Ellcrys Technical White Paper

Kennedy Idialu

April 15, 2019

Abstract

A decentralised source code repository can enable collaborators to build open, transparent, community-led software products and networked organisations without a central authority acting as an enabler or coordinator. Git, a decentralised version control system provides the ability for collaborators to share source codes and assets of software projects in a decentralised way, but the inconvenience and requirement needed to host and maintain an active server makes self-hosted repositories unattractive to most people. We propose a system that store repositories on a blockchain network, one that is highly accessible, censorship-resistant, requires little coordination between collaborators and empowers communities to create, share ownership, contribute and govern open shared software-driven enterprises.

1 Introduction

Open-source development began as a revolution against the high cost and often sub-par products that closed-source software development companies created.These companies were known to monopolise certain areas in IT. They were shipping software products that were buggy, less interoperable, expensive and had slow-release cycles.. Developers across the world began to collaborate, working asynchronously at times suitable for them to create alternatives that were more user-friendly, highly performant, free to use and has the backing of a huge community This was a game changer.

1.1 Open Source, Today

Today, open source software products run the world Everyone relies on open source software products directly or indirectly, and the movement has forced the once closed-source companies who were anti-open-source to begin to release open-source projects as well as contribute to existing open projects. However, companies are increasingly becoming irresponsible with the way they manage users data and experiences. We believe its time for open source to move towards a new phase where its communities provide competing and innovative services that are open, transparent and driven by a user-first philosophy. Right now, open source communities mostly rally to build static products that form part of a more significant product or service offered by closed-source, equity-driven companies who rarely contribute back to the community or have questionable ethics. These open communities have little power to effect changes on the misbehaving companies that use their product. Unethical practises by companies, regulators and government motivated the creation and sustainability of Bitcoin[1] A peer-to-peer electronic cash system that allows people to send, receive and control their money without any intermediary. Bitcoin is an example of a service designed to be open, transparent, pro-users and governed by a community of people spread around the planet.

1.2 Open Source, Tomorrow

The shift towards community-driven services spearheaded by open source communities is already happening in the cryptocurrency industry which is worth over $176 billion as at the time of writing this whitepaper, and it is the most prominent pioneer in the creation, incentivization and governance of systems that are created in the open with publicly disclosed mission and ethical standards. Projects like Bitcoin and Ethereum have a large community of participants who regularly hold administrators and core developers accountable to the standards they have publicly championed. If we intend to derive benefits from open source services quickly, then the tools and ideologies that make these communities successful must be adapted for communities building centralised applications which are undoubtedly the most accessible, convenient and faster than their decentralised versions. By adopting the open and collaborative structure of open source to centralised systems, making their development, governance and executions more transparent and accountable, we make it easy for communities composed of mostly mutually distrustful collaborators to build all kinds of applications (desktop, mobile, VR) together and integrate any business model like centrally-run organisations. However, for us to realise a future where open source services compete with centrally-led services, we need to find and fix the challenges and obstacles that may produce failures.

2 Obstacles Of Open Source Services

A host of infrastructural inadequacies challenge the dream of community-led open source software services directly solving end-user problems with the same infrastructure and support that centrally-governed organisations enjoy. In this section, we describe some of the significant issues we believe to be most worrying.

2.1 Collaboration Tools & Platforms

Open source developers are distributed across the world. They speak different languages, have different beliefs and live in different time-zones, but these differences have not slowed down the progress of the OSS community. Most developers contributing to open source projects today use a distributed version control system (DVCS). These tools provide a common development protocol that collaborators use to contribute and manage a codebase. Most developers make use of the Git[2] to share, track, package and version their contributions. Git is a distributed version control system developed in April 2005 by Linus Torvalds to aid the development of the Linux kernel. It has since grown to be one of the most important tools utilised by developers and enterprises across the world. Git allow developers to work in a decentralised approach where they manage replicated source code repositories, contribute and resync with other developers. Unfortunately, running and maintaining independent git servers is challenging for most users Requires understanding of some networking, good internet, high-availability, server cost if hosted on the cloud. For this reason, cloud-based code hosting services like Github[3] emerged to provide users with a service that host repositories, thereby eliminating the need for users to run and maintain personal servers. These services also provide additional collaboration utilities like issue tracking, reviews, pull request that enable users to communicate problems and manage new releases of their software. Millions of developers use Github and other code sharing platforms to build great software, but the central ownership, governance tools optimised for authoritarianism and execution of the service make it unfit to host community-owned open source services.

2.2 Shortcomings of Centralized Code Sharing Platforms

In this section, we highlight some of the reasons why centralised code hosting platforms are unsuitable for building true community-led, shared enterprises.

2.2.1 Censorship

Code is free speech; similar to human languages, it is a form of expression used to communicate ideas; This has been proven in a U.S court [4] in the case between Bernstein v. the United States where Bernstein wanted to publish a paper and source code of his encryption system [5]. However, the government required him to submit his ideas for review and register to acquire a license as an arms dealer, and failure to do so would result in criminal penalties. Bernstein sued the government and won when it was ruled that the First Amendment protected software source code. However, not many countries have laws that recognise code as free speech. As a result, code sharing platforms are exposed to request and demands from governments to censor projects created by individuals of interests. It is not news that Github continues to face threats and attacks from governments[6] who want repositories removed. While code sharing platforms face a real threat from external and hostile actors, these services themselves can also exert censorship actions upon users and their projects according to predetermined or arbitrary terms which are mostly driven by their business models. Additionally, not only are users exposed to censorship behaviours incited by external agents and the code sharing platform, but they are also vulnerable to censorship by project admins or maintainers. Project maintainers are free to cut access to a repository they manage at any time and on their terms. The various levels of censorship threats make centralised code sharing platform unsuitable for hosting communities building innovative, competitive and possibly contentious applications and services. As long as contributions are hosted centrally, repositories continue to remain under potential threat.

2.2.2 Ownership

Most tools and services on the internet support a single owner account architecture. An architecture that recognises and grants ownership and authority to one person identified by their email or phone number. Some of the motivation for this is the need to maintain security and to determine who pays the bills. Code sharing platforms are among these service providers built on this single-owner structure. All repositories must be owned and managed by one person who is granted full privileges to create, access, add members and destroy any resource associated with their account without notice. The consequence of this is that complicated, sovereign and community-led organisations cannot build software in a trust-less manner since these collaborators must trust the root account owner. One approach employed to remedy this problem involves the creation of committees and other secondary structures. However, we argue that these structures do not deliver real ownership and governance since there is a root password managed by an individual or organisation. In the crypto industry, they say If you do not own your private keys, you do not own the coins. Likewise for open source projects on a code sharing platform, If you do not own the password, you do not own the project. It is essential that contributors to community-led platforms can honestly and provably share ownership of a project without needing to trust a central authority such as the owner or maintainer of a repository or the platform operator. Real ownership of a community-owned software cannot be granted based on faith that the authority will always act in the best interest of collaborators.

2.2.3 Immutability

The concept of immutability in computing refers to the unchanging, unalterable state of data. It refers to the inability of a piece of data to be altered. Immutable data is easy to reason about; We can make assumptions and build on top of them with certainty that their state will not change unexpectedly. Interestingly, the Git version control system used by millions of collaborators and supported by most code sharing platforms utilises an immutable data structure. Git includes concepts like branches and commits; Commits are backwards-linked collections of changes that exist inside a branch. Collaborators can begin working from any commit and never have to worry that future commits may alter the state of the commits they are extending. However, code sharing platforms do not guarantee immutability of an entire repository. Repositories as a whole are not immutable; Their owners or the platform operators can delete them. It is important to note that the ability to delete repository can serve as a tool for censorship. An account owner or platform operator can prevent people from collaborating by removing a repository from existence. Although repositories can be cloned easily, it is more challenging to re-organise a community disrupted by an act of censorship. The lack of guaranteed immutability on code sharing websites makes them unappealing for hosting community-led services. It will be devastating for a community to one day find their shared enterprise deleted by the account owner or platform operator.

2.2.4 Governance

For open source communities to succeed in creating and managing decentralised organisations, they must first figure out governance, otherwise what would be obtainable is chaos, uncertainty, indecision, and unaccountability. These communities need to be able to formulate a governance system that is acceptable to all parties and enables them to make decisions quickly and fairly. On centralised code sharing platforms, there is only one kind of governance system One where the account owner dictates how and when collaborators interact with and contribute to the project. As a remedy, big projects delegate control to reputable, structured open-source organisations like Apache Software Foundation or the Linux Foundation. The ideal collaboration framework for decentralised organisations must take a vendor-neutral stance, only providing primitives and templates that can be used or adapted to create simple to complex governance models with enforcement carried out by the protocol where possible.

2.2.5 Economic Incentives

People do not contribute to open source software irrationally. They do it because there is something to gain. Many people contribute because they want to learn or be members of a community of like-minded individuals. There are those who do it to improve their reputation so that prospective employers may be interested in their abilities. Companies whose business model depends on open source software are also incentivised to contribute to it.

A popular misconception about open source is the idea that open source contributions should attract no financial incentive and remain a free and voluntary activity. However, the truth is, most open source contributions impose time and financial commitments on contributors who need to develop, test, document, evangelise the product and fix bugs to ensure the product is suitable for both individual developers and enterprises. It is unrealistic to expect contributors to consistently offer their time and money to a project that gets used by others for commercial purposes and to expect no form of financial compensation as incentives to continue to remain committed to the long term improvement and sustenance of the project. If the quality of open source software is to be maintained or improved, the best way to guarantee it is by bringing financial incentives to the table.

Two of the main channels from which open source collaboration may receive financial support are code-sharing platforms or open source foundations. Code sharing platforms can create mechanisms that could potentially allow open source developers to receive financial assistance in the form of donations or subscription to premium branches, but they do not do this. These platforms do a lot to enable collaboration but not enough to allow contributors to receive financial benefits.

Open source foundations such as Apache Software Foundation can receive donations and generate income through sponsorship, merchandise sales, support and hosting conferences but those funds are for the day-to-day running of the foundation of which none of it goes down to non-employee contributors. There needs to be a trusted, transparent and automated mechanism for creating, verifying and transferring value between users and collaborators of open source software. The introduction of such a mechanism will create many types of open source economies (such as bounty, bug hunting, support services).

In the era of community-led services, collaborators need to be able to generate or receive financial rewards to support continuous development and execution of their services. While centralised code sharing platforms are unsuitable for providing the needed infrastructure, blockchain technology can allow collaborators create shared software products, formulate protocol enforceable governance structure, vote and authorise actions, receive and distribute financial incentives without intermediaries.

2.2.6 Execution Environment

To go beyond projects like libraries, frameworks, CLI tools to community-led services, we need to begin to consider program execution environments that are suitable for running shared applications in the form of dApp, SaaS, API and other types of software. This execution model must lend itself to be deployed, operated and governed according to the governance structure instituted by the community.

Collaborators will need an environment to run tests, stage and deploy their applications. Most code sharing platforms only provide continuous integration environments with the configuration of this environment still subjected to the authorisation of the single-user account owner. There is a need for a mechanism that allows a community to collectively decide what kind of execution environment and configuration they want for their projects.

A deployment environment includes a tool that runs an application and makes it accessible to target users. The ideal deployment environment must be unalterable, autonomous and can only allow itself to be upgraded according to the governance rules of the community.

Historically, consumer applications have been executed in a centralised, managed environment but with the emergence of blockchain platforms like Ethereum, applications can now be executed decentrally on thousands of nodes in a trust-less and autonomous manner. Decentralised applications are immutable; they cannot be altered or destroyed once deployed.

While many believe blockchain applications should replace all centralised applications, we believe community-led services can leverage both types of execution environments depending on their goals; This way they are better positioned to compete with centralised services providers on all fronts.

Collaborators may decide to build completely decentralized application that cannot be altered once deployed if the nature of their value proposition demands it. Otherwise, they may simply create centrally executed applications like SaaS, API, mobile application and more. These communities building centralised applications can setup governance system that is strong enough to provide some of the qualities of decentralized applications (immutability, openness, transparent and autonomy) through enforcement of an open and transparent constitution.

The ability to build applications that can leverage the best of decentralisation and centralisation will help collaborators build useful applications that can easily gain adoption and deliver great value and experience to users.

3 The Ideal Platform for Open Source Service

In section 2, we discussed the reasons why centralised code sharing platforms are unsuitable to lead the change towards a future of community-led software products and businesses. These changes range from censorship, ownership, and governance to execution models. We can see that the major obstacle is the inability to enforce ownership, transparency and accountability.

We take the following approaches to solve the issues:

- Ellcrys will use blockchain technology to eliminate censorship concerns by creating a new type of decentralised collaboration framework that is open to anyone, anywhere and cannot be destroyed. There will be no entity to receive and carry out censorship requests.
- The use of public key cryptography as the primary mechanism to create identity and to authorise actions, will allow projects

to have multiple owners or signatories who can partake in the day-to-day governance of the project.

- By the very nature of blockchains, repositories will remain immutable: They will always be accessible to everyone. Although, owners of the repositories may enable the ability to apply access level rules to prevent unauthorised write operations.
- The introduction of autonomous procedures (a.k.a smart contracts) will allow communities to create immutable applications to enforce ownership, governance, incentive structure and disbursement and business rules. Many nodes on a blockchain network execute autonomous procedures in a way that ensures correctness and resilience to attacks that would normally cripple a centralised equivalent.
- The use of Git makes it easy for millions of developers who are already familiar with the tool to easily create repositories, contribute, import their existing projects and integrate with other git compatible tools and services they already use.
- The combination of public key cryptography and the ability to create complex governance structures, external service providers can deliver value based on the governance history or events (e.g. proposals) of a project. For instance, a community may create and vote for a proposal to run ads on Google Adwords, upon the approval, a designated service creates a campaign on Google Adwords using the information provided in the proposal.

4 Bitcoin

In this section, we briefly discuss bitcoin, its consensus mechanism and particularly its inability to process enough transactions per second to meet mainstream users needs.

4.1 Introduction

Bitcoin is a form of electronic cash. It is a decentralised digital currency that does not have a central bank or a trusted authority. It can be sent from person-to-person faster and cheaper without restrictions. With cryptography, transactions are verified and transmitted through a peer-to-peer network of computers that update account balances and record changes in a public distributed ledger known as a blockchain[7]. Bitcoin was invented by Satoshi Nakamoto[8] and released to the public as an open source software in 2009.

4.2 Consensus Mechanism

Blockchain networks require a means to reach an agreement on a single value. Consensus algorithms define the rules that must be followed by computers on the network in other to reach an agreement. The process of agreeing with other computers is known as the consensus problem, and it is a well-researched field in computer science. Bitcoin uses a consensus algorithm known as Proof of Work9. It is used to process blocks of transactions and append them to a chain of blocks the blockchain. In Bitcoin, the process of appending a block to the blockchain is known as mining, and the individuals who take part in the mining process are known as miners. PoW requires miners to solve complex cryptographic puzzles to gain the right to add a block.

Miners receive newly created Bitcoin as a reward for solving the puzzle. This reward is known as the block reward. Before the reward is issued, the generated block must follow the consensus rules. All other nodes on the network are responsible for verifying the block and enforcing the consensus rules. Once a miner discovers a valid block they did not create, they immediately stop and start mining the next block. There is a network difficulty that ensures that blocks are created within 10 minutes. It determines how hard it is for the miners to mine a block and is adjusted dynamically to maintain the 10 minutes window.

Bitcoins PoW has a disadvantage of being very slow. It has been described as a consensus system that wastes energy and is causing harm to the planet. However, it is the most battle- tested and well understood trustless consensus system in the cryptocurrency industry.

4.3 Low Transaction Throughput

Bitcoin pioneered the blockchain industry and made it attractive for different categories of people who believed in its vision. Developers have created clones and alternative implementations intending to improve on the capabilities of Bitcoin. One of the primary motivation fueling the creation of Bitcoin alternatives is the inability of the Bitcoin network to process large amounts of transactions sufficient to handle the needs of mainstream users. It takes 10 minutes for the bitcoin network to create a new block and even more to attain minimum confirmation of a transactions finality. Most people agree that for cryptocurrency to go mainstream and be capable of competing centralised incumbent, blockchains must be able to handle thousands of transaction per second.

4.4 Alternative Consensus Mechanisms

Since the introduction of Bitcoin, developers and researches have been working on alternative consensus algorithms that are more efficient and performant than proof-of-work. The most popular of these alternatives is Proof of Stake (PoS). In Proof-of-Stake consensus, the blocks are not created by miners solving computationally hard math puzzles, but by participants who stake their money to bet on the validity of a block. These participants take on the role of block forgers and validators. Proof of Stake is a faster and more energy efficient alternative to Proof of Work, but it suffers from a severe problem known as Nothing at Stake. The concern behind Nothing at Stake is that since it cost next to nothing to create blocks that support a fork, validators can vote for every fork that happens without any severe consequence; This is unlike proof-of-work where a computationally hard problem must be solved to extend the main branch or a forked branch. Modern PoS systems employ a strategy that punishes misbehaving participants who failed to follow the consensus rules by slashing their stake.

4.4.1 Bitcoin-NG

The performance of blockchain protocols is limited by two factors block size and block interval. Increasing the block size correlates to an increase in transaction throughput. However, bigger blocks take more time to propagate to nodes on the network and can lead to an increase in forks if these blocks arrive late. Decreasing the block size reduces the time it takes for blocks to propagate to nodes, but doing so rapidly leads to forks and frequent reorganisation. The Bitcoin system traded transaction throughput for latency by using a small block of 1MB with a 10 minutes block time: Blocks are sufficiently propagated throughout the network due to the block time, but minimal transactions are processed.

Bitcoin-NG[10] is a scalable blockchain protocol designed to improve the throughput of Bitcoin without affecting the same trust model that exists in Bitcoin. It achieves performance improvement by introducing the concept of a leader election and transaction serialisation. In BitcoinNG, time is divided into periods known as epoch where every epoch has a leader. Once a node becomes a leader, it is expected to produce blocks until a new leader is discovered. Leader election is performed randomly in the same way miners in Bitcoin find blocks infrequently. BitcoinNG ensures that the system can process transactions even when the network works to find a new leader unlike Nakamoto Consensus in Bitcoin where no transaction is processed during the 10 minutes block interval.

5 Ellcrys Protocol

In this section, we describe the engineering direction of our protocol. Existing knowledge of blockchain technology is essential. Please note that some concepts described below are not final and may be subject to change as we continue to execute on the journey to deliver our vision.

5.1 Consensus Model

5.1.1 Hybrid Protocol

The Ellcrys consensus mechanism is a hybrid consensus algorithm of both Proof-of-Work and Proof-of-Stake. Our consensus model is based on concepts described in the Proof of Activity[11] protocol by Iddo Bentov et al. They described a process of extending Nakamoto Proof of Work via Proof of Stake to increase the security of the network while reducing the dependence on miners as the only contributor to network security. Iddo Bentov et al. questioned the capability of Bitcoin miners to sustain the security of the system when PoW is no longer subsidised via the block reward. We agree with their assertion that a cryptocurrency protocol should be designed in such a way that its incentive structure encourages different participants in the system to sustain its health.

5.1.2 Bitcoin-NG Leader-based Block Creation

Bitcoin-NG extends Nakamoto consensus into a protocol that is capable of processing more transactions than Bitcoin. It does this by introducing a leader-based block creation scheme where a miner is entitled to create several more blocks at faster intervals after they mined a block known as a key block. Ellcrys PoW implementation is based on Bitcoin-NG to improve transaction throughput through improved latency and bandwidth.

5.2 Coin

In most public blockchain systems with a native cryptocurrency, new coins are generated when a new block is mined. The generated coins are shared to the miner and any other network participants who contributed to the creation or validation of the new block at a predetermined ratio. In Bitcoin and other similar proof-of-work blockchains, the miner of the block receives all newly generated coins.

We believe a system built on a coin generation model where a single participant receives all rewards is unfair and does not promote equitable distribution of wealth generated by the network. Centralization of rewards within a wealthy minority further replicates and increases the existing gap between the rich and poor. In matured proof-of-work blockchains like Bitcoin, a user needs specialised equipments to compete with well-funded corporate organisations. Although, one might argue that these networks provide security to the network, but they may also become future adversaries when incentives falling below their cost of operation due to low block rewards.

5.2.1 PeopleMint

Introduction

We introduced PeopleMint[12] A novel coin generation model that allows anyone to participate in the creation of a new coin by exchanging their national banknotes for new coins. PeopleMint gives everyone a chance to participate in coin creation using a material (a currency note) that is both valuable and accessible to all classes of people. Banknotes or currency notes are the raw material a miner needs to be able to create new coins and receive rewards for their service to the network. Without it, miners are compensated only through transaction fees.

Against Miner Centralization & Unfair Distribution

PeopleMint creates a symbiotic relationship between a global population of users and miners working together to sustain the system while also discouraging miner centralisation by diversifying the participants and processes involved in coin creation. In Bitcoin, PoW mining is the only mechanism through which new coins are introduced. Miners do not need any other party and are incentivised to invest more in superior hardware and infrastructure to outcompete other miners which further makes mining increasingly inaccessible to smaller miners. PeopleMint provides a fair coin generation model as well as a chain security mechanism to keep miners in check through a human-driven process of observing and implicitly voting of blocks on the main chain.

How It Works

- **Create Banknote Transaction**: Alice takes a short video in which she must capture an image of a banknote with the last 12 bytes of the hash of a recent block written on the said banknote. She encodes the video in a transaction and sends it to a Node A. Alices transaction must include a field called *banknoteHeader* which includes the serial, denomination, and the country code of the banknote.

- **Preliminary Validation**: Node A performs preliminary validation to weed out banknotes that have a serial that was previously processed, or notes that have unsupported denomination of currency code. After successful validation, the transaction gets added into the transaction pool. These notes in the pool are not eligible for inclusion in a block until sufficient endorsements from off-chain validators have been collected.
- **Off-chain Validation**: Off-chain validation begins when a banknote passes the preliminary check and gets added to the transaction pool. Validators attempt to analyse the banknote to check the correctness of the information provided by the banknote scanner. The validator constructs a new encrypted endorsement transaction named *bnEndorsement* that contain answers to the following protocol questions:
  - Does the video include a banknote that looks authentic? [Yes/No required].
  - Provide the block hash written on the banknote in the video? [12-bytes alphanumeric value required]
  - Does the serial specified in the transactions banknote header match the serial on the banknote in the video? [Yes/No required].
  - Are the currency code and denomination valid? [Yes/No required]

  A validator must construct a banknote endorsement transaction with answers to the above queries and send it to nodes on the network. All nodes collect these endorsements until enough is received from other validators.
- **Banknote Endorsement Revelation**: A banknote endorsement is encrypted to hide the answers of a validator to prevent other validators from spying and copying. It can only be revealed when a subsequent *bnEndorsementReveal* transaction containing the decryption key is received in the transaction pool. A miner cannot include a banknote in a block without collecting a sufficient number of *bnEndorsementReveal* transactions.
- **Banknote Endorsement**: As soon as a node receives /N/ *bnEndorsement* and /N/ *bnEndorsementReveal*, it checks whether

the majority accepts the note. A *bnEndorsed* transaction is created only when an endorsement has been accepted: It must include *bnEndorsement* transaction and the *bnEndorsementReveal* transactions collected. The *bnEndorsed* transaction is added back to the transaction pool and broadcasts to other peers. *bnEndorsed* transactions are eligible for block inclusion. During block validation, nodes verify that the *bnEndorsed* transaction includes a banknote that has been endorsed and accepted. A *bnEndorsed* transaction must include all the information required to validate it.

- **Network Exchange Rate**: When a banknote gets scanned, validated and added to a block, it does not automatically get coins proportional to the unit of the banknote. For example, a $100 note does not result in the creation of 100 ELL. Instead, there is first a devaluation function applied that reduces the USD value of the banknote by up to 80%. After the devaluation, the final USD value is used to purchase ELL from the network at a fixed exchange rate encoded into every node. The devaluation function serves to limit the number of coins created such that many scanners can participate without causing an unchecked increase in coin supply.
- **Network Reward Limitation**: The network reward limit determines the maximum number of coins that can be accumulated by a miner. A miner is allowed to collect as many endorsed banknote transactions necessary to reach the reward limit.

5.2.2 Coin Basics & Supply

The Ellcrys native coin is called /Ell/ (plural: /Ellies/) . It is divisible to 18 decimal places. Its official Unicode character is (U+0205). At the launch of the main network, the initial supply cannot exceed a total of 1,500,000,000 coins.

5.2.3 Initial Supply Allocation

The initial supply is allocated in the following ratio:

- Founders 14%
- Ellcrys PBC 25%
- Advisors 1%
- Public Distribution 60%

5.2.4 Public Distribution Phases

We will distribute the supply allocated to the public through a combination of public sale sessions and a distribution network designed to allow millions the chance to claim a share of the initial supply.

Public Sale Sessions

Session 1 (Private Sale)

- Started: 24th of December 2017.
- Ended: 24th of January 2018.
- Duration: 1 Month
- Distributed Supply: 10,346,202 ELL (includes bonuses)
- Price: 0.05 USD

Session 2 (Pre-Sale)

- Started: 1st of February 2018.
- Ended: 1st of March 2018.
- Duration: 1 Month
- Distributed Supply: 7,378,820 ELL (incl. bonuses and bounties)
- Price: 0.08 USD

Session 3 (Pre-Dist.Net Sale)

- Started: TBA
- Ended: TBA.
- Duration: 1 Month

- Distributed Supply: 20,000,000 ELL
- Price: TBA

Final Session (Distribution Network)

- Started: TBA
- Ended: TBA.
- Duration: 1 Month
- Distributed Supply: 862,274,978 ELL
- Price: Determined by PeopleMint exchange algorithm

5.2.5 Distribution Network

Introduction

As a hybrid consensus system composed of both proof-of-work and proof-of-stake, there is a need to ensure the fair and widespread distribution of the initial coin supply to as many people as possible. When a few individuals own enough coins to launch an attack, the security of the system is weakened.

The vast majority of ICOs have distributed their initial or total coin supply within a month, leading to a situation where large amounts of coins end up in the control of a few individuals or entities. More so, short-lived ICOs prevent a sufficient amount of people from getting to learn about the project and understanding enough to acquire a stake in the system. Ideally, for a blockchain network to be sufficiently decentralised, its coin supply must also be well distributed.

The Network

To ensure broad and fair distribution of the native coin, we are going to run a temporary network known as the Distribution Network. The network is going to serve a singular purpose of allowing anyone in the world to exchange banknotes for the native coin through the PeopleMint mechanism. Once the public allocation is distributed, a snapshot of the blockchain state is collected and used to seed the genesis block of the main network.

The network is to run the full consensus specification which allows anyone to participate (as a miner, endorser, validator and scanner) and earn regular network rewards. We can test the protocol, plan upgrades and onboard more network participants into our community in preparation for our main network launch.

Multipliers

Before the distribution network commences, the Ellcrys team would not have sufficient funding to continue to support the project and build a thriving ecosystem. To generate funds, we are going to sell unique signature tokens known as multipliers that are capable of increasing the allocation generated from scanning a banknote. For instance, if a $100 note produces 2.4 ELL, a 5x multiplier increases the output to 12 ELL. Multipliers are not required to participate in the network Anyone can still create/earn coins by freely scanning currency notes or participating in other departments of the network.

5.3 Network Participants

We now describe the types of network participants working together to sustain the system.

5.3.1 Miners

Miners are individuals who participate in Proof-of-Work attempting to create blocks by solving hard cryptographic puzzles which require the consumption of energy. In Bitcoin, a miner is responsible for finding a solution to the puzzle for the next block. They use the result of the puzzle to create a new block containing transactions and broadcast it to the network. Likewise in Ellcrys, miners are responsible for creating /key/ and /microblocks/. Key blocks are similar to Bitcoins block in that they are created by expending energy, while microblocks are created at a predetermined interval and do not require energy investment.

5.3.2 Witness

Witnesses are stakeholders who validate key blocks, microblocks and extend microblocks with additional transactions. Witnesses serve as validators that audit the work of miners. They enable power dynamics with PoW miners where they continuously ensure these miners abide by the rules.

When a block is received, every witness online checks whether the blocks header is valid; It checks if the header contains a previous blocks hash, if the difficulty matches the expected current difficulty and whether the block remuneration is valid. Once validated, the witness checks whether it was one of the /N/ witnesses selected to verify this block. It does this by running a pseudo-random operation to determine the N witnesses, and when it finds that it was selected, the witness signs the hash of the block and broadcast the signature to the network.

When the /Nth/ witness see that it was derived by the block, it creates a wrapped block that extends it. If the block is a microblock, it includes as many transactions as the remaining block capacity can support, the signatures of the other selected witnesses and its signature of this wrapped block. The Nth witness broadcasts the wrapped block to the network, and when other nodes see that it is valid, they append it to their main chain.

5.3.3 Transaction Endorser

One of our core mission is to make it convenient for collaborators to create blockchain applications written in standard, general-purpose programming languages. Current blockchains that provide an application execution environment require these programs to be written in domain-specific languages. The requirement to use domain-specific language limits widespread adoption and may lead to buggy or insecure applications.

A major reason for DSLs (Domain Specific Languages) is that blockchain applications written in general-purpose languages can produce non-deterministic code that can lead to the creation of forks; This can be exploited by bad actors to prevent the network from reaching consensus on the validity of a block, thereby halting the system. Many programming languages include standard features that produce non-deterministic behaviour (e.g. Go map keys are randomised).

Hyperledger Fabric[13], which is the first blockchain system to support the use of multiple, well-established languages solved this problem of non-determinism by separating transaction execution, ordering and validation using a novel approach known as /execute-order-validate/

architecture. In this model, a client sends a transaction to peers specified in an endorsement policy. Each transaction is then individually executed by the peers and its output recorded. This process is called /endorsement/. The client must collect /N/ endorsements up to the amount specified in the endorsement policy. After collecting enough endorsement, it creates a transaction and broadcasts it to the ordering service for inclusion into a block, after which the ordering service broadcast the block to the rest of the network to perform the validation phase. During validation, transactions that failed to pass the endorsement policy or have non-deterministic outputs are ignored.

Our execution model is based on similar pre-consensus transaction execution and endorsement strategy. On Ellcrys, endorsers are stakeholders who are responsible for executing and endorsing transactions based on a global endorsement policy. Endorsers are former witnesses of the previous epoch. When a client broadcasts a transaction, a node X checks whether it was a witness in the last epoch. If it was, it is required to endorse the transaction and broadcast a signed endorsement report to its peers. Node X collects endorsements, creates a wrapped transaction which includes the endorsements, adds it to its pool and broadcast it. During block validation, nodes will use the endorsement to ensure only deterministic transactions are processed.

5.3.4 Banknote Scanners (Scanners)

Banknote scanners are individuals who provide short videos of banknotes required to create new coins. As described in Section 5.2.1, PeopleMint allows anyone to exchange banknotes for the native currency. Scanners annotate the hash of a recent block, take a short video that captures the banknote and sends this video to the network for analysis and validation.

Scanners are required to annotate their banknotes to fulfill a chain observer role. The more observers a chain has the greater its chances of becoming or retaining the best chain position. To increase the security of the network, we take into account the number and value of banknotes a chain has received as a criterion in a fork choice situation.

5.3.5 Banknote Validators (Bettors)

Banknote validators predict the validity of a banknote that was created by a scanner. Their opinion, along with those of other banknote validators is used to determine the correctness and validity of a banknote, as well as any other metadata associated with the node.

Validators perform their duties by staking a predetermined amount of the native coin along with their response to the markets questions such as (1) whether or not the banknote looks authentic, (2) what the denomination of the note is, (3) the currency code of note and lastly (4) what the hash that was annotated on the note is. The information provided by a validator for a given banknote is compared with the prediction of other validators who observed and predicted on the same banknote. A prediction is considered correct if it matches the outcome predicted by the majority of banknote validators.

5.4 Block Creation

The protocol divides time into *epochs*. In each epoch, a single leader is chosen to extend the blockchain by creating blocks known as *key* and *microblock* blocks. Key blocks indicate the beginning of an epoch after which the key block creator is entitled to create microblocks at faster intervals.

5.4.1 Key Blocks

Miners are free to start an epoch by creating a *key* block via the same Proof of Work computation performed in Bitcoin. A *key* block header includes the unique reference of its predecessor, the current time, the public key of its creator to payout the reward, difficulty information and coin generation transactions. Unlike Bitcoin, the public key of the key block creator is added to the header and used to verify subsequent microblocks. Similar to Bitcoin, Key blocks creation time are determined by the network difficulty. For example, Bitcoins difficulty dynamically adjusts to target a 10 minutes block creation interval.

5.4.2 Microblocks

After a node has generated a key block to become the leader, it is allowed to generate microblocks which can include transactions. Microblocks must be generated at a deterministic rate. A microblock is considered invalid; if the difference between the timestamp of a microblock and its parent is less than a predetermined minimum or its timestamp is a future time. The constraints are put in place to prevent a malicious node from overwhelming the system with microblocks.

A microblock contains transactions and a header. The header contains the hash of the previous block, the current timestamp, the Merkle root of the transactions and the signature of the header. The signature must be created using the private key of the most recent key block in the chain.

5.4.3 Block Validation

After the creation of key or microblocks, the miner broadcasts the block to the network. Upon learning about the block, witness nodes must check whether they have been selected to validate the block by running a pseudo-random, deterministic algorithm to compute a set of witnesses for the block. Every online witness checks whether the block is valid, sign the hash of the block header with the private key that controls their staking account and broadcast their signature to the network.

The *Nth* peer in the set of eligible peers must create a wrapped block: It may include some transactions if the received block is a microblock and broadcast to the network. Upon receipt of a wrapped block, the node validates its header, any transactions it may have and appends it to its main chain. The wrapped block must contain signatures of *2/3* of the *N* peers in the computed validation set. The fee from the transactions in the wrapped block is shared between the miner and the witnesses.

5.5 Git Hosting System

In this section, we discuss issues and approaches relating to how we plan to implement a resilient, accessible and easy to use git hosting service that does not degrade the ease of use and convenience provided by centralized code hosting platforms.

5.5.1 Design Goals

- **No Server Requirement**: One of the challenges with the git technology is the hosting and maintenance of a server to enable easy transport of contributions between collaborators who may not be within the same network, geography, and timezone. To do this effectively, users need to have some technical experience managing and configuring servers and networks. Additionally, users who are behind firewalls or do not have a public internet address may be unable to host, serve or access repositories. Contributors can choose to collaborate on a single git server administered by an authority which will leave them vulnerable to censorship actions and single point of failure attacks against the authority. Flexibility, availability, and convenience are also affected as collaborators will need to be online and on their desk to sync up with others. An ideal system for enabling collaboration between globally distributed collaborators and organizations should not require collaborators to host their own servers or depend on trusted entities. It should not degrade the convenience offered by centralized code sharing platforms.

- **Cheap Service**: Centralized code sharing and hosting platforms today allow users to create repositories and contribute to them for free. The ability to offer free services to users is an advantage centralized hosting platforms have over a decentralized counterpart. They are well-funded, profit-driven entities that operate a business-model optimized to encourage free usage until a user reaches a limitation that forces them to begin to pay monthly rent. A decentralized alternative does not have the luxury of investors fund paid to all network infrastructure providers as subsidy. Every network participant needs to individually cover their cost of operation from network rewards (e.g. inflation and transaction fees). If usage is made free, the network will be exposed to spamming attacks that can quickly cripple the network, low security and infrastructure failures due to inadequate incentivization. In the end, it is imperative that the network finds a balance between adequately incentivizing network participants and making sure that collaborators access the functionalities of the network cheaply.
- **Decentralised, But Voluntary Storage**: A core quality of the blockchain technology is the ability to replicate a piece of data to all nodes on the network This behaviour allows the network to remain resilient to failures even if a large number of nodes suddenly went offline. However, this resilience comes at a cost of ever-growing storage requirement. The speed at which the network consumes disk space depends on the utility of the network. Bitcoin, a 10-year old financial settlement network and the oldest cryptocurrency has so far used less disk space than the much younger Ethereum which is a general purpose system for devel-

oping blockchain applications. For a decentralized git hosting system with smart contract capabilities, the storage complexity becomes compounded. Under the original Nakamoto Consensus, all network participants will be required to run a full node. We believe this is not a feasible approach as growing storage requirement will eventually drive away small network participants. Our goal is to implement a system that is decentralized and does not mandate all nodes to be full nodes, allowing anyone to participate in sustaining the network without worrying about storage.

- **Interoperability**: To make the transition from centralized collaboration networks seamless, a decentralized alternative must be interoperable with existing tools and services that users have come to enjoy and rely on. It must also be easy to migrate existing repositories without excessive reconfiguration of third-party services hooked to the repository.

5.5.2 Design Goals

- **Packaging**: When a user pushes to a git repository, the client needs to be able to determine the updates (e.g commits) and package it in such a way that it is quickly transported to other nodes on the network, validated, merged or appended into a branch. Our implementation will make use of the git-bundle tool to create bundles of git objects out of repositories, verify and unbundle them.
- **Service**: Any public node on the network must have the capability to act as git server allowing collaborators to fetch and pull git objects to/from repositories without needing to run their own servers. Users only need to use the *git remote* command to point the git client to the remote peer. The nodes on the network are able to bundle, unbundle, verify and perform other operations using git hooks [14]. If collaborator intends to run their personal servers, they can do so by starting the network client, sync up with the network and interact directly with their node.
- **Storage**: Decentralized storage protocols (e.g. Filecoin and Storj) have to enforce on-demand storage providers to comply with protocol rules that have mandated them to store all allocated objects for the duration covered by the fee received. We have opted to make storage voluntary and instead rely on a

delegate-based system where a category of network participants provide storage to the rest of the network for a share of the annual coin inflation; This category of participants are known as Archivers. An archivers sole purpose is to store and maintain all data objects generated from the operations of the network. With the archivers active, miners, endorsers, and validators may not need to store the entire state and objects of the network but may do so for latency and computational optimizations. Our storage layer will sit on top of IPFS where any node on the network can query or download any given piece of data at any time.

- **Cost**: In subsubsection 5.5.1, we listed the cheap cost of accessing the git functions of a repository as one of our core design goals. If the cost of contributing to a repository is unreasonably, then the benefits of using a decentralized collaboration framework will be overshadowed by the cost of using it; Which means users interested in starting and participating in networked organizations may do so on centralized frameworks. Most blockchains require transactions fees to prevent spamming and other forms of misuse. Without fees, the network will be unusable due to transaction spams. EOS took a different approach to minimize spam by requiring users to stake the native coin to acquire more bandwidth which allows them to freely interact with the platform up to a limit. Unlike fee-based systems, users can send money for free and can choose to recollect their staked coins.

On Ellcrys, there will be two types of repositories; *Staked* and *Unstaked* repositories.

Staked Repository: A staked repository is created when a user deposits a stake to rent a fixed amount of storage space for a repository of their choice. Staked repositories are free to access; A user interacting with a staked repository will not need to pay any fees. There is a risk of staked repositories getting spammed such that their capacity is quickly exhausted. Maintainers of these repositories can choose to enable a Push Fee configuration that will require a contributor to pay a fee for every push request. The fee is returned after the branch is merged into a primary branch. Push fees prevent spammers from misusing repository resources and spamming the network. Anyone can add more stake to the repository to increase its allocated resources.

Un-staked Repository: Un-staked repositories are the opposite of staked repos. They require all operations to include a fee, which means operations target at them are not free and are priced proportionally to the size of the transaction (transaction data + git objects).

5.6 Storage

5.6.1 Storage Problem

Today, most blockchain network storage usage and requirements continue to grow rapidly. The nature of the blockchain append-only, backwards-linked data structure causes the overall network state to grow indefinitely. For example, the Ethereum blockchain data size currently stands at 132GB and continues to grow. A troubling consequence of this trend is the potential for more full node operators to shut down their service to the network due to the increasing challenge of maintaining a fully synced node.

For many networks, storage is a problem whose solution is deferred to a future time. The number of available solution is further reduced by the type of ideology these systems are based on. For example, the Bitcoin community believes all nodes must be able to run full-nodes and tends to reject proposals that limit the abilities for all nodes to function equally.

Ellcrys is driven by a desire to collaborate; We are interested in building a sustainable ecosystem of diverse strength and capabilities with every entity working for a common core goal of enabling networked organizations capable of competing with any other structure, as such, we are more open to a system where some participants excel at providing a specific capability to the network.

5.6.2 Delegated Storage Nodes (DSN)

A delegated storage node (also known as an /archiver/) is a node that has been granted the responsibility of storing all data generated by the network from the processing of transactions. As the name suggests, a delegated storage node is selected to perform this service by users through a continuous approval voting process; This is similar to how block producers of the EOS network are elected.

Users of the network vote for their preferred storage delegates by staking some coins and casting a vote referencing the storage nodes of their choice. For a node to be eligible for consideration, it must also stack some coins to join a set of delegates.

There will be a total of 101 delegates of which the top 21 be active delegates while the remaining 80 stay on as standby delegates who will be ready to replace a delegate that has been voted out of the 21 delegate set. Additionally, the standby delegates will continuously run the proof-of-storage algorithm against the active delegates in hopes of constructing a proof that indicts an active delegate of not storing all data objects.

Delegated storage nodes do not participate in the process of creating blocks. Their sole purpose is to maintain a consistent, updated state of the network at all times. They are rewarded via inflation like other network participants.

5.7 Autonomous Functions

We describe autonomous functions as blockchain programs that self-executes instructions automatically; They are also known as smart contracts on Ethereum. Autonomous functions on Ellcrys will allow communities to create shared blockchain applications where the code-base, governance, and execution are all handled on-chain.

5.7.1 Multi-language

Today, most smart contract platforms force users to learn a new language, new best practices and developer tools. These requirements make the learning curve steep and do not help to increase developer adoption as developers need to learn new paradigms to be able to make useful bug-free applications.

However, the dominant reason for new languages is the need to prevent non-deterministic programs. Non-deterministic programs are programs that produce unpredictable output when executed multiple times. They can prevent nodes from coming to consensus on the correctness of a block.

By separating transaction execution from block processing using Hyperledgers /execute-order-validate/ model which allows verification of a transaction before inclusion in a block, we are able to support blockchain applications coded in some of the most popular, established programming language. Our function execution engine will support WebAssembly. With WebAssembly, collaborators are able to create autonomous functions using languages like C++, Go and Typescript. As WASM gains more support in new languages, these languages will be naturally supported on Ellcrys.

5.7.2 Transaction Fee

The network fee for transactions that invoke autonomous functions must be predictable. Users and developers need to be able to determine how much fees a function requires to be successfully executed. Having this information will enable users and service providers to make informed economic decisions.

In contrast, function invocation on Ethereum and associated fees are not accurately predictable This leads users to send more than required in the hopes that the necessary fee is deducted and the rest returned. An approach like this has the consequence of forcing users to purchase more coins than needed and possibly denying them of funds that would have been used for other purposes.

We are taking an approach that does not determine fees by the number or type of language operations that is executed within a function call. Instead, a transaction fee will be priced by the size of input and output data size. With this approach, contract developers can predict the cost of functions, network nodes are able to dry-run function calls to return accurate fees and users can call functions with the exact fee needed.

5.8 Autonomous Functions In Repository

In Ethereum, there are two types of accounts; Externally owned accounts (EOA) and contract accounts. EOA is an account that is controlled by private keys and has a coin balance. Whereas, contract accounts are accounts that have code associated with them; A transaction initiates the execution of the code. They also hold coin balance controlled by the code.

In Ellcrys, there are balance and repository accounts. Balance account is similar to Ethereums externally owned accounts. Repository accounts are git repositories; They include git objects which may represent any binary data. Autonomous function source codes existing in the repository. A repository can have many autonomous functions which are compiled to an executable state ready to be invoked by incoming transactions. Autonomous functions are addressed by their repository ID and the function name.

5.8.1 Fork Switch

One of open source movement core principle is freedom; Freedom to participate, to use a product and to have a divergent vision or goals for a given product. Collaborators and users must never have their ability to express their opinions and creative ideas suppressed. Developers should be able to fork autonomous functions and users should also have the ability to switch to a fork with extreme ease. Smart contract platforms of today do not support the ability for a repository to be cloned or make it easy for users to move. We intend to support this functionality by separating user data from the contracts data which will make switching between forks cheaper and without duplication of a users data.

5.8.2 Optional Immutability

Ethereum, being the pioneer of blockchain-based contracts has set a standard which includes an inability for contracts to be altered; This is understandable as the platforms goal is to support sole immutable applications that cannot be changed once deployed.

However, not all applications benefit from immutability or are meant to be contracts. Immutable smart contracts make it impossible to update the logic and state of an application that has been found to have bugs or security issues. Upgrading an immutable contract requires the application to have been initially designed to be accessible and extendable by another version of the contract which will have a different address. The fact that a different address is produced creates coordination problems as users and third-party applications need to know the new address.

Given that autonomous functions exist within a repository that may be subjected to a governance system that allows the owners and community members to manage updates, it makes sense to allow autonomous functions to be mutable or immutable. An immutable function will not permit any new upgrades while a mutable one will allow itself to be upgraded without a new address being generated for each upgrade. Communities will be able to decide what type of functions they need for themselves and their users.

5.9 Collaboration Primitives

Collaboration primitives are first class components and tools provided by the protocol to aid collaboration. These are the tools contributors will leverage to build and govern shared resources.

5.9.1 Repository

A repository is the smallest unit of collaboration within the Ellcrys network. It is the central point where collaborators organize themselves to create and govern. It includes a members list, source codes, governance configurations, many autonomous functions and an identity that links it to internal or external assets. A repository also has a base balance account and many branch accounts for receiving and managing funds. Repositories are the central point for collaboration on Ellcrys.

5.9.2 Membership

Membership determines how collaborators join or are accepted into a repository, what access or privileges they are entitled to and the duration of their membership. If a repositories membership property is set to open, it means membership is free and anyone can participate in the repositories events. Some communities may choose to have a structure that offers membership based on invitation signed by one or more persons; Potential collaborator may be required to present an invitation to gain access. Invitations can include properties that limit the invitee's access to a repositorys resources. Using autonomous functions, communities can create custom membership rules that potential contributors run to check whether they are eligible for membership. Members can be assigned roles at the point of admission or they gain their roles at a later time.

5.9.3 Role

In organizations, roles are created for various functions of a job. Roles are assigned certain permissions that enable an individual who had been assigned the role to carry a particular function. In Ellcrys, repositories can create different roles with different permissions that allow the assignee to perform various actions. Creation of roles and assignment of the roles to subjects (actions or resources) is carried out under the rules of the existing governances and membership setup of the community. Additionally, roles prevent the frequent need to call a vote for every decision which can become ineffective over time. Roles help communities know who is responsible for a resource or a specific duty.

5.9.4 Permission

Permission defines access or right to perform a specific action. Permissions are applied to roles: They grant the assignee the power to carry out a certain number of operations. A role can have one or many permissions. For instance, a community can create permission that grants member the ability to create tasks, review tasks or create invitations. There are protocol-based permissions and custom permissions; Protocol-based permissions are standard permissions provided by the protocol to manage native primitives. Custom permissions are created by repositories owners and maintainers to guard access to resources in ways that are not supported by the protocol.

5.9.5 Department (Dep)

A department is a division of a repository created to deal with a specific activity. They are similar to departments within large organizations. Members of a repository can be distributed into departments where their skills are most useful. For example, a community building a website can have frontend, backend and marketing departments where those with the right skill can function and be more useful. Departments make governance easier as they make teams focused on finding solutions specific to an area of expertise and they are suitable for solving relating disputes. Departments may also have child departments that further separate functions and activities to more granular levels. Departments have their own accounts for receiving funding from the main repository account.

5.9.6 Reputation

Reputation is a non-transferrable asset that is used to gauge someone's capacity to do something, to signal proficiency and belief in a particular characteristic exhibited by a member. They are like crypto assets that can only be transferred through engagement with specific events. For example, when a member concludes a task, she can be rewarded with reputation depending on how satisfied the repository maintainers are with the work delivered. Every repository is bootstrapped with a fixed amount of reputation that is transferred to initial founding members who are expected to award them to future members. In real-life, reputation can be gained and lost. Likewise, on Ellcrys, reputation can be lost in disputes or through a reputation decay.

In a dispute, the disputant will have their reputation slashed when they lose the dispute. Reputation decay is a mechanism that penalizes reputation holders who hoard it. When a member does not award reputation within a fixed time, a percentage of their reputation is destroyed.

5.9.7 Skill

A skill describes a collaborators ability to do something well. It indicates expertise. Skills have numeric measures which are affected by reputation gains or losses. When someone wishes to award reputation to another person, they must associate it to a specific skill they believe the recipient deserves a reputation for. Community members use skills to determine contributors proficiency in a specific field when they allocate tasks. Although, skills originate and are acquired from communities that a contributor engages with but they become attributes of an Ellcrys account that can be used as proof of proficiency within the network.

5.9.8 Task

Tasks are the unit of work in a repository. When a repository requires some work to be done, it is described as a task and includes with it other relevant properties specific to it. A task can include rewards that are claimed by whoever completes it. The reward can be in the form of reputation, tokens, the networks native coins (Ell) or something custom or peculiar to the project and coded as an autonomous function. Tasks can include reviewers who are assigned permission to verify and accept or reject submitted solutions.

5.9.9 Proposals & Voting

Most organizations today provide a mechanism by which stakeholders can formally present proposals, deliberate and vote for or against them. On Ellcrys, proposals allow members repository to raise issues, present an improvement or execute a custom action encoded in an autonomous function. Proposals solicit for votes; It may require simple or super majority to be considered accepted. The weight of a vote can be determined by how much coins, stake or reputation is held by voters. Collaborators can create custom weighting algorithms using autonomous functions.

5.9.10 Dispute Resolution

The Ellcrys protocol includes tools that make engagement between organizations fair and transparent. However, not all dealings and activities will completely happen on-chain. An organization may create a task that requires some physical, real-world activity that cannot be tracked on-chain. In this scenario, the likelihood of disputes occurring on processes that were never automated is significant.

There needs to be a dispute resolution system that is unbiased, thorough enough to reach a fair and acceptable resolution and has no connection with the parties involved. Aragon Decentralized Court[15] offers these qualities that make it a better model for organizations that may have on-chain and off-chain activities. Dispute resolution is decoupled from the organization and it can include people who have the expertise to judge specific cases. We have opted to build a system very similar to Aragon Decentralized Court system.

5.9.11 Stake

A stake is a measurement of ownership in an organization. Members of a repository can own stake in the repository such that they are entitled to any liquidity event such as an acquisition or dividend payment. Stakes can be allocated directly, through the completion of tasks or through some other custom rule built into an autonomous function. It is possible for stakes to be diluted from the admission of new members; Useful for organizations that intend to replicate real-world corporate structures in a virtual space. Additionally, members stakes may be configured to decay after a period of inactivity as a mechanism to motivate consistent engagement with the organization. Unlike reputation, stakes can be transferred between members of the Ellcrys Network.

5.9.12 Coin

Repository coins are similar to stakes except they do not grant the owners shares in the organization and as such, they are unable to benefit from liquidity event. Coins are cryptocurrencies mined through the Ellcrys merge mining functionality provided by PeopleMint. With PeopleMint, scanners on Ellcrys can specify two currencies they would like to exchange their banknotes for. Given that native coin is mandatory, they have an option to choose an additional coin. By providing the unique identity of the organization, they are able to mine new coins for the organization.

5.9.13 Middleware

A middleware is a custom autonomous function that acts as a bridge between operations executed by repository members and the Ellcrys protocol. It may be used to alter the properties of the request in question or run sub-operations. During the execution of operations, the Ellcrys protocol allows developers to hook autonomous functions to lifecycle events of operations which can be used to create more powerful behaviours.

5.10 Governance

Anytime a group of people comes together to achieve a common goal there is a need for an institution that organizes the process, makes rules and enforce decisions. Governance is how communities organize to make decisions towards a shared goal.

5.10.1 Protocol Governance

Introduction

Protocol governance refers to the mechanisms by which a blockchain community organizes themselves and make important decisions relating to the protocol. The type of governance structure adopted can have good or bad consequences There are two major governance structures in blockchain today On-chain and off-chain governance structure.

On-Chain Governance

This type of governance scheme involves a formalized decision-making process where a stakeholder creates a proposal that solicits votes from other stakeholders. An administrator must first approve the proposal before it is available to receive votes. A proposal is approved when it receives majority Yes votes. Stakeholders are required to stake funds to participate; The votes are coin weighted such that the higher the stake the more powerful a stakeholders vote becomes This has the disadvantages of fostering a system where the wealthy control the faith of the protocol and the less powerful members are ignored. Also, the possibility for stakeholders collusion is present; Large stakeholders can collude around a shared goal or agenda. It is also possible for administrators to censor proposals before they are seen by the larger community. There is also the issue of voter apathy which could expose proposals to easy manipulation due to the lack of interested members.

Off-Chain Governance

In off-chain governance, decision making does not happen in a formal system that is embedded in the codebase of the project. Instead, members of the network coordinate themselves using many channels to share their opinions and learn about the opinions of other members. These channels can be community forums, social media, meetups and other avenues provided by the project administrators.

Like off-chain governance, on-chain governance decision process begins with a proposal, but this proposal is mostly hosted on Github. Unlike off-chain governance, there is no voting. Instead, stakeholders discuss the proposal until the developers or project administrator get a sense of where the community support is strongest This process is sometimes referred to as rough consensus [16].

Off-chain governance is notorious for causing rifts within a community which results in delays in development or the creation of forks. The lack of a formal governance process leaves members dissatisfied as there is no measurable way to prove consensus. However, off-chain governance is more inclusive and resistant to manipulations by wealthy individuals or groups.

Ellcrys Governance

Ellcrys will adopt off-chain governance structures where a foundation will act as the official administrator of the project with the responsibility of coordinating the various stakeholders, promoting and building the ecosystem around Ellcrys. A major benefit to derive from this structure is speed. As a project that needs to quickly evolve, there is a need for an entity gauging the communitys preferences, carrying out research, actively contributing to the development and enforcing community decisions.

Future Governance Structure

When Ellcrys reaches a stage where the protocol and collaboration features are standardized or frozen. We intend to transition from a foundation-based governance system to a fully community-led governance system. The form this will take will be determined by experiments and the experiences amassed by community-led projects that will be built on Ellcrys.

6 Conclusion

Ellcrys is determined to create a system that allows networked organizations to be created, managed and flourish without concerns of censorship, ownership, and transparency. The ability for open source communities to build fast and scalable services that can compete with a centrally governed organization will create a world where users concerns and experience matter and where openness and transparency exist naturally. This document will continue to evolve to include new details and dimensions to our core vision.

References

[1] **Bitcoin** The decentralized digital currency https://bitcoin.org/bitcoin.pdf

[2] **Git** https://en.wikipedia.org/wiki/Git

[3] **GitHub** https://github.com/

[4] **Bernstein vs. United States** https://en.wikipedia.org/wiki/Bernstein_v._United_States

[5] **Snuffle** https://cr.yp.to/snuffle.html

[6] **Censorship of Github** https://en.wikipedia.org/wiki/Censorship_of_GitHub

[7] **Blockchain** https://en.wikipedia.org/wiki/Blockchain

[8] **Satoshi Nakamoto** https://en.wikipedia.org/wiki/Satoshi_Nakamoto

[9] **Proof of Work** https://en.bitcoin.it/wiki/Proof_of_work

[10] **Bitcoin-NG: A Scalable Blockchain Protocol** https://arxiv.org/abs/1510.02037

[11] **Proof of Activity: Extending Bitcoins Proof of Work via Proof of Stake** https://eprint.iacr.org/2014/452.pdf

[12] **PeopleMint: A Multi-Party Mining Protocol** https://storage.googleapis.com/ellcrys-docs/PeopleMint.pdf

[13] **Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains** https://arxiv.org/pdf/1801.10228.pdf

[14] **Git Hooks** https://git-scm.com/docs/githooks

[15] **Aragon Court** https://forum.aragon.org/t/aragon-court-v1/691

[16] **On Consensus and Humming in the IETF** https://tools.ietf.org/html/rfc7282