Storecoin Achieves +10,000 Transactions-Per-Second with Burst Traffic and 21 Validator Nodes

Storecoin moves blockchain closer to VISA network speeds while maintaining decentralization

Highlights

- Storecoin's DyPoS consensus algorithm achieves high consensus efficiency of **159,790 transactions per second** when small transactions (100 bytes) are sent to the validator nodes in a large burst (10,000 transactions in the burst) by a large number of clients (8 clients).
- High consensus efficiency of **10,282 transactions per second** is observed with continuous bursts (50 bursts, 50 transactions in a burst) sent by a large number of clients (8 clients) to 21 validator nodes. This setup is typical in a real-world settings.
- The consensus efficiency drops as the transaction volume increases. So when the transaction or burst sizes are too large, the consensus efficiency decreases.

Storecoin's mission is to become zero-fee, p2p, programmable payments infrastructure for the globe. The Dynamic Proof of Stake (DyPoS) consensus engine powering the Storecoin infrastructure is designed to process thousands of transactions

per second. When transactions arrive continuously but at lower rates, the consensus engine is capable of handling the incoming transactions, but how does it behave when transactions come in bursts? When Storecoin is used as the payment platform by merchants and app developers, the transactions are likely to come in bursts from multiple sources. So, we need to characterize the behavior of the consensus engine under such circumstances.

Purpose

In this series of tests, we want to measure the performance and stability of the validator nodes forming the consensus engine when the transactions are sent in bursts from multiple clients. The performance of the consensus engine is measured as the number of transactions processed per second. This is also called as *consensus efficiency*.

In our previous tests, we used **4** validator nodes to measure the consensus efficiency. This is the absolute minimum number of nodes required to tolerate Byzantine failure of **1** node. Such a setup offers minimum decentralization. In this test, we also want to increase the decentralization and see its effect on consensus efficiency.

Version **1** of DyPoS is partially built on top of Tendermint. The core consensus engine will be replaced eventually by BlockFin, Storecoin's leaderless, fork-tolerant, high-throughput consensus protocol. Version **1** of DyPoS is used to model various transaction load patterns to better understand the behavior of the validator nodes. These models help identify deficiencies in traditional approaches to consensus, so they can be better addressed in BlockFin. In this series, we examine the behavior of validator nodes when bursts of transactions are sent from multiple clients.

Figure **1** shows Storecoin application architecture, which follows Tendermint's ABCI Protocol.

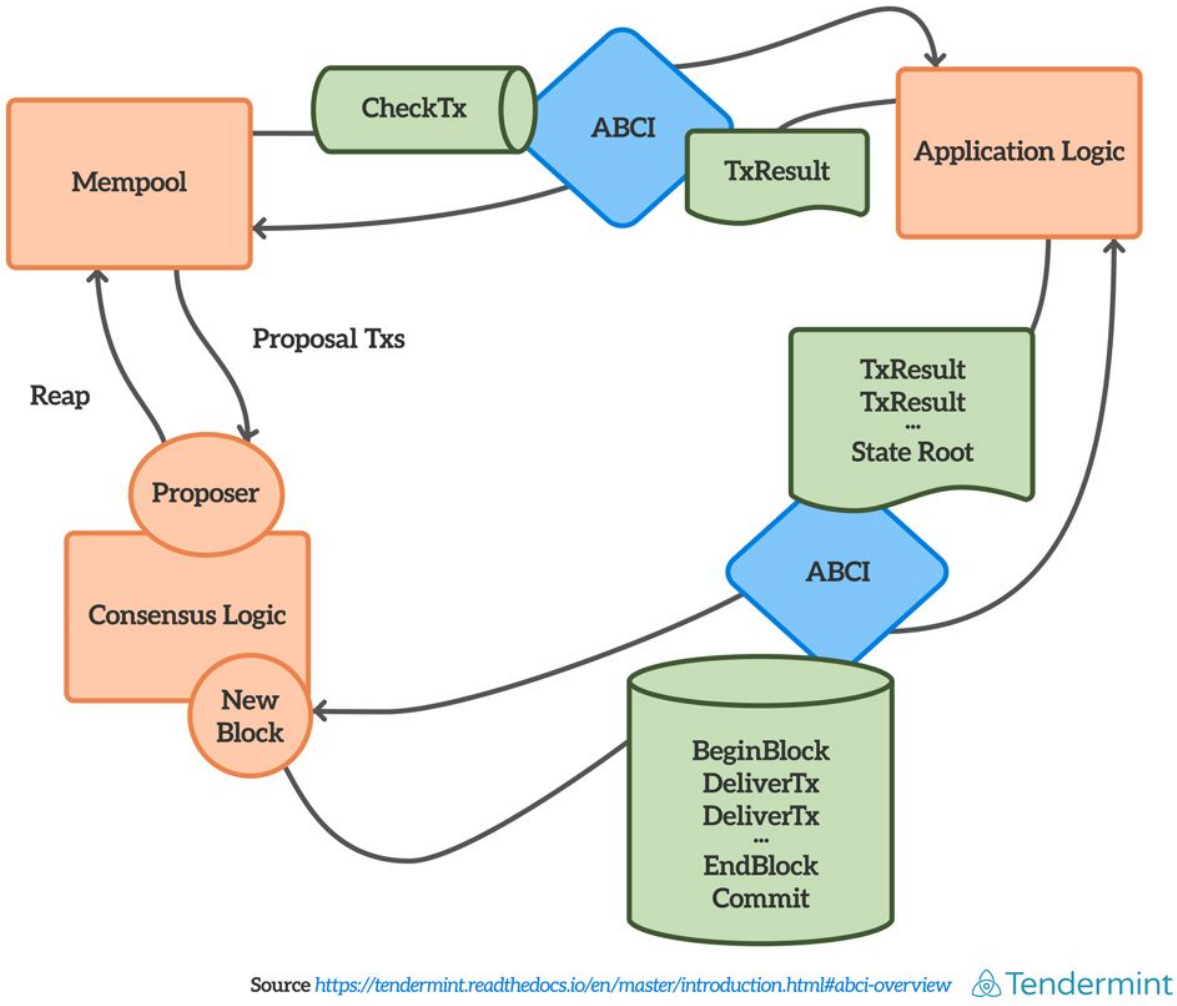

Figure 1 — Illustration of Storecoin Application and DyPoS Interface

Since the purpose of the tests is to measure the consensus efficiency and stability of the validator nodes, the application logic is replaced with Tendermint's *tm-bench* benchmarking tool. This tool is customized to generate bursts of transactions of different sizes.

About Storecoin

With its Dynamic, Fork-tolerant, and Auditable Proof of Stake-based consensus protocol (DyPoS), Storecoin will secure free transactions, high throughput, and a decentralized governance system with built-in checks and balances inspired by the U.S. Constitution. Also inspired by the supply and demand principles of Uber Surge Pricing (blockchain economics) and Power of Attorney model (blockchain scaling), Storecoin will secure crypto-powered incentives and payments inside of apps.

Test Summary - What We Are Trying to Learn

In this series of tests, we set up 8-node and 21-node test benches and measure the *consensus efficiency* in each setup. The 21-node setup provides greater decentralization compared to all the previous tests performed. In typical blockchain deployments, the greater the decentralization, the worse the consensus efficiency will be. This is inevitable because large numbers of nodes result in increased peer-to-peer communication, thus affecting the consensus efficiency negatively. We want to verify that the resulting drop in consensus efficiency is acceptable and the efficiency doesn't degrade to a level where it forces using fewer validator nodes.

The following tests are performed in this series.

4 clients, 8 validator nodes, fixed size transactions in multiple burst sizes

4 clients generate transactions of fixed size and send them to all the 8 validator nodes. The transactions are of size **500**, **1,000**, **5,000**, and **10,000** bytes in bursts of size **500**, **1,000**, **2,000**, **5,000**, and **10,000 transactions per burst. Each burst lasts for 5 seconds. The test setup, results, and observations for all the tests with 8 validator nodes are captured in** this report. Figure **2** visualizes the test setup.

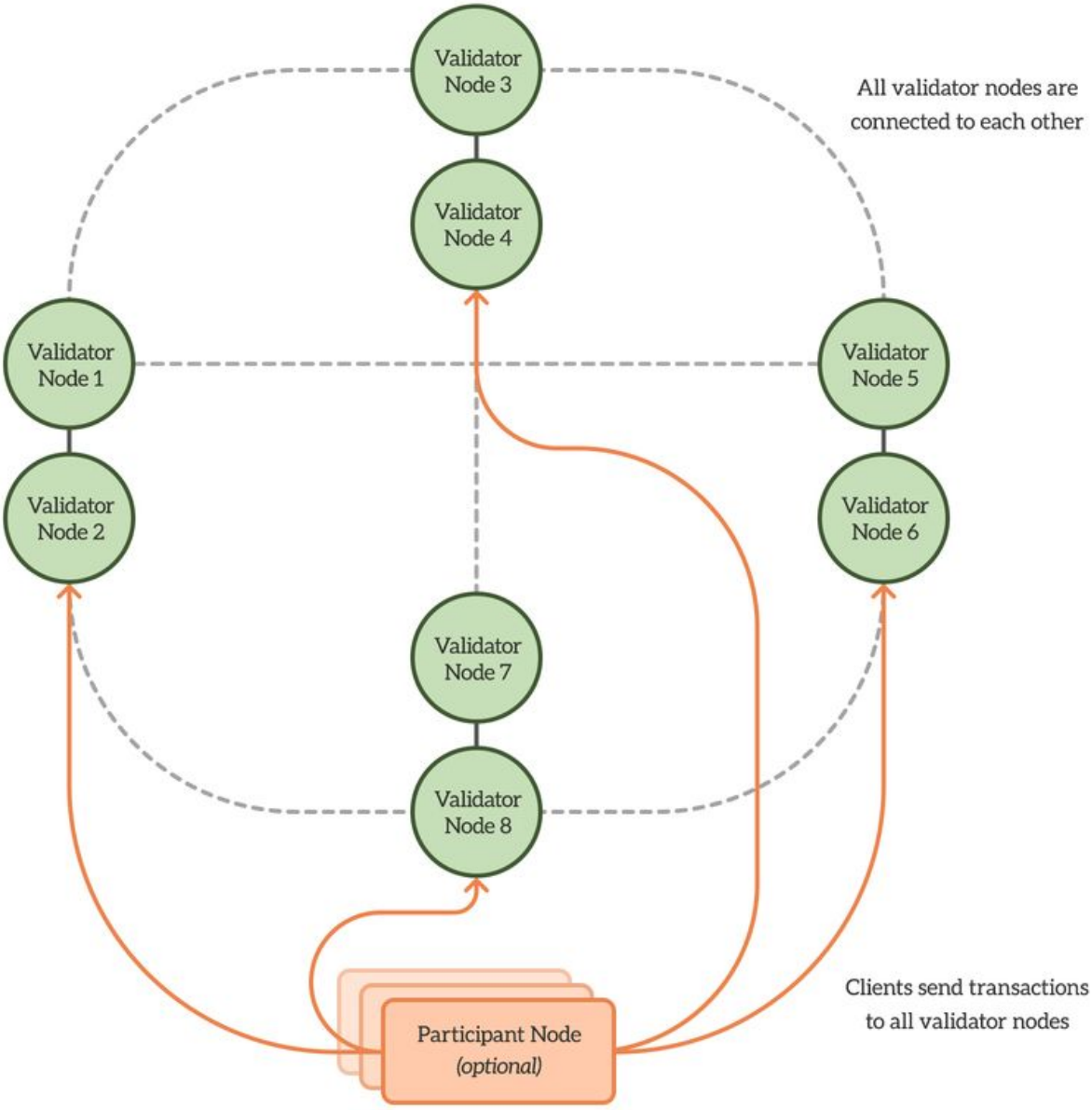

Figure 2 - Test setup with 8 validator nodes geographically distributed across U.S.

Table 1 below summarizes the test results.

TEST 4A, PART 1 - Clients - 4, Validators 8: Fixed Transactions							
Ser. No.	Run No.	Transaction Size	No. Of Transactions	No. of Clients	Total Transactions Served	Throughput (Txs/sec)	Test Results
1		100 bytes					
	A	100 bytes	500	4	8,000	4,000	Pass
	B	100 bytes	1,000	4	16,000	8,000	Pass
	C	100 bytes	2,000	4	64,000	16,000	Pass
	D	100 bytes	5,000	4	160,000	40,000	Pass
	E	100 bytes	10,000	4	320,000	80,000	Pass
2		500 bytes					
	A	500 bytes	500	4	8,000	4,000	Pass
	B	500 bytes	1,000	4	16,000	8,000	Pass
	C	500 bytes	2,000	4	64,000	16,000	Pass
	D	500 bytes	5,000	4	160,000	40,034	Pass
	E	500 bytes	10,000	4	Failed	Failed	Failed
3		1,000 bytes					
	A	1,000 bytes	500	4	8,000	4,000	Pass
	B	1,000 bytes	1,000	4	16,000	8,000	Pass
	C	1,000 bytes	2,000	4	64,000	16,000	Pass
	D	1,000 bytes	5,000	4	160,000	38,316	Pass
	E	1,000 bytes	10,000	4	Failed	Failed	Failed
4		5,000 bytes					
	A	5,000 bytes	500	4	8,000	4,000	Pass
	B	5,000 bytes	1,000	4	16,000	8,000	Pass
	C	5,000 bytes	2,000	4	64,000	3,184	Pass
	D	5,000 bytes	5,000	4	Failed	Failed	Failed
	E	5,000 bytes	10,000	4	Failed	Failed	Failed
5		10,000 bytes					
	A	10,000 bytes	500	4	8,000	4,000	Pass
	B	10,000 bytes	1,000	4	16,000	1,100	Pass
	C	10,000 bytes	2,000	4	64,000	512	Pass
	D	10,000 bytes	5,000	4	Failed	Failed	Failed
	E	10,000 bytes	10,000	4	Failed	Failed	Failed

Table 1 - 4 clients, 8 validator nodes, fixed size transactions

The best consensus efficiency of **80,000 transactions per second** is obtained in test run 1.E. At higher transaction and

burst sizes, the resulting transaction volume overwhelms the validator nodes, resulting in test failures. The test is characterized as *Failed* if the validator nodes are unable to keep up with incoming transactions and stop accepting them. Note that the validator nodes continue to run the consensus rounds for the transactions that are already accepted and at no time did they stop producing new blocks.

Test runs 3.D and 5.C are interesting. While 1.E produced the best consensus efficiency, it used smaller transaction size of 100 bytes. 3.D produced consensus efficiency of **38,316 transactions per second**, but with transactions of size **1,000** bytes and burst size of **5,000**. So, while the consensus efficiency is much lower than in test run 1.E, the run 3.D demonstrates processing large transactions in a large burst size.

The test run 5.C shows the effect of very large transaction sizes. The consensus efficiency dropped to **512 transactions per second**. This demonstrates that the DyPoS consensus engine favors small to midsize transactions in small to large burst sizes. The typical transaction size in Storecoin is less than **500** bytes, so this behavior doesn't affect the use cases that Storecoin is designed for.

8 clients, 8 validator nodes, fixed size transactions in multiple burst sizes

This test is similar to the test above, except 8 clients are used to generate transactions instead of 4. This test is intended to evenly saturate all 8 validator nodes, so they are not idling. The results are summarized in Table **2** below.

TEST 4A, PART 2 - Clients - 8, Validators 8: Fixed Transactions							
Ser. No.	**Run No.**	**Transaction Size**	**No. Of Transactions**	**No. of Clients**	**Total Transactions**	**Throughput (Txs/sec)**	**Test Results**
6		**100 bytes**					
	A	100 bytes	500	8	32,000	8,000	Pass
	B	100 bytes	1,000	8	64,000	16,000	Pass
	C	100 bytes	2,000	8	128,000	32,000	Pass
	D	100 bytes	5,000	8	320,000	80,000	Pass
	E	100 bytes	10,000	8	640,000	159,790	Pass
7		**500 bytes**					
	A	500 bytes	500	8	32,000	8,000	Pass
	B	500 bytes	1,000	8	64,000	16,000	Pass
	C	500 bytes	2,000	8	128,000	32,000	Pass
	D	500 bytes	5,000	8	320,000	86,933	Pass
	E	500 bytes	10,000	8	Failed	Failed	Failed
8		**1,000 bytes**					
	A	1,000 bytes	500	8	32,000	8,000	Pass
	B	1,000 bytes	1,000	8	64,000	16,000	Pass
	C	1,000 bytes	2,000	8	128,000	32,000	Pass
	D	1,000 bytes	5,000	8	Failed	79,086	Pass
	E	1,000 bytes	10,000	8	Failed	Failed	Failed
9		**5,000 bytes**					
	A	5,000 bytes	500	8	32,000	8,000	Pass
	B	5,000 bytes	1,000	8	64,000	16,000	Pass
	C	5,000 bytes	2,000	8	128,000	4,501	Pass
	D	5,000 bytes	5,000	8	Failed	Failed	Failed
	E	5,000 bytes	10,000	8	Failed	Failed	Failed
10		**10,000 bytes**					
	A	10,000 bytes	500	8	32,000	8,000	Pass
	B	10,000 bytes	1,000	8	64,000	2,533	Pass
	C	10,000 bytes	2,000	8	Failed	Failed	Failed
	D	10,000 bytes	5,000	8	Failed	Failed	Failed
	E	10,000 bytes	10,000	8	Failed	Failed	Failed

Table 2 - 8 clients, 8 validator nodes, fixed size transactions

Test run 6.E produced best consensus efficiency of **159,790 transactions per second**. The consensus efficiency is nearly doubled in this case. This is because 8 clients are used to generate the transactions, so overall twice the total transactions are produced in this test. Test runs 8.D and 10.B are similar to test runs 3.D and 5.C in the previous test.

4 and 8 clients, 8 validator nodes, random size transactions in multiple burst sizes

In this test, random transaction sizes are used instead of fixed sizes used in the previous tests. The size distribution is done as shown in table 3 below. This test models real world settings where majority of transactions are of smaller sizes with a small percentage of medium and large size transactions.

Transaction Group	Transaction Size	Percent Allocation
Group 1	50-250 bytes	60%
Group 2	251-500 bytes	20%
Group 3	501-1,000 bytes	15%
Group 4	1,001+ bytes	5%

Table 3 - Random transaction size distribution

The test results are described in Table **4** below.

Ser. No.	Run No.	Transaction Size	No. Of Transactions	No. of Clients	Total Transactions	Throughput (Txs/sec)	Test Results
11		**Random – 4 Connects**					
	A	100 bytes	500	4	32,000	4,012	Pass
	B	100 bytes	1,000	4	64,000	7,970	Pass
	C	100 bytes	2,000	4	Failed	Failed	Failed
	D	100 bytes	5,000	4	320,000	Failed	Pass
	E	100 bytes	10,000	4	640,000	Failed	Pass
12		**Random - 8 Connects**					
	A	500 bytes	500	8	32,000	3,993	Pass
	B	500 bytes	1,000	8	64,000	8,032	Pass
	C	500 bytes	2,000	8	128,000	16,067	Pass
	D	500 bytes	5,000	8	Failed	Failed	Failed
	E	500 bytes	10,000	8	Failed	Failed	Failed
13		**Percent - 4 Connects**					
	A	1,000 bytes	500	4	32,000	2,026	Pass
	B	1,000 bytes	1,000	4	64,000	3,970	Pass
	C	1,000 bytes	2,000	4	128,000	7,987	Pass
	D	1,000 bytes	5,000	4	Failed	Failed	Failed
	E	1,000 bytes	10,000	4	Failed	Failed	Failed
14		**Percent - 8 Connects**					
	A	5,000 bytes	500	8	32,000	4,010	Pass
	B	5,000 bytes	1,000	8	64,000	7,908	Pass
	C	5,000 bytes	2,000	8	128,000	16,052	Pass
	D	5,000 bytes	5,000	8	Failed	Failed	Failed
	E	5,000 bytes	10,000	8	Failed	Failed	Failed

Table 4 - 4 and 8 clients, 8 validator nodes, fixed size transactions

With 4 clients sending random sized transactions, the best consensus efficiency of **7,987 transactions per second** is observed in test run 13.C. Notice the sharp drop in consensus efficiency. The drop is explained with the same reason as for test runs 3.D and 8.D with fixed size transactions. Since we are using transactions of random sizes, overall size of the burst is larger

than that with the fixed size transactions. This results in lower consensus efficiency.

With 8 clients sending random sized transactions, the best consensus efficiency of **16,067 transactions per second** is observed in test run 12.C. Notice that the consensus efficiency is nearly doubled because the validator nodes are being used to their capacity.

Random size transactions result in a decreased consensus efficiency because the overall transaction volume is much higher than the fixed size transactions.

2 clients, 21 validators 50 bursts, 500 txs/burst, random transaction sizes, 30 second pause between bursts

In this series of tests with 21 validator nodes, our goal is twofold:

1. Measure the consensus efficiency with a greater decentralization and verify if the resulting drop in performance is acceptable for the practical use cases Storecoin is designed for
2. Characterize the behavior of validator nodes when *multiple* bursts of transactions are sent from the clients, resulting in a continuous load on the consensus engine

The test setup, results, and observations for all the tests with 21 validator nodes are captured in this report. Figure **3** visualizes the test setup.

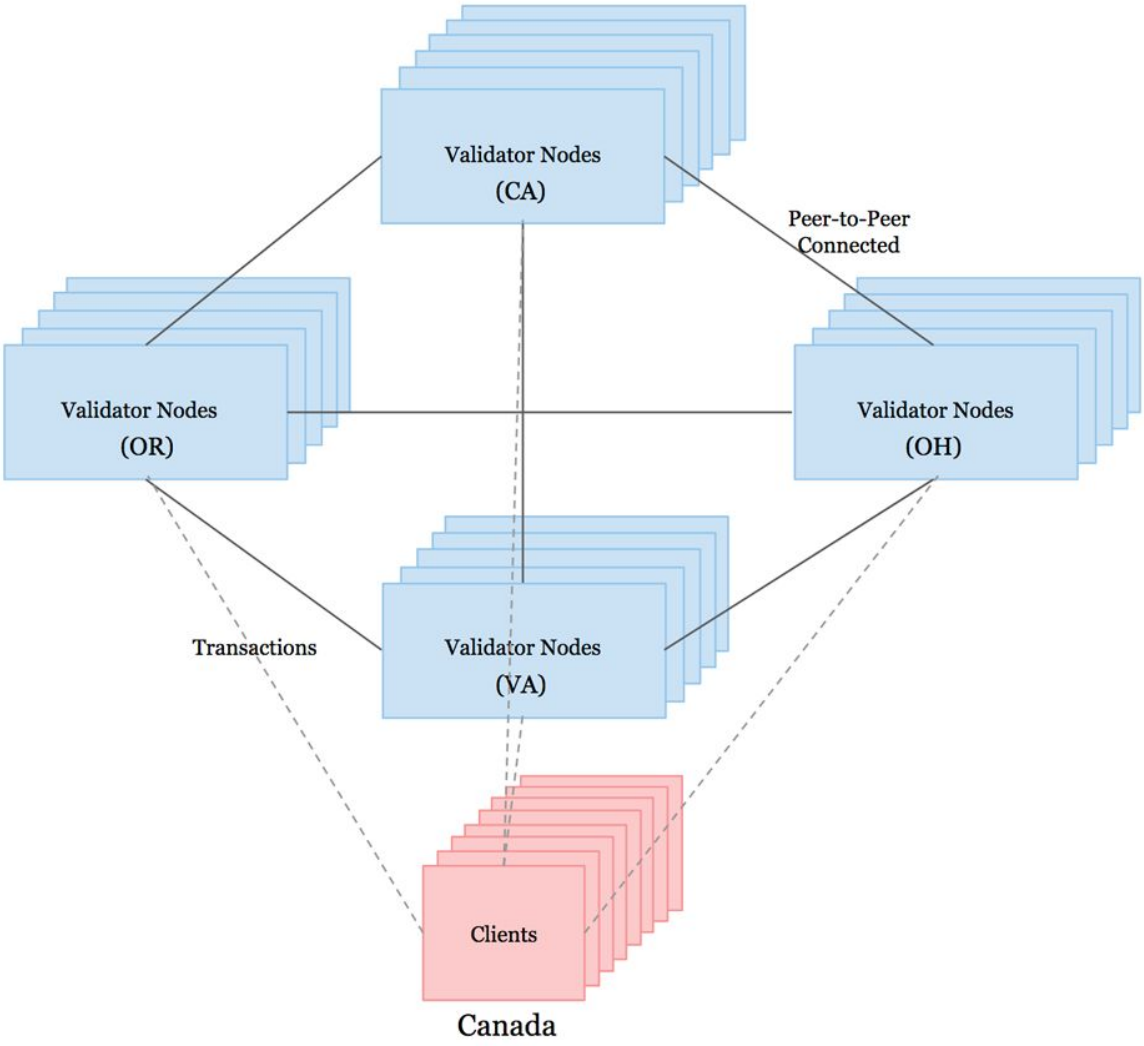

Figure 3 - Test setup with 21 validator nodes geographically distributed across U.S.

In this test, 2 clients generate random sized transactions with **500 transactions** in a burst. Each burst lasts for 5 seconds as before. Each client sends **50 bursts** with a 30 second pause between them. The consensus efficiency is measured per burst as before. The resulting list for **50 bursts** is as follows.

The consensus efficiency ranges from a low of **4,750** to a high of **7,759**transactions per second. We theorize two reasons for the wide range in the consensus efficiency numbers.

1. Transaction sizes - Since each burst generates transactions of random sizes independently, the transaction volume (transaction size * burst size) varies from burst to burst. The higher the transaction volume, the lower the consensus efficiency will be.
2. Transaction accumulation - As validators receive continuous bursts of transactions from clients, the transactions accumulate because of the limited amount of time to process them. As a result, the consensus efficiency varies depending on the amount of accumulation of transactions awaiting processing between bursts.

The second claim above on transaction accumulation must be verified because depending on the size of accumulation, the consensus efficiency can increase or decrease. This claim is verified in the next test.

2 clients, 21 validators 50 bursts, 500 txs/burst, random transaction sizes, 120 second pause between bursts

In this test, the pause between the bursts is increased to 120 seconds, so the validator nodes have sufficient time to cool off. This should minimize transaction accumulation between the bursts. The test results are listed here.

The consensus efficiency ranges from a low **2,348** transactions per second to a high **2,654** transactions per second.

In other words, the consensus efficiency dropped with a longer pause between the bursts. This is because the longer pauses result in under-utilizing the validator nodes. This observation brings up an important characteristics of Storecoin DyPoS consensus engine. If the validator nodes are under-utilized or overwhelmed with transactions, the consensus efficiency drops. If, on the other

hand, they are utilized to their capacity, the consensus efficiency would be at its peak. This leads us to the next test.

8 clients, 21 validators 50 bursts, 50 txs/burst, random transaction sizes, 30 second pause between bursts

This test is designed to verify the claim that the consensus efficiency peaks when validator nodes are loaded just right. In this test **8** clients are used to generate transactions instead of **2** in the previous tests. The burst size is reduced to **50** transactions per burst instead of **500** transactions. This results in smaller bursts arriving concurrently from a larger number of clients. The validator nodes get a continuous supply of transactions without overwhelming them and at the same time ensuring better utilization of them. The test results are available here.

The consensus efficiency ranges from a low of **5,592** transactions per second to a high **10,282** transactions per second.

This is a significant improvement compared to the consensus efficiencies observed in previous tests with 2 clients. The improved efficiency is due to optimal utilization of the validator nodes. It is also important to note that the consensus engine handles concurrent bursts from multiple clients without affecting the consensus efficiency.

Conclusion

In this series of tests we learned about the behavior of validator nodes with increased decentralization and continuous bursts of transactions. In typical blockchain deployments, greater decentralization severely affects consensus efficiency, resulting in centralized deployments. The tests performed here verified that Storecoin DyPoS is not affected by centralization. While consensus efficiency with 21-node setup did drop from the 8-node setup where transactions were sent in a single burst, the drop in

consensus efficiency is respectable because multiple bursts of transactions are processed by a larger number of validator nodes.

In the test with large bursts of transactions, consensus efficiency was observed ranging from a low of **4,750** to a high of **7.759** transactions per second. In the test with smaller bursts of transactions but with greater concurrency, consensus efficiency was observed ranging from a low of **5,592** to a high of **10,282**transactions per second.

We are also able to characterize the behavior of the validator nodes when they are overwhelmed. The 8-node setup helped us model the failure cases resulting in low consensus efficiency or nodes refusing to accept incoming transactions.

With initial testing of DyPoS complete, the behavior of validator nodes is now understood under a variety of load conditions. This behavior can be summarized as follows:

- The consensus efficiency is related to transaction volume, but the relationship is not constant across different loads. Up to a threshold, consensus efficiency is directly proportional to transaction volume. This is the threshold at which the validator nodes are utilized to their full capacity. Once this threshold is breached, the consensus efficiency is inversely proportional to transaction volume.
- The validator nodes are well behaved with large concurrency (a larger number of clients sending transactions concurrently) and small burst sizes. This is typically the case in real world settings. Transactions trickle down in smaller bursts from a large number of clients with occasional bursts involving a large number of transactions.
- Validator nodes never crashed or created incomplete blocks in any of the tests. When their transactional capacity is

reached, incoming transactions are rejected. Such events are categorized as a “failure” for testing purposes.

- Greater decentralization is practically achievable without sacrificing the performance.