Node Classification and Geographical Analysis of the Lightning Cryptocurrency Network

Philipp Zabka  
Faculty of Computer Science, University of Vienna, Austria

Stefan Schmid  
Faculty of Computer Science, University of Vienna, Austria

Klaus-T. Foerster  
Faculty of Computer Science, University of Vienna, Austria

Christian Decker  
Blockstream, Zurich, Switzerland

ABSTRACT

Off-chain networks provide an attractive solution to the scalability challenges faced by cryptocurrencies such as Bitcoin. While first interesting networks are emerging, we currently have relatively limited insights into the structure and distribution of these networks. Such knowledge, however, is useful, when reasoning about possible performance improvements or the security of the network. For example, information about the different node types and implementations in the network can help when planning the distribution of critical software updates.

This paper reports on a large measurement study of Lightning, a leading off-chain network, considering recorded network messages over a period of more than two years. In particular, we present an approach and classification of the node types (LND, C-Lightning and Eclair) in the network, and find that we can determine the implementation of 99.9% of nodes in our data set. We also report on geographical aspects of the Lightning network, showing that proximity is less relevant, and that the Lightning network is particularly predominant in metropolitan areas.

As a contribution to the research community, we will release our experimental data together with this paper.

CCS CONCEPTS

• Networks → Peer-to-peer networks; • Social and professional topics → Geographic characteristics.

KEYWORDS

Lightning Network, Classification, Geographical Analysis

1 INTRODUCTION

Blockchain technology enables mistrusting entities to cooperate in the absence of a trusted third party. The technology also forms the basis of cryptocurrencies such as Bitcoin or Ethereum. A main challenge faced by blockchains however regards their scalability: the usual literature example is that while custodian payment systems easily support thousands of transactions per second, blockchains currently merely support tens of transactions per second.

By allowing users to make payments directly, without global consensus protocols and withing having to commit transactions on the blockchain, emerging off-chain networks (also known as payment channel networks or second-layer blockchain networks) [12] can greatly improve the scalability of cryptocurrency payment systems. Indeed, over the last years, off-chain networks such as Bitcoin Lightning [15], Ethereum Raiden [21], and XRP Ripple [11], to just name a few, have received great interest.

As off-chain networks become more popular, the requirements on their performance and dependability increase as well. However, how to efficiently meet these requirements is still subject to ongoing research, and more critically, researchers often lack empirical insights into the currently deployed networks: the publicly available data on these networks is severely limited.

This paper reports on a major measurement study of Lightning, a most popular cryptocurrency network today. In a nutshell, in Lightning, nodes typically represent users (running different Lightning clients, e.g., LND, C-Lightning or Eclair) and edges represent funds that can be transacted between the endpoints of the edge. In order to improve scalability, Lightning supports multi-hop routing of transactions, and in incentivizes the intermediaries to contribute to the transaction routing through fee-based mechanism. To this end, Lightning relies on source routing and in order to support nodes in finding “cheap” routes, i.e., routes with minimal fees, Lightning provides route discovery and gossiping mechanisms.

We recorded network messages (e.g., generated by the gossiping mechanism) in Lightning over a period of almost two years. Based on this data, we contribute insights to two main areas, one related to the security of these networks, and one related to the performance:

(1) Node classification: It can be very useful to know the frequency and distribution of the different clients in an off-chain network. Such knowledge can also be relevant for security considerations, e.g., when planning the deployment of security patches.

(2) Geographic distribution: It is generally interesting to know the topological structure of geographically distributed off-chain networks. In addition to general considerations (e.g., related to economic or sociological aspects), the geographic distribution
We now introduce some of the basics of the Lightning Network and review related work in Section 5 and conclude in Section 6.

which is not necessarily directly connected with the sending node.

The remainder of this paper is organized as follows. Section 2 introduces some preliminaries and Section 3 describes the node classification, followed by the geographical analysis in Section 4. We review related work in Section 5 and conclude in Section 6.

2 PRELIMINARIES

We now introduce some of the basics of the Lightning Network and specific preliminaries for the remainder of this paper.

Clients. The Lightning Network can be accessed via three main implementations or clients: C-Lightning [2], written in C++, LND [6], written in Go, and Eclair [3], written in Scala. These clients have various features, but their fundamental purpose is to create nodes and channels with other participants and act as a ledger.

The Lightning Network. The Lightning Network consists of a collection of nodes and channels. Nodes can create bidirectional connections, called channels, with other nodes which can be then used to send payments almost instantly back and forth between the two participants. The network operates on the blockchain, but unlike Bitcoin, not each payment has to be published onto the blockchain itself, but only the first transaction, known as funding transaction, to fund a channel, and the last transaction, known as closing transaction, to close a channel and end the connection. Between these two transactions users can send an unlimited amount of transactions to each other, as long as they have enough liquidity. Although only a pair of nodes can create a channel, payments can be routed via multiple hops through the network to a receiver node, which is not necessarily directly connected with the sending node. Nodes helping in forwarding payments through their channels will usually collect a small fee for this service.

Gossip Messages. To utilize a path of more than one channel as payment route, nodes have to be aware of the network topology, in order to know which channels can be used to route the payment to the final receiver. For this purpose gossip messages are propagated from one node to another, to either announce a newly created node, channel or an update of both. In the following section we introduce the three most important gossip messages for our work and some of their contained information, which are specified in the Basics of the Lightning Technology (BOLT) [15]:

- node_announcement message: This message is propagated either when a node has been created and is now ready or it updates its information. The message contains important parameters which enable other participants in the network to start channels with the specific node. For example, when a node wants to connect to another node, the node_id is needed for identification. Other for us relevant parameters contained in this message are alias, a nickname for a node encoded in UTF-8, color encoded in hexadecimal and the addresses parameter, which can contain IPv4 or IPv6 addresses as well as Onion v2 or Onion v3 service addresses.

- channel_announcement message: Always when a new channel between two nodes is created a channel_announcement message is sent. This gossip message contains information regarding the newly created channel and is propagated exactly once in the network. Similar to the node_id each channel has an unique short_channel_id for identification. Furthermore, the message contains amongst other parameters the node_id of the two nodes connected by the channel.

- channel_update message: A channel is not practically usable until at least one side has announced its fees and expiry for the HTLC of the payment. A Hashed Time Locked Contract (HTLC) is a security measure to ensure that nodes along the routed path do not steal the payment. This gossip message is propagated at least once from each of the participating nodes, since the initial routing fee may differ depending on the direction the payment comes from i.e. from node A to node B or from B to A. Also every time a side decides to change its channel parameters a channel_update message needs to be propagated again through the network. Further relevant parameters are short_channel_id, the channel_flag indicating the direction the channel update is coming from and then four parameters describing important channel settings, namely cltv_expiry_delta, htlc_minimum_msat, fee_base and fee_proportional_millionths.

2.1 Data Set

Our unique data set is comprised of the three gossip messages introduced in the previous section, which were propagated through the network from March 2018 to January 2020. In this time span we recorded more than 400,000 node_announcement messages, more than 1,000,000 channel_announcement messages, and over 6.4 million channel_update messages. A first analysis shows that the real growth of the Lightning Network may also be relevant for the performance and dependability of these networks: network topologies with local biases may improve performance, but may be less robust.
started in 2018, which is also the year where LND and C-Lightning released their first major update for their clients.

3 NODE CLASSIFICATION

This section reports our main results from the node classification. The Lightning Network is currently comprised of three implementations: LND, C-Lightning and Eclair, each written in a different programming language and each using slightly different values for its parameters. Some of these values are public and can be obtained through message inspection of the gossip messages, others in turn are kept private for network security reasons.

One of these private parameters is \texttt{max\_concurrent\_htlcs}, which denotes the maximum capacity of HTLCS for a channel and which can play an important role in attacks on the networks topology: an attacker may want to determine how many HTLCS will be necessary to overload a channel, and hence make the channel unusable until the HTLCS resolve. These attacks can be targeted on the whole network or just affect a single node. For this reason precisely inferring a node’s implementation to deduce the values for these parameters is key. Furthermore we want to analyze how the implementations are distributed in the network.

3.1 Analyzing Default Parameters

We now take a closer look at the parameters of interest. Table 1 shows the default values for the three available implementations of the Lightning Network:

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>LND</th>
<th>C-Lightning</th>
<th>Eclair</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>-</td>
<td>Predefined</td>
<td>-</td>
</tr>
<tr>
<td>color</td>
<td>#3399ff</td>
<td>Derived from node_id</td>
<td>#49dAAA</td>
</tr>
<tr>
<td>cltv_expiry_delta</td>
<td>40 (144)</td>
<td>14</td>
<td>144</td>
</tr>
<tr>
<td>htlc_minimum_msat</td>
<td>1000</td>
<td>1000</td>
<td>1</td>
</tr>
<tr>
<td>fee_proportional-_millioths</td>
<td>1</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>fee_base_msat</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
</tbody>
</table>

Table 1: Default parameters table

\textbf{Alias.} The alias parameter for LND and Eclair does not have a default value and has to be manually configured when a new node is created. C-Lightning uses NSA-Style names, which are created from an adjective and a noun, both originating from predefined lists in the C-Lightning source code [1]. The BOLT [14] documentation gives us two example names: ‘IRATEMONK’ and ‘WISTFULTOLL’.

\textbf{Color.} Both LND and Eclair have a predefined color parameter, which is automatically set up with each node creation. For C-Lightning the node’s color is automatically derived from the three first bytes of its node_id. Hence, older LND nodes will probably still use 144.

Next, we analyze the distribution of the introduced parameters in the network. Other works such as [29], [28], [20] and [18] have already performed some analysis via snapshots taken with the help of LND’s \texttt{describe\_graph} command. However, as our data set covers gossip messages of almost two years, going vastly beyond single snapshots, we are able to obtain fairly precise insights on the parameter distributions.

\textbf{cltv\_expiry\_delta.} We now examine the parameters in the channel\_update message and start with the examination of the \texttt{cltv\_expiry\_delta} parameter, which denotes the minimum difference in HTLC timeouts a node that is forwarding a payment will accept. Figure 1 shows us that 144, the old LND and current Eclair value, is represented in the data by 53.8%. The new value for LND, 40, is represented by a share of 21.3%. The value of 14 corresponding to the C-Lightning implementation takes a share of 7.3% in the overall data. Since LND and Eclair used to have the same value, we can only estimate the share of Eclair’s default value to be under 32.5%. Overall 82.4% of the \texttt{cltv\_expiry\_delta} parameters use a default value.

We have also taken a closer look on how the distribution of 144 and 40 has developed over time, as LND changed its \texttt{cltv\_expiry\_delta} parameter from 144 to 40 in March 2019 [5], to get an first impression of how long it roughly takes for an update to be adapted by the majority of the nodes. Figure 2 shows us that roughly two months after the change only 0.44% of the \texttt{cltv\_expiry\_delta} parameters in the network used the value of 40 until May 2019. However we observe that after the initial stagnation, usage of this value started to continuously increase, whereas the value of 144 started to continuously decrease. The new value of 40 surpassed the old value of 144 approximately mid December, from which we
deduce that it took around 9 months for the update to reach the majority of LND nodes in the network.

**htlc_minimum_msat.** Figure 3 depicts the value distribution for the `htlc_minimum_msat` parameter, which denotes the minimum value in millisatoshi for a payment to be transferred of the channel. LND’s and C-Lightning’s default value of 1000msat has a share of 67.4% and Eclair’s value 1 has a share 15.1% in the data set. Interestingly, in 10.8% of the updates the value 0 has been used, indicating no minimum value for payment transportation.

**fee_proportional_millionths.** Next, we studied the distribution of the `fee_proportional_millionths` parameter, which is the amount nodes will charge for each transferred satoshi over their channel. The value 1 for LND can be found in 63.7% of the data, the value 10 used by C-Lightning is represented in 6.9% of the updates. Lastly, Eclair’s value of 100 can be found in 2.4% of the data.

Since this is the first parameter, which is different in all of the three implementations, Figure 4 gives us a first insight in a possible distribution of the implementations in the network.

**fee_base_msat.** The `fee_base_msat` parameter depicted in Figure 5, denotes the constant fee a node will charge for a transfer. As the parameter is the same in all three implementations, it does not provide much insight in the distribution of the individual implementations, but interestingly it has the smallest overall share in the network, when summing up all the default values for the other parameters.

Our analysis shows that because of how the data is structured, it is of little avail to classify the nodes with either a machine learning algorithm or a clustering approach.

Our classification algorithm can be seen as an extension from [18]. The algorithm takes six parameters as input, namely the `node_id`, `alias`, `color`, `cltv_expiry_delta`, `htlc_minimum_msat`, and `fee_proportional_millionths`. We chose not to include the `fee_base_msat` parameter, as it is the same for all three implementations; however one could use it as an additional check, as it was done in [20]. The `node_id` is, as mentioned earlier, needed for the derivation of the C-Lightning
Figure 5: fee_base_msat distribution

Figure 6: Labeled nodes after pre-processing: -1: unclassified, 0: LND, 1: C-Lightning, 2: Eclair. Darker points indicate a higher node density

Figure 7: Labeled virtual nodes: -1: unclassified, 0: LND, 1: C-Lightning, 2: Eclair. Darker points indicate a higher node density

3.4 Results and Evaluation

Figures 8 and 9 show that we have been able to classify almost all nodes in our data set, as 99.9% of the nodes have been classified successfully. Both data processing methods have classified the nodes very similarly. We can see that our analysis resulted in labeling approximately about 87% of the nodes as LND nodes, about 11% of the nodes as C-Lightning and about 2% of the remaining nodes as Eclair, making it the least used implementation in the network.
We have demonstrated, that with enough data, it is possible to pre-
1.31% of the nodes labeled as LND use 30 as a parameter, this value
of interest and previous channel or node setting data is not available
especially interesting when the implementation of a certain node is
incorporating some measurements concerning node performance in
the network, since all implementations use a different programming
language which could affect its efficiency. This method can be
the chance to discover default values for nodes, even if their users
changed these values at some point in the past.

As far as the accuracy of the classification is concerned, the next
graphs give us more insights. For all nodes labeled as a certain imple-
mentation, we considered the parameters based on which the clas-
sification algorithm made its decision. Figure 10 shows the para-
eter distribution for cltv_expiry_delta, htlc_min_msat and
fee_proportional_millionths for all nodes inferred
as LND. The left plot from Figure 10 shows us that in fact the two
most used values for the parameter cltv_expiry_delta 144
and 40, yielding 97.13% are default values for LND. Interestingly,
1.31% of the nodes labeled as LND use 30 as a parameter, this value
is often used by a certain node provider known as “LNBIG.com”. For
the next parameter htlc_minimum_msat, 91.58% of the nodes
use the value 1000, also a default value. Some individual nodes also
use a value of 0 or 1. For the last parameter fee_proportional
−_millionths, also very few nodes use a value other than the
default value of 1 500 is used by “LNBIG.com” and 100 is again
used by some individual nodes. We have evaluated labeling preci-
sion of C-Lightning and Eclair as well, showing a similarly high
distribution of the default values. We can deduce from these results
that the nodes have been classified precisely.

3.5 Discussion
We have demonstrated, that with enough data, it is possible to
precisely infer the implementations of almost all nodes. The problem
with node classification in the Lightning Network is that a user
in fact can easily change these parameters we based our classifi-
cation upon. Once all of these parameters have other values than
the default ones, a classification with these parameters is no longer
possible and new methods have to be explored. However, in Section
3.1 we have shown that the usage of default values for parameters
is very common, from which we can deduce that most users stick
with the default settings. Also, our data set covers messages of
almost two years. In this time span we have gathered numerous
data points for the individual nodes in the network, which increases
the chance to discover default values for nodes, even if their users
changed these values at some point in the past.

Another possibility of performing a classification could be by
incorporating some measurements concerning node performance in
the network, since all implementations use a different programming
language which could affect its efficiency. This method can be
especially interesting when the implementation of a certain node is
of interest and previous channel or node setting data is not available
or the user has changed these parameters. However, further testing,
data gathering and research will be necessary.

Lastly we want to address the implementation distribution. From
our results and also the results of [18] we can observe that LND is
by far the most popular client for the Lightning Network. Though
we can’t precisely substantiate, why this is the case, we can make
some assumptions based on facts. Firstly, C-Lightning has already
released first versions of its clients by 2016. These releases where
only provided as source code and had to be compiled manually
by the user using Linux or possibly other UNIX based operating
systems. LND released its first major version in 2017. However, in
contrast to C-Lightning or Eclair, LND offered already precompiled
versions of the client for the various CPU architectures and oper-
ating systems, including Windows. Especially, making the client
available for Windows could have made a huge impact on LND’s
popularity.

4 GEOGRAPHICAL ANALYSIS OF THE LIGHTNING NETWORK
In this section we present our findings with regard to the geographical
conditions of the Lightning Network. Similar studies have been
conducted in the past, mainly concerning cryptocurrency networks
for example Bitcoin. In this paper, we go beyond previous work
as we perform this kind of analysis on a novel payment channel
network, in particular on the Lightning Network. Our analysis in-
cludes 81 countries, in which the Lightning Network is present.
Our focus lies especially on North America, Europe and Asia, since
these are the continents with 94% of the overall node population.

4.1 Implementation Distribution in Countries
After obtaining the labels for each node, we now look at the imple-
mentation distribution in the individual countries. In the previous
section, we have seen how the labels are distributed in general. We
observed that LND is with about 87% by far the most frequently
used implementation, followed by C-Lightning and Eclair, with 11%,
respectively 2%.

By considering the IP-address for each classified node, if avail-
able, we can approximately determine its location, using an API [4].
We analyze the implementation distribution in each country, to see
if the results on a country level mirror our previous results.

Figures 11 shows the distribution in 40 individual countries.
The results show that the implementations within a country are
similarly distributed as in our general labeling results. For all the
countries, that we could determine through a node’s IP-address,
we observed that LND is predominant in 78 out of all 81 countries.
Only Turkey, Iran, and the Isle of Man shows a higher C-Lightning
usage. With respect to Eclair, there is no country where it has a
higher share than LND. However, in seven countries C-Lightning
has an equal distribution, and in Poland, Belgium, Lichtenstein and
the Philippines, Eclair holds a higher distribution. Further analysis
(Figure 11) shows that countries, where a single implementation is
represented by 100% of the nodes or where one of the other two
implementations holds a higher share than LND, only have a few
or just one single node. For example, in India there is a total of four
nodes, from which all use the LND implementation.
4.2 Channel Connection Behavior Analysis

We next analyze if there is a geographical preference in the connection behavior of nodes. For example, do nodes connect to more geographically closer nodes for network latency benefits, or does the location matter at all and are there other reasons for establishing a connection? Seres et al. [26] state, that nodes, because of the way they are implemented, tend to create channels with already large hubs rather than smaller nodes; this makes the network prone to topological attacks, since the removal of only one hub can already heavily impact the network’s connectivity.

Our empirical analysis shows that nodes indeed tend to connect to large hubs, even if there is a large distance between them. Almost every node from all 81 considered countries connects to the same six countries: the United States, Germany, Canada, the Netherlands, United Kingdom, France, and maybe interestingly, Switzerland. In almost all cases the USA is the leading country to connect to, followed by Germany. A significant portion of the nodes in the Lightning Network are located in these countries, but also the most connected nodes, with the United States having more than 280 000 channels, Germany having more than 3000, and Canada and the Netherlands having more than 3000 and 2000 channels. In Figure 12, depicting the channel connections in India and Japan, we can observe this pattern as well. Interestingly, a lot of countries, i.e., Japan, Poland, or China, also show a high node connectivity within the country, depicted in Figures 12 and 13. This pattern could be due to country specific node providers, whose nodes are interconnected within a country.

4.3 Analyzing Node Location

A large share of the nodes in the Lightning Network are located in North America with 44.8% and Europe with 43.1%. The remaining nodes are located in Asia with 6.2%, Oceania with 2.2% and lastly South America and Africa with each having 0.8% and 0.6% of the nodes in the Lightning Network. For 2.3% of the nodes we couldn’t determine a location because of missing IP-addresses. In Figure 14 we can observe the node location distribution on three continents: Europe, North America and Asia. By plotting the node latitude and longitude coordinates, we can clearly identify Europe and North America, due to the high node density in these continents. Looking at Figure 15 (left) we can see that most of the nodes are located in Central Europe. Figure 15 (middle) shows a very high node distribution on both the West Coast and the East Coast, as well as occasionally inside the country. In Asia, depicted in 15 (right), most of the nodes are located on the coasts of South Korea, China and Japan.
From Figure 14 we can also observe that locations, where the node density is higher, tend to have a better infrastructure, e.g., Central Europe - Eastern Europe, North America West/East Coast - North America Inland.

To further evaluate on this aspect, we studied the node location distribution inside of a country, e.g., in Germany and Japan. In Figure 16 we can see multiple node clusters in Germany, each centered around one of Germany’s larger cities, with the largest being in the metropolitan area of Berlin (52.52, 13.40) and second and third largest around Munich (48.13, 11.57) and Frankfurt (50.11, 8.68). For Japan depicted in Figure 17, the largest node hub is located in the metropolitan area of Tokyo (35.65, 139.74), followed by the metropolitan areas of Osaka (34.66, 135.49), and Kobe (34.68, 135.19).

4.4 Discussion

Our findings exhibit how the Lightning Network and its implementations are distributed in the world. We could observe that LND is popular in almost all countries and also showed that within a country nodes form clusters around cities and expand into their metropolitan areas. Also infrastructure plays a significant role in the distribution of nodes within a continent or country.

Our analysis of channel connections between countries, shows a pattern of nodes always connecting to the same countries, and we have found other possibly interesting patterns as well. By normalizing the number of channels each country shares with other countries by the number of nodes in these countries, we could observe that nodes in some countries, which either share the same or similar language or ethnicity tend to establish channels as well. Our first analysis has shown that Argentina shares 80% of the channels with Paraguay, then Peru with 10% of the channels and lastly with Chile and Venezuela with around 2-4%. Kenya shares more than 70% of the channels with South Africa, China shares most channels with Taiwan and also several with Japan and Hong Kong, Slovenia shares channels with Croatia, Czechia and Bulgaria and Mexico with Colombia, Chile, Puerto Rico and Argentina. However, more studies have to be carried out in order to evaluate this behavior.

5 RELATED WORK

For an overview of the blockchain and Bitcoin in general we refer the reader to Antonopoulos [7], and for an overview of off-chain networks specifically, to the survey by Gudgeon et al. [12]. There exist many clever route discovery algorithms in the literature, e.g., SpeedyMurmurs [23] and SilentWhispers [10], to improve the routing efficiency in off-chain networks. However, it has also been shown that the gossiping and probing mechanisms needed in off-chain networks to support efficient routing, may introduce
security issues, e.g., harm privacy [19] and/or performance if nodes behave selfishly [28]. Some papers already have explored node classification in the Lightning Network, mostly as a preparation measure for an attack. Mizrahi et al. [18] performs a classification for a congestion attack on the Lightning Network and also suggests mitigation techniques. The data for this work was gathered with the `describegraph` command of LND, which returns a JSON describing the networks topology to a given timestamp. Our data was gathered by a node logging the messages it received from March 2018 until January 2020. We also consider more parameters to ameliorate the classification results and perform some further analysis with the results. Pérez-Solà et al. [20] proposes an attack to discover channel balances in the network and also infers implementations for deriving a private parameter’s value.

Measurement studies, e.g., also considering geographic aspects, have been performed on many other peer-to-peer and social networks, also before cryptocurrencies. For example, Schiöberg et al. [25] conducted an analysis of the social network Google+, which also includes an examination of user locations. Scellato et al. [24] study how geographic distance affects social ties in a social network and Mislove et al. [17] geographical, gender and racial aspects of Twitter users to the U.S. population. Measurements and implications on attacks in the peer-to-peer network Kad have been discussed by Locher et al. [16]. Dotan et al. [9] recently presented an overview of cryptocurrency networks, which also includes a survey of empirical studies on geological characteristics of the Bitcoin and Ethereum networks. However, we are not aware of works investigating similar aspects on off-chain payment channel networks so far. The Lightning Network’s topology has been analyzed by Seres et al. [27]. The work studies the robustness of the network against random failures of nodes and targeted attacks. The authors also propose some countermeasures to make the network more resilient. A similar, but more in depth work as been work has been carried out by Rohrer et al. [22]. As far as privacy is concerned, Kappos et al. [13] performed an empirical analysis of the Lightning Network based on three attacks. The analysis also included measurements concerning network, node, and channel parameters.

6 CONCLUSION

Analyzing a big dataset collected on the communication in the Lightning networks, we have shown that it is possible to accurately classify the node types in Lightning network with high probability, potentially providing important security insights. We understand our work as a first step and we plan to continue collecting data for more extensive studies, also of the network evolution over time. We also believe that our work opens several interesting questions for future research. For example, it will be interesting to explore alternative classification algorithms, improving the accuracy further, or to investigate the applicability of our methods on alternative off-chain networks.

In order to support such future research, our data is available at https://github.com/lnresearch/topology [8].
Research supported by the Vienna Science and Technology Fund (WWTF) project ICT19-045 (https://www.wwtf.at/programmes/information_communication/ICT19-045).

REFERENCES