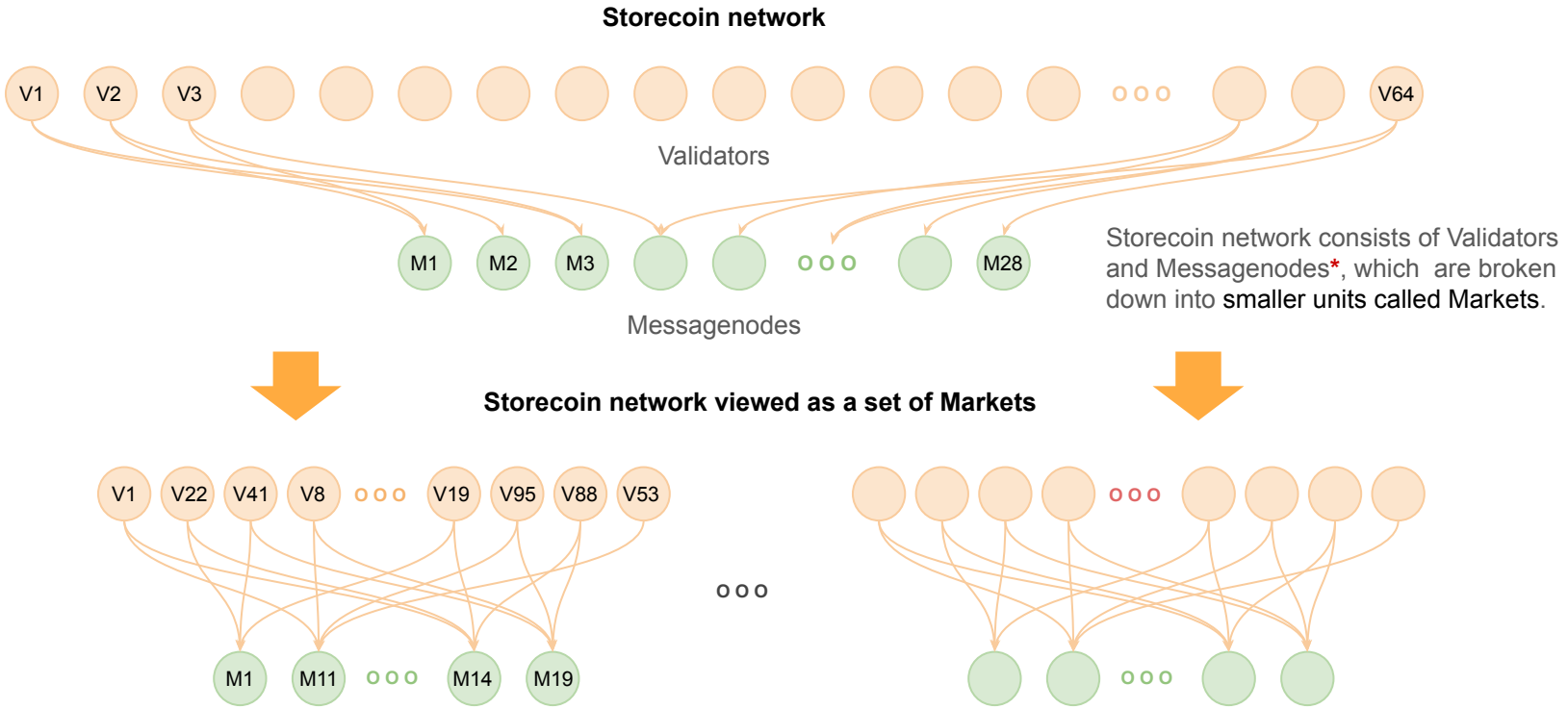
Storecoin ~~subnetworks~~ markets — what are they, why they exist, and how they work

Sept 2019

What is Storecoin market?



@Storecoin

* Storecoin launch phase is assumed in this illustration. The launch phase consists of 64 Validators and 28 Messagenodes, forming 4 “markets”.

** Each market is formed with 16 Validators and 7 Messagenodes. This is the minimum configuration required for a market.

Storecoin market characteristics

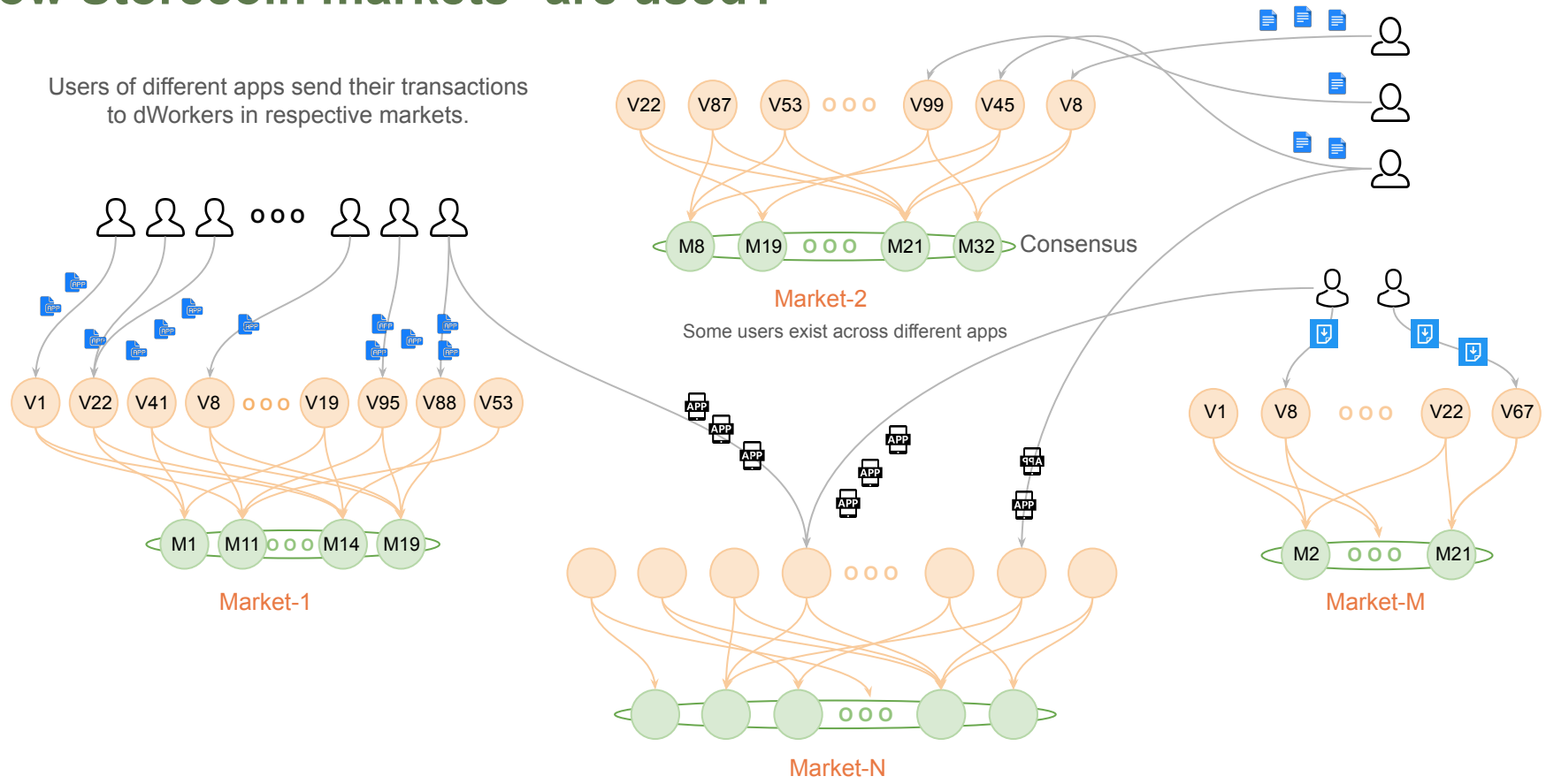
- Markets are *logical* entities
  - Physically, all Validators can connect to all Messagenodes. This is needed for securing settlement transactions.
  - BlockfinBFT consensus algorithm requires that Validators connect to different sets of Messagenodes for each Block to minimize the possibility of long-term collusions.
- Markets are *functional* only when Storecoin Platform launches
  - Markets are formed since the launch phase to minimize governance issues later on when dWorkers ratify governance and vote to support Storecoin Platform.
  - The settlement layer always uses the entire network to build, validate, and finalize blocks. Markets have no special meaning during settlement phases.
  - The security of the Storecoin blockchain is the responsibility of *all* Validators and Messagenodes.
- Each market consists of its own set of Validators and Messagenodes
  - A market has a minimum of 7 Messagenodes and 16 Validators to provide minimum *acceptable* Byzantine fault tolerance.
  - Markets have the same node count in earlier settlement phases, but once ratified, governance can decide the number of markets, minimum node count for each market, and its memberships.
  - A malicious market cannot weaken or defeat the security of Storecoin blockchain because that's the responsibility of the entire network.

Why Storecoin market?

- Markets are used to fulfill specific requirements of tokenized apps
  - Not all tokenized apps are created equal.
  - Apps have different runtime, storage, query performance, and scaling requirements.
  - Data created by the apps may have local or global interest from data buyers, so the cost of hosting different apps of same size will be very different.
- Markets are used to optimize the cost of hosting tokenized apps
  - The cost of hosting a tokenized app is based on the cost of collective resources pooled by dWorkers and the number of apps hosted on the Storecoin Platform.
  - Given different resource and scaling requirements for the tokenized apps, the cost for individual apps cannot be computed fairly. The costs may be disproportionately high for smaller apps.
  - Markets help with fine tuning the costs to the specific needs of an app.
- Markets help with efficiency
  - Unlike the base settlement layer transactions, validating app transactions is expensive, so it is inefficient to use the entire network to validate and finalize app transactions.
  - Markets can do this more efficiently because they are tuned to the needs of the apps.

How Storecoin markets* are used?

Users of different apps send their transactions to dWorkers in respective markets.



@Storecoin

* Storecoin markets are functional only after the platform is launched.

#STORECOIN

How Storecoin markets are used?

- Each tokenized app is hosted by a market
  - Users of the tokenized app send their app transactions to the Validators of the market.
  - Validators in the market provide the runtime environment for executing the app transaction.
  - Messagenodes in the market provide storage for the data created by the app transaction.
  - Messagenodes use BlockfinBFT consensus to ensure that the storage and replication needs of the app are met.
- App transactions are scoped to the market
  - All operations on the app, such as create new data, query, search, and access data are limited to the market in which the app is running.
  - Data buyers can discover data by sending their queries to any node on the Storecoin network, but the requests are serviced by the respective markets. All nodes are aware of which markets host particular apps and some metadata about their data categories and data classes, so they can route the requests to appropriate markets.

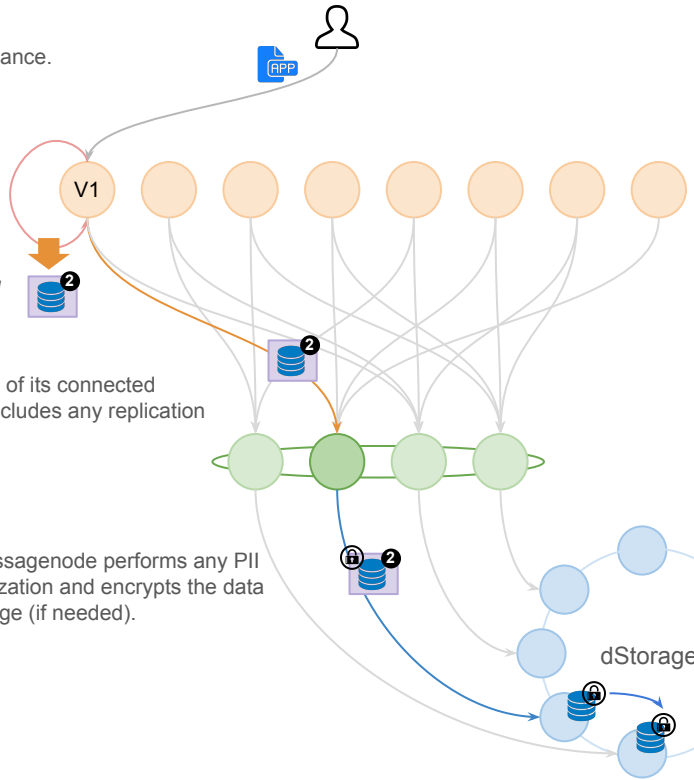
Proof-of-storage (PoSt), Proof-of-replication (PoRe), etc.

- Decentralized storage networks such as Filecoin use PoSt, PoRe, and other schemes to guarantee that storage promised is indeed provided
  - These approaches are designed to address Sybil, outsourcing, and generation attacks*.
  - While the concepts are applicable to any storage type, they are designed with file storage in mind.
  - The access pattern in these services are also pretty simple — the user uploads the files and later, accesses or downloads them.
- Storecoin Platform use cases and data access patterns are different
  - The data created by the tokenized apps are structured and enforced by a schema. The data is not file-based.
  - Part or whole of the data created can be categorized as personally identifiable, which requires it to be encrypted at rest and de-identified when accessed.
  - The data can be queried with rich semantics, similar to how traditional databases are queried.
  - The data is queried and accessed by potentially a large number of users and not just by those who created it.
  - For these reasons, the traditional PoSt, PoRe, and other schemes won't work, although conceptually they make sense.

* https://filecoin.io/proof-of-replication.pdf

How Storecoin market works?

When users create data it is wrapped in a transaction and sent to Messagenodes for storage.



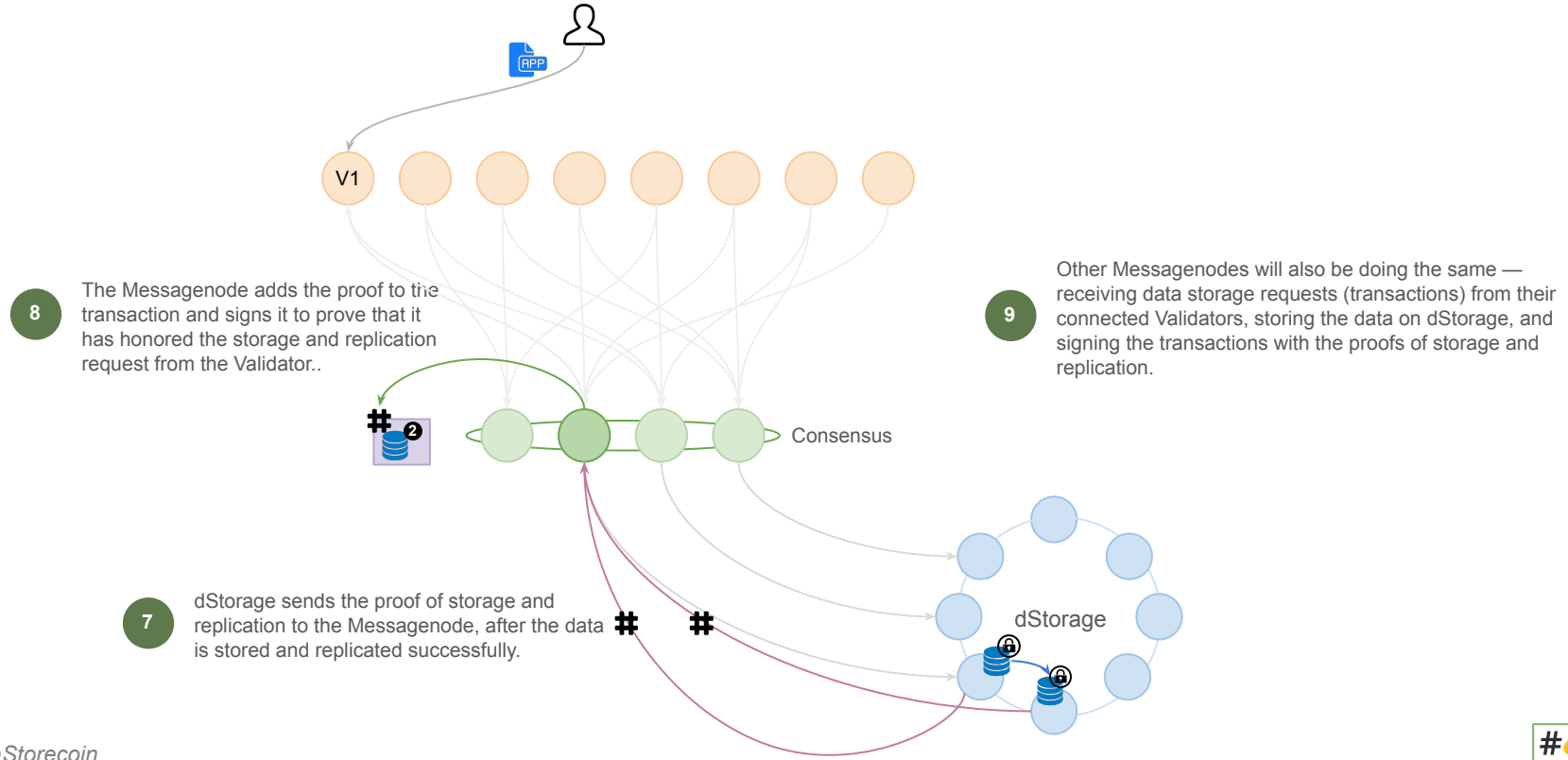
1. User invokes the app instance.
2. The app instance is executed in a secure sandbox in a Validator.
3. User creates some data in the app, which needs to be persisted. This data is *wrapped* in a settlement transaction*.
4. Validator sends the transaction created to one of its connected Messagenodes for storage. The transaction includes any replication needed (2 here), PII categorization, etc.
5. The Messagenode performs any PII categorization and encrypts the data for storage (if needed).
6. Messagenode stores the app data on decentralized storage. Any replication needs will be handled by the dStorage infrastructure.

@Storecoin

* Transaction type — `TXN_APP_DATA`.

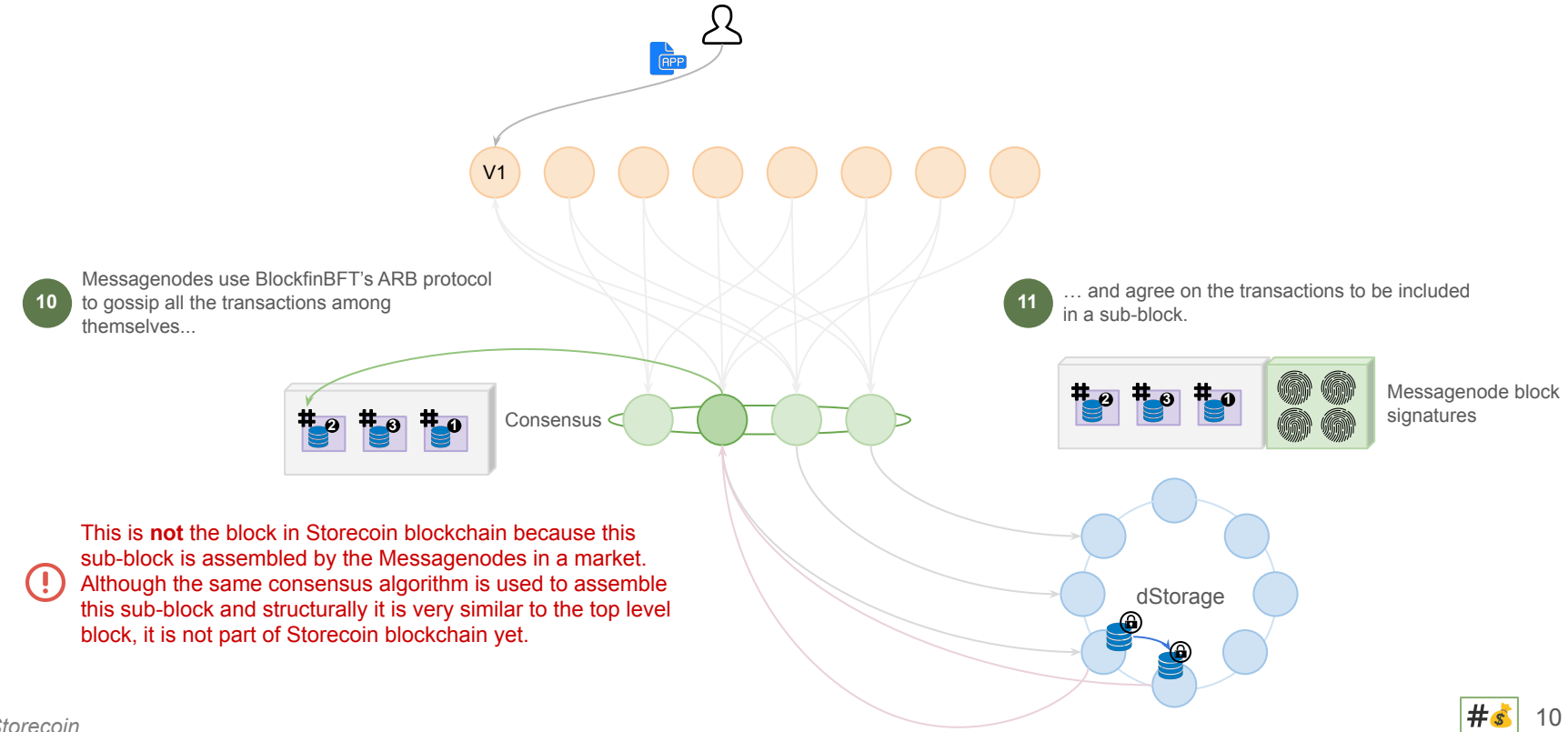
How Storecoin market works (continued)?

The proof-of-storage and proof-of-replication (if requested) are generated for the transaction.



How Storecoin market works (continued)?

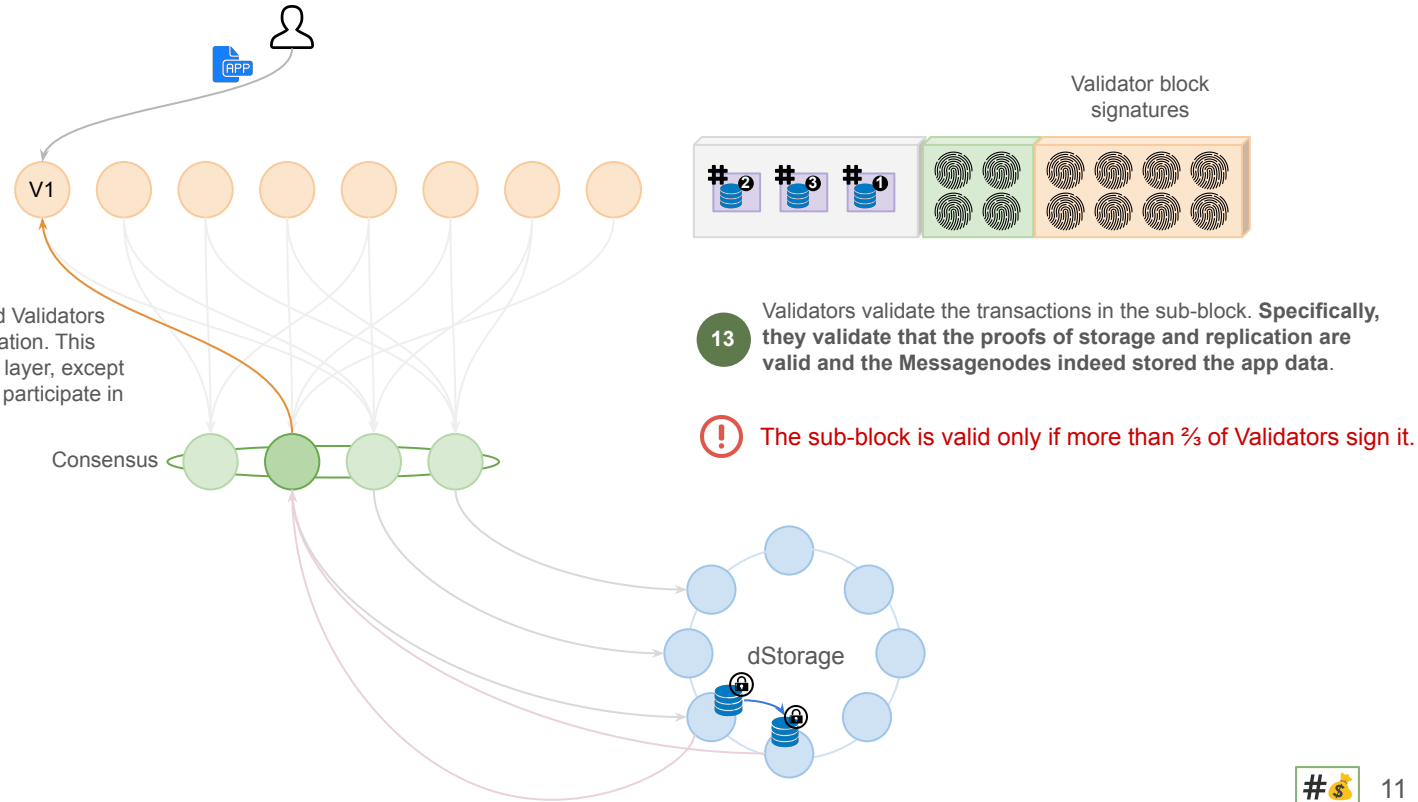
Messagenodes use BlockfinBFT's ARB protocol to assemble all such transactions into a sub-block.



This is **not** the block in Storecoin blockchain because this sub-block is assembled by the Messagenodes in a market. Although the same consensus algorithm is used to assemble this sub-block and structurally it is very similar to the top level block, it is not part of Storecoin blockchain yet.

How Storecoin market works (continued)?

Validators validate the transactions in the sub-block and sign it if proof-of-<data stored> is valid.



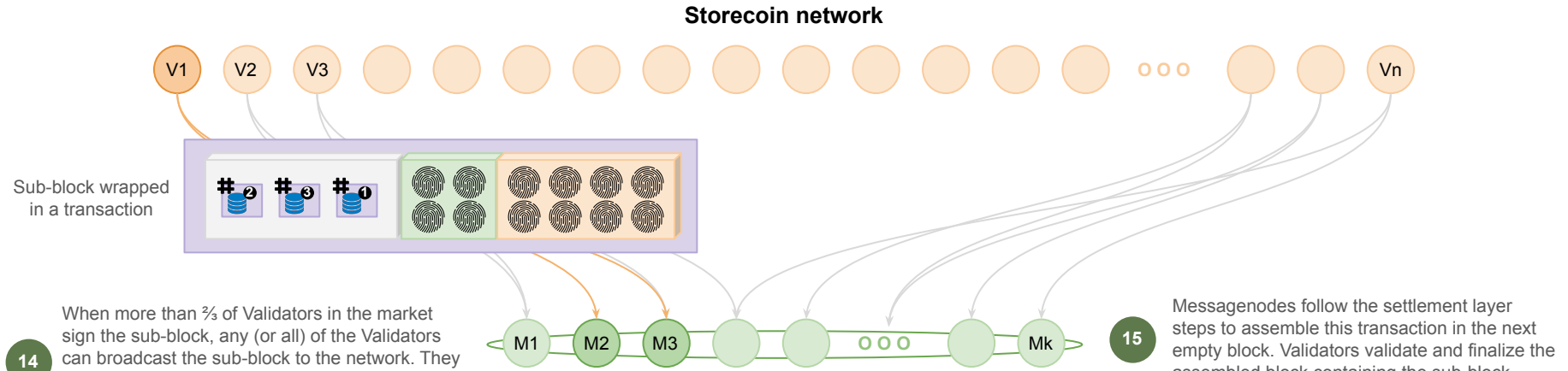
12 Messagenodes notify their connected Validators when the sub-block is ready for validation. This step is the same as in the settlement layer, except that only the Validators in the market participate in validating it.

13 Validators validate the transactions in the sub-block. **Specifically, they validate that the proofs of storage and replication are valid and the Messagenodes indeed stored the app data**.

The sub-block is valid only if more than ⅔ of Validators sign it.

Making sub-block visible to the entire Storecoin network

The sub-block is visible only within the market. It is secured by broadcasting it to the entire network.



14 When more than ⅔ of Validators in the market sign the sub-block, any (or all) of the Validators can broadcast the sub-block to the network. They do so by wrapping the sub-block in a transaction* and sending it to the connected Messagenodes as usual.

15 Messagenodes follow the settlement layer steps to assemble this transaction in the next empty block. Validators validate and finalize the assembled block containing the sub-block transaction as usual.

This is settlement layer consensus and block validation. In this phase, **all** Validators participate in verifying the proofs of storage and replication in the sub-block. The wrapped transaction is valid, only if more than ⅔ of total Validators agree.

* Transaction type — `TXN_SUB_BLOCK`.

Processing tokenized apps in a market — summary

- The tokenized app is deployed on a market consisting of a subset of Validators and Messagenodes.
- The execution runtime (secure sandbox) is provided by the Validators in the market.
- The storage service is provided by the Messagenodes in the market.
- The cost of hosting the app is borne by the nodes in the market.
- One Validator in the market executes the tokenized app instance.
- The data created by the user in the app instance is wrapped in a transaction (type `TXN_APP_DATA`) and sent to the Messagenodes of the market.
- The receiving Messagenode stores the data on Storage, which provides a proof of storage and replication.
- The Messagenode adds the proof to the transaction and signs it.
- The Messagenodes in the market use BlockfinBFT ARB protocol to create a sub-block of transactions of type `TXN_APP_DATA`.
- The Validators in the market validate the sub-block of transactions of type `TXN_APP_DATA` and sign them if the proofs of storage and replication are valid.

Processing tokenized apps in a market — summary

- When more than ⅔ Validators in the market sign the sub-block, any/all/some of them wrap the sub-block in another transaction (type `TXN_SUB_BLOCK`).
- This transaction is used to publish the sub-block to the entire network.
- The Validator sends the transaction of type `TXN_SUB_BLOCK` to its connected Messagenodes as in the settlement layer.
- This transaction is assembled into an empty block and is validated as in the settlement layer. When the assembled block is finalized, the sub-block is visible to the entire network.
- The same BlockfinBFT consensus algorithm is used to assemble and validate both the sub-blocks and blocks. So, if the security of one is proven, the other is automatically secure.

Possible malicious behaviors in markets

- A sub-block may contain malicious transactions (type `TXN_APP_DATA`) if:
  - a Validator in the market created that transaction maliciously. It created a dummy transaction without the user created data or it corrupted user data before wrapping it in the transaction
  - the Messagenode didn't persist the data created by the user, but faked the proofs and signed the transaction maliciously
  - the dStorage infrastructure didn't store or replicate the data, but faked the proofs for storage and replication
  - the Messagenodes in the market created the sub-block consisting of invalid transactions and yet, signed it for validation by the Validators
  - the Validators in the market signed the sub-block even when the proofs of storage and replication are failed to verify.
- A maliciously formed sub-block requires the collusion of majority of Validators and Messagenodes in the market.
- The only incentive for this collusion is saving the storage space. There is no other economic incentive for malicious behavior.

* https://filecoin.io/proof-of-replication.pdf

Possible malicious behaviors in markets

- If the market is used to host a zero-fee app, the dWorkers will lose the potential income from the data, if they fake storing the data. So, the economic reasons discourage them from acting malicious.
- If the market is used to host a paid app, they may be encouraged to fake storing the data. So, a malicious intent is practical.
- So, the Storecoin network as a whole has an obligation to ensure security and trust. A small subset of malicious markets can ruin the reputation of the entire network.

* https://filecoin.io/proof-of-replication.pdf

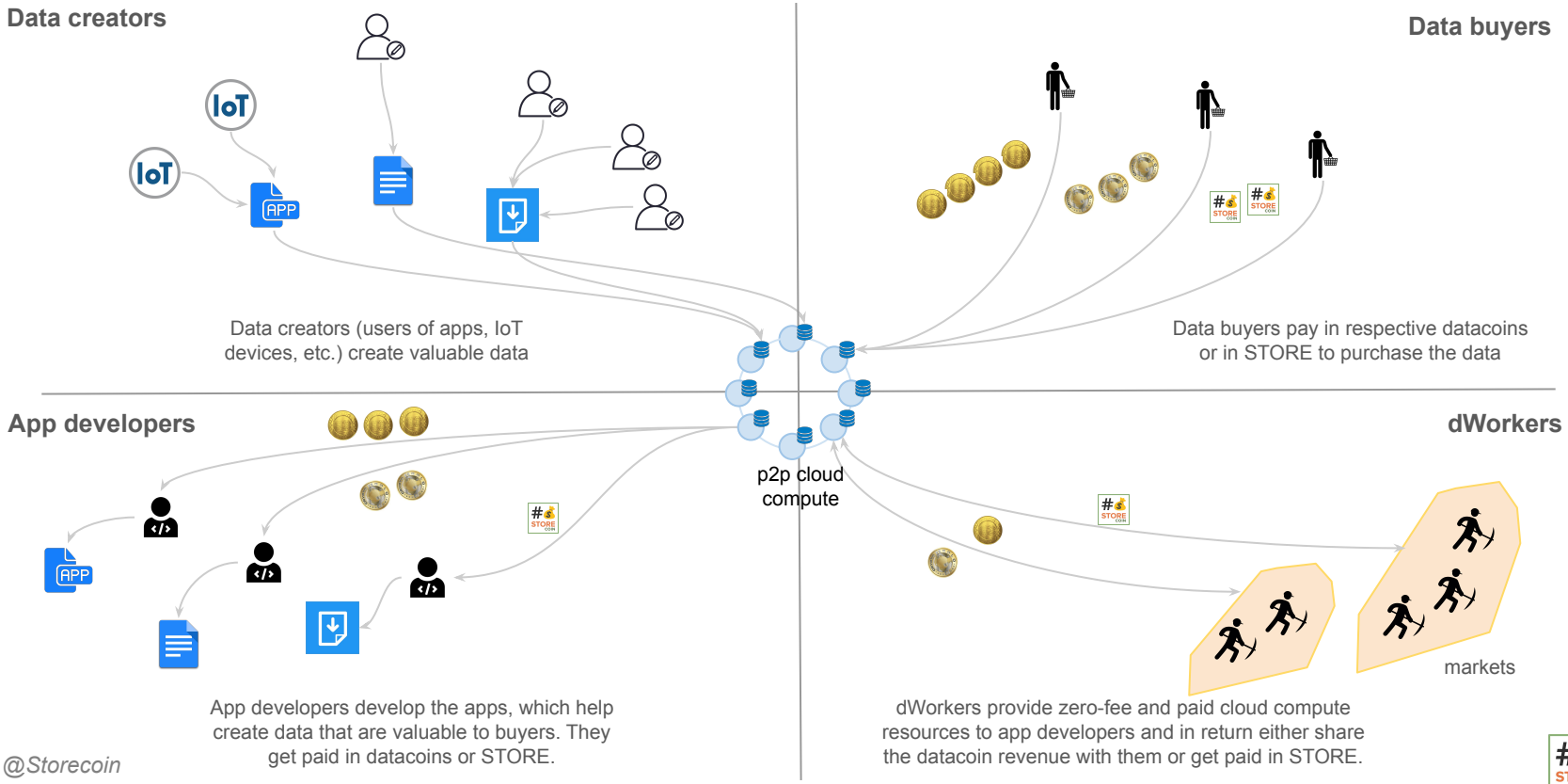
Addressing malicious behaviors in markets

- As described previously, an invalid sub-block is possible only if majority of Validators and Messagenodes collude to create it.
- However, the transaction of type `TXN_SUB_BLOCK` is validated by the entire network. Approving such a transaction at the network level requires more than ⅔ Validators across the entire network to collude. This violates the BFT threshold.
- If the transaction of type `TXN_SUB_BLOCK` is rejected, the dWorkers belonging to the market will be punished. Their stakes can be burnt.
- Considering the risk — the stakes of *all* dWorkers in the market will be burnt — the benefits of faking the storage are negligible. So, slashing addresses the network security.

* https://filecoin.io/proof-of-replication.pdf

Storecoin markets and networks

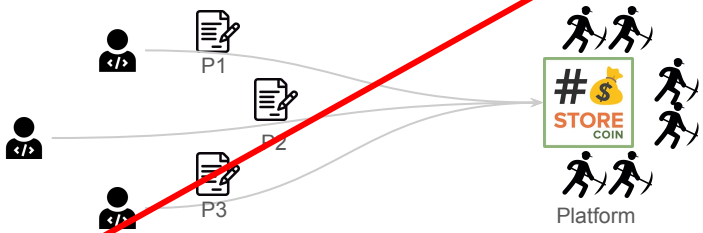
Storecoin market is made up of data creators, data buyers, app developers and dWorkers.



How Storecoin markets are formed? Shark Tank model

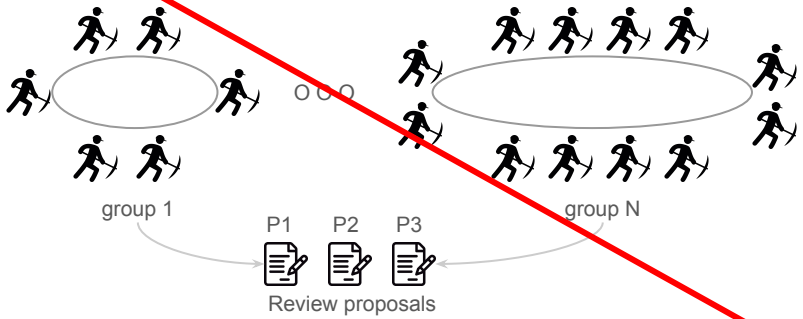
There can be infinite number of markets to fulfill the needs of data creators, buyers, and app developers.

1. App developers submit their proposals to Storecoin Platform describing their apps, classes of data the apps produce, potential revenue from the data, and so on. The proposals are startup pitches to dWorkers.



2. The app proposals are reviewed by dWorkers individually or in smaller groups. Over time, formal and informal groups of dWorkers are likely to be formed to attract developers and compete against each other.

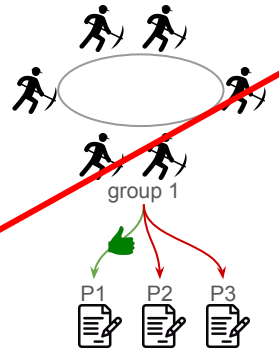
   The proposal review process involves dWorkers verifying the claims made by the developers, running their cost models, talking to developers, etc. Developers are also likely pitch their startups to dWorkers in the traditional form.



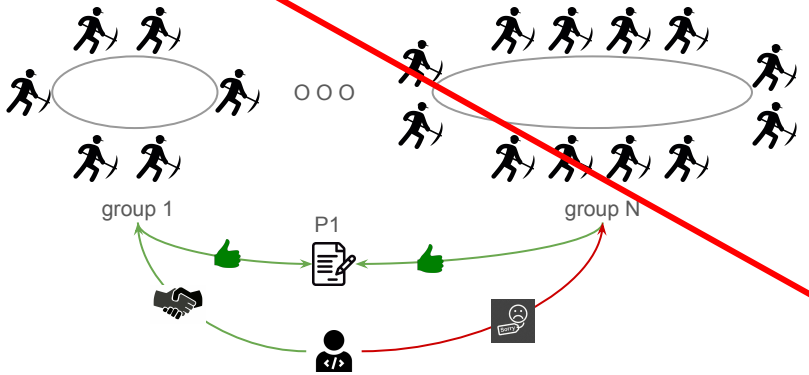
Developers compete against other developers and ...

… dWorkers compete against other dWorkers. Market economics determines the match.

3 In this case, the proposal, P1, wins over other proposals, which are rejected for various reasons. Group 1 dWorkers agree to host the app.

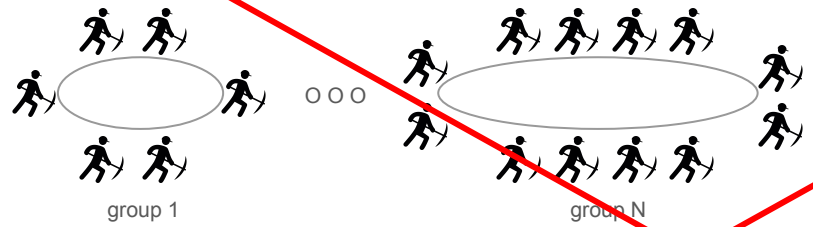


4 It is possible that multiple groups compete to host a particular app. In this case, two groups agree to host the app for the same proposal P1. Depending on the terms of hosting — % of revenue sharing or cost in STORE and other conditions — developers will choose one group over another.

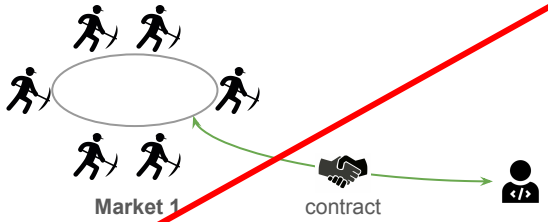


Groups to Markets to Markets

dWorkers form groups informally, which become markets, which eventually become markets.



dWorkers form groups to fulfill the needs of the apps or to address certain markets — such as location-based ads, patient data, etc. The groups may also be formed to provide specialized resources such as real-time database support, serverless infrastructure, etc. There is no predefined structure or requirements for a group, except for minimum number of nodes to ensure Byzantine tolerance.

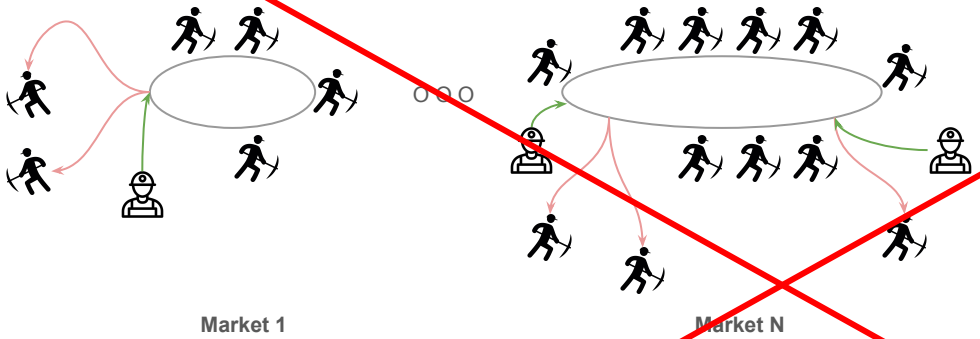


Once a group of dWorkers agrees to host an app, the group is contractually obligated to provide the promised resources to the app, in exchange for revenue either in datacoins or in STORE. These contracts can be viewed as a standard set of contracts drawn between a startup and its investors.

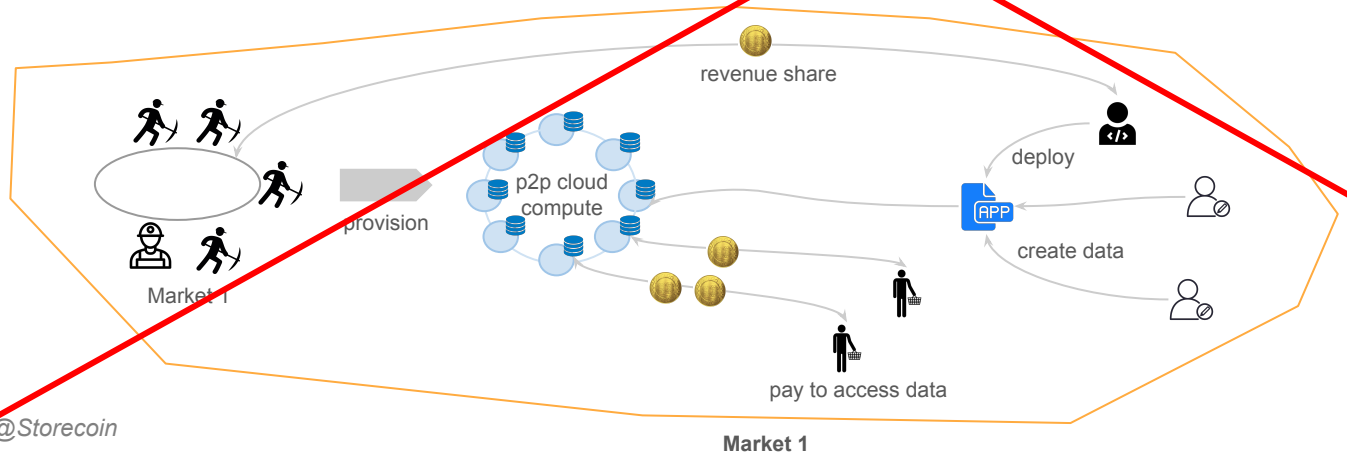
A contractually obligated group becomes a market.

Markets become markets when data trading takes place

Over time, certain markets specialize in serving certain markets or providing specialized services.



dWorkers can leave their markets for any reason after satisfying any contractual obligations (such as minimum period to serve, etc.) Similarly, new dWorkers can join existing markets either to fill a vacancy, or to meet new capacity requirements.



A combination of a market, app developers, the app, app users who create the data, and data buyers, who are interested in the data form a **market**. It is possible that a mature market can contain multiple of these combinations.