behind the zero-fee, p2p cloud.



Storecoin's one-entity, one-vote governance of checks and balances is the coordination mechanism for the zero-fee payments protocol

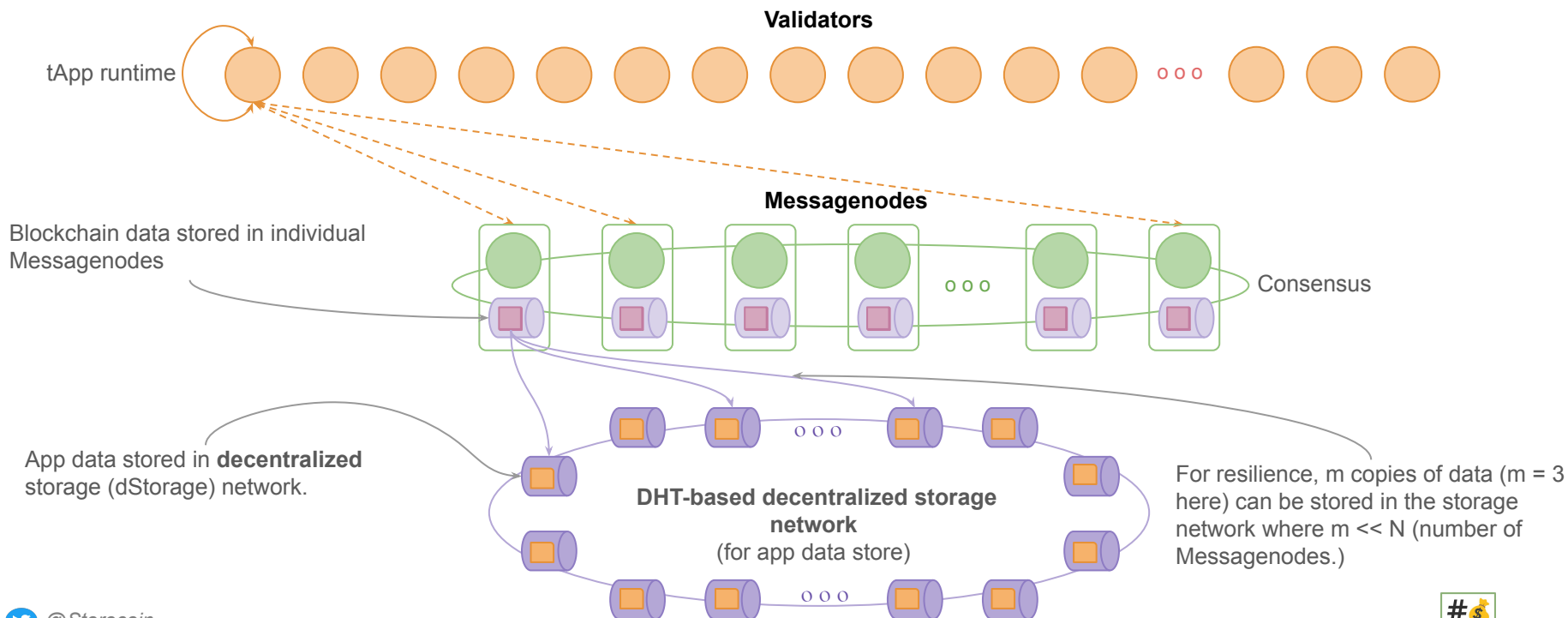


Storecoin tokenized p2p cloud — the vision



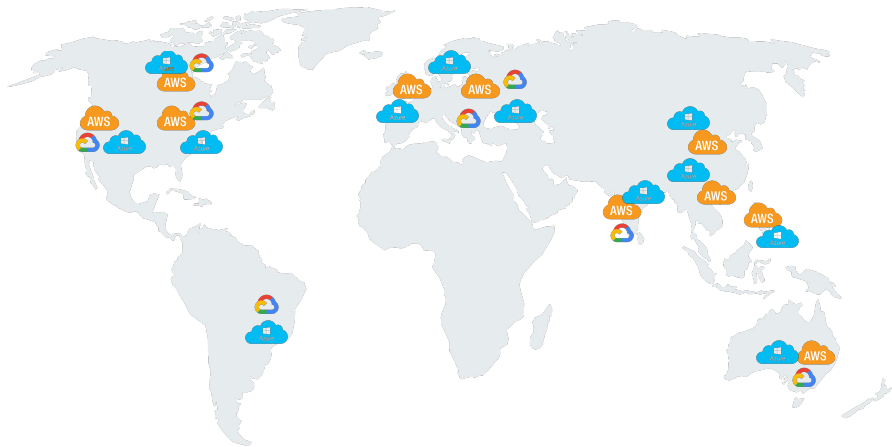
How it works?

Validators provide runtime environment to execute tApps and Messagenodes provide storage to store app data



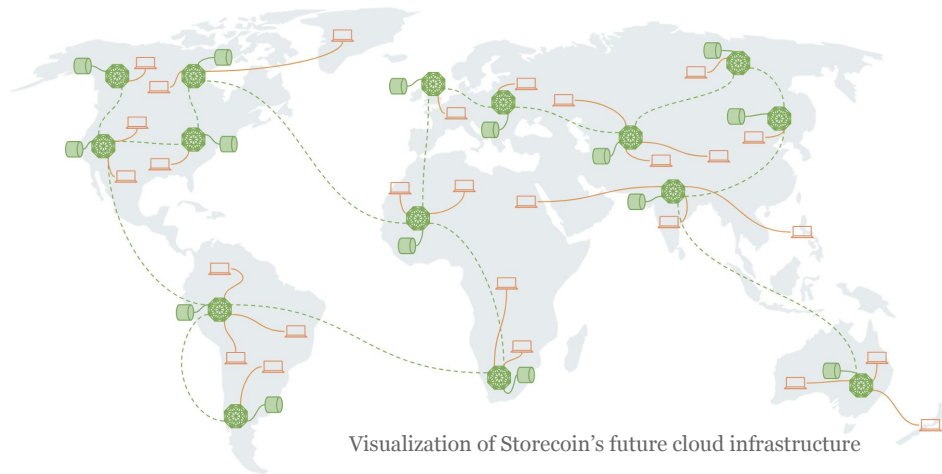
How it works — comparison to centralized apps

Apps hosted on cloud platforms like AWS and Azure today



- App developers host their apps on cloud platforms like AWS, Google cloud, Microsoft Azure, etc. Other centralized services like Facebook have their own distributed datacenters where their apps are hosted.
- Revenue is generated from third parties
- App developers pay the cloud service providers
- The services are typically free for end users
- The data is silo'd to respective apps and their partners
- End users are generally not rewarded for their data

Apps hosted on the Storecoin tokenized cloud platform

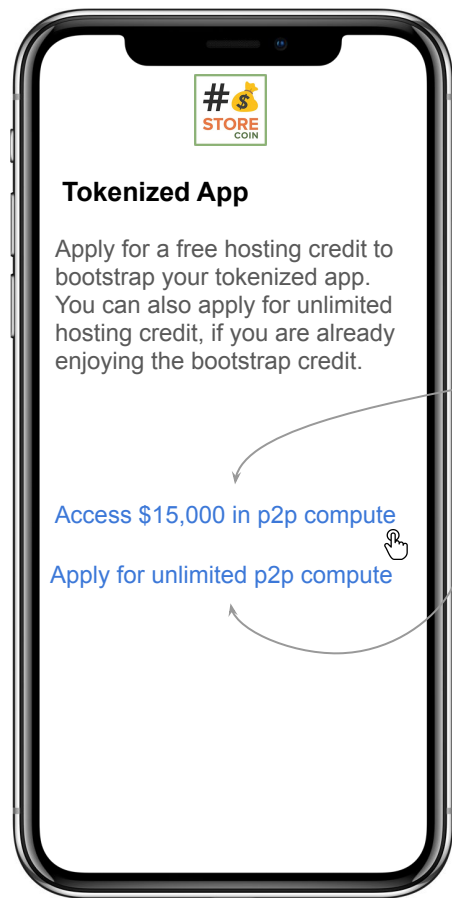


Visualization of Storecoin's future cloud infrastructure

- App developers host their tokenized apps on the Storecoin tokenized cloud, which in turn may use other cloud service providers for storage, computation, and bandwidth needs
- Revenue is generated from third parties
- App developers share their revenue with miners, who are paid with the app tokens
- The services are typically free for end users
- Because of tokenization, the data can be traded among tokenized apps
- App developers, who are refiners of the user data, may choose to reward users for their data

Step 1

Developer — preliminary tApp proposal by its developers



Creates a new preliminary proposal to the miners for early grant of resources.

Creates a full proposal to the miners after the expiry of the initial grant.

In this step, the app developers, who decide to decentralize their app data will apply for a bootstrap credit. The credit allows the app developers start testing, deploying, and running their apps.

The miners will likely use the recommendations of “committees” to decide which of the app proposals will be granted the bootstrap credit. The voting will happen quarterly and all the proposals submitted in that quarter will be voted. The app proposals that are “approved” to receive credits will be granted the promised credit. The apps that are “rejected” cannot use the Storecoin Platform, but they can resubmit their proposal any time in the future for reconsideration.

The app developers will need to wait between the time they submitted their proposal and a decision is made by the miners before they can start using the Platform.

The app proposal and approval flow happens out-of-band, meaning, the proposals and decisions are not recorded in the Storecoin blockchain itself. Once the apps are approved, the resulting app instance data will be recorded in the blockchain.

Step 1

Developer — developers describe app profile

App token (*datacoin*) symbol and initial supply of tokens. Used to determine the pre-mining %.

Storecoin wallet associated with the app. The initial supply of app tokens is deposited to this wallet address.

STORE COIN

Preliminary Proposal

Tokenized App Name

App Token Symbol

Initial Token Supply

Connect to Storecoin Wallet

My wallet 1

My wallet 2

Continue



URL to data schema, which describes the data being decentralized by this app.

This describes how the data is accessed. Is the data publicly and freely accessible, private and never shared, or authorized access to the buyers who pay for the data?

How do the app developers share their app tokens with the users?

STORE COIN

Preliminary Proposal

Data Schema URL

Data Access Type

Authorized

Public

Private

Token Sharing with Users

Never

Fixed payment

Percentage of Revenue

Continue

Step 1

Developer — developers describe app resource requirements

Describe expected resource usage of app runtime, storage, and bandwidth. Miners use this information to plan for the infrastructure needed to support the app.

The usage metrics here are per app instance, meaning, the memory, CPU, storage, and bandwidth required to execute **one instance** of the app.

Since the exact resource usage depends on the logic invoked in the specific app instance, a range is estimated for each resource type.

STORE COIN

Preliminary Proposal

App Resource Use

MB Memory MB

MB Storage MB

MB Bandwidth MB

Cycles Compute

Min Instances/Day

Submit

STORE COIN

Preliminary Proposal

Thanks for submitting the preliminary proposal to <app name>. You will be notified on the status of your application soon.

If <app name> is approved, you will receive the initial credit for \$15,000 worth of compute resources to run your app on the Storecoin Platform.

Done

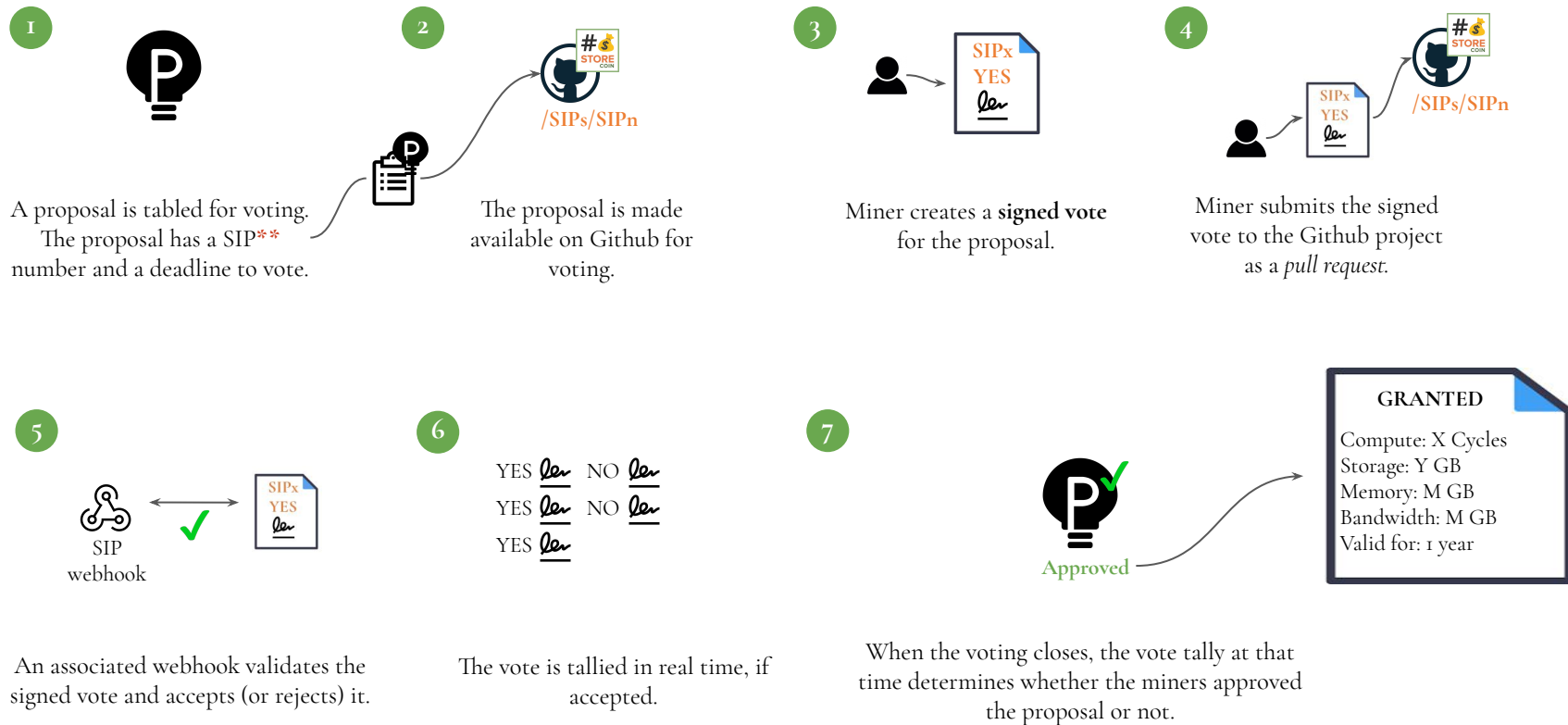
Every app proposal submitted will be reviewed quarterly by the miners. They vote either approving or rejecting the proposals.

The approved app receives the promised credit, which the app developer can use to onboard their applications on Storecoin Platform.

The app developers will wait for the approval before they can start testing/deploying their apps on Storecoin Platform.

Step 2

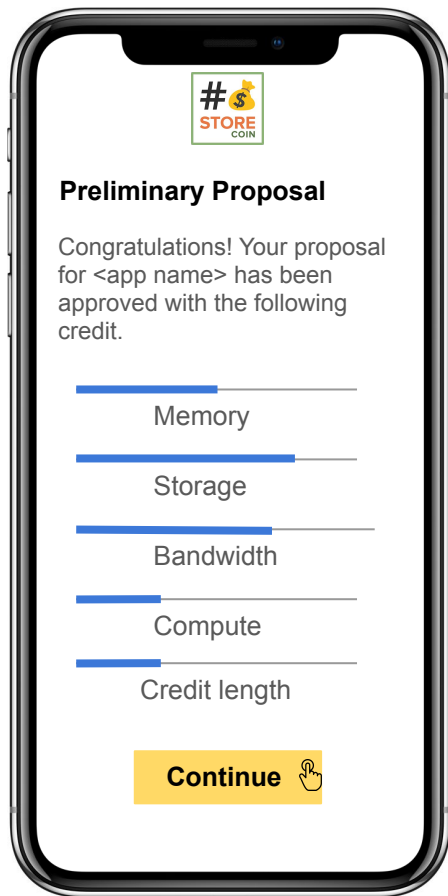
Miners vote* on the tApp proposal for offering the credit



* This flow is reproduced from Storecoin's "OpenVote" proposal.

** SIP = Storecoin Improvement Proposal. Although the app proposals are voted in this flow, the process is standard across as aspects of voting.

Developer — preliminary tApp approval by the miners



After the vote, the app developers are notified on the decision in the app itself. The approved apps will have their initial credits displayed in the app.

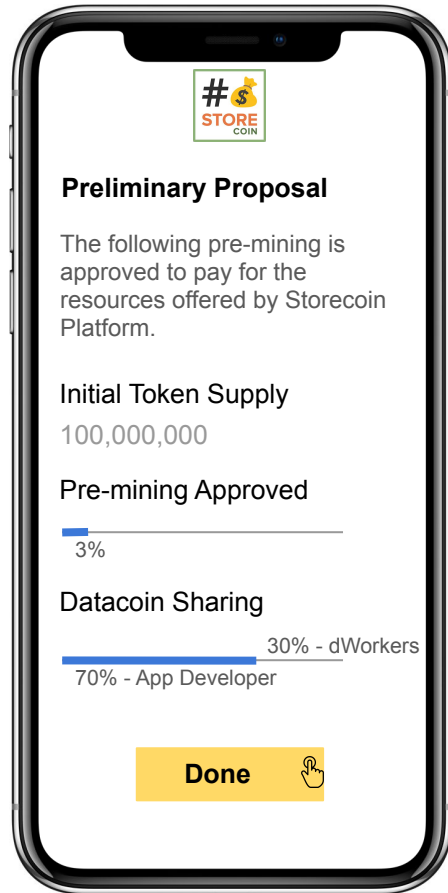
If the miners rejected the app, the app cannot run on Storecoin Platform. However, the app developers can resubmit their proposal at any time in the future for a better chance.

The credit applies to all the app instances combined. The developers can run any number of app instances during the credit period, subjected to the thresholds for the credits granted.

If the miners approve “unlimited” p2p compute for the app, the limits shown here will be at “unlimited” level. There is no specific UI required to show unlimited grant.

Step 3

Developer — pre-mining and revenue sharing decision by the miners



The approval also contains the following:

1/ Pre-mining % of tokens to pay for the services offered by the miners. This % is based on the credit length and the app parameters submitted by the app developer.


2/ The token sharing between the app developers and the miners.

Tokenized app transaction flow — submit a transaction

The user of the tokenized app creates a transaction of the new app type and sends the transaction to any of the Validators.

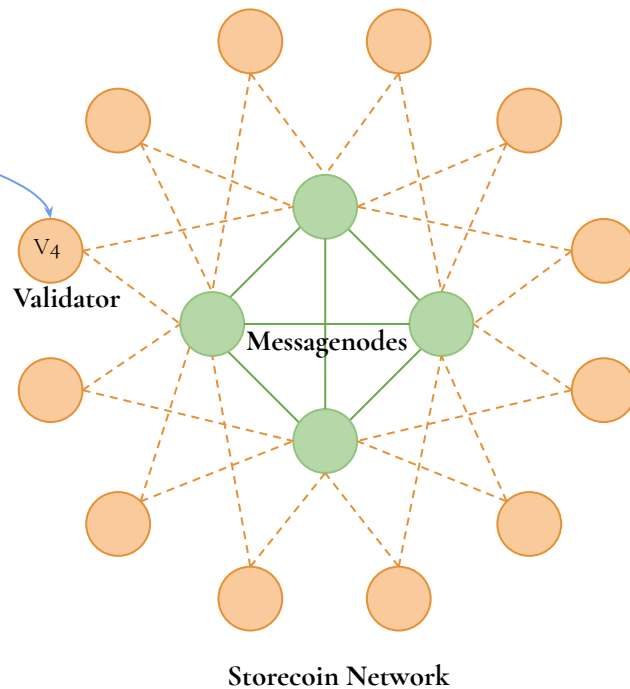
Step 1

App client

Tokenized app transaction 

```
{  
  "type": TXN_TOKENIZED_APP,  
  "nonce": 0,  
  "timestamp": 1542652106301,  
  "appId": "<Unique ID of the app>",  
  "appBundle": "<URL to the app bundle>",  
  . . .  
}
```

"appBundle" points to a URL that contains the Javascript code implementing the app logic

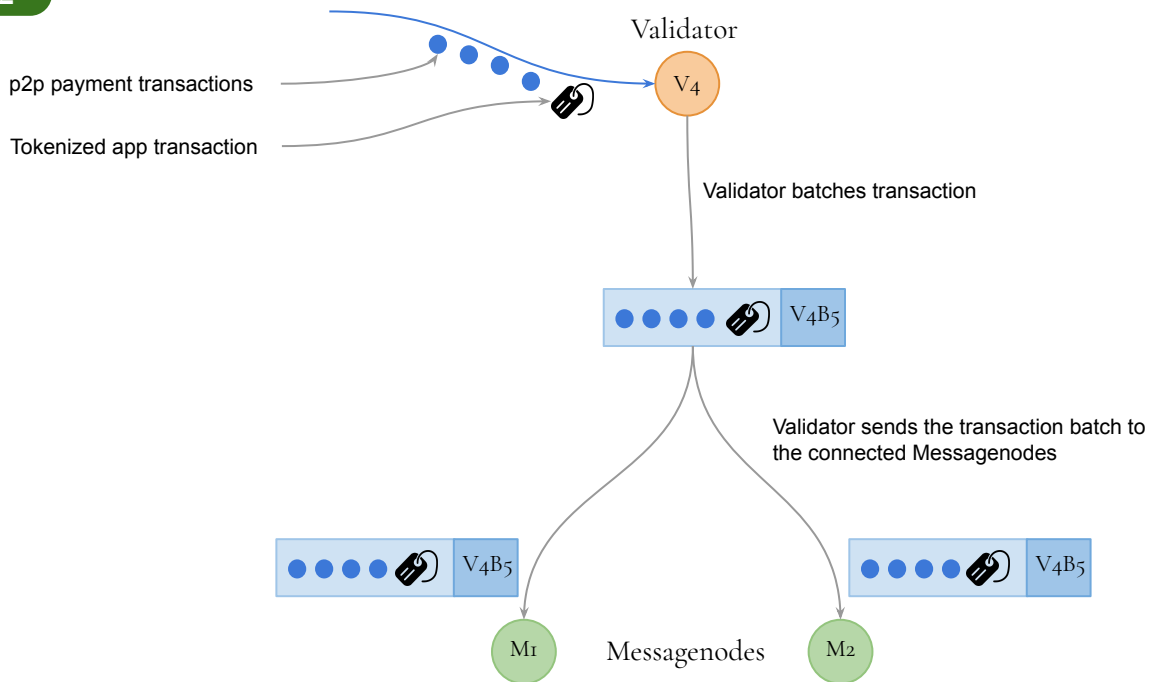


The user creates a transaction of type **TXN_TOKENIZED_APP** and sends the transaction to any Validator node.

Transaction batching* at the validator

The Validators batch incoming transactions as usual. The batch may contain normal p2p settlement transactions sent by other clients.

Step 2

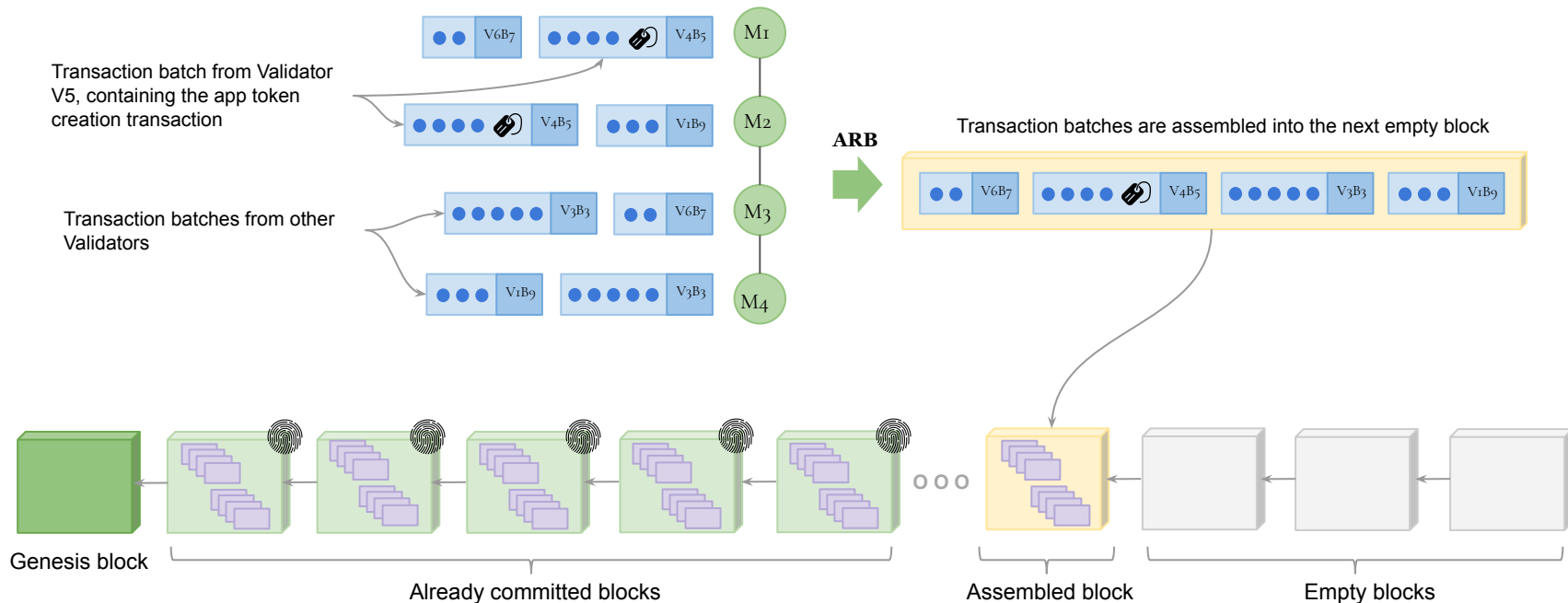


* Steps 2-4 are exactly the same as in the settlement layer. They are repeated here to capture specific nuances of this transaction type.

Block assembly at Messagenodes

Messagenodes use Asynchronous Reliable Broadcast (ARB) algorithm to assemble transaction batches into the next available empty block.

Step 3



Messagenodes notify the connected Validators when the block is assembled and ready for validation.

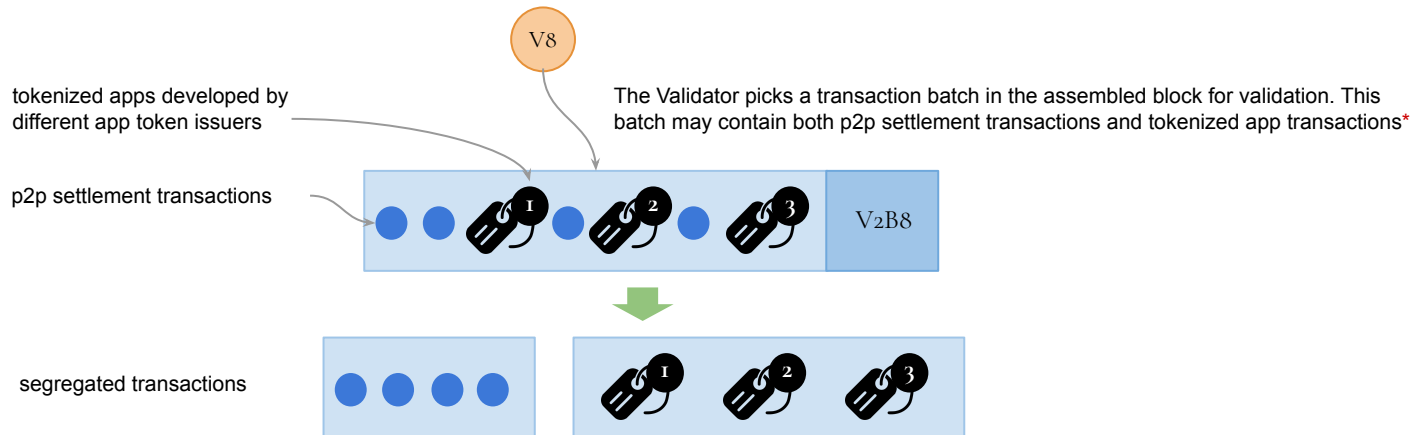
Block validation at Validator nodes

Validating tokenized app transactions is tricky for the following reasons:

- The Storecoin platform institutes a hands-off approach on tokenized app development. It doesn't dictate how the tokenized apps should be architected and implemented, except for defining entry and exit interfaces for the app.
- The hands-off approach allows tokenization of *existing* web-apps without complete rewrite in platform-specific smart contract languages.
- With the hands-off approach however, Validators will not be able to validate the tokenized app transactions because the app logic is opaque to them.
- At the same time, the apps cannot be trusted with the privilege to update the state on the blockchain. This operation must be well under the control of the Validators.
- With these restrictions in mind, the tokenized apps and the Validators collaborate as illustrated below to validate these transactions. The illustration shows the transaction validation flow in one of the Validators (V8 below).

Step 4.a

Validators segregate p2p settlement transactions from tokenized app transactions

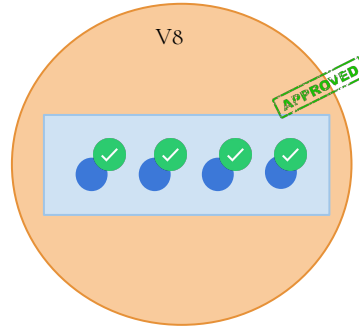


* A more complex scenario is illustrated here. This transaction batch doesn't exist in the assembled block in step 3.

Block validation at Validator nodes

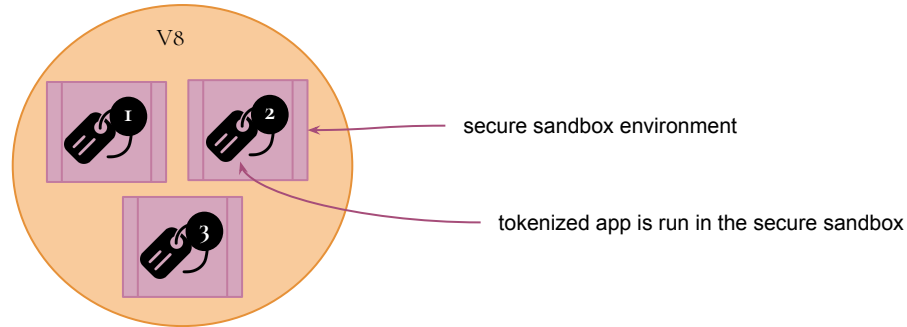
Step 4.b

The Validator validates and approves* p2p settlement transactions directly



Step 4.c

The Validator instantiates secure sandboxes to run the tokenized app transactions in them



* For simplicity all transactions are assumed to be valid. The transactions may get rejected also.

Block validation at Validator nodes

Step 4.c

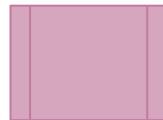
The securebox sandbox instantiation and tokenized app execution flow

Step 4.c.1

Validator creates the secure sandbox instance with the memory limit specified in the usage proposal

Validator context

```
const ivm = require('isolated-vm');  
  
let options = {"memoryLimit": /* memory limit from usageProposal */ };  
  
let sandbox = new ivm.Isolate(options);
```



secure sandbox is created

Step 4.c.2

Validator instantiates the tokenized app instance inside the sandbox. The app logic is opaque to the Validator and hence it subscribes to the result of app execution. The app will notify the Validator with the result when it completes the execution

Validator context

```
const App = require('<appBundle in the transaction>');  
  
// Create app instance with necessary context.  
let context = { /* The context for running the app instance */ };  
let appWrapper = () => {  
  let appInstance = new App(context);  
  // Subscribe to app's result when it is done executing.  
  subscribe('result:'+appInstance.id, resultHandler);  
  appInstance.init();  
}  
// Launch the app instance inside the sandbox.  
let [appContext, script] = await Promise.all([  
  sandbox.createContext(), sandbox.compileScript(`${appWrapper}`)],  
  await script.run(appContext);
```

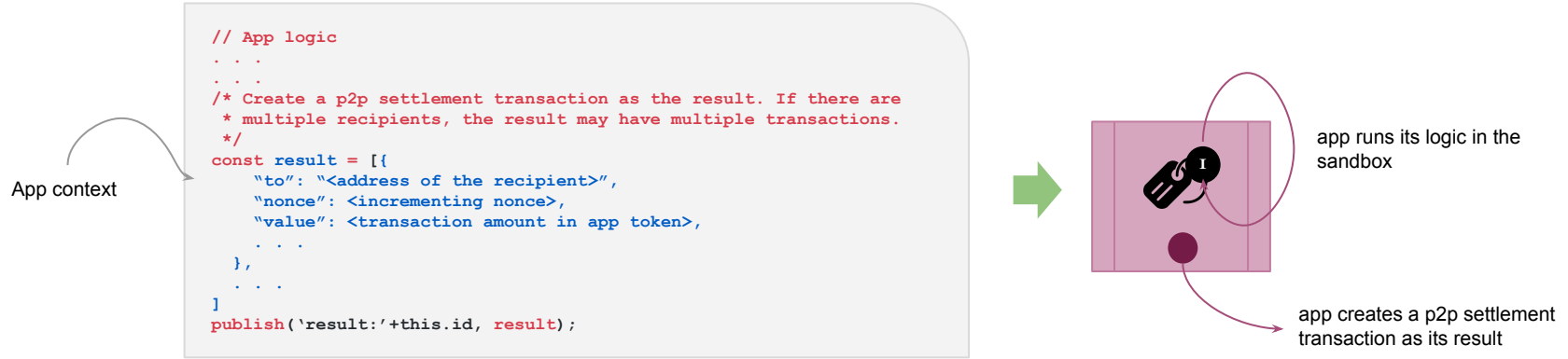


app is instantiated in the secure sandbox

Block validation at Validator nodes

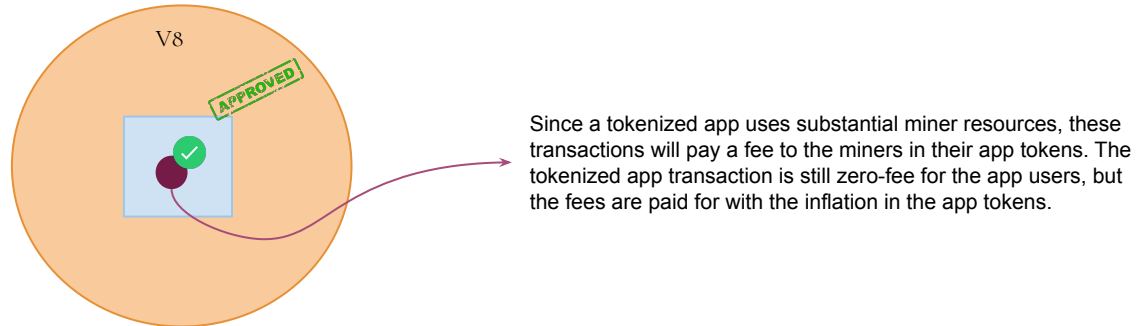
Step 4.c.3

The app creates a p2p settlement transaction as its result when it completes its execution.



Step 4.c.4

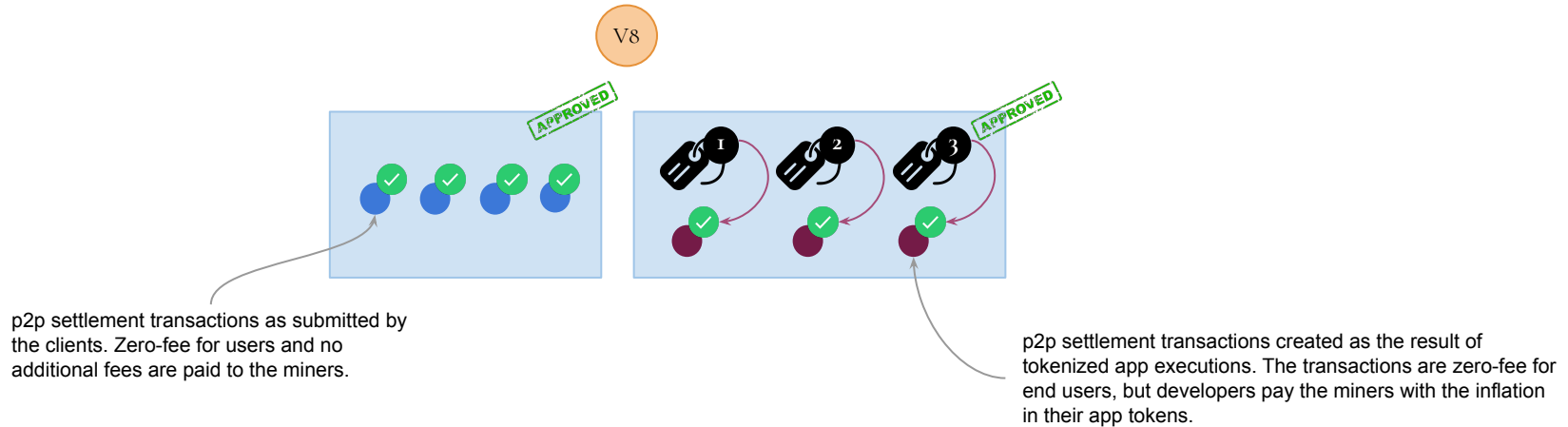
The Validator validates and approves the p2p settlement transaction(s) as described in step 4.b.



Block validation at Validator nodes

Step 4.c.5

Validator is now done with validating and approving both p2p settlement transactions and tokenized app transactions

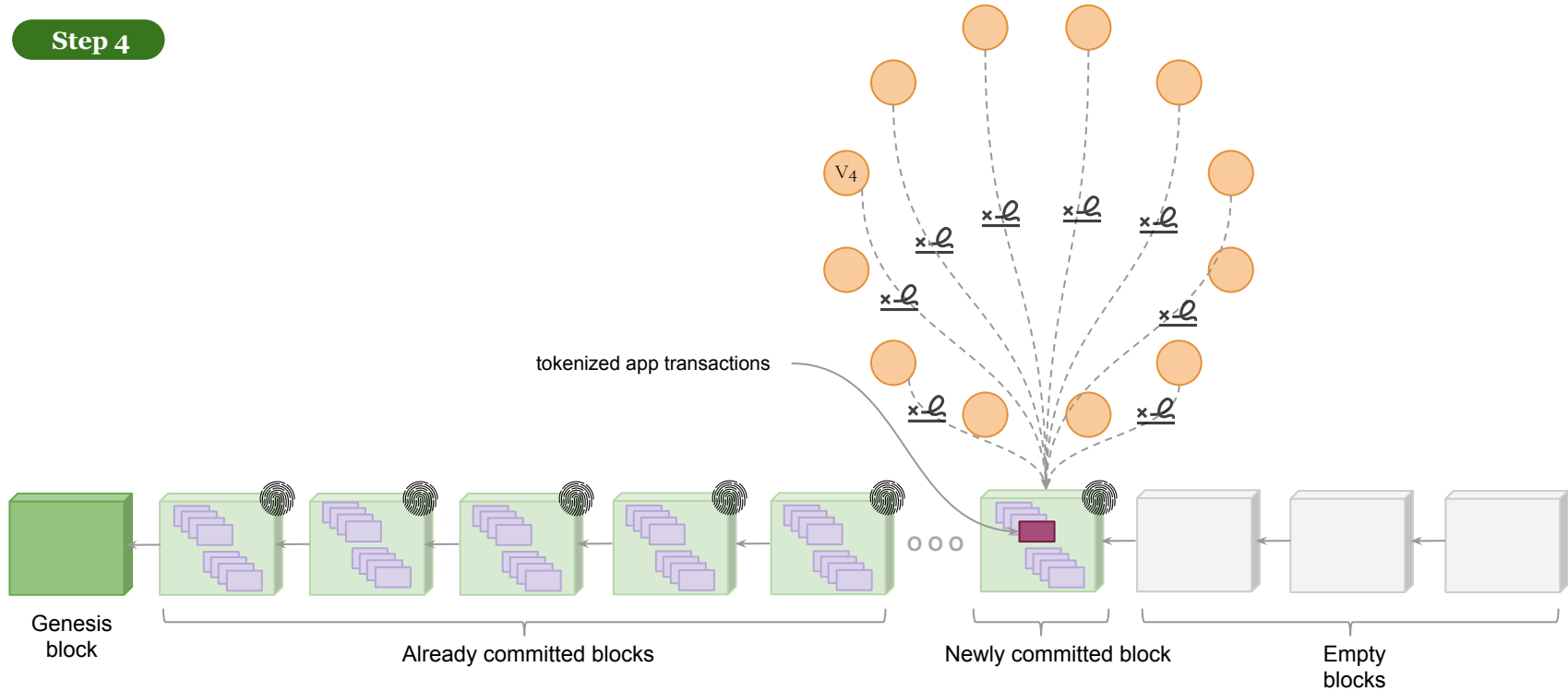


Since Validators implement the settlement logic natively, they are solely responsible for updating the states on the blockchain.

Block validation at Validator nodes

Validators validate the transactions in the assembled block and sign the block with their approval. When more than $\frac{2}{3}$ valid signatures are collected, the assembled block is committed.

Step 4

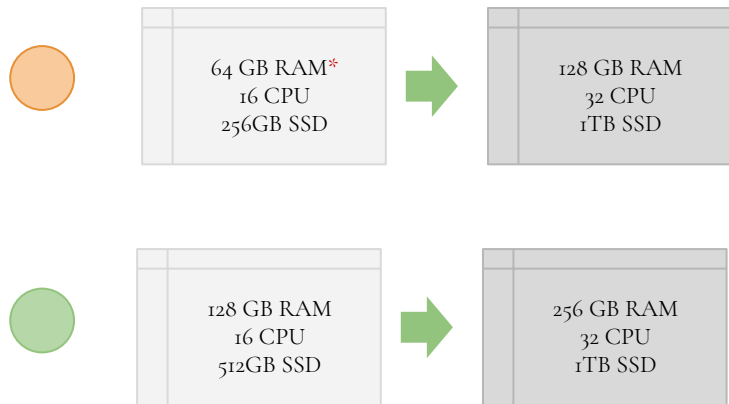


When the block is committed, all the transactions, including the tokenized app transactions, are finalized.

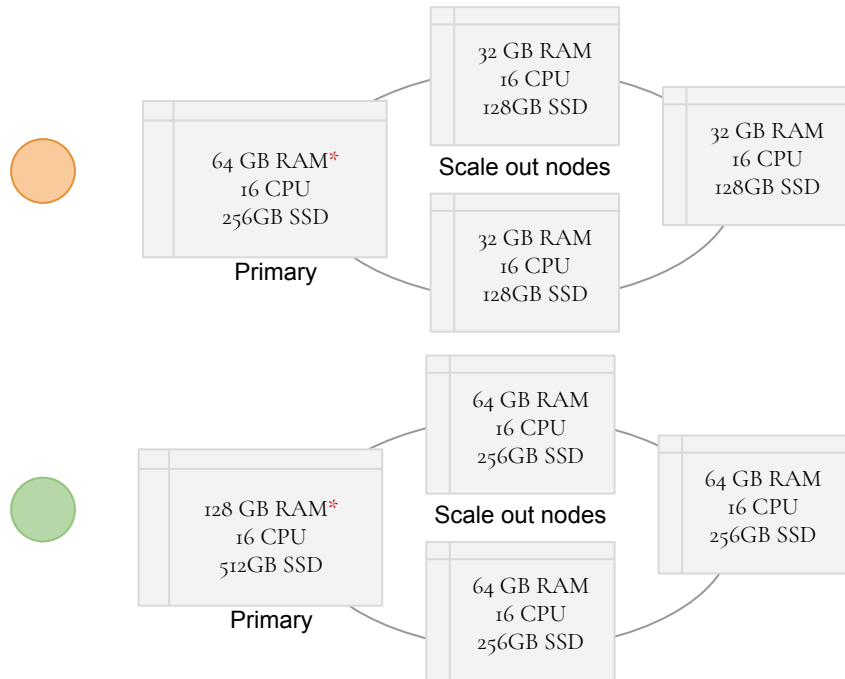
Scaling proposal for Storecoin miners

With the introduction of the tokenized apps, the runtime and storage requirements for the miners become unpredictable. Each tokenized app may have different CPU, memory, bandwidth, and storage requirements and each app may have different usage in terms of number of transactions created, the rate of transactions, etc. So, the following two scaling approaches are proposed.

Scale up



Scale out



* The capacity numbers are for illustration purposes only.