Scalable Blockchain
March 12, 2016

Rhett Creighton[1]

No Institute Given

Abstract. The purpose of this work is to demonstrate a fast, efficient, highly scalable blockchain. Current block chain implementations have performance limitations that make them unsuitable for high speed record keeping. For example, the Bitcoin blockchain has a restricted sustained transaction rate of 7tps[1]. Other blockchain implementations have maximum transaction rates in the hundreds or possibly thousands of transactions per second, but do not scale beyond that. In this paper, we describe the Domus Tower Blockchain, which has been benchmarked at ingesting almost 100,000 transactions per second on hardware costing approximately $20 per hour on Amazons AWS with the potential to scale to greater than 10 Million transactions per second.

Keywords: blockchain, block chain, merkle graph, big data

1 What is a Blockchain?

Satoshi first described a "block of items" on page 2 of his whitepaper[2]

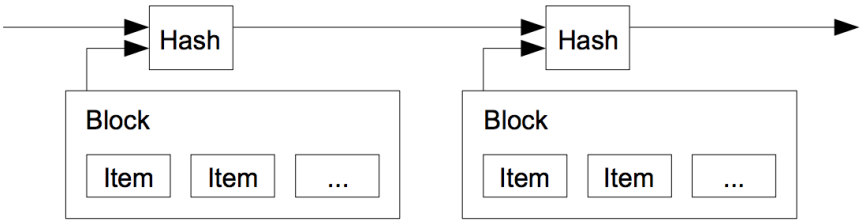

Fig. 1. Satoshi's timestamp server. Items are inserted into blocks. Each block is hashed along with the hash of a previous block. This creates the latest hash, or Merkle Root for the block chain.

This fits our basic definition of a blockchain. Additionally, we require that all transactions in our blockchain are digitally signed. Explicitly, the basic properties of a Domus Blockchain are:

1. Data storage is contained in a Merkle Directional Acyclic Graph (Merkle-DAG), a special case of a Merkle Tree, described in: US4309569[3] .
2. Data transmitted to the blockchain is signed with a cryptographic digital signature and the signature is verified before the data is included in a new node in the Merkle Graph.
3. The latest hash in our blockchain, known as the Merkle Root guarantees an immutable history of all of the signed messages stored in all the prior nodes of the entire Merkle Graph.

Unlike other Blockchains, such as the Bitcoin blockchain, our blockchain does not include Proof of Work, Mining, Proof of Stake, or any form of Byzantine Fault Tolerance.

	Max Transaction Rate	Tracks Assets and Liabilities?	Trust Model
Bitcoin	~7 tps	No	Proof of Work
Ripple	~few thousand tps	Yes	Federated Voting
Domus	10M+ (scalable)	Yes	Trusted

Fig. 2. How the Domus Blockchain compares to other crypto-ledgers.

2 Architecture Overview

We use a set of microservices to work together in an efficient, scalable architecture. The services are described below and include a signature verifier, transaction batcher, block maker, and a client.

3 Consensus

Consensus among agents on a network is a fundamental problem in distributed computing. The well-known CAP Theorem states that it is not possible to guarantee consistency and availability across all nodes.

The problem becomes more difficult when actors are anonymous and potentially malicious, incentivized to cheat or lie about transactions. This has been

phrased as the well-known Byzantine Generals Problem, for which Bitcoins Proof of Work provides a clever solution.

The Domus Tower Block Chain achieves high scalability and availability by adopting a permissioned, trusted-node architecture with eventual-consistency. Our model does not use a federated or any kind of vote-based approach to writing transaction data. Any agent that has write access to a block chain, and 100% authority to write any transactions to that chain. Authority is completely centralized under this model.

4 Why Not Use A SQL Database?

By adopting a permissioned, centralized, trusted-node block chain, one might ask why not use a SQL database, which is also permissioned and centralized. The advantages of using a block chain are:

1. A block chain gives us an immutable, permanent record of timestamped transactions. All transactions are cryptographically audited in real time with a Merkle Root.
2. Every transaction is signed with a public/private key signature that is impossible to forge, in real time.
3. The append-only, timestamped structure of a block chain actually makes it much easier to scale under a model of eventual consistency.

In summary, we feel that tracking asset ownership in a corruptible, mutable system that needs to be carefully, manually audited on a frequent basis is a fragile method of record keeping and prone to error.

5 Trusted-Nodes

The Domus Tower Block Chain uses a simple, ad hoc trusted node approach, similar to the GIT version control p2p system of trust. This is not different from how the industry works today where parties know each other and trust data feeds.

Disputes can be resolved outside the block chain. For example, if Bob thinks that his account balance is incorrect, he might call his broker on the phone and resolve the issue.

Because the Domus blockchain is immutable, resolutions to errors are appended to the blockchain. Take, for example, the well-known double-spend attack:

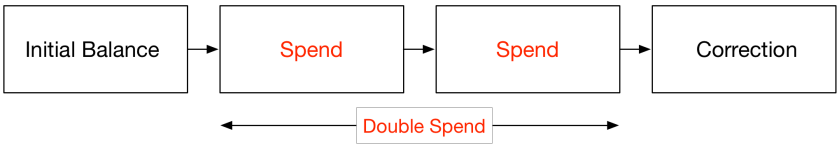

Fig. 3. In the Domus blockchain, though unlikely to ocur, double spends are theoretically possible for a short time. Once detected, a correction is applied in the form of an appended item to the blockchain

Because double spend records are not desired, The blockchain application layer can simply detect and filter out potential double spend transactions before applying them to the blockchain. This will introduce some additional latency in block creation, however.

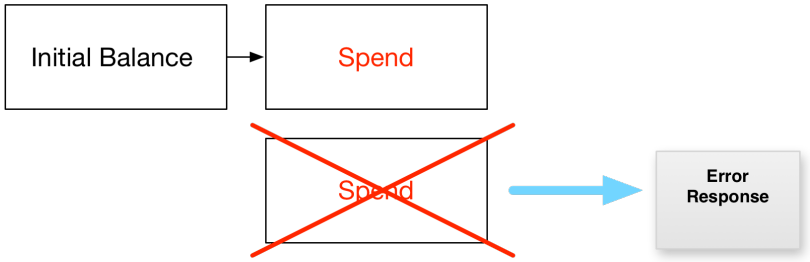

Fig. 4. In this example, the application layer detects a potential double-spend transaction and returns an error before adding it to the block chain.

6 Signature Verification

Verifying signatures on transactions is a computationally expensive process. However, the problem of verifying many signatures is also Embarrassingly Parallel, meaning that the jobs can easily be computed in parallel across many computers.

Because signature verification is one of the most expensive pieces in this process, we choose to use a high speed public key signature system such as Ed25519[4].

A signature verification process simply takes a message with a signature and public key as an input. If the signature does not match the message and the

public key, the process handles the input as an error. If the signature is verified, the process handles the input as verified, and passes it along to be written in the block chain.

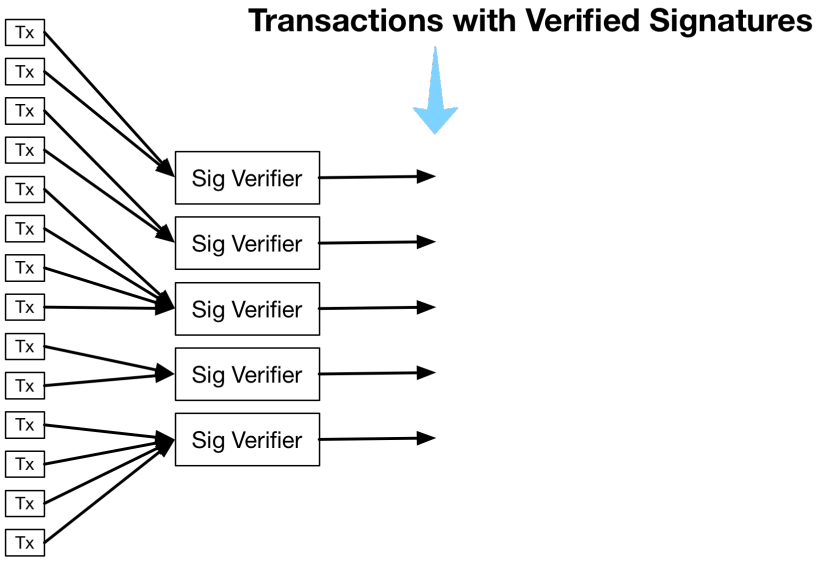

Fig. 5. An overview of the Domus signature verification microservices. If the signature is valid on a transaction, the signature verification service passes the transaction on to be processed into a block.

7 Transaction Batching

Verified signed messages are passed to transaction batchers. Transaction batchers can perform arbitrary operations on batches of input before passing it to a block maker.

One example is that a transaction batcher can compress transactions after every X transactions, or after every Y milliseconds have passed.

A single compressed message is passed on to a block making service, composed of many transactions.

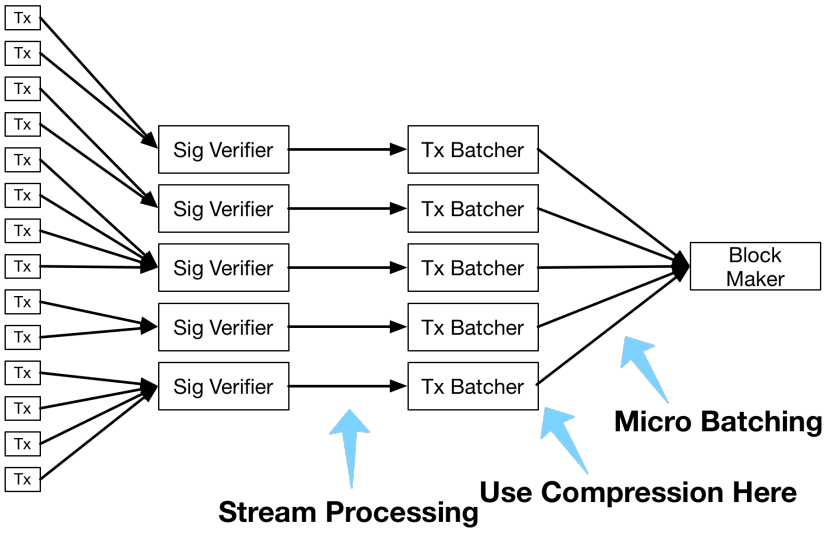

Fig. 6. Transaction batcher services receive verified transactions and batch them before sending them to a blockmaker service.

8 Block Making

A block maker takes as input compressed batches of transactions with verified signatures. It stores each piece of input in a linked-list that represents the data of the current block being written.

A block represents data written in a specific window of time. It must have been written after the previous block, and before the subsequent block in a block chain.

In our example, a new block is created for each second. We use a Key-Value Storage system, where the key is used to designate the block time window, and the value is the linked-list of compressed verified transaction data.

The Merkle Root of a block is generated by running a cryptographic hash function on [the data in the current block AND the previous blocks merkle root]. The sole exception is the first block in a block chain, which has no previous blocks merkle root. In this case, an empty string can be used as the previous merkle root, or any static value can also be used.

In order to accommodate a high transaction rate, block chain transactions are recorded using volatile high speed memory, such as RAM. Data persistence is achieved by by flushing the in-memory working set of data to a permanent storage device periodically. In our example, fsync is used to perform this operation once per second.

9 Client

A client uses a public/private keypair to digitally sign messages that are sent to a block chain building service.

10 Benchmarks

10.1 Hardware

- 1x Redis Server (where the block chain is stored)
- 1X Block Maker
- 2X Transaction Batchers
- 2X Signature Verifiers
- 2X Client Servers

Each server is c4.8xlarge on amazon at $1.675 per hour.
Total ingest rate 90k-100k tx per second sustained.
The Redis Server and Block Maker can handle about 100x the load.
We are CPU limited on the Signature Verifiers.
...

11 Double-Entry Accounting

The Domus Blockchain supports double-entry accounting. Under this system, assets are tracked, as well as liabilities.

This is very much in contrast to the Bitcoin blockchain where only assets are tracked, even though transactions are recorded in the form of inputs and outputs.

Date	From	To	Amount
March 1	Block Reward	A	25 BTC
March 1	A	B	5 BTC
March 1	B	C	2 BTC

Fig. 7. Single-entry accounting Bitcoin example where only assets are counted, and liabilities do not exist.

The advantages of double-entry bookkeeping are numerous, and beyond the scope of this paper. In a simple example, a stock broker might hold stock for the benefit of his clients. The clients' accounts will show a positive balance while the broker's account will show a negative balance. The negative balance reflects the total number of shares that are owed on this balance sheet.

Date	Account	Debits	Credits	Balance
March 1	A		+100 MSFT	+100 MSFT
March 1	C	-100 MSFT		-100 MSFT
March 1	B		+300 MSFT	+300 MSFT
March 1	C	- 300 MSFT		-400 MSFT

Fig. 8. Double-entry accounting on the Domus Blockchain tracking assets and liablities.

In the above example, we are tracking credits, and debits, as well as running balances. In the Bitcoin blockchain, for example, running balances are not tracked. This means that wallets need to either download the entire blockchain, or use an SPV thin wallet[5] just to know their own account balance.

12 Linked Blockchains

Linked blockchains can be created where the assets of an account on one block chain must match the liabilities on the account of another blockchain. Under this system, a parent chain is used to guarantee the solvency for account holders on a sub-blockchain.

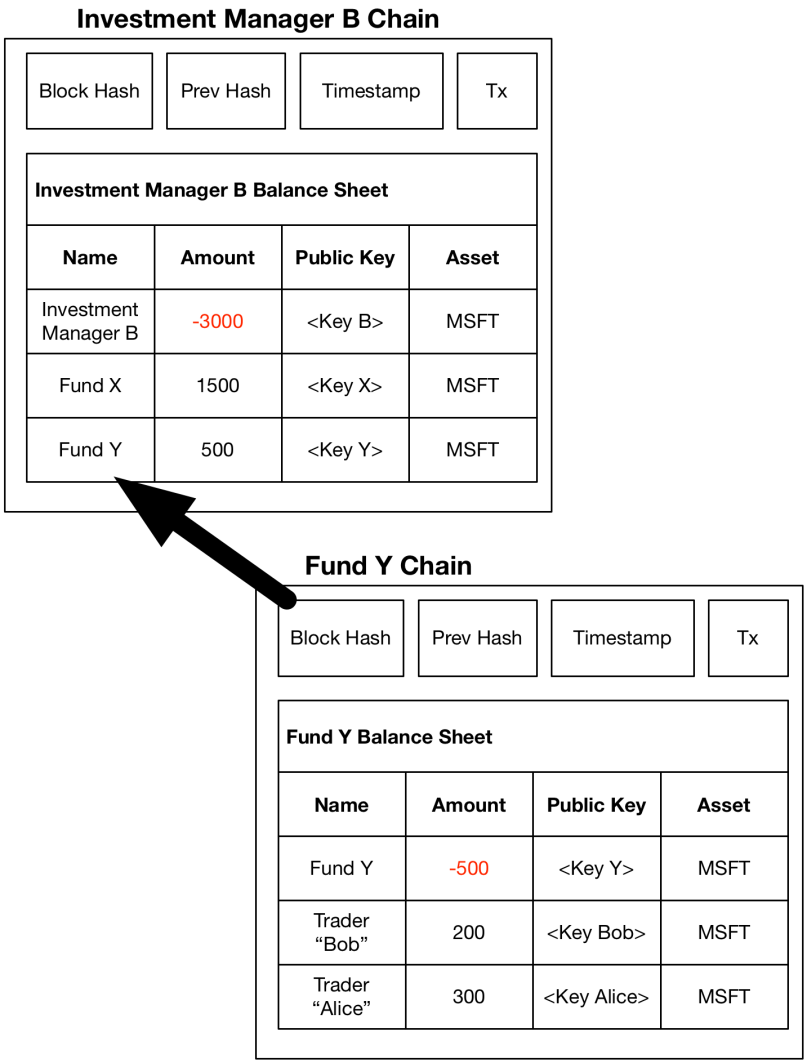

Fig. 9. An example of a linked blockchain. Fund Y has 500 MSFT on Investment Manager B's chain. A block on Fund Y's blockchain links to this positive balance. Fund Y is holding stock FBO Alice and Bob on its own blockchain.

13 Scaling Beyond a Block Maker

Because a Block Maker must process a hash function on a set of data, block creation is a difficult problem to parallelize. We recall that blockchains are usually implemented as a single long chain:

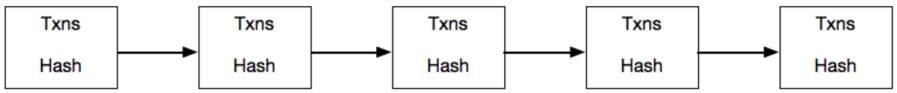

Fig. 10. A basic illustration of a block chain. Each block contains a set of transactions and a Merkle Root hash.

In order to scale, transactions can be sharded into many blocks. After transactions are sharded into blocks, a single block (metadata block) can be created from just the hash of the transaction blocks. Because much less data is contained in the metadata block, it can be generated faster than the transaction blocks.

This is similar to eDonkey2000 file hash system[6], which split all files into roughly 9MB parts and hashed each part.

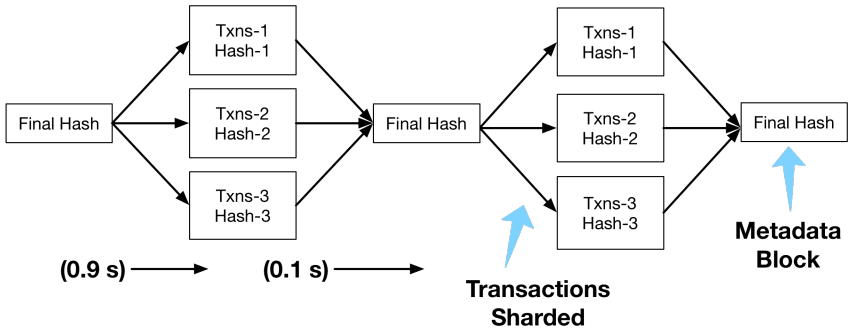

Fig. 11. A scalable Domus blockchain graph. The chain splits into many branches, and then contracts back to a single branch in a heartbeat manner.

14 Conclusion

With the Domus Blockchain, we have demonstrated that it is possible to design a blockchain which is capable of recording a high rate of transactions in a scalable

manner. Our blockchain is recorded in the form of a double-entry balance sheet which tracks credits and debits.

Using a blockchain ensures that data is immutable and append-only. Additionally, a Merkle Root ensures that the entire blockchain has not been altered or corrupted.

The system uses a hybrid approach in big data architecture. Rather adopting either a streaming or a batching framework, we designed a custom framework to use each technique where it is most appropriate. This allows us to optimize our service for desired low-latency and high throughput.

References

1. https://en.bitcoin.it/wiki/Scalability
2. https://bitcoin.org/bitcoin.pdf
3. Merkle, US 4,309,569, Method of Providing Digital Signatures
4. http://ed25519.cr.yp.to/
5. https://github.com/bitcoin/bips/blob/master/bip-0037.mediawiki
6. https://en.wikipedia.org/wiki/EDonkey2000