Securing Wireless Payment Channel Networks With Minimum Lock Time Windows

Gabriel Antonio Fontes Rebello, Maria Potop-Butucaru, Marcelo Dias de Amorim, Otto Carlos Muniz Bandeira Duarte

Securing Wireless Payment-Channel Networks With Minimum Lock Time Windows

Gabriel Antonio F. Rebello[1,2], Maria Potop-Butucaru[2],
Marcelo Dias de Amorim[2], and Otto Carlos M. B. Duarte[1]

[1]Universidade Federal do Rio de Janeiro, Brazil
[2]Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

***Abstract*—Payment-channel networks (PCN) enhance the impact of cryptocurrencies by providing a fast and consensus-free solution to the scalability problems of traditional blockchain protocols. However, PCNs often rely on powerful nodes with high availability, large storage capacity, and strong computational power, which hinders their adoption in mobile environments. In this paper, we consider a PCN architecture that extends the functionalities of traditional PCNs to wireless resource-constrained devices. We address the token theft problem, a vulnerability that is critical on wireless PCNs, and propose a countermeasure based on minimum time windows that lock tokens whenever a user disconnects. We evaluate our proposal with real data from Bitcoin's Lightning Network and 3G/4G mobile broadband networks. The results show that the countermeasure is most effective when devices present high availability and that there is a security-efficiency trade-off when connectivity is low.**

I. Introduction

Payment channels and payment-channel networks offer a scalable and efficient off-chain solution to improve the performance of blockchain-based systems. To open a payment channel, two users sign and publish a funding transaction that transfers a fixed amount of tokens to a joint address in the blockchain. The users can then continuously rebalance the funds of the address by sending private signed commitment transactions, as shown in Figure 1. This way, both parties transact with each other without paying the blockchain fees and waiting for the system to approve the transactions through consensus. To close the channel, either user publishes the last commitment transaction into the blockchain and waits for the *lock time window*[1] to recover their tokens. The lock time window serves as a security mechanism to prevent a user from stealing tokens when the other party disconnects. We explain this process in detail in Section III.

A payment-channel network (PCN) is the collection of payment channels that form a peer-to-peer network for token transfers. In PCNs, nodes can issue payments to destinations even if they do not share a payment channel. For example, in Figure 2, if Alice wants to send one token to Charlie, she can send the token to Bob, and Bob relays it to Charlie. Bob receives the token from Alice once he sends one token of his own to Charlie through Hashed Timelock Contracts (HTLC) [3], a special type of blockchain script. This enables micro-transactions to occur in real-time, unburdens the consensus protocol, and effectively narrows the gap between cryptocurrencies and everyday-life needs. PCNs have been improving the usage of cryptocurrencies, serving even as the basis of the recent official adoption of Bitcoin as an official currency in El Salvador [4].

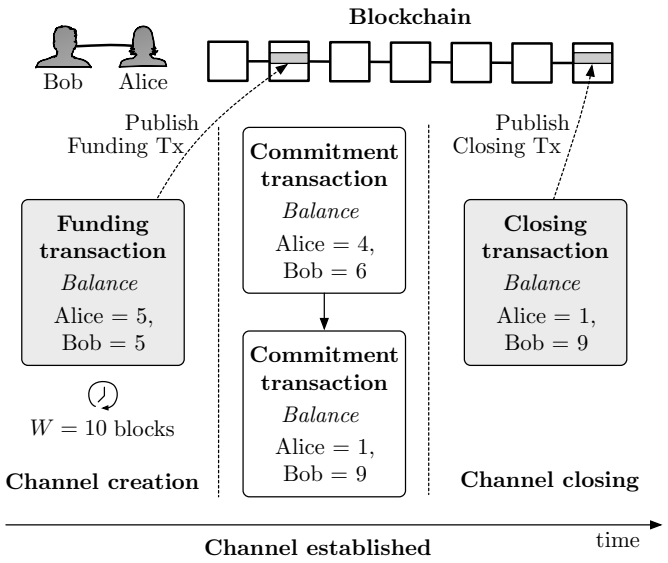

Fig. 1: A payment channel between users Alice and Bob. Both users issue private commitment transactions in real-time after establishing the channel. The funding transaction contains a lock time window W that locks the tokens for a predefined number of blocks once the channel is closed unilaterally.

Nevertheless, payment-channel networks present open challenges for resource-constrained wireless devices such as mobile phones, smart objects, and sensors. Current PCNs rely on high availability, large storage capacity, and strong computational power. For example, most implementations define payment paths through a synchronized global topology, assume nodes store a full copy of the blockchain, and adopt onion routing [2], [5], [6]. Such implementations are unsuited for wireless devices with limited resources and intermittent connectivity patterns that already account for over half of all the traffic on the Internet [7].

[1]The lock time window is often referred to as "to self delay" or "return delay" in the literature [1], [2]. We consider the terms equivalent.

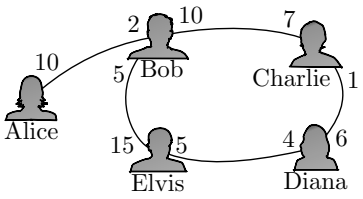

Fig. 2: An example of a payment-channel network (PCN) composed of bidirectional payment channels with limited capacity. User users who do not share direct links can route payments through intermediaries.

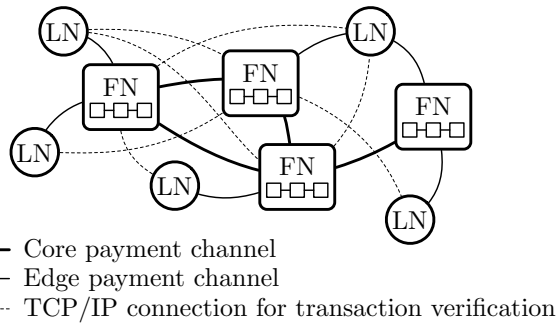

Fig. 3: An example of a topology of wireless payment-channel networks. The light nodes (LN) represent wireless devices, and the full nodes (FN) represent nods that store a copy of the blockchain. Light nodes establish TCP/IP connections to verify the states of their channels in the blockchain.

Besides, new security challenges appear in resource-constrained environments, such as how to secure payments with lossy connections and limited access to the blockchain. Although several works in the literature propose adaptations of the Lightning Network to mobile devices [8], [9], [10], our work is, to the best of our knowledge, the first to analyze the security of wireless payment channels in a PCN-agnostic manner. We believe that addressing such challenges and implementing a PCN for payments via mobile devices is the final step towards the mass adoption of blockchain technology.

Our contribution. We make several contributions towards implementing *wireless payment-channel networks* (WPCN): (i) we propose a hybrid PCN architecture that allows devices to issue payments despite presenting intermittent connectivity and lacking the capacity to store a blockchain node. This architecture is inspired by mobile broadband (MBB) and Internet of Things (IoT) architectures and can be easily applied to such environments; (ii) we formulate and analyze the *token theft problem*, a vulnerability that is present in all PCNs [5], [6], [2], [11], [12] but becomes critical in wireless environments due to device downtime and packet loss. We demonstrate that the system could collapse if the problem is not correctly addressed; (iii) we propose an efficient countermeasure to the token theft problem based on lock time windows that most adopted payment-channel networks already implement [5], [6], [11]. Our solution is PCN-agnostic and does not require any modifications to current PCN implementations; and (iv) we analyze the efficacy of our approach using real data from Bitcoin's Lightning Network and 3G/4G mobile broadband connections to simulate a real-world scenario [13], [14].

II. Wireless Payment-Channel Networks

We consider Wireless Payment-Channel Networks (WPCN), a hybrid PCN architecture composed of a static and reliable core as well as peripheral unreliable mobile devices with limited resources. We argue that a hybrid topology is the most impactful because the main IoT platforms and mobile device architectures today rely on gateways and edge computing to provide services. Figure 3 depicts the topology of our network architecture. We define two types of nodes:

- **Full nodes (FN)**, which compose the core network and act as routers. Full nodes may represent service companies, telcos, or any node with computing power to store a full copy of the blockchain. Full nodes are online with high probability at all times and communicate via a reliable transport protocol. Although churn can occur in the core network, the probability that a full node quits the network without closing its payment channels is negligible compared with light nodes.
- **Light nodes (LN)**, which are mobile devices that connect to the network via lossy wireless connections and are resource-constrained. Light nodes may represent mobile phones, sensors, smart objects, or any IoT device that presents limited computing and storage capacities. We assume light nodes can disconnect at any time without closing the payment channel properly due to battery issues, hardware malfunction, environmental conditions, and other limitations of mobile devices. We assume light nodes establish unreliable connections to send/receive transactions and request channel state verification. They can perform public-key cryptography to sign transactions but cannot store a copy of the blockchain.

Full nodes connect to other full nodes via payment channels with a large capacity to route payments. Light nodes connect to one or more full nodes via smaller and possibly unidirectional payment channels. Henceforth we refer to channels between full nodes as *core payment channels* and channels between a light node and a full node as *edge payment channels*. We do not consider payment channels between light nodes as it would be unlikely for two light nodes to continuously transact with each other for a long period. *Entry nodes* are the nodes a light node selects as its connections. Light nodes also establish TCP/IP connections to other full nodes to verify the states of their channels and avoid eclipse attacks. We formally define Wireless Payment-Channel Networks (WPCN) below [15]:

Definition 1 (Wireless Payment-Channel Network (WPCN)). A wireless payment-channel network is a time-varying directed graph $\mathbb{G}(t) := (\mathbb{V}(t), \mathbb{E}(t))$, where $\mathbb{V}(t)$ is the set of devices in the network at time t and $\mathbb{E}(t)$ is the set of payment channels that are open at time t. Any device $u \in \mathbb{G}(t)$ can alter the set

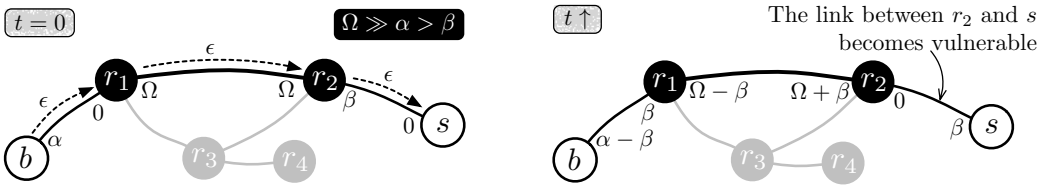

Fig. 4: An example of the token theft vulnerability in WPCNs. On the left, a continuous amount of ϵ token flows from buyer b to seller s until the channel between r_2 and s is depleted. Then, on the right, s becomes highly vulnerable if it loses connection before closing the channel because r_2 has nothing to lose by closing the channel with a previous balance.

$\mathbb{E}(t)$ of edges via three possible operations:

- `openChannel`$(\langle u, v\rangle, \langle \alpha, \beta\rangle, T, F, W)$ opens a payment channel $u \leftrightarrow v$ with capacity (α, β), timeout T, and fee F. The window W defines a lock time in which neither party can claim the tokens if the channel is closed unilaterally. The operation publishes a transaction in the blockchain that must be signed by both u and v;
- `closeChannel`$(\langle u, v\rangle, Tx(t))$ closes the payment channel $u \leftrightarrow v$ with transaction $Tx(t)$, which contains the latest balance that has been signed at time t by both parties, and publishes it in the blockchain. This operation can either be issued cooperatively by having both signatures or unilaterally by either party. If the channel is closed unilaterally, the party that closed the channel can only claim her/his tokens after the predefined time window W;
- `pay`$(\langle u, v\rangle, p, V)$ transfers a value of V tokens from u to v via path $p = \langle u, r_1, r_2, ..., r_n, v\rangle$. We assume the path p is defined by u before issuing the operation. Every hop from u to v will have its capacity decreased by V tokens in the direction of the payment receiver if the whole path has enough capacity. Otherwise, the operation fails and all channels remain unaltered.

III. The Token Theft Problem in WPCNs

Traditional PCNs like Lightning [5] and Raiden [6] assume that any node that transacts in the network remains online while the channel is open. This mitigates the *token theft problem*, in which one party publishes an old transaction to recover her/his sent tokens as soon as the other party disconnects. The system punishes malicious nodes by allowing the victim to spend all tokens in the channel if it recovers during the lock time window. *Hence, it is only worth it to attempt the attack if the malicious node can guarantee that the other party will not verify the blockchain until the time window expires.*

In our network architecture, however, light nodes can naturally disconnect for long periods or even indefinitely. Device downtime is especially challenging for use cases where the trend of payments is biased towards a light node, such as when a seller uses her/his device to receive transactions from multiple buyers or when a buyer uses her/his device to buy services from multiple sellers. We formulate the problem and demonstrate how imbalanced channels in resource-constrained environments enhance the probability of malicious behavior.

Let two resource-constrained devices, b and s, represent devices from a buyer and a seller, respectively, and be connected to entry nodes r_1 and r_2 via unidirectional payment channels as shown in Figure 4. Each payment channel $u \leftrightarrow v$ has a balance $bal_{u\leftrightarrow v}(t) = (bal_u(t), bal_v(t))$, where $bal_u(t)$ and $bal_v(t)$ are the balances of nodes u and v at time t, respectively. For edge payment channels between buyers and entry nodes, e.g. $b \leftrightarrow r_1$, we assume an initial balance of $bal_{b\leftrightarrow r_1}(0) = (\alpha, 0)$, where α is an amount of tokens that buyer b reserves for payments in the channel. Conversely, the initial balance of edge payment channels between sellers and entry nodes, e.g. $r_2 \leftrightarrow s$ is $bal_{r_2\leftrightarrow s}(0) = (\beta, 0)$ where β is the amount of tokens the entry node r_2 reserves for routing payments to the seller s. We assume for simplicity and w.l.o.g. that s and b only participate in a single payment channel.

Once payment `pay`$(\langle b, s\rangle, \{r_1, r_2\}, \epsilon)$ occurs from b to s in this configuration, r_2 and s sign a commitment transaction $Tx(1)$ containing the new balance of channel $bal_{r_2\leftrightarrow s}(1) = (\beta - \epsilon, \epsilon)$. If s disconnects at this moment, r_2 can close the channel with the operation `closeChannel`$(\langle r_2, s\rangle, Tx(0))$ and recover ϵ tokens. This is risky because r_2 would lose $\beta - \epsilon$ tokens if s recovers before the lock time window expires. However, as s receives more payments, the balance in $r_2 \leftrightarrow s$ will converge to $bal_{r_2\leftrightarrow s}(t) = (0, \beta)$. Once this happens, r_2 has nothing to lose by closing the channel with a previous transaction even if s recovers on time. Thus, acting maliciously is the optimal strategy for any rational entry node once a border payment channel to a seller is depleted, and the seller is prone to token theft even in the absence of actual malicious nodes. Malicious nodes may also decide to attack intermediary cases once the victim disconnects if they expect a good risk-benefit ratio. Note that the problem is unlikely to occur to senders because the other party can only lose tokens by publishing a previous transaction. For a generic situation in which light nodes act as senders and receivers, every light node becomes vulnerable as soon as it receives tokens.

IV. Defining a Minimum Time Window

A straightforward solution to the token theft problem is to hire "watchtower" nodes that constantly verify the blockchain to detect channels that have been improperly closed [2], [5], [6]. The solution, however, may cause privacy issues and only works if the node sends the commitment transaction information before disconnecting. Another countermeasure

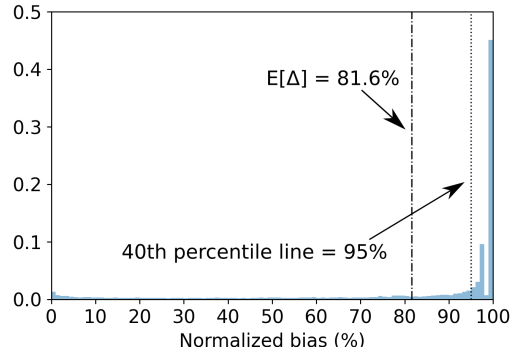

Fig. 5: Normalized bias Δ of payment channels in the Lightning Network. 60% of channels present over 95% bias towards one party and the average bias is 81%, which indicates a heavily asymmetric behavior of payment flows.

could be to create a reputation system for full nodes. Reputation systems, however, lead to centralization and introduce new attack vectors that would be difficult to handle in decentralized environments [16]. Instead of adopting watchtowers or creating a reputation system, we propose a simple statistical approach: *discover a lock time window W that minimizes the chance of attacks*. Since most payment-channel networks already adopt lock time windows as a security measure [5], [6], [11], we believe this solution is the easiest to implement and possibly the most impactful. To the best of our knowledge, no works have ever proposed to find a minimum time lock window value to prevent attacks. Our contribution also applies to traditional PCNs as a guideline for users to select the best window parameters based on their actual connectivity patterns.

Our proposal. The lock time window, W, defines the amount of time that the closing party must wait until it can recover the tokens and serves as a countermeasure against token theft. If a node on a channel disconnects and a token theft attack occurs, the victim has W blocks to recover, verify the blockchain, and punish the attacker before she/he claims the tokens. Hence, the larger W, the more secure the channel becomes. Conversely, setting a large W can create bottlenecks in routing and punish honest nodes that wish to close the channel correctly after the other party disconnects. In such cases, W should be as small as possible to improve token liquidity and overall throughput. Therefore, the value of W represents a trade-off between security and efficiency, and our goal is to minimize W while guaranteeing a minimum level of security.

Let s be a resource-constrained device. We propose a methodology that uses four parameters to estimate W:

(i) T_{off}, a random variable that models the time s remains disconnected from the system, which can occur due to device failure or packet loss. T_{off} can either be modeled through a continuous random distribution or be estimated with empirical data from a dataset;

(ii) D_{det}, a random variable that models the delay for s to detect the attack. D_{det} follows a Poisson distribution with expected value bound by the equation

$$E[D_{\text{det}}] = n\frac{E[T_{\text{off}}]}{b_t}\left(\frac{b_s}{d} + v\right), \tag{1}$$

where n is the number of full nodes from which s requests blocks, $E[T_{\text{off}}]$ is the average downtime of s, b_t is the block time[2], b_s is the average block size, d is the average download rate of s, v is the average time it takes for s to verify all transactions in a block. In this equation, $\frac{E[T_{\text{off}}]}{b_t}$ represents the number of lost blocks;

(iii) D_{pun}, a random variable that models the delay for s to punish malicious behavior after detecting it. As punishment incurs publishing a transaction in the blockchain, the distribution of D_{pun} follows the well-known Poisson distribution found by Nakamoto [17] with expected value

$$E[D_{\text{pun}}] = n_c b_t, \tag{2}$$

where n_c is the number of confirmations it takes for a transaction to be valid and b_t is the block time;

(iv) Δ, a random variable that estimates the relative bias of each payment channel in the network. For empirical data, we calculate the bias of each channel in the dataset using the equation

$$\Delta_i = \left|\frac{bal_u - bal_v}{bal_u + bal_v}\right| \forall \langle u, v\rangle \in \mathbb{E}(t), \tag{3}$$

where i is the channel index and bal_u and bal_v are the initial balances of each party. Although the actual balances are private, the bias Δ_i serves as an estimation of how payments are flowing in the channel, since we expect most flows to occur from nodes with more capacity to nodes with less capacity.

Finally, each parameter composes the equation of W:

$$W = (T_{\text{off}} + D_{\text{det}} + D_{\text{pun}})(1 + \Delta). \tag{4}$$

The rationale behind our definition is that the lock time window W must be at least $T_{\text{off}} + D_{\text{det}} + D_{\text{pun}}$, otherwise the victim cannot recover and punish the attacker on time. Then, we proportionally increase the lock time window by a factor of Δ to improve the security of biased channels, since we expect them to be more vulnerable. Note that because T_{off}, D_{det}, D_{pun}, and Δ are either model distributions or real statistical data, W is in fact a random distribution. The actual window size to be selected by a user depends on what level of security she/he wishes to adopt for her/his use case. Users who invest heavily in the channel should select higher thresholds to avoid great losses and users who are willing to risk can select smaller thresholds to provide token liquidity.

V. Prototype Analysis and Results

We use real data from the Lightning Network [5] and from mobile broadband connections [13], [14] in our prototype since they are the most widely adopted technologies today [2],

[2]The block time is the average time it takes to mine a block in the system.

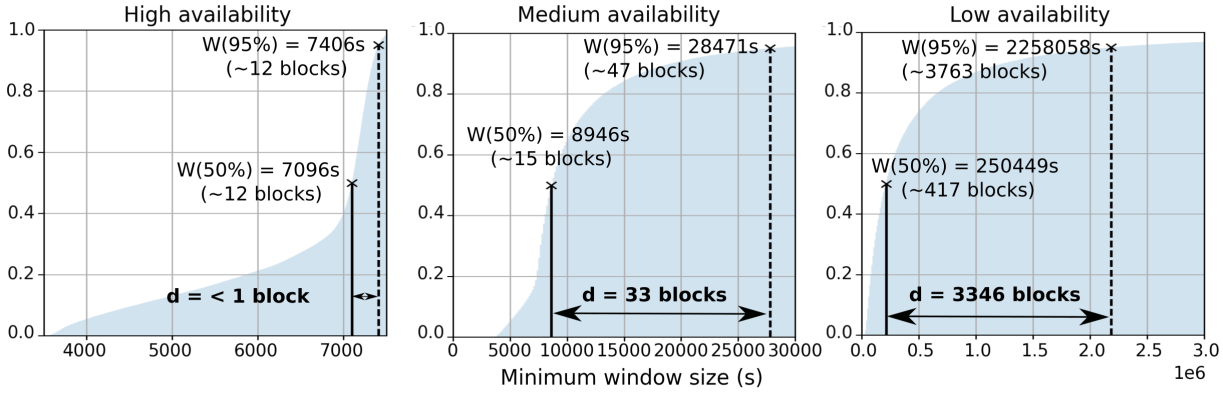

Fig. 6: Lock time window sizes for all levels of availability with 6-block confirmation. When the availability is high, the distance d between the 50% and 95% thresholds remains below one block time, which indicates a small window is safe for most users. For medium and low availability, the distance increases significantly and forces the user to select a time window that better fits her/his security and delay needs.

[18]. However, the methodology we propose is agnostic to blockchains, communication protocols, and PCN topologies. It suffices to estimate the parameters described in Section IV to find a safe time-window value that addresses any specific use case. We provide the code of our implementation on GitHub[3].

A. Evaluation Setup

We create three scenarios based on real availability measurements of mobile broadband devices to estimate the distribution of T_{off}. For the high-availability case, we use the downtime and packet loss distributions of MBB connections as measured by Elmokashfi *et al.* [14]. In their work, more than 90% of connections use 4G technology and the average downtime of a connection is 86.4s per day. The paper from Baltrunas *et al.* [13] serves as reference for the medium-availability case. The work measures the availability of mobile broadband connections that use 3G as the default technology and shows that the downtime can last for a few hours every day. Lastly, we simulate a low-availability scenario by shifting the medium-availability downtime distribution to the right by the average distance between the high-availability and medium-availability downtime distributions. This yields an average downtime of about one week. By simulating three roughly symmetrical scenarios based on real data, we can predict how different levels of availability impact the minimum lock time window. This could be extended to real-world device data of any kind.

For the detection delay D_{det}, we set the parameters as $n = 3$, $b_t = 600s$, and $b_s = 10Mb/s$. $n = 3$ represents the minimum number of different nodes to request blocks to in case one node is faulty. b_s and b_t follow the average block time and block size of Bitcoin, respectively. $E[T_{\text{off}}]$ is calculated according to the corresponding scenario and the download rate d is set using previous MBB evaluations: $d = 30Mb/s$ for high availability, $d = 2Mb/s$ for medium availability, and $d = 1Mb/s$ for low availability [13]. The average number of confirmations $n_c = 6$ for the punishment delay D_{pun} follows the 6-confirmation rule proposed by Nakamoto in Bitcoin [17].

We extract the values of Δ from LNChannels[4], an open-source tool that offers a data set of the Lightning Network. We download the channel balances from all closed channels since the beginning of the network and calculate the normalized bias Δ_i of each channel according to Equation 3. Figure 5 depicts the Δ distribution. Firstly, we observe a heavily asymmetric trend of payment flows, which confirms the token theft problem is not exclusive to WPCNs. Hence, adopting minimum lock time windows that depend on channel bias may fit a wide range of PCN implementations. Secondly, the Lightning Network implementation causes a gap around the 99% percentile because it defines a minimum payment amount `dust_limit_satoshis` that, if not met, converts the transaction into channel fees [1]. This amount prevents parties of paying when the channel is almost depleted.

Finally, we evaluate W through thresholds that correspond to the necessary value for a device to punish an attacker. A user that adopts $W(p)$ obtains p probability of being safe and assumes $(1 - p)$ probability of being attacked successfully. We use $W(50\%)$ as a reference for an unsafe threshold and $W(95\%)$ for a safe threshold, and measure the trade-off by calculating the distance d between the two thresholds. Short distances mean no significant gain from adopting a smaller window, while long distances mean the user should carefully select the value of W according to her/his needs. Figure 6 depicts the cumulative density functions for the minimum window sizes of all scenarios. The thresholds $W(p)$ are equivalent to the percentiles of the distribution of W.

B. Analysis and Discussion

In the high-availability scenario of Figure 6, 4G connectivity allows devices to be safe from attacks even with short time windows. The distance of less than one block between $W(50\%)$ and $W(95\%)$ demonstrates that increasing W to a secure level

[3]We will provide the code at: https://github.com/gfrebello/pcn-time-window in case of acceptance of the paper.

[4]Available at https://ln.fiatjaf.com/

generates no significant delay, so devices with good connectivity should adopt the safest W possible. The result also confirms our assumption that good connectivity profiles present in most traditional PCNs can effectively mitigate token theft.

The trade-off between security and efficiency becomes significant in the medium-availability scenario. The distance of 33 blocks between $W(50\%)$ and $W(95\%)$ corresponds to an increase of 5.5 hours in return delays for the party that closes the channel. T_{off} becomes the dominant parameter in Equation 4. The results indicate that a user with 3G connectivity should define minimum lock time windows of at least a few hours to reduce the probability of attacks; otherwise, attackers with better connectivity can easily exploit them.

The low-availability scenario demonstrates that users with low connectivity should either select W values in the range of days to weeks or use the main blockchain to transact. Delays in such order of magnitude may be economically worthwhile if the fees to publish transactions in the blockchain are too expensive for the user. However, more than 550 transactions could be published within the distance of 3346 blocks, which indicates the time window W may not be the most efficient countermeasure for devices that remain offline for long periods. Instead, we should adopt W with other security features, such as heavier punishment for attackers.

VI. Related Work

Several works propose adaptations of traditional PCNs for mobile devices. Kurt *et al.* propose LNGate, a lightweight protocol for IoT devices to use the Lightning Network [5] via untrusted gateways [8]. Hannon *et al.* propose a similar protocol and demonstrate its security and fairness using game theory [9]. Robert *et al.* propose an integration of the Lightning Network with existing large-scale IoT ecosystems [10]. Mercan *et al.* present alternative lightweight PCN implementations that focus on reducing the computational needs for mobile devices [19]. The works, however, focus on adapting the Lightning Network to IoT scenarios. We propose a new PCN design and a security feature that is agnostic to PCN implementations.

Other works analyze the security of traditional PCNs. Tochner *et al.* formulate topology-based attacks that aim to disrupt the routing protocol of traditional PCNs [20]. Erdin *et al.* compare the security and privacy of several PCN implementations and identify emerging attack vectors [2]. The works neither discuss attacks in PCNs with resource-constrained devices nor present efficient countermeasures for wireless environments. To the best of our knowledge, our work is the first to propose an architecture for wireless PCNs, formulate the token theft attack and propose a time window analysis as an efficient countermeasure.

VII. Conclusion

We proposed a hybrid architecture that allows resource-constrained wireless nodes to issue off-chain transactions and analyzed the impact of the token theft problem in such environments. Our main findings show that the problem is not exclusive to wireless PCNs and that our solution may work with traditional PCNs as well. A countermeasure based on minimum lock time windows is efficient when the devices present high to medium availability. For devices with low availability, the minimum lock time window becomes so significant that it may be better to publish the transactions directly in the blockchain.

In future works, we will investigate other types of countermeasures to token theft, such as more efficient punishment mechanisms and time-varying lock time windows.

References

[1] J. Poon and O. Osuntokun, "BOLT #2: Peer protocol for channel management," 2021, Last access: 12 October 2021. [Online]. Available: https://github.com/lightningnetwork/lightning-rfc/blob/master/02-peer-protocol.md

[2] E. Erdin, S. Mercan, and K. Akkaya, "An evaluation of cryptocurrency payment channel networks and their privacy implications," *arXiv preprint arXiv:2102.02659*, 2021.

[3] L. Lys, A. Micoulet, and M. Potop-Butucaru, "Atomic cross chain swaps via relays and adapters," in *3rd CryBlock*, 2020, pp. 59–64.

[4] Nasdaq, "Bitcoin lightning network sees massive growth and international adoption," 2021, Last access: 12 October 2021. [Online]. Available: https://www.nasdaq.com/articles/the-bitcoin-lightning-network-sees-massive-growth-international-adoption-2021-08-23

[5] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016, Last access: 12 October 2021. [Online]. Available: https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lightning-network-paper.pdf

[6] brainbot labs Est., "The Raiden network: Fast, cheap, scalable token transfers for ethereum," 2020, Last access: 12 October 2021. [Online]. Available: https://raiden.network/

[7] S. Geissler, F. Wamser, W. Bauer, M. Krolikowski, S. Gebert, and T. Hoßfeld, "Signaling traffic in internet-of-things mobile networks," in *2021 IFIP/IEEE IM*, 2021, pp. 452–458.

[8] A. Kurt, S. Mercan, O. Shlomovits, E. Erdin, and K. Akkaya, "LNGate: Powering IoT with next generation lightning micro-payments using threshold cryptography," *arXiv preprint arXiv:2105.08902*, 2021.

[9] C. Hannon and D. Jin, "Bitcoin payment-channels for resource limited IoT devices," in *IEEE COINS*, 2019, pp. 50–57.

[10] J. Robert, S. Kubler, and S. Ghatpande, "Enhanced lightning network (off-chain)-based micropayment in iot ecosystems," *Future Generation Computer Systems*, vol. 112, pp. 283–296, 2020.

[11] S. Roos, P. Moreno-Sanchez, A. Kate, and I. Goldberg, "Settling payments fast and private: Efficient decentralized routing for path-based transactions," *arXiv preprint arXiv:1709.05748*, 2017.

[12] V. Sivaraman, S. B. Venkatakrishnan, K. Ruan, P. Negi, L. Yang, R. Mittal, G. Fanti, and M. Alizadeh, "High throughput cryptocurrency routing in payment channel networks," in *17th USENIX NSDI*, 2020, pp. 777–796.

[13] D. Baltrunas, A. Elmokashfi, and A. Kvalbein, "Measuring the reliability of mobile broadband networks," in *14th ACM IMC*, 2014, pp. 45–58.

[14] A. Elmokashfi, D. Zhou, and D. Baltrünas, "Adding the next nine: An investigation of mobile broadband networks availability," in *23rd ACM MobiCom*, 2017, pp. 88–100.

[15] G. Malavolta, P. Moreno-Sanchez, A. Kate, M. Maffei, and S. Ravi, "Concurrency and privacy with payment-channel networks," in *2017 ACM SIGSAC CCS*, 2017, pp. 455–471.

[16] G. F. Camilo, G. A. F. Rebello, L. A. C. de Souza, and O. C. M. Duarte, "A secure personal-data trading system based on blockchain, trust, and reputation," in *3rd IEEE Blockchain*. IEEE, 2020, pp. 379–384.

[17] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, Last access: 12 October 2021. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[18] OECD, "Mobile broadband subscriptions indicator," 2021, Last access: 12 October 2021. [Online]. Available: https://doi.org/10.1787/1277ddc6-en

[19] S. Mercan, E. Erdin, and K. Akkaya, "Improving sustainability of cryptocurrency payment networks for iot applications," in *IEEE ICC Workshops*. IEEE, 2020, pp. 1–6.

[20] S. Tochner, A. Zohar, and S. Schmid, "Route hijacking and dos in off-chain networks," in *2nd ACM AFT*. ACM, 2020, p. 228–240.

This paper was funded by CNPq, CAPES, FAPERJ and FAPESP (18/23292-0, 2015/24514-9, 2015/24485-9 2014/50937-1).