Sylo Network: An incentivised peer-to-peer network

Sylo

March 6, 2020

Abstract

With the invention of blockchain, many traditional centralised services can now be decentralised. But with this comes a change in the relationship between consumers and producers, and how value is exchanged for these services. In a perfect altruistic world, producers would give up what they produce for free, but that is not always the case. We present the Sylo Network as a secure, scalable, incentivised P2P network where producers are rewarded for the services they offer.

1 Introduction

The Sylo Protocol allows developers to build decentralised communication applications. The protocol has been designed to work entirely on a P2P basis with the goal of being able to develop decentralised applications (dApps) with the same level of user experience (UX) as traditional messaging applications. Using a P2P system provides a high level of security and privacy, and avoids scalability issues related to the blockchain to which other dApps succumb.

The majority of users prefer to operate on a mobile device, which makes it challenging to provide a strong UX on P2P technology. Mobile devices are expected to go offline at any moment and are generally operating behind network address translation (NAT) gateways. They also have power consumption restrictions. Establishing P2P connections becomes difficult and expecting users to always be online and able to synchronise state with each other is entirely unreasonable. Asynchronous data exchange becomes a critical feature. Thus relying exclusively on the Sylo Protocol becomes untenable in achieving a satisfactory UX.

To solve this, the Sylo Protocol can be augmented through the use of *Service Peers*. Service Peers run the regular Sylo Protocol client application but advertise additional services that can assist Sylo Protocol peers in communications. Together, the Service Peers form a trustless, decentralised network of services that enable a decentralised UX similar to existing centralised application.

While Sylo has bootstrapped the initial Service Peer network, having all Service Peers controlled by Sylo contradicts our goals of having a trustless, decentralised system. We therefore introduce incentivisation to the Sylo Network and show how it plays a vital role in establishing a completely trustless, decentralized, and scalable communications network.

1.1 Overview

The Sylo Network is composed of application level users running the Sylo Protocol and a consortium of Service Peers that facilitate user experience demands in a modern world. Service Peers provide the means for peer discovery, NAT traversal, message relay, and asynchronous data storage.

Within the network, Service Peers are incentivised to provide the necessary services to peers. This prevents applications built on the Sylo Protocol from relying on external infrastructure, either provided by a third party or established by peers themselves.

The Service Peer market is built around a probabilistic payment system. When peers require service from the Sylo Network, they purchase them from Service Peers using payment tickets. Each ticket has a probability of winning ERC20 compatible Sylo Tokens. Winning tickets are redeemed by smart contract.

Redeeming tickets requires that Service Peers stake Sylo Tokens with a lock-in period to re-

duce liquidity of their stake. Additionally, a stake-weighted selection algorithm is employed by peers when selecting a Service Peer. Running a Service Peer therefore requires an investment and this barrier acts as the primary deterrent against Sybil attacks.

As peers consume services, the majority of the transactions occur off-chain[1], directly between a peer and Service Peer. In this way, the payment system is able to scale massively and allows payments to be exchanged for extremely granular levels of service. A highly granular payment system significantly strengthens cryptoeconomic incentives as the profit and loss to an attacker and victim, respectively, are minimised.

2 Background

2.1 Peer-to-peer History

The P2P application architecture has been prominent throughout the 21st century. The architecture removes the common client/server model in favour of a distributed model and has proven popular in a variety of applications. The Sylo Protocol [11] is a P2P protocol used for secure group messaging.

The success of peer-to-peer architectures can be directly attributed to decentralisation. P2P applications cooperate to create a network overlay on top of the traditional Internet Protocol (IP) suite. In addition to consuming resources, peers also *provide* resources. This can result in non-degrading performance as the number of users increases, a property rarely found in centralised applications.

P2P networks naturally lend themselves to applications focused on privacy and security. End-to-end encryption (E2EE) is used to ensure messages are transmitted between peers *securely* and the P2P network ensures messages are transmitted between peers *directly*.

2.2 Limitations of P2P

Decentralised P2P applications are rapidly growing in the market but they still face a number of challenges and limitations.

[1]The term "off-chain" refers to interactions that do not require a blockchain transaction.

The prevalence of mobile devices creates a number of network challenges. As devices are moved around spatially, they frequently switch networks and will often be reassigned new IP addresses. This is of little concern when the device is dialling centralised services with a static address but establishing a P2P connection can be extremely challenging.

Mobile devices are also challenged by having finite battery charge. Unlike stationary, continually-powered devices, the operating systems on mobile devices continually optimise power consumption by putting applications to sleep or killing them entirely. Finding ways to circumvent the OS, when possible, are not ideal and the increased battery drain often leads to a bad user experience. As a result, a P2P application operating in the mobile world needs to handle the volatile and hostile environment that comes along with it.

Applications operating in a distributed P2P environment will frequently find peers to be offline. Applications must therefore be; flexible enough to operate without the desired resources, or coordinated enough to ensure the necessary resources are available from alternative sources.

2.3 Services Needed by Peers

The limitations of P2P architecture make it extremely difficult for dApps to provide users with the same user experience to which they are accustomed from more centralised applications. Solutions to some of these problems may be on the horizon, but it may also be possible to enhance the user experience of dApps by creating an additional service network, built using the same principles of privacy, security, and decentralisation.

2.3.1 Bootstrapping

There is a 'chicken-and-egg' problem for P2P networks. When peers first create the network, how do they ask ask each other for their addresses? One solution is *bootstrapping*, and it refers to a list of peers who can provide enough information to get a new peer into the P2P network.

Bootstrapping must be performed each time a peer joins the network. Bootstrapping methods vary by application, but some common bootstrapping techniques may include:

- a list of peers encoded in the application

- a peer provided manually by the user
- a hybrid action, where the user coordinates with the application to bootstrap from another peer (such as scanning a QR code)

Once bootstrapped onto the network, the addresses of other peers can be easily accessed using, for example, a distributed hash table (DHT).

2.3.2 STUN

Peers operating within a private network will not necessarily know their public IP address due to network address translation. In some cases, they may not even have a public IP address.

Depending on the network conditions, it is sometimes possible that a peer is assigned a public IP address but is unaware of its value. If this is the case, it might be possible to ask a third party to provide you with your public IP address. Requesting this information is a process known as session traversal utilities for NAT (STUN) [10]. If a STUN request is completed successfully, a peer is able to learn their public IP address and can then make it available on the peer discovery network.

2.3.3 TURN

The STUN service is known to be insufficient for peer communication under some circumstances (for example, behind a symmetric NAT). In cases like this, another protocol exists to establish connections with peers, known as traversal using relays around NAT (TURN) [6].

Providing the TURN service requires a larger commitment from the service provider as it requires them to *relay* messages between the two isolated peers. In spite of the extra effort, a TURN-like service is required for the majority of P2P connections over mobile networks due to the complex network topology.

2.4 Incentivised P2P Networks

The stability and utility of P2P networks is largely dependent on the continued cooperation amongst self-interested actors. Effective cooperation has been demonstrated through the use of reciprocal incentivisation (i.e. tit-for-tat) strategy [2]. It is used in well-known protocols such as Tor [4] and BitTorrent [12].

A tit-for-tat approach is unfitting for the Sylo Network as individual peers are generally not capable of providing the needed P2P services due to the aforementioned restrictions of the mobile environment. Thus in the context of a Service Peer to peer relationship, peers are typically restricted to being leeches.

Paying Service Peers directly for their services is an alternative incentivisation scheme but would only be suitable for Sylo if it could avoid any dependence on a central authority, such as a bank. A blockchain is a obvious alternative, although processing payments for even a moderately sized application becomes a pressing issue.

2.5 Layer 2 Frameworks

Blockchain transactions are infamous for their slow processing times. A Bitcoin transaction can take 10 minutes to process and Ethereum, though much faster, still takes around 15 seconds for transaction confirmation. This makes it impractical for applications that require a high transaction throughput. Each transaction on a traditional blockchain also requires a processing fee, a property that makes it infeasible to facilitate large numbers of micropayments.

To support large numbers of payments, frameworks have built on top of an existing blockchain that attempts to perform the majority of transactions off-chain. Such frameworks are called *Layer 2* frameworks, and they help solve the scalability problem of blockchain transactions. Using a Layer 2 framework provides a multitude of possible solutions, though only trustless solutions are suitable for the Sylo Network.

2.5.1 Micropayment Channels

A *micropayment channel* is a system that allows a high number of micropayments between two peers. The process generally only requires two blockchain transactions. An initial transaction holds the funds in escrow via a smart contract, and a settlement transaction distributes the funds. Intermediate transactions can occur entirely off-chain, improving the scalability significantly.

The Lightning Network [5], which operates over Bitcoin, is a prominent example of a micropayment channel. Micropayment channels are still somewhat limited, as they require an initial on-chain transaction to setup the channel. In the Sylo Network, peers may use multiple Service Peers and can change Service Peers often, so the scalability of micropayment channels could still be insufficient at large scales.

2.5.2 Probabilistic Payments

Reducing the overhead of processing transactions can be further reduced by using *probabilistic payments*. Initially, a handshake is performed between a payer and payee. Following a successful handshake, any number of payment tickets can then be exchanged. Each payment ticket has a probability to win a payment. While tickets are never guaranteed to win, as more tickets are exchanged, the expected payout approaches the value of a winning ticket multiplied by the probability of winning. A previous example of such a system is Rivest [9], who presented a centralised *electric lottery ticket* system, where bank processing was significantly reduced using probabilistic payments.

2.5.3 LivePeer

The LivePeer project [8] has built a decentralised probabilistic payment system, named Streamflow, on top of the Ethereum blockchain. The LivePeer protocol involves two actors; a *broadcaster* and an *orchestrator*. Broadcasters send video segments to orchestrators. Orchestrators provide encoding, live video, and on-demand video services and are paid for each video segment via an attached probabilistic payment ticket. Orchestrators also participate in the economy by depositing a stake, which can be slashed if they fail a verification check. Staking is an important step to prevent Sybil Attacks. Broadcasters also place a deposit with a time-locked withdrawal. The broadcaster's deposit provides a guarantee of payment and prevents certain kinds of attacks against the network.

Orchestrators advertise their services through an on-chain registry. Broadcasters use the registration to connect with orchestrators and can then further negotiate off-chain. An interactive protocol is used to generate tickets using secret random numbers. This randomisation prevents either party from maliciously affecting the probability for a ticket to win. Additionally, broadcasters can request transcoded video segments from an orchestrator to verify that orchestrator is acting in good faith. Broadcasters use an on-chain smart contract to challenge bad actors. If the challenge is successful, the orchestrator's stake is slashed and the broadcaster is rewarded.

2.5.4 Orchid

The Orchid project [1] is another example of an Ethereum-based probabilistic payments. Orchid provides a decentralised market for virtual private network (VPN) services. Orchid uses probabilistic payments to scale trustless interactions between a client and VPN provider at high frequency. The lower payment overhead allows Orchid to route traffic through multiple VPN providers.

The Orchid system requires VPN providers to stake Orchid tokens with a delayed withdrawal period. This presents a high cost of entry, providing an effective defence against Sybil attacks. Orchid also employs a stake weighted selection algorithm, that routes more traffic to VPN providers with higher stake resulting in higher rewards.

3 Goals

The design of the Service Peer incentivisation scheme is intended to align with Sylo's vision of ushering in a decentralised future. Sylo considers the Sylo Protocol to be the primary driving force behind the pursuit of decentralised communication.

Despite the strength of the Sylo Protocol, operating within the mobile-centric world of today makes Service Peers a critical component. The Sylo Protocol was built on a philosophy of secure, scalable, P2P messaging. It is vital that the introduction of a Service Peer market does not undermine these values.

3.1 Scalability

The Sylo Network should scale to support millions of peers sending messages across the network simultaneously. The probabilistic payment system needs to allow a Service Peer to process payments from

hundreds of peers at the same time, where each peer is expected to broadcast many payments per second. The actual number of transactions that occur on-chain should be minimised as much as possible. Additionally, peers should select Service Peers in a way that naturally load balances peers across the global network of Service Peers.

3.2 Decentralisation

To align with Sylo's goal of ushering in a decentralised future, the addition of an incentivisation scheme needs to be operable with no reliance on trusted third parties. All necessary interactions that occur in this system are either P2P or recorded on the blockchain.

3.3 Security

All components of the Sylo Network should be secure. Utilising a Service Peer should not jeopardise a peer's ability to communicate privately. Communication content must be entirely obfuscated from a Service Peer and any metadata that may reveal a user's identity should leak as minimally as possible. Additionally, confidence that a peer will behave well in the system is provided via cryptoeconomic incentives, with some avenues for misplay being entirely nullified by appropriate use of cryptographic techniques.

3.4 Simplicity

The overall system needs to be easy to understand and simple to audit. The security of the system should be easy to validate. The incentivisation system should avoid relying on complex proofs. Parameters to the system, such as pricing, are determined organically over time by a competitive market.

4 Market

The market used by the Sylo Network is a P2P decentralised market that allows users to provide decentralised and P2P services in exchange for Sylo tokens. A smart contract makes services provided by the market discoverable to users.

4.1 Participants

- **Users** are anyone/anything using the Sylo Network
- **Service Peers** are users who provide service to users
- **Peers** are users who consume services

4.2 The Sylo Token

The Sylo Token (SYLO) is the ERC20 compatible utility token used within the market. Other tokens were not considered as they did not provide the features needed by the market. They were also undesirable as they would be subject to many influences outside the scope of this market.

4.3 Staking

Becoming a Service Peer requires staking a certain amount of Sylo Tokens for a minimum amount of time. Probabilistic payment tickets are **only redeemable by staked Service Peers** and are limited to the proportion of their stake relative to the rest of the network. The stake amount also serves as a way for Service Peers to increase their economic gain by increasing the traffic they receive. When a peer wishes to find a Service Peer, a smart contract performs a stake-weighted selection algorithm that returns a list of Service Peers, weighted towards Service Peers with higher stake amounts.

After requesting a withdrawal of any staked tokens, a user must wait for the duration of a lock-in period to pass. This is a security restriction that creates an obstacle for potential attackers. The lock-in period is based on the time it takes for the network to identify and respond to an attack. Once a withdrawal has been initiated, the Service Peer can still provide services to the network and earn additional Sylo tokens. The lock-in period also gives peers the opportunity to select another Service Peer and migrate anything they deem necessary.

4.4 Service Peer Selection

Similar to Orchid, Service Peer selection is performed on-chain. Peers request a service by querying a smart contract. The smart contract will return a random, stake-weighted selection of Service

Peers who provide the requested service. Peers choose a Service Peer from the list. Once a peer connects to a Service Peer, the Service Peer will subscribe to the peer's activity on the P2P DHT.

Depending on the use case, peers can have multiple Service Peers for the same service. One peer might create multiple inboxes to provide redundancy and ensure they receive their messages. Another peer might want more privacy and use multiple Service Peers to send messages across multiple hops on the network.

4.5 Censorship Resistance

The smart contract is the primary way that Service Peers are discovered but a few alternative methods are also supported. Service Peers can advertise themselves over the P2P network or out-of-band. Service Peers operating outside the random selection process can still stake and redeem payment tickets.

Even when the smart contract is preferred over other methods, we consider the decentralised nature of the blockchain to be sufficiently censorship resistant. It seems unlikely that users would find themselves unable to use the smart contract.

4.6 Sybil Attacks

Identities in the Sylo Network are pseudo-anonymous keypairs, so identity generation is essentially free. This creates an opportunity for Sybil attacks, where an attacker can create a large number of identities that can collude together to control the market.

The primary defence against a Sybil attack is to require Service Peers to stake tokens and to use a lock-in period before any withdrawals. This defence should make it prohibitively costly to establish enough nodes to control the market.

4.7 Curated Whitelists

A curated list of well-behaved Service Peers can exist on-chain. Peers can optionally use this list as a filter during the Service Peer selection process. Using a whitelist can help prevent peers from falling victim to certain forms of bad behaviour (e.g. packet sniffing).

Whitelists act as an intermediate solution, as more complex proof and validation systems are developed. Sylo will initially manage an on-chain whitelist but implementations of the client application will let users select whitelists curated by other parties as desired. This ensures that the whitelisting process remains decentralised and allows well-known third parties to emerge as whitelist curators.

5 Probabilistic Micropayments

Service Peers offer a wide range of time-critical services, from relaying traffic to serving up inboxes of data, but traditional transaction-based payments are too slow and too expensive to accommodate these services. To address this problem, Service Peers on the Sylo Network charge for their services using a Layer 2 probabilistic micropayment protocol. This allows services to be provided to peers at speed and scale, unencumbered by the usual delays caused by block confirmation times, and while avoiding transaction fees associated with traditional Layer 1 payments. This payment protocol allows for highly frequent and trustless payments on the Sylo Network that are pseudo-anonymous and censorship resistant.

5.1 Definition

Probabilistic micropayments work in a way similar to lottery tickets. Instead of issuing a confirmed payment with every service request, peers issue tickets that carry a chance of winning some previously negotiated value. With increasing probability, the Service Peer may receive a winning ticket that is immediately redeemable for the negotiated value. The Service Peer redeems this winning ticket as a Layer 1 transaction on-chain within a reasonable amount of time.

There are two parts to this protocol:

1. A smart contract that acts as an escrow system that holds SYLO tokens and pays out winning tickets.

2. An off-chain protocol that allows creating, sending, and verifying tickets between senders

(peers) and recipients (Service Peers), or any two parties using the network.

5.2 Smart Contract

The transfer of SYLO between peers and Service Peers is managed by a dedicated smart contract that serves to ensure parties are paid out correctly and that bad behaviour does not go unpunished.

Deposit escrow funds Peers must first deposit some SYLO into the contract to serve as an escrow fund. Once deposited, funds are locked until a manual unlock is requested. This serves to give Service Peers the reassurance that connecting peers have sufficient funds available prior to accepting service requests.

Deposit penalty funds Peers must also deposit some SYLO into a penalty contract. Funds locked up in this contract serve to disincentivise bad behaviour by the peer. If a peer is found to have acted in bad faith, such as intentionally racing winning tickets in order to prevent payouts, the peer's penalty deposit may be slashed.

Redeem Once a Service Peer receives a winning ticket they may submit the ticket and claim their winnings from the associated peer's escrow balance.

Unlock If a peer chooses to withdraw the SYLO from the smart contract they must first unlock them. This initiates a lock-in period before any outstanding winning tickets may be redeemed. This waiting period serves to give Service Peers sufficient time to claim any winning tickets still in their possession.

Withdraw Once a peer has unlocked their deposits and the duration of the lock-in period has elapsed the peer can finally withdraw the SYLO.

5.3 Tickets

5.3.1 Creating and Sending Tickets

Peers send tickets off-chain through the P2P network. The flow of information is summarised in the following example where peer C wants consume a service offered by Service Peer SP

1. C ensures that there's sufficient funds in escrow and initiates a service request with SP

2. SP receives the service request, checks C's balance in escrow, and chooses to accept. SP proceeds to generate a random seed $seed$ and derives $commitHash = \text{HASHFN}(\text{HMAC}(SP_{secret}, seed|C_{address}))$.
   SP then responds by sending the $seed$, $commitHash$, and a set of other parameters, such as the suggested winning amount, and a suggested winning probability, to C.

3. C receives the response, inspects the proposed winning amount and probability parameters, and accepts. C is now ready to start generating tickets.

4. C then constructs a ticket T incorporating the previously negotiated parameters, the $seed$, $commitHash$ and a monotonically incrementing nonce. C signs $\text{HASHFN}(T)$ to produce a signature T_{sig} and sends both T and T_{sig} to SP.

5.3.2 Validating Tickets

Continuing on from the previous example demonstrating the flow of initialising, creating and sending tickets, SP must now validate and check incoming tickets to see if they are winners.

1. SP first re-computes the commit $commit = \text{HMAC}(SP_{secret}, T_{seed}|C_{address})$ and compares it for equality to $T_{commitHash}$. SP further checks if all miscellaneous parameters are as expected.

2. Finally, SP checks if T is a winning ticket by checking that $\text{HASHFN}(T_{sig}|commit)$ is lower than $T_{winningProbability}$

5.3.3 Redeeming Winning Tickets

After validation has completed successfully and SP learns that T is a winning ticket, SP may now redeem the ticket on-chain by submitting a transaction for redemption including T, T_{sig}, and $commit$.

5.3.4 Penalty Claiming

If the escrow balance of a peer is below the value of a winning ticket then their penalty balance will be drawn from. This is handled within the smart contract and requires nothing different for a Service Peer redeeming a winning ticket.

5.4 Pricing

Service Peers may advertise their pricing, including expected winning probabilities and amounts, in a number of ways. They could, for example, advertise their pricing along with the services they offer on the market smart contract, or interactively as peers connect.

Service Peers may also offer pricing to specific peers based on internal configuration, such as white- or blacklist of peers. Keeping such lists would further allow them to block bad actors, provide services at discounted rates (or for free) to select subsets of the network, and so on.

5.5 Economic Incentives To Prevent Attacks

Occasionally, it is not possible to redeem a winning ticket. This happens when there is insufficient balance in the escrow to cover all winning tickets, either due to attempted gaming of the system ("front running") or by accident when the deposit is across multiple Service Peers.

Both the problem of front running, as well as the problem of accidental races between winning tickets, have been thoroughly studied and deemed cost-prohibitive in the Orchid whitepaper [1] which employs a similar, micropayment based design.

6 Services

This section details the set of standard services that Service Peers may choose to offer.

6.1 Peer Discovery

Within a P2P network, there is no centrally maintained lookup table mapping peer IDs to dialable addresses[2]. Within a local area network, contacting peers is possible through multicast DNS (mDNS) but, failing this, peers have no means of discovering each other.

Peer Discovery is a service to address this problem. It provides peers with lookup access to the addresses of other peers. The service is implemented as a DHT [7] that is collectively maintained between all Service Peers on the Sylo Network.

This service is not explicitly incentivised, as participating in the DHT is a requirement for being able to offer other services.

6.2 NAT Traversal and Message Relay

Network address translation is ubiquitous and poses one of the key challenges in establishing direct connections between peers. Unless connecting peers are either on the same LAN or are manually configured to be able to accept incoming connections on the wide area network (WAN), NAT is unlikely to allow peers to connect over IP.

To address this problem, Service Peers offer NAT related services, such as responding to STUN queries, or relaying traffic outright, similar to classical TURN.

When relaying messages between peers, Service Peers may charge for both receiving and delivering messages. In this setting, the sending peer submits a payment ticket with the data being relayed, while the receiving peer is expected to submit a payment ticket after receiving data. Failure of the recipient to submit the payment ticket is discouraged because the Service Peer will not continue relaying messages to peers who do not respond with payment tickets.

Relying on Service Peers exclusively for establishing connections with other peers, however, is also a problem. Because Service Peers are incentivised to relay information for economic gain, they might be inclined to withhold information regarding alternative, potentially more direct, routes to peers. One possible solution is to slash the Service Peer's stake if they are found to be acting in bad faith. This requires the implementation of a proof and validation system, which is considered to be a future body of work.

[2] A typical dialable address might be an IPv4 or IPv6 address.

6.3 Inboxes

Communication between peers is only possible when both peers are online and connected to each other. This severely limits usability for users operating from mobile devices, since connectivity often fluctuates due to restrictions imposed on them by external forces.

Without an intermediary to hold data, peers would be required to return to the network frequently, in the hopes that the target peer would be online and able to communicate.

To address this problem, Service Peers may choose to offer an Inbox Service that stores data earmarked for a given recipient. The duration of storage is finite and configurable. Available data items are delivered to the intended recipient on demand. Service Peers may charge for receiving, storing, and delivering the data.

Configuration, such as the longevity of data stored and maximum storage capacity is configured by the Service Peer and this information is made available to peers. The configuration parameters are driven by economic competition between Service Peers.

7 Attack Vectors and Prevention

7.1 Incomplete or Poor Relay Service

Since Service Peers receive payment for relaying messages prior to actually forwarding them, it is conceivable that sufficiently motivated players might choose to simply drop or delay messages. It would be impossible to determine whether the intended recipient of a message is genuinely unavailable at the time the message was sent, or whether the Service Peer acted unfaithfully by dropping or delaying the message.

Poor service is likely to result in a loss of revenue for the Service Peer, as they will not receive payment from the intended recipient. This loss of revenue is expected to be a sufficient deterrent for this form of attack.

7.2 Involuntary Service Peer Nomination

A Service Peer may falsely advertise themselves as a paid relay for a peer without their consent.

This is easily prevented by requiring that the advertisement be signed by the peer. All advertisements have an expiry. This is useful when a peer wishes to revoke the nomination of a Service Peer, as they can submit a new advertisement with a very short expiry.

7.3 Centralisation of Services

A sufficiently resourced organisation could attempt to control the market by staking a majority of tokens, or by undercutting other Service Peers with their pricing to encourage peers to use their services exclusively. Of course, the cost of attempting this would be incredibly high and become less and less viable as the network grows in size. However, an organisation may want to achieve this to specifically infringe on the ideals of decentralisation.

We do not consider this to be a critical problem, as peers can easily set up or choose their own Service Peers, and the Sylo Protocol itself can fully operate without the need for Service Peers.

7.4 Spam and DDOS

In an effort to route more traffic to their own nodes, Service Peer operators might launch targeted attacks at competing nodes in the network.

One such attack might be to spam Service Peers with invalid tickets. Service Peers would waste CPU cycles attempting to redeem such tickets. This attack is mitigated by the mandatory setup phase performed by the peer and Service Peer before tickets are issued. During this setup phase, the targeted Service Peer can query on-chain state to confirm the peer has sufficient funds for the service. If confirmation fails, the peer can be blocked.

Yet, Service Peers may still suffer from more sophisticated distributed denial of service (DDOS) attacks, where attackers generate a large number of identities for free and use them to initiate service requests. We consider this problem to be similar to traditional DDOS attacks, and expect similar solutions to apply. Possible solutions might include:

purchasing more bandwidth, rate limiting, load balancing, traffic shaping, and so on. It is the responsibility of the Service Peer operator to employ such strategies as they see fit on a per deployment basis.

8 Future Work

8.1 File Storage

Service Peers are only discouraged to improperly provide their advertised services for fear of not maximising their economic gain. This is considerably less effective at discouraging poor behaviour compared to slashing, and this approach is not appropriate for data which requires high levels of availability and redundancy. Without a robust proof and validation system in place, utilising Service Peers for file storage in a trustless manner is infeasible. Research into this area is a nascent space, with projects such as Filecoin expending a huge amount of resources to devise their Proof of Storage solutions[3]. A future body of work for Sylo will be to explore the viability of integrating a File Storage system, as the research in this space continues to mature.

8.2 Payment Anonymisation

Redeeming a winning ticket leaks an explicit relationship between a particular peer and Service Peer. When a winning ticket is claimed there becomes a record on the blockchain of payment which includes information on who created the ticket and who redeemed the ticket. The severity of this issue is lessened by using psuedo-anonymous keypairs for peer and Service Peer identities. The use of Non-interactive zero-knowledge proofs is a potential avenue for exploration, and has seen successful use by projects such as ZCash for anonymising Bitcoin payments[13].

8.3 Blocking Nodes

Since P2P networks have no predefined topology, and are without a central switch-board that would connect peers to each other, peers joining the wider network must have at least a single, pre-configured, and dialable entry-point to bootstrap their view of the network. In the context of the Sylo Network, the list of such bootstrap nodes is the list of staked nodes on the blockchain.

Yet having these endpoints be publicly accessible could be considered a weak point, as governments, internet service providers, or other large entities might be able to block well-known endpoints in an effort to break apart the P2P network.

We plan on exploring ways to prevent this such as using non-IP based protocols like Bluetooth.

8.4 Multi-Hop System

Implementing multiple hops across the network is a viable strategy to improve privacy when relaying data via Service Peers. When using this strategy, data is routed through several intermediate Service Peers before finally being delivered to the intended recipient.

Messages are wrapped in several layers of encryption. Each layer is encrypted specifically for one of the Service Peers on the route. This effectively hides the origin of the message from everyone except the entry Service Peer and hides the destination of the message from everyone except the exit Service peer.

Selection of the intermediate Service Peers could be implemented using the aforementioned smart contract selection algorithm.

9 Summary

The Sylo Network brings together a P2P network with an incentivisation protocol based on a Layer 2 probabilistic micropayment protocol that provides the granularity and scalability required for services offered by the network. It is secured with the help of the blockchain to ensure a sustainable economy exists. The first application of the Sylo Network is the Sylo Protocol [11], a secure group messaging protocol that utilises the services provided by the network. Future development will see more services being offered on the network, improved privacy, and probabilistic micropayments extending beyond the scope of just the Service Peer network.

10 Acknowledgements

The Sylo Network is only possible because people believe in the ideas it represents. A big thank you to everyone who shares our visions and believes in what we're doing.

References

[1] Jake S. Cannell et al. *Orchid: A Decentralized Network Routing Market*. Nov. 18, 2019. URL: https://www.orchid.com/assets/whitepaper/whitepaper.pdf.

[2] Michal Feldman et al. "Robust Incentive Techniques for Peer-to-Peer Networks". In: *Proceedings of the ACM Conference on Electronic Commerce* 5 (Aug. 2004). DOI: 10.1145/988772.988788.

[3] *Filecoin: A Decentralized Storage Network*. Protocol Labs. July 19, 2017. URL: https://filecoin.io/filecoin.pdf.

[4] David Goldschlag, Michael Reed, and Paul Syverson. "Onion Routing for Anonymous and Private Internet Connections". In: *Communications of the ACM* 42 (Feb. 1999). DOI: 10.1145/293411.293443.

[5] Thaddeus Dryja Joseph Poon. *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*. Jan. 14, 2016. URL: https://lightning.network/lightning-network-paper.pdf.

[6] R. Mahy, P. Matthews, and J. Rosenberg. *Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)*. RFC 5766. http://www.rfc-editor.org/rfc/rfc5766.txt. RFC Editor, Apr. 2010. URL: http://www.rfc-editor.org/rfc/rfc5766.txt.

[7] Petar Maymounkov and David Mazières. "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric". In: *Peer-to-Peer Systems*. Ed. by Peter Druschel, Frans Kaashoek, and Antony Rowstron. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 53–65. ISBN: 978-3-540-45748-0.

[8] *Probabilistic Payments*. Oct. 2, 2019. URL: https://github.com/livepeer/wiki/blob/master/spec/streamflow/pm.md (visited on 02/25/2020).

[9] Ronald L. Rivest. "Electronic lottery tickets as micropayments". In: *Financial Cryptography*. Ed. by Rafael Hirschfeld. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 307–314. ISBN: 978-3-540-69607-0. URL: https://people.csail.mit.edu/rivest/pubs/Riv97b.pdf.

[10] J. Rosenberg et al. *Session Traversal Utilities for NAT (STUN)*. RFC 5389. http://www.rfc-editor.org/rfc/rfc5389.txt. RFC Editor, Oct. 2008. URL: http://www.rfc-editor.org/rfc/rfc5389.txt.

[11] Sylo. *The Sylo Protocol*. URL: https://developers.sylo.io.

[12] *The BitTorrent Protocol Specification*. BitTorrent. Jan. 10, 2008. URL: https://www.bittorrent.org/beps/bep_0003.html.

[13] *Zerocash: Decentralized Anonymous Payments from Bitcoin*. Electric Coin Company. May 18, 2014. URL: http://zerocash-project.org/media/pdf/zerocash-extended-20140518.pdf.