Indranet White Paper

Programmable onion routing distributed virtual private network protocol with anonymised payments to create scaling incentives

David Vennik September 2022

Abstract

The current state of counter-surveillance technologies has remained largely unchanged in the 20 years since the inception of the Tor network. The primary use case has always been obscuring the location information of users from clearnet sites, and the more it has been used for this purpose, the more hostile clearnet sites have become towards this network, due to its frequent use to launch attacks on web services.

With the increasing amounts of value being transported in data packets on the Internet since the appearance of the Bitcoin network, the need for eliminating the risks of geographical correlation between payments and user locations continues to rise.

However, without any way for users to pay routers without creating an audit trail, the networks have a severe scaling problem in that in anonymising data, there is an increase in privacy with the larger number of nodes and users, and thus attackers have largely been able to keep pace and pluck off high value targets with state-sized players, such as the Carnegie Mellon University: blog.torproject.org/did-fbi-pay-university-attack-tor-users/.

Thus, it is the central thesis of this paper to demonstrate how decorrelation between payments and session usage can be achieved and create a marketplace in routing services which can economically increase to a size that is beyond the capabilities of a state sized actor to fund an attack.

Indra creates mechanisms for anonymous purchase of chaumian vouchers used to initiate traffic sessions with router nodes, which then compensates routers for their running costs, and further, focuses on hidden services and Bitcoin/Lightning (and potentially other Bitcoin related systems) in order to reduce the attack surface from large actors who have thus no open justification for censoring the network.

1. Chaumian Routing Vouchers

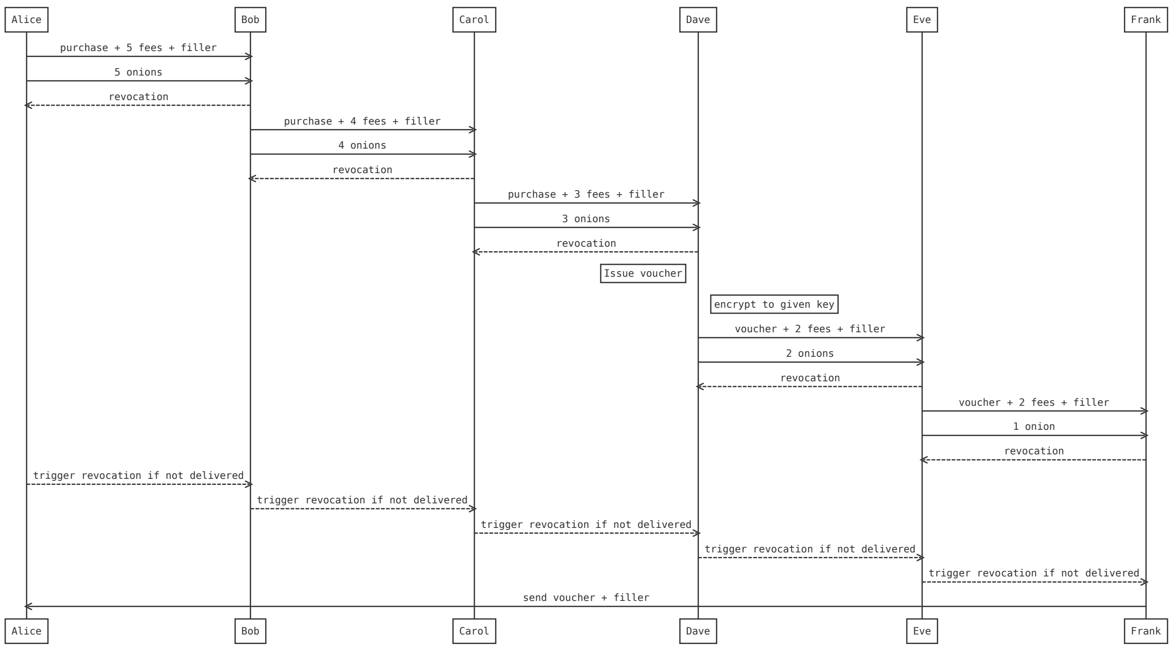
Through the use of blinded signatures, it becomes possible for a token to be created with some arbitrary data, usually a denomination of a currency, and the buy and sell cannot be correlated to each other, the signature is valid, but it cannot be directly linked with the minting.

However, in a distributed VPN, there is a need for fairly close correlation in time between purchase and spend, which presents a problem for counter-surveillance and eliminating traces connecting traffic timing, origins and destinations.

Thus, the purchase of these vouchers is protected by a 6 stage onion route that delivers Lightning transactions and returns vouchers in such a way that no single participant in the process can know more than their adjacent nodes, and not know, with exception of the voucher seller, what position they are within the circuit.

Purchasing Protocol Flow

Thus, the purchases are made via payments, and each node passes on the decrypted message, which then provides the payment destination for each subsequent hop, in a circle that goes through 5 nodes. This is the top level view of the process:


Alice
Bob
Carol
Dave
Eve
Frank
purchase + 5 fees + filler
5 onions
revocation
purchase + 4 fees + filler
4 onions
revocation
purchase + 3 fees + filler
3 onions
revocation
Issue voucher
encrypt to given key
voucher + 2 fees + filler
2 onions
revocation
voucher + 2 fees + filler
1 onion
revocation
trigger revocation if not delivered
trigger revocation if not delivered
trigger revocation if not delivered
trigger revocation if not delivered
trigger revocation if not delivered
send voucher + filler
Alice
Bob
Carol
Dave
Eve
Frank

Each fee is different by 1, for example: 5, 6, 7, 8, 9, 10, enabling a check by the remainder at the last step of the route. This gives one source of information about the success of the circuit as well as whether any node has taken a greater fee than was specified in the onion. Filler is required to eliminate any leak of information about a node's stage in the process. Seller obviously will know they are 4th but they only know their previous and next. Thus, in a given circuit, if the remaining buffer does not match expectation, and nodes have overcharged, all nodes in the circuit will gain a ban score, and over time as the nodes are reused ban scores will rise on the ones that are common to circuits that are the cheaters. There is inherently a low cost in small gouges of fees in this process, sometimes. This can be considered to be a risk in exchange for the security, bounded by the filler.

Revocation for each hop in the loop is an onion message that is encrypted to the previous sender, which unlocks the reversion transaction. If Alice doesn't get her voucher from Frank, before the period of the timeout, she can send the revocation key to Bob and Bob sends on through the other 5 and the channel state is reverted.

It is critical that no single entity in this chain knows any more than the origin of the message and the next place it is to go. So the construction of the message goes in the reverse order. In order to incentivise this process each step has a small fee, likely in the order of tens of satoshis.

1. Message to Frank, send packet to Alice plus fee
2. Message to Eve, send packet to Frank plus fee
3. Message to Dave, issue voucher and encrypt to provided key, remainder to fees
4. Message to Carol, pay forward to Dave minus fee
5. Message to Bob, pay forward to Carol minus fee

The message segments are randomly positioned in the payload and obscure the sequence point of each participant's message in the process, coin flip between append or prepend for each layer, and the next message segment is padded with random values so each hop has an identical sized message with a section they decrypt and the remainder with the next hop message.

In this way, a user pays for a voucher, and receives it without there being a direct trace either in the message forwarding or the ordering of lightning payments. The base of the fee size is a random amount excess, between 2 and 3 times and at the end part of Alice's original forward payment goes back, eliminating any correlation between fee size and protocol sequence.

Onion Routing and Revocation

The scheme is similar to the "Sphinx" onion routed scheme. We are basing a large part of this design on Blitz:

https://www.usenix.org/system/files/sec21fall-aumayr.pdf

Blitz has a one way flow and as it mentions deep in the appendix this can be done using onion routing, to protect identity.

The revocations in plain Blitz include all the identities along the chain. The key distinction here is that in the revocation packet each node gets, there is encrypted to the previous node the revocation code, so in order to close the loop they send the revocation back, and forward the revocation onion they have after sanitizing their segment. In this way, the anonymity is preserved while triggering the revocation forwards.

If a griefer or wormhole attacker causes the chain to break, the revocation breaks the transactions going forwards that were honestly propagated. The entire data structure is contained in the packet that is initially sent out by the buyer, wrapped in layers of encryption. The only thing that has to be done is if, based on the sender's measurement of the network topology and latency the chain has probably been broken, is to send out the revocation onion message which creates a revocation circuit that will then terminate at the attacker's channel.

Because the reverse transaction will have a total amounting to the extent the chain propagated until it stopped, the sender also is then able to determine the node which broke the chain, and with current data on the status of the node appearing to be online, determine whether it is a transient network failure or a malicious node, and if malicious, then it is simple to then push this node to the bottom of the list of options with a record of their mischief.

Spending Vouchers Flow

Once a user has acquired these vouchers, they can then use them in their onion routing packets to initiate sessions and spend the tokens.

The issuers receive special expiring payments that time out in case of the node going offline. When they receive the chaumian voucher, it contains the revocation key to stop the payment expiry and finalize it, which they can then apply at any point until the LN payment's expiry. Typical expiry would be something of the order of one day.

They send back the session cookie, a 256 bit value, encrypted to a provided key, using a rendezvous routing packet, and after this, the user then attaches this to every packet in the relevant node's layer of the onion and the node will continue to forward them to the specified destinations until the session count of packets is used up.

2. Routing Patterns

In the Tor network there are two predefined routing patterns used, the three hop to exit, and the 6 hop rendezvous path.

Indranet will use a modular, extensible onion packet construction, and exit routing is not a priority, but rather to focus on hidden services and rendezvous routing, and special types of exit paths that deliver messages to Bitcoin full nodes and Lightning Network nodes.

Because all Indranet nodes must be running a Lightning node, and therefore a Bitcoin node, these are valid exit networks that can be coded into the final inner layer of the outbound route onion. Potentially in later iterations a clearnet exit method may be added, but the liability and risk imposed on exit nodes means that this will be a premium priced service, and outside of the scope of the first generation of Indranet.

Because only the rendezvous paths are going to involve large quantities of data, it will need to be possible to vary the routing pattern to provide different properties to the traffic footprint.

Reliability, Latency and Obfuscation

These three properties can be improved via the structure of the onion construction.

Reliability

Reliability can be produced by expanding packets with Reed Solomon Forward Error Correction (FEC), and sending these packets in parallel. Any balance of N of M redundancy can be specified to the onion construction engine, most likely patterns of 2, 3 and 4 parallel paths would be used.

Latency

Latency can be improved by using parallel paths with two instead of three hops. Instead of, or in addition to redundancy, packet data is split into segments using Shamir's Secret Shares, and N of M must be received over a fan out/fan in two hop path for each side of the Rendezvous. The reliability can be tuned in parallel with this when packet drops occur.

Obfuscation

In addition to these simple parallel path patterns, it is also possible to open multiple sessions with a larger number of routers and vary the onion path in each packet, in addition to also potentially using short path for latency, in a way that further obscures the traffic's pathways.

Notes

These features may not be as useful as they sound in practice, but the means to implement them should be available.

Note that parallel paths incur a proportional bandwidth cost, which should reasonably match up with the benefit of lower latency, increased reliability or higher security.

3. Rendezvous, Forwarding and Inbound Routing

Because of the interference of routers and especially Network Address Translation, it can be that a node may not be able to directly receive inbound traffic. One of the big advantages of running a very large scale distributed VPN is that there is usually many infrastructure nodes that can proxy inbound access to nodes that cannot.

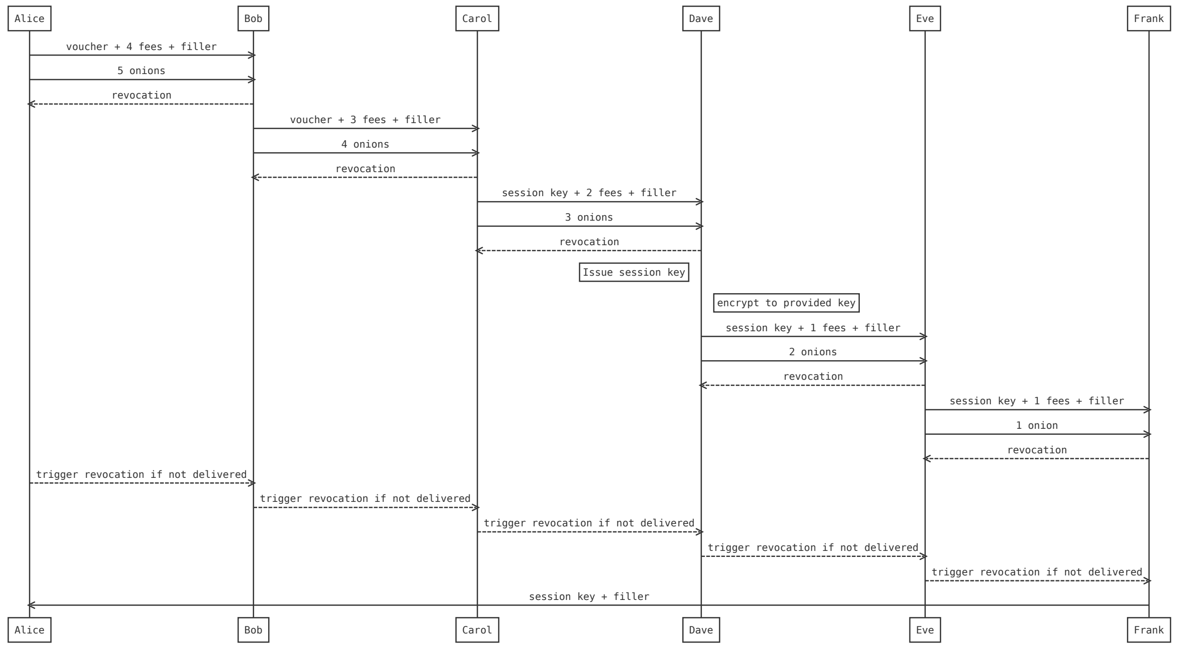
Peer to Peer network systems all have this difficulty of negotiating inbound routing in order to provide services. Thus, there is always a need to enable this proxying of inbound routing.

Normally this is done simply through Rendezvous routing, for hidden services, but because this inbound routing issue can be a problem, the programmability of the routing paths in the previous section also means it can be simple for nodes to create "open" rendezvous points that do not attempt to hide the location of the server. This still results in traffic on the network that adds the anonymity set for the anonymising services, and can be charged for the same way.

4. Session Initiation

In order to open a session with a router, a client node has performed a purchase operation where the desired router is the seller, and have a blinded signature on a packet that encodes the claim while not identifying when or who made the purchase by paying for it using the purchase protocol.

Session initiation follows the same pattern as the purchase protocol, except instead of a forward payment for the voucher, the voucher is sent forward and there is only 5 hops:



Alice
Bob
Carol
Dave
Eve
Frank
voucher + 4 fees + filler
5 onions
revocation
voucher + 3 fees + filler
4 onions
revocation
session key + 2 fees + filler
3 onions
revocation
Issue session key
encrypt to provided key
session key + 1 fees + filler
2 onions
revocation
session key + 1 fees + filler
1 onion
revocation
trigger revocation if not delivered
trigger revocation if not delivered
trigger revocation if not delivered
trigger revocation if not delivered
trigger revocation if not delivered
session key + filler
Alice
Bob
Carol
Dave
Eve
Frank

The failure mode here would require accounting the failure against the node and their voucher. A node would have to put the entry to the bottom of the list, as there can be a simple offline failure here, which may resolve in a period of time.

It is not practical to account for this situation, as the payment was made anonymously and a cancellation would necessarily require a repeat of the purchase process but for refund, and for the non-use case, is a trivial fee for one block of traffic and thus, there is no expiry, only the small risk that the router is offline and their voucher can be retried later, and eventually expired if storage requirements dictate.

The worst case scenario here is a user may have to wait a few seconds to purchase a few sessions if they have run short, and lose a few parts of sessions or vouchers if the node goes offline.

There is an attack potential in the creation of nodes that attract purchases and then go offline before their services are delivered. For this reason, nodes will prefer to buy vouchers from the longest lived routers and new routers will have a longer lead time in acquiring regular work.

5. Creating Circuits

In most descriptions of Onion Routing protocols, it is usual to discuss the circuit creation protocol. The foregoing sections describe the Purchase and Session Initiation protocols which form a shared secret which allows the fast establishment of circuits.

The process of establishment of an onion routed path usually requires the creation of an secret key value that can be used to establish an initial cipher set for the onion path. Because these blinded signature Chaumian Vouchers are private between the client and router, these form a base to establish the send and receive keys used as well as establishing a bandwidth tracking protocol to enforce a session protocol.

Sessions then can be tracked by the use of a hash chain on each side of the routing protocol, so each subsequent packet must be the hash of the previous hash rooted in the original session key based on the blinded signature based voucher.

The router has a value stored in their paid routing service tokens, and the user has the blind signature token inside the blinding, enables both sides to have a private value that enables a faster session initiation than standard onion circuit establishment.

6. Session Encryption

In order to reduce the potential for capturing encryption keys, all sessions, on each onion layer, use the Double Ratchet key negotiation algorithm. This key change will be triggered by the sender at a rate of around once per 4 seconds, to reasonably bound re-keying overhead. This will mean on average 1 key change per second per onion layer for exit paths and a little more for circular paths.

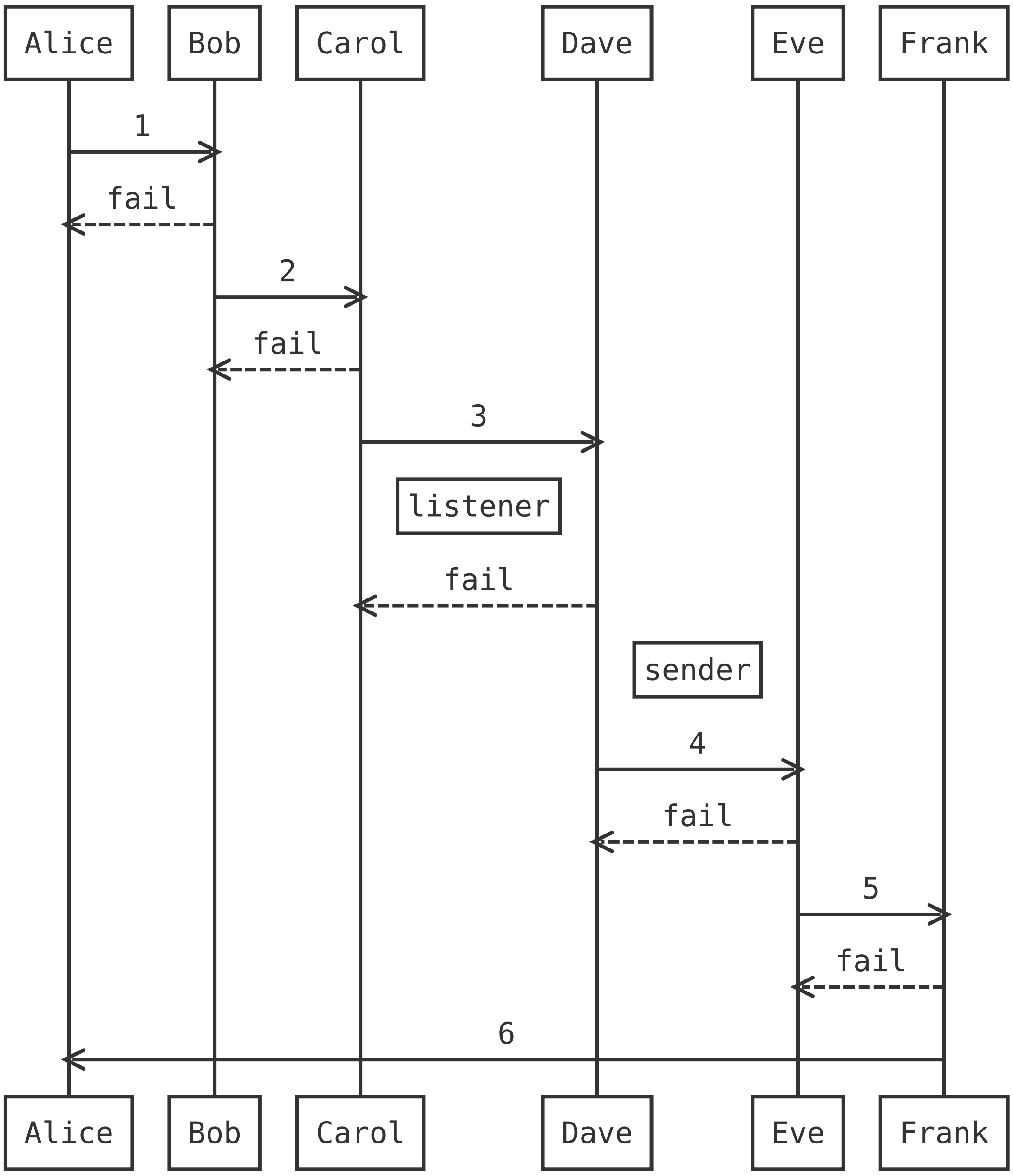
The individual message segments, composed of one or several sequential UDP packets, will be identifiable by a hash chain sequence which is used by the nodes and clients to keep an account of the bandwidth remaining in a session. Since this is not cryptographically feasible attack target it will use HighwayHash 64 bit hash chains, which will restart every 4 seconds with the key change of the session re-key.

7. Rendezvous and Hidden Services

Since other than Bitcoin and Lightning networks, Indra does not provide, by protocol, exit nodes to tunnel out of the network, the third and equally important endpoint for a 3 hop circuit is a rendezvous.

On each side of the rendezvous, the client and the server create a 6 step circuit, not reusing the intermediary, so involving 5 other nodes, the middle point being the rendezvous, equivalent to the voucher purchase and session initiation.

The topology of the onion is the same as the Voucher Purchase and Session Initiation, except each layer only contains a session hash chain sequence, the onion route, a time to live, and the payload.



The last 3 layers of the onion only provide the directions, but are encrypted to the sender's key, with the pre-negotiated hash chain sequence header and symmetric key for the next hop, concealing whether the packet is outbound or inbound, as their format is the same.

On the other side for the hidden service, the same pattern is provided, and circuits complete when one side sends and then the other side completes its half way journey, and vice versa.

Failure modes

Note the "fail" paths - these are messages that propagate backwards if the TTL does not get an acknowledgement from the next step in the path. These allow the customisation for the type of traffic, for interactive versus bulk transfer.

If the failure return cascades are triggered, it means a node in the circuit is either offline or congested, and the onion routing will be reconfigured with a different node in the failure point, which returns as the hash chain counter value, which identifies the node that failed to relay within the time limit.

All traffic must be prompted by the client

In order to enable this dynamic path changing, this network has a constant chatter. For every message that is desired to return to the client, there must be an onion constructed to shepherd the payload back from the rendezvous point or exit. Thus a ping will come with an onion that routes the pong back to the client, for example.

Obviously this incurs a substantial cost in AES encryption for the onion layers, and forces a constant signaling pattern, but this improves the anonymity set anyway. The volume/structure of onion circuit construction has a signature in the traffic, which Indra works around by separating the voucher purchase and session initiation, which means that large scale surveillance operators do not have very much useful timing data as packets are largely uniform in size and constant frequency. This also dictates that the maximum payload size an onion can carry is limited, in order to ensure there is a uniform packet size and frequency, removing timing data of what is carried by the onions.

Note that return circuits the endpoint is provided the encryption keys for the three hops back, but these keys are built from the secret knowledge of the client from the session double ratchet, and change with every new message. Thus they give no information to the endpoint about the path back to the client.

8. Anonymity is not everything

In addition to creating rendezvous paths of arbitrary structure to rendezvous points, there can be "clear" exit points, which essentially amount to connecting to servers running on the same node as the router. This is by default lightning and bitcoind, but could feasibly be anything, the security isolation would be a factor of the protocol's structure and sensitivity and value of the data it handles. In simple terms, it is like port forwarding on NAT.

A second use case that is not related is providing internet service through an "open" hotspot. The hotspot would refuse to relay normally, but has an Indra listener which can be negotiated with to send vouchers or LN payments and then becomes a usable access point. This ends the conflict between open hotspots and abusive users, as all users have to then pay for bandwidth, at minimum, as for one hop in a chain of the Indra network.

Where you can go from there, depends on the policy of the router, which will generally mean you are inside Indra using LN, Bitcoin or Indra messaging. Because the router is running LN and Bitcoin, it can freely provide sync data for the chain and for the user's channels, and thus combined with Neutrino, enables ubiquitous full SPV nodes on mobile devices. Users running these hotspots can also alternatively levy a higher charge for clearnet exiting, but as a general default, the purpose of this network is to enable access to hidden services, not tunneling. Hidden service access costs are lower because the endpoint cannot correlate to the entry point, and thus do not potentially present a security liability for the node operators.

Owners of such networks will then have special owner keys which let them send traffic on their own nodes without paying, and this service can then be exposed on unsecured wireless access points and become a direct source of income for the owner when others use it, while remaining secure by only carrying in-band traffic destined for other onion routers lightning/bitcoin and hidden services.

This also can ultimately facilitate more security for IPFS and Bittorrent networks as well, because everything adds to the anonymity set, if it is tuned to work with it well. One of the big problems with Tor is it is tuned to the TCP/HTTP use case and this is only part of network traffic usages. So additionally to LN/BTC there can be specific "exits" for IPFS and Bittorrent ports.

Thus, as a later stage of implementation, these features should be included, and to enable it, an extensible proxy/socket protocol needs to be devised, built for the smallest use case set, and designed to be extensible for these several cases.