Loki

Private transactions, decentralised communication.

Kee Jefferys, Simon Harman, Johnathan Ross, Paul McLean

Version 3

July 13th 2018

Abstract

A hybrid proof of work/proof of service system offers a unique way to financially incentivise the operation of full nodes. Loki leverages these incentivised nodes to create a secondary private routing layer. Minimum node functionality on the second layer is monitored and enforced by a novel method called swarm flagging. Loki is based off a modified version of the Monero source code, assuring that all transactions achieve a high degree of privacy.

This white paper outlines the technology used in Loki. We anticipate that changes to this technology will occur as Loki continues to be developed. New versions of this white paper will be released to reflect any substantial future changes and updates.

1 Introduction

The demand for privacy in digital communications and transactions is ever increasing. User data is being collected, processed, and traded at unprecedented levels. Everything from a users browsing data and email contents, to credit score and spending habits, are gathered and sold between the worlds largest corporations and state level actors. Loki aims to provide a censorship-resistant suite of tools that will allow users to transact and communicate in private.

Bitcoin came with the promise of privacy, but what has resulted is more traceability than ever. Companies like Chainalysis and BlockSeer have taken advantage of Bitcoin's transparent blockchain architecture to track and follow specific transactions [1]. Loki is built off Monero, a cryptocurrency that has established itself as one of the most secure and private transaction networks to date [2]. However, we recognise that Monero has inherent drawbacks. Monero transactions are orders of magnitude larger than Bitcoin transactions, with significant bandwidth, processing, and disk space requirements. As the network grows, this results in a large burden on Monero node operators and offers no incentive or reward for

their contributions to the network. This makes running a node a costly and often thankless exercise. The introduction of a node reward scheme, called *Service Nodes*, mitigates this by providing economic incentives for node operators.

Service Nodes can also be used to provide a range of other privacy-centric functions if properly incentivised. Primarily, the Service Node network will allow users to transmit and receive data packets anonymously. This private communication is facilitated by each Service Node acting as a relay in a novel Sybil resistant onion routing network, having similar properties to Tor and I2P [3][4]. Furthermore, this emergent communications network is to be used as the backbone of a decentralised and end-to-end encrypted messaging service, called *Loki Messenger*, which will allow users to communicate directly and without relying on any trusted third-party, and without the requirement for both parties to be simultaneously online.

Loki is not only a resilient medium of private exchange but a platform for decentralised and anonymous internet services.

2 Basic Parameters

Loki difficulty target (blocktime)	120 Seconds
Difficulty algorithm	Zawy LWMA [5]
Hashing algorithm	CryptoNight Heavy
Elliptic curve	Curve25519 [6]

3 CryptoNote Elements

Although a full-node incentives scheme could be implemented on top of any cryptocurrency, Loki uses the Monero source code because of the high level of privacy it affords to transactions. Monero is an evolution on the CryptoNote protocol, which uses ring signatures, stealth addresses, and RingCT, giving users the ability to sign transactions and obfuscate amounts while maintaining plausible deniability [7].

For the Loki ecosystem to maintain privacy, it is important to not only provide a medium of exchange that underpins the internal economy but to also minimise the risk of temporal analysis when interactions occur across Loki's independent layers. For example, when engaging in layer-one transactional services, users should never lose the privacy guarantees they receive from the second-layer and vice versa.

3.1 Ring Signatures

Ring signatures work by constructing a ring of possible signers to a transaction where only one of the signers is the actual sender. Loki makes use of ring signatures to obfuscate the true history of transaction outputs. Ring signatures will be mandatory for all Loki transactions (excluding block reward transactions), and uniquely, a fixed ring-size of ten is enforced on the Loki blockchain. This means that each input will spend from one of ten possible outputs, including the true output (see 7.3).

3.2 Stealth Addresses

Loki makes use of stealth addresses to ensure that the true public key of the receiver is never linked to their transaction. Every time a Loki transaction is sent, a one-time stealth address is created and the funds are sent to this address. Using a Diffie-Hellman key exchange, the receiver of the transaction is able to calculate a private spend key for this stealth address, thereby taking ownership of the funds without having to reveal their true public address [8]. Stealth addresses provide protection to receivers of transactions and are a core privacy feature in Loki.

3.3 RingCT

RingCT was first proposed by the Monero Research Lab as a way to obfuscate transaction amounts [9]. Current deployments of RingCT use range proofs, which leverage Pedersen commitments to prove that the amount of a transaction being sent is between 0 and 2^{64}. This range ensures that only non-negative amounts of currency are sent, without revealing the actual amount sent in the transaction. Recently a number of cryptocurrencies have proposed implementing bulletproofs as a replacement to traditional range proofs in RingCT because of the significant reduction in transaction size [10]. Loki will utilise bulletproofs, reducing the information that nodes are required to store and relay, thereby improving scalability.

4 Service Nodes

Although Loki implements novel changes on top of the CryptoNote protocol (see 7), much of Loki's networking functionality and scalability is enabled by a set of incentivised nodes called Service Nodes. To operate a Service Node, an operator time-locks a significant amount of Loki and provides a minimum level of bandwidth and storage to the network. In return for their services, Loki Service Node operators receive a portion of the block reward from each block.

The resulting network provides market-based resistance to Sybil attacks, addressing a range of problems with existing onion routing networks and privacy-centric services. This resistance is based on supply and demand interactions which help prevent single actors from having a large enough stake in Loki to have a significant negative impact on the second-layer privacy services Loki provides. DASH first theorised that Sybil attack resistant networks can be derived from cryptoeconomics [11]. As an attacker accumulates Loki, the circulating supply decreases, in turn applying demand-side pressure, driving the price of Loki up. As this continues, it becomes increasingly costly for additional Loki to be purchased, making the attack prohibitively expensive.

To achieve this economic protection, Loki encourages the active suppression of the circulating supply. In particular, the emissions curve and collateral requirements must be designed to ensure enough circulating supply is locked and reasonable returns are provided for operators to ensure Sybil attack resistance.

4.1 Block Reward

Distribution of block rewards in Loki is conducted through proof-of-work, a robust and well-studied system for the creation of blocks and the ordering of transactions. Miners collect and write transactions into blocks and collect fees for doing so. As a consensus rule in Loki, each block contains multiple reward outputs of which only one goes to the miner.

Mining Reward:
As well as collecting transactions fees, 45% of the block reward is awarded to the miner that constructs the block.

Service Node Reward:
The second output in each block (50% of total reward) goes to a Service Node, or two Service Nodes if a relay is selected (see 6.3). Service Nodes are rewarded based on the time since they last received a reward (or time since they registered), with a preference for nodes that have been waiting longer. Each time a Service Node registers with the network it assumes the last position in the queue. If the Service Node maintains good service and is not ejected from the queue by a swarm flag (see 8.3), it slowly migrates to the higher positions in the queue. Nodes at or near the front of the queue are eligible for a reward, and once awarded, the node again drops to the last position in the queue and begins slowly working its way back up.

Governance Reward:
The final 5% portion of the block reward is distributed towards governance operations (see 9); 3.75% is sent to the Loki Foundations address which is derived deterministically each block and the remaining 1.25% is reserved for the outputs of a funding block (see 9.2.3).

4.2 Verifiable Collateralisation

Service Nodes must prove to the network that they are holding the required collateral. Privacy features inherent in Loki's design make this difficult, specifically the inability to audit public address balances or to use viewkeys to see outgoing transactions.

Loki makes novel use of time-locked outputs, which allow Loki coins to be time-locked until the blockchain reaches a defined block-height. Until this defined height, the Loki network will invalidate attempts to spend these time-locked outputs. Loki utilises this process to prove that an amount is being held by a specific Service Node, preventing shuffling of collateral.

To register as a Service Node, an operator creates a time-locked output of the required amount which unlocks after a minimum of 21,600 blocks have elapsed (approximately 30 days). In the extra field of the transaction, the Service Node operator includes the Loki address which may receive Service Node rewards. This address will also be used as the public key for Service Node operations such as swarm voting. Wallets may avoid using these Service Node registration transactions as mixins, as their true amounts and destination are disclosed and therefore are not useful in providing extra anonymity to a transaction.

Before each node joins the Service Node network, other nodes must individually validate that the said nodes collateral outlay matches the required amount, as per the decreasing collateralisation requirement. Although collateral transactions expire after 30 days, the wallet will have an opt-in automatic re-collateralisation feature.

5 Lokinet

Onion routing protocols allow for users to form tunnels or paths through a distributed network, using multiple nodes as hops to obfuscate the destination and origin of data packets. Service Nodes on the Loki network will operate a low latency onion routing protocol, forming a fully decentralised overlay network, called Lokinet. The network does not rely on trusted authorities and its state is fully derived from the blockchain. Users can connect to individual Service Nodes and create bidirectional paths for packets to be routed through. The network can be used to access internally hosted services called *SNApps* (see 6.2). Users can utilise Service Node exit functionality to browse the external internet without their IP address being exposed (see 6.3).

5.1 Low Latency Anonymous Routing Protocol (LLARP)

Underlying all applications for Service Nodes is an anonymous routing protocol, which defines the way each Service Node communicates with its peers. Loki proposes a new routing protocol called *LLARP* [12] which is designed as a hybrid between Tor and I2P to provide additional desirable properties versus any existing routing protocol. LLARP is built specifically to run on top of the Loki Service Nodes network and all LLARP optimisations consider this architecture. To understand the goals of LLARP, it is best to conduct an analysis of existing routing protocols and consider how LLARP improves upon them.

The Onion Router (Tor)

In recent years, Tor has been the most popular anonymous overlay network. The Tor network maintains a high-level of censorship resistance and has proved a valuable tool for preserving internet privacy. However, Tor is not a decentralised network as much as it is a hierarchical one. Tor is reliant on a group of directory authorities which are centralised servers operated by a group of volunteers close to the Tor Foundation [13]. These directory authorities perform two main functions. Firstly, they act as trusted reporters on the state of nodes in the network. When a Tor user (or relay) connects to the network for the first time they can connect to one of ten hard-coded directory authorities. These directory authorities provide the user or relay with a file called the consensus. This file provides a list of all of the relays, guard nodes, and exit nodes currently in operation (excluding bridges) on the Tor network. Secondly, the directory authorities also measure the bandwidth that each relay can provide to the network. They use this information to triage relays into categories, deciding whether nodes can operate as relays, guard nodes, or exit nodes.

This high level of centralisation creates points of failure that leaves Tor vulnerable. In 2014, Tor received information of a credible threat to take down the directory authority servers [14]. If the directory authorities in the United States and either Germany or the Netherlands were to be shut down, that would be enough to shut down five of the ten directory authority servers. This would result in a highly unstable Tor network, with new relays being greatly diminished in their ability to interact with the network.

Methods of communication in Tor are also limited, as Tor only allows communication over TCP. IP over Tor is possible, but it lacks support for UDP based protocols (such as VoIP).

Invisible Internet Project (I2P)

I2P takes a different approach, maintaining a higher level of trust agility by referring to a Distributed Hashing Table (DHT) to ascertain the network state instead of trusted directory authorities [15]. I2P also allows for both TCP and UDP traffic, supporting a larger scope of protocol interactions. However, I2P has not had a steady development process and over time it has accumulated technical debt, specifically in its cryptography usage. I2P uses 2048 bit ElGamal, which makes encryption and decryption slow in contrast to elliptic curve operations. While plans to migrate away from ElGamal exist in the I2P roadmap, progress has been slow.

Additionally, I2P lacks formal support for exit nodes, meaning the majority of traffic on the network is accessing internally hosted websites, called *Eepsites*. This has greatly reduced the ability for the I2P network to reach users whose main purpose for using anonymising networks is to access the wider internet.

Furthermore, the manner in which I2P is built means that the majority of users that connect to the network also become routers, which is problematic as the resulting network often lacks sufficient bandwidth to be able to build fast paths. Network speeds in onion/garlic routing networks are bottlenecked by the least capable node in each circuit, and as a result of low-performance users becoming relays in I2P, a reduction in overall performance is seen.

Finally, I2P differs from Tor in that it offers a packet-switched (rather than circuit-switched) network. Instead of establishing a single longer-term tunnel which all traffic travels through, I2P establishes multiple paths that each packet being communicated can use to use to take a different route through the network. This gives I2P the ability to transparently route around network congestion and node failures.

Both I2P and Tor have not fully mitigated Sybil attacks. A sufficiently motivated attacker that has enough time and capital to buy large amounts of relays can perform temporal analysis which undermines user privacy. The effectiveness of this analysis increases the more exit nodes, relays and guard nodes the attacker operates [16]. Tor and I2P are operated entirely by volunteers that donate both their time and money to the operation of nodes. We surmise that a network constructed from financial incentives rather than altruism can achieve a greater resilience against attacks, while providing a more reliable service.

LLARP

LLARP operates without the need to make use of directory authorities and, instead, relies on a DHT built from blockchain staking transactions, which allows Service Nodes to act as routers in the network. Bandwidth is not monitored or recorded in the DHT. Instead, bandwidth measurement and triage result from swarms (see 8.3.1) that assess each node and make a judgement on the nodes ability to provide appropriate bandwidth to the network.

In the Open Systems Interconnection model (OSI model), LLARP only attempts to provide an anonymous network layer. This means that it supports a larger range of internet protocols and it also minimises the overhead for storing file descriptors should exit nodes pass through User Datagram Protocol (UDP) traffic[17]. Additionally, LLARP opts for packet-switched based routing instead of tunnel-based routing, allowing for better load balancing and redundancy across the network.

End users of Lokinet are not expected (or even allowed) to route packets, this means that Lokinet exposes itself to a much lower attack surface for a Sybil attack due to the significant capital outlay required to begin Service Node operation.

6 Loki Services

Similar to the investment that miners make into hardware, each Service Node operator freezes Loki coins when they begin to operate a Service Node. This frozen capital serves two purposes.

1. Every Service Node operator has a sufficiently large stake in the success of the network. Should any Service Node operator provide poor performance to the network, or act dishonestly, they undermine and risk devaluing their own stake within the network.
2. It provides an opportunity for more aggressive enforcement; if the network is able to effectively limit dishonest nodes from receiving a reward, then dishonest nodes must bear the opportunity cost of both the reward loss and the remaining lockup time on their collateral.

If we take the above points to be true, and we can enforce aggressive punishments for poorly behaving nodes (see 8.3), then we can create groups of Service Nodes which can be queried to come to consensus on the state of the blockchain or to enforce special off-chain node behaviour (see swarms 8.3). In Loki, this behaviour pertains to both networking and storage activities. These off-chain activities are combined to be the back-end of user-facing applications that leverage these desirable properties, which are called *Loki services*.

6.1 Loki Messenger

The first Loki service to be developed and deployed on the Loki network will be a decentralised, end-to-end encrypted private messaging application called *Loki Messenger*.

End-to-end encrypted messaging applications that provide a platform for users to send messages without revealing their contents already exist, however they rely on centralised servers that can be targeted, blocked and shut down [18][19]. These centralised service models present a high-risk for the anonymity of communicating parties, as they often require the user to register a phone number or other identifying information and connect directly via the IP address of the user. This information could be extracted from servers through data leaks or legal processes and used against the user. Leveraging the Service Node architecture on the Loki network, we can deliver a service similar to popular centralised encrypted messaging apps, such as Signal, with a higher degree of privacy and censorship resistance.

6.1.1 Messenger Routing

Message routing on the Loki network changes depending on whether the receiving user is online or offline. When both users are online, higher bandwidth communications can take place due to the fact that messages do not need to be stored on the Service Nodes.

In Loki, a public key acts both as long-term encryption key and a routing address. In the most simple case, this key should be exchanged out-of-band to ensure protection against a man-in-the-middle attack. Such an exchange should take place either in person or through another secure mode of exchange (see user authentication 6.1.2).

Online Messaging

Once Alice knows Bobs public key, she assumes he is online and tries to create a path to him. Alice does this by querying the DHT of any Service Node and obtains any introduction set that corresponds with Bobs public key. In LLARP, introduction sets list the *introducers* that each user maintains. It is through these introducers that paths can be established. With Bobs introducer, Alice now chooses three random Service Nodes to act as intermediary hops between her origin and her destination (Bobs introducer). A path has now been established, through which Alice and Bob can transmit messages. If correctly authenticated, and using OTR (see 6.1.2), Alice and Bob can now communicate while maintaining a high-level of privacy.

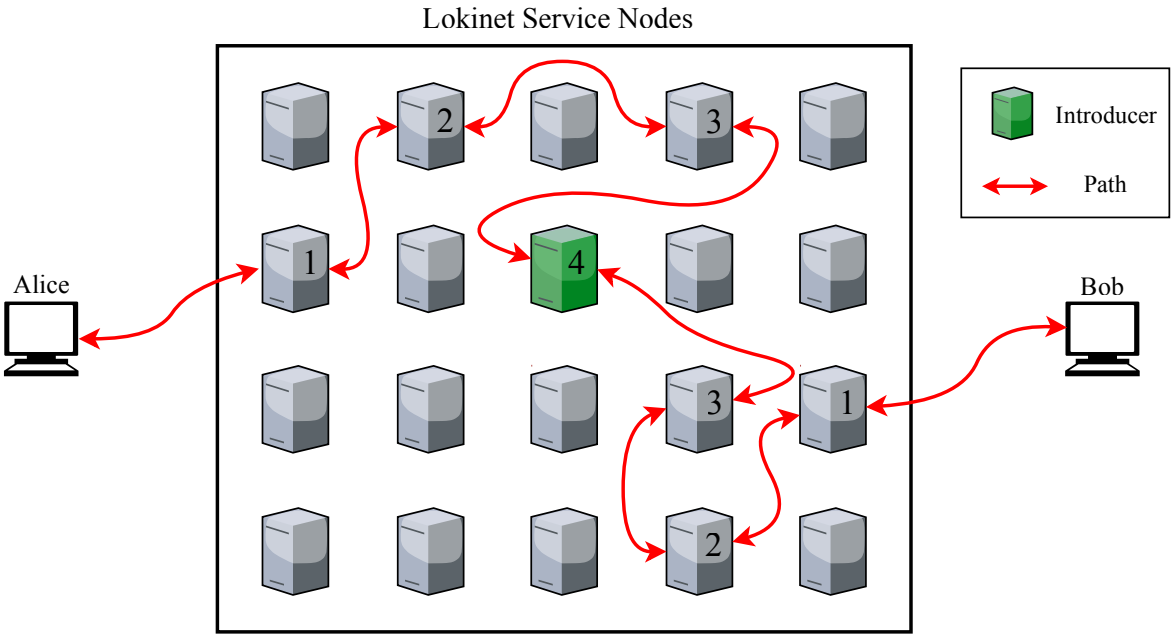

Figure 1: *Simplified version of online routing where Alice communicates with Bob, using random Service Nodes to establish a path through the network.*

Offline Messaging

If Alice fails to receive a response from Bob, she can then initiate the offline messaging process. Offline routing uses a modified version of Postal Services over Swarm (PSS) [20]. Swarms are logical groupings of Service Nodes, based both on their public keys and the hash of the block that their staking transaction first appeared in. Each swarm has a swarmID and consists of nine nodes. To send a message to Bob, Alice can use his public key to calculate which swarm Bob belongs to. With this information, Alice can anonymously route a message through the network to a random Service Node in that swarm. When a Service Node receives a unique message destined for its swarm, it must distribute that message to the other eight nodes in the swarm. All nodes are additionally required to store messages for their allocated Time-to-live (TTL), (see 8.3.1). When Bob comes online, he can query any two nodes in his swarm for messages he can decrypt. Offline messaging is protected from spamming with a small proof-of-work that is attached to each message (see 8.2)

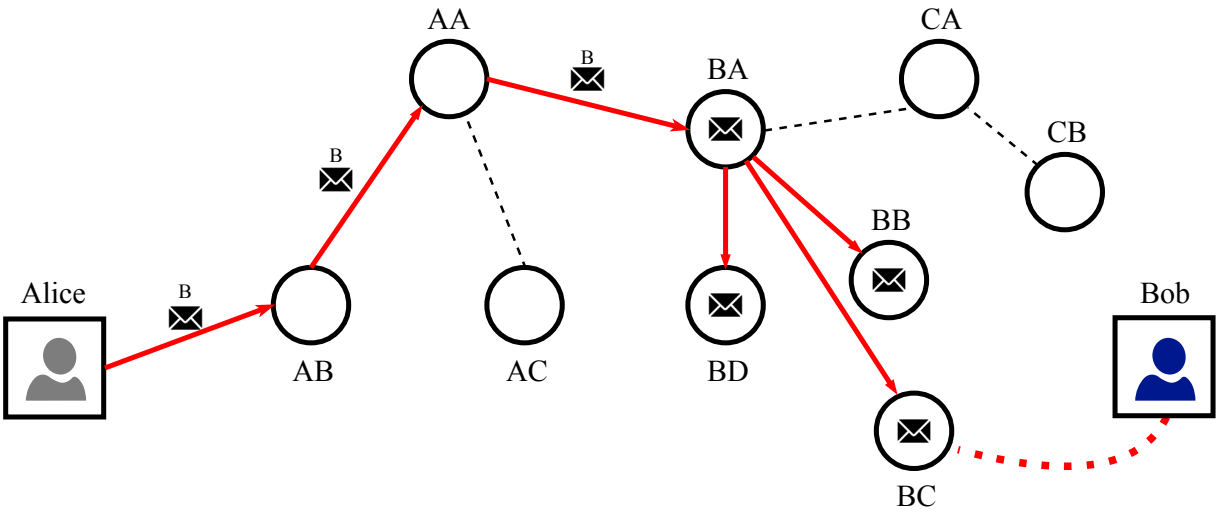

Figure 2: *Alice sends a message to Bob, Bobs assigned Swarm is B, When Bob comes online, he queries a random node in his swarm and receives Alices message*

6.1.2 Messenger Encryption and Authentication

Once a message chain is established, Loki Messenger enforces Perfect Forward Secrecy (PFS) and Deniable Authentication (DA). PFS and DA are key concepts of the Off The Record (OTR) messaging protocol [21]. Centralised services, such as Signal and WhatsApp, use encryption features that maintain OTR protections. Loki models its OTR implementation off the existing Tox protocol, which is a distributed, peer-to-peer instant messaging protocol that uses the highly audited NaCl library [22].

PFS enables resistance from attacks where a long-term key is exposed. A new shared encryption key is used for each session, so if a single session key is revealed, the whole message chain is not compromised. If a third-party wanted to break the encryption of a message chain they would need to obtain the keys for every individual session. PFS ensures that Loki Messenger is extremely difficult to compromise when compared to existing methods, such as Pretty Good Privacy (PGP) encryption, where only one long-term key pair is required to compromise the whole message chain.

DA refers to the ability for two parties to prove to each other that they are the sender of each new message. However, a third-party cannot ascertain who the true sender of any message is. When using DA, Message Authentication Codes (MACs) are published after each session, allowing third-parties to plausibly create messages that appear as if they originate from the senders public address. When correctly implemented, it is impossible for any third-party to prove that a sender of a specific message was the actual sender.

User Authentication

Authentication of users is important to ensure protection against man-in-the-middle attacks. For example, if Bob is expecting a message from Alice but does not yet know what her public key is, then a third-party (Eve), could send a message to Bob pretending to be Alice. This is why users should authenticate each other before sharing personal information.

Like Pidgin and other OTR messaging services, Loki Messenger uses Pre-Shared Key (PSK) authentication. Users have multiple options for the establishment of a PSK. They can establish a key out-of-band, or alternatively, they can agree on a PSK over Loki Messenger by

asking the other a question which no third-party would know the answer. Loki will implement PSK authentication based on a modified version of the Pidgin encryption authentication plugin [23].

6.2 SNApps (Service Node Applications)

The function of SNApps is similar to so-called hidden services in Tor which have flourished. SNApps provide an even higher-degree of anonymity than can be achieved when accessing externally hosted content. SNApps allow for users to setup and host marketplaces, forums, whistle-blowing websites, social media, and most other internet applications on their own machines or servers while maintaining full-server and user-side anonymity. This greatly expands the scope of the network and allows users to build meaningful communities within Lokinet.

SNApp operators use the traditional server-client model with the key difference being that Service Nodes will be intermediaries in a users connection through Lokinet. When a SNApp wishes to register on the network, it must update the DHT with its descriptor. This descriptor contains various introducers, which are specific Service Nodes that users can contact to form a path to the SNApp. When these paths are set up, users can connect to the SNApp without either party knowing where the other is located in the network.

6.3 Exit Nodes

Exit nodes allow users to make requests to the wider internet and return those requests through a onion routing network. If used correctly, exit nodes allow users to browse the internet privately and without the users IP address being exposed to the server.

Although the operation of exit nodes is essential to Loki's extended utility, forcing all Service Node operators to act as exit nodes could be detrimental. Acting as an exit node may expose the operator to legal risks, as users of the exit node may perform malicious activity whilst using it as a proxy. As exit nodes simply relay traffic from the internet to the end user, exit nodes often receive Digital Millennium Copyright Act (DMCA) requests or are often assumed to be the source of hacking attempts. Although in most jurisdictions safe harboring laws may protect exit node operators, internet service providers that carry Service Node traffic on their servers may fear legal risks and often cut off service to the exit node.

Upon startup, a Service Node is assigned a relay flag and is restricted to routing packets within Lokinet, but never makes requests to the wider internet. An operator must opt-in if they wish to become an exit node, in doing so they demonstrate an understanding of the additional risks while also submitting to additional Swarm tests (see 8.3.1).

Opting-in as an exit node affords an operator double the reward of a normal relay when selected for a block reward. This incentive is provided to ensure that exit node operators have sufficient financial incentives to operate exit nodes, helping to protect against Sybil attacks specifically targeted to take over the exit node network. This is a vulnerability which Tor suffers from due to its low ratio of exit nodes to relays.

6.4 Remote Nodes

On any given cryptocurrency network, storing a full copy of the blockchain is not possible or practical for many users. In Bitcoin and Ethereum, users can choose to connect to a public full node that holds a copy of the blockchain and can query and submit transactions to the network. This works because Bitcoin and Ethereum full nodes can efficiently search the blockchain for transactions that have the users public key as the target.

Due to the construction of CryptoNote currencies, public full nodes (called remote nodes) are put under much more stress. When a user connects to a remote node, they must temporarily download every block (upon wallet creation or since last checked block) to their local machine and check each transaction for a public transaction key which can be generated from the users private view key. This process can cause a significant performance impact on remote nodes. Considering that there is no reward for this service, it can dissuade users from operating syncing services for light clients. CryptoNote mobile wallets are often unreliable and sometimes have to switch between remote nodes multiple times before establishing a reliable connection to either scan the blockchain or to submit a transaction.

Additionally, malicious remote node operators running one of the few popular nodes can record the IP address of users as they broadcast specific transactions. Although no information about the actual transaction is revealed by this attack, specific IP addresses can be linked with transactions which can then be used to establish a link to a real-world identity, compromising privacy.

Loki circumvents these issues by requiring each Service Node to act as a remote node that can be used by general users. Service Nodes naturally lend themselves to this work as they already hold a full copy of the blockchain and form a widely distributed network of high bandwidth nodes. By using Service Nodes as remote nodes, there is an inherent financial limitation as to how much of the remote node network any given party can own, and therefore, how much data a malicious node operator can collect.

6.5 Blink

In a typical blockchain system, the confirmation time for any given transaction is the time it takes for a transaction to be included in a block. Because of competing miners, withheld blocks, and Finney attacks, recipients usually require a number of additional blocks to be created on top of the block which holds a transaction before it is considered to be complete [24]. Depending on a multitude of factors specific to each blockchain, this process can often take 10-60 minutes, which is inconvenient for merchants and customers who must wait for confirmations before they release goods or commence services.

Because of Loki's Service Node architecture, near instant transactions are possible. *Blink* enables the same transactions that would occur on the Loki mainchain to be confirmed before being included in a block, assuring both the sender and the receiver of the validity of the transaction and protecting the receiver against a double spend.

Blink works in a similar fashion to DASH's InstantSend. Each block, a Service Node swarm is deterministically selected to act as a set of witnesses that confirm a transactions validity and lock the transaction from being spent twice. Instead of the unspent outputs used in the transaction being locked (like in DASH), key images are locked. Key images are unique keys that are attached to each unspent output in a ring signature. To provide immediate

confirmations, Blink gives authority to the selected swarm to signal to the network that a key image associated with an output should be locked until the transaction is included in a block. If a double spend of the same unspent output is attempted, an identical key image is produced, which would be rejected by the swarm and thus the network as a whole.

Users will have the ability to pay a higher fee to send a Blink transaction which will confirm in seconds rather than in minutes. This opens up a range of new use cases for Loki where face-to-face payments become increasingly practical and online payments become easier to integrate. All of the privacy features inherent in Loki are uncompromised throughout this process.

7 CryptoNote Alterations

As a cryptocurrency, Loki is functionally similar to its fellow CryptoNote coins. However, there are key differences beyond the addition of Service Nodes and the associated functionality that comes with them.

7.1 ASIC Resistance

An Application-Specific Integrated Circuit (ASIC) is a computer chip that is built specifically for a single function. In the context of mining, ASICs are used to compute for specific hashing algorithms. They pose a risk to decentralisation because they outpace all other mining methods, are manufactured by specific companies, have very limited distribution channels due to the specialised nature of the hardware, and they require significant capital costs to develop and operate profitably. There are potential benefits to ASICs, such as the capital cost requirements that miners must undertake to invest in algorithm specific hardware which makes it less likely that they would behave in a manner that undermines their own investment by acting dishonestly. However, the distribution and manufacture of ASIC chips, with mature hashing algorithms, is still centralised around a few large companies. These companies can refuse shipment to certain areas, decide what regions and customers get the best performing ASICs, and they can structure limited runs and manipulate prices.

To prevent ASIC miners from monopolising the network hashrate, many cryptocurrencies developed ASIC resistant hashing algorithms, like Scrypt and Ethash [25][26]. Until recently, Monero used the CryptoNight hashing algorithm, which requires large amounts of L3 cache to operate. In theory, this should have made it difficult to produce an ASIC chip due to large memory requirements. However in 2018 Bitmain released the X3, a CryptoNight specific ASIC that could effectively mine at ten times the speed of a graphics processing unit (GPU) [27]. Other hashing algorithms have suffered similar fates, with Scrypt, Ethash, and Equihash all now being mined by ASICs.

To combat the use of ASICs, Monero proposed a strategy of hard forking every 3-6 months to slightly change the CryptoNight hashing algorithm (the first fork moving to CryptoNightV7 [28]). The capital and time required to build an ASIC is significant, and with highly specific hardware designs, slight tweaks in a hashing algorithm should invalidate the chip design, wasting the time and capital investment of ASIC manufacturers. However, this approach introduces its own issues. If changes made to the algorithm are insufficient to prevent ASICs being reprogrammed, then the network can become vulnerable to hashrate centralisation until another hard fork is possible. Field Programmable Gate Arrays (FPGAs) should also

be considered in ASIC resistance strategies, where infrequent, slight changes to hashing algorithms can be easily reprogrammed for FPGAs. Another concern is that regular changes to core consensus mechanisms introduce the chance of unintended bugs and generally centralise the development of such changes around the core team of developers.

A number of alternative proof-of-work algorithms have been proposed to combat the need to hard fork regularly, including provably memory-hard hashing algorithms like Argon2, Balloon hash, and polymorphic hashing algorithms like ProgPoW and RandProg [29][30][31][32]. The Loki team will be publishing additional research on the aforementioned algorithms to develop a long-term solution to ASIC resistance.

While this work is undertaken, Loki will incorporate a version of CryptoNight called CryptoNight Heavy, which maintains ASIC resistance against CryptoNight ASIC miners. CryptoNight Heavy differs from CryptoNight V7 in a number of ways: it provides an increase in scratchpad size to 4mb, and a change in the way implodes and explodes are handled [33]. These changes differentiate it from the largest target for ASIC miners which is Monero's CryptoNight V7 and also provide more robust protection against ASIC development until a more permanent solution is proposed.

7.2 Dynamic Block Size

Like other CryptoNote coins, Loki does not have a fixed block size. Instead, the block size changes over time, growing to include more transactions as the network reaches higher transaction throughput. The Loki block size scales by observing the median block size over the last 100 blocks and slowly retargets the maximum size of any new blocks accordingly.

The long-term concern in other cryptocurrencies is that large block sizes burden the nodes that store and verify transactions. As block sizes grow, nodes that run on lower grade hardware are unable to process and propagate new blocks, leading to centralisation of the node network among those with a commercial interest in maintaining nodes. This can be concerning because distributing the blockchain across many nodes allows for the state of the chain to be confirmed among many different parties, adding to its validity and censorship resistance.

In Loki, a portion of the block reward is given to Service Nodes that process and propagate blocks as full nodes. Because Service Nodes with insufficient bandwidth and performance are dropped from the Service Node network, (see 8.3) the reward pool self-enforces a minimum performance requirement. This incentive structure not only ensures that the node count remains high, but that the said nodes are of a sufficient performance level to successfully share blockchain data across the network, irrespective of how large the blockchain grows or how demanding the bandwidth requirements are. Even so, transaction size optimisations are still required to ensure that the network scales efficiently so as to keep the Service Node operating costs down so that a high node count can be sustained in the long term.

7.3 Ring Signature Size

Ring signatures are used to hide real outputs amongst others in any given transaction. The size of a ring signature refers to how many mixins are used to construct the ring. Monero currently has an enforced minimum ring signature size of seven, with six mixins used alongside the real unspent output in a transaction.

The effect of larger ring-sizes has been sparsely studied, however, in paper 0001 (published by the Monero Research Lab), the effect of differing ring-sizes was analysed versus an attacker who owned a large number of outputs on the blockchain [34]. It was found that higher ring-sizes reduce the timeframe in which a malicious attacker who owned a large number of unspent outputs would be able to perform effective analysis of transactions. Mandating larger ring-sizes also protects against a theoretical attack known as an EABE/Knacc attack [35], where a third-party (i.e. an exchange) can perform limited temporal analysis on transactions between two users.

Additionally, Monero has no maximum ring-size enforced by network consensus rules. Many wallets like the Monero GUI wallet cap the ring-size at 26. However, a user is free to manually create a transaction with whatever ring-size they wish, as long as it is above a ring-size of seven. This is problematic since most wallets have a default ring-size of seven. Increasing a transactions ring-size above seven makes it stand out (Figure 3). Further, if an individuals transactions were to always use a non-standard ring-size in Monero (ten for example), a passive third-party could analyse the blockchain and infer patterns using temporal analysis.

transaction hash	ring size	tx size [kB]
3feaff3f48de0bc4c92ec027236165337b64df404aca098e212c1215e9456697	7	13.47
39d484f7c0a2e8f3823a514056d7cb0bf269171cb4582e05955d4c5ee995cad0	7	13.47
e08f5a937e725011bedd44075334ae98dcca32749da231c56da1278d49c0a231	7	13.50
ab35e69d9cca39219c90df8b2b7aab4a54c82127fb1fbaae65d76357f8f76387	7	13.50
6d8ccd56dc2d3eb7de03ba767f0dbf4d5f42ae91e67f4c28f16d6f8b0229c272	10	13.87

Figure 3: *xmrchain.net (Monero block explorer) showing how non-standard ring sizes stand out*

Loki improves on both of these problems by statically enforcing ring-sizes, and setting the ring-size to ten. Statically setting the maximum ring-size protects users who construct rings with more than nine mixins and setting the ring-size minimum to ten more effectively prevents an attacker who owns a large number of outputs from discerning the true outputs spent in a ring signature. Larger ring-sizes also increase the default churning effectiveness non-linearly, becoming more effective as ring-sizes grow.

In the current transaction scheme, increasing the ring-size to 10 would lead to a 2.6% increase in the size of the transaction. However, when Bulletproofs are implemented it will account for about a 8 - 13% increase in the size of a transaction. This is because of the overall reduction in transaction size caused by Bulletproofs. Increasing the minimum ring-size may present a problem on a network that lacks architecture to support larger sized transactions, due to the increased overhead. With Loki however, this burden can be carried by Service Nodes that are incentivised to operate and provide sufficient bandwidth.

8 Attack Prevention

8.1 IP and Packet Blocking

Although the Service Node network has no central points of failure, two significant censorship threats face the network; namely harvesting attacks and deep packet inspection [36][37]. Harvesting attacks would seek to gather the IP addresses of all operating Service Nodes on the network and use ISP level firewalls to block connections to those particular addresses. This type of censorship is regularly performed on the Tor network in China [38]. Deep packet inspection (DPI), aims to investigate the structuring of each individual packet that passes through a firewall, and selectively drop or block packets that appear to relate to a particular service. Again, DPI has been used extensively by state-level actors [39].

Much work has been done to design systems which evade DPI. Users can leverage types of pluggable transports which alter the signature of each packet aiming to appear as normal unblocked traffic. IP blocking is generally avoided by running domain fronting bridges which will encrypt traffic as HTTPS requests to unblocked services like Azure or Cloudflare. Once they reach the unblocked service, the bridge will forward the request to the desired location. In the case of domain fronting, it becomes difficult for a state level actor to prevent the flow of all traffic to popular bridges without causing significant disruption to the general usage of the internet.

Governance mechanisms built into Loki (see 9) can be used to operate domain fronting bridges so that users can access Loki services in countries where large-scale internet censorship policies are at play. Additionally, OBFS4 pluggable transport support will be bundled with the Service Node release of the Loki wallet to help further protect against DPI [40].

8.2 Denial of Service Attacks

Users of decentralised blockchains are not required to provide digital or physical identifiers. This can be beneficial to users who lack identity or are being persecuted because of it. However, systems that do not require identification render themselves vulnerable to Sybil attacks, where a malicious actor produces numerous false identities (in Loki's case, numerous public-private key pairs) and uses these identities to spam the network with requests.

Many cryptocurrencies have struggled with this problem, and are forced to implement either a fee-for-service model or a proof-of-work model. In fee-for-service models such as Siacoin, users pay for the services that they use. In Siacoins case, the cost is determined per TB of storage per month [41]. Fee-for-service models are effective at reducing Sybil attacks, however, they drive many users away from the system especially when similar services are available for free (such as Google Drive and Onedrive in the case of Siacoin). Proof-of-work systems such as those used in Hashcash and Nano require users to calculate a small proof-of-work before sending a message or transaction [42][43]. These small proof-of-work systems are arguably more egalitarian than the fee-for-service model but can fall prey to attackers who possess large amounts of computing power.

Loki proposes a modified proof-of-work scheme to address the two largest Sybil attack surfaces in the Loki system; offline messages and path creation. Offline messages present a potential target because each message must be stored by a swarm of nine nodes. Potential abuse could arise where a malicious user overloads a particular swarm with a high volume of

messages that it would have to store. In path creation attacks, the attacker seeks to engage in the path creation process with as many nodes as possible, taking up bandwidth resources and denying service to users who create paths through the network for legitimate purposes.

To prevent both attacks, the Loki network requires that a short proof-of-work be attached when both messages and paths are created. For messages, this proof-of-work is calculated as a Blake2b hash of the message. For path creation, the proof-of-work is sent along with the request for a node to be included in the path building process. To ensure scalability and accessibility for mobile users, the proof-of-work difficulty requirement is fixed based on the Time-to-live (TTL) of the message or the path, and not based on global network activity.

8.3 Swarm Flagging

When nodes operate in a trustless environment without a centralised leader enforcing overarching rules, maintaining proper node behaviour on the network becomes difficult. Although Service Nodes in Loki must hold the correct collateral requirement, they may choose to not route traffic or store data in their memory pools. Because this option is financially beneficial (using less bandwidth/CPU cycles/storage), a system of distributed flagging must be proposed to remove underperforming nodes.

For Loki, such distributed flagging faces major implementation issues. Fundamentally, every Service Node is financially incentivised to flag every other Service Node as a bad actor. This is because when a Service Node is flagged it will face removal from the staking pool and thereby increase the flaggers chance at winning a reward. One potential method of distributed flagging is one in which evidence is provided when a flagging event occurs, however, this solution falls prey to nodes fabricating evidence in their favour. Conversely, flagging without restrictions allows either single nodes or groups of collaborating nodes to intentionally flag honest nodes in order to improve their chances of winning block rewards. To circumvent these issues, Loki proposes *swarm flagging*.

Swarm flagging works by using existing swarms (see 6.1.1) to choose members that will participate in each testing round. Each Service Node holds a copy of the blockchain, and each block created by a miner will deterministically select a number of test swarms. Every block, 1% of the networks swarms are selected for participation in a testing swarm. To calculate participating swarms, the hash of the five previous blocks is used to seed a Mersenne Twister function which then selects swarms by order of their position in the deterministic list.

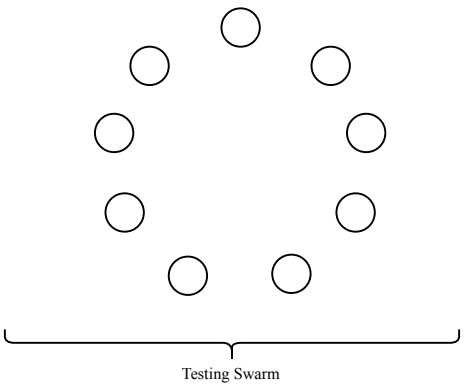

Figure 4: *A testing swarm is a selected swarm of 9 nodes*

When a swarm has been selected to participate, each node in that swarm is expected to conduct a number of tests on every other node in the swarm. These are not active tests; rather each node stores historical information about its interactions with every other node within its swarm. Information about bandwidth, message storage, blockchain requests, and exit node functionality are collected and retained over time. New swarm entrants that have yet to gather this information can query Service Nodes outside of their immediate swarm so as to gather data on each of the Service Nodes they test.

Each Service Node decides how to vote on each of the other swarm members. Once it has made its decision based on the aforementioned tests, it collects and broadcasts its votes to the swarm. Each node in the swarm can now check the votes for all members. If any single node in the swarm has over 50% of the nodes voting against it, any swarm member has the required information to construct a deregistration transaction. Once this transaction is validated and included in a block, all Service Nodes update their DHT, purging any nodes that were voted off.

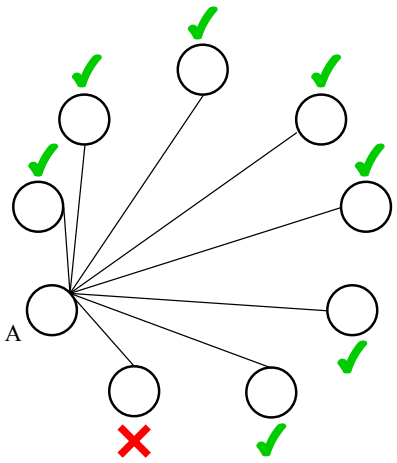

Figure 5: *Dishonest node is tested by node A and fails a test. Node A comes to local understanding of which nodes are failing or passing tests.*

8.3.1 Testing Suite

In order to allow the network to self-enforce performance standards, Service Nodes must be equipped with the required tools so as to test other Service Nodes. These tests should cover the scope of all functionality provided by Service Nodes to prevent lazy masternode attacks [44]. In this initial design, four fundamental tests are proposed. Further tests may be added to the test suite as the function of Services Nodes expands.

When an operator first runs the Service Node software, an empty file with a predetermined size is allocated on disk to ensure that space is present for tasks that require storage. Next, a simple bandwidth test is conducted between the Service Node and a geographically distributed set of testing servers run by the Loki Foundation. These checks are optional, and Service Nodes are allowed to skip, ignore or fail them, and join the pool of untrusted Service Nodes. However, running and passing these tests provides a good indicator to any would-be Service Node operator as to whether they should risk locking collateral in a node that may not meet minimum requirements. Once a Service Node joins the untrusted Service Node pool, their collateral is locked and they are tested by the next chosen swarm. Swarm tests are enforced via consensus and new entrants to the Service Node network cannot evade these

tests. If a node passes all swarm tests, they are awarded the trusted node flag and can begin routing packets. Failing this, they are removed from the network and their collateral remains locked for 30 days.

Bandwidth Test

The bandwidth test forms the backbone of the Loki network test suite. If a node passes this test then it is assumed to be honestly routing packets above the minimum threshold.

Each time a node interacts with another Service Node, it will make and retain a record of the incoming bandwidth provided. Over time, nodes will be included in thousands of paths and route millions of messages. These interactions will form the basis of each nodes bandwidth tables. From this table, a node can respond to bandwidth tests about Service Nodes inside its swarm.

All nodes are also expected to respond to queries of their own bandwidth tables from other nodes. This means that even nodes who have recently joined the network can query the wider network for information about any specific node in their swarm.

Message Storage Test

Message storage is essential for offline messaging functionality for users of Loki Messenger. Service Nodes must be tested for their ability to cache messages and serve them to users over the course of the message's Time-to-live (TTL).

Users sending offline messages randomly select a Service Node within the destination users swarm. This node must distribute a copy of the message amongst the rest of the swarm. Depending on the proof-of-work attached to the header of the message, Service Nodes that receive a copy will store the data for the TTL. As the TTL on the original message reaches finality, the distributing node sends a nonce to all other members of the swarm. The swarm uses the nonce adding it to the message then hashing the result and then finally sending it back to the distributing node. This test ensures that Service Nodes hold messages until TTL finality, and face eviction if they are unable to produce the correct message digest. As the sampling of the distributing node is random, over time each Service Node will be able to collect performance data on their swarm peers.

Blockchain Storage Test

Service Nodes are expected to hold a full copy of the Loki blockchain. By holding a full copy of the blockchain, Service Nodes can perform a number of tasks that are essential to users of the network including acting as a remote node, validating transactions, and locking transactions in Blink.

As honest nodes also hold a copy of the blockchain, a dishonest node could avoid holding a full copy by simply requesting blocks from an honest node when tested. To avoid this outcome, the blockchain storage test is designed so that honest nodes that hold a copy of the blockchain can pass this test, while dishonest nodes cannot.

To achieve this, the testing node requests each tested node to make a selection of K random transactions within the history of the blockchain which are then concatenated and hashed. This hash is then be returned to the testing node. By measuring the latency of this request, the testing node can compare the latency with the expected return time T. The exact value for T will be set to accurately differentiate expected latency between loading from disk and downloading blocks from the network. For any attacker, it should be infeasible to download and hash K blocks within T, and thus piggybacking attacks become difficult.

Exit Node Test

Service nodes that opt to act as exit nodes receive additional rewards, and so functional tests are required to ensure this extra reward is not abused.

For functional exit testing to occur, a Service Node must be able to emulate the natural search behaviour of a human. If a Service Node can detect that it is being tested, it can respond only to tests and discard legitimate user requests. Emulating natural page request behaviour is difficult, however, exit tests can be designed in such a way so as to make the overhead of sorting between legitimate requests and tests sufficiently difficult so that the difference in bandwidth cost between running a legitimate node and a malicious node is negligible.

Service Nodes use a list of search engines, held locally, combined with a dictionary so as to construct pseudorandom natural search terms. The search terms are then fed into the search engines and web pages are randomly chosen from the results. The Service node can now build a path with random nodes acting as relays and the node being tested as the exit node. From this exit, the Service Node requests the webpage result generated from its pseudorandom search. If the result returned by the exit node matches the result as generated by the Service Node, then the exit node is deemed to have passed the test.

9 Governance, Funding, and Voting

Governance is an essential part of cryptocurrency design and should be supported at the protocol level. The risk of weak, informally defined governance has been studied extensively throughout the history of blockchain technology. Bitcoin and Ethereum experienced contentious hard forks that split the focus and efforts of their respective communities. Although hard forks can be used as a governance strategy, they should always be considered as a last resort rather than the solution to every contentious issue. The Loki governance system is designed to resolve potential issues by providing a structured environment for discourse and representation, and also to source funding for the development of Loki without reliance on external influence or altruism.

Beyond the prevention of hard forks, governance structures should create the means to internally fund new projects which improve upon the Loki ecosystem. Internally funding projects can prevent the formation of special interest groups that do not necessarily have motives that are in line with the users, miners, or Service Nodes. We have seen this in Bitcoin and various Bitcoin forks with the formation of for-profit companies, such as Blockstream, Bitcoin ABC, and Bitcoin Unlimited, that have been frequently accused of hiring developers to make protocol-specific changes to Bitcoin and Bitcoin Cash aimed to further their own business objectives or follow their specific ideology.

It is for this reason that in every Loki block, 5% of the reward is allocated for the purpose of network governance. This provides a steady flow of Loki that will be distributed amongst community projects, software developers, and integration teams. Of this 5% block reward, 3.75% is controlled by the Loki Foundation and 1.25% is controlled by the Service Nodes through the Loki Funding System. Which encourages fair representation of the Service Nodes and allows for community funding proposals that can occur outside of the direct control of the Loki Foundation.

9.1 The Loki Foundation

The Loki Foundation is a registered non-profit organisation based in Australia. This central legal entity exists to allow the Loki Project to operate within a well defined legal framework and to give those working on the project legal protections and obligations. The Loki Foundation was incorporated in Australia in 2018 and uses the same constitution as the example provided by the Australian Charities and Not-for-profits Commission (ACNC) [45]. This constitution gives the Foundation the same corporate governance structure as many other non-profit organisations, where the company has no shareholders or beneficiaries, the governing board members each have seats with term limits, and conduct actions by voting on resolutions put forward by their fellow members. The Loki Foundation is structured to achieve registered charity status in Australia.

This organisation is constitutionally bound to spend any income (including the governance block reward) on the furthering of the project and aligned initiatives. As an externally audited organisation, transparency is critical to maintaining any registered charity status the Loki Foundation receives, and to assure the general public that the Loki Foundation remains honest and keeps spending within reasonable bounds. The Loki Foundation is accountable both to the community and its auditors. Should this system ultimately fail to serve Loki and its surrounding projects, hard protections exist. Should a hard fork with enough network consensus arise, there exists an opportunity to remove or replace the Loki Foundation as the recipient of this block reward.

9.2 The Loki Funding System

Although the Loki Foundation is made from a diverse group of individuals who represent the Loki Project, the Foundation is subject to both its own governing constitution and the laws of Australia. This could prove to be a limiting factor in the range of decisions the Foundation can make. The Loki Funding System allows for a portion of the block reward to be acted on purely by a vote from the Service Nodes. Service Nodes represent entities from all over the world and are not beholden to input from the Loki Project Team or Foundation, this allows them to reach a new level of autonomy in the decisions they can make. Service Nodes are the most staked participants in the network and they are financially incentivised to make decisions that grow the value of Loki.

9.2.1 Proposals

Every proposal that is put before the Service Nodes is published on the Loki blockchain. If a given party wants to present a proposal to the Service Nodes, the party must construct a proposal transaction. Because the proposal transactions contents must be readable and outputs must be burned, they forgo the privacy features of typical Loki transactions.

Funding blocks are created every 43,000 blocks (approximately 60 Days). Proposal leaders can submit their proposals at any time during this period. However, it should be considered that the closer they submit to the beginning of each proposal phase, the more time they have to gain votes from each Service Node.

Attached to each transaction is an extra field which contains the information that each Service Node needs to understand to vote on the proposal. This information includes; a

proposal title, a URL linking to a detailed explanation of the proposal, the amount of Loki the proposal is seeking, a payment address, and an escrow agent if chosen.

Pending agreement from the Loki Foundation, users who make proposals can also elect for the Loki Foundation or any other third-party to act as an escrow agent, releasing funds as milestones are reached. Additionally, to encourage a high-standard of proposals and prevent spamming of these transactions, each proposal transaction must burn a non-trivial amount of Loki.

9.2.2 Voting

Each Service Node carries a specific key for voting. This key can be exported and utilized to vote on behalf of a Service Node without having to login to the server where it is hosted.

Voting does not occur on chain, rather, each Service Node signals their support, dissent, or abstinence for each active proposal on the blockchain. Service Nodes can vote on proposals as soon as they are committed to the blockchain until the next bimonthly funding block. Shortly before the creation of the next funding block, a swarm is chosen to collect a tally of all of the votes that have been cast. This tally is then submitted into the nodes mempool and lives there until a miner reaches the funding block. This information is then used to construct the block which allocates a reward to the winning proposals. Proposals are only passed when the result of the yes votes minus the no votes is equal to 15% of the node count on the Service Node network.

9.2.3 Funds Distribution

All proceeds from the Loki Funding System are paid through funding blocks. Funding block rewards operate similarly to traditional block rewards, as an entirely non-custodial way to distribute Loki. Every 43,000 blocks (approximately 60 Days) a funding block is constructed by miners. This block contains 1.25% of the overall block reward for the entire funding block period.

To construct a valid funding block, miners must be able to assess proposals that have reached the required percentage of votes. This is done by using the information that the Service Nodes commit to the blockchain, which contains both the addresses to be paid and the state of all votes. All Service Nodes will validate the miners funding block and discard any funding blocks which pay invalid addresses.

Often the sum of Loki required by approved proposals will either exceed or fall below the total amount built up in that 60 day period. Should the total sum of approved proposals exceed that which is available in the funding block, the miner will construct the funding block prioritising proposals that were committed to the blockchain earlier. Remaining approved proposals will remain committed to the blockchain until the next Funding block.

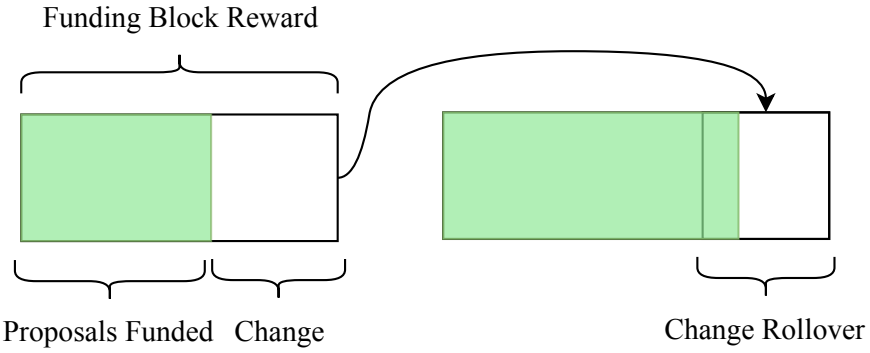

Figure 6: *Funds left unused create change which grows the reward of the next funding block*

10 Conclusion

Loki proposes a model for anonymous transactions and decentralised communication built on a network of economically incentivised nodes. Loki uses the foundations of the CryptoNote protocol to ensure privacy and implements a collateralised node system to enhance network resilience and functionality.

Additionally, Loki proposes improvements upon previous research and open source projects and presents a new anonymous routing protocol which offers significant advantages over existing protocols. The combination of a unique architecture and protocol design creates a network with market-based Sybil resistance, decreasing the efficacy of temporal analysis, and providing users with a high degree of digital privacy.

References

[1] Mike Orcutt, *Criminals Thought Bitcoin Was the Perfect Hiding Place, but They Thought Wrong* (September 11, 2017), https://www.technologyreview.com/s/608763/criminals-thought-bitcoin-was-the-perfect-hiding-place-they-thought-wrong.

[2] *Monero*, https://getmonero.org.

[3] *Tor Project*, https://www.torproject.org.

[4] *I2P Anonymous Network*, https://geti2p.net/en.

[5] *LWMA Difficulty Algorithm*, https://github.com/zawy12/difficulty-algorithms/issues/3.

[6] Daniel J. Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters, *Twisted Edwards Curves* (2008), https://eprint.iacr.org/2008/013.pdf.

[7] Nicolas van Saberhagen, *CryptoNote v 2.0* (2013), https://cryptonote.org/whitepaper.pdf.

[8] Whitfield Diffie and Martin E. Hellman, *New directions in cryptography*, IEEE Trans. Information Theory **IT-22** (1976), no. 6, 644–654. MR0437208

[9] Shen Noether, Adam Mackenzie, and Monero Core Team, *Ring Confidential Transactions* (2016), https://lab.getmonero.org/pubs/MRL-0005.pdf.

[10] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell, *Bulletproofs: Short Proofs for Confidential Transactions and More* (2017), https://eprint.iacr.org/2017/1066.pdf.

[11] Evan Duffield and Daniel Diaz, *Dash: A Privacy-Centric Crypto-Currency*, https://github.com/dashpay/dash/wiki/Whitepaper.

[12] *GitHub - loki-project/loki-network*, https://github.com/loki-project/loki-network.

[13] *Tor Project: Docs*, https://www.torproject.org/docs/faq#KeyManagement.

[14] *Possible upcoming attempts to disable the Tor network — Tor Blog.* (December 19, 2014), https://blog.torproject.org/possible-upcoming-attempts-disable-tor-network.

[15] Petar Maymounkov and David Mazières, *Kademlia: A Peer-to-peer Information System Based on the XOR Metric*, https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf.

[16] Philipp Winter, Roya Ensafi, Karsten Loesing, and Nick Feamster, *Identifying and characterizing Sybils in the Tor network* (February 25, 2016), https://arxiv.org/abs/1602.07787.

[17] *OSI model - Wikipedia*, https://en.wikipedia.org/wiki/OSI_model.

[18] Farid Farid, *No Signal: Egypt blocks the encrypted messaging app as it continues its cyber crackdown* (December 26, 2016), https://techcrunch.com/2016/12/26/1431709.

[19] Matt Burgess, *Russia's Telegram block tests Putin's ability to control the web* (April 24, 2018), http://www.wired.co.uk/article/russia-google-telegram-ban-blocks-ip-address.

[20] *Go Ethereum - Postal Services over Swarm*, https://github.com/ethersphere/go-ethereum/blob/ddfc0a2a02ce574f4c252068ce81f0f5ada1c1ff/swarm/pss/README.md.

[21] Nikita Borisov, Ian Goldberg, and Eric Brewer, *Off-the-record Communication, or, Why Not to Use PGP*, Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society, 2004, pp. 77–84, DOI 10.1145/1029179.1029200.

[22] *NaCl: Networking and Cryptography library*, https://nacl.cr.yp.to.

[23] *Pidgin-Encryption - SourceForge*, http://pidgin-encrypt.sourceforge.net.

[24] *Irreversible Transactions - Bitcoin Wiki* (March 15, 2018), https://en.bitcoin.it/wiki/Irreversible_Transactions.

[25] *Scrypt - Litecoin Wiki - Litecoin.info* (February 12, 2018), https://litecoin.info/index.php/Scrypt.

[26] *Ethash ethereum/wiki Wiki - GitHub*, https://github.com/ethereum/wiki/wiki/Ethash.

[27] *BITMAIN*, https://shop.bitmain.com/product/detail?pid=00020180314213415366s4au3Xw306A4.

[28] *Monero Cryptonight V7 - GitHub*, https://github.com/monero-project/monero/pull/3253/files/e136bc6b8a480426f7565b721ca2ccf75547af62.

[29] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich, *Argon2: the memory-hard function for password hashing and other applications* (December 26, 2015), https://password-hashing.net/argon2-specs.pdf.

[30] Dan Boneh, Henry Corrigan-Gibbs, and Stuart Schechter, *Balloon Hashing: A Memory-Hard Function Providing Provable Protection Against Sequential Attacks* (2017), https://eprint.iacr.org/2016/027.pdf.

[31] *GitHub - A Programmatic Proof-of-Work for Ethash*, https://github.com/ifdefelse/ProgPOW.

[32] *GitHub - hyc/randprog: Randomly generate a C (or javascript) program*, https://github.com/hyc/randprog.

[33] *GitHub - curie-kief/cryptonote-heavy-design: Cryptonote Heavy deign essay*, https://github.com/curie-kief/cryptonote-heavy-design.

[34] Surae Noether, Sarang Noether, and Adam Mackenzie, *A Note on Chain Reactions in Traceability in CryptoNote 2.0* (2014), https://lab.getmonero.org/pubs/MRL-0001.pdf.

[35] *Github Comment - EABE/Knacc Attack*, https://github.com/monero-project/monero/issues/1673#issuecomment-312968452.

[36] *I2P's Threat Model - I2P*, https://geti2p.net/en/docs/how/threat-model#harvesting.

[37] *Deep packet inspection - Tec Gov*, http://tec.gov.in/pdf/Studypaper/White%20paper%20on%20DPI.pdf.

[38] Philipp Winter and Stefan Lindskog, *How China Is Blocking Tor* (2012), https://arxiv.org/abs/1204.0447.

[39] *Egypt Quietly Blocks VOIP Services Skype, Whatsapp - TorGuard* (October 26, 2015), https://torguard.net/blog/egypt-quietly-blocks-voip-services-skype-whatsapp.

[40] *GitHub - Yawning/obfs4: The obfourscator (Development mirror)*, https://github.com/Yawning/obfs4.

[41] David Vorick and Luke Champine, *Sia: Simple Decentralized Storage* (2014), https://sia.tech/whitepaper.pdf.

[42] Adam Back, *Hashcash - A Denial of Service Counter-Measure* (2002), http://www.hashcash.org/papers/hashcash.pdf.

[43] Colin LeMahieu, *RaiBlocks: A Feeless Distributed Cryptocurrency Network*, https://raiblocks.net/media/RaiBlocks_Whitepaper__English.pdf.

[44] *Lazy Masternodes: do you actually have to do any work to get paid/vote?*, https://www.reddit.com/r/dashpay/comments/5t6kvc/lazy_masternodes_do_you_actually_have_to_do_any/.

[45] *ACNC template constitution for a charitable company*, https://acnc.gov.au/CMDownload.aspx?ContentKey=2efea0fa-af4f-4231-88af-5cffc11df8b7&ContentItemKey=6046cbc5-d7fd-4b6b-93ba-c8e3114b07ba.