commun /

Commun

Social network of the people, by the people, and for the people
(version 1.0., 29 November 2019)

Commun is the next generation social network, where users can create interest-based communities and share content, read, watch, rank it within a community of like-minded people. The communities set their own rules, reward users for activities and gain from selling advertisements.

Abstract

The current trend in social media is a shift towards communities. Facebook announced lately its emphasis on groups, Reddit is growing successfully due to its focus on interest-based directories. However, the centralised services like Facebook have compromised themselves severely in terms of privacy, and there are no prerequisites that it will change in the future. The other problem with the centralised solutions is that their users de facto are used (or better to say abused, as they lose the rights of their content at the moment when they upload it to the platforms and do not earn anything in exchange for their 'service', be it writing or reading activities) to make richer the stakeholders of the companies. The blockchain technology, however, offers a solution to these problems. The idea of decentralised social media platforms was tested in the past years, check steemit.com. But these platforms proved to be very unsuccessful in terms of value creation. Logically, the next step in terms of blockchain-based social media projects should be an attempt to bring the best from the centralised solutions on blockchain preserving the advantages that blockchain offers in terms of security, rewards for content creation, etc. Commun is the next generation social network, where creators share content, and users read, watch and rank it within interest-based communities. Both categories get rewards in community tokens, and advertisers burn tokens for ads placement.

commun /

Introduction

The rapid development of social-purposed applications such as Facebook or Reddit led to the enormous growth of data. The uncontrollable ownership of the user data made social network projects to misuse them. The misuse stems from the fact that information uploaded by users is not protected. In 2018 the world learned that Cambridge Analytica exploited the personal information of dozens of millions of Facebook users without their knowledge or permission. It appears the information collected included everything these individuals had on their Facebook pages and, according to some reports, even included private direct messages between users.

Another well-recognised problem is that social networks are perceiving users as the 'means' in the revenue process. Users' behavior is turned into data. These data are subjected to sophisticated analyses to make behavioral predictions, to turn these predictions into products and to earn on them.

The recognised solution is a move towards social network applications built on top of decentralised databases and governed by users. Such solutions range from trivial chat protocols to complex social communities. Among others are Diaspora, a Medium-like network, launched in 2014, built on open-source servers, Minds, a Medium-like network, launched in 2015, currently running on Ethereum mainnet, Mastodon that operates on open source servers and is more Twitter-like. Steem, a Medium-like network, launched in 2016, built on Graphene protocol. The networks built on blockchain protocols have reward pools.

However, the first generation of decentralised platforms built on blockchains severely suffered from technical problems. First, data growth on these platforms led to a mess as there were no mechanisms of efficient content categorisations. The popular 'tag' solution did not solve the problem. Better tuned categorisation turned out to be required. Second, the reward pool in these systems consisted often of one token in different forms that severely hampered the value creation. Third, in the most popular systems like Steem and Golos it was impossible to introduce any significant changes in the application business logic without hardforks.

Preliminaries

This particular section discusses previous social application protocols available and explains why they are not suited well to the demanding from a technical perspective social networks.

The first social-purposed database replication protocol was introduced back in 2012 and it was called Bitmessage. This protocol meant to become an e-mail replacement and was based on the non-authenticated transaction log. Spam protection purposes required some consuming computations to be done on the client side, so this protocol has not gained popularity it deserved.

Another significant social application protocol was introduced back in 2016, and it was called Steem. This protocol introduced the public content platform, paying rewards for the most liked content. In comparison to other open-source decentralised social media platforms Steem enjoyed an explosive growth due to its functionality (anonymity, free to use business model, rewards for writing and reading activities, satisfying transaction speed, organic growth of viewers due to public storage of data). However, the in-built reward system with a single base token led to abuse, and explosive growth of content amount produced a problem of categorization.

The rewarding mechanism of Steem was based on reward pools, paying users out the sums calculated, taking into account the amount and strength of the post's upvotes and downvotes. The system rapidly got "content-boosting" services, which actually used most wealthy accounts to force non-content-quality-based reward pool redistribution. This was a classic example of the tragedy of the commons, when users acting independently and rationally according to each one's self-interest started behaving contrary to the networks long-term best interests.

The most significant attempt to categorise content by its' type was presented by Steem in 2017, and it was called Smart Media Tokens, abandoned later. This extension to Steem protocol was supposed to provide an opportunity to create a group with an isolated economy similar to the parent's one. However, this meant no customized emission rules, no customized content-management, and no significant customization of reward distribution.

The customization requirements needed an imperative description system, which was only implementable via virtualized turing-complete storage consistency conditions. Such a feature turned out to be implemented in various existing protocols (e.g. Ethereum or EOS). Some of them (EOS) turned out to be implemented with the same framework Steem used.

Closer consideration of these protocols showed up several problems. Ethereum's protocol turned out to have a limited programming language, used to define the application logic (mostly because of very limited complexity of structures, the inefficiency of generated bytecode and lack of availability to use complex software from the outside of Ethereum ecosystem), and low replication protocol throughput coupled by high bandwidth cost. EOS's protocol has problems as well. Particularly, these are the signs of not being ready for processing large amounts of data, which are usually being produced by social applications due to highly limited bandwidth rate available, low processing throughput, and undefined write-in/finalization time.

Proposal

Commun is a platform for building autonomous self-governing communities based on smart contracts of the CyberWay Blockchain. The CyberWay Blockchain fixed issues of Ethereum,

EOS, and Steem described above. Unlike Facebook and Reddit, Commun uses the permissionless and decentralised CyberWay blockchain. Everybody can join the CyberWay blockchain and Commun does not have control over it. The users can create their Communities and set up their own rules within the Community within Commun.

Our mission is to enable individuals to build and run self-governing online communities that are to be monetized by users easily.

Such a goal can be achieved only if the following points are observed:

- User data management is verifiable by users themselves
- User data security services are verifiable and manageable by users themselves
- Community affiliation is not enforced by technical or financial bounds

1. Management

As no community is capable of existing without some disputes and misunderstandings, there is a need for leaders who could be final judges in case of a conflict. Our platform includes in-built mechanics of selecting and ranking "leaders" for each community. Any user has a right to choose leaders of the community, where he is a member. The Commun also assumes remixability: instead of fighting for power and influence, any user can leave a group where he does not feel belonging or welcome and create a new community.

2. Rules

Communities require public, transparent, and understandable rules. The rules are critically important. Being well-drafted, they eliminate tension, but improperly composed or ignored provoke disputes and difficulties that could scare away community members. Commun allows members of each community to set the rules for the community. These rules are written in the blockchain, which makes them public for everybody. Their interpretations in different situations well could be added and written down to the blockchain as well. The rules could be changed if there is a consensus between the leaders that express the will of their voters that are users.

3. Self-reliance

Communities thrive to control better finances at their disposal. If you keep a community on a dry ration, it could become more creative, but the key participants would be emotionally burned out. If you give it too much money, a community could lose its diversity and creativity. Since the Commun is blockchain-based, one of the main mechanics is the emission of points and the creation of a common pool (a fund, if you like). The distribution of funds from this pool can be customized according to the wishes of the community itself. For example, the pool can be used for

- Software development and updates;
- Rewards to authors and readers/viewers;

- Contributions to community leaders.

At the same time, blockchain technology provides a transparent distribution and expenditure of funds.

The construction of such a platform became possible using blockchain smart contract technology. Such an approach allows to ensure guaranteed fairness and transparency in all processes occurring within the community.

4. Economy of Commun

To be sustainable in the long run, Commun has to be built in a way when there is a match between supply and demand of Commun tokens.

The key idea of Commun is to create an environment for the competition of diverse interest- or locality-based communities providing them with all necessary resources like content sharing and evaluating mechanics, easy community setup and free to use model, community governance, and economic parameters. The successful communities have opportunity to grow and make money on advertisement.

To be active on Commun there is no need to have any Commun tokens (CMN). The Commun tokens are used to create a community via bonding curve mechanics. Those users who want to have a bigger say in governing the chosen communities to select leaders or to promote their posts will be able to buy Commun tokens and exchange them to any community tokens.

The total supply of Commun Tokens is originally limited to 4 500 000 000 Communs. Each Commun token is subdivided into ten thousand (10^4) smaller units. This supply will start to increase slowly after one year from Commun Social will be launched , and will be used as rewards to Commun Leaders. Commun will produce an inflation rate of 1% per year.

To provide liquidity to the communities, the bonding curve mechanism is introduced. Unlike a regular exchange, where a transaction takes place between users who have reached a price agreement, in the bonding curve algorithm, a user purchases tokens from a special smart contract. When purchasing, a corresponding amount of community tokens is emitted, a fee in the form of reserve tokens, in this case, Commun tokens, is stored on the balance of the smart contract. Similarly, when selling occurs, the user does not need to look for a customer. The community tokens are returned to the contract, thus withdrawn from the circulation. The user gets the appropriate amount of Commun tokens in return.

The advertisement mechanic that is used in Commun is Harberger’s tax economic policy that balances pure private ownership and total commons ownership with a goal to raise the general

welfare of society. The buyer purchases a banner and sets a price for a banner, and every day pays a tax rate percent for holding a banner by burning community tokens. Anyone can buy the banner for a price that is set at any moment and start paying taxes in turn. If the ad buyer does not pay a tax, the price goes to 0.

5. Bonding Curve Mechanics

In the bonding curve scheme, there is no traditional mechanism for studying the equilibrium price of an asset in the form of supply and demand. It is replaced by the formula according to which the price is a function of the current number of tokens in circulation (p = p (S), where p is the price, S is the current quantity of Community Points). Due to this, users can at any time be guaranteed to buy or sell an arbitrary number of community points for Commun tokens at a projected price. Any purchase of community points increases the total volume of these tokens, while the sale reduces it. Thus, the user's actions automatically affect the price of community points.

The following parameters influence the formation of future Community Points price:

Initial Supply (IS) of Community Points. The number of Community Points that will be originally issued when creating community.
Initial Reserve (IR) of Commun Tokens. The number of Commun Tokens that will be reserved while issuing of Community Points.
IS and IR are the parameters by which the initial price of the community points is set.
Connector Weight (CW): The CW parameter determines how quickly the price reacts to the change in the amount of tokens as a result of the purchase, i.e. it determines the concavity of the function p (S). When CW is close to zero, the price increases sharply, and at cw = 100%, it is constant.
Fee: The commission that the user must pay when converting Commun Token to Commun Point.

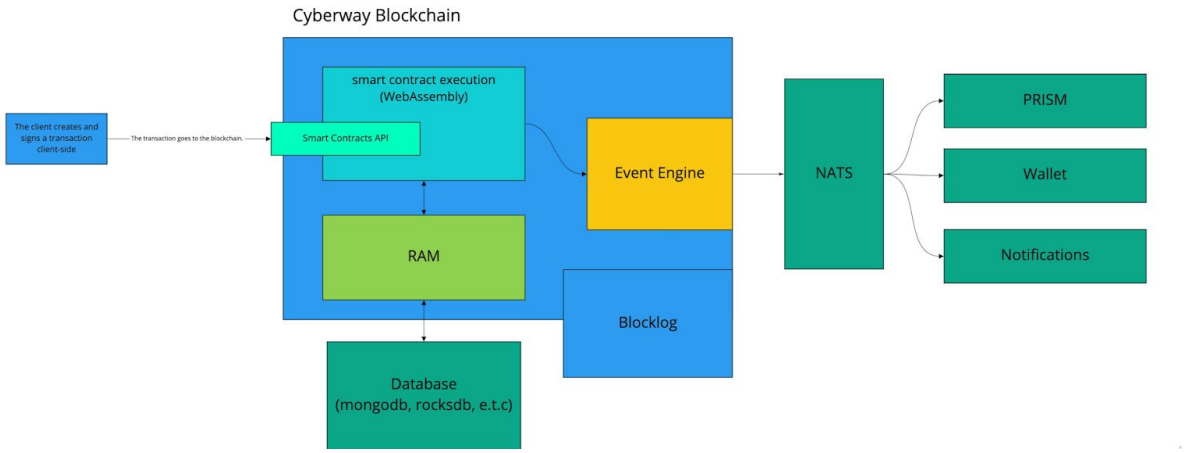
Certainly, the Bancor algorithm is not perfect, and some authors criticise it. Among the drawbacks mentioned, there is a vulnerability to a frontrunning attack: since with almost simultaneous buying/selling of Community Points, prices depend on the order of operations, and any tokenholder of sufficient amount of Commun tokens can arbitrarily change it to his advantage. To prevent these malicious actions a commission is charged for selling community tokens in Commun. Thanks to it, this attack with small amounts becomes unprofitable, and with big amounts attracts attention, that could translate into reputational costs for the attacker.

Also, critics note that a smart contract with a fixed formula, unlike the real market is not flexible and does not respond to information. With a sharp increase in price, the first buyers will purchase community tokens at a lower level, making a smaller reserve amount than the rational

sellers would agree to. When the cheapened Community Points are sold, there is an accelerated depletion of the reserve. Though a quick sale is possible only by those users who have not blocked their tokens to receive rewards for curating. The opportunity to react quickly to changes in prices is offset by a dilution of user's share due to emission.

However, for the low liquid tokens, the benefits of the Bancor algorithm outweigh the drawbacks mentioned. Moreover, with a growth in popularity some communities would be able to list their tokens on the regular exchanges. In the future the Community Point contract can be improved by combining bonding curve algorithm with a traditional exchange model. For example, Bancor algorithm could be combined with an idea of the Equilibrium Bonding Market, a limit on the amount of tokens to be updated during emission could be introduced. As a result, small communities will use Bancor algorithm, and larger ones will determine the price of their tokens on a market.

Implementation



Commun consists of:

- A set of smart contracts based on the CyberWay Blockchain
- A Set of libraries implemented on Swift, Kotlin, Javascript;
- A set of services implemented with NodeJS
- A set of clients: Web, iOS, Android.

The high-level scheme of Commun interaction with the blockchain is given on an illustration 1. As you can see Commun website and mobile apps are merely clients that allow to interact with the CyberWay Blockchain.

commun /

1. Application Description. Glossary

A “Commun Leader” is a person elected by Commun users by using their stakes of Commun tokens who governs Commun doing the following:

- Defining what and how it should be developed
- Initiating and approving updates

A “Community Leader” is a person elected by community members by using their stakes of community tokens who governs the community doing the following:

- Defining the rules of the community
- Moderating content feed (deleting or hiding a post from community feed, adding additional tags)
- Managing a community

A “User” is a person performing one or more of the following actions on a platform:

- Posting, upvoting and downvoting content
- Managing content
- Choosing leaders
- Creating a community

A community is a group formed around some point of interest. The community can be created by any user or group of users. Each community has its own parameters.

- Community Name
- Community Rules
- Community Description
- Community Avatar
- Community Cover
- Inflation rate (a value set between 5 and max value <=50%, step 5%)
- Distribution of reward in percentages between authors and curators from variants (25 to Author and 75 to Curator, 50 to Author and 50 to Curator, or 75 to Author and 25 to Curator). The current pre-set distribution of rewards in percentages between authors and curators is 45/45
- Max number of leaders
- Maximum possible number of Community Points

This list of parameters can be changed, if necessary, by updating the corresponding smart contracts. This feature is implemented at the blockchain level. The smart contract update rights are held by the Commun leaders.

2. Commun Smart Contracts

Commun consists of a set of interconnected smart contracts placed on the CyberWay Blockchain, which define interactions between users, communities and the CyberWay Blockchain. All smart-contracts are implemented and loaded into a blockchain once and not considered unique for each community.

2.1 Commun.list

It is a registry of available communities.

2.2 Commun.point

It is a registry of “Community Points”, which is also used to exchange “Community Points” into “Commun Token” and back. The power of voting for a post and the ability to get curator’s rewards depend on the number of community points owned by a user. The more “Community Points” the user has, the more rewards the user gives to the author when voting for a content and the more points the user himself earns on managing the content. “Community Points” can be exchanged into the Commun Token according to the rules of bounding curves.

2.3 Commun.gallery

The smart contract commun.publication provides work with posts, including the ability for users to perform the following actions: publish, edit, delete and vote for posts and comments. Turning to this contract, the user can establish the rules for distributing rewards to authors and curators and establish the end beneficiaries of the rewards. In addition, this contract provides calculation and payment of rewards to the authors and curators of posts.

2.4 commun.ctrl

The smart contract commun.ctrl contains the logic for selecting community leaders, including the procedure of registering an account as a leader, of voting for leaders, as well as of determining a list of top-rated leaders.

2.5 commun.emit

The smart contract golos.emit ensures the issuance of new “Commun Points” and their distribution in accordance with reward pool settings determined by the leaders.

2.6 commun.social

The smart contract commun.social enables the following: creating and editing user profiles (metadata); creating a list that allows its owner to receive information about publications of users and communities of interest to him; creating a "black" list that allows a user to block all the communications with the undesirable to him users.

3. Infrastructure Architecture

The most common problem of the overwhelming number of blockchain projects is scalability, and there are several reasons for this.

- Only applications with a small amount of users can rely on public nodes, since the load on these nodes is very high. With the explosive growth of traffic, product developers have to launch their own private nodes. If the growth continues, then the developers have to increase the number of nodes and skillfully balance traffic between them.
- In most cases, the blockchain software architecture is a monolith that consumes a huge amount of computing power and disk space. As a result each additional node significantly increases the cost of servicing the product. Moreover, the problems periodically arise that lead to node resynchronization, which might disable some of them for several days.
- A hard-coded API without the ability to change it quickly (each API change requires a soft-fork or a smart contract update) imposes additional restrictions on the use of data stored in the blockchain. Users have long been used to receiving notifications about events relevant to their accounts, for example, likes/flags, new comments on posts, etc. However, the blockchain node cannot provide such functionality, not to mention push notifications for mobile devices, email notifications, account verification and other equally important functionality.

Commun is developed taking into consideration all mentioned above. So, we revised fundamentally the approach of working with data and chose a decentralised Blockchain solution (CyberWay Blockchain) that suits our needs. When designing Commun, we set the following goals:

1. The design should be focused on the needs of the user, and the components themselves can be easily modified and/or replaced if necessary
2. The onboarding of new users should be simple and clear
3. The working speed of the application should be comparable to similar web resources
4. The application should scale easily. The infrastructure nodes of the system should be implemented as microservices so that, if necessary, they could be upgraded and updated easily.
5. The application could allow developers to add new API calls and update the used API easily.

The infrastructure architecture is supposed to be built with micro-service architecture approach. This means the core service set is defined as follows.

3.1 Core Services:

1. Gate-service. The entry point for data exchange between a client (whether it is a Web, iOS or Android application) and the infrastructure of microservices. It uses WebSockets interface with JSON-RPC protocol to communicate with the clients. The gate only deals with the forwarding incoming messages. Routing and validation of parameters is performed on the Facade-service.
2. Facade-service. It is a microservice of routing requests between the front-end and the microservices. To work the service requires a compatible frontend gate, which provides final communication with the frontend and checks for the need to authorize the user when processing incoming requests.
3. PRISM-service. It is a service of CyberWay Blockchain data dispersion, which can be used in applications after secondary processing by other microservices. It provides a flexible and extensible API for working with data.
4. Bandwidth-service. It is a service for providing the necessary bandwidth to users, for example, to publish the posts. The service signs up transactions after a user has signed them, thus enabling retaining of the authorship.
5. Registration-service is a service for signing up new users. Different registration strategies are supported, for example, the user sends sms to the specified number for verification (the safest, but the least user-friendly way), we send the user an sms with the code that he sends us (convenient and secure, but expensive verification), verification by mail, or verification through social networks. Also there are methods to dynamically select a strategy for a user, defining a server, including A-B testing.
6. Sms-service is an sms messaging service.
7. Auth-service is a service for authorized user requests to an application. To work, the service requires a compatible facade-service, which provides final communication with the system.

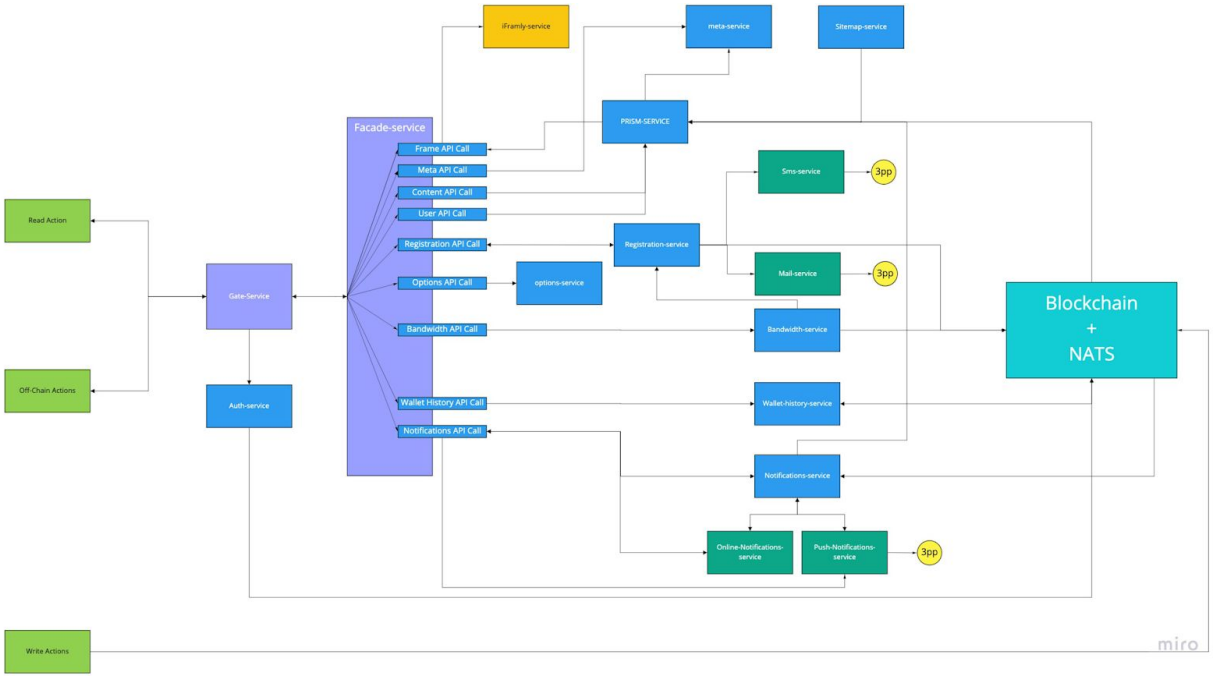
3.2 Additional Services:

1. Notify-service is a service for alerts. The service extracts the necessary data from the blockchain, determines the events that have occurred and groups them putting into database. The story is available on request, indicating the type of event of interest. If the user is online, he receives real-time notifications via related online-notify-service.The users of mobile clients, browsers get push notifications. For this Push-service is used. The data is stored for a limited time and then deleted from the database.
2. Online-notify-service is a microservice for sending online notifications to users, that also stores notification settings. It is assumed that there is an event registration microservice that notifies this microservice about new events leaving the right of filtering and direct distribution to this microservice.
3. Push-service is a microservice of sending push notifications to mobile devices and user browsers. It also stores user settings including specific types of notifications and their languages.

4. Options-service is a service for storing settings and metadata for the application. Each user is assigned an entry in the database that stores the settings in the options field, where the key is usually a microservice alias and the value is arbitrary data, the format of which is determined by microservice itself. Also it supports requests from the front-end gate.
5. Sitemap-service is a service generating a sitemap.
6. Mail-service is a microservice mailing users.
7. Meta-service is a service that stores various metadata, in particular the number of post's views.

3.3 Generic Infrastructure Overview

Below there is a diagram of Commun structure



Further Improvements

Privacy and Storage of Content

Commun is supposed to be implemented in several steps. The MVP will include everything needed for public content management. After launching Commun will provide a user with an opportunity to create communities, post content, comment on the posts, to like and dislike posts

and comments. **At a later stage,** private data management will be added; namely, it would be possible to send and receive private messages, to post and comment privately, to send encrypted/mixed/blinded transactions, etc.

Coming to private content, a must have list seems to contain the following:

- Private messaging. Since data types transferred with modern messengers vary from small text messages to large data blobs, the requirements for private messaging protocol are highly demanding. Small text messages should be transferred instantly, media files should be stored in a separate storage, but still protected by authenticated transaction log (forget the “hash-only storage” - it’s non-viable). This makes it impossible to process all the required operations by a single protocol. There is a need for a separate messaging protocol with seamless integration with the main Commun protocol database, that should provide some separate decentralized cluster infrastructure for messaging data.

- Private content management. This has to be introduced for private communities or event groups. Since the data are supposed to be available for a limited set of users, they should be encrypted. The keys for encrypted data are required to be spread across members of the group and renewed every time someone leaves or enters the community (a type of Public Key Infrastructure-like construction). It is highly important to get rid of the requirements for complete data download, decryption, encryption with a new key.

We also will look at new storage options, namely Urbit and IPFS, to ensure better reliability of content retrieval for our users.

Conclusion

Commun makes to unleash the power of blockchain technology for its Users an attempt to solve the problem of tragedy of commons in decentralised social networks content monetization problem in centralised social networks with the introduction of the platform that enables easy setup and development of communities, and provides a mechanism for their monetization. It makes users to follow the rules proposed with financial stimulation. Moreover, in the future privacy messaging, private content and private group solution will be implemented solving problems of current social networks with privacy and data ownership.