FOWcoin blockchain and Streembit decentralized P2P communication system for humans and machines.

Author: Tibor Z Pardi
Email: tzpardi@streembit.com

Co-authors: Matthew Cartwright, Dan Davies

Contents

The FOWcoin blockchain and smart contracts

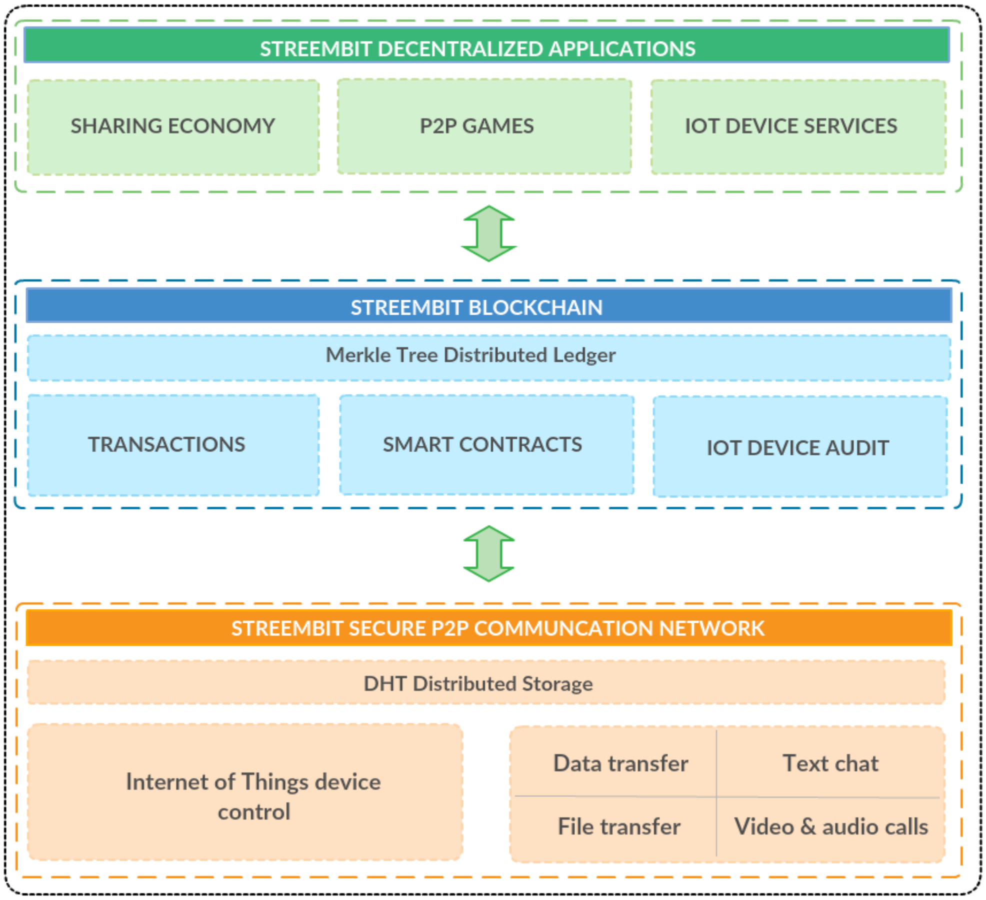
Preface

John Nash convincingly argued the need for 'Ideal Money'. Nash explained that money should have the function of a standard of measurement. Thus, it should become comparable to the watt, as a defined unit of power, or a degree of temperature. Money, as an efficient practical means of transferring utility, naturally links directly with the game theory idea of games having transferable utility. At the same time Nash admitted, although this scheme of money with ideal qualities would work well on the other hand, it would be politically difficult to arrive at the implementation of such a system. Governments with hugely different geopolitical interests cannot trust each other to create this ideal money or "World money" as many call the idea lately. We believe blockchains can solve this problem and can create the ideal money. But not current blockchains. Despite the large investments flowed into blockchain from the date of inception of Bitcoin and increasing interests in blockchain, not even a small regional bank or insurance company can use blockchain in its business process. We aim to solve the fundamental underlying issues of current blockchains by releasing the FOWcoin blockchain.

In step one; we will address a concrete use case with the FOWcoin blockchain: adult content payment processing. FOW Interactive, also known as "StudioFOW" is the artistic arm of Streembit Ltd. StudioFOW is the creator of the popular PC game "Subverse" which has recently launched the Early Access version of the game with encouraging results and feedback. This demonstrates a keen and loyal customer base and therefore an active market. However, StudioFOW is unable to find a payment processor that facilities payment for adult content as this sector increasingly faces bans and restrictions from mainstream providers. This provides us with the opportunity to demonstrate the capability of the FOWcoin blockchain by addressing real-world payment processing for StudioFOW, Subverse and wider appeal to the global video games industry use case.

The communication element

We believe a blockchain cannot exist without a robust, secure and decentralized communication framework. Why do we need a decentralised communication framework for a blockchain? Transactions require communication between users. To manage transactions a robust, secure authentication, access control and data transmission that guarantees data integrity must be in place. Ideally, this should be a decentralised communication framework. Thus, first we built Streembit, a secure P2P communication system that manages both human and machine (IoT) communication in a decentralised manner. In step two we released our blockchain that utilizes the Streembit communication framework.



The FOWcoin blockchain adds decentralized peer-to-peer transactions and smart contracts to the Streembit communication network. The addition of a blockchain to Streembit results in a decentralised permissionless platform capable of human and machine communication, financial transactions, and smart contracts. Every interaction with a person or machine, from video chats to complex smart contracts, are now possible on Streembit without any need or reliance on a central authority and via P2P processes.

An example of this all-inclusive functionality would be in the case of a bank issuing a bond. Firstly, the broker could interact with the customer using Streembit video or audio chat functions. The communication is P2P and it is therefore highly resistant to surveillance and industrial espionage. The parties could then exchange files using the file sharing function of Streembit and finalize the details of the transaction through the secure text chat of Streembit. Communication is the vital and underlying element of any financial transaction. The problem is, the communication must be private, and the data integrity must be guaranteed during the discussion. Streembit provides users with these functions. Following the process in the scenario above, the parties agreed on the deal, the bond could then be sold on the distributed ledger through a Streembit smart contract. Every step is secure and decentralised - so it is private to the parties and the communication is fully encrypted, and peer-to-peer.

Problems in current blockchains and protocols

Informally, the Byzantine Agreement is a communication protocol enabling a set of players, each of which potentially holding a different initial value, to agree on a single value. Agreement is reached by all honest players - that is, by those who scrupulously follow the protocol - even though a minority of the players may be malicious and can deviate from the protocol in an arbitrary and coordinated manner. The users of current blockchains aim to reach a consensus that is a Byzantine Agreement. We argue, the problem is, it is not possible to define and enforce protocol-level functions or rules in a scalable, efficient and secure manner in a relatively short time within the Byzantine Agreement consensus protocol. The relatively short time frame means it is impractical to manage financial transactions. Normally these transactions take place in fractions of a second in the financial world. At the same time, the completion of blockchain financial transactions takes minutes, often hours. Due to the weak Byzantine Agreement and consensus, current blockchains need a relatively long time to agree on the longest chain (i.e. accept new blocks that include transactions). This elapsed time between the creation of blocks slows down the transaction processing speed. Such slow transaction processing networks means blockchains currently aren't suitable for banking or other financial transactions (e.g. stock exchanges). The current 10-20 transactions per seconds worldwide on the Bitcoin blockchain, for example, or the 25-35 transactions per second of Ethereum is not practical for global financial transactions. A small regional bank must be able to process many thousands of transactions per seconds. The capability of current blockchains is far from the real-world transaction throughput that global businesses require.

The best minds of computer science attempt to create a Byzantine-tolerant system with very little success up to date. Since the Byzantine Agreement is so fragile on current blockchains, as well as the process of reaching consensus slows down the process, new consensus protocols have emerged. Such as Delegated Proof of Stake (DPoS) that use delegates and voting (i.e. replicate our trust in banks and financial service authorities to a few privileged individuals or delegates). Protocols like DPoS or blockchains such as Ethereum - that assigns power to a foundation - in fact implements a centralised system which any time can steal from users, create forks, ban users or refuse to forward transactions just to mention a few that happened on blockchains in the last few years.

Blockchain could certainly address many business requirements such as creating a distributed ledger that offers tamper-proof data storage or can create a fault-tolerant system so that no authority can alter the data making it immune to both dictatorship and fraud. But blockchain is only useful if it can scale. Due to using to the Byzantine Agreement consensus protocol, current blockchains cannot scale. Those blockchains that claim scalability are using delegates or trusted nodes, and therefore are not decentralized, they are not much different from any client-server system.

The assumption of current blockchains is that users will support the network and follow the protocol. However, a user or group of users can deviate from the Byzantine Agreement any time. In fact, a 51% attack or a fork could be in line with the economic or political interest of some users. For some users it could be that the optimal and risk-free strategy is actually to not keep the consensus. Thus, the Byzantine Agreement is fragile and the possibility of 51% attack is programmed into current blockchain. Only the goodwill and the economic interest of those users at that moment in time keep the consensus intact in current blockchains.

We argue a system that requires Byzantine Agreement is inefficient and can never achieve consensus in a timely manner, that is required for financial transactions.

We believe, to make blockchain suitable for business and social processes the Byzantine Agreement must be replaced with a more robust and scalable consensus process whilst improving security and transaction speed. Further sections of this white paper will explain how we aim to make the consensus more robust and intact, and how our blockchain incentivises users to agree on the longest chain and keep the consensus, so they do not deviate from the agreement that leads to the longest chain. By implementing a stronger consensus protocol, we can speed up block creation and as a direct consequence we can then make blockchain suitable for business and social processes and the wealth of use cases so often discussed in commercial and public sectors across the world.

Mining. Proof-of-Work (PoW)

Mining has several weaknesses when relied upon as a consensus protocol. The more computational power a miner has, the greater the chance he gets to create the block. This fact alone will lead to centralization of block creation to only the most powerful mining farms, which in turn creates an imbalance in the system and heavy weighting in distinct locations rather than a true distributed and fair network. This danger is compounded when miners pool together their hashing power and split the block reward. Recently one pool approached 50% of Bitcoins hashing power. This didn't result in a 51% attack because that would contradict the interests of the pool, it does however provide a glaringly obvious weak point. There are five pools that make up near 100% of Bitcoin's current computing power and 70% of Bitcoin mining power is currently in China, where it is widely recognized that there are strong links between the state and commercial interests. Their totalitarian government could easily seize that power and end any semblance of decentralization, effectively taking control of the currency. Centralization in Bitcoin also allows

for pools to only accept transactions that put forward a minimum fee, and because they have no competition they are free to raise the minimum as much as they would like so there is no regulation in this space which must be a concern to serious businesses and governments wishing to use blockchain.
Becoming a Bitcoin miner is no longer easy or fair. Miners using an ordinary desktop to attempt to generate a block stand to lose money. A system that makes participation in its consensus not easily accessible and fair is destined for centralization. Bitcoin has an unpredictable block time due to difficulty variance. If too many transactions are sent the network is slowed. This makes many use cases unfeasible as they rely on stability and timeliness.
Mining also takes a hefty toll on the environment by requiring an insane amount of electricity to secure the network. Ideally this computer power should be doing something more constructive while securing the network. In a time when many of the global governments are being forced by populations and the voting public to recognize damage to the environment the adoption of such power-hungry practices such as mining may not be congruent. For example, there are public articles noting the power consumption for BTC is the same as the total consumption of small nations such as the Republic of Ireland and larger nations such as Hungary and New Zealand - this is surely unsustainable and counter to public opinion.

Proof-of-Stake

Current implementations of PoS are relatively untested. Most lean towards centralization by giving more power to higher wallet balances. The biggest flaw is that there is "nothing-at-stake". There is little disincentive to attempt an attack. This opens the door to several attack vectors. The only risk is that you shake confidence in the protocol by successfully executing an attack, which would lower the value of your currency.

Developer dictatorship

If the developers of a cryptocurrency have any advantage over a user, the system is inherently flawed. Premines and instamines give a single developer, or group of developers, too much influence on the network and in most cases, creates a dictatorship.

How does Streembit solve these issues?

Our blockchain protocol does not favour higher computational power or wallet balances. Every user has an equal chance of participating in the creation of a block and the barrier to entry is very low. Instead of wasting electricity doing PoW computations, ordinary computer users can be collaborators to strengthen the Streembit network. The "nothing-at-stake" problem is solved with guardians that will lose their deposit if they attempt an attack. So-called 'Gouging' transaction processing charges will be difficult because of the competition allowed by the fairness of the protocol. All of this, in addition to using random cryptography functions to select collaborators, allows for a much more decentralized solution. The developers of the FOWcoin blockchain will have no advantage over any other user. The FOWcoin blockchain aims to restore the hope that the average hobbyist could fire up the client, donate a few clock-cycles and earn money from minting.

Does the Streembit Blockchain use 'trusted' or 'permissioned' nodes?

No, a FOWcoin blockchain node is not a permissioned/trusted node. If a blockchain is based on permissioned or trusted nodes it means the developer, business, organisation or the combination of these determines who can run a node - essentially some entity can provide permission or choose to exclude specific users. On the FOWcoin blockchain, however, anyone who fully complies with the publicly available blockchain rules can operate as a collaborator. Even the developers of the blockchain cannot refuse anybody to operate as a collaborator so long as they obey the rules and meet all the requirements of the blockchain - these are published and transparent. Similarly, a business, foundation or any other type of organisation that is related to the blockchain in any way cannot deny the participation from any collaborator who complies, in full, with the requirements and rules of the blockchain.

Proof of Collaboration consensus protocol (PoC)

The idea of a blockchain came with the invention of Bitcoin in 2008. A blockchain is a permissionless distributed ledger controlled by its stakeholders without the influence of a central authority. This revolutionary system allows for consensus to be reached among peers even if a minority of those peers send conflicting messages. The consensus mechanism used by Bitcoin is called Proof-of-Work (PoW). Another popular method is called Proof-of-Stake (PoS).

Streembit replaces the PoW and PoS consensus mechanisms with the Proof-of-Collaboration (PoC). Using the Proof-of-Collaboration protocol the users of FOWcoin blockchain agree on the longest chain and the set of valid blocks. Instead of seeking consensus via Byzantine Agreement, the FOWcoin blockchain uses a Cooperative Game Theory-based economic collaboration that assumes a binding contract between collaborators and relies on existing legal frameworks. This process requires accountability from block creators. Therefore, it dramatically reduces the risk of 51% attack as well as has the potential to exponentially increase the transaction throughput.

In the PoC consensus protocol, the initial values are candidate blocks, and the block in which agreement is reached is certified by the digital signatures of a set number of collaborators. That block is then propagated through the network, so that all users can add it to the blockchain. So, no matter how many different initial candidate blocks there are, the network will agree on only one of those blocks to add to the blockchain. This is how Proof-Collaboration achieves an agreement on the set of blocks and the longest chain.

In PoC, protocol users are called Collaborators. Collaborators are responsible for block creation through the proposal and validation of blocks. Once a randomly selected collaborator has proposed a block, other Collaborators are randomly chosen to verify the legitimacy of that block. Once the collaborators verify that no fraudulent transactions are included in the block, they sign the block with their digital signature. The Collaborators must execute a commonly agreed upon and adopted software application to produce blocks, register transactions, execute smart contracts, orchestrate communication between peers, and maintain the distributed ledger. The Collaborators are eligible to mint coins in exchange for the block creation.

Apart from transaction processing, Collaborators must perform supporting functions on the blockchain and Streembit network. These include, but are not limited to, providing REST API-based web services for light wallets and end-users, maintaining the blockchain explorer and expose interfaces to it, provide contact discovery on the Streembit network and store content on the Streembit network.

In PoC, an attacker with majority control over the stake of the system could attempt a 51% attack, but he would also need to convince the rest of the network that the fork is the valid blockchain. The honest minority will always ignore dishonest forks; therefore, the attacker's malicious fork would be disregarded and worthless. In PoC, the honest users stay on the same blockchain by executing the commonly agreed and adopted software application. This puts the attacker in an impossible situation, as he must execute malicious software to attempt the 51% attack.

The following premises and key assumptions are made:

a) There is no pre-mine in the FOWcoin blockchain.
b) To prevent developers from gaining early adopter advantages, no spendable minting reward can be granted for the first 10000 blocks. All users have sufficient time to set up the software to collaborate and gain minting rewards. Thus, the initial blockchain developers do not have an advantage over any other users.
c) Users agree upon what is the valid, commonly adopted, and honest software that produces transactions and blocks. The software must be open source. The source code of the software must be published in a public repository to allow users to peer review it and build the software application directly from its source.
d) To establish accountability within existing legal frameworks and make users accountable to the Court, only registered businesses such as Corporation, Ltd, LLC with verifiable and sufficient assets in their accounting books, and verifiable identity of the officers and directors can create blocks in the FOWcoin blockchain. The required asset value will be agreed by the users of the blockchain and the figure is published at the open-source repository.
e) The block creator users i.e. the businesses mentioned in point "d" must make a deposit to a smart contract, which will serve as a guarantee against any wrongdoing, malicious activity and fraud of the bock creator user. The deposit amount is published in the Blockchain Rules at the open-source repository.

f) The block creator users must make a deposit into a smart contract that allows the network to confiscate the deposit in case of any wrongdoing such as participating in a 51% attack or any fraudulent activity.
g) Any anonymous user can create blocks by participating in a Minting Consortium. These Minting Consortiums must be operated by a registered and accountable business and they are wholly accountable for those anonymous users within the consortium.
h) Users who have been honest and trustworthy in the collaboration process are listed in a Whitelist of Honest Collaborators (WHC). The WHC list identifies the honest collaborators by their public key.
i) Malicious users who have been trying to submit a bogus block or transaction are registered in a Black List by their public key.
j) To provide users with equal chance in block production, cryptography functions with true random number generators must be used to select collaborators for block production.
k) Any Streembit user can participate in PoC consensus by expressing their intent to collaborate and witness the block creation by submitting their public key to the list of collaborators. The witnessing users for each block creation are selected using true random number generators.
l) A generic mobile laptop or desktop computer, and an average internet connection (1 Mbps) are all that is required to participate as a witnessing collaborator.
m) Users with higher computation power, stronger computer hardware or a higher wallet balance, cannot have a better chance to become a witnessing collaborator. Also, their influence on block production cannot be directly proportional to their stake or wealth.

The FOWcoin blockchain and game theory principles

A robust blockchain protocol is secure against colluding minority groups. That means the protocol must incentivize miners to follow the protocol as prescribed. Researchers [15] indicate that the Bitcoin protocol is not incentive-compatible. Colluding miners could obtain revenues larger than their fair share. Such an attack can have significant consequences for Bitcoin: rational miners will prefer to join the selfish miners, and the colluding group will increase in size until it becomes a majority. At that point, the Bitcoin system ceases to be a decentralized currency. We have incorporated Cooperative Game Theory principles in our system design to create incentives for following the protocol.

Game Theory is concerned with decision-making in strategic settings, where you must factor the preferences and rational choices of other players into your decision to make the best choice for yourself. In many such settings, you’re on your own i.e. the choice you must make is yours and yours alone, because cooperation with other players is either impossible to implement or without any possible benefits. However, in some situations it is both possible and fruitful to cooperate with other players. If players can make binding agreements with each other, and there is some added value available by cooperating with others, then it can make sense for players to form coalitions that will work together to mutual advantage.

We believe a stable blockchain must be a coalition of a Cooperative Game Theory-based economic process that is governed by enforceable and binding contracts.

Currently this is not how blockchains work. Current blockchains are not cooperative game processes. For instance, the Prisoner’s Dilemma outlines that the most optimal strategy is to remain in solidarity. Similarly, in blockchain the most optimal strategy is to keep the Byzantine Agreement intact. Still, just like in the law enforcement scenario of Prisoner’s Dilemma the cooperation in blockchain is rather unstable. Most of the major successes in law enforcement prosecutions for gang and mafia related cases are directly helped by deflecting gang members so they give evidence and testimony against one another (e.g. divide and conquer). Similarly, in blockchains players can deviate from the optimal strategy at any time by causing losses for others and without any consequences. With this in mind it is no wonder that forks, protocol changes and 51% attacks are part of the current blockchain landscape.

Cooperative Game Theory suggests that a necessary condition for coalition formation is that the coalition is stable, in the sense that no members of the coalition have any incentive to walk away from it. [18] By applying Cooperative Game Theory we assume that groups of collaborators, called Coalitions, are the primary units of decision-making, and may therefore enforce cooperative behaviour. Due to a lack of a binding contract this is not how blockchain works currently. Current blockchains are unstable processes glued together by the momentary economic interest of users (mainly miners). One cannot implement a bank or in fact any business process. The base assumption of our PoC protocol is that the system operates as a cooperative game in which the users will form a coalition *N* to operate and secure the blockchain. This main assumption is that in PoC a coalition *N* is always formed. The theory is concerned with rational choice in decisions involving two or more interdependent decision makers. We assume that in our FOWcoin blockchain a rational agent has complete and consistent preferences among the available outcomes. The theory assigns numerical utilities to the outcomes in such a way that players who always choose utility-maximizing options (strategies or gambles) can be shown to be **acting in their own best interests and therefore to be instrumentally** rational. When accountability exists via binding contracts – which is how PoC works – the players will remain in cooperation and solidarity. In game theory, utilities are represented by payoffs, and the theory, as presented by von Neumann and Morgenstern, is primarily normative, in as much as its basic aim is to determine what strategies rational players should choose to maximize their payoffs. [19] We will outline methods such as accountability and deposit schemes in this white paper that greatly influence the possible outcomes of the blockchain collaboration, thus the collaboration framework of PoC influences the decision of the participants. In cooperative games, participants are not constrained to choose strategies independently but are able to negotiate coalitions based on binding and enforceable agreements with one another. The accountability and deposit scheme methods are indirect and yet they act as enforceable agreements between blockchain users. The grand coalition of all participants receives the whole payoff in the process of creating blocks and securing the network. In the PoC process – in conformance with Cooperative Game Theory - the core of the cooperation is an imputation in which every possible coalition of players receives at least as much as it could guarantee for itself by acting collectively.

Suppose the grand coalition *N* forms [23] and they then obtain the value *v(N).* The system must ensure how the value *v(N)* should be divided among the collaborators *N*. The *Shapley value* provides a principled way to do this. It proposes that each collaborator in set *N*, (so $i \in N$) should be given an amount ϕ_i that satisfies the following axioms:

- **Efficiency**. The total value *v(N*) should be distributed.
- **Dummy collaborator**. Collaborators who make no contribution should receive nothing.
- **Symmetry**. Collaborators who make the same contribution should receive the same.
- **Additivity**. The value should be additive over the set of all games.

Central to this axiom is the notion of a collaborator's contribution. We can measure a collaborator's contribution by simply looking at the value the collaborator adds to a coalition. Formally, the contribution of collaborator *i* to a coalition *C* is simply $v(C \cup \{i\}) - v(C)$, or the amount extra that *C* could obtain if they admitted collaborator *i* as a member. If this value is 0, then there is no benefit to be obtained. Given this definition, the symmetry axiom, for example, means that two **collaborators should receive the same value if they make the same contribution** to all coalitions. Thus, we can implement a blockchain-based collaboration on this model.

The system must ensure the core of the collaboration is non-empty in game theory terms. In computational terms, the design must support two requirements relating to the core:

- whether the core of a given cooperative game *(N, v)* is non-empty (whether the grand coalition is stable)
- whether a given payoff vector $x = (x_1, ..., x_n)$ is in the core of a given collaboration (i.e. game) *(N, v)*

No collaborator receives less than what he could get on his own $x_i \geq V(S), \forall S \subseteq N$. The solution concept can be calculated efficiently (i.e. in polynomial time with respect to the number of players |*N*|.

The solution concept known as the core suggests that a necessary condition for the formation of a coalition is that no subset of collaborators within the coalition have any incentive to deviate from it. A notation can describe this axiom. A payoff vector is a tuple of real numbers $x = (x_1, ..., x_n)$ that divides the value *v(N)* among all the players in *N*; thus x_i is the amount given to player *i* in this payoff vector. A coalition $C \subseteq N$ objects to the payoff vector *x* if they could collectively earn more than *x* allocates them, which formula is $v(C) > \sum_{i \in C} x_i$. If this condition is satisfied, then the payoff vector *x* could not be implemented, because *C* would not accept it; they would do better to work on their own, and could divide the surplus obtained among themselves. Now, the core of a game *(N, v)* is the set of payoff vectors to which no coalition has any objection in the sense we just described. If the core is empty, then this means that the coalition consisting of all agents cannot form: there is no way of distributing the value *v(N)* to which there are no objections. Conversely, if the core of the game is non-empty, then there is some way of distributing the value *v(N)* to the players in *N* such that no coalition can reasonably object, in the sense that no coalition could do any better.

The formula for this

$$c(V) = \{x \in R^N : \sum_{i\in N} x_i = v(N); \sum_{i\in S} x_i \geq V(S), \forall S \subseteq N\}$$

Where the objective is to allocate the rewards $v(N)$ among the participants in a fair way. The solution concept known as the core suggests that a necessary condition for the formation of a coalition is that no subset of players within the coalition has any incentive to deviate from it. A solution concept is a vector $x \in R^N$ that represents the allocation to each player. Efficiency is addressed by exactly splitting the total value $\sum_{i\in N} x_i = v(N)$. The solution concept exists for any transaction and process *V*. The solution concept is unique for any transaction and process *V*.

In this system, the core is the set of imputations under which no coalition has a value greater than the sum of its members' payoffs. Therefore, no coalition has any incentive to leave the grand coalition and receive a larger payoff.

To use Cooperative Game Theory concepts in the context of blockchain, the practical implementation must seek representations that strike a practical balance among compactness, representational power, and computational tractability. This system selects the Weighted-Voting Games to represent the collaboration based on cooperative gaming theory. A weighted-voting game is a type of simple cooperative game: a game where every coalition either gets the value *0* (they are "losing") or *1* ("winning"). In a weighted-voting game, each player i ∈ N is associated with a weight "*wi*". The overall game has a quota, given by a real number *q*. A coalition C ⊆ N is then said to be winning if the sum of their weights meets or exceeds the quota and losing otherwise:

$$\text{v(C)} = \begin{cases} 1\ if\ \ \sum_{i\in C} w_i \geq q \\ 0\ otherwise \end{cases}$$

Weighted-voting games have a compact representation. The formula just needs to represent the weights and overall quota. Checking if an outcome is in the core for weighted-voting games is computationally easy.
In the blockchain, the players are the users that form consensus. The coalitions are the group of users that compete for the longest chain. The winners (i.e. "*1*" in the formula) are the players who form the **longest chain**. (The longest chain means the set of blocks that are adopted by the most users). The more users join the longest chain, the larger the weight of that coalition becomes, thus it can satisfy the weighted-voting games formula. The weight of a user corresponds to how many total users adopted the longest chain. The quota is the number of users required for a consensus to agree on the longest chain. In such settings, the Shapley value has an interesting interpretation: it measures how much power each coalition has, and its ability to influence the overall decision in forming the longest chain.

We generalize the extended Shapley Value for the Blockchain with "h" ordered alternatives in the input level. [15]
In contrast to the classical Shapley Value, in this Blockchain process we distinguish between the block creator nodes who do not enter the process and nodes who enter the process but contribute nothing, i.e. their entry into the Blockchain process leaves total product unchanged. We implement this distinction to allow for Blockchain to include inactive nodes that are ready to produce blocks any time. In practice, the blocks created by nodes i.e the production of nodes fluctuate with the level of production of the whole Blockchain. In slow periods, when there are no transactions the nodes may not work but are still considered as part of the Blockchain.
We begin by assuming all nodes start with an initial labor contribution of h = −1. Therefore, we set

$$\overline{Z}_0 = (-1, -1, -1, \ldots, -1)$$

as the initial labor allocation in our model. At this stage, the nodes have not entered the Blockchain with effective production of blocks. We define |Z|∈R to be the total number of nodes who have entered the Blockchain in labor (block creation) allocation Z. This includes currently inactive nodes who enter the Blockchain but do not contribute to the production process, as they wait for the random allocation of block creation right.
For any predetermined labor allocation ¯ Z, the extended Shapley Value for Organizations is given by:

$$\phi_i(f(\bar{Z})) = \sum_{\substack{Z \rhd \bar{Z} \\ z_i=-1}} \frac{(|Z|)!(n-|Z|-1)!}{n!}[f(z+e_i[z_i+1+r(z_i)])-f(z)] \quad (1)$$

$$Z \rhd \bar{Z} \quad \text{if} \quad \begin{cases} z_i = \bar{z}_i & \text{if} \quad z_i > -1 \\ z_i = -1 & \text{otherwise} \end{cases}$$

$$|Z| \in \mathbb{R}$$
$$i = 1, 2, ..., n$$

The value of |Z| can be estimated using the steps outlined at the beginning of this paragraph.

We aim to provide more details for the above design in the subsequent versions of this white paper.

Substituting the Byzantine Agreement with Cooperative Game Theory economics

The greatest challenge for any permissionless decentralized system, especially one dealing with transactions, is keeping the peers responsible for maintaining the system honest. This is usually done by providing incentives to follow the protocol, and thus making attacks uneconomical. The gain from acting honestly needs to be greater than the expected gain of malicious activity.
Another important layer of mitigation is keeping participation in the consensus protocol fair. Fairness ensures that everyone can participate in the PoC consensus and prevents the concentration of power.

As noted above, Streembit keeps the barrier to participation very low by not allowing users with greater computational power, or a larger wallet balance, any greater chance of influence on block production. Streembit also incentivizes honest activity, but **also introduces the action of penalizing dishonest collaborators**.

The PoC consensus protocol achieves this and incentivizes honest collaboration through the following methods:

1) Deposit scheme
2) Accountability

Blocks can be proposed by nodes that use either the deposit scheme or the accountability method. **Randomly selected nodes must be involved in the block production.** The deposit scheme or accountability method cannot produce a block without randomly selected anonymous collaborators. **Anonymous collaborators (anonymous mint consortium members) do not have to reveal their personal information.** This allows collaborators who don't want to reveal personal information to participate in PoC and earn money from block production. The anonymous collaborators can create blocks by joining to a **Mint Consortium**. The Mint Consortium is responsible for the honesty of transactions. That means, the Mint Consortiums will introduce security measures and compliance policies to filter out malicious users.

PoC goes to great lengths to mitigate attacks by adding economic and legal risk to a successful attack. However, even if an attacker manages to create a malicious block, that block would be ignored by the honest participants. In PoC, the honest users stay on the same blockchain by executing the commonly agreed and adopted software application. This puts the attacker in a difficult situation as he must execute malicious software to attempt the attack. Not only would his malicious fork be disregarded, but the attacker would be penalized and potentially face legal consequences in accordance with the rules of the blockchain and the smart contract deposit.

Secure the consensus with deposit schemes

The PoC consensus protocol implements a deposit scheme to disincentivize attacks on the Streembit consensus protocol. Essentially, the block creator node will suffer a significant financial loss, legal consequences and will be the subject of criminal proceedings in event of it acting dishonestly. The protocol aims to mitigate the "Nothing-at-Stake" issue by imposing a financial penalty on the fraudulent collaborators. The financial loss makes the attack on the blockchain uneconomical. In practice, in order to get permission to create a block, the block creator node N_1 must deposit at least two times more money D_1 than the total transaction value of the block TV_1 , thus $D_1 > (TV_1 * 2)$. The deposit is locked and will be confiscated by burning it in case of the node acting dishonestly. In case of large volume transactions that exceed the deposited amount the full deposit is locked. Once the transactions Tx are collected, the node signs S_1 the transactions. The process is governed by a smart contract SC_1. The money is deposited to a smart contract address A_1. The smart contract monitors the status of the block B_1. Once the block is matured enough, after at least 300 notifications Nx the multi signatures on the contracts are activated and the deposited amount is returned to the block creator node. At the same time the block creator node becomes eligible to collect reward R_1, which is the right to mint coins FC_1 in exchange for the block creation and support of the blockchain. If the node signs a bogus transaction or attempts to perform a double spend, the deposited amount D_1 will be lost by burning B, the amount in the deposit address. The formula for the smart contact is the following

$$SC_1(((R_1 (TF_1 + FC_1)) = (N_1 \text{->}(D_1 > (TV_1 * 2)) \&\& (S_1 \text{->} Tx) \&\& (Nx >= 300)) \&\& (A_1 \text{->} N_1)) || (A_1 \text{->} B))$$

The PoC consensus protocol incentivizes keeping the Byzantine Agreement with this simple, but effective deposit scheme. Block creator nodes can generate income by minting coin (this right is derived from production of legitimate blocks). Unlike in Proof-of-Stake protocols, the process does not favour large wallets, or provide more wealthy users with more block producing options. All nodes, even nodes with little deposit power, can take part in block creation by signing low volume transactions.

Blocks cannot be created solely by depositing nodes, there is an additional layer of verification that is not performed by depositing nodes. The purpose of this is to ensure the validity of the transactions. The reward R_1, for block creation, is always shared with all users that assist in creating the block.

Secure the consensus with the Accountability Method

With the Accountability Method, the block creators reveal their personal or business details, express their interest in earning money by producing blocks, and assume responsibility for their actions. Consequences for being dishonest will disincentivize deviation from the Byzantine Agreement. Once the identities of block creators are known, they can be held accountable in the event of any attempt of malicious activity.

With PKI cryptography, using PKI digital signatures and PKI certificates, we have the means to implement individual and organizational accountability on the blockchain. Business, banks, and government departments that use the blockchain must comply with laws and regulations. Knowing the identity of the block creators, law enforcement agencies can seek a court order for identity disclosure when a legitimate court deems it necessary for the protection of public safety. Accountability in the PoC consensus protocol ensures that due process prevails even in jurisdictions that are not known for adherence to due process. Having accountability makes cryptocurrency much more attractive and accessible to businesses, banks, and governments. These are real world use cases that haven't been tapped by cryptocurrency projects. Even an attempted attack could result in legal consequences. These consequences provide another layer of attack mitigation. PoC benefits from this mitigation but does not rely at all on legal action. We argue, accountability of the collaborators is simply necessary to bring cryptocurrency to a higher level of adoption.

In practice, a business or individual that runs a Streembit node to produce blocks can reveal its identity by using a PKI digital certificate. In most cases, it will be a X.509 digital certificate issued by a Certificate Authority (CA) organisation. This is in fact already in practice by most banks, businesses, and government entities that require the verification of identity through a digital certificate. To make the blockchain usable for businesses, the PoC consensus protocol allows nodes to identify themselves using X.509 digital certificates. The X.509 digital certificate of the node is revealed and stored in the Streembit DHT as a key-value pair, so any Streembit user can look up the details and identify the block producers.

PoC also requires that the officers of the collaborating businesses take legal responsibility for their actions. The accountability of the collaborating business will be acknowledged by the officers of the business. The designated officer such as director, CEO, CIO, CFO, etc. must sign the binding contract of the Cooperative Game Theory process. The signature must be submitted via a recognized legal and technology framework such as the Estonian E-Residency scheme. The accepted schemes will be reviewed regularly and published in the Streembit public software repository.

At the beginning of the blockchain, the collaborator businesses must submit and use an extended SSL certificate that sufficiently verifies the business details. The binding contract must be signed with the private key of the extended SSL certificate. Later this requirement must be reviewed and using a PKI digital certificate won't be mandatory in PoC. We recognize that CA organizations introduce centralization to the blockchain. Relying on centralized CA organizations means that the blockchain is not permissionless anymore. In remedy of this PoC allows Streembit nodes to identify themselves by other means, such as by revealing company details (like the company registration number), or the personal details of the node owner. PoC defines these details as "*Accountability Info".* The Accountability Info of the block creator node is also revealed and stored in the Streembit DHT as a key-value pair.

The coins

A blockchain cannot operate without a crypto currency, a coin. The coins on the FOWcoin blockchain are called FOWCoin and Streembit Stable Coin. FOWCoin (FOW) coin is a value store. The Streembit Stable Coin (SSC) is a "stablecoin" that is pegged directly to FOWCoin and via this peg indirectly to US dollar, USD.

The FOWCoin coin (FOW)

The FOWCoin cryptocurrency that has the currency acronym "FOW" in many ways is similar to existing crypto currencies such as Bitcoin or Ethereum. The coins are created during the block creation. Instead of calling this process mining, we call this coin creation "minting" to reflect the less resource required by the process. The trustworthy users who provide service on the blockchain such as processing transactions are rewarded with FOWCoin coins. The collaborators are eligible to mint coins in exchange for the block creation. The Blockchain rules determine the amount of minted coins with a block. These rules are published in the open source repository of the blockchain. All nodes must operate and validate the amount of minted coins based on these rules.

Just like Bitcoin, FOWCoin coins can be sent to blockchain addresses, exchanged between addresses via blockchain transactions, can be traded at crypto currency exchanges, and kept in the wallet.

The Stablecoin of FOWcoin blockchain – SSC

The high volatility of cryptocurrencies is one real barrier to the mass adoption of blockchain for commercial use in real world transactions. The volatile nature of cryptocurrencies can cause contingent losses for merchants. [24] Such volatility means that merchants cannot pay their bills, pay suppliers, repay loans, or meet payrolls. Therefore, merchants cannot sustain their trade on cryptocurrency. Stable money is a necessity for a successful business, and cryptocurrency currently is inherently unstable. To solve the volatility of cryptocurrencies more and more developers have launched or planned for cryptocurrencies which are pegged to a precious metal, to the dollar, or to other currencies which might provide more stability than what digital currencies normally enjoy. [25] The FOWcoin blockchain achieves this value stabilization by issuing **Streembit Stable Coin (SSC)** that value is guaranteed via the collateralized buyback peg method. The value of 1 (one) SSC is always 1 (one) US dollar. The currency acronym of Streembit Stable Coin is **SSC**.

Each SSC is associated with a unique identifier, a GUID. Users who own **SSC** will be able to send **Streembit Stable Coins** to any other users on the FOWcoin blockchain.

Collateralized Buyback Peg

The FOWcoin blockchain allows - in the same way as the minting of FOWCoin coin – that the accountable block creator nodes can mint Streembit Stable Coin. By employing a peg mechanism, one Streembit Stable Coin will be worth one US dollar. The value of the Streembit Stable Coin is based on the assumptions that;

1) Customers who want to buy goods and services using the Streembit Stable Coin (SSC) will buy coins at one US dollar exchange rate from the minting nodes or from their subsidiaries.
2) The merchants will deliver one US dollar value of goods or services for one Streembit Stable Coin (SSC). The customers/partners of the merchant have no incentive to get less value for the coin than what they paid for it.
3) The merchants can exchange one Streembit Stable Coin (SSC) for one US dollar. The merchant has no incentive to exchange one Streembit Stable Coin (SSC) for less than one US dollar, the value of goods or services for what the Streembit Stable Coin (SSC) was acquired.

Such value guarantee is achieved via the Collateralized Buyback Peg method. This method attaches a collateral to each minted Streembit Stable Coin (SSC). The collateral is always freshly minted FOWCoin coins "FOW". This is the pegging. The method ensures there is a policy enforced to buy back the SSC coin from anyone who holds the currency and wishes to sell it. The minting node cannot spend the pegged FOW coin until the node operator buys back the circulating SSC coin. Since the collateral is significantly more than the value of SSC coin, and the pegged FOW coin can be sold for significantly more than the value of the SSC coin, to own the deposited FOW coins the minting nodes will buy back the SSC coin. Apart from the financial incentives, accountability enforcement (mainly at the court of law and by the prospect of revoking of the accountable node status) has primary role in this method.

The steps of the Collateralized Buyback Peg method are:

- The block creator node can mint SSC coin only if at least 10 US dollar value FOW coin per SSC was deposited (pegged) to a smart contract address. The smart contract must
  I. Record the minted amount of SSC coins.
  II. Record the current FOWCoin exchange rate.
  III. Calculate the total collateral by recording the amount of pegged FOW coin multiplied by the current exchange rate to ensure each minted SSC coin is collateralized by at least 10.00 USD value FOW coin.
  IV. Facilitate the sale of SSC coin to interested buyers.
  V. Facilitate the buyback of the SSC coin, typically from merchants who sell goods and services for SSC coin.
  VI. Ensure that the collateral cannot be spent before the smart contract is closed.
  VII. Release the collateral FOW coin once the transaction Streembit upon buying back the SSC coin is complete and must close the smart contract.
- The minting node must sell SSC coin to anyone who pay 1 USD for one SSC coin.
- The minting node must charge one USD for 1 SSC coin.
- The minting node must include an unconditional buyback clause in the smart contract.
- Once a seller triggers the buyback clause, the minting node must buy back at least as much SSC coin than it has minted from the seller.
- The minting node must not refuse to serve any seller for any reasons.
- The minting node must provide or use a working currency exchange platform and FIAT payment processing to accept USD for the SSC coin and pay USD for the SSC coin during the sell and buyback process, respectively.
- To incentivise the minting, handling and healthy circulation of the stablecoin SSC coin the minting node acquires the right to mint 2% value FOW coin of the minted and handled SSC coin.
- The FOWCoin exchange rate is the previous day average USD-FOW rate recorded in the opens source repository.
- Accountable nodes of the blockchain must validate all steps of the method and take actions against the minting node in case it does not obey these rules. The actions include revoking the accountable node status, take law enforcement actions and initiate court procedures to recover financial losses.

The financial gain from acquiring FOW coins is significant, as well as the potential penalties and financial loss are so large that the minting node has no incentive to act dishonestly.

Example of a life cycle and trade of SSC coins:

The accountable node mints 500 FOW coins during the block creation process. The current exchange rate of FOW is USD 4.00. The minting node decides to mint 100 SSC coins that requires 100.00 x USD 10.00 = USD 1,000.00 collateral. The minting node must peg (deposit) 1,000.00 USD worth of FOW coins to the currency address of the smart contract (250.00 x USD 4.00 current exchange rate). Following the minting process, the SSC coin is transferred to a valid currency address. Customer “A” buys 50.00 SSC coins from the minting node. The minting node sends the 50.00 SSC coin to Customer “A”. Customer “B” buys 50.00 SSC coins from the minting node. The minting node sends the 50.00 SSC coin to Customer “B”. Both customer “A” and “B” buy USD 50.00 worth of services from Merchant “A”. Merchant “A” receives 100.00 SSC coins from the customers. Merchant “A” triggers the buyback clause on the relevant smart contract. The minting accountable node buybacks the 50.00 SSC coins. Upon completing the buyback the collateral FOW is released by the smart contract. The minting node acquires the right to get access to the 2% value of handling fee FOW coins. This handling fee reward can be minted and withdraw in subsequent block creations in FOW coins.

How to become an Accountable Collaborator, and management of the list of Accountable Collaborators

The accountability-based block production is the collaboration of Accountable Nodes. In this collaboration the users of FOWcoin blockchain must recognize the accountable nodes. To do this the system creates a list of Accountable Nodes and maintains this list permanently. These actions require blockchain-wide, clear and transparent rules that define:

1) The attributes of an Accountable Node.
2) The attributes of a binding contract
3) The attributes of the list of Accountable Nodes.
4) Methods and functions to become an Accountable Node.
5) Verification methods to validate information about Accountable Nodes.
6) The management of the Accountable Nodes List.

The attributes of an Accountable Node and the requirements for Accountable Nodes.

To operate a blockchain within a Cooperative Game Theory economic process requires that the parties of the process act honestly, refrain from any malicious activities and cause no financial losses to each other. Within our Cooperative Game Theory-based economic process, such honest, collaborative and law binding attitude is enforced with binding contracts. The parties keep each other accountable via a legal agreement that is manifested in a binding contract. Accountability means that the parties accept the contract will be enforced in case of wrongdoing such as double spending or collusion for a fraudulent transaction. A node that accepts accountability and willingness to enter into a binding contract is called an “Accountable Node”. The attributes of the Accountable Node are in the first phase of the FOWcoin blockchain (first phase means the first 12 months from the date of inception of the blockchain):

- Name, registration number, address of the collaborating business.
- Proof of business assets such as latest accounts.
- PKI certificate of the business and signature the binding contract using the certificate’s private key.
- Name of delegated officers and digital signatures of the officers

The management of Accountable Nodes List

An elementary business requirement is that this list must be updated frequently. That means, by following the rules, blockchain users must be able to remove and add Accountable Nodes. We propose a flexible and multi-option scheme for Accountable Node management. This means that the validation of accountability-related information can be done via different methods and using all kinds of different functions. The primary place to maintain this information is the Streembit decentralized Kademlia network.

What is "claim validation"?

The nodes that want to be accepted by other blockchain users will present information about their trustworthiness. The node that wants to be accountable puts itself forward to be considered by other users. The node will do this by presenting important information such as company registration number or the latest accounts that proves the assets of the company. This is the "claim about accountability". This claim must then be validated.

This validation can be in different forms. The first method is at the inception of the FOWcoin blockchain that the developers of the system will validate the claims. This is performed via the Github repository where the validated details of the Accountable Node is published, so the system can look-up and compare the claimed details against the validated details. This Github-based publishing and validation is transparent, and the developers must perform the validation upon the Accountable Node request. To avoid undesired developer power, other methods should emerge to validate the claims about accountability. These could include:

- Streembit Github Publishing (first method at the inception of the blockchain)
- Accountability service providers (e.g. The Authenticity Institute)
- Government authorities. These are normally part of a regulatory framework (e.g. UK Financial Service Authority)
- User initiatives such as IRC channels or Facebook lists
- Any type of agreement between users e.g. professional association or alumni user groups

We expect all kinds and innovative forms of validation, but the validation method must comply with the rules and the validation must provide information that related to the rules. For example, in the first year of the FOWcoin blockchain only legally registered companies can be Accountable Nodes. Thus, the claim validation methods must establish and prove that the Accountable Node is indeed a company. The important rule is that there can be many types of validation, and nobody can dictate what users should accept. That means the validation process is permissionless – no central authority can dictate what the validation method is. The users will decide which validation method they accept. Even as little as two users can agree on a validation method. Of course, banks most likely will accept a validation that is based on a known and trusted regulatory framework and backed by financial service authorities. While users who have become disenfranchised with the banks and the establishment will prefer Accountable Nodes that are validated by blockchain community-based processes.

Since accountability is the cornerstone of the FOWcoin blockchain, there are rules.

The base premise of Accountable Node management is that it **must remain decentralised and permissionless**. That means:

a) A central authority cannot decide (deny or permit) who can be an Accountable Node
b) Users can define their methods and functions to validate the list of Accountable Nodes. There can be various methods to validate of the claims of an Accountable Node prior to listing.
c) Anyone who complies with the rules (so in first year of the blockchain registered companies with suitable assets) can participate as an Accountable Node and there is no user, entity or authority that can stop other users to become an Accountable Node.

So how is this decentralised and permissionless? Any entity that complies with the rules has the right to participate and there is no entity that has the power or authority to give or deny permission to participate. Just like in Bitcoin where anyone can mine coins, in the FOWcoin blockchain anyone can be an Accountable Node if they obey the rules of the blockchain. Once an entity (at the beginning of the blockchain this can only be a registered business) complies with the rules of the blockchain it can be an Accountable Node.

In terms of point "b", typically a node will present some information which indicates accountability. The information can be many types including company registration number, company assets or a list of directors. This information must be validated. It is important that such validation cannot depend on one centralized entity and the users of the FOWcoin blockchain are able to validate the accountability information as they wish and in the manner of their choosing. The validation of the information cannot be the privilege of the developers or a government office.

Further to point "c", while anybody can put forward itself as an Accountable Node, it doesn't mean all listed Accountable Nodes will be trusted by other users of the blockchain system. Accountability is the composite of many attributes: the history of the entity, financial status, volume of assets, level of insurance policies, size of operation, amount of deposit, among others. Whether or not a node will ultimately grow into an established Accountable Node is entirely up to the users. For instance, it is very possible and most likely that a bank will only trust other banks so establishing accountability for some users will be sector-based while other users can accept Accountable Nodes based on different criteria. User will make their individual choices as to how they should validate and what their individual acceptance criteria will be.

Accountable Node requirements in the first year of the FOWcoin blockchain

The system must maintain the list of Accountable Nodes to facilitate accountability-based block production. During the block production, the system selects the participating Accountable Nodes from this list. To be listed, the Accountable Nodes must publish their personal or organizational details in the Streembit public source code repository by submitting a pull request. If the pull request passes the unit test, complies with the coding convention, and the published personal or organization details can be verified, the maintainers of the Streembit source repository cannot refuse accepting the pull request. Correct and verifiable pull requests must be accepted within 14 days of the submission of the pull request. This rule ensures fairness and that everyone can participate as an Accountable Node. The coding conventions and requirements of the pull request will be published in the readme section of the repository at https://github.com/Streembit/accountability. The Streembit Foundation, a non-profit, community interest organisation will oversee this to make the process fair and transparent.

From the second year of operation, the requirements can change by learning from the first-year of operation and from collecting feedback from the users and by assessing innovating new methods. The blockchain community will decide what is to become the corner stone of the FOWcoin blockchain. The system must maintain the list of Accountable Nodes to facilitate accountability-based block production.

Implementation & design details

While the underlying design of FOWcoin blockchain is fundamentally different from any existing blockchains, to enable easy exchange integration the FOWcoin blockchain follows Bitcoin design conventions where it is possible. For instance, the length of FOWcoin blockchain address is same as the Bitcoin address length or the set of RPC command is very similar to Bitcoin RPC commands.

Address

FOWcoin blockchain addresses are case sensitive. The address is a 160-bit hash of the public portion of a public/private ECDSA keypair.

The Wallet

The FOWcoin blockchain wallet follows the workflow and approach of existing blockchains. The wallet stores the addresses, the PKI key pair of the addresses and the transaction history of the wallet. The wallet provides cryptocurrency functions such as 'send' and 'receive' money.
The Streembit Blockchain wallet is a BIP32 compliant deterministic HD wallet.

Encryption of the wallet

The wallet is secured with the standard Streembit user credentials.
The wallet encryption starts with securing the passphrase. The passphrase must be a complex string. Please refer to the Streembit security section that explains the strength of the password. The passphrase secures the wallet data by deriving an AES 256-bit encryption key from the passphrase and then encrypt the wallet data using the derived encryption key. In step one a 32 bytes salt is created using strong random data generator. Next a cryptography key is derived using PKBDF2 with the salt, SHA-512 hashing and at least 10,000 iterations. Finally the system encrypts the wallet with this PKBDF2 based cryptography key using an AES 256-bit encryption.

If you lose your wallet entirely, all of your coins are lost and can never be recovered!

The seed of the master private key is encoded into random words from a dictionary, which can then be written down. If your hard drive crashes you can find the paper with the mnemonic phrase and restore the entire wallet. It is worthwhile to keep copies in several locations so that even if your home burns down and nothing remains you can still recover the bitcoins. As with any secure information, you should always have security policy to enable safe retrieval without providing opportunities for others to steal your passcodes and credentials.
Possession of your crypto coins comes from ensuring your ability to keep the private keys under your exclusive control at all times. Any malware or hackers who learn what your private keys are can create a valid bitcoin transaction sending your coins to themselves, effectively stealing your bitcoins. The average person's computer is usually vulnerable to malware so that must be taken into account when deciding on storage solutions.

Block production

The block production of PoC significantly differs from existing blockchain systems such as PoW and PoS consensuses.

Manage the UTXO set

We believe the input/output based bookkeeping that Bitcoin has introduced is a sufficient method. The FOWcoin blockchain also tracks the unspent transactions (UTXO). To optimize the UTXO management of current blockchain systems, the majority of users in this protocol will simply store the set of currently unspent transaction outputs, rather than the whole blockchain history. The size of such UTXO set is significantly smaller than the size of all blocks. This enables more efficient operation, requires fewer resources and lowers the risk of centralization. The UTXO set easily fits in memory which allows fast data lookup. Its growth probably tracks the size of the entire blockchain. The system will use advanced data structures like Patricia trees to ensure that data encoding is efficient and that lookups are very fast.

The genesis block

A genesis block is the first block of the blockchain. The network assigns it the block number 1. The genesis block is hardcoded into the software. It does not reference a previous block and it produces an unspendable subsidy.

The first 10 blocks

Following the genesis block, the system publishes the first 10 blocks that include system information, the rules of the blockchain, algorithms used by the blockchain, smart contract templates, etc. This step establishes the blockchain.

Producing subsequent blocks

Step 1 – The right to create a block.

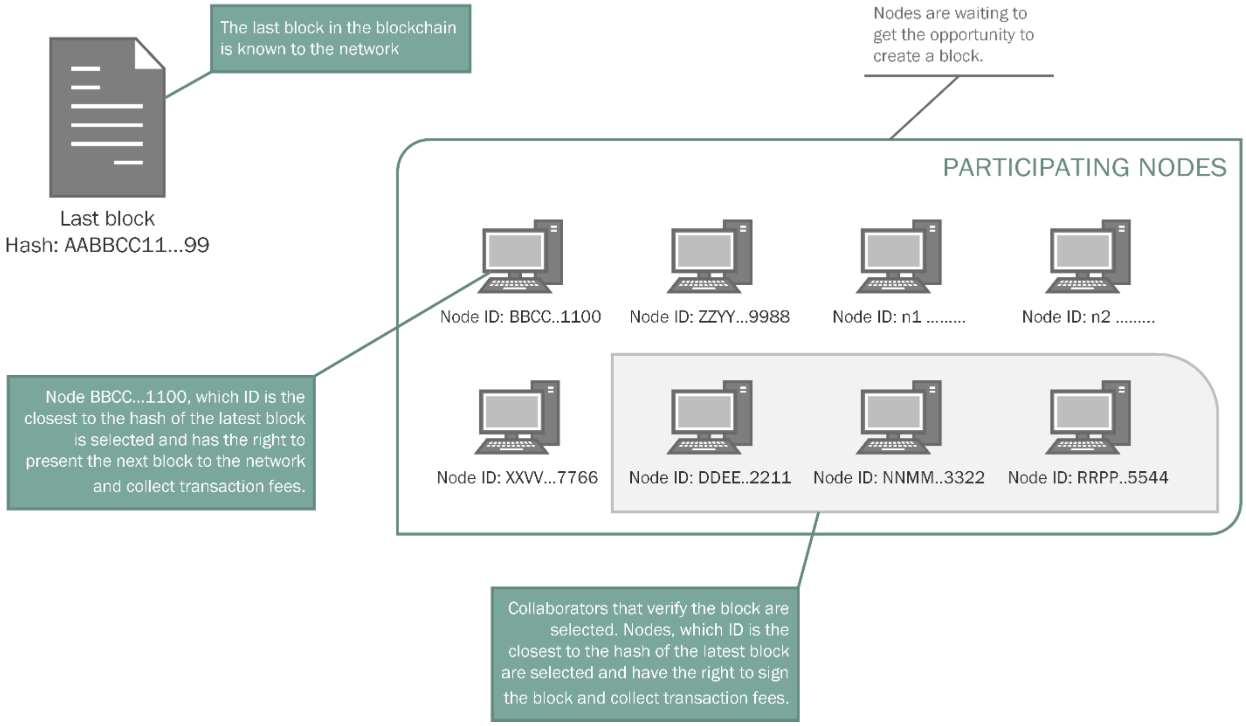
The collaborators register their intent to produce and validate a block. Any Streembit UI user or CLI (command line interface) node can join the intent list at any time. This intent list is stored in the DHT. To maintain the list of intent would be challenging for a permissionless system, but the Streembit DHT is a particularly suitable medium to maintain such a list. The block creator is chosen based on the last produced block hash. The system adds a nonce that has been predetermined in the software, such as 1, to the previous block hash and hashing the result. The collaborator in the white-list with the closest NodeID to this hash will be the block creator. The closest node is determined using the *Levenshtein distance* and *Wagner–Fischer dynamic programming algorithms*. The lowest score means closest distance. The node with a lowest *Levenshtein score* obtains the right to create the block. In case of unlikely event of identical scores between multiple nodes, the node with the oldest operational history has the right to create the block.

The block creator can use either the Accountability or Deposit scheme methods.

Using similar methods, by determining the closest NodeID the collaborators are selected as well. The collaborators validate and sign the transactions and sign the new block. The collaborators can use either the Deposit Scheme or Accountability methods or can be an anonym node. The anonym nodes operate without revealing their identity or depositing.

The hash of the previous block is used for selecting the block producer and collaborators from the intent list in a non-deterministic manner to keep block production fair. It does this by adding a nonce to the end of the previous block's hash and selecting the collaborator whose DHT NodeID is closest, where the nonce increases by 1 for each collaborator. This results in an unpredictable and fair selection of collaborators. The user makeup of the collaborators will be at least 1 accountable collaborator, 1 depositing node and 1 anonymous collaborator. However, the block creator must communicate and form consensus with at least 6 eligible collaborators to ensure the creation of the block, to have enough collaborator in the process in case some of the eligible collaborators is offline or not responsive.

Streembit blockchain. Block creation Step No. 1 – select the block creator and collaborators



The last block in the blockchain is known to the network

Nodes are waiting to get the opportunity to create a block.

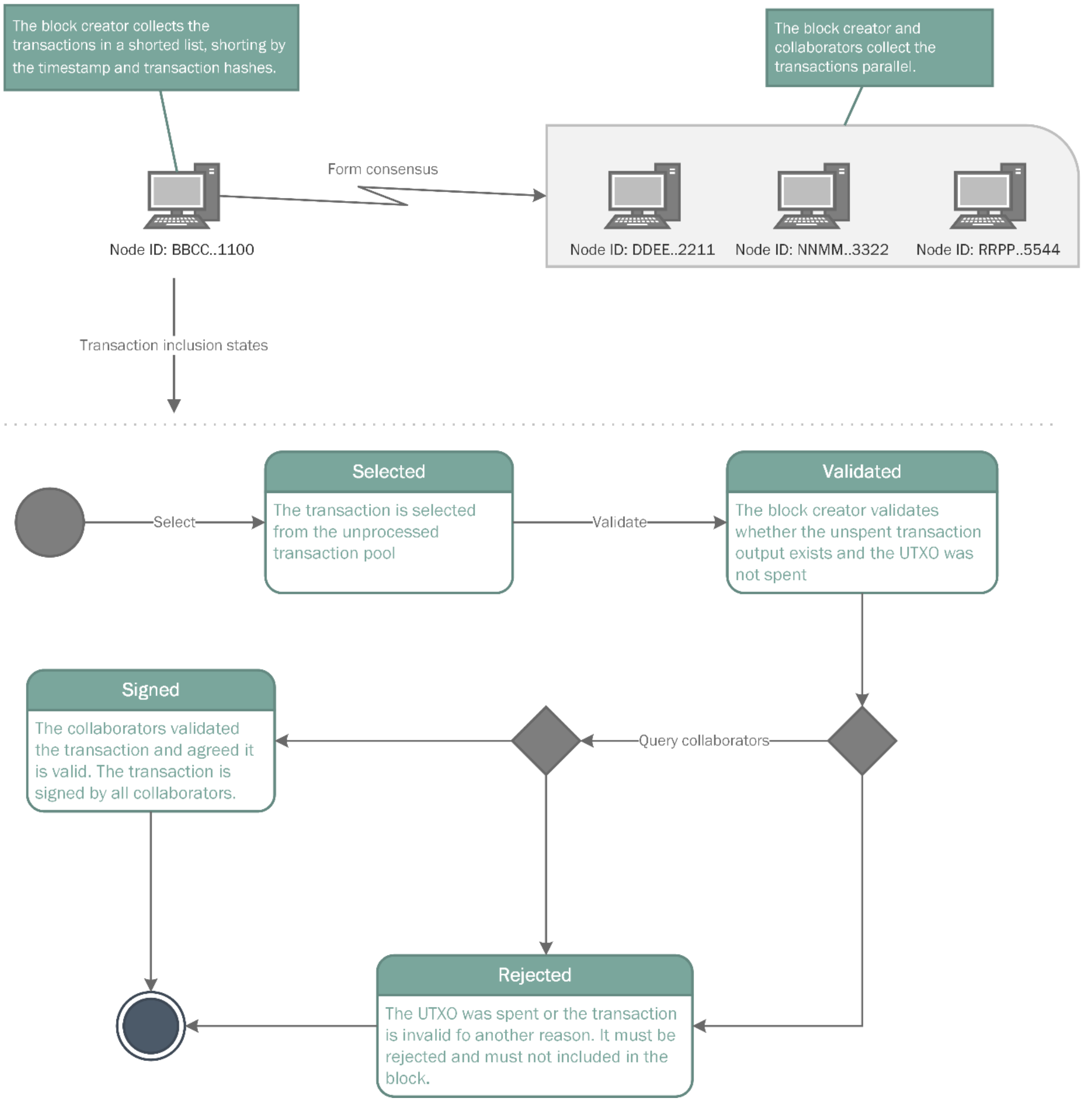
Node BBCC...1100, which ID is the closest to the hash of the latest block is selected and has the right to present the next block to the network and collect transaction fees.

Collaborators that verify the block are selected. Nodes, which ID is the closest to the hash of the latest block are selected and have the right to sign the block and collect transaction fees.

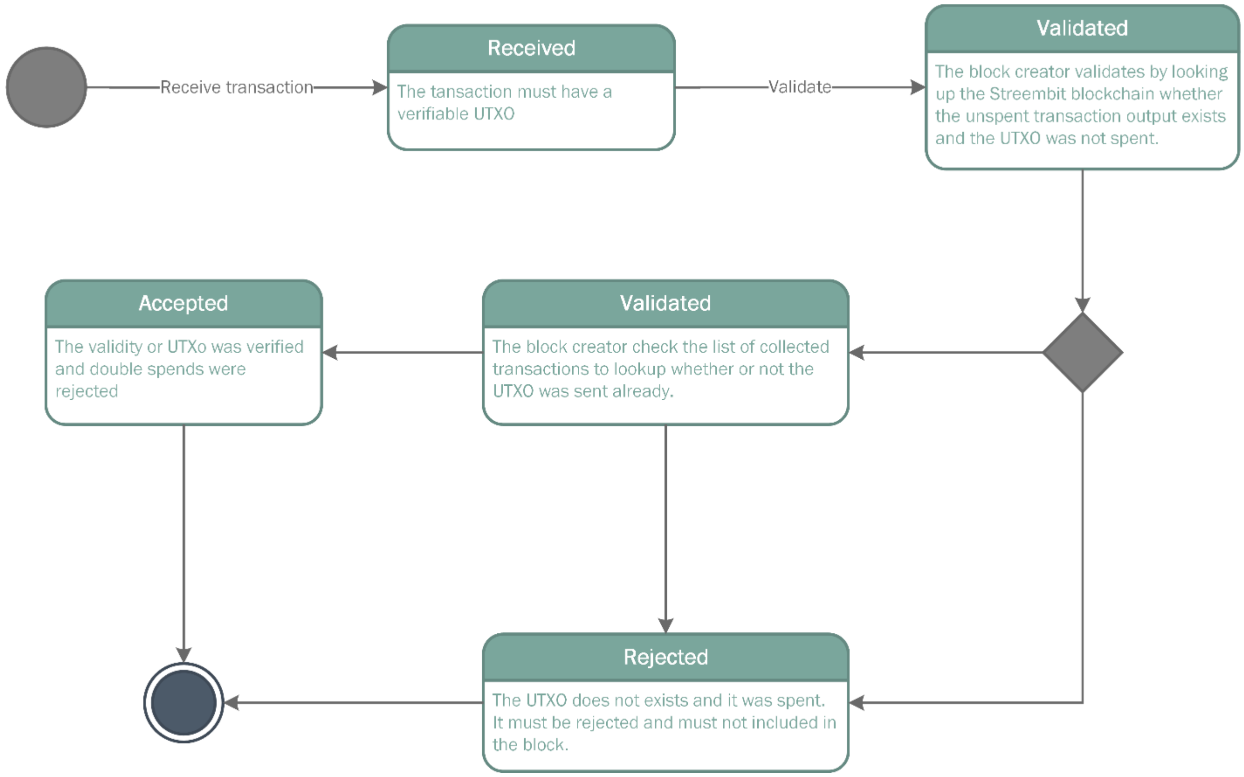
Step 2 - Process transactions.

The block creator and the collaborators work together to validate pending transactions and produce the block. The block creator and collaborators collect the pending transactions parallel. The software validates the inputs to verify that each transaction is valid. Honest collaborators of the Byzantine Agreement reject transactions that include a fraudulent input. The block creator must include all valid transactions in the block. The transactions must be in an ordered list by sorting the items based on their timestamp and transaction hash. Parallel, the collaborators check the integrity of each pending transaction that exists in the pool of unprocessed transactions. The block creator and collaborators communicate to form a consensus about the list of valid transactions. In case if the block creator is offline the next closest collaborator steps in and continues coordinating the block creation. The block creator sends each valid transaction to the collaborator. Since the process is parallel and the collaborator validates the pending transactions as well, at this phase of the process the collaborator must have the list of valid transactions. The collaborator signs with its private key the valid transactions and send them back to the block creator. If the majority of collaborators believe a transaction is valid and signed it, the block creator includes the transaction in the block. The block creator validates the signatures.

Streembit blockchain. Block creation Step No. 2 – transaction collection and processing state diagram


The block creator collects the transactions in a shorted list, shorting by the timestamp and transaction hashes.
The block creator and collaborators collect the transactions parallel.
Form consensus
Node ID: BBCC..1100
Node ID: DDEE..2211
Node ID: NNMM..3322
Node ID: RRPP..5544
Transaction inclusion states
Select
Selected
The transaction is selected from the unprocessed transaction pool
Validate
Validated
The block creator validates whether the unspent transaction output exists and the UTXO was not spent
Query collaborators
Signed
The collaborators validated the transaction and agreed it is valid. The transaction is signed by all collaborators.
Rejected
The UTXO was spent or the transaction is invalid fo another reason. It must be rejected and must not included in the block.

Streembit blockchain. Mitigate double spending with PoC sate diagram


Receive transaction
Received
The tansaction must have a verifiable UTXO
Validate
Validated
The block creator validates by looking up the Streembit blockchain whether the unspent transaction output exists and the UTXO was not spent.
Accepted
The validity or UTXo was verified and double spends were rejected
Validated
The block creator check the list of collected transactions to lookup whether or not the UTXO was sent already.
Rejected
The UTXO does not exists and it was spent. It must be rejected and must not included in the block.

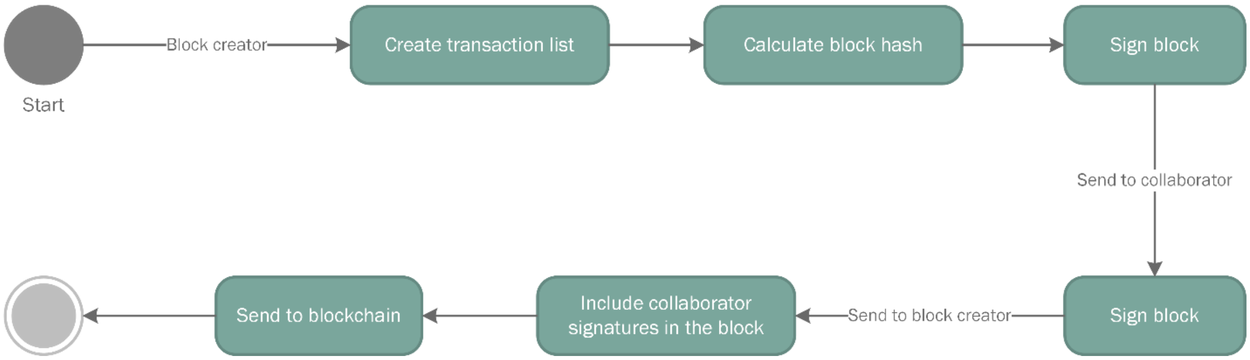
Detached or Orphaned blocks can happen and are valid blocks in many blockchain systems like Bitcoin or Ethereum. Orphaned blocks are not part of the main chain. They can occur naturally when two miners produce blocks at similar times or they can be caused by an attacker (with enough hashing power) attempting to reverse transactions. Orphaned blocks occur frequently. Blockchain.info keeps track of all the orphaned blocks that appear on the network, and there is usually 1-3 of these mishaps every single day of the week. There is a new block mined roughly every ten minutes, which means there are 144 block rewards available to miners every day. This also means a little over 1% of the blocks that are mined every day are orphaned.
The fact the orphaned blocks could exists in most blockchains significantly increases the chance of 51% attacks and orphaned blocks can also pop up in attack situations. The infamous 51% attack is one of the few flaws of the Bitcoin blockchain where an attacker can use their majority share of the network hashrate to create their own version of the blockchain. The longest chain is the most important factor when it comes to which block is going to be viewed as valid by the Bitcoin network, but a miner who owns 51% of the network is able to work on their own blockchain at a faster rate than everyone else. In other words, an attacker could create their own chain of orphaned blocks in order to take control of the Bitcoin ledger. If someone is able to centralize power over Bitcoin's history of events, then they also have the power to double-spend their bitcoins and block others from using the network.

It is not possible for an orphaned block to exist in PoC. Therefore, we believe the double spending prevention mechanism of PoC is more effective than in any other blockchain system.
The main difference lies in the atomic nature of PoC block creation. In PoC, the block creator applies a set of distinct changes as a single operation. Always only one node, the "lucky one" can present the block to the network. There is no similar parallel process exists on the network. Thus, orphaned blocks are a significantly smaller problem in PoC than in other blockchain systems. Detached or Orphaned blocks are valid blocks which are not part of the main chain.

Step 3 – Finalize the block.

After 60 seconds (if at least one transaction is available) the transaction collection and validation is closed and the block creator creates the block by including the verified, valid transactions in the block. At this point the collaborators already validated the transactions and formed a consensus with the block creator about the list of valid transactions. The block creator calculates the hash of the block, signs it and sends it together with the list of transactions to the collaborators. This is the list of transactions that the parties agreed in the previous step. The collaborators validate the transaction list, ensure only the agreed transactions are included, and validate the signature of the block hash that was produced by the block creator. If the transaction list and the signature of the hash block are correct, then the collaborators sign the hash of the block and send it back to the block creator. The block creator validates the signature of the collaborators. If the signatures are valid the block creator adds them to the end of the block. The block data is complete.

Streembit blockchain. Block creation Step No. 3 – create the block activity diagram.



Step 4 – Minting right from block creation

The block creator sends the invalid or disputed transactions to the closest identity and depositing nodes. The recipients of the disputed transactions are the arbitrators. The arbitrators resolve the dispute by complying with the arbitration rules of PoC. At this stage, when the arbitration rules are under debate the disputed transaction are rejected and deleted.

The collaborators who participated in the production of the block acquire the right to mint coins. Collaborators cannot mint, nor spend the reward, earlier than 30 days from the time of block creation. This allows sufficient time to filter out dishonest users that created fraudulent blocks.

Random methods for collaborator selection

The base for randomness is the hash of the previous block. The system uses the hash of the last valid block. The collaborators are determined by finding at least 5 full Streembit node collaborators whose DHT NodeID is closest to the previous block hash. The closeness is determined by finding the closest nodes to the hash value with XOR and string comparison functions. The XOR method is similar like how the Kademlia protocol finds the distance between the peer-to peer nodes.

Privacy

The most popular cryptocurrency blockchains such as Bitcoin are ironically, both anonymous and traceable. Every transaction can be tracked on the Bitcoin blockchain, and linked to the public keys involved in the transaction. By definition, public distributed ledgers (blockchains) store and make visible the transactions and balances of all addresses on the distributed storage. This public nature of the blockchain is a concern mainly for business users, as a competitor could watch their every move. Bitcoin wallets are anonymous until a transaction is linked to the owner of the wallet. Ordering a pizza could ruin that anonymity. Once the owner of the wallet is identified it is relatively easy to reconstruct every single transaction carried out by the owner.

The key is to find a balance between anonymity and traceability. We identify untraceability and unlinkability as the two main system requirements to ensure privacy on the blockchain. **Untraceability**: for each incoming transaction all possible senders are equally probable. **Unlinkability**: for any two outgoing transactions it is impossible to prove they were sent to the same person.

Transactions

To follow the Bitcoin design and naming conventions the FOWcoin blockchain implements two types of transactions, Pay-to-PubkeyHash and Pay-to-Script-Hash.

To send money between blockchain addresses, peers typically use Pay-to-PubkeyHash transactions. When redeeming coins that have been sent to an address, the recipient provides both the signature and the public key. The script verifies that the provided public key does hash to the hash in scriptPubKey, and it also checks the signature against the public key. This method is very similar to the transaction processing of Bitcoin or any other cryptocurrency system.

Pay-to-Script-Hash can be used to perform complex smart contract transactions. Unlike in Bitcoin, the Streembit Pay-to-Script-Hash transactions are always based on a predefined smart contract template. Pay to script hash (P2SH) transactions allow transactions to be sent to a script hash instead of a public key hash (which are performed with Pay-to-PubkeyHash transactions). To spend coins via P2SH, the recipient must provide:

1) A smart contract template.
2) A script that's hash matches with the script hash.
3) The data which is the parameter of the smart contract and evaluates the script to true.

Designated Executor transactions

In the PoC, users that submit transaction can request the transaction to be signed by a known entity, typically a node that can be held accountable. This function is essential for business and government users. For instance, we assume that a bank will allow transactions that are signed by a node which is known by the bank to be accountable and responsible for its actions. In most cases, it will likely be a node run by the bank. These are the identity based transactions – the identity of the transaction signee node is known to other users. In PoC, apart from the requested signees, at least 3 randomly selected nodes must sign the transaction to ensure the block integrity and that the transaction complies with the rules of PoC. The randomly selected users are entitled to receive a transaction processing fee, but not entitled to receive coin forge reward.. The transaction processing fee is agreed between the requesting and processing nodes, and it paid by the requesting node to the processing node directly. During an Identity Based Transaction, Step 2 of the above Produce Subsequent Blocks section differs as the block creator is the designated node. Thus, this node proposes the candidate node to the network.

Which nodes can be a Designated Executor?

Any accountable collaborator node can be a designated executor.

Simplified script to hash transactions
In the Streembit PoC consensus users can create a block that registers a maximum 256 bytes data in the blockchain. The data typically is the hash value of a smart contract, document, contract, agreement, promissory note, certificate, personal statement, software source, etc. We envision the hash value of many types of digital entities will be registered in the blockchain. The maximum 256 bytes of data is the combination of value and reference. The value is a maximum of 128 bytes enough to store the hex value of SHA512 hash. The reference (e.g. a URL or document location in the DHT) is also 128 bytes. With simplified script to hash transactions – just like with identity based transactions – the user can request that specific nodes must sign the transaction. If no signee collaborators are requested, randomly selected collaborators will create the block. The randomly selected collaborators are entitled to receive rewards to forge coins.

Addressing scalability problems of blockchain

Centralized systems (servers) are expensive to maintain and do not scale easily. At the same time and ironically, growing decentralized, permissionless blockchains can become centralized. The larger the blockchain grows, the larger the requirements become for storage, bandwidth, and computational power. Such a large data load must be handled by powerful nodes in the network, leading to a risk of much higher centralization if the blockchain becomes large enough that only a few nodes are able to process a block. Once the inevitable size increase of blockchain happens, the few powerful nodes are no different from client-server systems.

In current blockchain systems, all users (except light clients, but those use centralized servers already) must download the full blockchain and maintain the data by updating their local database all the time with any new block generated on the system. The scalability challenge in conventional blockchain systems is that more users produce more data, and then the whole ledger is distributed to all users. It is clear such design cannot work for most business use cases and transactional systems. A decentralized market or bank that produces many terabytes of data cannot function if every user must maintain the whole dataset.

Current blockchain systems can only handle a very small number of transactions. The average number of transactions is around 2000 per block in Bitcoin. That is equivalent to around 3 transactions per second. As of now, Bitcoin can handle 604,800 transactions/day (estimated 7/second), and Ethereum 1,296,000 transactions/day (estimated 15/second). This is a far cry from viability. A mid-size bank process would require more transactions than can be handled by either network per day.

The lack of scalability prevents the blockchain from being applicable to demanding business use-cases, as well as hinders future growth. Applying the blockchain to the Internet-of-Things, and especially machine-to-machine payments such as managing national smart meter grids or electric vehicle charging, will require the processing of tens of millions of transactions a day per use case. Since the beginning of the Bitcoin network 8 years ago, there have only been 220 million transactions. In the case of Bitcoin, this scalability problem has already come to light. Recent transactions have taken hours because of the block size limitation.

Solutions like Bitcoin and Ethereum have scaling issues primarily brought on by their protocols requiring that each full node process every transaction.

Large financial systems must manage 50,000 transactions per second. That means, to manage such a use case with a blockchain system, several hundred blocks must be created every second. Currently Bitcoin creates 1 block every 10 minutes.

Ethereum has identified this problem. The Ethereum developers have put forward the question to the blockchain community: is it possible to create a new mechanism, where only a small subset of nodes verifies each transaction? As long as there are enough nodes verifying each transaction so that the system is still highly secure, but sufficiently few that the system can process many transactions in parallel, could we not use such a technique to greatly increase a blockchain's throughput?

It is clear that with the current method of blockchains, which is sequential block processing, the transaction volume is very limited. When the system produces a block, it must include the hash of the previous block. One block must finish before a new block’s production begins. This workflow creates a sequential process. Even if the time between blocks is dramatically reduced to 5 seconds instead of the current 10 minutes of Bitcoin, and the block size is increased to 10 megabytes instead of the current 1 megabyte, with the average current 600 bytes per transaction size of Bitcoin the system would manage around 16,000 transaction in a block. With a 5 second transaction time that would be 3,200 transactions per second. This is more volume than the current model permits, but is still far from serious business requirements like the transaction volume of banks or stock exchanges. However, would be it possible to mitigate the issue of double spending with sequential 5 second block creation on a permissionless system? We believe that would be not possible with that short of a block time. The time interval of 5 seconds between two blocks is so small that the nodes wouldn't be able to form a stable consensus about the longest chain. Eventually of course the consensus would be formed, but until that happens the attacker will have many opportunities to double spend and defraud the system. There is a reason Satoshi picked a 10-minute block time: it is enough time to form consensus, reject double spending and agree what the latest longest chain is. While dramatically reducing the block time would improve scalability and transaction speed, such a short block time would make mitigating double spending impossible using current designs.

We need a different design that improves both scalability and speed, and still allows for mitigation of double spending.

The FOWcoin blockchain and the Proof of Consensus protocol addresses the problem of scalability by storing data and processing transactions in parallel subsystems. This technique is called blockchain Partitioning. In PoC, the system creates parallel collaborative processes and transforms data between these parallel processes. The base assumption is that Partitioning must result in significant increase of

- Number of blocks created in the block forming time window
- Number of transactions processed
- Speed of transaction processing

The main design principle is that - rather than storing the whole blockchain - each partition stores only data that is relevant to the partition. The rules of determining what data belongs to a certain partition are based on a mapping method. The mapping method creates lists by correlating the hash of data with NodeIDs using a string comparison function. E.g. the system allocates the data which start with "aabcc…" to a node whose NodeID is closest to the "aabcc…" hash value.

In Partitioning the nodes perform two major tasks

1) Maintain the set of unspent transactions (UTXO) that belong to the partition
2) Maintain the set of blocks that belong to the partition
3) Provide data lookup and reporting services for nodes that operate in other partitions

Due to CAP theorem (Brewer's theorem) principles, it is impossible for a distributed computer system to simultaneously provide more than two out of three of the following: consistency, availability and partition tolerance. Partitioning guarantees consistency and availability, while compromising on partition tolerance by delegating data processing to partition subsystems instead of attempting to maintain partition tolerance across the whole network.

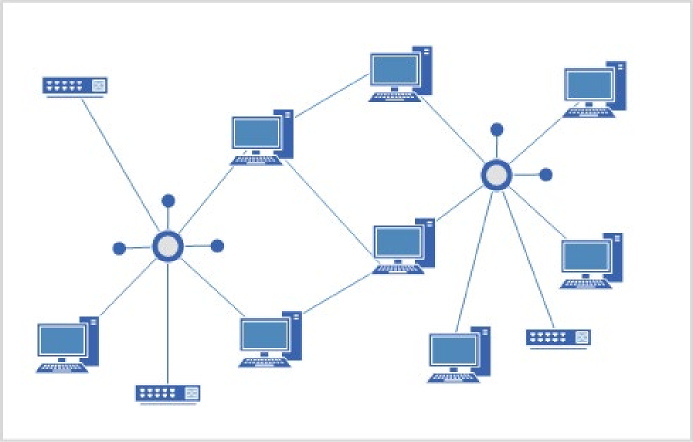
Base assumptions with regards to the Partitioning implementation

Relational database theory developed from mathematical logic and set theory, but formalisms have been introduced over time to tackle specific extensions of these core theories. PoC uses category theory as a unifying formalism for the blockchain database (distributed ledger). This system and PoC manages migration of data between partitions by applying category theory principles. Category theory was invented to strengthen the connection between topology and algebra, but it quickly spread to neighboring fields. By providing a precise language for comparing different universes of discourse, category theory has been a unifying force for mathematics. Category theory has been remarkably successful in formalizing the semantics of programming languages. It has been proven that database theory and category theory are naturally compatible, in fact a basic database schema and a category are very much the same thing. [21] Once a simple dictionary is set up, classical category theoretic results and constructions capture many of the formal results that appear in database literature. Distributed databases can work in the same way as relational databases do in terms of the formalism of the schema. Thus, PoC uses categories and functors to uniformly represent the distributed database, to describe the logic of moving data between entities across the distributed database, which then allows for forming of partitions. The categories and functors will then be used for partition mapping and data migration between partitions. Partitions are crucial for this system. **Using partitions allows parallel block generation** in the blockchain. However, the data must be migrated between partitions.

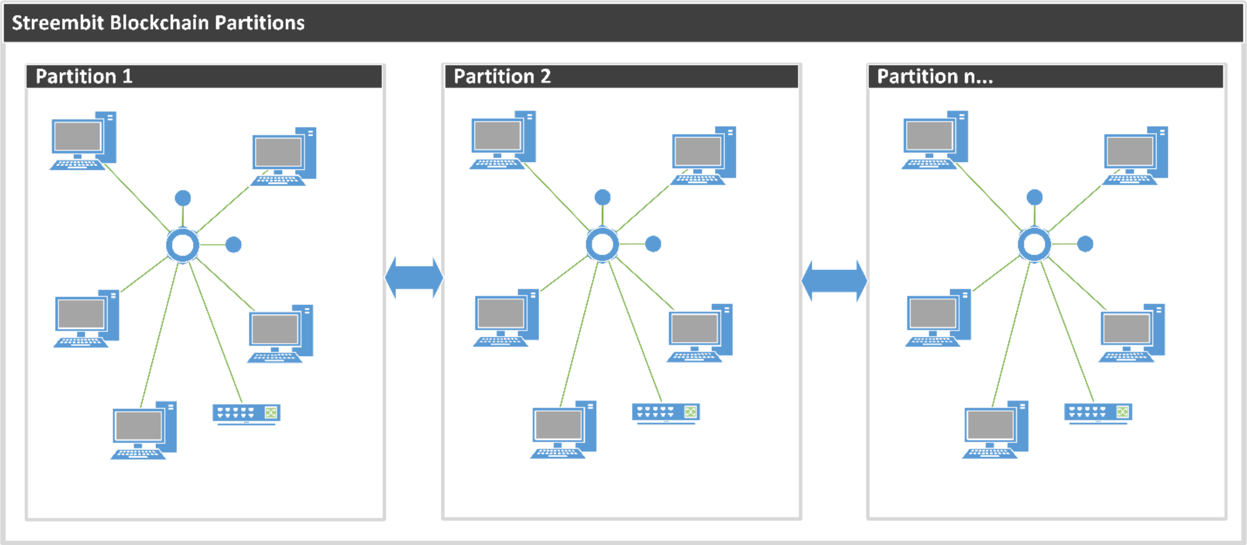
Through morphism of the mathematical category theory we identify objects and determine whether the objects belong to a particular partition, and whether or not a partition should handle an object. This is synonymous to relational database schema migration. Similar to relational database methods that migrate data between schemas, PoC uses morphism and category theory functors to move the data between partitions.

Forming partitions

Current blockchain systems are one big distributed database - all users must download, maintain, and share the same set of blocks.



On the other hand, the FOWcoin blockchain divides the data into partitions. **PoC partitions are autonomous collaborative processes.** In each partition a set of users (nodes) collaborate. These are the set of the categories as far as the category theory is concerned. Each partition maintains its own collection of blocks. In practice that means, instead of downloading and maintaining a 100 GB and soon 1 terabyte full blockchain, each partition - and its users - manages significantly less data (100 megabytes – 1 gigabytes). Even mobile devices can participate in block creation by joining a partition. PoC partitioning means that the system divides the large blockchain to smaller datasets.



The number of partitions grows proportionally with the number of users in the system. As the number of users grows, there are more portions created. A larger distributed ledger is divided into more partitions. The multiple parallel processes result in each partition maintaining its own block creation.

The design must be a loosely coupled, event driven, and asynchronous architecture. In the partitioning workflow, the first step is end-users creating transactions. This data must be included in a block. The transactions and data are forwarded to the network by saying “here is the data, include it in a block as soon as possible”. Separation of concern is the principle of this design: the user who produces the data is not concerned when the transaction will be included in the block. The user creates the transaction, raises an event to notify the block creator nodes of PoC that the transaction has been created. The system maintains the new transactions list. The block creators pick up the transaction from this list. At this stage, the block creator must verify that the UTXO of the transaction is valid, and it is not a fraudulent double spend. The first key difference from a conventional blockchain system appears in this step: the block creator verifies whether or not the UTXO is valid by querying the partition that maintains the particular UTXO. In conventional blockchain systems, regardless the number of UTXOs, transactions, and blocks, each node must maintain the full data set. If there are 1 billion transactions, and the data exceeds one petabyte the nodes still must download the data. As we pointed out earlier, this requires powerful computers and excludes average nodes from block creation. This leads to centralization and makes it impossible to implement a scalable system. In our system, the block creators will pick up the transaction and include it in a block when they are able to. Thus, the process in which the user and the block creator node operate is asynchronous – the two parties are not concerned when the other party will complete its task.

Partitioning states

A partition functions as a database. In fact, partitions store data in the Streembit distributed database as a key-value store. Similar to relational databases, states exist in the key-value store. The challenge is to represent a database state.

We describe the states of partitions with category theory axioms. In compliance with category theory, there are functors. A database state is a collection of sets/functions. In conventional relational databases one sets/functions for each table/column, satisfying the schema structure, data types and integrity constraints. In the world of categories, it is natural to represent this as a functor from the category CS representing the schema to the category et of all possible sets. Normally, in relational database a database state is a functor in which: each table (object) in the schema is mapped to the set (in Set) of row identifiers stored in that table, and each column (arrow) is mapped to an arrow (in Set) that associates row identifiers to column values for this specific state. However, in case of the Streembit key-value store there is only one such table. In the Streembit key-value store, columns don’t exist and to represent complex data structure we use the keys. The key includes tags that are separated by forward slashes. E.g. “UTXO/partitions/1/abcd..123” is a key that references a UTXO value in partition 1. Functionality wise, the forward slash separated elements of the keys are equivalent to relation database columns.

This fully characterizes a state. Using functors that preserve identity and composition, it also guarantees that the state is valid or is a model for S. This means that only functors representing valid states can be defined between CS and Set. More formally our formula states that database states are set-valued Functors:

A database state of schema S is a functor γ: $C_s \rightarrow$ **Set**. The functors γ takes each object t in C_s to a set $\gamma(t) \in$ **Set** (representing the key-value element identifiers of table t for this state), and each column (i.e. forward slash separated element in the key) c : t → t’ in ArrCS (t, t’) to an arrow $\gamma(c)$: $\gamma(t) \rightarrow \gamma(t')$ in **Set** that associates each row identifier $\times \in \gamma(t)$ to its corresponding column value in the set $\gamma(t')$.

One of the strengths of category theory is to allow us to change the level of abstraction very naturally. [21]. We exploitthis here, by representing database states as the objects of a much larger category called C_s - **Set**. C_s - **Set** thus is the category of all valid database states for schema S. The arrows of C_s - **Set** are called natural transformations. Using transformations, we can also capture database insert and delete operations.

Further editions of this white paper will provide more details on the Partitioning and its relation to category theory, functors, and object mapping.

Compliance

Banks, business, and organizations must operate in an auditable manner and comply with laws and regulations. The users of FOWcoin blockchain can implement compliance with certificates. The FOWcoin blockchain allows digital certification integration. Addresses and transactions can be associated with digital certificates. Digital certificates, issued by a proper registration agent, could be used to ensure that only some given entities can identify the owner of a given public key and consequently a blockchain address. On the FOWcoin blockchain in certain transactions, typically those that are governed by a smart contract, one may require that each user has digital certificate issued by a suitable entity (e.g., by a CA, bank, or an identity management framework such as Authenticity). When making payment to another user, the sender may forward the certificate along with the information about the certification issuing authority, and signature using the private key of the certificate. Including the certification in the transaction means the identity of users can be stored in the blockchain. This enables compliance with laws and regulations. Also, there are many use cases where the identity of the parties involved must be revealed. For example, a bank can issue bonds, or a publicly listed company can allot shares on the FOWcoin blockchain where the identity of the issuer is included using its certificate. There is a designated field for certificates in FOWcoin blockchain transactions.

In summary, we can put to rest the noncompliance concern about blockchain systems, without sacrificing the privacy of the users. Some users can perform anonymous private transactions while other users can comply with regulations by using digital certificates.

Kademlia DHT based blockchain storage

Streembit is built on a Kademlia distributed hash table. The FOWcoin blockchain uses the underlying DHT of the Streembit communication network. The key/value store of the Kademlia DHT is the decentralised storage used to orchestrate the blockchain related functions. Most of the data structures of the blockchain are temporarily, or permanently stored in the DHT using the underlying message and contact security of the Streembit system.

Smart contracts

Smart contracts represent the next step in the progression of blockchains from a distributed financial ledger to an all-purpose utility. A smart contract is an agreement whose execution is both automatable and enforceable. Automatable by computers and software, although some parts may require human input and control. We see enforceability by either legal enforcement of rights and obligations, or through tamper-proof execution in the case of simple smart contracts.

A smart contract is a method to form agreements via the blockchain. The agreements are typically between people, business, machines or IoT devices. Smart contracts aim to solve common problems in a decentralised manner, in a way that minimizes trust, and by excluding centralized service providers from the process. Minimal trust allows minimal human judgement, thus allowing partial or complete automation. [12] There are many types of smart contracts.

A smart property is a type of smart contract whose ownership is controlled via the blockchain. Examples include, but are not limited to, physical property such as cars, phones, or real estate properties. Smart property also includes non-physical property, like equity in a business, or access rights to a remote supercomputer. Using smart properties allows trustless and permissionless trading, the trust isn’t provided by a centralised service provider (i.e., EBay) but by the smart contract. This reduces fraud, reduces mediation fees, and allows trades to take place that otherwise would never have happened. For example, it allows strangers to loan you money over the internet taking your smart property as collateral, which should make lending more competitive and consequently credit cheaper. [13]

Another type of smart contract is the transferable virtual property. Typically, digital objects such as a video file, or an asset in a video game, can be infinitely duplicated or controlled by a central authority, or the video game service provider respectively. Transferable virtual properties are digital objects that are not controlled by a central authority, can be exchanged between peers, and can only have a single owner. For instance, game asset ownership in online games (virtual worlds) can be handled by smart contracts by creating transferable virtual property. Currently, games will store the game assets on the centralised game servers of the game provider. That means all exchanges depend on the centralised service provider and the transactions are not permissionless – users can't use the assets in other games or applications [14].

A service contract is a type of smart contract that governs, and often automates, business deals between service providers and consumers. A taxi smart contract that facilitates the deal between taxi drivers and passengers is a service contract.

A smart contract can be both a “smart legal contract” (where the agreement is a legal agreement, which is then capable of automatic execution in software) and a “smart contract code” (which may not necessarily be linked to a formal legal agreement, yet must be executed automatically). For smart legal contracts, the enforceable aspects could be complex rights and obligations, whereas for smart contract code, what is being enforced might simply be the execution of the code. The smart contract “is automatable”, which means that it in practice there are parts of a legal agreement whose execution might not be automatic and will require human input and control. However, to be a “smart contract” the design and system must support some part of the execution that is capable of being automated (otherwise it is not “smart”). [11]

The execution of a Streembit smart contract means the system validates the conditions of the contract and performs predefined actions that depend on whether the conditions have been satisfied or not. For instance, in a service provider-customer relationship smart contract, upon delivering the service, the smart contract pays out the service fee to the service provider. In case of insufficient service delivery, the smart contract pays out compensation to the customer. In the case of a dispute, the parties would normally select traditional methods for dispute resolution. It is not practical to automate all aspects of everyday business deals, and the parties will most likely select traditional dispute resolution methods such as binding (or non-binding) arbitration, or recourse to the courts of law [11]. These traditional methods are backed by the power of government as embodied in the law. For disputes related to contracts, the courts have extensive experience of awarding damages or other reliefs as appropriate, and in some cases assisting in the enforcement of payment of damages. We agree with Clack, Bakshi, and Lee Braine [11] that a fully tamper-proof execution is not practical. It is quite common in business deals that the agreement to be varied dynamically e.g. to permit a payment holiday, or give discount. It is not possible to code all variations in advance and therefore tamper-proof execution will be applicable to only simple smart contracts.

Examples

Practical example 1. Streembit Taxi Service Smart Contract

One exciting use case is for the provision of a decentralised taxi service. This will be possible because of the combination of the Streembit communication network and blockchain. The key is leveraging smart contracts to remove the need for a central authority (i.e. Uber).

Here is a high-level explanation of how it could work. A driver providing a taxi service would publish his business details and availability to the Streembit DHT. A potential client could then search for a ride. Let’s say they need a ride from the London Heathrow airport. The customer would then search the DHT for nearby drivers. Details about the driver, such as their availability, price, rating, or any number of other metrics, will be available for review. The customer can then make an offer to a driver using the Streembit text, audio, or video chat, or if everything looks fine simply press the "Order" button. Upon the agreement of the two parties a smart contract will be made. The contract could either have a fixed price or a price based on tariff and distance driven. Once they arrive at the destination the customer just hits the "Pay" button and it’s all done! In the event of partial execution of the smart contract, the driver and customer could agree to reduce the price. Maybe the vehicle had issues that negatively affected the ride, and the driver offers a discount. Or if the driver went above and beyond expectations the customer may want to increase the price to leave a tip. The customer and driver would have a chance to rate each other through the smart contract as well. This is all done, even the payment, without need for a centralised service with Streembit. This is a fully P2P transaction.

In the case of a dispute or breach of contract the personal details of the driver or customer can be retrieved from the DHT or smart contract so the issue can be settled in court. Serious problems would of course be resolved by calling the police.

This whole process is peer-to-peer. The system won't be subject to the laws that are meant to hinder service providers like Uber that deal with the sharing economy. We are simply providing a tool to facilitate this sharing economy, and keep the middlemen out. Just as Photoshop facilitates the creation of art, Streembit will facilitate peer-to-peer services. This could also be done for many other service industries such as a cleaning, yard-work, or accounting services. Like the Internet-of-Things, the sharing economy may require decentralisation to be successful.

Practical example 2. Transfer of property ownership contract.

Real estate ownership transfer could be handled by our smart contracts. Ownership of the contract could be transferred upon certain criteria. For instance, the ownership of a collateral property can be transferred to the creditor via smart contract upon the borrower failing to repay and meet conditions of a loan.

Implementation wise, the smart contract defines the parties that are the existing owner and beneficiary. The contract defines the criteria, so the parties agree that upon which event the property ownership must be transferred. That means the owner accepts that upon the agreed event, the ownership of the property will be transferred to the beneficiary. The contract stores reference documents such as the property ownership document (e.g. the Land Registry title register document in the case of the United Kingdom), the birth certificate of the owner, and the public key that associates with the owner and verified by an identity service provider.

At the start of the contract, the system verifies the creditor transferred the loan to the borrower’s address. As the repayment dates of the loan are defined, the contract monitors the address that should receive the loan repayment. If the loan is not repaid the system start notifying relevant parties that the ownership of the property must be transferred to the creditor (beneficiary). In the not so distant future, the Land Register office could connect to the smart contract system to fully automate the execution of the contract. The real estate smart contracts enable the trading of properties between users that reside in different continents and jurisdictions.

Practical example 3. Financial smart contracts.

Various financial smart contracts can be created in the FOWcoin blockchain to manage financial assets and value trade. The contracts must be capable of managing highly complex actions, such as fixing payment priorities in a structured note.

Financial instrument smart contracts could be

- Bonds (fixed-income securities)
- Bills e.g. T-Bills
- Commercial paper (unsecured promissory notes)
- Stock options
- Equity futures
- Forex/currency futures
- Interest rate swaps
- Currency swaps
- Trade clearing and settlements
- Micro insurance
- Insurance claim processing

Financial smart contracts are capable of managing inter-bank and inter-exchange transactions. Using the contracts, businesses can access a global audience. The financial smart contracts could offer number of benefits such as increased speed and real-time updates. By eliminating manual process and automating the transaction, the accuracy of the business process is also increased. Automated transactions are not only faster, but less prone to manual error. Using smart contracts makes the transaction more economical by requiring fewer intermediaries in the process.

Practical example 4. Manage drone devices with Streembit and smart contracts

Streembit is an IoT framework and capable of managing and controlling IoT devices such as internet connected drones. Many drone related use cases can be automated with smart contracts. For instance, drones that provide aerial photography or goods delivery services can enter into contract with buyers using smart contracts. Instead of using complex client-server systems, the drones and drone operators can sell services on the blockchain using smart contracts. Device control and management tasks can also be facilitated with smart contract. For example, our smart contracts allow for implementation of secure self-destructing messages for military or law enforcement drone devices. The self-destruction of messages is executed based on a set of declaratively predefined criteria. If a drone is down and fails to satisfy the preprogrammed conditions of smart contracts (i.e. the device is not moving and stopped in enemy territory) then the messages in the drone are self-destructed. We achieve this by configuring a simple JSON data structure within the rules section of the smart contract template, without reprogramming the firmware of the drone device. Such flexible, declarative message management can be implemented using an auditable blockchain and the Turing complete smart contract engine of Streembit.

Smart contract templates

The Streembit smart contract is always based on a predefined smart contract template.
A template is an electronic representation of a legal document. The legal document could be the agreement between two private citizens, between businesses, or issued by a standards body. A template may contain both legal prose and parameters. Agreements are derived from templates, and both the legal prose and parameters may becustomized during negotiation. Values are mandatory for all parameters in a signed agreement. Some values, such as the name or the location of the template, are read only. The customization of legal prose and parameters at this stage is commonplace and results from negotiation between the counterparties. It is common for agreements to comprise multiple documents, such as Framework Agreements (e.g. a Master Agreement) with various Annexes (e.g. a Schedule) and Credit Support Documentation (e.g. a Credit Support Annex). Thus, the legal prose of an agreement

will be derived from that of the template, but need not be identical, and similarly the parameters of the agreement will be derived from the template, but need not be identical. [11]

The Streembit smart contract template is a data structure that includes

I. Properties of the smart contract. The property definitions are the important attributes of the contract such as the name, subject, description, parties, the business case, the contract handles, start time, expiry time, roles of parties e.g. beneficiary, arbitrator, validator, etc. Each property definition includes the 1) name, 2) data type, 3) value 4) read only flag (optional, default is false), and 5) allowed values or limitation on values (optional).

II. Rules and criteria which upon the smart contract is executed and/or determine the outcome of the contract. Typically, the criteria will require input that could be either manual or automated. For example, a betting contract can automatically validate the result of a sport event. A property transfer contract however, will require manual input from the owner of beneficiary, so that the contract can validate the criteria.

III. The function, or collection of functions, that the Turing complete software module of Streembit can execute. Typically, the properties are the parameters of the function, the function then implements the rules and criteria handling. In the case of a sports betting contract, an example of this would be a function checking the result of a Manchester United – Liverpool game, and then paying money to the winner of the bet.

IV. Reference documents of the contract. These could be digital certificates, business licenses, land registry titles, ID cards, court orders, employment contracts, and any other personal, business, or legal document that is related to the contract. Not all smart contracts will have reference documents. The aforementioned betting contract most likely will include only the gambling license of the betting company.

The Streembit smart contract templates must be published in a public source code repository, so that the software application can access it, and create the contract based on the populated template. By default, the templates are published in the Streembit public source repository, but users can agree on any type of location for the smart contract template. The Streembit software application lists the published smart contacts in a simple JSON data structure. The "URI" field refers the location where the template is available for download. Client software applications pull the template from the URI location for validation and verification purposes.

Smart contract publication for the decentralised Taxi template:

```
{
    "contract": "Taxi",
    "uri": "https://github.com/streembit/smartcontrtacts/templates/taxi"
}
```

A skeleton smart contract template:

```
{
    "properties": [],
    "rules": [],
    "functions": [],
    "references": []
}
```

The properties field of the Streembit Taxi smart contract:

```
{
    "properties": [
        {
            "name": "name",
            "data_type": "string",
            "value": "Taxi Contract",
            "read_only": true
        },
        {
            "name": "description",
            "data_type": "string",
            "value": "Smart contract for decentralised, P2P taxi service",
            "read_only": true
        },
        {
            "name": "service_provider",
            "data_type": "string",
            "value": ""
        },
        {
            "name": "passenger",
            "data_type": "string",
            "value": ""
        },
        {
            "name": "start_place",
            "data_type": "string",
            "value": ""
        },
        {
            "name": "destination",
            "data_type": "string",
            "value": ""
        },
        {
            "name": "tariff_unit",
            "data_type": "string",
            "value": "km",
            "value_selection": [ "km", "mile", "minute", "hour" ]
        },
        {
            "name": "tariff",
            "data_type": "duble",
            "value": "1.56"
        },
        {
            "name": "currency",
            "data_type": "string",
            "value": "USD"
        },
        {
            "name": "total_price",
            "data_type": "double",
            "value": ""
        }
    ]
}
```

Bond specific properties of the financial Bond smart contract:

```
{
    "properties": [
        {
            "name": "principal",
            "data_type": "duble",
            "value": 1000
        },
        {
            "name": "currency",
            "data_type": "string",
            "value": "GBP"
        },
        {
            "name": "maturity_unit",
            "data_type": "string",
            "value": "year"
        },
        {
            "name": "maturity_value",
            "data_type": "duble",
            "value": 3.5
        },
        {
            "name": " coupon_base",
            "data_type": "string",
            "value": "LIBOR"
        },
        {
            "name": " coupon_ratio",
            "data_type": "string",
            "value": "“percentage”"
        },
        {
            "name": "coupon_ratio_value",
            "data_type": "double",
            "value": 2.2
        },
        {
            "name": "interest_frequency",
            "data_type": "string",
            "value": "annual"
        },
        {
            "name": " credit_quality ",
            "data_type": "string",
            "value": "AAA"
        },
        {
            "name": "credit_quality_issuer",
            "data_type": "string",
            "value": " Moody's"
        },
        {
            "name": " credit_quality_signature",
            "data_type": "string",
            "value": "\"068521dd4086d903af5df32b57b4d1809e81f2ff"
        },
        {
            "name": " borrower_signature ",
            "data_type": "string",
            "value": "2a545d0a3448ab528ee30d652ac73ee2c04d5636"
        }
    ]
}
```

Functions
Other smart contract engines create scripting languages to handle smart contract functions and implement a Turing complete system. The function of a Streembit smart contract is implemented in the well tested and popular JavaScript language. The system runs the functions by creating a JavaScript script element and loading it into the DOM. In other words, the system dynamically injects the JavaScript function into the application instance of desktop, mobile or browser. This elegant, but effective, mechanism allows for the execution of complex functions. Using JavaScript makes smart contract design accessible to many software developers. For instance, complex, multi signature, several criteria dependent, bank issued bond contracts can be handled with the dynamic JavaScript injection method.

Reputation management on Streembit

Reputation management is a system that collects, distributes, and aggregates feedback about past behaviour. A reputation management system maintains trust in online communities. For instance, on eBay's Feedback Forum after each transaction the parties involved may evaluate each other. Such a reputation system plays a double role. The first is to encourage trade by making trade safer and increasing participants' trust. The second role is promoting satisfactory trade and increasing participants' trustworthiness. Reputation management is essential for traders and businesses in both client-server and peer-to-peer trading systems.

According to Claudia Keser, if there was no reputation management in an online market like eBay, there would most likely be a "market for lemons" problem. Because there is typically no opportunity for inspection, we would not be able to distinguish between sellers offering good or bad quality items. Buyers would then be reluctant to pay high prices, resulting in high-quality sellers leaving the market. In the end, only "lemons" would remain on the market, and customers would not want to transact there. With reputation management, as it has been shown in many studies, sellers with high reputation obtain higher prices. Reputation management mitigates the lemons problem.

According to Paul Resnick a good reputation management system identifies three basic principles:

- Entities in the system must be long-lived enough to establish an expectation of future interaction.
- Feedback concerning current interaction is elicited and distributed and must be visible in the future.
- Feedback must have influence over the actions/trust of entities in the future.

The Streembit reputation system is used by smart contracts to establish trust in smart contract-based trades. The reputation data is stored in the reference section of a smart contract.

Streembit manages reputation in a decentralized, peer-to-peer manner. The Streembit reputation system utilizes both the Streembit DHT and Merkle Tree blockchain concept.
In Streembit, the reputation ratings and evaluation of users are stored in the Streembit DHT storage. The reputation score is an integer number from 0 to 1000 that indicates the previous performance and trustworthiness of the user. In practical terms, a user who provides nothing more than an email address can get a score of 1, a user with an X.509 SSL certificate can get a score of 10. A user with extensive and immaculate trading history, high value verifiable (i.e. real) trades, with identity that has been verified by multiple third party reputation service providers, and who provides collateral for the decentralised, peer-to-peer trading could reach the 1000 maximum score.

The reputation data is stored as simple key-value pairs in the DHT by composing the key from keywords, the public key of the user who receives the rating, the public key of the user that provides the rating, and a timestamp. The score DHT key is the following
“reputation/rating/public_key_of user/public_key_of_marker/timestamp”
where the public_key_of trader tag of the key is the RMD160 hash of the trader public key. The public_key_of_marker tag is the RMD160 hash of the marker public key that provides the scores. This way an unlimited number of ratings can be stored for each user in the DHT.

The user who is evaluated maintains their ratings data in a private Merkle Tree blockchain structure. That maintaining user presents the data to other users upon request. The Merkle Tree contains all ratings related data stored in the DHT in a structured, tamper proof, and auditable manner. This Merkle Tree is a type of private blockchain that is exclusive to each user and functions as a distributed database of the reputation data. It is the

responsibility of the user who receives the rating to maintain this Merkle Tree mini-blockchain by ensuring that the DHT is frequently updated with the data. This Merkle Tree mini-blockchain is also stored in the DHT with the following key
“reputation/blocks/public_key_of user/hash”
where the “hash” tag is the hash of the block in the Merkle Tree mini-blockchain.

The buyer can verify the seller's reputation score by looking up the seller specific Merkle Tree mini-blockchain. The seller's Merkle Tree mini-blockchain is stored in the DHT and the seller can give permission for the buyer to look it up.

In every reputation management system it is hard to detect strategic behaviour. For example, since the ratings are not weighted by the monetary importance of the transaction, it is easy to build up a positive reputation by buying, or selling, a large number of cheap items, maybe even trade with friends, and then suddenly sell a very expensive item and not deliver it. To mitigate this issue third party reputation service providers can detect such strategic behaviour and reflect it in the score. Such third party reputation service providers – that build their service on top of Streembit - can verify the data entries to ensure the legitimacy of the reputation ratings.

Private transaction with smart contracts

The Streembit private transaction is an attempt in addressing the previously mentioned privacy issues of the blockchain.

The Streembit private transaction is always based on a smart contract so that the contract can be enforced outside of the blockchain, and within conventional enforcing frameworks, such as in court or involving other law enforcement methods. The details of the transaction are embedded in a smart contract. The transaction details could include any agreements and it could be any type of transactional or business information. Typically, it is a money send transaction, but it could be a property ownership transfer, promissory note, employment contract, share option agreement, etc. The smart contract must be created using a smart contract template, typically a publicly verifiable JSON smart contact template.
The parties must agree about the smart contract details by exchanging the smart contract. The parties typically exchange the smart contract via the Streembit communication software and as a part of a transaction (i.e. when sending money), but it can be exchanged via other communication channel such as encrypted email. Some smart contacts are complex and to finalize the contract could be a process of several steps. Thus we describe the exchange of smart contract as a step in the process.
Once the smart contract details are agreed the parties compute the hash of the contract. The hash of smart contract is the transaction data TD_1 in the private transaction. To keep the blockchain compact the smart contract itself never stored within the transaction and in the blockchain, only the hash of the smart contract is stored the transaction data TD_1.
In relation with the private transaction a Private Transaction Register (PTR) is created to store the transaction smart contracts. The PTR can hold an unlimited number of smart contracts. The transactions hashes are registered within the PTR in a Merkle tree that is encrypted and only make visible to the parties of the private transaction. (The cryptography keys are exchanged with generic Streembit key exchange methods). Storing the PTR itself is the responsibility of the parties of the private transaction. Naturally, the Streembit system stores the PTR in the Streembit DHT as a simple key/value data. The key is the hash of the PTR to enable easy lookup and the value tag of the key-value pair is always encrypted via the usual Streembit ECDH key exchange and AES cryptography functions.
Similarly to the *Simplified script to hash transaction* a private transaction registers only the hash of the private transaction in the publicly visible FOWcoin blockchain. In the case of a private transaction the hash is based on the PTR $ptrh_1$. Thus, one of the parties, typically the sender of the private transaction registers the hash of the PTR $ptrh_1$ in the public FOWcoin blockchain. This way the recipient can ensure the validity of the transaction data.

For money transfers, the source of the money must be always included in the encrypted private transaction. The source of money is the public address spa_1 that holds the coins in the public ledger. This public address spa_1 is the last known place of the money from the public blockchain viewpoint. In a money send private transaction, the PTR $ptrh_1$ is always include an address that is the private transaction address pa_1. Effectively the sender transfers the money from spa_1 to the private address pa_1. This transaction of spa_1 -> pa_1 is performed within the encrypted smart

contract and the mini block chain of the PTR $ptrh_1$. In other words, private transaction claims ownership over the money by sending it from the source public address spa_1 to private address pa_1 of the PTR $ptrh_1$.
It is necessary to associate the public address spa_1 with the Private Transaction Register, so that the recipient can reenter the money into the public ledger if it is required at any time in the future. In that scenario, first the initial spa_1 -> pa_1 transaction is registered in the public blockchain and then the money is transferred to a destination public address dpa_1 by quoting address pa_1 so the source and legitimacy of money can be verified. Such transaction is signed with the public key of pa_1 address. This way the owner of the money – that currently holds the money anonymously in a private transaction – has the option to re-enter it to the public ledger by sending it to destination public address dpa_1. As a generic rule, the amount which was sent to the destination public address dpa_1 must be equivalent to the balance of source address public address spa_1.

The below table demonstrates a multi-party money send transaction within a private transaction in which Bob, Alice, Pete and Frank send money to each other. Initially Bob sends the money to Alice, where the hash of the PTR is computed $ptrh_1$ and registered in the public blockchain. The transaction itself between Alice and Bob is never revealed to the public blockchain – only its hash. Any time a new transaction is preformed within the private transaction the PTR is updated and the new hash of the PTR is registered in the public blockchain. For instance when Pete sends money to Frank $ptrh_3$ it is registered in the public blockchain. This example demonstrates that the last private participant Frank can reenter the money to the public blockchain by sending it to the destination public address dpa_1. Please note the last step “Exit” from private transaction is not required. The transaction can be remain private, but the existence of pa_1 address allows the owner of the money to re-enter the money into the public blockchain.

Normally a private transaction is between two parties and used for a simple asset transfer. The below example demonstrates that the money sent to a private transaction cannot disappear even if it was traded between multiple parties, and it can be reintroduced to the public blockchain any time. Since the transaction can remain private forever, the smart contract based private transaction is a suitable medium to manage trading of assets, shares, bonds, property ownership, etc. without revealing transaction details in the public blockchain.

Workflow of a coin send private transaction

Life cycle of a money send Private transaction										
Entry	**Transactions TD_1**									**Exit**
Source public address owned by Bob spa_1 receives 100 coins by forging or purchasing them. Sends them to the private transaction address pa_1. Create a legally enforceable smart contract (SM).	→	Create a Private Transaction Register (PTR), its hash $ptrh_1$ and associate it with a valid address pa_1. These are the parameters of the smart contract.	Owner of spa_1 Bob sends money to Alice and transfers the ownership of hpa_1 to pa_1. Hash of PTR $ptrh_1$ is computed. The smart contract registers the change of ownership.	→	Owner of pa_1 Alice sends money to Pete and transfers the ownership of pa_1. Hash of PTR $ptrh_2$ is computed. The smart contract registers the change of ownership.	→	Owner of hpa_1 Pete sends money to Frank and transfer the ownership of pa_1. Hash of PTR $ptrh_3$ is computed. The smart contract registers the change of ownership.	→	Current owner of pa_1 Frank decides to send the money to a public address dpa_1. Frank can sign the transaction with pa_1 private key and prove the ownership of 100 coins to the public blockchain.	Public address dpa_1 holds the previously privately managed money.
		$ptrh_1$ & pa_1 addresses are registered in the private blockchain	$ptrh_1$ registered in the private blockchain of the smart contract		$ptrh_2$ registered in the private blockchain of the smart contract		$ptrh_3$ registered in the public blockchain			dpa_1 transaction is registered in the public blockchain

Streembit Communication System - Abstract

Streembit is a decentralized, peer to peer (P2P), permissionless, real time communication system for humans and machines. The application implements a system that securely manages humans-to-machine and machine-to-machine (M2M) communication without using a central server or client-server infrastructure. The actors of the system - both human users and Internet-of-Things (IOT) devices – are the nodes of the Streembit P2P network. The system uses a distributed hash table (DHT) for contact discovery services. The system ensures data integrity using public-private key cryptography. Messages are signed with the contact's private key on the client side using ECDSA cryptography. The data between users is encrypted using 256-bit AES symmetric encryption. The symmetric keys of the 256-bit AES cryptography are exchanged between contacts using ECDH key exchange. The video and audio communication between contacts uses the WebRTC protocol that allows end-to-end encrypted video and audio conversation in a true P2P manner. For all use cases the system end-to-end encrypts the communication between the peers without routing the conversation through any central server.

Background

Internet-of-Things (IOT) refers to the network of uniquely identifiable embedded hardware devices accessed through the Internet infrastructure. IOT devices are normally semi-autonomous, wireless devices participating in internet-worked communications using the familiar and established TCP/IP protocol stack. Typical applications are sensory equipment which, without the need for extra wiring, can collect and relay data to a central host - either on premise or in the increasingly popular cloud environment. The interconnection of these embedded devices is expected to usher in a new era of automation in nearly all fields and industry sectors. It enables advanced applications such as smart grids, remote surveillance systems, wireless heart monitors and many more.

According to Gartner, there will be nearly 26 billion devices connected to the Internet-of-Things by 2020. ABI Research estimates that more than 30 billion devices will be wirelessly connected to the Internet-of-Things by 2020. Per a recent survey and study done by Pew Research Internet Project, a large majority of the technology experts and engaged Internet users who responded, 83 percent agreed with the notion that the Internet-of-Things and embedded and wearable computing will have widespread and beneficial effects by 2025.

We focus on the following areas by implementing the Streembit system:

Security

Human-to-human digital communication and Internet-of-Things devices need to have some form of connectivity, resulting in significant security issues that businesses and residential users need to consider. The connected devices require a security protocol and its security policy should be in line with best security practices. These Internet-connected devices can easily expose businesses and homes to various security threats. At the same time there is no common, standardized, robust and proven authorization and access control protocol in existence for IOT devices. Of the numerous service providers rolling out devices and services, almost all of them implement their own security protocol, API and infrastructure. Consequently the development of custom security implementations increases the price of the product and the lack of standards increases the security risks.

The standard way would be to secure the communication and manage Internet-of-Things nodes from an authorization perspective using the robust, widely adopted and well tested public/private key infrastructure (PKI). To ensure confidentiality and secure communications, the core part of security should ideally be based on PKI paired with some kind of PKI certificate management. Existing systems tend to use custom authorization and access control schemes with domain specific login portals for user name/password based logins instead of using robust PKI based security. Using the public/private key infrastructure would pave the way for the deployment of a robust and secure authentication and access control scheme. The parties are identified by their public key. The authenticity of messages and the identities of the actors are then verified using PKI cryptography routines. Using the well-established public/private key infrastructure based security scheme would greatly simplify the authentication, access control and identity management aspects of IOT security.

Data Control

From a user perspective, this is one of the more significant barriers to the large-scale adoption of Internet-of-Things. Data control is commonly mistaken for data ownership. In a conventional computing system the issue was who owns the data. In Internet-of-Things the challenge is about deciding who gets access to the data. Enabling access to private data is a serious concern from a privacy standpoint. The P2P communication of Streembit excludes cloud providers, owners of client-server systems and central authorities from controlling, analyzing and data mining users' data. With the "serverless" Streembit approach, only the intended user can control the data.

Standards

There are almost as many software systems as there are Internet-of-Things device manufacturers. The lack of open standards is directly affecting the adoption rate of IOT devices as well having a negative impact on the user experience. Well-documented and robust APIs by providers could be a step towards open standards, but so far no such API usable by IOT devices exists. Sensors use custom APIs instead. Such domain specific product development dramatically slows down innovation as development resources are allocated to custom software development. This once again increases the price of the product and consequently lowers the adoption rate of these modern devices.

Decentralised, blockchain-based technology

Blockchain is a transaction database and ledger shared by all nodes that are normally participating in a digital currency system. A full copy of a currency's blockchain contains every transaction ever executed regarding to the currency. The blockchain can also perform distributed contracts. Such contract management is a method to form agreements between parties via the blockchain.

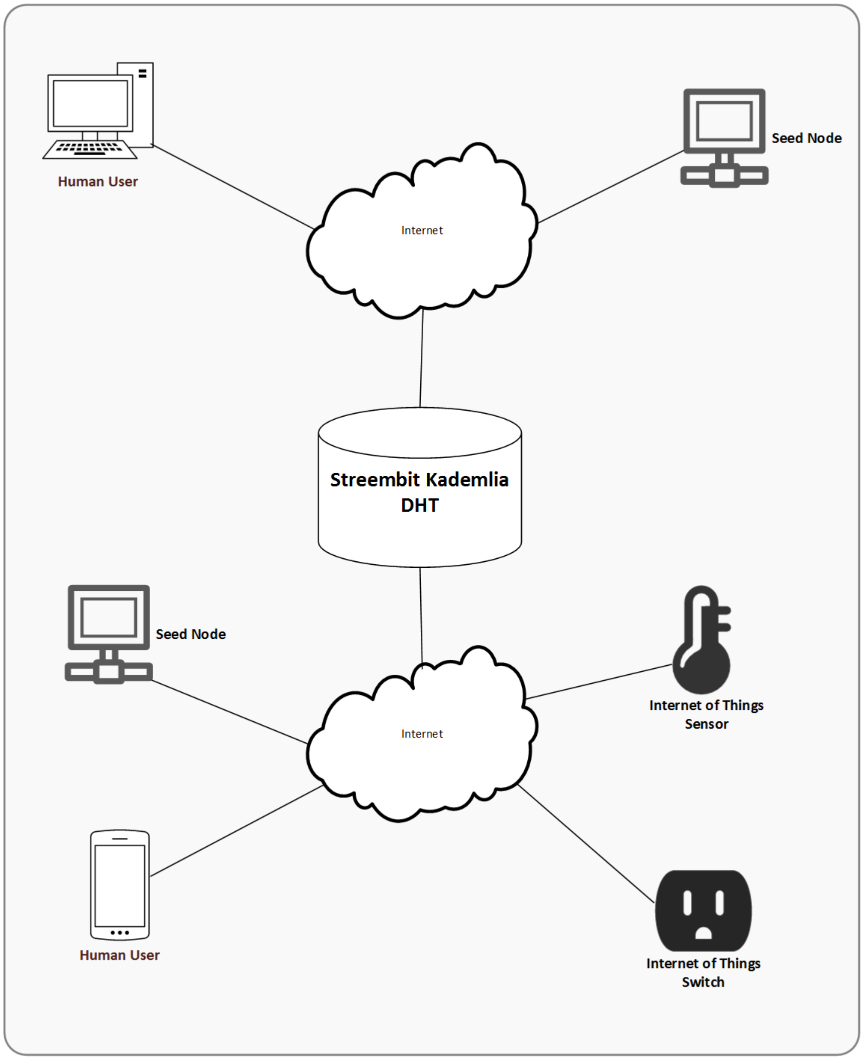
IDC recently concluded that blockchain technologies could be key tools for confirming data origin and accuracy, tracking updates and establishing true data authority for millions of different data fields. The blockchain is a solid model for establishing an audit trail, in addition to transferring and monitoring distinct entities that represent items of value. As a result, blockchain has the potential to serve as a foundation for improving the authenticity and accuracy of commercial, regulatory and government records. A blockchain-based system can track activities via a shared record that's resistant to hacking and unauthorized changes. Once this shared version of the "truth" is established, via a P2P network, multiple nodes ensure the integrity remains intact even as new records are added.

The Streembit System

System overview

The Streembit system forms a decentralized P2P overlay network to manage connections between human and IOT peers. The primary purpose of the P2P network is to facilitate connections between users in a "permissionless" manner, without using a central server or central authority. The participants in the network are the peer nodes. The P2P network is scalable and an unlimited number of nodes can participate in the network.

The system performs public private key infrastructure based authentication and access control functions to securely connect peer nodes. The application generates at least one private/public key pair on each peer node including on the Internet-of-Things devices. The actors of the system publish their public keys to the peer to peer network via a Kademlia distributed hash table (DHT). Each peer node knows the public key of the other connected peer nodes. The system identifies peer nodes in the peer to peer network by their public key. To ensure data integrity the nodes sign the messages with their private key. The nodes sign all messages - there are no unsigned messages circulated in the system. The requirement for signing the messages also help to mitigate the risks of Sybil attacks and DDoS attacks.



Streembit supports two types of network implementations, public and private Streembit networks. The main purpose of both network is the facilitate contact information exchange. The difference between them is the accessibility of the networks.

Public network. Anyone can connect to the public Streembit network. All valid, signed, cryptographically verified messages from any node are registered in the DHT. A node that is connected to the network can receive any messages from any other node. As the main purpose of the network type is to facilitate node discovery, humans and machines can find each other on the network in a decentralized manner, without using a corporate owned centralized system. Nodes publish their availability and network information such as IP address and port information in encrypted form using their contacts' ECDH public key. The DHT acts as a ledger to exchange contact information. The participating nodes of the network are the workers who route the information to any new connecting nodes upon request. The main Streembit network is public.

Private network, private hub. Decentralization addresses many infrastructure issues such as high availability and scalability. A decentralized system can achieve that by forming a collaborative network from the participating nodes. The democratic, libertarian concept of decentralized computing assumes that the nodes contribute to the network by routing messages to other nodes. The participating nodes keep alive the decentralized network by interacting with other nodes. A decentralized network provides users with many benefits, and in exchange for the benefits the nodes must route messages on the network. The issue is, potentially there are millions of nodes in a large decentralized network. At the same time, a limitless interaction with any node is not allowed in many use cases. For example, it is neither optimal nor practical from a resources viewpoint if an IOT-enabled garage door controller of a family home acts as a full node on the Streembit network. A full node by definition routes any messages to any connected nodes. A low power garage door controller device that is dedicated to one task – i.e. open the garage door upon the request of an authorized user – should not perform such message routing function. An IOT gateway of a family home or a communication network of a business should exclude actors from the communication that aren't part of their particular use case. Therefore, we have introduced the private network on the DHT concept in Streembit. The nodes of a private Streembit network uses the public Streembit network for a one-time information sharing: they publish their encrypted IP address or multicast DNS name, and then immediately disconnect from the public network to wait for the connection of their private node partners via the private network. In terms of authentication and access controls schemes, a private Streembit network is isolated from the public Streembit network meaning only certain peer nodes are allowed to connect to the private Streembit network. A private Streembit network is therefore functioning as a firewall: only preconfigured contacts are allowed to connect. The collection of the preconfigured contacts is maintained in a lookup list. The connection of devices and users which are not in the preconfigured list is refused by the private network so thataccess control is governed by built-in authentication functions. Typically, IOT devices within a building, teams, communities or businesses would run a private Streembit hub to establish an even more secure communication mechanism than the public Streembit network provides.

Streembit Kademlia DHT

Kademlia is a distributed hash table (DHT) for decentralized P2P computer networks. It specifies the structure of the network and the exchange of information through node lookups. Kademlia nodes communicate among themselves using UDP or TCP. Streembit primarily uses TCP. A virtual or overlay network is formed by the participant nodes. Each node is identified by a number or node ID. The node ID serves not only as identification, but the Kademlia algorithm uses the node ID to locate values. The node ID provides a direct map to message hashes and that node stores information on where to obtain the file or resource. We selected this implementation because using public/private key cryptography allows a simple yet robust authentication and access control (i.e. to identifying the node by verifying the public key based signature on messages). Having the public key validation integrated into the DHT layer allows the filtering of malicious nodes prior the messages reach the application layer.
We extended the generic Kademlia protocol and Streembit adds an extra security layer to the DHT to validate and authenticate messages using the ECC public key of users and devices.

The Streembit DHT primary role is to manage user/device discovery on the Streembit network. One of the main problems in IOT device management is how to share the characteristics of a device, like serial number, manufacturer and model with other devices or human users. Sharing device information is required for installation, data retrieval, device management, and device control. Streembit is a network that provides a means to safely install, configure, find and connect massive amounts of IOT devices together, and at the same time minimizing the risk that devices get hijacked. In order to achieve this the Streembit network implements distributed registry that allows simple access to private and public devices without risking their integrity.

Node ID

The node ID of the nodes could be calculated different ways. The most common method is to calculate the node ID based on the public key of the node. The system hashes the public key using SHA-1 to create the node ID.
Certain nodes can create the node ID based on their IP address and port that the Streembit application uses. The node ID is the result of the SHA-1 hash of the IP address and port.

Contact management on the Streembit network.

From security, authentication and access control standpoints contact management and how users find their contacts (i.e. finding other peer nodes on the Streembit network) are key functions in Streembit. Client-server, centralized

systems such as Skype or Signal provide users with convenience in contact management. However, users pay a heavy price for such easy-to-use and convenient systems by giving up their privacy to the centralized application providers and their associated government surveillance entities. Hardly a day goes by without news headlines that hackers, cyber criminals and industrial spies compromised the network of centralized providers. On the other hand the permissionless, cryptographically secured and decentralized Streembit DHT isn't an economical attack target. Attackers can deny access for millions of users by paralyzing a DNS server. To succeed with an attack on a P2P network the attacker (e.g. a hostile government) must deny millions of peers one by one. Since there is no central service provider in a P2P network each of the nodes must be attacked directly. This makes the attack on a P2P network highly uneconomical. To secure the network and communication we aim to keep the communication between contacts truly peer-to-peer. This makes less feasible that government surveillance, industrial spies and criminals could compromise the network and data.
There are the following base premises in Streembit P2P contact management:

- The peers who wish to communicate with each other must exchange their PKI public keys to enable ECDH key exchange methods
- All data exchanges must be signed to guarantee data integrity and encrypted to guarantee privacy.
- The contacts publish to each other their network information that is typically their IP address and port
- The public DHT storage can hold only encrypted contact related data
- Communication and access to contacts can be terminated any time

There are two methods to establish connection to contacts:

a) Offline contact request. Users create a contact offer by clicking on the Contacts/Create offline contact menu item, and exchange contact offers offline by sending it to each other by email, external chat, phone SMS, postal mail, HAM radio, etc. The contact offer is accepted by clicking on the Contacts/Add contact menu item.
b) Centralized contact request. This method uses the configured WebSocket hub to find the contact by clicking on the Search magnifier icon of the Contacts tab.

Naturally, we strongly recommend that users make connection via method "a". During method "a" the peer creates an offline request A typical offline contact offer appears the following.

```
{
  "account": "testuser01",
  "public_key":
"45GnKeY77xct9BT3Hz55Db111mSEBKB6ARWbXctoB3UR8p75GYjtMhmx6CSZjRENvYdGG…",
  "transport": "ws",
  "port": 32318,
  "host": "www.streembit.org",
  "user_type": "human"
}
```

This contact offer is wrapped in a base64 string in the following format (the actual content is shortened here):

```
--- BEGIN STREEMBIT CONTACT OFFER ---
eyJ0eXAiOiJKV…..
--- END STREEMBIT CONTACT OFFER ---
```

Streembit Security

The cornerstone of Streembit security is elliptic curve public/private key cryptography infrastructure (PPKI). PKI allows the implementation of robust security. We use PKI to identify the entities of the system (based on their public key and PKI signature), perform authentication, and ensure data integrity (using cryptography signatures).
Using the public/private key cryptography infrastructure paves the way for the deployment of a robust, secure authentication and access control scheme. The parties are identified by their public key. The authenticity of messages and the identities of the actors are then verified using PKI cryptography routines. Using a PKI

infrastructure based security scheme greatly simplifies the authentication, access control, and identity management aspects of Internet-of-Things security.
The elliptic curve cryptography (ECC) scheme is particularly suitable for IoT devices. The small footprint of ECC allows security modules to be implemented on embedded devices. ECDH is a trusted and proven key agreement protocol. Using the ECDSA digital signature algorithm ensures data integrity.
The system performs public/private key infrastructure based authentication and access control functions to securely connect machines to machines or machines to humans. The Streembit security module of the IoT device must generate at least one public/private key pair for each connected IoT device. Human users generate their ECC public/private key pair when they create their account. The actors of the system publish their public keys to the network, where each entity knows the public key of the other connected human or machine. The system identifies each of the entities by their ECC public key.

The collision resistant SHA-256 hash function is used to create a hash of data which can be signed using the private key to guarantee data integrity, as well as provide information about the originator of the data. To ensure data integrity the messages, control commands, event data, requests, and responses are signed with the ECC private key; the signature can be verified using ECDSA. To ensure integrity of data, each entity must sign all messages with their private key. The messages in the Streembit network are based on the following standards: JSON Web Token (JWT), JSON Web Encryption (JWE), and JSON Web Signature (JWS).

The basic premises of the Streembit security are

- Human users and Internet of Things devices use public/private key (PPK) infrastructure and PPK cryptography functions to secure messages
- Each actor of the system must generate a public/private key pair. (Typically keys are generated prior to configuring the device and will be burned into the devices' firmware).
- The device or user publishes the public key to other users of the system.
- The data integrity and authenticity of the messages is guaranteed with PPK signatures.
- Each session between users is secured with strong symmetric cryptography keys.
- All messages between users are secured with 128-bit and 256-bit AES symmetric encryption/decryption.
- The system uses elliptic curve Diffie Hellman (ECDH) key exchange algorithms to facilitate the exchange of session keys.

Protecting the ECC public/private key pair

The ultimate issue of all applications, systems and devices that encrypt data is the protection of the cryptography symmetric key or protection of the private key of the public/private key pair. Similarly to the iPhone and PGP, which both use passcode to protect their keys, Streembit requires users have a passcode to protect their PKI private key. The Streembit passcode that protects the user's PPK key pair is relatively complex and it requires the following minimum number of iterations with a brute force attack:

$$26^2 \times 10 \times 33 \times 95^4 \times \frac{8!}{4!} \approx 3.05 \text{ x } 10^{16}$$

That means 3,050,000,000,000 (3.05 trillion) times more iterations than the 10^4 iterations is required to break the latest iPhone security. (Of course if the passcode is longer than 8-digit then the number of iterations is even more). This is related to protecting the PKI key pair. To protect real time communication, Streembit uses a randomly generated 256-bit AES session key that is exchanged between peers via ECDH. This means that real time communication, which is the main functionality of Streembit, is secure even from the brute force attacks of NSA super computers and large zombie computer clusters of cyber criminals.

Mitigate the risk of Sybil attacks

A known issue with regards to P2P networks and their practical limitation is that they are frequently subject to Sybil attacks. This means that malicious parties can compromise the network by generating and controlling large numbers of shadow identities. A malicious node may present multiple fake identities to a P2P network in order to appear and function as several distinct nodes.

A Sybil attack is most effective on anonymous and reputation systems where if the malicious nodes outnumber the honest nodes the outcome of the application could be compromised. For example, on a file sharing network a successful horizontal Sybil attack allows the attacker to sniff most of the control messages, hijack the system, and deliver bogus content.

Contrarily, Streembit facilitates communication between contacts that you know, such as your family members, teammates, and business partners. You are also aware of the location and identity of your Internet of Things device(s) you control via Streembit, lessening the likelihood of a Sybil attack. Malicious nodes, regardless of their weight and presence on the network, are unable to change the public keys of your contacts (as the public key is based on the cryptographically secured PPK infrastructure).

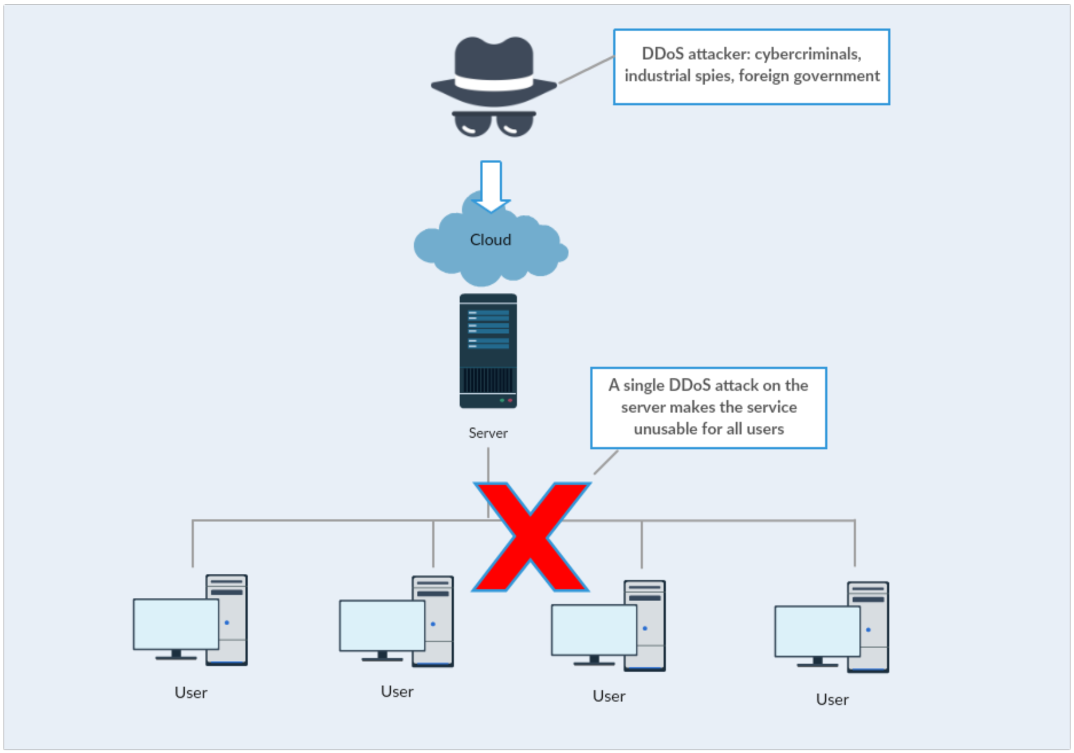
Additionally, Streembit introduces another layer of security in the form of private networks. Using private networks, the public keys of the contacts are loaded to the seed nodes manually; mitigating further the risk of a Sybil attack as well as a Man-in-the-middle attack (MITM). A Sybil attack, which generally speaking is a serious security issue for P2P networks such as file sharing applications, is much less of a problem for the Streembit P2P system.

Mitigate the risk of DDoS attacks

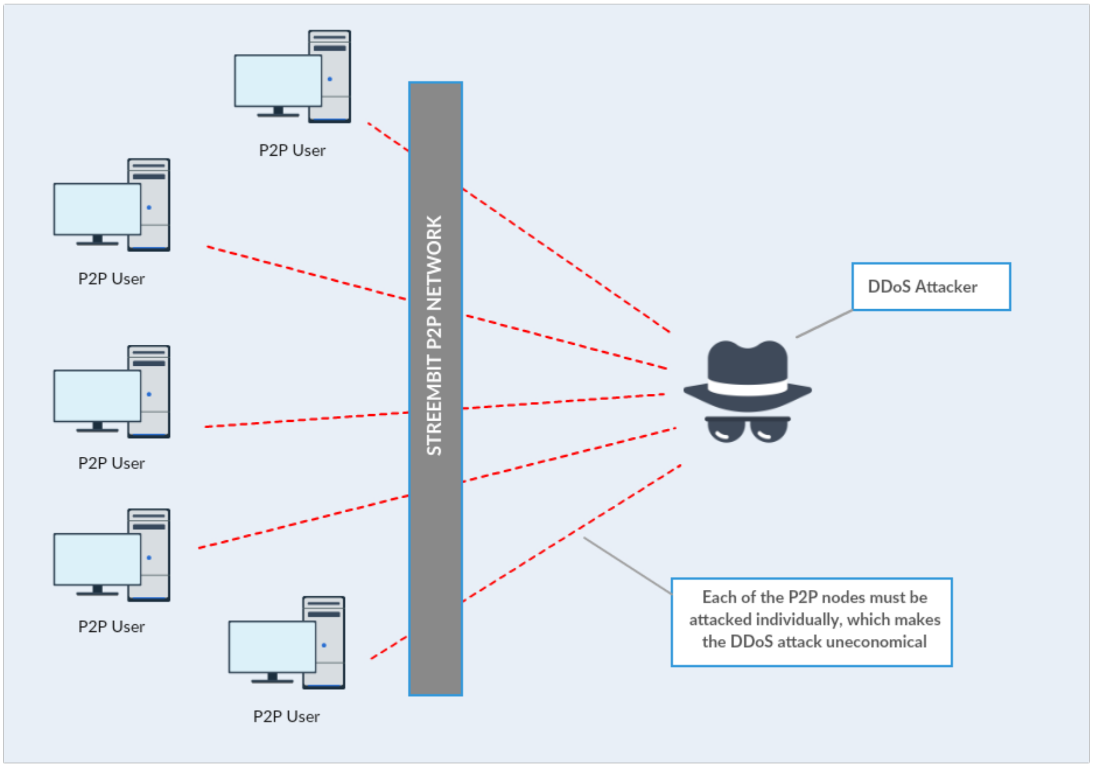
Distributed denial-of-service (DDoS) attacks are designed to knock web sites, systems and services offline. For instance, the enormous power of the September 2016 cyber-attacks paralyzed the Internet along the US East Coast. The attack created problems for Internet users and afflicted an array of sites including Twitter, Amazon, Tumblr, Reddit, Spotify and Netflix. It was an extremely large and unusually distributed denial-of-service (DDoS) attack designed to knock many popular sites and services offline. The power of first attack reached 1.1 Tbps while a follow-on attack was measured at 901 Gbps. The DDoS attacks were delivered through a collection of hacked Internet-connected cameras and digital video recorders. No centralized, client-server network can withstand such powerful attack. It is no wonder the attack left millions of Twitter, Amazon and Netflix users without software services.

We argue that the **only logical and available solution against such attacks is a decentralized P2P system**. DDoS attack on a decentralized, P2P network wouldn't be effective – no wonder Bitcoin has never experienced a network outage since the inception of the network. In order to paralyze many millions users of a client-system server the attackers have to concentrate their attacking power only on the centralized server. Once the attackers take out the server the clients (users) are without service. It is a relatively simple and doable task from the attacker's viewpoint and a very economical one as well - one concentrated attack can affect millions of users. However, since there is no central server present in a P2P network, to deny the service the users of a decentralized, P2P network would have to be attacked individually. **The DDoS attacking power would have to be divided to attack each peer separately. This defies the economy of the attack.** Therefore, the configuration and orchestration of such attack on a P2P network would be impractical and highly uneconomical.

Attack on centralized services


DDoS attacker: cybercriminals, industrial spies, foreign government
Cloud
Server
A single DDoS attack on the server makes the service unusable for all users
User
User
User
User

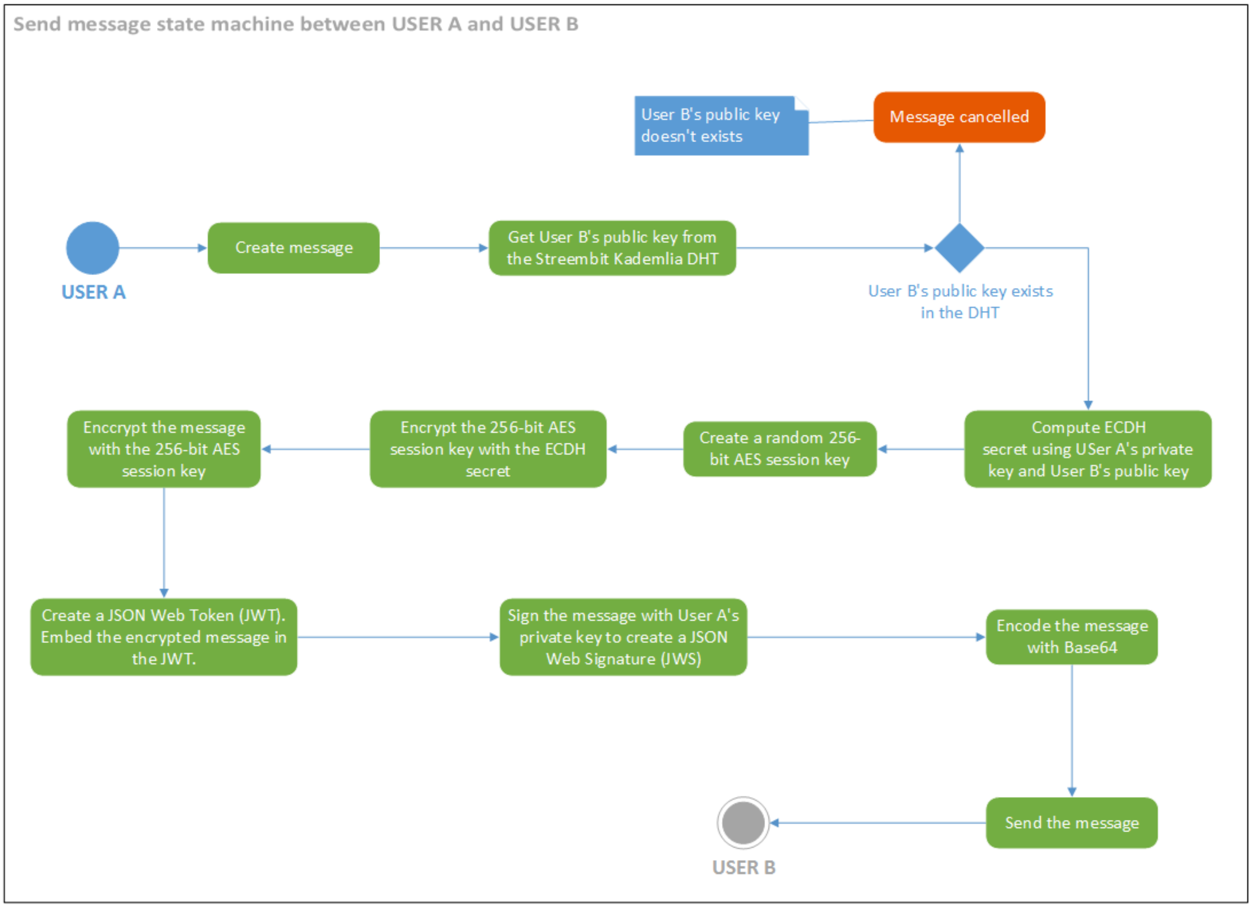
Attack on a P2P network


P2P User
P2P User
P2P User
P2P User
P2P User
STREEMBIT P2P NETWORK
DDoS Attacker
Each of the P2P nodes must be attacked individually, which makes the DDoS attack uneconomical

How secure the cryptography of Streembit?

Both 128-bit and 256-bit AES symmetric encryption, the cryptography schemes of the messages in the Streembit network, are safe from any brute force attacks. Bruce Schneier points out: "*These numbers have nothing to do with the technology of the devices; they are the maximums that thermodynamics will allow. And they strongly imply that brute-force attacks against 256-bit keys will be infeasible until computers are built from something other than matter and occupy something other than space.*"

The following UML diagram describes the key exchange using ECDH and securing the message with 256-bit AES symmetric key between "User A" and "User B"


Send message state machine between USER A and USER B
User B's public key doesn't exists
Message cancelled
USER A
Create message
Get User B's public key from the Streembit Kademlia DHT
User B's public key exists in the DHT
Enccrypt the message with the 256-bit AES session key
Encrypt the 256-bit AES session key with the ECDH secret
Create a random 256-bit AES session key
Compute ECDH secret using USer A's private key and User B's public key
Create a JSON Web Token (JWT). Embed the encrypted message in the JWT.
Sign the message with User A's private key to create a JSON Web Signature (JWS)
Encode the message with Base64
Send the message
USER B

Internet-of-Things Device Handling

It is troubling that robust security was not taken into account with the majority of Internet of Things implementations. The data of IoT devices are being harvested in an automated fashion but who has access to the data? What functions can a human or another machine execute on the IoT device? Is my office door actually being opened by a former employee who is not supposed to enter the premise anymore? Is the data being forwarded by the IoT device compromised at all during its way to the end-user, never mind whether or not it was sent by the actual device and not in fact by an intruder?

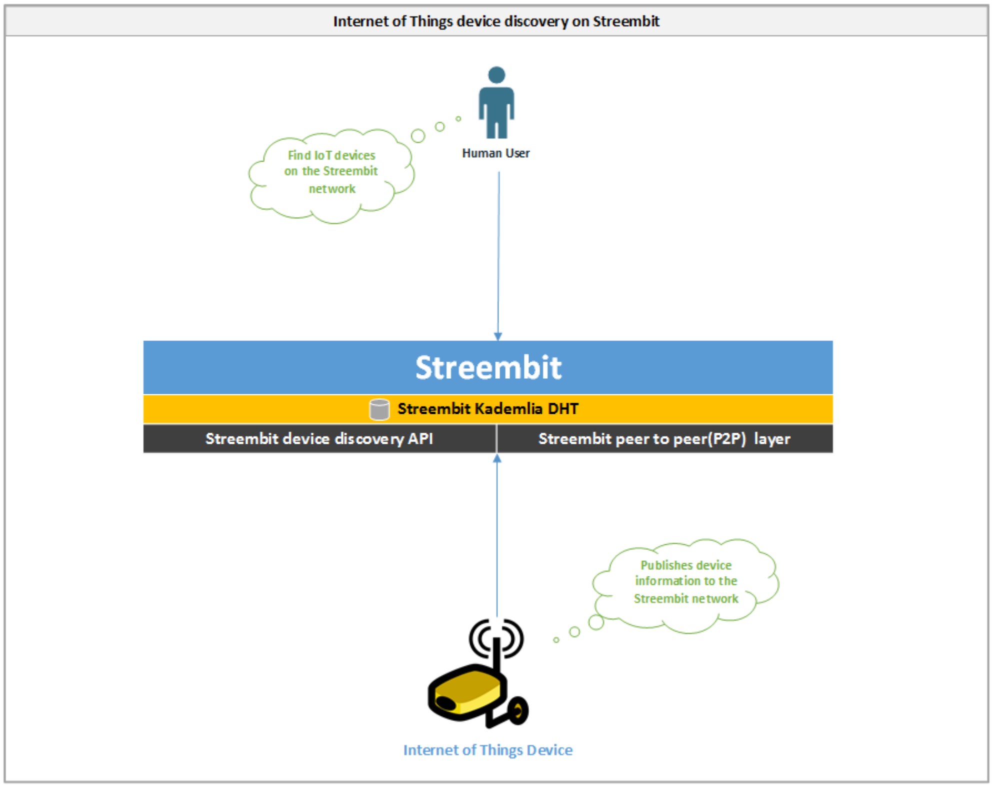
Streembit manages device discovery as well as authentication, access control and provisioning of devices without using a centralized authority server. The system facilitates device discovery and device control in a P2P manner. For many IoT use cases a decentralized, P2P network topology is the most secure, robust, scalable and reliable method of operation. Instead of using expensive corporate owned third party cloud systems, Streembit manages Internet of Things devices without the need of proprietary centralized cloud platforms.

Centralized, corporate owned cloud is certainly an easier way to build out IoT platforms especially for large scale networks. However, the owners and authorities in these topologies all have an influence upon the network and can be exploited. They can ban devices, spy on devices and compromise data integrity of devices. In fact, governments can order them to do any or all of these. In the near future, the doors of your garage and home, air conditioning units and your home security system will be fully internet connected. You will be able to control your home automation system from your mobile phone. It is essential that only you can control your IoT devices. Streembit excludes third party service and cloud providers from the ecosystem to give full control to the end users over the devices.

The Streembit IoT implementation is based on open standards. The Streembit developers contribute to the standardization process of W3C Web of Things Interest Group and mirror all W3C open IoT standards in the Streembit source code.

Internet of Things device discovery on Streembit

Streembit allows IoT application providers and context producers to register their IoT Objects on the decentralised Streembit network, and in turn allow context consumers to discover them in a secure and peer to peer manner.

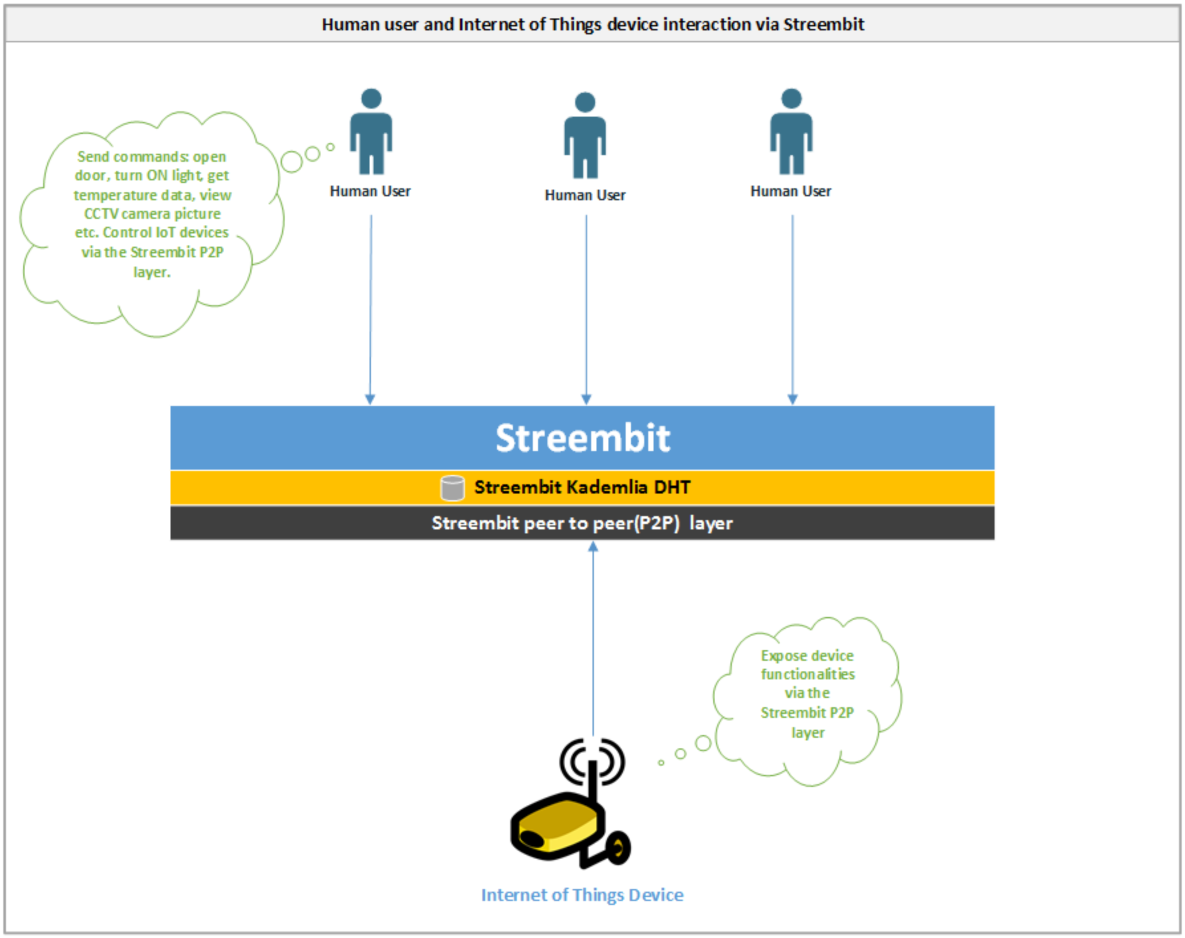

Internet of Things device discovery on Streembit
Find IoT devices on the Streembit network
Human User
Streembit
Streembit Kademlia DHT
Streembit device discovery API
Streembit peer to peer(P2P) layer
Publishes device information to the Streembit network
Internet of Things Device

Example data structure for device information:

```
{
    "account": "058dd8eb72d1936df31eebdd9afb49e1415787e3",
    "public_key": "03e9a353b86cc482af497a8afb56209ee35eaa80bee228c2af8ce48747b90c50e7"
}
```

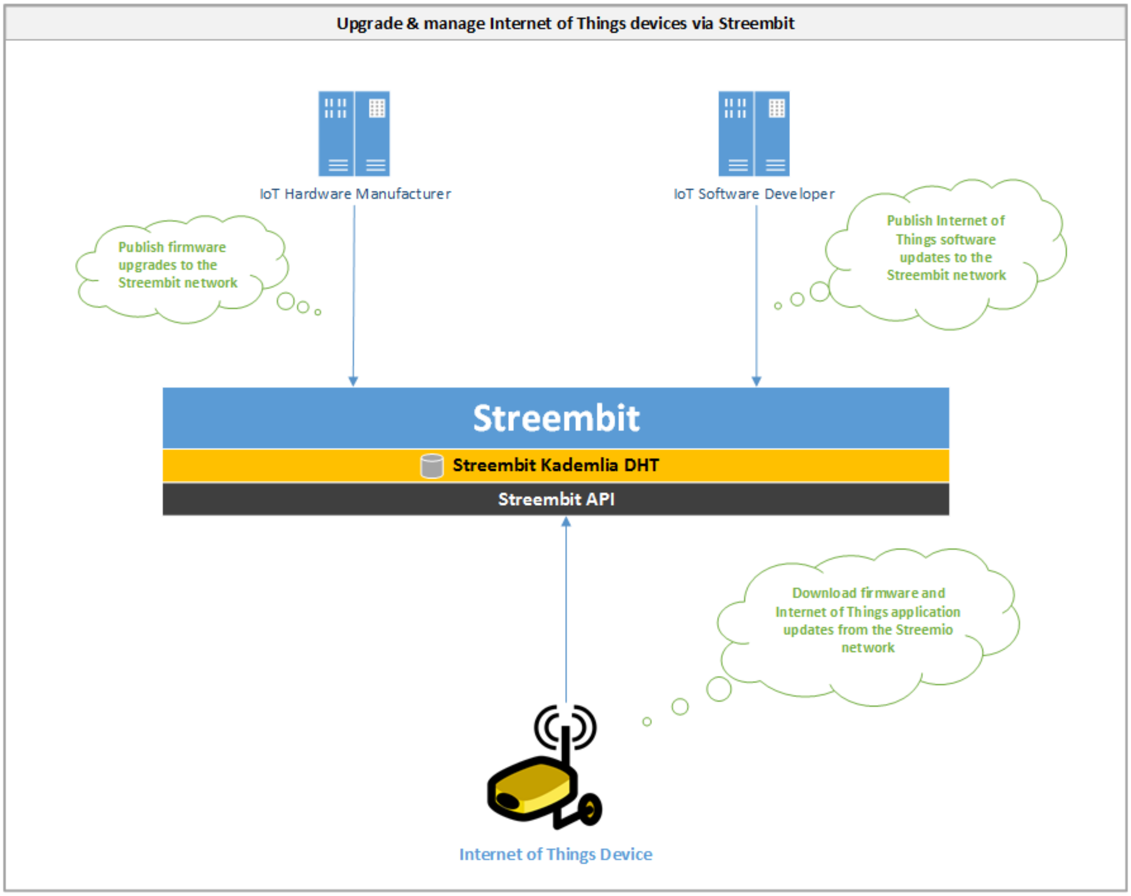
Control Internet of Things devices via Streembit

- The Internet of Things devices and human users communicate with each other directly in a peer to peer manner
- The data is end to end encrypted between the human user and IoT device. The encryption key is shared only between the user and device – never with any third parties.
- The device exposes functionalities via the Streembit network and user interface using W3C WoT standards
- The user interacts with the device via the Streembit P2P layer and UI. For example, opening a door, turning ON a light, controlling motor speed, getting temperature data, viewing CCTV pictures etc. all peer to peer, without a centralised solution and via the Streembit user interface.


Human user and Internet of Things device interaction via Streembit
Send commands: open door, turn ON light, get temperature data, view CCTV camera picture etc. Control IoT devices via the Streembit P2P layer.
Human User
Human User
Human User
Streembit
Streembit Kademlia DHT
Streembit peer to peer(P2P) layer
Expose device functionalities via the Streembit P2P layer
Internet of Things Device

Upgrade and manage Internet of Things devices via Streembit

- Hardware and software providers upgrade Internet of Things devices on the always up and running Streembit network.
- Internet of Things device manufacturers and software designers publishes firmware and software updates via the Streembit network.
- Internet of Things devices run the Streembit system and ensure the origin and data integrity of the updates by verifying the public key of the publisher.


Upgrade & manage Internet of Things devices via Streembit
IoT Hardware Manufacturer
IoT Software Developer
Publish firmware upgrades to the Streembit network
Publish Internet of Things software updates to the Streembit network
Streembit
Streembit Kademlia DHT
Streembit API
Download firmware and Internet of Things application updates from the Streemio network
Internet of Things Device

Interface with legacy client-server systems

Why should anyone switch to the completely new decentralized, P2P topology from the well tested client-server paradigm? Why should a business throw away the existing infrastructure and invest into a new one? After all, the businesses have already invested into MS SQL, PostgreSQL, etc. data stores. Businesses use already centralized data mining and data analysis applications; client portals to serve end users via web applications - why should any business throw that away?

We don’t suggest that businesses replace the existing system with a completely new one. Streembit complements and fits into your existing IoT infrastructure. It takes away the heavy load of device handling from existing centralized server based application.

A central server that connects to IoT devices is a mission critical component. Centralized servers that communicate with devices are a single point of failure. Any mission critical components must address scalability and high availability business requirements using expensive load balancing, clustering hardware and software applications.

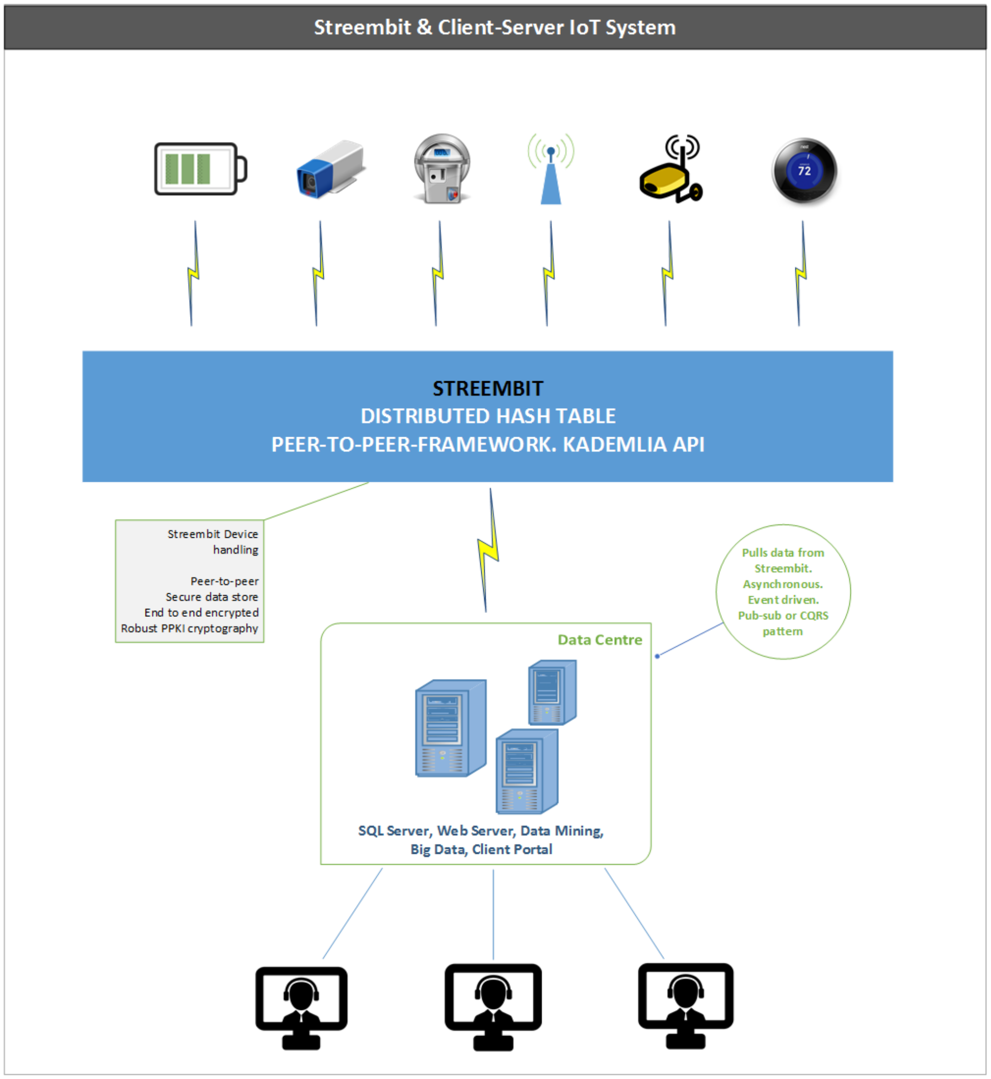
Also, a centralized server that receives device communication is a convenient target for cyber criminals and attackers.

Alternatively, IoT devices can publish the data to the **always available and economical** Streembit distributed hash table rather than writing the data to a centralized web service or TCP/IP server. Then, the server can pull the data from the distributed storage. In this loosely coupled system the central server doesn't have to be a mission critical, 99.999% uptime component. It can be a lightweight system that pulls the data from the distributed storage. The topology we propose is an event driven, asynchronous mechanism, pub-sub messaging and it implements the separation of concerns (SoC) design principle. The devices publish the data to the distributed storage and aren't concerned about the existence of the server. In such a topology the central server is not a single point of failure in the system anymore.

Streembit can manage in this set-up:

- Secure device authentication and access control
- Secure messaging between the device and server
- Secure device upgrade and management

We propose to delegate device handling tasks and processes to the P2P network. Device management can be done via P2P and without the need of a central server.


Streembit & Client-Server IoT System
72
STREEMBIT
DISTRIBUTED HASH TABLE
PEER-TO-PEER-FRAMEWORK. KADEMLIA API
Streembit Device handling
Peer-to-peer
Secure data store
End to end encrypted
Robust PPKI cryptography
Pulls data from Streembit. Asynchronous. Event driven. Pub-sub or CQRS pattern
Data Centre
SQL Server, Web Server, Data Mining, Big Data, Client Portal

References

1. https://github.com/w3c/web-of-things-framework
2. https://github.com/w3c/web-of-things-framework/blob/master/security.md
3. http://bit.ly/2nIZp7c
4. http://www.mindspring.com/~dmcgrew/gcm-nist-6.pdf
5. http://xlattice.sourceforge.net/components/protocol/kademlia/specs.html
6. https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf
7. https://en.bitcoin.it/wiki/Blocks
8. https://www.igvita.com/2014/05/05/minimum-viable-block-chain
9. https://arxiv.org/abs/1607.01341 ALGORAND, The Efficient Public Ledger
10. https://arxiv.org/pdf/1612.04496.pdf Smart Contract Templates: essential requirements
11. https://arxiv.org/pdf/1608.00771.pdf Smart Contract Templates: foundations
12. https://en.bitcoin.it/wiki/Contract
13. https://en.bitcoin.it/wiki/Smart_Property
14. https://en.bitcoin.it/wiki/Transferable_virtual_property
15. Priyanka Ravindra: The extended Shapley value for organizations
17. Quiet enjoyment, by Wes Kussmaul, PKI Press, Waltham, Massachusetts 02451
18. http://www.cs.ox.ac.uk/people/michael.wooldridge/pubs/ieeeis2012d.pdf
19. https://www2.le.ac.uk/departments/npb/people/amc/articles-pdfs/reasabou.pdf
20. https://www.oreilly.com/ideas/blockchain-scalability
21. http://math.mit.edu/~dspivak/informatics/notes/unorganized/CTDB--spivakCurino.pdf
22. David I. Spivak: Category theory for the sciences
23. Cooperative Game Theory; Georgios Chalkiadakis, Edith Elkind, Michael Wooldridge
24. https://cointelegraph.com/news/paypal-cfo-says-merchants-arent-interested-in-crypto-due-to-volatility
25. https://www.investopedia.com/tech/goldpegged-vs-usdpegged-cryptocurrencies/