

Exploiting the Lightning Network for Cross-layer Deanonymisation of Bitcoin Nodes

Charmaine Ndolo
Dresden University of Technology

Nepomuk Bosse Linsenbarth
Dresden University of Technology

Florian Tschorsch
Dresden University of Technology

Abstract—While the network-level security of Bitcoin and the Lightning network (LN) have each received a fair amount of attention, their relation is yet to be taken into consideration. The limited body of cross-layer research focuses on privacy but overlooks the peer-to-peer (P2P) layers and how it can potentially support cross-layer deanonymisation. This work bridges the two lines of research and presents the first cross-layer, network-level study of Bitcoin and the LN. We show how co-location of a Bitcoin and LN node can be exploited by a network adversary, e.g., an autonomous system, to support deanonymisation of the Bitcoin node. We introduce an attack that is capable of linking an IP address to a Bitcoin address and a real-world entity. It uses information gathered from the LN’s P2P network to find a relevant address in the blockchain and the alias announced in the LN to match the node to a real-world entity. Our measurements of the live networks and emulation-based evaluation demonstrate the practicality and merit of the attack. Among others, we find that institutional entities are more likely to be fully deanonymised than individuals. We discuss different countermeasures that can be taken at different levels to reduce the viability of this and similar attacks. We examine the impact of using non-default ports for P2P communication and show that encrypted traffic can still be detected reliably. This suggests that both networks need traffic shaping mechanisms to be resistant to traffic analysis.

1. Introduction

Although Bitcoin [1] is the most popular cryptocurrency by market capitalisation, it faces scalability limitations [2] which hinder its ability to compete with traditional payment systems. Layer 2 solutions are a promising approach to address these challenges by facilitating off-chain transactions. One such solution is the Lightning network (LN) [3] which is a peer-to-peer (P2P) network of off-chain bilateral channels, i.e., a payment channel network (PCN). Payments can be settled faster as they do not require consensus from the Bitcoin network and are secured by cryptographic protocols that guarantee atomic execution [4].

Bitcoin places strong emphasis on user anonymity with various protocol-level mechanisms in place to protect security and privacy. For one, the true origin of a transaction is obscured by its P2P broadcast protocol that makes it difficult to identify the origin. Additionally, the origin and destination of a transaction are identified by pseudonymous addresses which inherently contain no mapping to real-

world identities. Nonetheless, prior work has shown that it is possible to deanonymise transactions in spite of these measures [5]–[11]. Deanonymisation in this context refers to identifying the node that actually originated a transaction or clustering transactions based on ownership.

These deanonymisation risks are further amplified by centralisation. Both Bitcoin and the LN exhibit a high degree of centralisation at the network layer. Bitcoin’s network infrastructure is concentrated at a small number of autonomous systems (ASs) [12]–[14]. This has led to research on attacks in Bitcoin by a network-level adversary, e.g., an AS, such as eclipse attacks [12], [14], [15] and deanonymisation [16]. The LN exhibits similar centralisation patterns [17]–[19] and network layer attacks have become of interest, especially with regard to traffic analysis. Although all of these works have produced notable results for the respective applications, none of them have considered the two in unison or examined potential cross-layer effects.

This work combines observations from both networks and presents a passive, network-level deanonymisation attack on a Bitcoin node. We show how a network-level adversary, e.g., an AS, can exploit co-location of Bitcoin and LN nodes on the same host to discover the full identity of the Bitcoin node. The identity refers to the tuple of IP address, Bitcoin address and real-world entity name. Linking a node to a pseudonymous Bitcoin address can, for example, enable profiling, surveillance and censorship. Our eight-week measurements show that over 400 IP addresses were present in both overlay networks, proving that co-location is more than just a theoretical scenario (Section 3). To the best of our knowledge, this is the first study operating on multiple layers of the TCP/IP stack as well as of the Bitcoin stack. The adversary combines network-layer observations in the LN with application-layer information from both Bitcoin and the LN to conduct reconnaissance on the victim (see Fig. 1 for an overview). Our attack is different from prior work [20] as it is passive, requires no monetary investment and pursues a different objective. It also infers the linkage between nodes from co-location and not from blockchain analysis [21].

Starting from the victim node’s IP address that is known to the adversary, we use the opening of a channel in the LN to infer that a transaction with an address associated with the victim will soon be in a block. The channel opening event is detected by message type classification at the LN’s P2P layer. As the specific block containing the transaction is not known to the adversary, we show how they can utilise the timing between observed LN messages to precisely deter-

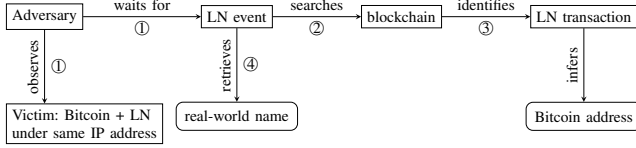


Figure 1. Overview of the deanonymisation attack.

ine the exact block. This is challenging as the number of confirmations required before the transaction is considered valid varies and is not available to the adversary at this stage. The adversary then applies a heuristic to the block to obtain the specific transaction and thus an address belonging to the victim node (Section 4). The heuristic uses transactions’ structural properties and is a revised version of an outdated heuristic [22]. We complete the tuple by using the LN node’s public alias to link it to a real-world entity’s name.

We evaluated the attack using a combination of measurements on the live networks and emulation in a realistic testbed (Section 5). We confirm that message type identification in the LN is highly accurate and show that our heuristic for identifying funding transactions is reliable. Our analysis of co-located nodes’ aliases shows that revealing aliases are more common among institutional entities and less so among individuals: 20% can be attributed to organisational entities but only 4% can be attributed to individuals.

As a final contribution, we discuss countermeasures that can be taken by protocol developers and node operators against our attack and, more generally, network-level attacks (Section 6). We evaluate the impact of using non-default ports, which is a commonly-suggested, practical measure [12], [17], by examining if Bitcoin and LN traffic can be detected as belonging to the respective protocol. We train a machine learning (ML) classifier on a comprehensive dataset of traces from P2P networks and common communication protocols. Our results show that detection of Bitcoin and LN traffic is accurate and fast, meaning that using non-default ports serves primarily as a deterrent to simple port-based filtering but is not adequate on its own.

2. Background

2.1. Bitcoin

Bitcoin [1] is a decentralised P2P network in which nodes maintain a shared ledger (the blockchain). Each node maintains up to 125 connections by default and P2P communication may be encrypted by the V2 P2P transport protocol [23]. Transactions, blocks and node addresses are exchanged through these connections over a gossip protocol. A transaction is first propagated by the origin to its peers who then validate it before propagating further to their peers. Each node stores valid transactions in their *mempool* (local storage of pending and unconfirmed transactions) and miners select transactions from their mempool when assembling candidate blocks. Transactions are then grouped

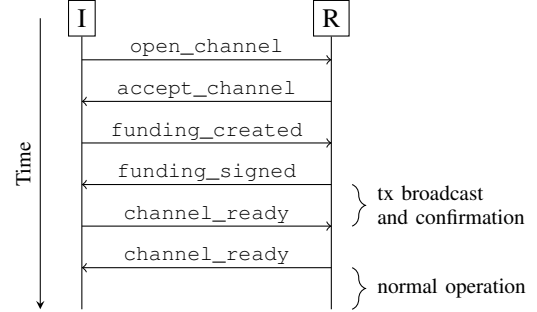


Figure 2. Sequence of messages exchanged to open a channel between an initiator (I) and a responder (R).

into blocks by miners and appended to the public blockchain once they have found a valid proof-of-work.

The blockchain stores a chain of transactions from one owner to the next where owners are identified by a public key. An address, which is derived from a public key using different formats, serves as a pseudonymous identifier of ownership and does not contain any binding to a real-world identity. A transaction reassigns ownership of bitcoins to one or more new addresses and has at least one *input* and *output*. Inputs reference the funds from previous transactions that are to be spent by the current transaction. Outputs determine the new owner of the transferred coins by locking them to the recipient’s address. Common address formats include Pay-to-Script-Hash (P2SH) [24], Pay-to-Witness-Script-Hash (P2WSH) [25], Pay-to-Witness-Public-Key-Hash (P2WPKH) [25] and Pay-to-Taproot (P2TR) [26]. Coins may also be locked to multiple addresses in an *m*-of-*n* multisignature output [27] which requires at least *m* of the given *n* signatures to unlock it.

Ownership of coins is defined entirely by the ability to satisfy the spending conditions encoded in the unspent transaction output (UTXO), i.e., control over coins is equivalent to possessing the cryptographic key required to unlock them. UTXOs are consumed by future transactions as inputs. The sum of all inputs in a transaction must be equal to or greater than the sum of all outputs. If greater, the difference is considered a fee for the miner that first includes the transaction into the blockchain.

Bitcoin’s design allows for transparency and verifiability without a central party as every transaction is stored on the blockchain. However, this same design severely limits the achievable transaction throughput [2] and user privacy [7], [8], giving rise to layer-2 solutions. Their objective is to reduce the number of on-chain transactions by providing alternative ways of transacting with bitcoins.

2.2. The Lightning network

The Lightning network (LN) is a PCN (payment channel network) over Bitcoin, i.e., a P2P network of bilateral off-chain payment channels [3], [4]. A payment channel represents a financial relationship between two nodes in which a fixed amount of funds (the *capacity*) is locked through a

transaction on the Bitcoin blockchain. An initiator I opens a channel by sending an `open_channel` message to the intended responder R as shown in Fig. 2. This message contains various parameters for the intended channel, e.g., the capacity and required number of confirmations, and is accepted by R through an `accept_channel` message. I then constructs the *funding transaction* and sends it in a `funding_created` message to R . This transaction is a regular Bitcoin transaction that creates a 2-of-2 multisig output on the blockchain. R provides their signature in the `funding_signed` message, making the transaction valid. I passes the raw transaction to a Bitcoin node it is connected to which then broadcasts it to the Bitcoin P2P network. Once the transaction has gotten the required number of confirmations, the two nodes each send a `channel_ready`, signifying that the channel is ready for operation.

During normal operation, an arbitrary number of payments can be made over the channel. Payments change how the channel’s capacity is divided (*balances*) between I and R and can be routed over multiple hops. Payments are secured via Hashed Timelock Contracts (HTLCs) which guarantee atomicity. An HTLC is a conditional transfer of funds that can be claimed by providing a preimage revealed by the payment’s beneficiary upon receipt of the payment. During channel establishment, each node specifies and broadcasts a set of parameters related to its routing policies that other nodes take into account when choosing a path to route their payment. We refer to relevant literature for further background on the LN beyond the scope of this work [3], [4], [28], [29].

2.3. Interaction between Bitcoin & LN

The two networks interact exclusively at three points during a channel’s life cycle [3], [4]: (1) wallet funding, (2) channel opening and (3) channel closing. Opening a channel requires that I has a Bitcoin wallet with sufficient funds. I therefore transfers funds from their Bitcoin wallet to their LN wallet’s integrated Bitcoin wallet in an ordinary Bitcoin transaction. The input for the channel’s funding transaction is the previous transaction’s output and its output is a 2-of-2 address shared by I and R (and potentially another output for change). When the channel is eventually closed, a *closing transaction* is broadcast to the Bitcoin network. The closing transaction redistributes the funds locked in the shared address back to each party’s LN wallet according to the final channel balance.

3. Attack preliminaries

3.1. Threat model

We assume a passive, network-level adversary, e.g., an AS, that has control of key network infrastructure between a victim node and the rest of the Bitcoin network. Multiple works have studied Bitcoin’s security in the presence of such an adversary [12], [14], [16] and it has also been applied to

the LN [17], [18] as well as other blockchain networks [30]. The adversary’s objective is to deanonymise a Bitcoin node by linking its IP address to an on-chain identity and ultimately to a real-world entity. This may be to facilitate financial surveillance, profiling or even censorship. As a network-level adversary, it is expected that they begin with the IP address associated with the victim node. We discuss the ethical considerations of the attack in Appendix A.

The victim is a regular Bitcoin node connecting to the network via TCP/IP and not via Tor [31] (discussed in Section 6.2). They use V2 P2P transport [23], which has been the default protocol since December 2023, and thereby encrypt all P2P messages. We require that the victim operates a Bitcoin and an LN node on the same host and that they each run on the respective default ports: `tcp/8333` and `tcp/9735` respectively. The adversary exploits the co-location to infer information about the Bitcoin node from observations in the LN and uses the ports to deduce that the encrypted communication is Bitcoin and Lightning. We also assume that the LN wallet was funded by the Bitcoin wallet on the same host. This is unlike funded from an exchange withdrawal, a mining reward or a self-transfer over the LN.

The attack requires that the adversary is able to identify the type of an application message in an encrypted TCP payload exchanged between the victim node and its peers in the LN. As demonstrated by prior work [17], [18], the payload length and the sequence of messages in the stream provide highly reliable side channels for message type identification regardless of fragmentation or TCP coalescing. The adversary also requires access to application-layer information from both networks in the later stages of the attack. In both cases, the data can be obtained either by operating a node or by querying public data sources.

3.2. Attack surface

In order to assess how realistic the assumption of co-location is, we implemented a crawler and conducted measurements in both networks. The crawler uses the Bitnodes API¹ to obtain a list of reachable Bitcoin nodes and our own LND node to collect a similar list for the LN. It computes the intersection of IP addresses announced in both networks before mapping them to ASs using the GeoLite2 database [32]. The measurements were run twice daily over eight weeks between 2 February 2026 and 3 April 2026.

The number of IP addresses announced in both overlays is shown in Fig. 3. With the exception of one measurement on 5 March 2026 during which our LND node temporarily fell out of sync, consistently around 220 addresses were active in both. This corresponds to 3% of reachable Bitcoin¹ and 3.5% of public LN² nodes during the measurement period on average. Yet, a not insignificant total of between 559 BTC and 685 BTC was locked in the involved channels which is about 10% of the LN’s total value. We also observed more than 400 unique addresses across all

1. <https://bitnodes.io/api> (currently offline as of May 2026)

2. <https://1ml.com/statistics>

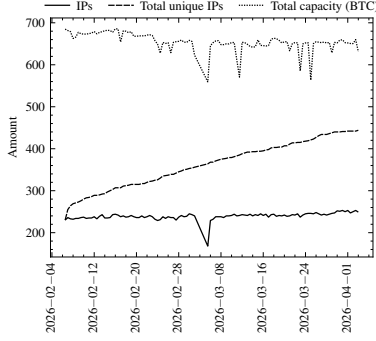


Figure 3. Number and channel capacity of co-located Bitcoin and LN nodes.

measurements. About 15% of them were present in every measurement while 49.5% were present in at least 50% of the measurements, suggesting that such setups are not just transient but persist over extended periods of time.

We also analysed the distribution of ASs among the overlapping addresses. As depicted in Fig. 9 in the appendix, more than 40% of the co-located channels are located within five ASs and with roughly 20% residing in AS 14618 alone. This highlights the known network-level centralisation of both networks [12]–[14], [17]–[19]. Co-location inference from shared IPs is susceptible to false positives in cloud ASs as one IP may represent multiple tenants. However, over 54% of the ASs in our data are residential networks where one IP typically maps to a single household endpoint. As a result, our measurements may overestimate the true extent of co-location, but they still substantiate our threat model and show that it describes an observable risk.

3.3. Traditional approach

An adversary can deanonymise Bitcoin nodes by operating a large number of nodes and establishing connections to as many nodes in the network as possible. When a node receives a transaction from one of its peers, it relays the transaction further after validating that it is eligible for broadcasting and the transaction trickles through the network that way. A super node, i.e., an adversarial node with connections to all active Bitcoin nodes, can therefore infer the likely originator of a transaction by observing which of its peers first announced it [5]–[7]. In addition to some nodes using Tor, Bitcoin introduced several measures to improve the privacy of transaction relay and limit inference of transaction origin. As of the release of Bitcoin Core 0.12.0, the trickle mechanism was replaced with a Poisson-distributed batching mechanism. Nodes transmit transactions to their peers with an independent, exponential delay of rate λ .

A study [7] on whether the new diffusion protocol actually improved Bitcoin’s anonymity properties estimated the likelihood of successful deanonymisation by a first-spy estimator. The estimator simply assumes that the transaction originated from the node that first reported it to the super node. The super node is passive and has $\theta > 1$ connections to each honest node in the network. Each honest node

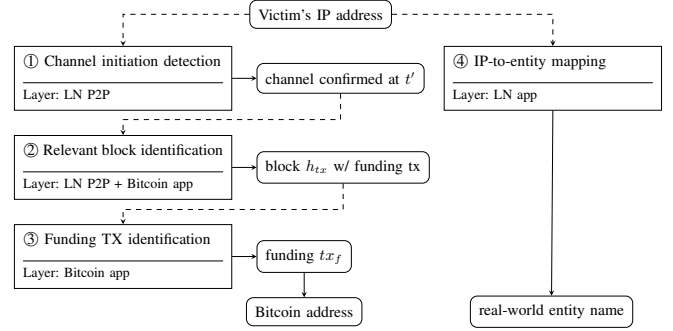


Figure 4. Steps of the deanonymisation attack showing the respective inputs (---), outputs (—) and layers of operation.

is assumed to have d peers. The study proved that the probability of such an estimator for $\theta = 1$ approaches $\frac{\log(d)}{d}$ asymptotically in d and approaches zero as d grows. The probability is increasing in θ (with diminishing returns for higher θ) because more connections per node mean that the adversary is more likely to hear from the legitimate source first. Simulations showed a probability of approximately 0.3 for $\theta = 1$ which rises up to about 0.8 as θ increases to 20.

Successful deanonymisation in this manner quickly becomes impractical as more and more connections to the network are required. Practicality aside, this does not lead to the full multi-layer identity sought after by our adversary as transactions alone do not contain mappings to real identities.

4. Deanonymisation of Bitcoin nodes

In the following, we present a deanonymisation attack by a network-level adversary on Bitcoin node that is co-located with a LN node. The attack comprises four primary steps and operates in multiple layers of the TCP/IP stack as well as of the Bitcoin stack as shown in Fig. 4. The objective of the first three is to identify an on-chain identity belonging to the victim node using information gathered from the LN’s P2P layer and the Bitcoin application layer. In the fourth step, which is independent of the others, the pair of IP address and on-chain address is complemented with a real-world entity’s name. In the future, under ‘gossip v2’, UTXOs will not be tied to channel announcements in the LN.³ This will improve nodes’ privacy by limiting the public information available for on-chain analysis. Our attack gains more relevance under gossip v2 as it establishes the UTXO to channel link without relying on public information.

The discovered tuple contains valuable information the adversary may use to bootstrap existing deanonymisation techniques such as address clustering [8]–[11], [33], [34]. This allows them to build a broader picture of the victim.

① Identification of channel initiation

The objective of this initial step is for the adversary to determine when the victim node opens a channel in the LN.

3. <https://github.com/lightning/bolts/pull/1059>

They passively monitor all LN P2P traffic between the victim and its peers until the victim initiates a channel with one of its peers. They exploit the fact that the types of encrypted LN messages transported in TCP payloads can be identified very reliably using the payloads' length [17], [18]. Specifically, the adversary inspects all payloads' lengths until they identify a `funding_created` message (cf. Fig. 2) from the victim to some peer and takes note of the current time t . According to the specification [35], this message contains 96 bytes, but our empirical analysis shows it is usually transmitted in a 148-byte payload by LND nodes and in a 166-byte payload by CLN nodes. However, as this particular payload length is not unique to the `funding_created` message, inferring the type based on just the length will inevitably lead to false positives. Much to the adversary's advantage, the sequence of payloads transmitted between peers can also be used to infer message types. As we show in Section 5, message type identification is highly accurate when combined with the length of surrounding packets. In this case, the adversary confirms that a payload with a length of 148 (LND) or 166 (CLN) bytes was indeed a `funding_created` message if the length of the next payload the victim receives from the peer is 114 (LND) or 132 (CLN) bytes, i.e., a `funding_signed` message. Identifying the `funding_signed` message also lets the adversary know that the channel opening was initiated successfully and they then proceed to the next step.

② Identification of relevant block

The objective of this step is to identify the block in the Bitcoin blockchain containing the funding transaction tx_f of the observed channel initiation. The adversary continues monitoring LN P2P traffic but can now focus on the connection of interest exclusively. When a funding transaction receives the required number of confirmations, the initiator sends a `channel_ready` message (cf. Fig. 2) to the responder immediately [35]. By default, LND scales the required number between three and six linear to the channel size while CLN requires three confirmations. As the `channel_ready` message's length is deterministic, 83 bytes under LND and 111 bytes under CLN, the adversary uses its detection as a signal that tx_f has been included in a block and confirmed at the current time t' .

In the final part of this step, the adversary operates at the Bitcoin application layer and needs access to a copy of the blockchain, e.g., by either running their own node or querying a third-party explorer. The adversary uses the timestamps t and t' and the block height h' at t' to identify the block at height h_{tx} with tx_f . They scan the blockchain downward from h' until they find the first block at height h whose timestamp is less than or equal to t . Assuming tx_f was included in the next immediate block after broadcast, the adversary concludes that the number of confirmations it required was $h' - h$ and tx_f must be in block $h_{tx} = h + 1$. We discuss the limitations of this assumption in Section 5.

③ Identification of LN funding transaction

At this point, the adversary knows the block containing the LN funding transaction, but still needs to identify the specific transaction tx_f in the blockchain so as to determine an on-chain address belonging to the victim's Bitcoin wallet. tx_f 's input is the output o_b of a previous UTXO that was used to fund the LN wallet. Under our threat model, the victim node's Bitcoin wallet directly funded the LN node's wallet. This means that the address encoded in o_b 's `scriptPubkey` field is attributable to the victim node's Bitcoin wallet and can therefore be linked back to the victim. If tx_f has multiple inputs, the respective addresses obtained from their outputs must also belong to the same wallet under the common input ownership assumption [8].

We make use of the fact that LN transactions exhibit structural patterns that distinguish them from other Bitcoin transactions to identify the specific transaction. Kappos et al. introduced a property heuristic that aggregates common characteristics of such transactions and used it to identify private LN channels [22]. However, as we show in Section 5, the heuristic is no longer reliable due to protocol developments since its publication in 2021. We therefore revise it to reflect the current state of the LN and present the properties P3 to P5 in addition to the original properties P1 and P2. We note that the properties are also of independent interest for other research such as identifying private channels.

P1 The transaction contains at least one P2WSH output.

P2 One of the transaction's P2WSH output addresses appears at most once as an input and at most once as an output in the blockchain. If the channel is still open, the address can only appear once as an output and never as an input.

P3 The transaction has either $n \leq 2$ outputs, or if $n > 2$, exactly one output is P2TR and the remaining $n - 1$ are P2WSH. The latter accounts for batch channel openings which have become increasingly common.

P4 The transaction was funded by either P2WPKH or P2TR addresses. In contrast to the original heuristic, which allowed P2SH or P2WPKH, we exclude P2SH as it is now considered legacy.

P5 We impose an upper bound of 10 BTC on the output. The original 0.167 BTC limit became obsolete with the introduction of 'wumbo' channels [35] which technically allow unlimited capacities. As LND, the dominant client in the LN, sets a default limit of 10 BTC⁴, we expect that very few channels, if any, will be bigger.

We also point out the following two properties that are potentially weaker indicators and not included in the heuristic. We hypothesise that they will become stronger indicators as adoption of Taproot [26] increases.

(P6) The transaction contains exactly one P2TR output. This would replace P1 as support for Taproot has been available since mid 2022.

(P7) All transaction outputs are either P2WSH or P2TR. There is no comparable property in [22].

4. <https://github.com/lightningnetwork/lnd/blob/v0.20.0-beta/config.go#L1063>

The adversary applies this heuristic to all transactions in the block and obtains a set of candidate transactions in the block. A candidate transaction contains the transaction ID and its inputs. The input addresses belong to the Bitcoin wallet that funded the transaction, i.e., the LN node’s wallet. Ideally, this set contains exactly one element, however, it is possible for some false positives or false negatives to occur. One way the of validating a candidate transaction if the node is running LND is by using the number of confirmations the transaction required. Applying the inverted LND scaling function bounds the channel capacity to a specific range which can then be compared to the value of the outputs in the identified transaction. Another way is by waiting for the channel to be announced to the network. In that case the adversary could opt to skip the heuristic-based identification in entirety and wait for the announcement. However, this only works for public channels as private channels are not announced to the network.

④ Mapping on-chain identity to real-world entity

Having identified tx_f and its associated on-chain address, the adversary now complements the pair of IP address and Bitcoin address with a real-world entity. Such may not always be known to the adversary, e.g., when the infrastructure is provisioned with a pseudoname and paid for with cryptocurrency. Linking on-chain identities to real-world entities has been addressed in prior work, e.g., with a re-identification attack after making purchases from merchants [8]. [6] also proposed a similar approach but requires that the adversary maintains many connections to multiple nodes in the Bitcoin network.

In contrast, our approach does not require monetary resources. We exploit the fact that LN nodes announce a self-chosen UTF-8 string of up to 32 bytes known as an ‘alias’ [36]. The alias serves as a human-readable identifier in place of a public key. Using the alias to infer the real-world identity behind the two co-located nodes requires access to LN application data which is achievable by running a node or querying third-party services. The latter is only possible if the victim is involved in at least one public channel as it will otherwise not be announced to the network. As depicted in Fig. 4, this step can be performed concurrently with ② and ③ as the only prerequisite is knowledge of the victim’s IP address. As aliases are self-reported, they may be uninformative or deliberately false. This step of the attack is therefore best-effort and may fail in such cases.

5. Evaluation

In this section, we evaluate ① to ④ and demonstrate that the attack is viable in practice. Subsequent mentions of the measurement period refer to the eight-week window between 2 February 2026 and 3 April 2026 used in Section 3.2. All code is available as detailed in Appendix B.

To implement and evaluate steps ① to ③ of the attack in a controlled environment, we extended a Bitcoin testbed [37] with support for the LN. Our extensions cover the full life

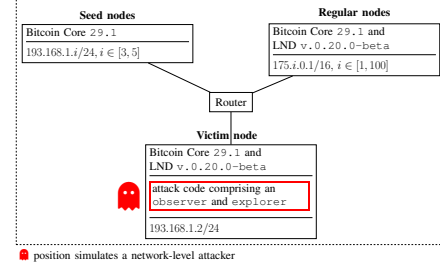


Figure 5. The services running on each host and the attacker’s position in the private network created for our evaluation.

cycle from automatic wallet initialisation to the establishment of payment channels. We used a snapshot of the LN channel graph from 16 February 2026 comprising 13,265 nodes and 44,699 channels to bootstrap a realistic topology. We applied a random walk starting from the node ‘LNBIG [Hub-3]’, the most influential co-located node identified in our measurements, to sample a small sub-graph of the network. We included all edges between each pair of visited nodes to retain the occasional multi-channel structure. The resulting sub-graph provides a representative topology as it preserves local connectivity and node centrality properties. We subsequently recreated the sub-graph in the testbed as shown in Fig. 5 on an Ubuntu 24.04 LTS server with 12 cores and 32 GiB RAM. The network consisted of 103 hosts in total and was configured as follows:

- 3 Bitcoin Core nodes serving purely as bootstrap nodes;
- 100 hosts in distinct subnets running Bitcoin Core 29.1 and LND v.0.20.0-beta (the latest releases at the time of writing). They represent regular Bitcoin and LN networks; and
- 1 Ubuntu 24.04 Linux host with our implementation of the attack as well as a Bitcoin Core 29.1 and LND v.0.20.0-beta client (the victim node). An iptables rule set directs all packets on TCP port tcp/9735 to the netfilter queue that is attached to the attack code. The victim and the attacker were placed in the same container for the sake of implementation simplicity. The setup is representative of an in-path attacker as traffic must flow through them, i.e., the adversary’s position is different, but the traffic they can intercept and their capabilities remain the same.

All Bitcoin Core and LND instances were started with the default settings. We configured a global block generation time of ten minutes and generated a transaction every 1.92 seconds (corresponding to a daily average of roughly 45k transactions) to emulate realistic Bitcoin network conditions. Channels with uniformly-random capacities between 0.01 BTC and 1 BTC were then opened following an exponential distribution with a rate parameter of 6892 and a maximum interval between events of 12.9 hours. The values are the result of our analysis of channel open frequencies in the public network during the measurement period. We also generated payments of random amounts at random intervals throughout the emulation which ran for 96 hours.

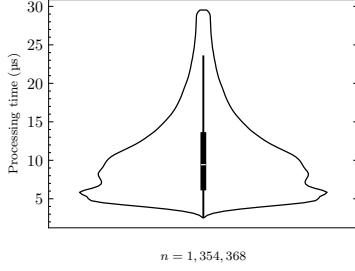


Figure 6. Attack code’s processing times in microseconds for n packets.

5.1. How well does the adversary detect LN events?

Methodology. The `observer` component of the attack code inspects the lengths of all TCP payloads and infers the message type accordingly. Upon detection of a `funding_created` message at time t from the victim to some peer p , it continues monitoring payload lengths until a `channel_ready` message to p at time t' is sent. It returns t and t' and continues the monitoring loop. No packets are modified and they are all returned to the kernel with the `ACCEPT` verdict. We evaluated how accurately the attack code identifies the relevant events by cross-referencing the observer’s logs with the victim’s LND logs and computing precision and recall for each message type.

LN message type classification is highly accurate.

The attack code achieved a precision and recall of 1.0 for the `funding_created` (used in ①) message and 0.86 for the `channel_ready` message (used in ②). All but one of the relevant messages were identified correctly which is due to a message of another type being misclassified and could be avoided by a more complex state machine. This result is consistent with prior work on message type classification in the LN [17], [18] and further highlights the need for effective countermeasures against traffic analysis.

The latency overhead is negligible. Fig. 6 shows how long the attack code processed each packet during the entire emulation. This is the time from first accessing a packet in the `netfilter` queue to the the packet being returned to the kernel after issuing the `ACCEPT` verdict. Our implementation is very efficient and re-queues a packet within a median time of $9.4\mu\text{s}$ and mean time of $10.7\mu\text{s}$. We believe that the increased latency does not affect the viability of the attack negatively as delays in the lower end of tens of microseconds are highly likely to go unnoticed.

5.2. How well does the adversary determine relevant blocks?

Methodology. After completion of the monitoring phase, the adversary must next identify the block containing the funding transaction to complete ②. We implemented an explorer that queries a local Bitcoin node for transactions using the `getrawtransaction` RPC. It calculates the block height h_{tx} of the block containing the funding transaction based on the timestamps t and t' reported by the

observer and the block height h' at t' as described in Section 4. We compared the concluded block height h_{tx} against the ground truth obtained via LND’s `listchannels` RPC on the victim node and computed precision and recall.

Block identification is accurate despite varying confirmation requirements. Similar to the previous results, the attack code achieved precision and recall of 1.0. This is because the `channel_ready` message is sent immediately when the target number of confirmations is reached and thereby makes the timing signal unambiguous. Consequently, requiring a non-default number of confirmations offers no meaningful protection. However, a sufficiently large delay between reaching the target height and sending the `channel_ready` message would disrupt the inference.

Our formula for block inference rests on the assumption that the funding transaction is included in the immediate next block after broadcast. As this may not always hold under dynamic mempool and fee conditions, we considered evaluating it on the test networks using our own nodes. However, as per numerous block explorers, transaction volumes on both `testnet3` (now deprecated) and `testnet4` are so low that most blocks contain only a handful of transactions, meaning that the assumption would hold there too. Running the attack on `mainnet` was not a viable option due to the financial resources required to open payment channels. We performed a preliminary check in Appendix C which suggests that funding transactions are confirmed quickly.

5.3. How identifiable are LN funding transactions?

Methodology. We analysed all public LN funding transactions to validate the proposed properties P1 to P5. Their validity is a prerequisite for the success of ③. We used an LND node to retrieve the channel points (transaction IDs and output indices of the funding transactions) of all new channels during our measurement period. We then implemented the properties in a Rust program that takes a list of transaction IDs, queries Blockstream’s Esplora HTTP API⁵ for transaction data and analyses the various properties. We recorded 1400 transactions during the eight weeks and report the analysis results in Table 1. It shows the percentage of transactions satisfying each property.

We also evaluated how well LN transactions can be differentiated from other Bitcoin transactions. We selected 50 random blocks within the measurement window, filtered known funding transactions and analysed the remaining 55,282 transactions in the same manner. We limited the analysis to a subset as analysing all blocks in the window was infeasible given the volume of transactions involved, particularly due to rate limits imposed by the API.

The structural properties of LN funding transactions are well-described by P1 to P5. Our heuristic retains the original P1 and P2 properties and they continue to apply to all funding transactions: P2WSH channels are still the most common and payment channels are not reusable. The revised P3 to P5 properties achieve consistently higher

5. <https://github.com/Blockstream/esplora/blob/master/API.md>

Table 1. PROPERTIES OF BITCOIN TRANSACTIONS BETWEEN 2 FEBRUARY 2026 AND 3 APRIL 2026 (BLOCKS 935,175 TO 943,421). PROPERTIES INCLUDED IN OUR HEURISTIC ARE MARKED IN **BOLD**.

	Retrieved funding transactions		Other transactions	
	Original	Revised	Original	Revised
P1	100%	-	7.46%	-
P2	100%	-	4.65%	-
P3	85.79%	99.0%	91.7%	91.71%
P4	35.93%	99.57%	77.87%	84.02%
P5	64.5%	100%	6.42%	7.43%
(P6)	85.64%	81.29%	6.58%	7.4%
(P7)	-	84.43%	-	8.43%

scores than the original ones. The original P4 and P5 properties perform particularly poorly which is to be expected given the legacy status of P2SH and the introduction of wumbo channels. Although (P6) and (P7) applied to more than 81% of the transactions, we do not consider this level of certainty high enough to include in the heuristic. In future work, we plan to monitor their performance as well as apply the heuristic to enumerate private channels in the LN.

The percentage of non-LN transactions meeting all requirements is very low. The proportion of transactions satisfying P3 is relatively high and is to be expected: standard Bitcoin transactions involve a payment to a recipient and a change output which results in at most two outputs. Similarly, the percentage of transactions meeting P4 is high. In general, the fraction of transactions satisfying all criteria simultaneously is small and P1, P2 and P5 are the main discriminators. Those that do may in fact be funding transactions for private LN channels which are no different from other funding transactions for public channels.

5.4. How well does the adversary identify funding transactions?

Methodology. During the initialisation of the topology, when funding the LN wallets, we introduced variation in the address types to reflect a realistic setting where identification is not trivial. Each node selected one of the three address types supported by LND (P2WPKH, P2TR and NP2WKH), with the first two jointly assigned a probability of 0.9. This is the address that is used to fund the opening transaction and is therefore the one P4 applies to. We implemented the properties P1 to P5 in the `explorer` and applied them to every transaction in the previously-determined block to obtain the set of candidate transactions. We then compared the set with the ground truth obtained from the `listchannels` RPC’s output and computed precision and recall. The results are the combined outcome of ① to ③ in a complete workflow.

All relevant funding transactions are identified. We achieved precision of 0.85 and recall of 1.0. The perfect recall means that every real funding transaction for channels initiated by victim node was identified correctly. The lower

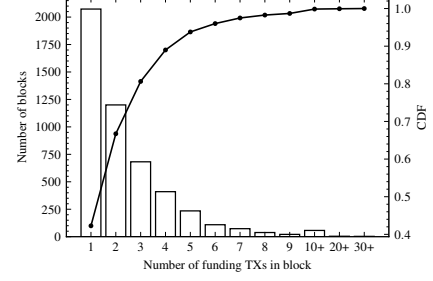


Figure 7. Distribution of known funding transactions in the same block.

precision is because of blocks containing multiple funding transactions which are all (rightfully) flagged as candidates. The high recall means that the relevant funding transaction is never missed and is identified as a candidate. The outcome of this step is either a single identity tuple or a set of tuples that is significantly smaller compared to a naive search.

To assess how probable it is that the candidate transaction is unique, we examined how frequently multiple funding transactions are included in the same block on mainnet. We extracted the block heights of the funding transactions gathered in Section 5.3, counted the number of occurrences at the same block height and present the distribution in Fig. 7. The distribution is heavily skewed towards small values and indicates that several funding transactions within the same block are not particularly common. The candidate transaction is unique in 42% of the cases while the set of candidates contains at most five transactions 90% of the time. The small size means that further processing of all obtained transactions, e.g., address clustering, is practically feasible. Furthermore, the candidate transactions can be validated once the channel has been announced.

5.5. How informative are LN nodes’ aliases?

Methodology. To determine how revealing LN node aliases are, we recorded the aliases of all co-located nodes identified in each of the crawls throughout the measurement period. This yielded a set of 347 unique aliases. The number is slightly lower than the number of unique addresses in Fig. 3 as not all nodes announced an alias. We examined each alias manually and assigned it to one of five categories:

- 1) aliases matching a hexadecimal public key pattern are labelled *public key*;
- 2) aliases identifiable as a business, service provider, exchange, financial institution or other formal entity are labelled *institution*;
- 3) aliases identifiable as a specific person are labelled *individual*. Such aliases contain valid references to a personal domain, an email address or an online handle;
- 4) aliases with a reference to a potential identity that we however could not confirm, e.g., due to ambiguity, are labelled *uncertain*; and
- 5) all remaining aliases are labelled *unknown*. We could not infer the identity from these aliases.

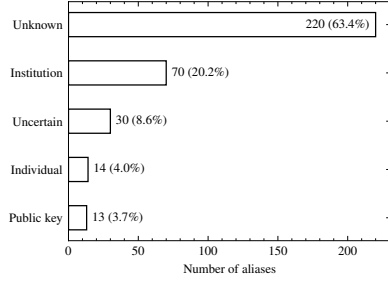


Figure 8. Distribution of co-located aliases in various categories.

We initially tried to automate the classification using regular expressions and substring matching but found it difficult to generalise due to the highly heterogeneous nature of the aliases. We searched for references on the web for aliases that we did not immediately recognise as belonging to one of the categories before making the final assignment. Furthermore, we confirmed that each alias assigned to the *institution* and *individual* categories has a plausible connection to Bitcoin/LN, e.g., through their website or public forums.

The majority of informative aliases belong to institutional entities. We generally find that a significant portion of the aliases do not reveal the operator’s identity as visible in Fig. 8. The figure shows the percentage of aliases assigned to each of the categories. Of the 347 examined aliases, 20.2% were attributed to the *institution* category. This is likely because businesses and other service providers have a financial incentive to be easily recognisable in the network. While not entirely absent from the dataset, *individual* aliases were rare and account for only 4% of the aliases. This indicates that personal node operators prefer to remain pseudonymous in the network. The unattributed majority showed recurring patterns such as following CLN’s default naming conventions, containing references to geographic locations and ideological pseudonyms without any real reference to an identity. These results are a positive finding for the LN. They show that operators do not compromise their pseudonymity with trivially attributable aliases. This means that ④ is more likely to succeed for larger entities as opposed to individuals.

During the analysis, we observed that a fair share of the aliases contain references to a specific implementation. Future work could explore the feasibility of inferring the implementation used by a node based on the alias and contrast the results to existing approaches [18], [38]–[40].

6. Countermeasures

Our attack poses a practical threat to the privacy of Bitcoin users without exploiting a specific vulnerability in either protocol. It combines various weaknesses across multiple layers which makes it challenging to identify a single effective countermeasure. In what follows, we discuss different measures that can be taken by both protocol developers and node operators to mitigate the presented attack and network-level attacks in general. We first evaluate the

Table 2. FEATURES EXTRACTED FROM EACH TRACE FOR TRAINING.

Category	Feature	Description
Volume	<code>duration_sec</code>	duration of observation window
	<code>packet_count</code>	total number of packets
	<code>total_bytes</code>	sum of exchanged data
Packet sizes	<code>median_pkt_size</code>	50 th percentile of lengths
	<code>p_i, i ∈ {10, 25, 75, 90}</code>	<i>i</i> th percentile of lengths
	<code>size_variance</code>	distribution of lengths
Directionality	<code>up_down_ratio</code>	ratio of sent and received bytes
Small packets	<code>pct_pkts_leq_100</code>	% of ≤ 100-byte packets
	<code>tiny_pkts_min</code>	≤ 66-byte packets/minute

impact of using random ports, which is a frequently-cited defence, and then discuss further measures.

6.1. (In)effectiveness of port randomisation

The adversary relies on both applications running on the respective default TCP ports to identify the protocol. At the time of writing, tcp/8333 was used by over 95% of the reachable nodes in the Bitcoin mainnet (according to the Bitnodes crawler) while tcp/9735 was used by 89% of the LN nodes in our snapshot of the mainnet. Prior work [12], [17] has recommended the use of random ports as a measure against network-level attacks and it has even been raised in an issue in Bitcoin Core [41]. Port randomisation refers to either peers communicating over a new, random port after negotiation or listening for connections on a random port by default. While the LN supports port advertisement via SRV records [42], listening on a non-default in Bitcoin puts a node at a disadvantage as the DNS bootstrapping seeds do not include port information. As result, this measure has broader implications involving protocol-level changes and should only be pursued if the expected gain is sufficiently high. We therefore analyse the impact of port randomisation and investigate if, and how fast, a TCP stream can be determined to be carrying Bitcoin or LN traffic.

Methodology. Informed by the state of the art in detection and classification of encrypted traffic [43], [44], we apply supervised ML, specifically a random forest (RF) [45] classifier. RF is an ensemble of decision trees and is well-suited for this study due to its robustness to noise and heterogeneous features without hyperparameter tuning. We implemented and trained a binary classifier which distinguishes Bitcoin traffic from non-Bitcoin traffic (equivalently for LN traffic) on the 12 features listed in Table 2. Our selection of features was guided by related work and captures coarse trace-level properties of TCP streams which remain observable after encryption, i.e., size and direction.

The model consists of 100 decision trees and was trained and evaluated using a stratified 80/20 train-test split which preserves the class distribution across both sets. We evaluate its performance using various metrics such as precision, recall (equivalent to the true positive rate (TPR)) and the false positive rate (FPR). We extracted the features from sliding observation windows of increasing duration to assess how classification evolves as traffic accumulates.

Table 3. CLASSIFICATION SCORES ACROSS VARIOUS OBSERVATION WINDOWS (SECONDS) FOR BITCOIN AND LN TRAFFIC.

	Scenario	Window (secs)	Precision	Recall (TPR)	FPR
Bitcoin	P2P	≤ 0.05	0.922	0.826	0.015
		$[0.05, 1]$	0.985	0.975	0.002
		$[1, 10]$	1.0	1.0	0.0
	Services	≤ 0.05	0.926	0.835	0.02
		$[0.05, 1]$	0.984	0.978	0.003
		$[1, 10]$	0.999	0.999	0.0
LN	P2P	≤ 0.05	0.983	0.994	0.0001
		$[0.05, 1]$	0.999	0.999	0.0
		$[1, 10]$	1.0	0.999	0.0
	Services	≤ 0.05	0.995	0.993	0.0001
		$[0.05, 1]$	0.999	0.999	0.0
		$[1, 10]$	1.0	0.999	0.0

Dataset. We surveyed modern P2P networks and identified three primary classes of transport-layer protocols: unencrypted, custom encryption and libp2p (cf. Table 4 in the appendix for a summary). We compiled a dataset of network traces from representative networks in the encrypted and libp2p categories, i.e., Bitcoin, Ethereum [46], IPFS [47] and LN traffic from nodes running the respective latest releases. Despite our best efforts, we were unable to source LN traffic from real routing nodes and deploying our own public routing nodes was not feasible due to the required monetary investment. Instead, we captured traffic from three nodes during the previous evaluation – the victim and two randomly selected nodes – all of which participated in payment routing. As these traces lack realistic network latencies, we intentionally excluded timing-related features from the classifier to avoid introducing features that would not generalise to real-world traffic. We also incorporated non-P2P over TCP traffic into our dataset to model the scenario in which typical services are running alongside a single node. In this case, an adversary can readily distinguish that the traffic corresponds to a P2P network but cannot trivially infer the underlying protocol. We applied a preprocessing step to transform the input data into a unified CSV structure. We then extracted the features from the CSV and fed them to the classifier. An overview of the dataset is provided in Table 5 in the appendix; availability of the dataset and the code are discussed in Appendix B.

Table 3 shows that Bitcoin and LN traffic can be detected with high accuracy even in short observation windows. For windows between 50ms and 1s, both protocols are classified almost perfectly while windows between 1s and 10s result in (near-)perfect scores across all metrics. Performance improves substantially as the observation time increases for Bitcoin. Precision and recall rise from 0.922 and 0.826 respectively in the sub-50ms window to 1.0 in the 1s to 10s window. LN traffic is highly detectable from the onset which we attribute to two factors. The deterministic nature of the protocol produces highly consistent traffic patterns which already makes basic fingerprinting very reliable. Further, our LN traces originate from a controlled testbed which may reduce noise and overestimate performance. We intend to validate these results on open-world LN traffic,

contingent on data availability. The classifier generalises well across various connection types which leads to negligible differences between the P2P and services scenarios.

Overall, the results show that Bitcoin and LN traffic can be detected with near-certainty after just a few seconds. Hence, port randomisation is unlikely to provide significant protection against network-level adversaries. As it stands, it would merely add a level of complexity in that the adversary must first identify whether a given stream is relevant.

6.2. Further measures

The primary assumption underlying this attack is that of Bitcoin and LN node co-location. LN clients implicitly assume that both nodes are on the same host and their interaction is designed around local, trusted communication channels as opposed to encrypted, remote connections. While co-location is understandably common due to simplicity, we encourage operators to run their nodes on separate hosts and establish a secure remote connection between them using encrypted channels such as SSH tunnels. The overhead is negligible as this is a one-time setup effort.

We contend that the attack would still be practical even in the absence of co-location albeit more challenging for the adversary as they must first identify a Lightning node under the control of the same entity as the victim node. Prior work [21] demonstrated that Bitcoin and LN nodes can be linked based on Bitcoin address usage patterns. The main measure against such linkage is avoiding re-usage of Bitcoin addresses. When a Bitcoin address is reused across transactions, it is trivial to link those transactions to the same entity without relying on any heuristic. Not only would using a different address for every transaction make linking the two nodes impractical, it would also reduce the potential impact of the attacks bootstrapped by the tuple of identities. Earlier work even found that address clustering was highly effective because address reuse is so prevalent [48]. Furthermore, the adoption of mixing techniques limits what the adversary can meaningfully achieve with the obtained tuple. CoinJoin [49] combines inputs from multiple users into a single transaction with uniform outputs and thus makes it difficult to match inputs to the corresponding outputs. CoinShuffle [50] is similar but allows users to shuffle their outputs in a decentralised manner. However, operators should be aware of the fact that the privacy gained from mixing can be significantly reduced by their post-mix behaviour, e.g., combining mixed outputs with unmixed ones [51].

The reason nodes using Tor are excluded from our threat model is that it provides basic resistance to traffic analysis by splitting data into fixed-size cells. This makes it considerably more difficult for the adversary to fingerprint the communication efficiently and effectively. In general, we caution against an over reliance on Tor but recognise that it can contribute to the broader defence strategy meaningfully. In addition to the expected latency overhead, using Tor makes nodes susceptible to other potential threats [52]–[54]. Prior work even showed that using Bitcoin over Tor creates a new attack vector [55]. Instead, both the Bitcoin and the

LN protocols should provide better anonymity by including traffic shaping. The need for message type obfuscation in the LN is apparent as has been shown in prior work [17], [18] and further highlighted in this work. There was a pull request⁶ proposing message padding that has since been closed. Regardless, an adequate defence against traffic analysis must obfuscate both the length and sequence of messages as either side channel remains useful on its own [18]. Such should make correlating cross-layer activity based on LN traffic less feasible and perhaps even impractical. In this case, we contend that the adversary could resort to analysis of Bitcoin traffic. Although Bitcoin’s P2P communication is now encrypted, the lack of traffic shaping mechanisms suggests that some information must still be deducible from the traffic patterns. The inference may not be as straightforward as Bitcoin is significantly less deterministic, but we expect it to nonetheless yield useful information for an adversary.

The need for traffic shaping as a fundamental, long-term measure is emphasised by our results that demonstrate the limited utility of port randomisation. Defining traffic shaping mechanisms for either protocol is beyond the scope of this work as doing so requires careful consideration of the trade-off between privacy/security goals and usability. Padding and delays introduce bandwidth and latency overhead that may interfere with the user experience or conflict with other protocol goals. The feature importance analysis for protocol detection shows that the classifier relies primarily on the rate of packets smaller than or equal to 66 bytes per minute. Hence, traffic shaping mechanisms could start by obscuring the size and/or frequency of these small control packets.

7. Related Work

Network attacks on Bitcoin have been the subject of several works prior to this one and are mostly concerned with partitioning the network [12], [14], [15], delaying block propagation [12] or transaction origin inference [16]. The practicality of these attacks has since been limited by the recent introduction of encrypted communication [23]. More recently, there has also been research on network attacks in the LN [17], [18]. While this work shares a common threat model with all of them, it is the only one that examines how weaknesses in one overlay network can be exploited in the other. Some of these works [12]–[14], [16] also performed measurements of the Bitcoin network and exposed its AS-level centralisation. Earlier work also used properties such as latency to infer its topology [56], [57] or to associate Bitcoin addresses with IP addresses [5]–[7], [58].

Research on cross-layer interactions between Bitcoin and the LN is limited. [21] presents on-chain clustering heuristics that group Bitcoin addresses based on their interaction patterns with the LN. They also link LN nodes to Bitcoin addresses based on address reuse and common funding addresses. The latter is complementary to our work and is an alternative way of establishing the cross-layer node link that we infer from co-location. [59] proposes heuristics

for identifying LN transactions in the blockchain. Their heuristic is different from the property heuristic in [22], which we revisit in this work, in that funding transactions are identified after the channel has been closed. Our method works on open and closed channels and does not require waiting for a channel to be settled on-chain. [48] presented a denial of service attack which floods the blockchain with transactions by simultaneously forcing the closure of many LN channels. Our attack follows an entirely different objective and is performed by a passive network-level adversary.

Beyond these works, deanonymisation in Bitcoin has received significant attention with address clustering being of particular interest [8]–[11], [33], [48]. Our work strengthens the utility of address clustering by providing the linkage between IP address, Bitcoin address and real-world entity. Transaction graph analysis has also been studied extensively [10], [34], [60]. We refer to surveys [61], [62] for overviews on attacks in Bitcoin and cryptocurrencies.

There is also a long line of work on traffic analysis in various types of encrypted networks including anonymisation networks such as Tor and various protocols used over the Internet. The latter is most relevant for our work as they study whether encrypted protocols can be detected based on network-layer fingerprints, e.g., [43], [44], [63], [64]. The common use of traditional ML and the applied metrics for evaluation guided our chosen approach. During the course of this work, we became aware of an early-stage project⁷ on the detection of Bitcoin. Our study is more extensive as it includes the LN protocol and considers other P2P networks.

8. Conclusion and Future Work

This work investigated Bitcoin and LN co-location and presented the first network-level, cross-layer study of Bitcoin and the LN. We described a network-level deanonymisation attack on a Bitcoin node co-located with an LN node. Based on comprehensive measurement data, we confirmed that co-location is a reality and the risk of such attacks is not just theoretical. Our attack gathers information about a Bitcoin node from multiple layers of the Bitcoin stack and the TCP/IP stack to complete the tuple of IP address, Bitcoin address and real-world entity. Our evaluation demonstrated that the heuristics used in the attack provide accurate results such that the adversary can deanonymise the victim node successfully. Much to the relief of individual node operators, we found that uncovering the real-world identity behind commercial/organisational nodes is significantly more likely. Lastly, we proposed countermeasures that can be taken by protocol developers and operators in response to this work. We showed that port randomisation is not a sufficient mitigation for neither networks as the resulting traffic can be detected reliably. However, this result should be validated for the LN in future work using real-world traces. In addition to extending the testbed with realistic network conditions, the identified protocol-level measures present interesting avenues for future work.

6. <https://github.com/lightning/bolts/pull/1304>

7. <https://github.com/RazorBest/bip324-traffic-analysis>

Acknowledgements

We thank b10c and Leonhard Balduf for contributing packet captures to our dataset of network traces. We also thank Sebastian Henningsen, Mikolai Gütschow and the anonymous reviewers for their valuable feedback.

References

- [1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: <https://nakamotoinstitute.org/bitcoin/>
- [2] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. E. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer, “On scaling decentralized blockchains - (A position paper),” in *Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers*, ser. Lecture Notes in Computer Science. Springer, 2016, pp. 106–125.
- [3] J. Poon and T. Dryja, “The bitcoin lightning network: Scalable off-chain instant payments,” Jan 2016. [Online]. Available: <https://lightning.network/lightning-network-paper.pdf>
- [4] “Bolt: Basis of lightning technology (lightning network specifications).” [Online]. Available: <https://github.com/lightning/bolts>
- [5] P. Koshy, D. Koshy, and P. D. McDaniel, “An analysis of anonymity in bitcoin using P2P network traffic,” in *Financial Cryptography and Data Security - 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 8437. Springer, 2014, pp. 469–485.
- [6] A. Biryukov, D. Khovratovich, and I. Pustogarov, “Deanonymisation of clients in bitcoin P2P network,” in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*. ACM, 2014, pp. 15–29.
- [7] G. Fanti and P. Viswanath, “Deanonymization in the bitcoin P2P network,” in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017*, pp. 1364–1373. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/hash/6c3cf77d52820cd0fe646d38bc2145ca-Abstract.html>
- [8] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, “A fistful of bitcoins: characterizing payments among men with no names,” in *Proceedings of the 2013 conference on Internet measurement conference*, 2013, pp. 127–140.
- [9] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, “Evaluating user privacy in bitcoin,” in *Financial Cryptography and Data Security*. Springer Berlin Heidelberg, 2013, pp. 34–51.
- [10] D. Ron and A. Shamir, “Quantitative analysis of the full bitcoin transaction graph,” in *Financial Cryptography and Data Security*. Springer Berlin Heidelberg, 2013, pp. 6–24.
- [11] M. Möser and A. Narayanan, “Resurrecting address clustering in bitcoin,” in *Financial Cryptography and Data Security*. Springer International Publishing, 2022, pp. 386–403.
- [12] M. Apostolaki, A. Zohar, and L. Vanbever, “Hijacking bitcoin: Routing attacks on cryptocurrencies,” in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. IEEE Computer Society, 2017, pp. 375–392.
- [13] S. B. Mariem, P. Casas, M. Romiti, B. Donnet, R. Stütz, and B. Haslhofer, “All that glitters is not bitcoin - unveiling the centralized nature of the BTC (IP) network,” in *NOMS 2020 - IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, April 20-24, 2020*. IEEE, 2020, pp. 1–9.
- [14] M. Tran, I. Choi, G. J. Moon, A. V. Vu, and M. S. Kang, “A stealthier partitioning attack against bitcoin peer-to-peer network,” in *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 2020, pp. 894–909.
- [15] W. Fan, S. Chang, X. Zhou, and S. Xu, “Conman: A connection manipulation-based attack against bitcoin networking,” in *IEEE Conference on Communications and Network Security, CNS 2021, Tempe, AZ, USA, October 4-6, 2021*. IEEE, 2021, pp. 101–109.
- [16] M. Apostolaki, C. Maire, and L. Vanbever, “Perimeter: A network-layer attack on the anonymity of cryptocurrencies,” in *Financial Cryptography and Data Security - 25th International Conference, FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part I*, ser. Lecture Notes in Computer Science, vol. 12674. Springer, 2021, pp. 147–166.
- [17] T. von Arx, M. Tran, and L. Vanbever, “Revelio: A network-level privacy attack in the lightning network,” in *8th IEEE European Symposium on Security and Privacy, EuroS&P 2023, Delft, Netherlands, July 3-7, 2023*. IEEE, 2023, pp. 942–957.
- [18] C. Ndolo and F. Tschorsch, “Payment censorship in the lightning network despite encrypted communication,” in *6th Conference on Advances in Financial Technologies, AFT 2024, September 23-25, 2024, Vienna, Austria*, ser. LIPIcs, vol. 316. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, pp. 12:1–12:24.
- [19] —, “Network-level side-channel attacks in the lightning network,” *ACM Trans. Internet Technol.*, 2026.
- [20] J. Harris and A. Zohar, “Flood & loot: A systemic attack on the lightning network,” in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*. Association for Computing Machinery, 2020, p. 202–213.
- [21] M. Romiti, F. Victor, P. Moreno-Sanchez, P. S. Nordholt, B. Haslhofer, and M. Maffei, “Cross-layer deanonymization methods in the lightning protocol,” in *Financial Cryptography and Data Security - 25th International Conference, FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part I*, ser. Lecture Notes in Computer Science. Springer, 2021, pp. 187–204.
- [22] G. Kappos, H. Yousaf, A. M. Piotrowska, S. Kanjalkar, S. Delgado-Segura, A. Miller, and S. Meiklejohn, “An empirical analysis of privacy in the lightning network,” in *Financial Cryptography and Data Security - 25th International Conference, FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part I*, ser. Lecture Notes in Computer Science, vol. 12674. Springer, 2021, pp. 167–186.
- [23] D. Mehta, T. Ruffing, J. Schnell, and P. Wuille, “Version 2 p2p encrypted transport protocol.” [Online]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0324.mediawiki>
- [24] G. Andresen, “Pay to script hash.” [Online]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0013.mediawiki>
- [25] E. Lombrozo, J. Lau, and P. Wuille, “Segregated witness.” [Online]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki>
- [26] P. Wuille, J. Nick, and A. Towns, “Taproot: Segwit version 1 spending rules.” [Online]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0341.mediawiki>
- [27] G. Andresen, “M-of-n standard transactions.” [Online]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0011.mediawiki>
- [28] M. Jourenko, K. Kurazumi, M. Larangeira, and K. Tanaka, “Sok: A taxonomy for layer-2 scalability related protocols for cryptocurrencies,” *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 352, 2019. [Online]. Available: <https://eprint.iacr.org/2019/352>
- [29] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, “Sok: Layer-two blockchain protocols,” in *Financial Cryptography and Data Security - 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10-14, 2020 Revised Selected Papers*, ser. Lecture Notes in Computer Science. Springer, 2020, pp. 201–226.

- [30] M. Saad and D. Mohaisen, "Three birds with one stone: Efficient partitioning attacks on interdependent cryptocurrency networks," in *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*. IEEE, 2023, pp. 111–125.
- [31] R. Dingledine, N. Mathewson, and P. F. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 13th USENIX Security Symposium, August 9-13, 2004, San Diego, CA, USA*. USENIX, 2004, pp. 303–320. [Online]. Available: <http://www.usenix.org/publications/library/proceedings/sec04/tech/dingledine.html>
- [32] "Geolite databases and web services," accessed on 12 January 2026. [Online]. Available: <https://www.maxmind.com>
- [33] M. Bartoletti, B. Pes, and S. Serusi, "Data mining for detecting bitcoin ponzi schemes," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, 2018, pp. 75–84.
- [34] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *2011 IEEE Third Int'l Conference on Privacy, Security, Risk and Trust (PASSAT) / 2011 IEEE Third Int'l Conference on Social Computing (SocialCom)*. IEEE Computer Society, 2011, pp. 1318–1326.
- [35] "Bolt #2: Peer protocol for channel management." [Online]. Available: <https://github.com/lightning/bolts/blob/master/02-peer-protocol.md>
- [36] "Bolt #7: P2p node and channel discovery." [Online]. Available: <https://github.com/lightning/bolts/blob/master/07-routing-gossip.md>
- [37] C. Ndolo and F. Tschorsch, "Security analysis of bitcoin's v2 transport protocol: Exploiting design implications for sustained eclipse and downgrade attacks," 2026. [Online]. Available: <https://arxiv.org/abs/2605.19715>
- [38] P. Espinasa-Vilarrasa, S. Sanvicente, C. Pérez-Solà, and J. Herrera-Joancomartí, "Decision tree based inference of lightning network client implementations," in *Modeling Decisions for Artificial Intelligence - 21st International Conference, MDAI 2024, Tokyo, Japan, August 27-31, 2024, Proceedings*, ser. Lecture Notes in Computer Science. Springer, 2024, pp. 103–114.
- [39] A. Mizrahi and A. Zohar, "Congestion attacks in payment channel networks," in *Financial Cryptography and Data Security - 25th International Conference, FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part II*, ser. Lecture Notes in Computer Science, vol. 12675. Springer, 2021, pp. 170–188.
- [40] P. Zabka, K. Förster, S. Schmid, and C. Decker, "Node classification and geographical analysis of the lightning cryptocurrency network," in *ICDCN '21: International Conference on Distributed Computing and Networking, Virtual Event, Nara, Japan, January 5-8, 2021*. ACM, 2021, pp. 126–135.
- [41] vasilid, "Bitcoin core issue #31036," 2024. [Online]. Available: <https://github.com/bitcoin/bitcoin/issues/31036>
- [42] "Bolt #10: Dns bootstrap and assisted node location." [Online]. Available: <https://github.com/lightning/bolts/blob/master/10-dns-bootstrap.md>
- [43] I. A. Alharbi, C. C. Zou, and R. N. Alharbi, "Encrypted network traffic analysis and classification utilizing machine learning," *Sensors*, vol. 24, no. 11, p. 3509, 2024.
- [44] A. Sharma and A. H. Lashkari, "A survey on encrypted network traffic: A comprehensive survey of identification/classification techniques, challenges, and future directions," *Comput. Networks*, vol. 257, p. 110984, 2025.
- [45] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
- [46] V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," 2014. [Online]. Available: https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum_Whitepaper_-_Buterin_2014.pdf
- [47] J. Benet, "IPFS - content addressed, versioned, P2P file system," *CoRR*, vol. abs/1407.3561, 2014. [Online]. Available: <http://arxiv.org/abs/1407.3561>
- [48] M. Harrigan and C. Fretter, "The unreasonable effectiveness of address clustering," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, 2016, pp. 368–373.
- [49] G. Maxwell, "Coinjoin: Bitcoin privacy for the real world." [Online]. Available: <https://bitcointalk.org/index.php?topic=279249.0>
- [50] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "Coinshuffle: Practical decentralized coin mixing for bitcoin," in *Computer Security - ESORICS 2014 - 19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7-11, 2014. Proceedings, Part II*, ser. Lecture Notes in Computer Science. Springer, 2014, pp. 345–364.
- [51] R. Stütz, J. Stockinger, P. Moreno-Sanchez, B. Haslhofer, and M. Maffei, "Adoption and actual privacy of decentralized coinjoin implementations in bitcoin," in *Proceedings of the 4th ACM Conference on Advances in Financial Technologies, AFT 2022, Cambridge, MA, USA, September 19-21, 2022*. ACM, 2022, pp. 254–267.
- [52] R. Nithyanand, O. Starov, P. Gill, A. Zair, and M. Schapira, "Measuring and mitigating as-level adversaries against tor," in *23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*. The Internet Society, 2016. [Online]. Available: <http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2017/09/measuring-mitigating-as-level-adversaries-against-tor.pdf>
- [53] A. Kwon, M. AlSabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit fingerprinting attacks: Passive deanonymization of tor hidden services," in *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015*. USENIX Association, 2015, pp. 287–302. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/kwon>
- [54] Y. Sun, A. Edmundson, L. Vanbever, O. Li, J. Rexford, M. Chiang, and P. Mittal, "RAPTOR: routing attacks on privacy in tor," in *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015*. USENIX Association, 2015, pp. 271–286. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/sun>
- [55] A. Biryukov and I. Pustogarov, "Bitcoin over tor isn't a good idea," in *2015 IEEE Symposium on Security and Privacy*. IEEE, 2015, pp. 122–134.
- [56] T. Neudecker, P. Andelfinger, and H. Hartenstein, "Timing analysis for inferring the topology of the bitcoin peer-to-peer network," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, Toulouse, France, July 18-21, 2016. IEEE Computer Society, 2016, pp. 358–367.
- [57] A. Miller, J. Litton, A. Pachulski, N. Gupta, D. Levin, N. Spring, B. Bhattacharjee *et al.*, "Discovering bitcoin's public topology and influential nodes," 2015.
- [58] T. Neudecker and H. Hartenstein, "Could network information facilitate address clustering in bitcoin?" in *Financial Cryptography and Data Security - FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers*, ser. Lecture Notes in Computer Science. Springer, 2017, pp. 155–169.
- [59] M. Nowostawski and J. Tøn, "Evaluating methods for the identification of off-chain transactions in the lightning network," *Applied Sciences*, vol. 9, no. 12, 2019.
- [60] S. Goldfeder, H. A. Kalodner, D. Reisman, and A. Narayanan, "When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies," *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 4, pp. 179–199, 2018.

- [61] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, “Sok: Research perspectives and challenges for bitcoin and cryptocurrencies,” in *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*. IEEE Computer Society, 2015, pp. 104–121.
- [62] F. Franzoni and V. Daza, “Sok: Network-level attacks on the bitcoin P2P network,” *IEEE Access*, vol. 10, pp. 94 924–94 962, 2022.
- [63] R. Alshammari and N. Zincir-Heywood, “Machine learning based encrypted traffic classification: Identifying SSH and skype,” in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, Canada, July 8-10, 2009*. IEEE, 2009, pp. 1–8.
- [64] C. V. Wright, F. Monrose, and G. M. Masson, “On inferring application protocol behaviors in encrypted network traffic,” *J. Mach. Learn. Res.*, vol. 7, pp. 2745–2769, 2006. [Online]. Available: <https://jmlr.org/papers/v7/wright06a.html>
- [65] M. D. Bailey, D. Dittrich, E. Kenneally, and D. Maughan, “The menlo report,” *IEEE Secur. Priv.*, vol. 10, no. 2, pp. 71–75, 2012.
- [66] S. Špaček, P. Velan, P. Čeleda, and D. Tovarňák, “Encrypted web traffic dataset: Event logs and packet traces,” *Data in Brief*, vol. 42, p. 108188, 2022.
- [67] M. Naas and J. Fesl, “A novel dataset for encrypted virtual private network traffic analysis,” *Data in Brief*, vol. 47, p. 108945, 2023.

Appendix A. Ethical considerations

The primary goal of this work is to contribute to the further advancement of the security and resilience of the Bitcoin ecosystem for all involved users. Uncovering, presenting, and addressing potential concerns in the network is a central part of that process. The attack we have presented relies on a known vulnerability (message type classification in the LN) as well as regular protocol and user behaviour. As such, the attack does not call for responsible disclosure.

As far as the practical evaluation of the attacks is concerned, we followed the guidelines of the Menlo report [65] and general security research best practices. In particular, with the exception of conducting measurements using our own node, we did not interact with the public mainnet in any way. The measurements do not reveal any information about other nodes or entities that is not available publicly. All adversarial experiments were conducted in our private testbed that was set up for the precise purpose.

We will not publish our implementation of the attack as we cannot guarantee that it will not be misused. We, however, made the code available to reviewers in full and will consider doing the same to researchers upon request.

Appendix B. Open science

In line with the principles of open science, we have uploaded all of our source code to public repositories. We have made our best efforts to document it clearly enough for independent use. The following artefacts are available:

- 1) the implementation of the testbed that we extended with support for the LN⁸;

8. <https://github.com/tud-dud/btc-ln-play-ground>

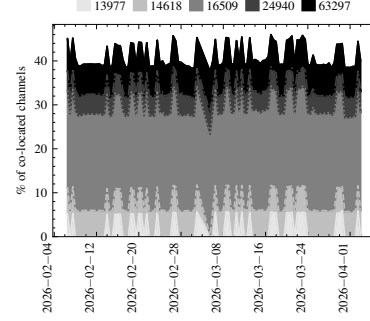


Figure 9. Share of co-located LN channels among the top 5 ASs.

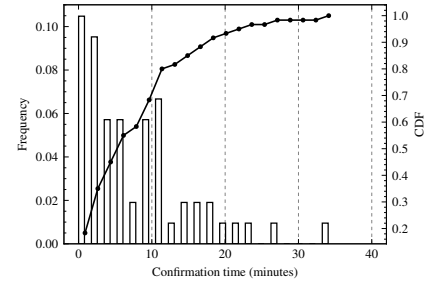


Figure 10. Histogram and cumulative distribution function of the confirmation times. The vertical lines indicate the block boundaries at 10-minute intervals and show which block each transaction would be included in.

- 2) the crawler to measure co-location in Bitcoin and LN⁹;
- 3) the code to analyse the properties and frequencies of funding transactions¹⁰, including the transactions we collected over the measurement period; and
- 4) the traffic classification pipeline.¹¹ We have included a selection of the processed traces to accommodate file size limitations; the remaining traces will be provided upon request.

Our implementation of the attack is not available publicly in order to mitigate potential misuse. We will, however, provide the code for research purposes only upon request contingent on it not being distributed beyond the intended scope. The dataset of raw network traces will also be provided upon request after consultation with and approval from the owners.

Appendix C. Preliminary analysis of confirmation times

Our attack assumes that the funding transaction tx_f is included in the immediate next block after broadcast in ②. However, this assumption may not always hold in practice due to variable mempool conditions such as congestion. We therefore conducted a preliminary check by manually

9. <https://github.com/tud-dud/btc-ln-crawler>

10. <https://github.com/tud-dud/ln-channel-analysis>

11. <https://github.com/tud-dud/btc-ln-traffic-detection>

Table 4. OVERVIEW OF MODERN P2P NETWORKS TRANSPORT LAYER.

Network	Encryption	Description
Bitcoin	✓	TCP + (optional) BIP 324
Bitcoin derivatives (Litecoin, Zcash, Bitcoin cash)	✗	Unencrypted TCP
Ethereum	✓	TCP + RLPx protocol
libp2p (e.g., Filecoin, IPFS)	✓	TCP/QUIC/WebRTC +
Lightning network	✓	TLS/Noise
Monero	✗	TCP + Noise
		Unencrypted TCP

examining 70 randomly selected funding transactions from our dataset using data provided by a block explorer.¹²

Fig. 10 shows the distribution of the transactions’ confirmation times. Around 70% confirmed within 10 minutes and around 90% within 20 minutes, suggesting that funding transactions are typically confirmed quickly. We recognise that confirmation time only approximates next-block inclusion because block intervals vary. A more rigorous verification of the assumption would involve analysing broadcast-to-confirmation time for funding transactions; however, such data was unavailable to us at the time of writing.

Table 5. DATASET FOR TRAINING AND TESTING THE RF CLASSIFIER.

	Protocol	Description
P2P	Bitcoin	1x Bitcoin Core 30.0 2x Bitcoin Core 31.x
	LN	3x LND v.0.20.0-beta
	Ethereum	1x Geth 1.17.2 + Lighthouse 8.1.3 1x Nethermind 1.36.2 + Lighthouse 8.1.3
	IPFS	2x Kubo v0.40.1 relay 2x Kubo v0.40.1 non-relay
Services	SSH	Ubuntu 22.04 VM with OpenSSH_8.9p1
	HTTPS	published in [66]
	SSTP	published in [67]

12. <https://mempool.space>