Loki

Private transactions, decentralised communication.

Kee Jefferys, Simon Harman, Johnathan Ross, Paul McLean

Version 2
1-Mar-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 network of privacy focused services. Access to these services is limited by cryptographic keys, which represent a commitment to a precomputed proof of work. These keys can be mined or purchased using Loki, the underlying currency. Loki is built from a modified version of the Monero source code, giving all transactions 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 is developed. The white paper will be updated to reflect any future changes.

1 Introduction

Every day, governments and private corporations are slowly increasing passive and active surveillance on citizens and users. Soon we will reach a tipping point where citizens and users will start demanding their human right to privacy. Bitcoin promised privacy, but what it delivered was more traceability than ever. Companies like ChainAnalysis and BlockSeer have taken advantage of Bitcoin's transparent blockchain architecture to link specific transactions together. Loki solves this with a completely decentralised private transaction network, which rewards users to run nodes and process data.

Loki's core objective is privacy. It is built off Monero, a cryptocurrency that has established itself as one of the most secure and private transaction networks to date. Loki does not aim to compete with Monero. Rather, it utilises proven Monero privacy features as the foundation for a network of *Service Nodes* that enable a second layer of services. These Service Nodes use an architecture similar to other private internet protocols like Tor and I2P to enable private communications on the Loki network.

While Loki draws heavily on the Monero source code, we recognise that Monero has some inherent issues that impact the performance of the network. 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 with

no incentive or reward for their work. This makes running a Monero node a costly and often thankless exercise. We have made significant changes to the Monero source code to address these issues and ensure that they don't impact Loki.

Loki integrates a Service Node system similar to the masternode system used by DASH. This means that a percentage of the block reward goes to a network of nodes, economically incentivising them to operate. These Service Nodes have two key functions: they provide greater network resilience, and also act as a secondary network that can perform various functions. The first among these secondary services is Loki Messenger, which allows users to send encrypted messages across a decentralised network.

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

2 Basic statistics

Loki supply	150,000,000
Loki difficulty target (blocktime)	120 seconds
Emission speed factor	21 (10^{-21}) plus 0.5% annual compound growth
Hashing algorithm	CryptoNight
Elliptical curve	Curve25519

3 Adapting Monero source code

Loki draws on the Monero source code for a number of its privacy features. Monero is an evolution of the Cryptonote protocol, which uses ring signatures and stealth addresses to enable users to sign transactions while maintaining plausible deniability[1]. Monero has improved on the Cryptonote protocol in a number of ways, such as enabling RingCT which uses range proofs to commit to sending amounts without revealing the actual amount being sent[2].

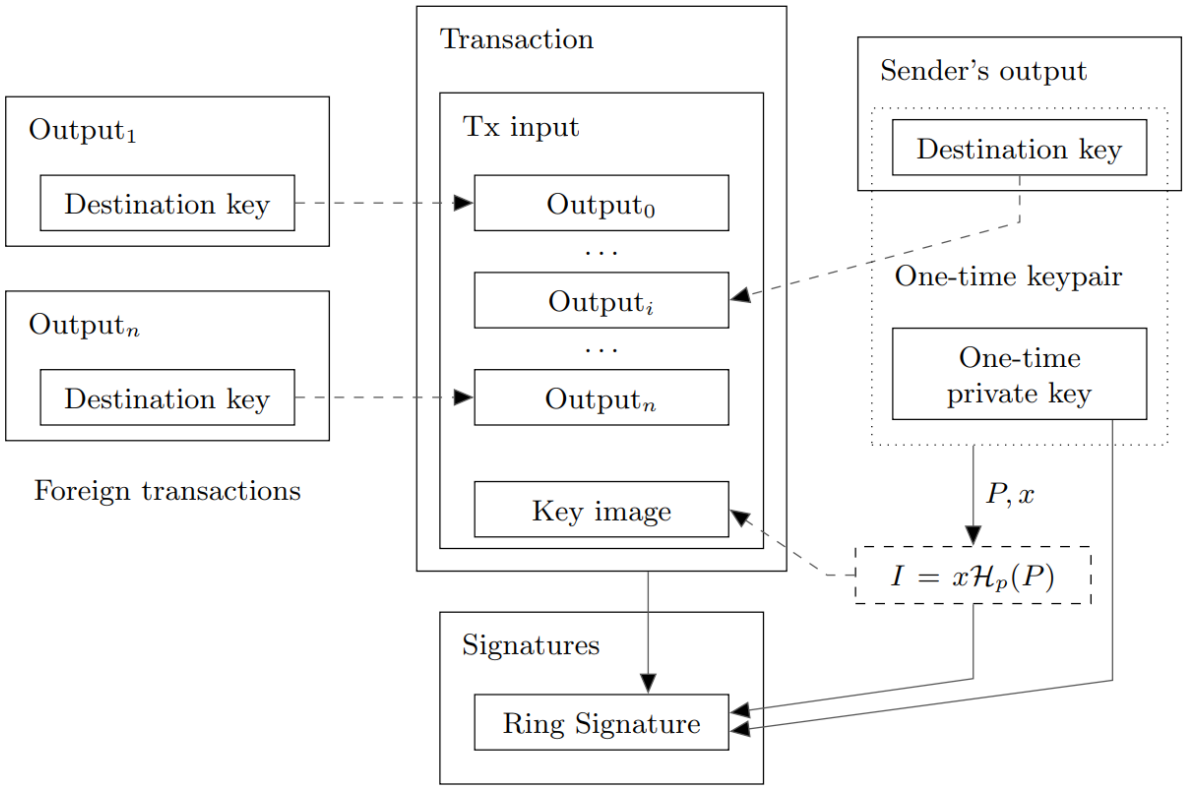
3.1 Ring signatures

Ring signatures work by constructing a 'ring' of possible signers to a transaction, where only one of the signers is the actual sender. Loki uses ring signatures in the same way that Monero does; to obfuscate the real history of transaction outputs (Figure 1). Ring signatures will be mandatory for all Loki transactions (excluding block reward transactions) with an enforced ring size of ten. This means that for any transaction there are is a upper and lower limit of ten possible signers, including the true signer, this change is discussed in detail in 4.2. Loki also uses a modified method for choosing ring signature mixins, to further obfuscate output distributions, detailed in section 4.1.

1 "CryptoNote Whitepaper." https://cryptonote.org/whitepaper.pdf.

2 "Ring Confidential Transactions - Monero Research Lab." https://lab.getmonero.org/pubs/MRL-0005.pdf.

Figure 1 Ring signature generation in a Cryptonote transaction[3]



3.2 Stealth addresses

Monero uses stealth addresses to ensure unlinkability, so that the true public key of the receiver is never linked to their transactions. Every time a Monero transaction is sent, a one-time stealth address is created and the funds are sent to this address. Using 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[4]. Stealth addresses provide unparalleled protection to receivers of transactions and are adopted as a core privacy feature in Loki.

3.3 RingCT and Range proofs

RingCT was first proposed by the Monero Research Lab as a way to obfuscate transaction amounts. RingCT currently uses range proofs which leverage Pedersen commitments to prove that an amount of Monero being sent is between 0 and 2^{64} [5]. This range ensures that only non-negative amounts of Monero are sent, without revealing the actual amount of Monero sent in the transaction.

3 "CryptoNote Whitepaper." https://cryptonote.org/whitepaper.pdf.
4 "New Directions in Cryptography - Stanford University." https://www-ee.stanford.edu/~hellman/publications/24.pdf.
5 "MRL-0005: Ring Signature Confidential ... - Monero Research Lab." https://lab.getmonero.org/pubs/MRL-0005.pdf.

We propose a new usage of range proofs, specifically *Bulletproofs* (a more compact range proof method[6]), to prove that a Service Node holds a predetermined amount of Loki. We call this method a Bulletproof declaration; it is explained in section 4.5 of this paper.

3.4 Kovri

Kovri is a C++ implementation of an I2P router that is currently in development by the Monero open source project[7]. I2P is a peer-to-peer protocol that forms the basis for a decentralised network that is able to route traffic through the internet without revealing the true IP addresses of connections. Kovri is still in development and has not yet been released.

Loki intends to implement Kovri through its own nodes to securely route traffic, removing the possibility of a connection between IP addresses and transactions. The use of Kovri will allow greater privacy when transacting and using Service Nodes.

3.5 ASIC resistant hashing algorithm

An *Application-Specific Integrated Circuit* (ASIC) is a computer chip that is built specifically for a single purpose. In the context of mining, ASICs are used to solve specific hashing algorithms. They pose a risk to decentralisation because they outpace all other mining methods, are manufactured by specific companies, and require significant capital cost to develop and operate. ASIC miners tend to be used by large conglomerates who can pool their hardware in countries with cheap or free electricity, geographically and financially centralising a cryptocurrency's hashing power.

To mitigate this risk, Monero and Loki both run the CryptoNight hashing algorithm which requires large amounts of L3 cache to run, which makes it difficult to develop a CryptoNight ASIC. If an ASIC was to be developed for the CryptoNight hashing algorithm, in principal Loki would hard fork, pending community consensus, and make changes to the CryptoNight algorithm to invalidate ASIC chips. Mining that does not occur on ASICs is likely to happen on GPUs, which have a lower capital requirement, and are more difficult to run at scale. CPU and GPU mining is more likely to be undertaken by a distributed group of individuals rather than larger conglomerates. This will increase the chance that the network remains decentralised.

3.6 Smooth emissions curve

Many coin reward structures have 'Halvenings.' In Bitcoin, they occur every four years, meaning the Bitcoin block reward halves from what it was previously[8]. When a halvening occurs, the network often sees a short, temporary drop in hashing power. This leaves the network vulnerable to hostile take-over by a party that could bring a large amount of hashing power online and perform a 51% attack.

6 "Bulletproofs: Efficient Range Proofs for Confidential Transactions." https://eprint.iacr.org/2017/1066.pdf.
7 "GitHub - monero-project/kovri: The Kovri I2P Router Project." https://github.com/monero-project/kovri.
8 "Bitcoin: A Peer-to-Peer Electronic Cash System - Bitcoin.org." https://bitcoin.org/bitcoin.pdf.

Loki prevents this by adopting the Cryptonote model, where a smooth emissions curve is followed. This means that the block reward decreases slightly every block, instead of abrupt reductions that could cause hash rates to fluctuate.

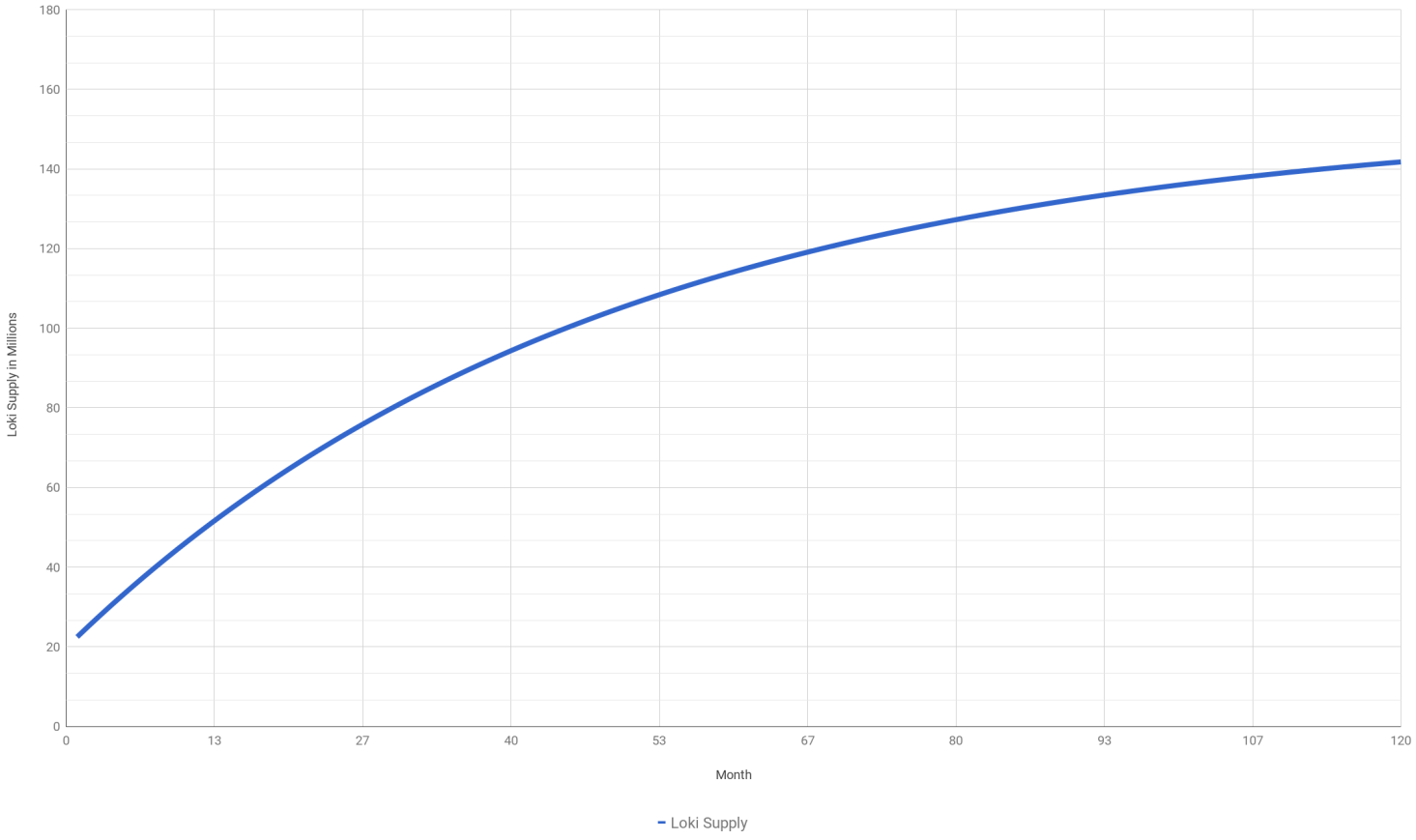
The Loki emission curve is determined by the equation below, where M is the total supply expressed in atomic units, A is the circulating supply expressed in atomic units, λ is the emission speed factor.

$$(M-A) \times \lambda + 0.00000002A$$

Where:

$M = 1.5 * 10^8$
$A = Circulating\ supply$
$\lambda = 2^{-20}$

Figure 2 The Loki emissions curve



3.7 Tail emission and inflation

There has been much debate into whether a deflationary or inflationary model makes more sense for a cryptocurrency that is targeted towards being used as a currency rather than a store of value. We envision Loki as a currency that allows a private method of transacting value and purchasing access to second layer services. It is generally accepted that deflationary currencies encourage saving and disincentivize spending, which is negative for the growth of economies. Reserve banks in most successful economies target annual inflation rates at 1-2% to stimulate spending and increase the velocity of money.

Taking this into account, Loki's tail emission differs from Monero's in that it offers an inflationary rate of 0.5% per year, instead of having a fixed amount emitted per new block found. In practice a steady reward every block does increase total supply over time, but the proportion of the total supply growth lowers every year the fixed block reward continues. This creates a situation where Monero is inflationary, however decreasingly so over time. With compound inflation, the Loki total supply will maintain proportional growth compared to it's supply, which is important to maintain a growing Service Node network described in section 4.4.

3.8 Dynamic block size

Unlike other cryptocurrencies that have a fixed block size, Monero's block size changes over time, growing to include more transactions as the network reaches higher volumes. The Monero block size algorithm scales by observing the median block size over the last 100 blocks and slowly retargets the maximum size of any new blocks accordingly. Loki plans to use the same block size algorithm to scale the blockchain as the network grows. Large block sizes would typically be considered to be a burden on the nodes that store and verify transactions. However, because Loki incentivises nodes to process transactions, the large block size should have very little impact on the performance of the network.

4 Unique Loki features

4.1 Mixin distribution

Monero's ring signatures are usually composed of one true unspent output (the sender's) and a number of decoy unspent outputs which are referred to as *mixins*. Mixins masquerade as unspent outputs even though they may not be. The advantage of a ring signature is that all of the unspent outputs should have an equiprobable chance of being the true unspent output spent in the transaction. However, in practice Monero and other Cryptonote based coins have had historic difficulties choosing outputs correctly. This is primarily because the older a mixin is, the more likely it is to already be spent; thus making it less likely to be the true unspent output.

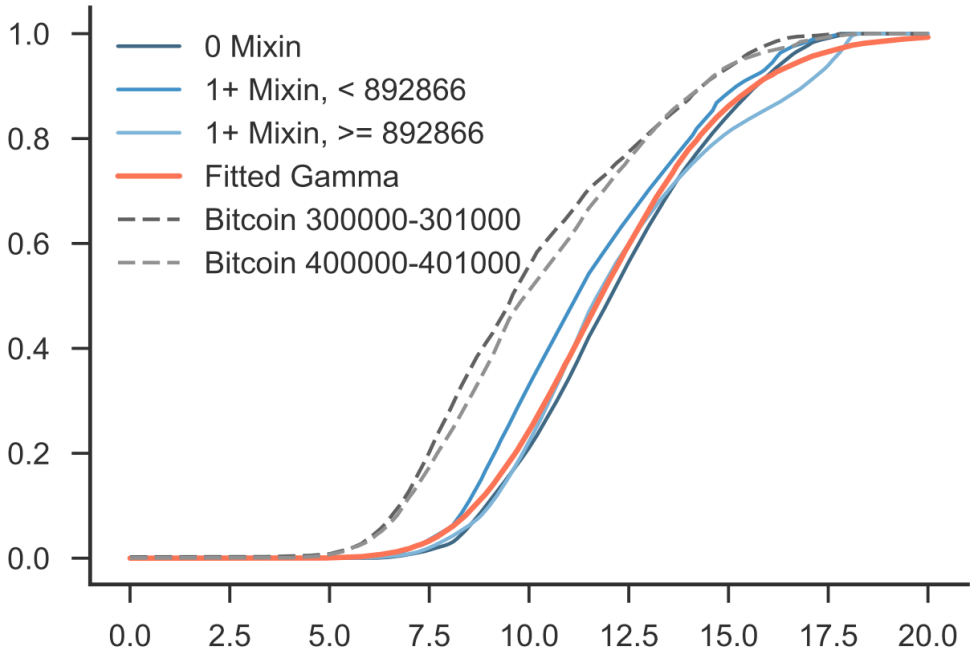
Monerolink published a paper in early 2017 exploring true unspent outputs in ring signatures[9]. Monerolink found that, on average, in 80% of ring signatures the true unspent output was the output with the highest block height (i.e. the newest block) . The Monero Research Lab and others have done extensive research on the best method to choose mixins appropriately[10].
Monero allows users to choose the mixins to be used in each transaction. This is usually done by the wallet software (as per the code below) which specifies that 50% of unspent outputs must be chosen from the 'recent' zone (transactions from the last 1.8 days) and 50% of unspent outputs must be chosen using a triangular distribution from the remaining pool of available unspent outputs. While this method favours using 'newer' unspent outputs in rings, it still uses constant values instead of sampling an actual distribution which would more accurately match the real spending habits of Monero users.

```
uint64_t i;
          if (num_found - 1 < recent_outputs_count) // -1 to account for the real one
we seeded with
          {
            // equiprobable distribution over the recent outs
            uint64_t r = crypto::rand<uint64_t>() % ((uint64_t)1 << 53);
            double frac = std::sqrt((double)r / ((uint64_t)1 << 53));
            i = (uint64_t)(frac*num_recent_outs) + num_outs - num_recent_outs;
            // just in case rounding up to 1 occurs after calc
            if (i == num_outs)
              --i;
            LOG_PRINT_L2("picking " << i << " as recent");
          }
          else
          {
            // triangular distribution over [a,b) with a=0, mode c=b=up_index_limit
            uint64_t r = crypto::rand<uint64_t>() % ((uint64_t)1 << 53);
            double frac = std::sqrt((double)r / ((uint64_t)1 << 53));
            i = (uint64_t)(frac*num_outs);
            // just in case rounding up to 1 occurs after calc
            if (i == num_outs)
              --i;
            LOG_PRINT_L2("picking " << i << " as triangular");
```

The Monerolink paper posed a solution to this problem which will be implemented in Loki. By sampling a spending distribution in both Monero and Bitcoin, the paper found that spending habits of users are highly predictable (Figure 3).

9 "An Empirical Analysis of Linkability in the Monero... - MoneroLink." https://monerolink.com/monerolink.pdf.
10 "MRL-0004: Improving Obfuscation in the ... - Monero Research Lab." https://lab.getmonero.org/pubs/MRL-0004.pdf.

Figure 3 Spend-time distributions in Bitcoin and Monero over multiple distributions[11]



Using this data along with the pseudocode provided in the Monerolink paper (referenced below), Loki will sample mixins based on how users spend.This means that a third party analysing the outputs in a ring signature cannot assume that the oldest 'unspent' output is a decoy output, because the distribution at which the outputs were chosen reflects the appropriate chance that an 'older', unspent output would be included in the ring signature. This new sampling method will increase Loki's effectiveness against a temporal association attack.

```
SAMPLEMIXINS(RealOut, NumMixins)
Let TopGIdx be the index of the most recent
transaction output with denomination
RealOut.amount;
BaseReqMixCount := b(NumMixins+1)×1.5+1c;
Let RecentGIdx be the index of the most recent
transaction output prior to 5 days ago with
denomination RealOut.amount prior to 5 days ago;
BaseReqRecentCount := MAX(1, MIN(
TopGIdx - RecentGIdx + 1,
BaseReqMixCount × RecentRatio));
if RealOut.idx ≥ RecentGIdx then
BaseReqRecentCount -= 1
MixinVector := [];
while |MixinVector| < BaseReqRecentCount do
i ← UniformSelect(RecentGIdx, TopGIdx);
if i ∈/ MixinVector and i 6= RealOut.idx then
MixinVector.append(i);
while |MixinVector| < BaseReqMixCount do
i ← TriangleSelect(0, TopGIdx);
if i ∈/ MixinVector and i 6= RealOut.idx then
MixinVector.append(i);
Let FinalVector be a uniform random choice of
NumMixins elements from MixinVector;
return sorted(FinalVector+[RealOut.idx]);
```

11 "An Empirical Analysis of Linkability in the Monero ... - MoneroLink." https://monerolink.com/monerolink.pdf.

4.2 Ring signature size

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 five, meaning that four mixins are used alongside the real unspent output in a transaction.

In January 2016, Monero introduced a minimum ring size of three. This was done to defend the network against the negative impacts of zero mixin transactions, that is rings where the only output signed in the ring is the true output. In September 2017 the Monero development team again raised the ring size to a minimum of five. This was part of a gradual move to slowly increase the ring size once the effects of a ring size of three was observed.

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 amount of outputs on the blockchain[12]. It was found that higher ring sizes reduce the timeframe in which a malicious attacker who owned a large amount 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. An EABE attack is when an active party (usually an exchange) can sit between two sides of a transaction and can link transactions together. If we consider the transaction flow below:

$$Exchange \rightarrow Alice \rightarrow Bob \rightarrow Exchange$$

In this send flow, an exchange that follows KYC/AML regulation sends Monero to Alice, creating a record of Alice’s true identity and her public key. Alice then sends some Monero to Bob. Bob then deposits that Monero back onto an exchange.

Looking at this scenario from an exchange’s point of view reveals some interesting details. When the exchange sends an unspent output to Alice, they know that Alice now owns that output. When Alice sends Bob some Monero she will send her unspent output mixed with 4 other ‘unspent outputs to Bob. If the exchange is monitoring the blockchain passively they can see that Alice ‘used’ her output as one of the mixins for a ring signature. If the exchange now receives money from Bob in the form of a deposit they can know the true unspent output in his ring signature as well.

This situation by itself is not actually an issue because outputs sitting in anyone's wallet will be spontaneously used in other ring signatures often. Any adversary who sends an output and watches for its appearance on the blockchain cannot know whether that output was truly spent or not. However if a third party can interact with the receiver of the real transaction, they can see that Alice's output is always included in a ring signature sending to another person, they can also see the same output is used by Bob when he is depositing. If Alice makes regular transfers to Bob a government or third party could use this to establish probable cause to conduct an investigation.

12 "MRL-0001: A Note on Chain Reactions in Traceability in CryptoNote" https://lab.getmonero.org/pubs/MRL-0001.pdf

The Monero Research Lab has done some limited research into this this problem and recommends churning. That is sending your full balance to yourself multiple times.

$$Exchange \rightarrow Alice \rightarrow Alice \rightarrow Alice \rightarrow Alice \rightarrow Bob \rightarrow Exchange$$

The flow above represents three distinct churns, where Alice sends her own balance back to herself. Churning is effective because each time a user churns they increase the amount of times their output has potentially been spent, hiding their real output in a number of ring signatures.

The anonymity set of a churn A is represented in the below equation

$$A = \mathcal{R}^{c}$$

Where:

$\mathcal{R}$ = Ring size
c = Times churned

With three churns, it is extraordinarily hard for a third party to follow possibilities where Alice did engage with Bob since.

$$A = 5^3$$

Monero has no maximum ring size enforced by network consensus rules. Many wallets like the Monero Official GUI wallet 'cap' the ring size at 26, however a user is free to manually create a transaction with whatever ring size they wish, aslong as it is above a ring size five. Practically speaking this is problematic since most wallets have a default ring size of five, increasing your ring size above five makes your transactions standout (Figure 4). Further if you were always to use a non standard ring size in Monero, such as seven, a passive third party could analyse the blockchain and start to infer patterns using temporal analysis.

Figure 4 showing how non standard ring sizes stand out

tx hash	fees	ring size	tx size [kB]
44be6f04715100dbb88ab5121be60e531d15bc71db3e15003fd924d73d38439e	0.011	5	12.79
40cdeaaabcaf26a50e66a85614b2d83b3bdbdc7c9e9172759c3bcf5959acbf34	0.011	5	12.79
e69942c010d1e67e8254553248993004d40a958b66954cf7613234be84db76b6	0.011	5	12.79
967f754d5b3b6b69c67074cb714bf732d1561194943b1804d4bbf28afab204a7	0.013	5	14.90
ff768c6e4a96e709a036523d856171b25c3cd070d93929c1ed964acc1dd32dce	0.003	10	13.09

Loki improves on both of these problems by statically enforcing ring sizes, and setting the ring size minimum 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 would more effectively prevent an attacker who owns a large amount of outputs from discerning the true outputs spent in a ring signature. Large ring sizes also increase the default churning effectiveness non linearly becoming more effective as ring sizes grow, shown below.

$$A = 10^3$$

In the current form of a Monero transaction doubling the ring size to 10 would lead to a 4.3% increase in the size of the transaction. However when bulletproofs are implemented it will account for about a 10 -15% increase in the size of a transaction. This is because of the overall reduction in transaction size cased 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 relatively low bandwidth and hardware requirements for nodes. However because Loki incentivizes nodes this burden can be carried by the Service Nodes.

4.3 Block reward

Most proof of work cryptocurrencies like Bitcoin and Litecoin award 100% of the block reward to miners. This model rewards miners for performing proof of work, which provides network security but is computationally expensive. However, it fails to reward the valuable work that full nodes do in relaying transactions and enforcing the network's consensus rules. When 100% of the block reward is awarded to miners, it has the side effect of giving miners a disproportionate amount of power to control the system. Not only do miners perform all of the proof of work operations required to create blocks, they are also one of the only groups that has an incentive to run full nodes (to enable their mining activity), giving them power to set the consensus. In Bitcoin and other cryptocurrencies that use this model, situations can occur where miners will push for a fork of the blockchain to follow the most profitable software change, even if that is not in the best interest of the broader community of users.

In order to reward full-nodes for the consensus enforcement and network resilience they provide, Loki will enable a system similar to the Dash implementation of masternodes. In Loki, they are called *Service Nodes*, because of the additional networking services they can perform.

In a hybrid proof of work/proof of service system, hash rates are generally lower than full proof of work systems due to the split block reward. In Dash, the block reward split does not change, always awarding 45% of the block reward to both Service Nodes and miners. However, if profitability does drop, miners tend to leave the system, and the difficulty can often adjust slower than price movements. In these periods of low hash rates, the entire network can be vulnerable to 51% attacks.

We address this through a *block reward equilibrium algorithm* (see below), this will strike a balance between miners and Service Nodes by adjusting how the block reward is allocated at regular intervals. When hashing power and difficulty is low on the network, miners will receive a higher proportion of the block reward than usual. Conversely, when the number of Service Nodes actively communicating with the network decreases below an expected amount, Service Nodes will be granted a higher portion of the block reward. This model should encourage miners to respond faster to drops in hash rate, even during periods of price volatility.

Loki also features a governance reward which is to be spent by the Loki Project Team and feed into the Loki Funding System, ensuring that the ecosystem can remain self-funded and outside of the influence of third parties (see section 7).

```
blockRewardEquilibrium

int mineChange = prev100Dif - curr100Dif // Measure change in difficulty
int stakeChange = prevActiveStake - currActiveStake // Measure change in active stak-
ers

if (mineChange < 0 && stakechange <0){
       return // do nothing if both rates fall
}
if(mineChange < 0){
       mineBlockReward(0.01);  //Increase Mining reward by 1% if difficulty decreases
return
}
if(stakeChange < 0){
       stakeBlockReward(0.01) //Increase Staking reward by 1% if Active nodes has fall-
en in the last 100 blocks
       return
}
```

As a target for the equilibrium algorithm, miners will, on average, receive 50% and Service Nodes will receive 45% of the block reward. However, regardless of the split as determined by the algorithm, there will always be a 5% stake of the block reward that will be awarded to the governance pool.
Mining reward: On top of the block reward split, transaction fees from the network always go to the miner who constructs the valid block.

Service Node reward: The miner of each block selects a single Service Node to be rewarded based on its position in the index of Service Nodes, in accordance with the deterministic ordering code as outlined in Dash[13]. The winning Service Node must be live and staking at the time of the reward. In order to check if a node is live, it will be a requirement for nodes to keep 'lists' of other active nodes. This will be done using the Chord protocol which uses a *Distributed Hashing Table* (DHT)[14]. These DHT's will include the public Loki address associated with that node as well as the IP address (or I2P address) that it is broadcasting on. The mining node should pick three other nodes at random to verify that the candidate Service Node is active and staking. If the three other nodes do not return a response from the winning node, the next random node in the DHT will be chosen. The process of checking will continue until a valid candidate is found for the Service Node reward. This incentivises Service Nodes to always be online.

Governance reward: The governance block reward will go to a fixed address held by the Loki Foundation. In the header of every block, miners must include the transaction ID, and transaction key of the transaction that rewards the governance pool. With this information, nodes and third parties can verify this candidate block pays the governance address. Additionally, the governance address will have its view key published publicly so that third parties can audit incoming flows.

13 "Whitepaper · dashpay/dash Wiki · GitHub." https://github.com/dashpay/dash/wiki/Whitepaper.
14 "Chord: A Scalable Lookup Service" https://pdos.csail.mit.edu/papers/chord:sigcomm01/chord_sigcomm.pdf.

4.4 Operating a Service Node (staking)

In order to run a Service Node, a node must store a dynamically selected amount of Loki based on the block height. This differs from other cryptocurrencies that have a masternode systems. For example, Dash requires nodes to store a fixed amount of Dash (1000 Dash) in order to run a masternode. However, this is economically challenging, at the time of writing, 1000 Dash costs around $800,000 USD. The sheer cost associated with running a Dash masternode prohibits the creation of a large network of masternodes.

To avoid this outcome, Loki proposes to introduce a dynamically adjusting Service Node collateral requirement. The minimum amount of Loki required to run a node will decrease over time, ensuring that as time passes and adoption increases, the financial barrier to running a Service Node is reduced, facilitating the creation of a large network of Service Nodes. The benefit of a large Service Node network is greater network security, speed, anonymity, and a reduced likelihood of network centralisation. However, if the collateral requirement is set too low in the beginning, an attacker could set up a large number of Service Nodes and damage the integrity of the network. For example, if there are only 5,000 Service Nodes on the network and the collateral requirement is set at 1,000, the attacker would only need to control about 1.1% of the circulating supply to control enough Service Nodes to undermine the trustless quorum (discussed in section 5).

The equation to determine the amount of Loki that a node is required to stake in order to become a Service Node takes into account the circulating supply of Loki at the current block height. This amount is based on the theoretical limit of Service Nodes that can be operating at any one time. There is a hard limit based on supply and a soft limit based on price. For example, if 10 million Loki were to be in circulation with a theoretical staking cost of 10,000, a maximum of 1,000 Service Nodes could be running on the network. However, if the supply increases and the staking requirement to create a Service Node is reduced, the Service Node cap will increase significantly. At a supply of 70 million and a theoretical staking cost of 1,900 Loki, the maximum Service Node count would be around 36,000. The soft limit is imposed by the price it costs to acquire the correct amount of Loki on exchanges to create new Service Nodes. Exchanges are likely to have limited liquidity due to the usage of Loki in existing Service Nodes.

Max theoretical nodes on the network

$$E = A / S$$

Where:

E = Theoretical max Service Nodes on the network.
A = Circulating supply
S = Staking requirement

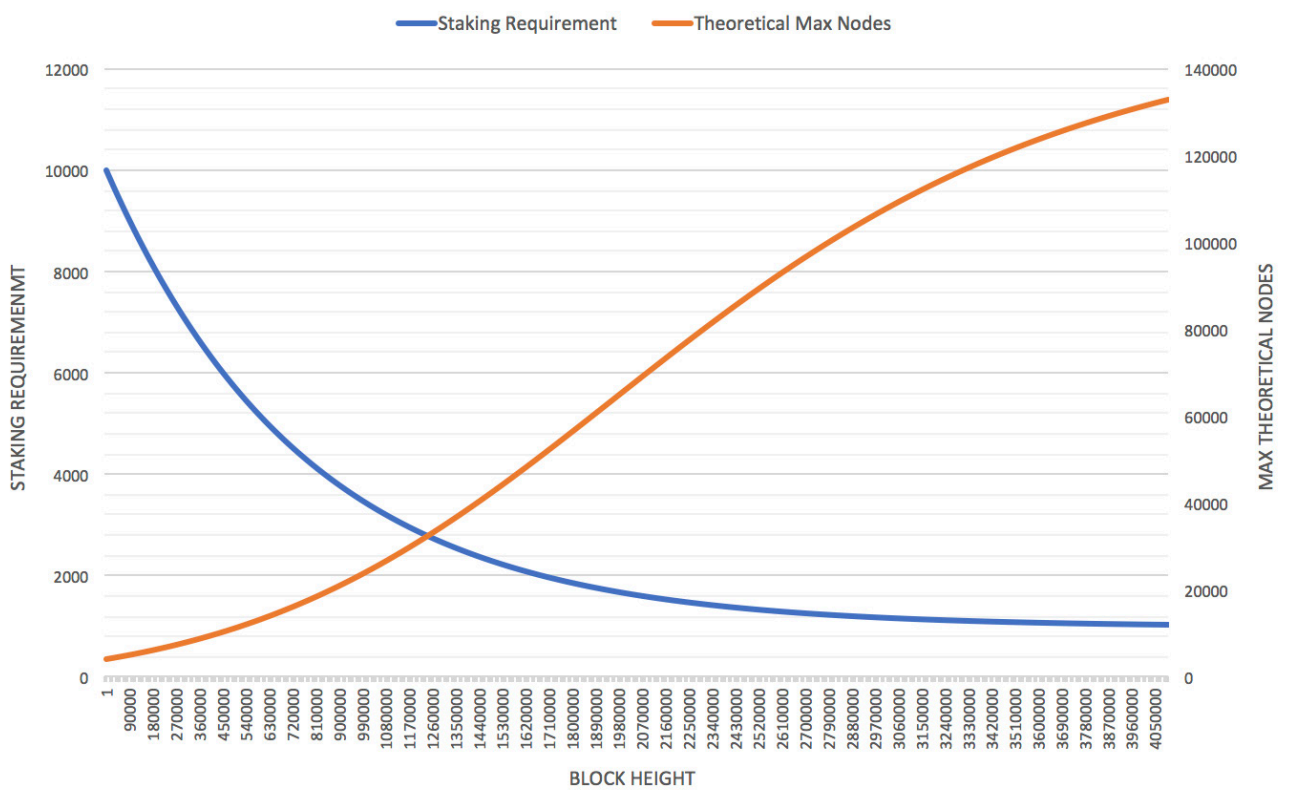
Staking requirement

$$S = 1000+9000e^{-1.3*10\wedge-6\ BH}$$

Where:

S = Staking requirement
BH = Block height

Figure 4 Relationship between the Service Node staking requirement and the maximum theoretical Service Node limit



4.5 Bulletproof declaration

Loki uses a modification of Monero's existing range proofs method to verify that Service Nodes are holding the required amount of Loki. Range proofs are currently used in Monero to confirm that when a user sends a transaction, the amount they are sending is between 0 and 2^{64}. Range proofs prove an amount is non-negative without revealing the amount that is being sent.
We propose to use range proofs (specifically Bulletproofs) to verify that a Service Node in the Loki network has the required amount of Loki to operate a Service Node, without revealing the amount of Loki that they actually hold.

To do this, we first construct a Pedersen Commitment C committing to an amount a using a mask (blinding factor) represented by x:

$$C=x\times G+a\times H$$

Where G and H are protocol-defined independent base points.
To prove that the committed amount falls between a specific range, such as 1,000 and 1,000,000 Loki, we could define two more commitments:

$$C_1=C-1000\times H$$

$$C_2=1000000\times H-C$$

and generate a new range proof for each of C_1 and C_2

This approach could be generalised to involve multiple outputs and prove that the sum of the committed amounts fall within a specified range.

The range of the Bulletproof will be adjusted downwards every five blocks. This will prevent the rebroadcast of 'valid' but old Bulletproofs and also lower the requirements to stake as per the staking equation. All Service Nodes in the Loki network will be required to request and verify the Bulletproofs of five other random nodes in the Service Node network every 15 minutes. If a Service Node's Bulletproof is found to be invalid then they will be flagged. If a flagged Service Node produces an invalid Bulletproof, or fails to provide any response multiple times within a 24-hour window, a message will be broadcast to the network to remove this node from the Service Node DHT.

5 Service Node Applications (SNApps)

Services Nodes are incentivised to be honest participants in the network as a result of the significant collateral requirement. Game theory suggests that no player will intentionally sabotage a system that they directly benefit from. This means we can create a trustless quorum from groups of Service Nodes picked randomly from the DHT index. We can then query the Service Nodes to come to consensus about the state of the blockchain, or build *Service Node Applications (SNApps)* on top of these trustless quorums and use their services to reliably store and serve private messages.

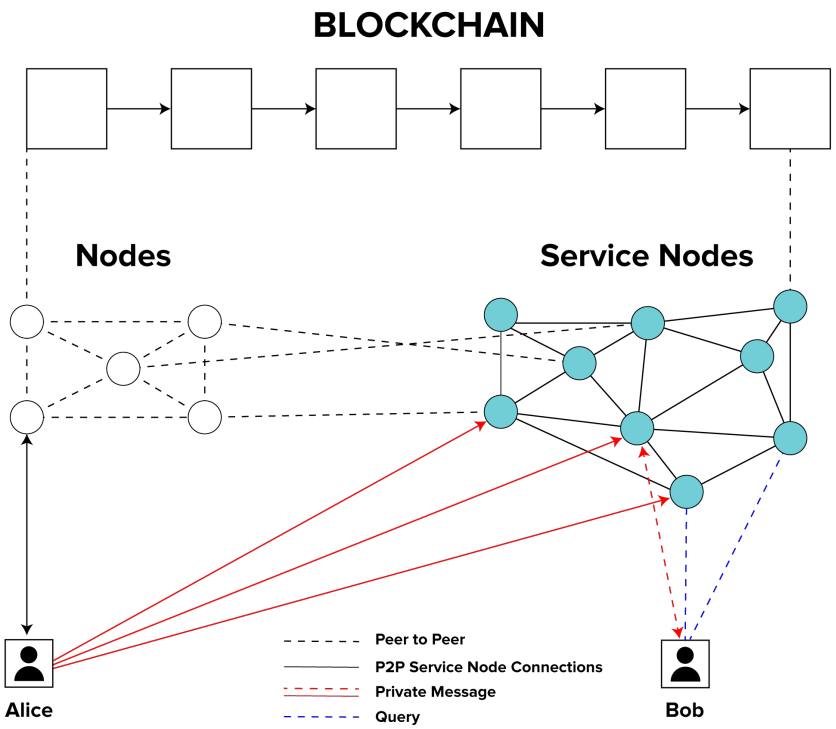
5.1 Loki Messenger

The first SNApp to be developed on the Loki network will be a decentralised, end-to-end encrypted private messaging service called Loki Messenger. End-to-end messaging applications that provide a platform for users to send messages without revealing their contents already exist, however they rely on centralised servers that governments or third parties can target. By leveraging the power of public-private key cryptography and the Service Node architecture on the Loki network, we can create a similar service to the popular encrypted messaging app *Signal*, with the added benefit of decentralisation.

The Loki messaging system uses public-private key cryptography, where the receiving address is a Loki holder's public key. The sender will broadcast their message, signed with the receiver's public address, to three randomly selected Service Nodes. Each Service Node will have their own 'message pool', and will hold messages for up to 24 hours. The receiver will send a query signed with their private key to the Service Node network asking if they have any messages that are destined for their public key. If a Service Node finds a message that matches the public key of the receiver, then that message is released (spam protection for this network is discussed in section 6).

Users will be able to send messages over the Loki network using a mobile app. Messaging functionality will also be available in the Loki wallet software, which users can seamlessly switch to if required, so that in an event where the mobile front end is restricted, there will always be an alternative access to the messenger. (Figure 5) illustrates how messages will be sent and received on the Loki network while regular network functions also take place.

Figure 5 Loki's decentralised, peer-to-peer private messaging system



To fully shut down the Loki messaging system, a third party would need to own a majority of the Loki Service Nodes. However, this tends toward financial infeasability because of the collateral required to run a Service Node.

5.2 Loki Messenger encryption

Loki Messenger uses two encryption features that put it on the cutting edge of private messaging services: *Perfect forward secrecy* and *Deniable authentication.*

Perfect forward secrecy is a feature that enables the Loki messaging system resistance from attacks where a long-term key is exposed. A new shared encryption key is used for each message, so if one key is revealed, the whole message chain does not become compromised. If a third party wanted to break the encryption of a message chain, they would need to obtain the keys for every individual message. Perfect forward secrecy ensures that Loki Messenger is extremely difficult to compromise, as opposed 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.

The second quality is deniable authentication. Which 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.

Both of these qualities become useful in different situations. An example might be a whistleblower attempting to safely leak some sensitive text. If this whistleblower uses PGP encryption to leak text to a journalist, and either the journalist or the whistleblower are targeted and their private keys are stolen, the whole message chain is compromised. This is undesirable and why perfect forward secrecy is needed.

In the same scenario, imagine the journalist's keys are stolen and the whistleblower's messages to the journalist are unencrypted. In this case, authorities know the information that was leaked and they also can prove that the leaked documents came from the whistleblower, because they sign all messages with their private key. Once more, this is an undesirable quality for a encryption scheme. When using deniable authentication, after each session *Message Authentication Codes* (MACs) are published allowing third parties to plausibly create messages that appear as if they originate from the senders public address. When this is correctly implemented, it is impossible to prove that a sender of a specific message was the actual sender by any third party.

Perfect Forward Secrecy and Deniable authentication are key concepts in the Off the Record messaging protocol[15]. Centralised services like Signal and WhatsApp use encryption features similar to this but have improved upon offline messaging by using Axolotl Ratchets. To incorporate these two features, Loki will implement a modified version of the open source Signal protocol which maintains both perfect forward secrecy and deniable authentication [16].

15 Off-the-Record Communication, or, Why Not To Use PGP." 28 Oct. 2004, https://otr.cypherpunks.ca/otr-wpes.pdf.

16 WhisperSystems/libsignal-protocol-c: Signal C Library." https://github.com/WhisperSystems/libsignal-protocol-c.

6 Attack prevention

By nature, private decentralised blockchains remove the requirement for users to provide digital or physical identifiers to use the system, which can be beneficial to people who lack identity or are being persecuted because of it. However, systems that lack identity 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. Blockchains typically use fee models to dissuade Sybil attacks, the significant investment in resources required by these models means that attacking the network is cost prohibitive.

It can become more difficult to prevent attacks when a second layer is built on top of a blockchain with relative seperation between the layers. 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 Siacoin's case, the cost is determined per TB of storage per month[17]. 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 Raiblocks require users to calculate a small proof of work before sending a message or transaction[18] [19]. These small proof of work systems are arguably more egalitarian than the fee-for-service model, but they fall prey to attackers who possess large amounts of computing power and are not available to mobile users or users who do not have access to powerful GPUs / ASICS.

Loki proposes a hybrid system to prevent Sybil attacks, called *Runes*. Runes are a modified version of *Medallions* which were first described in the OrchidProtocol whitepaper[20].

6.1 Runes

Runes are necessary for users to access Service Node applications. Initially, Runes will only enable access to Loki Messenger, but over time will expand to include all SNApps as more are deployed.

Runes are used to prevent Sybil attacks on the Service Node network. Requiring each Rune to be bound to a public address provides nodes with a way to confirm 'identity' while still maintaining privacy. A limit can be set on the number of requests that a public address can make to the network in a specified timeframe. The relatively high computing power required to mine a Rune means that attacking the network is time-consuming and expensive.

17 "Sia: Simple Decentralized Storage - Sia.tech." 29 Nov. 2014, https://sia.tech/whitepaper.pdf.
18 "Hashcash - A Denial of Service Counter-Measure." http://www.hashcash.org/papers/hashcash.pdf.
19 "RaiBlocks: A Feeless Distributed Cryptocurrency." https://raiblocks.net/media/RaiBlocks_Whitepaper__English.pdf.
20 "White-Paper - Orchid Protocol." 8 Dec. 2017, https://orchidprotocol.com/whitepaper.pdf.

Each Rune in the Loki system represents an arbitrary amount of work done by a computer. A Rune is created and given as a reward to a Rune miner. When first mined, a Rune is considered 'unbound', meaning that the miner can either bind it to their own public key or sell the Rune to be 'bound' by another user. Once a Rune is bound to a public key it cannot be transferred. Runes are valid for 20,000 blocks (approximately 30 days) from when they are mined. Runes are designed to be egalitarian, as users who cannot afford to buy Runes can access them by mining. Users who are on mobile platforms or who do not wish to mine for Runes can pay for them with Loki in an open market.

The ownership and binding of Runes occur on the *Runechain*, a secondary blockchain in the Loki network that is maintained specifically for Runes. The Runechain is relatively lightweight as it can be progressively pruned to the last 20,000 blocks (the expiry date of all newly minted Runes). The Runechain only has to be maintained by the Service Node network because of its relative separation from the Loki base layer. A decentralised exchange will be implemented in the GUI wallet to allow users to trade Loki for Runes, much like the Counterparty IndieSquare wallet[21]. This secondary market will incentivise Rune miners to mine for users who cannot meet proof of work requirements on their own.

Initially to harbor ease of implementation, Runes will be mined using the CryptoNight hashing algorithm which is ASIC resistant and provides a level playing field for most users. Eventually, Loki plans to implement a proof of space system as outlined in the Spacemint whitepaper[22]. Proof of space will provide an even more egalitarian way to 'mine' that provides accessibility above and beyond the requirement for computational power alone.

6.2 Confirming SNApp work

To remain in the staking pool, Service Nodes must actively communicate their Bulletproofs and network status to surrounding nodes. However, this does not confirm that they are actively processing or routing traffic. A dishonest Service Node could communicate messages back to its surrounding nodes that it is active and staking, but never process or route network traffic.

To solve this problem, Loki proposes a system where every packet of data sent across the network of Service Nodes is structurally indistinguishable. Special test messages called *dummy routes* are mixed in with these indistinguishable packets and are used to test whether a targeted node is performing its duty. Dummy routes use the targeted node as a relay between two Service Nodes. The first Service Node sends the dummy route to the targeted node and expects the targeted node to route the message to the second Service Node. If they do not receive the message response from the second node they flag the targeted node and move on. The flagged Service Node will be tested three more times at random intervals. If in each instance a response is not received from the second Service Node, then the targeted node is removed from the staking pool.

21 "GitHub - IndieSquare/: Mobile counterparty wallet.." 14 Jun. 2017, https://github.com/IndieSquare/indiesquare-wallet.
22 "SpaceMint - Cryptology ePrint Archive." https://eprint.iacr.org/2015/528.pdf.

6.3 Provable flagging

Flagging is one of the mechanisms to protect the network against poor node behaviour. However it also creates an avenue to attack the network. For example, malicious nodes could collude to send out thousands of flags to legitimate nodes, thus removing them from the staking pool and increasing the chance that the malicious nodes would win a staking reward. To prevent this scenario, we migrate flagging to the Runechain and limit service nodes to a reasonable rate of one flag per 60 blocks. Because removal from the staking pool requires absolute consensus in terms of the rounds conducted in flagging, the legitimate part of the network will always be able to protect against non-legitimate flagging of active Service Nodes, assuming at least half of the network is honest.

7 Governance

Loki has an inbuilt self-funding system as a function of the governance block reward split (outlined in section 4.3). A designated party must control the private keys for the receiving address that contains the pool of governance funds. As a result, there must be a mechanism to ensure that the funds acquired through the governance block reward are spent on the advancement of Loki in an efficient and effective manner.

We propose to establish an entity called the *Loki Autonomous Government* (LAG) which exists to fund the continued development of the Loki project, community and ecosystem. The LAG will be comprised of three distinct components: the Loki Foundation, the Loki Project Team and the Loki Funding System. Each of these are described in turn.

The Loki Foundation – The Foundation is a not-for-profit organisation that will have authority over the governance block reward and are responsible for the signing of transactions. It will be governed by a board of people whose principles are aligned with Loki's vision and goals. The board will assess the spending of the Project Team and operate the Funding System. Decisions will be made by consensus vote. The board will have no say in the outcome of Funding System proposals once they are submitted to the network of Service Nodes for voting.

The Loki Project Team – The Project Team is the executive branch of the ecosystem. The team is tasked with ensuring the ongoing development of the project, operating the bounty system for development, and managing all public facing aspects of the project. The Project Team is governed and funded by the Foundation. Funding is reviewed by the Foundation every twelve months to ensure that the Project Team remains aligned with the vision and goals of Loki and are managing their finances appropriately.

The Loki Funding System – The Funding System will be allocated a set percentage of the block reward. Anyone can put forward a proposal for funding, no matter the subject. The Foundation will review and vet proposals to determine whether they align with Loki's vision. Proposals deemed to be valid will be communicated to the network. Service Nodes can signal their support for proposals through on-chain voting. If successful, the project will be funded. Funds will be released by the Foundation as agreed milestones are met.

Checks and balances are built into this governance model. For example, it is possible for the Foundation to be overthrown through a hard fork where the receiving address for the governance pool is changed. For this to happen, a majority group of users must agree on the outcome. This is a mechanism to incentivise the Foundation to act for the benefit of the Loki ecosystem and reflect community interests in their decision making. Further, the Project Team can have their funding restricted if they fail to adhere to the vision and principles of Loki. Funding may be reinstated when they realign with Loki's goals, or the Project Team may be replaced altogether.

This governance model will ensure that Loki's funding remains secure and independent into the future, guaranteeing the continued development of the project and community.

8 Conclusion

Loki proposes a model for anonymous transactions and decentralised communication, built on a network of economically incentivised nodes. Loki uses the foundations of Monero to ensure privacy and implement an incentivised full node system to enhance resilience. Full nodes are awarded a percentage of the block reward for their work, and are used as a platform to provide a second layer of decentralised services to the network. These services are referred to as SNApps, the first of which will be Loki Messenger. Ultimately, we envisage a suite of SNApps that will allow users to privately transact, communicate and access information.