Card Payment Protocol for Cryptocurrencies with Payment Channel Network

ANUPA DE SILVA, University of Galway, Galway, Ireland
SUBHASIS THAKUR, University of Galway, Galway, Ireland
JOHN BRESLIN, University of Galway, Galway, Ireland

Cryptocurrency, despite the upsurge as a speculative investment, is still a long way from being people's money. The extreme technicality poses a significant barrier for the general public to adopt it as a medium of exchange. Therefore, simplifying the payment process is a predominant necessity; for example, facilitation in adopting conventional payment instruments can bring convenience to both merchants and consumers. This article proposes a novel cryptocurrency card payment protocol leveraging the Payment Channel Network (PCN) concept, facilitating high throughput and affordable transactions. Our design is based on the Bitcoin-backed Lightning Network (LN) with adjustments to make it executable by a smart card. It also preserves the decentralized nature of cryptocurrencies, reduces operational costs in several ways, and allows instant settlement for the recipients as compared to both conventional and cryptocurrency card payment systems. Our game theoretical analysis, where we model the engagement between the cardholder and the card agent as a long-run extensive form game, attests that they accord with the protocol without experiencing any honest loss under pragmatic conditions. We also discuss the privacy features compared to conventional card payments and LN. The protocol can function independently, without the need for trust, and can also be regulated for those seeking government mediation. This approach can potentially revolutionize the payment landscape by allowing the public to use cryptocurrency for everyday transactions conveniently and allowing existing cryptocurrency holders to conduct affordable micropayments.

CCS Concepts: • **Applied computing** → **Electronic funds transfer**; **Secure online transactions**; • **Networks** → *Network protocol design;*

Additional Key Words and Phrases: Payment card, smart card, payment channel network, cryptocurrency, bitcoin, game theory

ACM Reference Format:
Anupa De Silva, Subhasis Thakur, and John Breslin. 2024. Card Payment Protocol for Cryptocurrencies with Payment Channel Network. *Distrib. Ledger Technol.* 3, 3, Article 19 (September 2024), 30 pages. https://doi.org/10.1145/3653681

1 INTRODUCTION

The concept of cryptocurrency emerged as an attempt to decentralize the central control held by elite financial institutes. However, its decentralized nature and high volatility soon captivated the investment community as a

This publication has emanated from research supported in part by a grant from Science Foundation Ireland and the Department of Agriculture, Food and the Marine on behalf of the Government of Ireland under Grant Number SFI/16/RC/3835 (VistaMilk), and also by a grant from SFI under Grant Number SFI/12/RC/2289_P2 (Insight). For the purpose of Open Access, the author has applied a CC BY public copyright license to any Author Accepted Manuscript version arising from this submission.
Authors' address: A. De Silva, S. Thakur, and J. Breslin, University of Galway, University Rd., Galway, Galway, Ireland, H91 TK33; e-mails: anupa.shyamlal@insight-centre.org, subhasis.thakur@insight-centre.org, john.breslin@universityofgalway.ie.

ACM 2769-6472/2024/09-ART19
https://doi.org/10.1145/3653681

high-risk but high-return asset. The same demand overshadows its adoption in day-to-day transactions. Despite its decentralized and privacy-preserving nature, the general public found no added advantage in using cryptocurrencies over traditional currencies [21]. Apart from the technological factors, transaction convenience matters in adopting such a technology. Card-based cryptocurrency payments are one step toward gaining that convenience for the general public. Conventional card-based payments have achieved immense popularity in the last two decades, even though there was resistance from consumers and merchants to adopting the technology at the beginning. Similarly, cryptocurrencies were introduced for payment cards a decade ago, but statistics attest that their usage is minimal [19]. Regardless, the solution only complements the existing card payment architecture rather than the decentralized nature. Preserving the decentralized nature is impossible because payment cards expect near real-time transactions. Cryptocurrency transactions do not inherently support instant settlement, which left no option other than using centralized wallets to achieve swiftness of payments.

PCN is a promising second-layer solution for the scalability problem in cryptocurrencies. With its ability to facilitate low-latency and low-cost payments while maintaining the same level of security as the underlying layer, PCN is well-suited for retail environments. However, certain conditions, such as the need for users to be online to prevent double spending, computational cost, and refill cost, could be challenging for non-tech users or those with limited computational resources. In this article, we manipulate features of retail transactions to control the PCN's conditions and customize the PCN payment protocol to fit inside a smart card. We aim at enabling card-based cryptocurrency payments for the general public, including existing cryptocurrency holders who wish to make low-fee micropayments. Unlike existing card payment systems, including cryptocurrency-based ones, we preserve the decentralized nature of the proposed design, resulting in cost savings and scalability to accommodate mass adoption. To the best of our knowledge, this marks the first attempt to create a decentralized card payment protocol, and we hope it will lead to greater accessibility and convenience in the world of cryptocurrency payments.

The proposed protocol is based on a Bitcoin-based PCN, Lightning Network, and smart cards such as Java cards. We developed a proof of concept[1] using the Java Card simulator, jCardSim.[2] However, its application can be further extended to other devices and cryptocurrencies. We analyze the proposed protocol using Game Theory and prove that there is no honest loss or incentive for the participants to deviate from expected behavior if participants satisfy several realistic conditions. The rest of the articles are outlined as follows. In the first Section 2.1, we discuss about payment cards, their operations and historical sentiments toward adoption. Next Section 2.2, we present PCN as a potential technology to enable cryptocurrency payments along with in detail the description of LN operations. With insights from those two chapters, we design our proposed protocol in Section 3 and analyze the protocol by modeling the card and the agent interaction as a game in Section 4. Finally, we evaluate the overall protocol in terms of utility and security.

2 BACKGROUND

2.1 Card Payments

Retail businesses have witnessed a growing preference for card payments as a convenient mode of payment. Unlike cash, the consumer only needs to hold the payment card and present it to the **point of sale** (**POS**) machine to make the payment. Then, a set of regulated third-party entities ensures a secure transaction toward the merchant. The widely adopted framework for this is the *four-corner model* where consumers and merchants connect with their respective banks, and banks are connected with each other, forming the four corners. A detailed process can be found in Appendix A. The transactions are near real-time; however, the system differs from clearing the funds for the merchant, which will be performed batch-wise. The merchant receives the funds

[1] https://github.com/anupasm/CryotoCardPay

[2] jCardSim (https://jcardsim.org) is an open source simulator which implements Java Card, v.2.2.1/2.

in their bank account after deducting the fees. Along the process, the merchant bears the fees imposed by the external entities for facilitating the transaction. Generally, the fee is charged as a portion of the transaction value, ranging from 1% to 5%. In some countries, charging the fee from the customer is prohibited; for example, in Europe, customer surcharges are banned under the **Payment Services Directive** (**PSD2**). Still, the price of the good or service may comprise the transaction fee.

2.1.1 History. The payment card industry is a market that requires both consumer and retailer participation. It is a "two-sided" market where consent from both parties is essential for its success. However, previous studies exemplify that the perception toward card payments was scarcely positive even 30 years after their introduction. The survey conducted by the researchers of [16] unveils how it failed to build confidence in consumers and retailers at that time. Even though those surveys had geographical limits, the reasons raised were insightful. The users found minimal comparative gain from card payments compared to cash [23]. Although card payments provided novelty and convenience, they also came with transaction costs and privacy concerns. Additionally, new infrastructure had to be introduced (e.g., POS machines, payment network). Nevertheless, after many reforms [5], it eventually gained the intended popularity to a level that could displace cash payments.

2.1.2 Card Payments for Cryptocurrency. In the past decade, cryptocurrencies gradually enhanced their capabilities to match mainstream mediums of transactions. One such step was integrating a facility to make cryptocurrency transactions through payment cards. In 2014, Xapo and Mastercard jointly issued the first cryptocurrency payment card backed by Bitcoin, and there are currently several commercial solutions associated with the existing card schema. They align with the prevailing payment card system rather than adhere to the decentralized nature. Analogous to the conventional method, when the user initiates the transaction, POS connects to the user's wallet, which is held by a centralized entity instead of the issuing bank. The main reason is not the compatibility issue with the existing design. Cryptocurrencies are inherently slow in settling transactions despite the emergence of new currencies with low latency. Even intermediaries (payment processors and gateways) can only handle the situation partially with a possible loss, as users can double-spend on another entity during the confirmation time. Therefore, centralized wallet management is inevitable.

Nevertheless, according to the Blockchain Trilemma, we can achieve low latency only by relinquishing security or a decentralized nature. For example, GRAFT [7] is designed with increased processing time to gain the expected convenience for card payments; however, the design relies on trusted super nodes for transaction processing. There are also theoretical attempts to solve the payment card-related latency issue in the literature. For example, Asplund et al. [2] propose a design to enable in-store payments using Bitcoin with low latency. The proposed solution directly contacts the Bitcoin network through the POS. However, this method assumes that the cardholder cannot access the smart card's private keys to ensure they will not double-spend.

We find studies carried out around Europe to examine the sentiments of consumers and retailers toward cryptocurrency. At the end of 2014, Baur et al. [3] survey the consumer's perspectives regarding Bitcoin adoption in general, where its primary outcome highlights their concern regarding Bitcoin's level of ease of use. Later, in 2016, Jonker [10] surveyed retailers' opinions on adopting cryptocurrency as a payment medium. According to the study, retailers were enthusiastic about the new technology and attracting new customers, rather than the key selling points of cryptocurrency, privacy, independence, and security. Also, he stresses the need for more consumer demand. Therefore, he concludes that a substantial improvement in the cryptocurrency acceptance rate will not be sought immediately. Apparently, at this initial stage, after introducing card payments for cryptocurrencies, both consumers and retailers wanted more from cryptocurrency payments, although they were hopeful about the future. However, it is questionable whether it has improved even after the sudden hype occurred in 2018. The survey [19], conducted in 2022 using small and medium-scale enterprises, demonstrates that they adopted cryptocurrency as a marketing strategy but did not gain a consistent increase in sales, which again highlights the lack of demand. It implies that cryptocurrency remains in its role as a speculative investment [4] ever since it was introduced.

Mediation from regulatory bodies can also positively affect the general public's confidence toward cryptocurrency, according to the survey [1]. However, the extent to which a regulatory body can intervene in a decentralized system such as Bitcoin remains debatable. Nevertheless, we can see a tendency toward cryptocurrency by governments. Either they have implemented or intend to implement a digital currency, Central Bank Digital Currency, for general purposes [9].

2.2 Payment Channel Networks

PCN is a second-layer solution that can operate on top of a blockchain, enabling low latency, high throughput, and inexpensive transfers through a new network of payment channels anchored on the blockchain. For example, experts predict that a widely adopted LN can process transactions in millions per second while Bitcoin can only process seven transactions per second, and Visa can handle 65,000 transactions per second. PCN's payment channel is analogous to opening a joint bank account between two parties who mutually agree regarding how they spend the deposited value. The benefit of creating a channel is to perform theoretically unlimited transactions between the two participants without on-chain fees except for the first and the last transactions. It can extend to the rest of the world by creating a network of payment channels, allowing transactions between parties without pre-established channels.

2.2.1 Lightning Network. Many PCN implementations are already in production or in theoretical proposals, which are detailed in Appendix B. Lightning Network is the first and the most prominent PCN implementation on Bitcoin, developed by Poon et al. [17]. Here, we elaborate on the LN functionality in the context of a card payment environment, where a cardholder connects to an agent and establishes an LN channel. The channel lifetime can be divided into three stages: setup, update, and close, as given below.

Setup: Let us assume a **cardholder/smart card** (**SC**) and a **card agent** (**CA**) intend to establish a channel with initial funding of $c_0(> 0)$ from the cardholder and $a_0(= 0)$ from the agent. Initially, both parties exchange their expectations (e.g., self-delay), capabilities (e.g., a maximum number of pending transactions), and security keys (e.g., public keys for funding, base points, and first commitment transaction). Then, the cardholder creates the funding and refunding transactions. A funding transaction is a 2-by-2 MultiSig transaction requiring both parties' signatures to spend. The cardholder must create the refund transaction (the first commitment transaction) to avoid his funds being stuck on the chain forever in the absence of the agent. The cardholder obtains consent for both transactions from the agent. Finally, the cardholder posts the funding transaction, and both wait until the funding transaction reaches the necessary number of confirmations to start the channel. The cardholder already retains the refund transaction if, in case, the cardholder disappears at this stage.

Update: After opening the channel, both parties have an unspent MultiSig transaction on-chain. Hence, the cardholder and the agent can pay each other by spending the on-chain transaction. However, they refrain from posting the transaction until they intend to close the channel. They can keep updating the balances before the sending amount exceeds the sender's balance. These updates are called commitments, which are ready to post signed transactions.

The sender can also make payments to the node with no direct channel by routing the payment through already connected third-party nodes. Therefore, some outputs are in a pending state during the routing process. LN separates transaction outputs for the pending transaction from the already settled balance, and there can be multiple such outputs in a transaction. Each output includes a hash and time locks to make it redeemable only for those who know the hash lock's key within the specified time. This type of contract is called a **Hashed Time Lock Contract** (**HTLC**). The lock is identical for all the participants in the routing path, including the sender and the receiver. Now, if the only person who knows how to release the hash lock is the payment recipient, all the intermediate nodes must reach the receiver with the payload amount to release their incentive fee. Therefore, the intermediate's strategy is to pass the payload amount to the next node, keeping the incentive fee and making it redeemable only if the next node knows the secret. Once they realize the secret, they can release the previous lock, recover the payload, and unlock the incentive fee.

Table 1. Agent's Version of the Transaction Outputs

Output	Value	Time	Requiremet(s)	Recipient
CA Balance	$a_i - x_2$	Immediately	SC Revokes	SC
		CSV Delay	CA Signature	CA
SC Balance	$c_i - x_1$	Immediately	SC Signature	SC
Offered HTLC	x_2	Immediately	SC Revokes	SC
		Immediately	Preimage, SC Signature	SC
		CLTV Delay	Stage 2 Tx	SC/CA
Received HTLC	x_1	Immediately	SC Revokes	SC
		Immediately	Preimage, Stage 2 Tx	SC/CA
		CLTV Delay	SC Signature	SC

Close: There are three possibilities for closing a channel: unilateral close with the latest or stale state and collaborative close. Unilateral close is essential in cases where the counterparty does not respond. However, unilateral close enables any states to be published, including the discarded ones, as the previous commitments are still acceptable on the chain. Assume that someone decides to spend the funding outputs with an old commitment that does not reflect the latest state and get confirmation from the miners. Then, it is impossible to reverse the transaction as the funding output is no longer spendable, and Bitcoin's blockchain is immutable. Therefore, LN discourages cheating with old commitments by punishment rather than prevention. The first step in penalizing the old commitment is identifying the cheater, which cannot be achieved if both parties share the same transaction. Hence, LN uses asymmetric transactions. Also, LN embeds a revocation key inside the output of the withdrawing party, enabling someone who knows the secret to the revocation key to redeem the total output immediately. When both move to the next state, each reveals the revocation key's private key. By doing so, the revealing party agrees to accept the punishment of losing total balance if they post a stale transaction. LN also imposes a delay for self-withdrawal, creating room to challenge the posted transaction.

Example: Say the current state is (c_i, a_i), and the cardholder sends an amount of x_1 to the agent and receives x_2 from the agent simultaneously. Here, at least one party is involved in transaction mediation, which is why the transaction is bi-directional. Table 1 shows the agent's version for this scenario, who will receive each output, and under what conditions.

When the agent publishes the transaction, the cardholder can immediately spend his already settled balance using her signature. However, the agent's portion of the transaction may not have been posted with the latest state. Therefore, the agent's portion is delayed by a relative delay (i.e., CSV delay[3]) to allow the cardholder to challenge the transaction with the previously provided revocation key. The same applies to the rest of the outputs. In this example, the agent sends x_2 to the cardholder, and it is presented in the offered HTLC section. Again, the cardholder can redeem the output immediately if he knows the secret. If not, the agent can utilize a second-stage transaction to make a refund after an absolute delay (i.e., CLTV delay[4]). The reason for a second-stage transaction is to avoid a race condition that may incur using CSV and CLTV delays together. CSV delays are generally lengthier than CLTV, which can prolong up to weeks depending on the protection needed for the channel capacity. The lengthier it is, there is more time challenge. Conversely, CLTV time should be as brief as possible to achieve the finality of the payment (the preceding node depends on the secret revealed in HTLC). If we consider the offered HTLC by the agent, she should be able to reclaim the funds after some time (i.e., refund). However, we should also simultaneously prevent him from revocation. If both delays are at the same stage, the

[3]CheckSequenceVerify is relative timelock, which causes the embedded script to fail until the specified time relative to when the transaction got mined.

[4]CheckLockTimeVerify timelock is an absolute time lock which causes the embedded script to fail until the specified timestamp is reached.

shorter one (i.e., CLTV) creates no effect. Hence, LN first allows the cardholder to seize the output after the HTLC timeout and then claim the funds using a MultiSig transaction consisting of a revocation window. The same happens in received HTLC. The agent first has to seize the output before HTLC timeout by giving out the preimage, then use the second stage MultiSig transaction to claim the funds where the cardholder can challenge the claim.

2.2.2 Cost of a Node. Maintaining a PCN node is less practical for an ordinary user. The primary concern for an independent node operator is the need for an online presence to detect malicious behavior from peer nodes while keeping the blockchain up-to-date. Instead, they can delegate the surveillance to a third-party watchtower. LN has inbuilt support for this purpose while preserving the user's privacy, but the user must report every transaction to the watchtower, which is less convenient. Further, the user who initiates a transaction must determine a path to the recipient and compute encrypted messages for the nodes along the way. Therefore, pre-initialization of the transaction is a computationally heavy task for users with lightweight clients. Peers also need to exchange signed commitments in a mandatory sequential manner, which results in a delay until the response returns to the node to settle balances. These signature generations, encryptions, waits, and sequential message passing make the transaction process more complex than other digital transactions. Lastly, when the account balance drains out, the required top-up process is not straightforward in PCN. Either the user needs to renew the contract, bearing a costly on-chain fee, or route a payment toward the node itself through other nodes.

2.2.3 PCN for Retail. In the retail business, retailers and consumers directly engage with each other to sell goods or services in small quantities. As a result, these monetary transactions are typically valued in small amounts and must be handled in real-time for the convenience of both parties. Current payment cards successfully achieve real-time transactions with their centralized design. However, retailer settlements take time, and facilitation is currently under a duopoly, with questions being raised regarding the fairness of the fee. Cryptocurrencies, on the other hand, do not support instant settlement. Therefore, cryptocurrency-enabled payment cards are mere adoption of existing card payment infrastructure rather than being able to complement the decentralized nature of the cryptocurrency. PCN can play a role here with its low cost and low latency transfer capability, as discussed in Section 2.2. It satisfies both stakeholders' main demands: convenience for the consumer and low cost for the retailer. Additionally, PCN can eliminate the requirement of having a payment schema between the consumer's agent (i.e., issuing bank) and retailer's agent (i.e., acquiring bank), which can further reduce the transactional cost. However, PCN needs to be compatible with card payment technology, with limitations discussed in Section 2.2.2. Nonetheless, the unidirectional nature of most retail payments, flowing predominantly from the consumer to the retailer except for top-ups, can be leveraged to overcome PCN's online presence requirement. Further, smart cards are computationally constrained devices without direct access to the external world, whereas PCN expects communication and performing cryptographic operations.

3 PROPOSED CRYPTOCURRENCY PAYMENT CARD PROTOCOL

In this section, we elaborate on the design of our cryptocurrency-based card payment protocol for retail payments. The design is for but not limited to smart card devices and LN. We begin introducing the network setup, which aligns with the current payment card network. Then, we present the essential segments required for the protocol, which we will introduce toward the end of the text.

3.1 Network Setup

We follow the Four Corner Model of the existing card schema, which is less restrictive than the three-party model. It is also the cheapest model to reach a broader audience in the PCN context. Nevertheless, the PCN model can theoretically accommodate any number of intermediaries or no intermediary at all when forming the path of transaction execution.

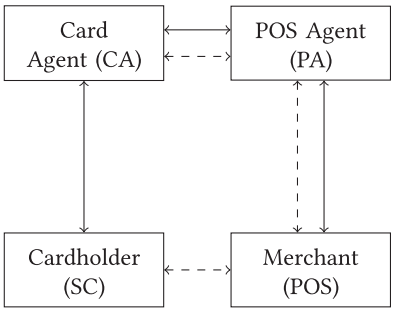


Fig. 1. The overall architecture of Cryptocurrency Card Payment based on the four-corner model. ⟶ and ⇢ represent payment and communication channel formation, respectively.

Figure 1 illustrates a four-party set-up in the context of payment execution. The consumer and merchant are two entities that have already established the payment channels with their respective agents. They connect to the network through their devices, SC, and POS machine, respectively. We also assume a channel between two agents; if not, another intermediary(ies) must be involved in making the connection. When the cardholder presents the smart card to the merchant's POS, they form a communication channel, and we expect it to be consistent until payment gets executed. Additionally, communication channels between agents and POS with its agent are required. Note that the smart card and its issuer have no communication channel in between, even though they have formed a payment channel.

3.2 Channel Design

The proposed protocol requires modifications in LN's building blocks (refer to Section 2.2.1) to comply with the limits of smart cards. Therefore, we construct new time delays, alter key usage, and revise transaction scripts. Additionally, in contrast to the LN protocol, the smart card decides the lock, which will be disclosed to POS to mark the success of the payment to POS. This change prevents POS from fraudulently pretending to be a failure of payment delivery as the smart card depends on POS to communicate with its agent. Moreover, the engagement between the cardholder and his agent always includes an unsettled HTLC portion until a top-up. Finally, we also limit the key rotation and diversification to reduce the computation overhead on the smart card.

3.2.1 Keys. In the design, we use only four types of keys for funding, payment, and revocation. The funding key (FundPoint) is one of the keys inside the multi-signature contract of the channel's funding transaction. Its lifetime is the same as the funding contract, and during that period, it is the signing key of the remote and local commitment transactions in commitments. All the payments are made to the payment key PayPoint. Both the smart card and its agent must discard the PayPoint at each top-up and generate new ones. Additionally, PayPoint establishes secure communication between the smart card and the agent. The revocation base key (RevBasePoint) is blinded with a randomly generated commitment key (ComPoint) to derive a unique revocation key RevPoint. Blinding prevents the revocation key to be used by the commitment point owner. The process is similar to LN, which employs the commutative property of elliptic curves based on Abelian groups.[5] The difference is that the card agent's RevPoint is unchanged for all transactions before a top-up. The smart card must randomly generate a ComPoint at each commitment so that revocation is unique for each update. Both must renew the revocation base keys at each top-up.

3.2.2 Time Delays. Timelocks play a vital role in the finality of PCN transactions. Essentially, they ensure that a transaction cannot be included in a valid blockchain block until the time lock has expired. For HTLCs, it ensures

[5] https://en.wikipedia.org/wiki/Abelian_group

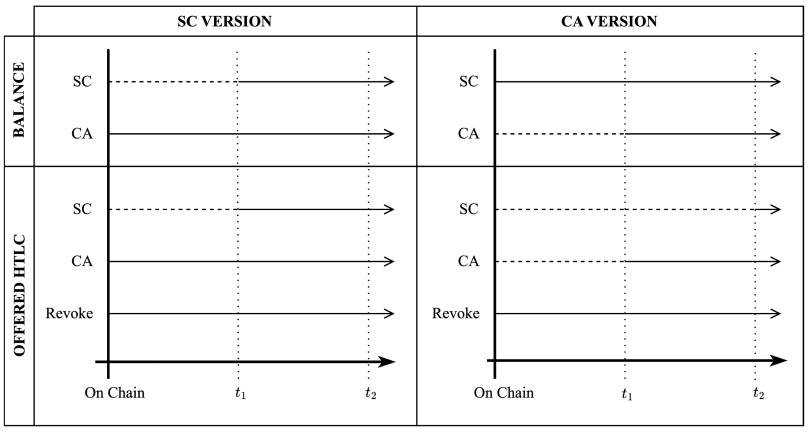


Fig. 2. Ability to redeem outputs of SC and CA versions of transactions over time. Solid lines indicate redeemable time, and respective parties cannot redeem during dashed lines.

that HTLC gets resolved within a pre-specified time using absolute time locks (i.e., CLTV). PCN also mandates a decreasing HTLC expiry along the routing path toward the receiver such that nodes close to the receiver obtain the preimage earlier than the rest. This decreasing order forces them to resolve their HTLC either on-chain or off-chain before the preceding node, which is how the preimage propagates toward the sender. For example, if the POS agent decides to publish on-chain, this approach ensures that the smart card agent has enough time to obtain the preimage from the chain and settle the HTLC, even on the chain.

LN specification[6] uses different stages to avoid a possible race condition between the CSV and CLTV timelocks. In the context of card payments, the card agent always positions at the end of the revealing path as our design does not expect the card agent to reveal the preimage to the smart card as a verification (see Section 3.5). Hence, the purpose of the absolute delay is pointless. Instead, we use another lengthier relative delay to achieve the exclusive redeem window for the receiver, eliminating the need for absolute timing restrictions. As a result, both parties can post their designated transactions at any time but with restricted outputs. This manipulation significantly abates the overhead on the smart card. In the normal process, the smart card must have constructed four transactions, two signatures, and two verifications. In contrast, it is now reduced to two transactions and a single signature generation and verification. The network overhead is also declined accordingly. Furthermore, POS can let go of the smart card early as we do not occupy absolute expiry time. Absolute expiry imposes the receiver (i.e., the agent) to reconcile with the smart card to remove the HTLC portion at the end of each payment before the expiration time. Since we use only relative delays, the smart card can pull back without settling the balances. As an added advantage, the smart card does not need to be aware of the current block of the blockchain to set the absolute delays in the transaction or depend on a third party (i.e., POS).

Figure 2 shows how the withdrawal delays work for each transaction output. The same transaction carries the smart card's and agent's versions. Delays t_1 and t_2 are relative to when the respective party posts the transaction. As LN, the CSV delay t_1 imposes self-delay for the publisher. It can be seen in the already resolved balance portion and offered HTLC section. However, the offered HTLC portion in the agent's version differs from the rest. The agent is subjected to a redemption delay as a challenge window (i.e., Revoke) because she is publishing the transaction. Then, she is given a while to redeem the offered amount. If not, after t_2 time, the cardholder can reclaim the amount. Regardless of who posted the transaction, the card agent has an exclusive time window to

[6]https://github.com/lightning/bolts

Table 2. Smart Card's (SC) Version of the Transaction Outputs

Output	Value	Time	Requirement(s)	Recipient
CA Balance	a	Immediately	CA Signature	CA
SC Balance	$c - x$	Immediately	CA Revokes	CA
		CSV delay t_1	SC Signature	SC
HTLC	x	Immediately	CA Revokes	CA
		Immediately	Preimage, CA Signature	CA
		CSV delay t_1	SC Signature	SC

redeem the output if she knows the correct preimage. The cardholder can only redeem the output if the agent is unaware of the preimage, which is highly unlikely in the practical scenario.

3.3 Transactions

PCN payments are primarily routed through other nodes using HTLCs, which means that a peer's total balance is not static, as routed payments may eventually fail. Therefore, transaction outputs are twofold: already settled peer balances and pending outputs (i.e., offered or received HTLC outputs). These features are similar to those of a typical card payment scenario but are more simplified than ordinary PCN transactions. PCN accommodates multiple routed transactions per update and handles receiving and sending transactions identically. On the contrary, cardholders are mostly payment initiators in this context and occasionally receive payments as top-ups. Also, cardholders make payments one at a time, and we do not expect payment cards to mediate other transactions. Therefore, we can consider in and out payments separately when designing the transactions. Further, in both cases, the smart card is the first or the last in the routing path. However, smart cards have limited computational power and storage, and cardholders expect convenience from the payment experience that requires minimal engagement with the card.

We design the following simplified transaction for the card payment protocol adhering to the above limitations and requirements. Fundamentally, we handle sending and receiving scenarios separately. In the first part, we discuss how the sending transaction works in card payments. Notably, we reduce the number of transactions and stages in conventional PCN to a single transaction and stage. Then, we discuss receiving the transaction, which is the top-up transaction.

3.3.1 Payment Transaction. A payment transaction consists of three segments: two for self-balances and another for the payment transaction, which is in a pending state. In the beginning, as payments are always made from the card to its agent, the value of the new payment is deducted from the card's base balance and contained in an HTLC lock. The agent's balance remains the same. Then, the three outputs can be redeemed independently under different conditions. Here, we explain the outputs and respective conditions in tabular form, and their Bitcoin scripts can be found in Appendix D. Let the smart card initiate x payment, assuming (c, a) be the initial balance state of the cardholder and the agent, respectively. The preimage is the secret of the HTLC. Tables in 2 and 3 provide information on transaction outputs meant for the smart card and its agent, respectively. Each table contains information about when output can be spent, what is needed (i.e., keys to generate signatures and preimage) and who the recipients are.

In general, the output to the remote party is always immediately redeemable, provided the remote party's signature using PayPoint for the lock script in Listing 1. On the other hand, output to themself is redeemable only after a delay and protected by a revocation key. It creates a reasonable window for the remote party to challenge the posted commitment. For example, assume that the cardholder published an old commitment to the blockchain, which means the agent already knows the key for the RevPoint. Then, she can immediately claim the

Table 3. Card Agent's (CA) Version of the Transaction Outputs

Output	Value	Time	Requirement(s)	Recipient
SC Balance	$c - x$	Immediately	SC Signature	SC
CA Balance	a	Immediately	SC Revokes	SC
		After CSV Delay	CA Signature	CA
HTLC	x	Immediately	SC Revokes	SC
		CSV Delay t_1	Preimage, CA Signatures	CA
		CSV Delay t_2	SC Signature	SC

smart card's output by submitting the signature generated using that key. If not, after the t_1 delay, the cardholder can redeem the output using the smart card's PayPoint. Lock script can be found in Listing 2.

There are two versions of the HTLC script, one for the sender (cardholder) and one for the recipient (card agent). In the cardholder's version, the offered HTLC from the smart card is immediately redeemable by the agent, given the preimage and the agent's signature. Otherwise, the output should refund the smart card if the HTLC does not settle after a specified time. It must not be able to claim the refund before the expiry time, and at the same time, it must enforce a delay to allow the penalty window, as this is a self-refund. We handle this using CSV delay, which allows posting the transaction on-chain, but the output is not claimable until the CSV delay. Therefore, CSV delay is exclusive for the smart card agent to claim the HTLC. Note that absolute delays are not involved in the HTLC output, as the cardholder is the last in the transaction execution path. Listing 3 gives the Bitcoin script for the offered HTCL of the smart card.

The card agent may also initiate the transaction and claim the output using the preimage but with a delay. We set it as a CSV delay to prevent spending until CSV time reaches. If it is further delayed, the cardholder must be able to reclaim the funds. We occupy a second CSV delay t_2 ($> t_1$) to enable the smart card holder to redeem the transaction. Then, there are exclusive windows to challenge the output and claim the funds or else the cardholder can obtain the refunds. Listing 4 gives the Bitcoin script for the received HTCL of the agent.

Finally, both smart card and agent can build both versions of transactions (see the template in Listing 5) and generate and verify signatures during the transaction.

3.3.2 Top-up Transaction. If a smart card's balance is running low but the cardholder wants to continue using it, they have the option to initiate a transaction through the agent rather than renewing the channel on-chain. To do so, the smart card must connect to the agent using a card reader. We anticipate that top-ups will occur periodically but infrequently, creating a less restricted environment than a typical merchant payment. This allows the cardholder more time and external computational power to handle the original LN transaction.

3.3.3 Closing Transaction. Each update can be recorded on the blockchain, and the transactions mentioned above represent the finalization of the channel contract. It's worth noting that this is a unilateral close. Alternatively, if both parties work together to close the channel, they can remove the self-imposed delays, similar to the LN.

3.4 Provisioning the Smart Card

Provisioning a smart card involves generating a key and storing transaction-related information within the card. Cardholders can use a card reader to complete this step. Generating the funding transaction is simultaneously performed along with the process of provisioning, except that the cardholder initiates key generation within the card itself and retrieves the public keys, as explained in Section 3.2.1. Transaction generation does not need to start from scratch, as a significant portion of the transaction remains static for both the cardholder's and agent's versions. Therefore, this data can be saved during the provisioning phase and updated as needed, such as with

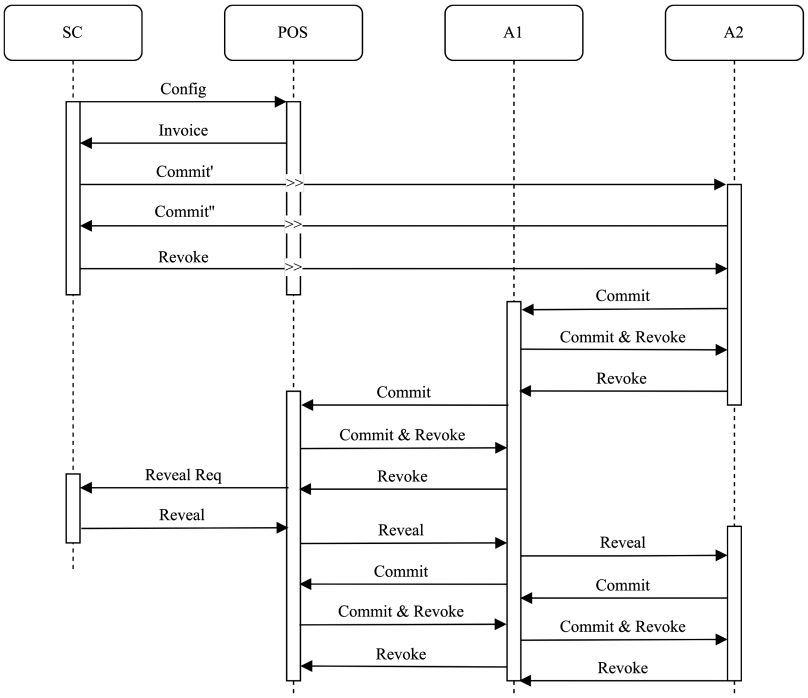


Fig. 3. Routing the payment from smart card to the POS.

the revocation key. We have used block letters to highlight the static data for the scripts (Listing 1–4) and the transaction segment (Listing 5).

3.5 Payment Routing

This section presents how to make a successful LN payment using a provisioned smart card. Similar to LN, payment execution has two phases: the locking and releasing phases. In the first phase, all parties make pairwise commitments, which are locked by a hashed secret. The locking phase ends when the last node in the path, POS, commits with its agent and receives the revocation of the last state. The release stage begins when the smart card releases the lock's key to the POS. Then, the Figure 3 illustrates the whole process rest of the nodes can pass the secret to release the HTLC and claim the funds, including the transaction fees.

3.5.1 Locking Phase. Here, we list the steps in the locking phase and discuss the events causing payment failure.

(1) When the cardholder presents the smart card to the terminal, POS and smart card negotiate the configuration details (Config), including agent information. POS determines the path toward the agent and the expected fees for the intermediaries (There are two intermediaries in our setting). POS throws an error if the connection cannot be established (i.e., no live connection).
(2) If peers are contactable, POS issues Invoice command to the smart card with the payment amount provided by the merchant and requests authorization to proceed with the transaction.
(3) After the cardholder authorizes, the smart card replies with Commit′, which consists of an identifier, encrypted payment hash, and a signature. We consider the last six bytes of the previous transaction's RevPoint as the identifier. For the payment hash, the smart card generates a secure random preimage,

from which the encrypted hash is created for the card agent, who can only decrypt the hash. Then, the smart card starts building the agent's version of the commitment transaction. Smart card derives the respective balances for the transaction from wallet balances and the invoice value. The rest is static data stored during the card provisioning, including the agent's RevPoint. Finally, the smart card generates the signature, one of the multi-signatures to spend the funding transaction. POS forwards the data to the card agent. Note that the payment hash's integrity is protected through the commitment signature.
In the meantime, POS issues a command to the smart card to build its own transaction version in which the smart card is required to generate a new secure random commitment point, followed by deriving the revocation point. The rest of the data has already been generated or stored as static values. After successfully building the transaction, the smart card can generate its signature, which will be stored securely.

(4) Upon receiving Commit′, the card agent identifies the related channel using the identifier given in the message. Then, he or she verifies the signature provided using the decrypted payment hash. If the verification and other typical wallet checks (i.e., sufficient balance; note that there might be an agreed minimum balance to be maintained.) are in order, the agent generates the signature for the smart card's transaction version. The signature is forwarded to the smart card with Commit′′. Note that the revocation secrets are not surrendered at this point. The cardholder may need to close the contract unilaterally in the future. Therefore, details of the latest transaction (at least) must be stored inside the card for future use.
(5) Smart card can verify the signature received in Commit′′ and reply with Revoke message. It includes the revocation secret for the previous transaction and a signed commitment point for the next commitment.
(6) Smart card's agent can then continue forwarding the HTLC until it finds the POS where the locking phase completes from the sender to the recipient. POS can also learn the payment hash at this point.

Failures during this section are not significant as the lock is still in the locked state. However, note that the smart card has already been revoked in the previous state. Therefore, it cannot simply neglect the newly negotiated state to restart the process but keep and resume it if the agent is willing to proceed. Nevertheless, all the payments sent are safe as the locks have yet to be released.

3.5.2 Release Phase. The next stage begins with the release of the lock by the smart card as follows:

(7) POS requests the preimage of the hash using the command RevealReq from the smart card. POS must also provide the correct payment hash.
(8) Upon the success of the previous step, the smart card replies with Reveal with the preimage. POS's reply with a success message concludes the payment for the smart card successfully.
(9) The rest of the peers can settle their balances using the preimage.

Failure setting in this phase is analogous to LN in this stage. All intermediaries are incentivized to hand over the secret to preceding payers. Failure to do so within a pre-specified time (CLTV time lock) will cause a loss of the intermediary fee. Note that POS may lose the payment amount if POS fails to pass the secret after the success message to the consumer. The lost amount goes to the card agent. Alternatively, the smart card can reclaim it if the cardholder publishes the latest transaction, as the agent did not receive the secret. Therefore, the POS is responsible for recovering from failures before the time lock expires.

3.6 Consecutive Payments

The engagement between the smart card and POS ends after successfully handing over the preimage, which marks the successful payment from the smart card to the POS and the agent. However, the release phase is yet to be finalized between the rest of the channels. Suppose the cardholder intends to make another payment. In that case, the cardholder must agree that the previous transaction was completed, and the balances must be adjusted accordingly, even though the agent has yet to receive the preimage. Therefore, it is safe for the agent to accept a subsequent payment from the cardholder. For the POS, making a subsequent transaction from another

card or the same card is appending another pending received HTLC. Therefore, making consecutive payments is possible even during the release phase of another transaction.

3.7 Channel Top-up and Protection

When the smart card balance deteriorates, the holder should be able to recharge it. He can always reach the blockchain to renew the multi-signature transaction with additional credit. However, that process comes with an on-chain fee which is expensive. Instead, the cardholder can utilize his agent's balance for the top-up. When the smart card balance drops, the agent's share goes up. Hence, if the cardholder initiates an off-chain transaction toward the smart card through the agent, he can perform a top-up with only off-chain fees. The cardholder must use a smart card reader to mediate the communication between the agent and the smart card. It is similar to the LN procedure to receive a payment. Note that the revocation key received at the end of the top-up is for all the prior commitments since the last top-up. The holder must preserve it until channel closure to prevent future illegal posting.

Publishing discarded commitments is possible in PCN. In general, smart card-like constrained nodes handle this situation by outsourcing it to a third-party watchtower. In this card payment scenario, such monitoring is pointless until a recharge where the smart card receives credit. Recharge makes all the prior states vulnerable. As a solution, the cardholder can outsource a watchtower but he must submit the required details (i.e., identifier and signature) through an untrusted entity, the POS. Instead, we propose a different approach using CSV delays assuming the cardholder periodically checks the channel's status. In the retail context, it is rational to presume such behavior as payment cards must be recharged recurrently. Accordingly, when the CSV delay exceeds the gap between two consecutive online presences, the cardholder is given space to detect if the opponent spent the channel's funding transaction. If it is an obsolete state, the cardholder can initiate the revoke process, creating the signed transaction using the relevant revoke key.

3.8 Channel Close

Channel close can be either unilateral or collaborative. Like LN, a collaborative close can remove the self-imposed balance restriction and redeem immediately. It also allows the resolution of unsettled HTLC portions.

4 GAME THEORETICAL ANALYSIS

This section theoretically investigates how well the cardholder and the agent adopt the protocol in a decentralized setting. We model the interaction between the smart card and its agent as an **Extensive Form Game** (**EFG**) with perfect information. The model allows examining participants' incentive to deviate from the protocol and any loss incurred to an honest party.

There are several attempts to game theoretically model PCN in different aspects, out of which we find Rain et al. [18] and Zappalà et al. [24] who model the peer interaction and payment routing in HTLC-based PCN are close to our work. Our approach extends their work by modeling the payment channel's lifetime with the temporal nature taken into consideration. The game demonstrates a modular long-run nature; hence, we employ EFG with stateful abstract subgame definitions.

4.1 Preliminaries

Initially, we define EFG, which consists of subgames, followed by how to define them with a state abstractly. Next, we define the equilibria used in the analysis, followed by the One-Shot Deviation theory to calculate the equilibrium. Finally, we define the weak immunity of a game.

Definition 4.1. (EFG) An Extensive Form Game is a tuple $\Gamma = \langle \mathcal{N}, \mathcal{H}, \mathcal{P}, v \rangle$, where $\mathcal{N}$ is a set of players, and $\mathcal{H}$ is a set of game histories which consist of a sequence of actions in the game, including an empty set (i.e., $\phi \in \mathcal{H}$). Players choose the actions from the action set $\mathcal{A}$, and they are sequenced as $(a_k)_1^{K'} \in \mathcal{H}$ when $K' < K$

and $(a_k)_1^K \in \mathcal{H}$. A history is terminal $(a_k)_1^K \in \mathcal{T}(\subset \mathcal{H})$ if $(a_k)_1^{K+1} \notin \mathcal{H}$. $\mathcal{P}$ is the next player function which assigns the next player $p \in N$ to every non-terminal history $h = (a_k)_1^K \in \mathcal{H}\backslash\mathcal{T}$, that is $P(h) = p$, and after h, the player $P(h)$ chooses an action from the action set $\mathcal{A}(h) = \{a : (h, a) \in \mathcal{H}\}$. v is the function which determines each player's utility, given the history, $h \in \mathcal{H}$.

A strategy profile of player p in EFG Γ is a function $\sigma_p : \mathcal{H}_p \rightarrow \mathcal{A}(.)$, which maps every $h \in \mathcal{H}_p$ with $\mathcal{P}(h) = p$ to an action from $\mathcal{A}(h)$. We denote strategy profile space by Σ. A tree data structure provides a clearer understanding of an EFG. To achieve this, we transform EFG features into a tree representation, where each internal vertex corresponds to the action player denoted by $\mathcal{P}(h)$ following a given action history h. The history h is represented by the path from the tree's root to the corresponding vertex, while $\mathcal{A}(h)$ is given by the edges originating from a node with the history h. The leaf vertices are labeled with the players' payoff resulting from the history.

Definition 4.2. (EFG Subgame) The subgame that follows history $h \in \mathcal{H}$ of an EFG $\Gamma = \langle \mathcal{N}, \mathcal{H}, \mathcal{P}, v \rangle$ is denoted by $\Gamma(h) = \left\langle \mathcal{N}, \mathcal{H}_{|h}, \mathcal{P}_{|h}, v_{|h} \right\rangle$ where for every history $h' \in \mathcal{H}_{|h}$ there exists $(h, h') \in \mathcal{H}, \mathcal{P}_{|h}(h') := P(h, h')$ and $v_{|h}(h') = v(h, h')$.

A subgame is a subtree under a vertex in the tree representation. Then, we define an abstract subgame in long-run EFG with a game state which can be used to calculate the utility instead of the history. It allows analysing an abstract subgame independently, given the initial state.

Definition 4.3. (Abstract Subgame) Abstract subgame $\mathcal{S}_i^p(q_h)$ is a subgame $\Gamma(h)$ of an EFG Γ where $\mathcal{P}(h) = p$ and q_h gives the state at history h on which $v_{\mathcal{S}}$ independently depends.

In the tree representation of the abstract subgame, the leaf vertices of the tree may contain either the utility value or a reference to another subtree. This modular and recursive nature simplifies the visualization of a long-run EFG.

Nash equilibrium (**NE**), **Subgame Perfect Nash Equilibrium** (**SPNE**), and Weak Immunity are key concepts in our analysis to assess the security properties of the protocol. The definitions are given in Appendix C along with the one-shot deviation theorem, which will be employed to determine the SPNE of a game.

4.2 Security Properties of Card Payments

4.2.1 (P1) No Honest Loss. The gain of an honest player must not gain negative value regardless of how other players behave in a byzantine setting. We express this feature using Weak Immunity defined in the Definition C.4.

4.2.2 (P2) No Deviation. By definition, Nash Equilibrium indicates that no player can obtain a better outcome by changing their own strategy. However, some NE strategies are not sensible in EFG as the strategy may not provide the best outcome a player could have gained. Therefore, we use the refined version of NE, SPNE.

4.3 Game Design

Here, we formally define the off-chain game (Γ_O) between the cardholder and the card agent in the four-corner model as a two-player EFG with perfect information. The game starts after opening the channel when the funding transaction receives the expected amount of confirmations. After that, the card and its agent can update their states (i.e., respective balances) off-chain until the channel is closed. The nature of the process is repetitive, although the states are changing. Therefore, we design the game modularly with a dynamic state of abstract subgames so that we can reuse them recursively.

Definition 4.4. (Channel State) The state of a channel between the cardholder and the agent is denoted by b_i where $i \in \mathbb{Z}^+$ and $b_i = (c_i, a_i)$ is a tuple of card and agent balances inclusive of a temporary amount.

It is essential to consider the effect of the HTLC section of the transaction in the channel state. According to the proposed protocol, the agent and cardholder do not settle the balances, so the signed transaction contains a separate and unsettled HTLC portion. In contrast, credit action negotiates the respective balances and settles the HTLC portion. In both cases, we denote the state as a whole, including the payment amount, denoted by x_i. The game stops when the channel is closed on the chain where each player can determine their gain. Both parties may honestly or dishonestly close the channel unilaterally or collaboratively with each other's consent.

4.4 Utility

Closing a channel allows us to determine the utilities gained by each party. We have the following assumptions. The waiting period to obtain funds costs both parties in proportion to the time and the value. On-chain and off-chain transaction fees are f_c and f_o, respectively and for simplicity, they are constant throughout the game. The merchant indirectly imposes the transaction fee on the cardholder even though the merchant bears the transaction cost.

Given the action history h, we define the generalized utility function $v(h)$ as in Equation (1).

$$v(h) = RA - DA - OC + \phi, \tag{1}$$

where RA denotes the received amount at the closure of the channel, and only the recipient must be able to claim it (because there can be cases both players can claim); DA denotes the amount the user deserved at the same point; OC denotes the opportunity cost due to t time delay; hence $OC = \epsilon \times t \times RA$ where ϵ is a player specific constant; ϕ denotes the gain collected through fees thus far. We use ϕ to denote the accumulated gain through the fees. Assume there had been n_i^- debits and n_i^+ top-ups at the ith state of the channel. Then, for the cardholder, the gain is the difference between the total cost of making off-chain transactions and the total cost had them been done on the chain. We also reduce on-chain fees for the funding and closing transactions and off-chain fees for the top-ups.

$$\phi_i^c = n_i^-(f_c - 2f_o) - n_i^+ f_o - 2f_c. \tag{2}$$

The gain of the card agent is the off-chain fees collected through facilitating debit and credit actions.

$$\phi_i^a = (n_i^- + n_i^+)f_o. \tag{3}$$

It is noteworthy whether these fee gains are inside their respective balances. For the smart card, fee gains are external to the current balance as its gain is a fringe benefit. On the contrary, the card agent collects the transaction fees to her balance, and the credit action extracts them from the channel. Therefore, the balance may contain the fee gain since the last top-up. Therefore, it is safe to interpret ϕ as the fee gain external to the balance.

4.5 Game Definitions

This section outlines the phases of stateful EFG between two players, smart card and card agent, $N = \{\text{SC}, \text{CA}\}$.

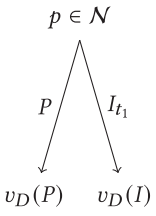
4.5.1 Abstract Dishonest Close Subgame. Abstract dishonest close subgame, $\mathcal{S}_D$ models a player's behavior when the opponent dishonestly closes the channel. During the given challenge window, the player can challenge and prove that the posted transaction is an old commitment. If it is established, the dishonest player's assets, except the external gain, go to the opponent. We have also included a variable opportunity cost incurred in the previous action. If it is not challenged, the dishonest player and the opponent receive the posted balance in the old state. The transactions between the smart card and the agent are mostly in an unsettled state (except immediately after a credit action). Therefore, the card agent who receives the payment has to redeem two outputs, self-balance and HTCL amount, obeying the delays imposed by the contract. Proving action is not delayed except the previous delay, whereas outputs from ignore action are subjected to a minimum of t_1 delay.

Definition 4.5. (Abstract Dishonest Close Subgame $\mathcal{S}_D$) Abstract Dishonest Close Subgame $\mathcal{S}_D = (\mathcal{N}, \mathcal{H}_D, \mathcal{P}_D, v_D)$ is a subgame of Γ_O, initiated by $p \in \mathcal{N}$ with the initial state $(b_i, b_{i'}, t)$. $b_i = (c_i, a_i)$ are current

Table 4. Player's Moves, Along with How the Move Affects the State of the Game

Action	Description
I_t	Ignore the previous action of the counterparty where the subscript indicates the minimum time of ignoring. The balance state remains the same. Any subsequent move may inherit the delay in the state.
H	Close the channel unilaterally and honestly while ignoring the previous requests. The current balance state is forwarded.
D	Close the channel unilaterally and dishonestly while ignoring the previous requests. Balance state changes from current state $b_i = (c_i, a_i)$ to previous state $b_{i'} = (c_{i'}, a_{i'})$ where $i' < i$.
C	Request to close the channel with the latest state. The balance state remains the same.
Dr	Request a debit of worth $x_i (\geq 0)$ which changes state $b_i = (c_i, a_i)$ to $b_j = (c_j, a_j)$ where $j > i$, $c_j = c_i - x_i$, $a_j > a_i + x_i$
Cr	Request a top-up which changes state $b_i = (c_i, a_i)$ to $b_j = (c_j, a_j)$ where $j > i$, $c_j = c_i + x_i$, $a_j > a_i - x_i$
A	Agree with the previous request to change of state and proceed with signing where necessary. The proposed state becomes the current state.
W	Unlock the HTLC by revealing the secret where possible.
P	Successfully prove that counterparty dishonestly closed the channel.

* Delay time in the state depends on preceding I action.



(a) Subgame tree for $\mathcal{S}_D^p$

	Action	Utility
$\mathcal{S}_D^{\text{SC}}$	P	$(a_i - \epsilon_c t c_i + \phi_i^c - f_c, -a_i + \phi_i^a)$
	I	$(c_{i'} - c_i - \epsilon_c t c_i + \phi_i^c, a_{i'} - a_i - \epsilon_a (t + t_1) a_{i'} + \phi_i^a - f_c)$
$\mathcal{S}_D^{\text{CA}}$	P	$(-c_i + \phi_i^c, c_i - \epsilon_a t a_i + \phi_i^a - f_c)$
	I	$(c_{i'} - c_i - \epsilon_c t_1 c_{i'} + \phi_i^c, a_{i'} - a_i - \epsilon_a t a_i + \phi_i^a)$

(b) The utilities of $\mathcal{S}_D$, given the initial state $(c_i, a_i, c_{i'}, a_{i'}, t)$.

Fig. 4. Abstract Dishonest Close Subgame $\mathcal{S}_D$.

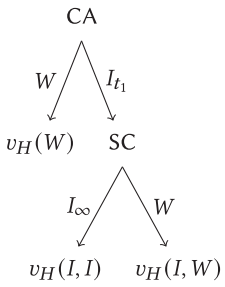
balances of SC and CA respectively, $b_{i'} = (c_{i'}, a_{i'})$ are past balances chosen by $p' \in \mathcal{N}$, and t is the inherited delay. $\mathcal{S}_{\mathcal{D}}$ is depicted by the game tree in Figure 4(a), which defines $\mathcal{H}_D$ and $\mathcal{P}_D$ with the actions being summarized in Table 4, and utilities are tabled in Figure 4(b) for both players.

The formal definition of $\mathcal{S}_D$ is provided in Definition 4.5, and we theorize the feasibility of dishonest close as follows. Their proofs are in the Appendix C.

Lemma 4.6. *The action P has a higher payoff than I in $\mathcal{S}_D$ for the player $p \in \mathcal{N}$ if the opponent $p' \in \mathcal{N}$ held a balance higher than f_c.*

Theorem 4.7. *(Dishonest Close Feasibility) Posting an old state is feasible iff the difference between posted and current balances is higher than the cost of releasing the past state.*

Corollary 4.8. *Dishonest closing is not feasible for the card agent in between two top-ups.*



(a) Game tree for $\mathcal{S}_H^{\mathrm{CA}}$

	Action	Utility
$\mathcal{S}_H^{\mathrm{CA}}$	W	$(-\epsilon_c(t+t_1)c_i + \phi_i^c, -\epsilon_a t a_i + \phi_i^a - f_c)$
	I, W	$(x_i - \epsilon_c(t+t_1)c_i + \phi_i^c - f_c, -x_i - \epsilon_a t a_i + \phi_i^a)$
	I, I	$(-\epsilon_c(t+t_1)c_i + \phi_i^c, -x_i - \epsilon_a t a_i + \phi_i^a)$
$\mathcal{S}_H^{\mathrm{SC}}$	I, W	$(-\epsilon_c t c_i + \phi_i^c, -\epsilon_a(t+t_1)a_i + \phi_i^a - f_c)$
	I, I, W	$(x_i - \epsilon_c t c_i + \phi_i^c - f_c, -x_i - \epsilon_a(t+t_1)a_i + \phi_i^a)$
	I, I, I	$(-\epsilon_c t c_i + \phi_i^c, -x_i - \epsilon_a(t+t_1)a_i + \phi_i^a)$

(b) The utilities of $\mathcal{S}_H$, given the initial state (c_i, a_i, t)

Fig. 5. Abstract Honest Unilateral Close Subgame $\mathcal{S}_H$.

4.6 Abstract Honest Unilateral Close Subgame

In this subgame, we examine a player's reaction when their opponent unilaterally closes the channel. In conventional LN protocol, a player does not need to do anything to own the respective balance from a unilateral close if it is the latest commitment. However, the proposed protocol does not impose to settle the HTLC payment, and most of the time, there will be an HTLC output in the signed transaction except just after a credit action. Notably, the HTLC portion is intended for the agent, although she might not know how to unlock it or skip withdrawing it. Similarly, the cardholder has the option to claim or skip. Claiming the payment amount is an undeserved benefit for the cardholder as the payment has already been finalized with the merchant. However, This can only occur when the irrational POS becomes unresponsive after the card releases the lock.

We primarily focus on the worst-case transaction with the unsettled state, which is the most common. When the cardholder closes the channel unilaterally with an unsettled state, the agent has the option to reveal the secret and claim the due HTLC amount or ignore the withdrawal (i.e., when the secret is unknown), which enables the cardholder to redeem the amount. Similarly, the agent can unilaterally close the channel and decide whether to withdraw the HTLC amount. Both scenarios cost an on-chain fee. On the other hand, if the agent closes the channel, the cardholder cannot do anything technically but wait for the agent's move.

Definition 4.9. (Abstract Honest Unilateral Close Subgame ($\mathcal{S}_H$)) Abstract Honest Unilateral Close subgame $\mathcal{S}_H^{\mathrm{CA}} = (\mathcal{N}, \mathcal{H}_H, \mathcal{P}_H, v_H)$ is a subgame of Γ_O, initiated by $\mathrm{CA} \in N$ with the initial state (b_i, t) where $b_i = (c_i, a_i)$ is the current balances of SC and CA respectively, and t is the inherited delay. $\mathcal{S}_H^{\mathrm{CA}}$ is depicted by the game tree in Figure 5(a), which defines $\mathcal{H}_H$ and $\mathcal{P}_H$ with the actions being summarized in Table 4. Similarly, $\mathcal{S}_H^{\mathrm{SC}}$ is depicted by $\mathrm{SC} \xrightarrow{I} \mathcal{S}_H^{\mathrm{CA}}$. The utility function v_H depends on the current player and the given initial state. Respective utilities are tabled in Figure 5(b).

Lemma 4.10. *If $x_i \geq f_c$, strategy to choose W is NE, if not I is NE in $\mathcal{S}_H$*

We defer the proofs to the Appendix C.

4.7 Abstract Ignore Subgame

We introduce a subgame to analyze how a player reacts to an ignore action taken by his or her opponent. It consists of previously defined subgames for honest and dishonest close. The opponent can also omit to react, which will cause both parties to lose their respective amounts. Inherently, ignoring action carries a delay, which will be passed to the subsequent subgames as the game's initial state.

Definition 4.11. (Abstract Ignore Subgame ($\mathcal{S}_I$)) Abstract Ignore subgame $\mathcal{S}_I = (\mathcal{N}, \mathcal{H}_I, \mathcal{P}_I, v_I)$ is a subgame of Γ_O, initiated by $p \in \mathcal{N}$ with the initial state (b_i, t) where $b_i = (c_i, a_i)$ is the current balances of SC and CA

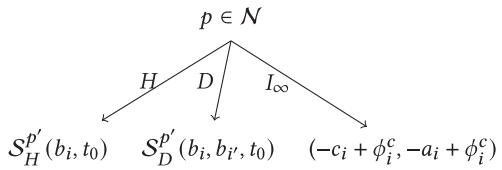


Fig. 6. Game tree for $\mathcal{S}_I^p$ with references to abstract subgames $\mathcal{S}_U$ and $\mathcal{S}_D$.

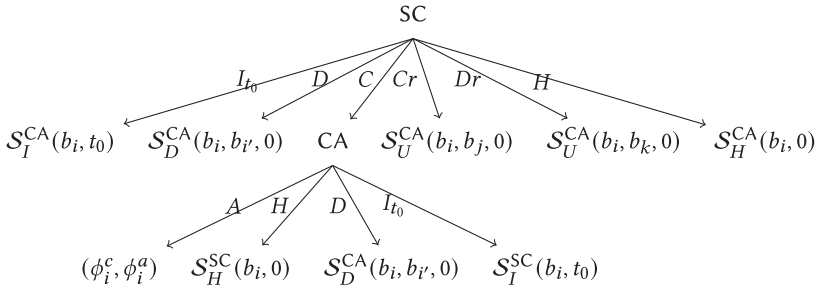


Fig. 7. Game tree for $\mathcal{S}_C^p$ with references to abstract subgames $\mathcal{S}_H$, $\mathcal{S}_D$, $\mathcal{S}_I$, and $\mathcal{S}_U$.

respectively, and t is the inherited delay. ($\mathcal{S}_I$) is depicted by the game tree in Figure 6, which defines $\mathcal{H}_I$ and $\mathcal{P}_I$ with the actions being summarized in Table 4, and subgames are as defined in definitions 4.5 and 4.9.

Lemma 4.12. *If the respective balances are worth releasing after $t_0 + t_1$ time and $a, c > f_c$, the strategy to choosing unilateral close (H) is SPNE in $\mathcal{S}_I$.*

We defer the proofs to the Appendix C.

4.8 Abstract Close and Update Subgames

We concurrently present the Close and Update subgames because of the mutual dependency. Note that the smart card is not a connected device, and it becomes online when the cardholder intends to make or receive a payment. Therefore, any collaborative action must be triggered by the cardholder. For example, the card agent cannot request a collaborative close unless as a response to smart card action.

Definition 4.13. (Abstract Closing Subgame - $\mathcal{S}_C$) The Abstract Closing Subgame $\mathcal{S}_C = (\mathcal{N}, \mathcal{H}_C, \mathcal{P}_C, v_C)$ is a subgame of Γ_O, initiated by SC $\in \mathcal{N}$ with a given initial state (b_i, t) where $b_i = (c_i, a_i)$ is the current balances of SC and CA respectively and t is the inherited delay. $\mathcal{S}_C$ is depicted by the game tree in Figure 7, which defines $\mathcal{H}_C$, $\mathcal{P}_C$, and the utility, with the actions being summarized in Table 4 and subgames are as defined in definitions 4.5, 4.9, 4.11, and 4.14.

Definition 4.14. (Abstract Update Subgame - $\mathcal{S}_U$) The Abstract Update Subgame $\mathcal{S} = (\mathcal{N}, \mathcal{H}_U, \mathcal{P}_U, v_U)$ is a subgame of Γ_O, initiated by CA $\in \mathcal{N}$ with a given initial state (b_i, b_j, t) where $b_i = (c_i, a_i)$ and $b_j = (c_j, a_j)$ are the current and proposed balances of SC and CA respectively and t is the inherited delay. $\mathcal{S}_U$ is depicted by the game tree in Figure 8, which defines $\mathcal{H}_U$, $\mathcal{P}_U$, and the utility, with the actions being summarized in Table 4 and subgames are as defined in definitions 4.5, 4.9, 4.11, and 4.13.

4.9 Off-chain Game

We define Off-Chain Game Γ_O between smart card and agent using subgames $\mathcal{S}_H$, $\mathcal{S}_I$,$\mathcal{S}_D$,$\mathcal{S}_U$,and $\mathcal{S}_C$. The game begins with the initial balances, and subsequent actions are embedded as individual actions or subgames with different states.

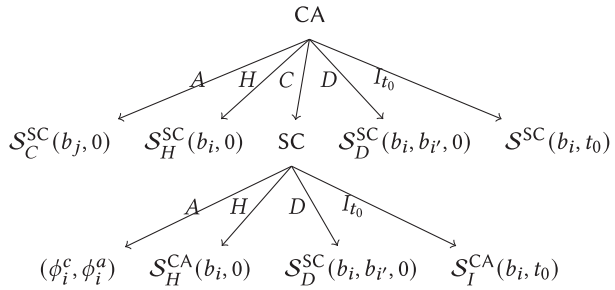


Fig. 8. Game tree for $\mathcal{S}_U^p$ with references to abstract subgames $\mathcal{S}_H$, $\mathcal{S}_D$, $\mathcal{S}_I$, and $\mathcal{S}_C$.

Definition 4.15. (Off-Chain Game - Γ_O) The Off-Chain Game $\Gamma_O = (\mathcal{N}_O, \mathcal{H}_O, \mathcal{P}_O, v_O)$ is an EFG with two players $\mathcal{N} = \{\text{SC}, \text{CA}\}$. Starting from the subgame, $\mathcal{S}_U(c_0, a_0, 0)$ where $c_0 > 0$ and $a_0 = 0$, the game trees of $\mathcal{S}_H$, $\mathcal{S}_I$,$\mathcal{S}_D$,$\mathcal{S}_U$,and $\mathcal{S}_C$ define $\mathcal{H}_O$ and $\mathcal{P}_O$ and v_O of Γ_O, with the actions being summarised in Table 4.

Further, we define two joint strategies σ_c and σ_u, which yield collaborative and unilateral actions, respectively, to analyse settings where players are cooperative and they are not.

Definition 4.16. Let σ_c be a strategy profile in the game Γ_O where the cardholder chooses Dr initially and for the rest of the stages, except the following occasions: Cr if $c_i - x_i <= c_{min}$ and cardholder intends to continue; C if cardholder intends to end the channel where c_i is the current balance, x_i is the anticipated debit, and $c_{min}(> 0)$ is the minimum balance. Also, both players accept any collaborative action $(\ldots, C/Cr/Dr)$, challenge (P) dishonest close (D), and close unilaterally (H) if ignored (I). If the payment amount is worth releasing, both players choose W; if not, I.

Definition 4.17. Let σ_u be a strategy of Γ_O where both players choose H wherever possible except choosing to ignore move (I) after any request for collaborative actions $(C/Cr/Dr)$, and proving action (P) against any dishonest close (D). Also, choose W if the payment amount is worth releasing; else I after H.

4.10 Honest Behavior Analysis

In this section, we analyse honest behaviors, which are two-fold; honest collaborative actions yielded by σ_c, and honest unilateral actions yielded by σ_u. We defer the proofs to the Appendix C.

Theorem 4.18. *(Weak Immunity) The honest behaviors are weak immune in Γ_O if the respective balances are worth releasing after $t_0 + t_1$, $a, c \geq f_c$, $\phi^c \geq 0$ and $\phi^a \geq min(f_c, x)$.*

Theorem 4.19. *(Incentive Compatability of Collaborative Actions) σ_c is SPNE in Γ_O if the top-up cost is less or equal to the fee gain, respective balances are worth releasing after $t_0 + t_1$ time, and $a, c > f_c$.*

Theorem 4.20. *(Incentive Compatability of Uniliteral Actions) σ_u is NE in Γ_O if $t_0 \geq t_1$, the respective balances are worth releasing after $t_0 + t_1$ time, and $a, c > f_c$, but not SPNE.*

4.11 Feasibility of Assumptions

The assumptions made during the proofs indicate the requisites that must be satisfied by the two players in order to carry out a mutually beneficial game. Here, we consolidate all the assumptions made in the theories and examine their feasibility during the game.

In general, we assume respective balances are worth releasing even after $t_0 + t_1$ time as there is an opportunity cost during the locked period. Also, the Lemma 4.6 dishonest subgame assumes that the old state posted on-chain by the opponent must be larger than the on-chain fee in order to make a credible threat. Together, in

other words, balances must maintain a minimum balance of worth on-chain fee and worth releasing even with an opportunity cost extended up to $t_0 + t_1$. Next, we have theory-specific assumptions. In addition to the above assumptions, Theory 4.18 expects that the external gains are non-negative and, more specifically, for the agent, $\phi^a \geq min(f_c, x)$. Further, Theory 4.19 introduces a new assumption regarding debit gain and top-up cost of the cardholder, which we can reduce it to $\phi^c + 2f_c \geq 0$ if we consider the game from the beginning. However, we have already assumed that $\phi^c \geq 0$ covers the above assumption in Theory, 4.18. Moreover, it is assumed that $t_0 \geq t_1$, in Theory, 4.20. It poses a credible threat to the opponent not to ignore and wait.

The following are the consolidated assumptions and how they can be practically implemented.

(1) $a_{min}, c_{min} = f_c$ *and both balances must worth to be redeemed after* $t_0 + t_1$ *time.*: This is directly related to dishonest close and symmetrical for both players, who must keep the balances above the minimum level. Note that the agent's balance is zero at the beginning. However, it is infeasible for the agent to close dishonestly before the first top-up (Corollary 4.8). Afterward, the cardholder can top-up his balance, leaving the minimum balance at the agent. Also, at the contract initiation, both players must determine whether the minimum level is worth redeeming after $t_0 + t_1$. Specifically, it affects the agent more as an agent could have used the collateral for other channels and earned.

(2) $\phi^a \geq min(f_c, x)$ *and* $\phi^c \geq 0$: This is specific to weak immunity, which assures no honest loss due to the opponent's action. Naturally, these conditions are not always true as the external gains grow along with the game, and it does not necessarily satisfy the conditions initially. For example, at the initiation of the game, the cardholder's external gain is $-2f_c$. Assuming the ratio between on-chain and off-chain fees is φ, the external gain remains negative until $\frac{2\varphi}{\varphi-2}$ number of debits happens before a top-up. Similarly, the agent has to wait for φ debit actions if we assume $x \geq f_c$. In other words, there is a possibility of loss during that period. If we make this period equivalent for both parties, no player will end the game abruptly. This happens when $\varphi = 4$, where four debits must occur before a top-up to pass the danger period. Although it satisfies $f_c > f_o$, $\varphi = 4$ is far from reality, and we cannot change the agent fee for a limited period as it creates complications in the design.

(3) $t_0 \geq t_1$: This prevents the opponent from waiting until the player closes the channel. On the other hand, it gives a sense of time to close the channel unilaterally.

The second assumption states that $\phi^a \geq min(f_c, x)$ and $\phi^c \geq 0$ must be true; if not, a loss may incur for an honest player. It is obvious that the requirement cannot be fulfilled at the beginning. As the threat is posed by each other, we can neutralize the situation by making the insecure period common for both players. For that, the current design expects $\varphi = 4$, which is unrealistic. Hence, we propose a one-time charge for the cardholder.

Theorem 4.21. *Γ_O is played for at least three debits if the agent charges a one-time fee, $\frac{\varphi-4}{\varphi-1} f_c$ from the cardholder.*

Proof. Let us adjust the external gains and the inequalities to reflect the assumption, with the agent charging a fee of $\frac{\varphi-4}{\varphi-1} f_c$ and assume $x \geq f_c$, for the simplicity of the proof.

$$\bar{\phi}^c = n^-(f_c - 2f_o) - n^+ f_o - \left(2 + \frac{\varphi - 4}{\varphi - 1}\right) f_c \geq 0, \tag{4}$$

$$\bar{\phi}^a = (n^- + n^+) f_o + \frac{\varphi - 4}{\varphi - 1} f_c \geq f_c. \tag{5}$$

Initially, there will be no top-up actions, hence $n^+ = 0$. If we determine the minimum debit actions required for no honest loss assumption to be true, we get $n^- \geq \frac{3\varphi}{\varphi-1}$ for both cardholder and agent, which means there should be a minimum of three debits. □

5 DISCUSSION

Section 2.1 highlights that card payments struggled to convince users for an extended period as a payment instrument during their initial stages. A similar phenomenon can be perceived in cryptocurrency card payments. To address this challenge, offering consumers a simple and familiar service while providing low transaction fees for retailers is important. In this work, we propose an affordable and high throughput card-based payment and recharging mechanism utilizing PCN. It aims at bridging the gap between the general public and decentralized currencies by enabling day-to-day cryptocurrency transactions and allowing existing cryptocurrency holders to spend their digital assets on micropayments. The following section will evaluate its capabilities and argue how it can potentially outperform traditional card payment systems.

5.1 Design Choices

We made three design choices in the proposed protocol: using HTLC-based LN as PCN, utilizing Bitcoin as the cryptocurrency, and incorporating smart cards as the payment instrument. We opted for HTLC due to its minimal computational impact on the PCN node; nevertheless, the drawbacks in HTLC-based PCN do not stem in our model, as discussed in Section 5.4.1. Further, Bitcoin scripting, which LN is based on, has limitations compared to other smart contract languages, such as solidity, and the smart card is one of the most constrained devices. The ability to implement a PCN protocol on such a device using such scripting language suggests that it can be effortlessly deployed for other currencies and devices with higher capabilities (i.e., IoT devices).

5.2 Proposed vs. Conventional Card Payments

The decentralized nature of cryptocurrencies is perfectly complemented by our proposed design, eliminating the need for third-party intermediaries like card schemes between agents. This results in minimal transaction fees, which sets us apart from conventional centralized card payment protocols. At the same time, it opens up the market to securely mediate in the transaction, even among untrusted entities, lowering the entry barrier and fostering healthy competition to reduce fees. Unlike traditional card payments, where merchants must wait for bulk settlement, the proposed protocol offers instant settlement. In the design phase, we curtail the PCN protocol to minimize the computational burden on the smart card. Still, the cryptographic operations executed by the smart card are considerably higher than what a conventional payment card performs during a transaction. The discrepancy in time consumption can be avoided by using specialized hardware with higher computational power. Nonetheless, the throughput of the overall payment process is expected to be higher than the conventional one, as a decentralized PCN network can scale indefinitely. These benefits, including significant fee reduction, increased speed, and potential for mass adoption, make our proposed protocol superior to conventional card payment systems.

5.3 Optimal Strategy

In our game theoretic analysis, we prove that honest behavior is optimal if channel participants satisfy the following;

(1) Cardholder pays an one-time fee of $\frac{\varphi-4}{\varphi-1} f_c$ to the card agent.
(2) Both keep balance higher than f_c, including subsequent top-ups and it must worth releasing after $t_0 + t_1$ time.
(3) Wait at least t_1 time for unilateral close.

We presented two strategies, σ_c and σ_u as the honest behavior. σ_c suggests always making moves collaboratively for updating and closing the channel, whereas σ_u suggests closing the channel honestly and unilaterally if the opponent is not responsive.

5.4 Security Features

5.4.1 Losts and Deviations. Theory 4.18 shows that the off-chain game gives no honest loss for the cardholder and the agent if they behave honestly, given the additional conditions mentioned in Section 5.3. However, LN protocol is inherently susceptible to wormhole attacks during payment routing [18], where a collation between two parties in the payment path can seize the intermediary fee of another. The same applies to the proposed protocol when POS collide with the card's agent, which is also possible in real life. However, the merchant bears the transaction fees, so that collation is irrational, as the POS could have used the three-party execution path in the first place.

According to the Theorem 4.19, both cardholder and agent have no incentive to deviate from the honest behavior, regardless of how the opponent behaves, given the additional conditions mentioned in 5.3.

5.4.2 Atomicity. Atomicity is a key feature for secure payment, especially when the payment is routed through third parties. In PCN, a single transaction from the sender to the recipient is made as multiple intermediary transactions. Hence, failures are natural; however, a failure at some point must fail all transactions, including the preceding ones. In other words, atomicity assures that either all parties update their respective balances according to the payment or no one does. By design, HTLCs in LN protect the atomicity of LN payments. The proposed protocol inherits that property during the routing. We also discussed the failures that might occur during the locking and releasing phases.

5.4.3 Privacy. We design the proposed protocol for the four-corner model of conventional card payments, but it can be extended at a cost. In the four-node setting, POS directly interacts with the card; however, the protocol does not expose enough information to profile the cardholder. The exposed identifier is dynamic (i.e., partial commitment point of the previous transaction) from transaction to transaction. The follow-up commitment point is encrypted and does not allow tracing between transactions. However, the card agent has the ability to find who paid whom, as the protocol does not implore POS to connect the card agent through the POS agent. This is similar to conventional card payments; however, there is room to mitigate this at a cost in our model, which we consider to be future work. For perfect privacy, the cardholder may occupy a predefined node between the POS and card agent and use encrypted communication to mask the POS and agent. It costs an additional intermediary fee and computational power. Nevertheless, the cardholder can utilize the pseudo-anonymity provided by the underlying blockchain.

5.5 Regulations

In the literature, we found that the involvement of regulatory bodies can reinforce the general public's confidence in cryptocurrencies. It is possible to regulate entities operating using cryptocurrencies if they are external to the network (e.g., exchanges, online wallets). However, single entities are not in a position to arbitrate within decentralized cryptocurrencies at the transaction level. On the contrary, the proposed protocol allows regulatory bodies to enact regulations at the transaction level through the agents. However, it is a voluntary compliance that can attract a certain group of people who seek government regulations. Otherwise, PCN alone can exist even in a hostile environment.

6 CONCLUSION

Cryptocurrency has a significant journey ahead before becoming widely recognized as a form of currency that the masses embrace. In the meantime, adopting payment instruments such as payment card technology is essential to enhance the convenience of conducting cryptocurrency payments, which can draw the general public. However, preserving the core concept of cryptocurrencies is critical for recognition. On the contrary, existing cryptocurrency-enabled payment cards are operated centrally, contradicting cryptocurrencies' fundamental con-

cept, decentralization. This article presents a card payment protocol that employs a payment channel network, Lightning Network and works alongside the traditional card payment framework while maintaining a decentralized nature. We utilize the low latency feature of PCN to make swift transfers and modify the protocol such that a computationally constrained device such as a smart card can abide. Notably, it eradicates the need for card schema between agents of consumer and producer, resulting in low transaction fees. Our game theoric analysis proves the protocol's robustness in a decentralized setting. We also discuss how it accomplishes the security goals as a payment protocol and the potential room for regulations should users require endorsement from governing bodies. In the endeavor of cryptocurrencies to achieve a state on par with traditional currencies, the proposed payment instrument is a step forward.

APPENDICES

A CARD PAYMENT OPERATION

The payment card industry has widely adopted the four-corner model as its framework. In this model, financial institutions (i.e., banks) act as intermediaries between the consumer and the merchant. The acquirer or the merchant's bank is the entity that issued the POS to the merchant, whereas the issuer is the bank of the payer who issued the payment card. This model is less restrictive than the three-party model, where the issuer and acquirer are indifferent. Once the cardholder presents the payment card to the merchant and authorizes the payment, the information related to the transaction is passed through the payment gateway to determine the best acquirer connected with the merchant. The payment processor of the chosen acquirer then routes the transaction information to an appropriate network, eventually reaching the issuer through an intermediary called card schema. The issuer and acquirer might be in different geographical locations or under different regulations; hence, an intermediary called card schema facilitates communication between them. As long as the issuer and acquirer support the schema, they can proceed with transaction processing, as do their clients. In most schemes, the issuer and acquirer must comply with the **Payment Card Industry Data Security Standard** (**PCI DSS**) to ensure that obtained personal data is secured across open and public networks. Upon receiving the payment request, the issuing bank checks the validity of the transaction information provided and the sufficiency of the bank balance. The issuing bank's approval completes the transaction from the consumer's perspective. However, the clearing and settlement of the funds are yet to be performed batch-wise, typically once per day. Then, the merchant receives the funds in their bank account after deducting the fees.

B PCN - STATE-OF-THE-ART

Poon et al. [17] were the first to devise PCN for Bitcoin in 2016, manoeuvering Bitcoin's ability to make programmable transactions. They used hash locks and timed transactions (i.e., HTLC) to steer the transaction to the recipient and a penalty mechanism to prevent double-spending. However, researchers have identified flaws and deficiencies in the proposed PCN. Poon's PCN is susceptible to the wormhole attack [12]. If two parties in the payment execution path collude, they can steal the intermediary fee of the node(s) in the middle. The main reason is using the same hash lock for every node involved in the transaction. It can also expose the transaction path or make transactions likeable, raising privacy issues in the network [11].

In 2017, LN was deployed commercially for Bitcoin, and the total collateral locked in the network has passed 5,000 bitcoins. Raiden was Ethereum's version of the PCN, which has its own token, RDN, for transactions. Moreover, the literature consists of a series of theoretical works that put forward enhanced variants of PCN. Eltoo [6] removes the requirement of having a penalty mechanism but expects the introduction of new features for Bitcoin. Sprites [14] and Spider [20] can be identified as attempts to gain higher throughput and low collateral on the chain. As per privacy preservation, Bolt [8] was designed for cryptocurrency with zero knowledge proof and offers anonymity for transactions where payments on a single channel are not linkable. Different signature

mechanisms were introduced to replace primitive HTLC. Malavolta et al. [13] proposed **anonymous multi-hop locks** (**AMHL**) using zero-knowledge protocols, whereas Mohanty et al. [15] introduced n-HTLC to enhance the privacy of PCN. Thyagarajan et al. [22] put forward a PCN that satisfies atomicity, un-linkability, and generality with the use of BLS signatures.

Notably, the improvements introduced for the privacy and efficiency of PCN demand additional computational power. For example, replacing hashing with signature inflate the computational cost of the PCN nodes, and they expect additional rounds of communication.

C GAME THEORETICAL DEFINITIONS AND PROOFS

C.1 Definitions

Definition C.1. (Nash Equilibrium - NE) A Nash Equilibrium of an EFG $\Gamma = \langle \mathcal{N}, \mathcal{H}, \mathcal{P}, v \rangle$ is a joint strategy $\sigma \in \Sigma$ s.t $\forall p \in \mathcal{N}, \forall \sigma'_p \in \Sigma_p : v_p(\sigma_p, \sigma_{-p}) \geq v_p(\sigma'_p, \sigma_{-p})$.

Definition C.2. (Subgame Perfect Nash Equilibrium - SPNE) A subgame perfect equilibrium is a joint strategy $\sigma \in \Sigma$, s.t. $\sigma_{|h}$ is a Nash Equilibrium of the subgame $\Gamma(|h)$, for every $|h \in \mathcal{H}$

Theorem C.3. *(One-Shot Deviation Principle) In an EFG, strategy profile σ is subgame perfect iff there is no player $p \in \mathcal{N}$ and a history $h \in \mathcal{H}$ such that p can gain higher payoff by deviating from σ any where except h.*

Definition C.4. (Weak Immunity) A joint strategy $\sigma \in \Sigma$ in a game Γ is called weak immune, if $\forall p \in \mathcal{N}, \forall \sigma' \in \Sigma : v_p(\sigma_p, \sigma'_{-p}) \geq 0$.

C.2 Proofs

Lemma C.5. *The action P has a higher payoff than I in $\mathcal{S}_D$ for the player $p \in \mathcal{N}$ if the opponent $p' \in \mathcal{N}$ held a balance higher than f_c.*

Proof. Let us assume the opponent holds a balance larger than the on-chain fee. Hence, for the cardholder $a_{i'} > f_c$ and for the agent $c_{i'} > f_c$. The difference between the utilities of two actions can be deduced to $v_D^{\mathsf{SC}}(P) - v_D^{\mathsf{SC}}(I) = a_{i'} - f_c$ and $v_D^{\mathsf{CA}}(P) - v_D^{ca}(I) = c_{i'} - f_c$ (from total capacity being a constant). Both differences are positive according to our assumption. Hence, challenging the old transaction gains more utility for each player if the opponent's balance is worth more than the on-chain fee. □

Theorem C.6. *(Dishonest Close Feasibility) Posting an old state is feasible iff the difference between posted and current balances is higher than the cost of releasing the past state.*

Proof. Consider the player's utility if the opponent ignored the move, meaning that illegal withdrawal was a success. For both players, the payoff generally contains the difference between the old and current balance, external gain, and redemption costs, including opportunity and on-chain fees (agent only). Both are covered by the condition in the theory when the amount is worth releasing. Therefore, the player may close the channel dishonestly when the given condition is satisfied. The gain from off-chain transaction fees might be higher than all other negatives. However, by definition, the ϕ is outside the balance, and the agent already owns that. Therefore, it has no positive effect, making vice versa also true. □

Corollary C.7. *Dishonest closing is not feasible for the card agent in between two top-ups.*

Proof. Let us assume the card agent posts an old channel state $b_{i'}$ when the current state is b_i. By definition, $a_{i'} - a_i$ can also be interpreted as total credit minus total debit during $b_{i'}$ to b_i states. If there are no top-ups in between, the difference, $a_{i'} - a_i$, becomes negative, making Theorem 4.7's condition false. □

Lemma C.8. *If $x_i \geq f_c$, strategy to choose W is NE, if not I is NE in $\mathcal{S}_H$*

Proof. We use backward induction to prove NE of $\mathcal{S}_H$. If we start from the bottom, comparing the cardholder's actions to ignore and withdraw, we find the difference to be $x_i - f_c$ in both $\mathcal{S}_H^{\text{SC}}$ and $\mathcal{S}_H^{\text{CA}}$. If we assume x_i is larger than the redeem cost (i.e., f_c), the withdraw action yields the best payoff. With that, we can compare the agent's withdrawal and ignore actions where we find the difference to be $x_i - f_c$ in both games. With the same assumption, the agent's withdrawal action yields better. Similarly, if we assume otherwise, we can prove that ignoring action yields the best. Hence, NE in $\mathcal{S}_H$ depends on the payment amount. □

Lemma C.9. *If the respective balances are worth releasing after $t_0 + t_1$ time and $a, c > f_c$, the strategy to choosing unilateral close (H) is SPNE in $\mathcal{S}_I$.*

Proof. First, assume that (1) respective balances are worth releasing after t_0+t_1 time and (2) $a, c > f_c$. Suppose the player chooses to close the channel dishonestly (D). In that case, a rational counterparty will choose to challenge the old commitment (P) to maximize its gain (Lemma 4.6 assumption is also satisfied here). It causes the first player to lose the total possession. The player can also do nothing (I), causing him or her to lose the current balance, similar to the dishonest close loss. Quantitatively, they are $-c+\phi^c$ and $-a+\phi^a$ for the cardholder and the agent, respectively. According to Lemma 4.10, NE of the game $\mathcal{S}_H$ grants $-\epsilon_c(t_0 + t_1)c_i + \phi_i^c$ in $\mathcal{S}_I^{\text{SC}}$ and $-\epsilon_c t_0 c_i + \phi_i^c$ in $\mathcal{S}_I^{\text{CA}}$ to the cardholder without depending on the payment amount. The difference between this and other actions is similar to the difference between the self-balance and redeem delay cost (delay up to $t_0 + t_1$ time). We have assumed c_i is worth redeeming after $t_0 + t_1$; hence, the unilateral close gives the highest payoff for the cardholder. On the contrary, the agent's payoff depends on the payment amount; however, the first assumption covers opportunity cost up to $t_0 + t_1$ of delay. If the $x > f_c$, the agent withdraws and bears an on-chain cost; otherwise, she ignores and bears a cost of x_i. If we compare the payoff from other actions, unilateral close gains are higher because the agent balance covers both costs (from the second assumption and $a \geq x$). Therefore, if the total cost is less than the withdrawing amount, the honest unilateral closing yields better than other actions in $\mathcal{S}_I$. □

Theorem C.10. *(Weak Immunity) The honest behaviors are weak immune in Γ_O if the respective balances are worth releasing after $t_0 + t_1$, $a, c \geq f_c$, $\phi^c \geq 0$ and $\phi^a \geq min(f_c, x)$.*

Proof. Let us assume (1) respective balances are worth releasing after $t_0 + t_1$, (2) $a, c \geq f_c$ (3) $\phi^c \geq 0$ and $\phi^a \geq min(f_c, x)$. Considering the update actions in general, we assume credit and debit actions have no negative effect of incurring loss for both players (assumption 3). Then, we examine the honest closing actions which end the game. According to the definitions 4.16 and 4.17, the two strategy profiles yield honest collaborative and unilateral closing, respectively. We prove both strategies have weak immunity properties, which makes respective actions have weak immunity.

According to the strategy σ_c, the game concludes with the collaborative close initiated by the cardholder regardless of previous events. Let us assume that the game is at state $(b_i, 0)$ and above assumptions are true. Suppose the agent deviates from accepting the collaborative close action to close the channel dishonestly. In that case, the smart card will get a non-negative gain $a_i + \phi_i^c - f_c$ because the on-chain fee is assumed to cover the agent's balance (Lemma 4.6 with assumption 2). According to σ_c, if the agent deviates to unilateral close action or ignores responding (Lemma 4.12 with assumption 1 and 2), the game ends in the unilateral close subgames, $\mathcal{S}_H^{\text{SC}}$ and $\mathcal{S}_H^{\text{CA}}$ respectively. In $\mathcal{S}_H^{\text{SC}}$, the cardholder's payoff is non-negative with no assumptions because of instant redemption. $\mathcal{S}_H^{\text{CA}}$ mandates t_0+t_1 delay, which is covered by the first assumption. Hence, deviating from accepting cannot create a loss for the cardholder if the given assumptions are true.

Next, we examine the agent's payoffs if the cardholder deviated from the collaborative close to any other closing action or ignore action. According to σ_c, if the cardholder deviated to honest unilateral close or ignore

responding, the game ends in the unilateral close subgames, $\mathcal{S}_H^{\text{CA}}$ and $\mathcal{S}_H^{\text{SC}}$ respectively. According to the table in Figure 5(b), there is a $t_0 + t_1$ delay for the agent in $\mathcal{S}_H^{\text{SC}}$ which the first assumption will cover. Additionally, if $f_c \geq x_i$, the agent's action cause a cost worth the on-chain fee or else a cost worth the last payment value. As the third assumption assumes, if $\phi^a \geq min(f_c, x)$ is true, $\phi^a \geq f_c$ when $x \geq f_c$ else $\phi_i^a \geq x$ where the external gain covers these costs. Hence, the agent gains a non-negative payoff if the cardholder deviates to either I or C. Finally, a dishonest close also yields a positive outcome as we assume the opponent's balance exceeds f_c. This concludes that σ_c has weak immunity so does the collaborative close.

Consider strategy σ_u, which yields honest unilateral closing with an unresponsive opponent. Let the game again be at $(b_i, 0)$ state. The cardholder moves with H following the σ_u; however, suppose the cardholder deviated to ignore or to any collaborative action. In that case, the game ends in $\mathcal{S}_H^{\text{SC}}$ as the agent chooses to close the channel unilaterally according to the strategy. Similar to the first section, payoffs are non-negative. Instead, if the cardholder deviated to close the channel collaboratively, he yields non-negative ϕ^a. The cardholder can also deviate to dishonest close, which will be penalized and grant non-negative gain, $c_i + \phi_i^c - f_c$ to the agent. This concludes that σ_u has weak immunity so does the honest unilateral close. □

Theorem C.11. *(Incentive Compatability of Collabarative Actions) σ_c is SPNE in Γ_O if the top-up cost is less or equal to the fee gain, respective balances are worth releasing after $t_0 + t_1$ time, and $a, c > f_c$.*

Proof. σ_c is a conditional strategy with three main actions, debit, credit and collaborative close. Assume the game is at the state $(b_i, 0)$ where the cardholder chooses Dr, subsequently picks Cr where necessary and C at state $j(> i)$. We start applying the one-deviation rule at state i while the rest of the moves accord with σ_c. Accordingly, the cardholder yeilds ϕ_j^c adhering to σ_c. We begin the analysis, assuming (1) the top-up cost is less or equal to the fee gain, (2) respective balances are worth releasing after $t_0 + t_1$ time, and (3) $a, c > f_c$.

According to the strategy at the state $(b_i, 0)$, the anticipated gain by Dr action of the cardholder is ϕ_j^c. Consider when he deviated from Dr and $c_i - x_i > c_{min}$. Deviating to collaborative close yields a utility, ϕ_i^c, better than the gain through ignoring the action, which contains an additional opportunity cost (Lemma 4.12 with assumptions 2 and 3). Deviation to the unilateral close action also incurs an extra opportunity cost regardless of the payment amount (Lemma 4.10), whereas the gain from a dishonest close is losing all assets (Lemma 4.6 with assumption 3). The credit action is subjected to an external cost of an off-chain fee. Therefore, the best paying-off action can be determined by comparing ϕ_i^c and ϕ_j^c. The difference is $(n_j^- - n_i^-)(f_c - 2f_o) - (n_j^+ - n_i^+)f_o$, which is equal to the difference between top-up cost and the fee gain within two states. The difference is non-negative according to the first assumption. Therefore, the cardholder cannot yield a higher payoff by deviating from debit action at this state where $c_i - x_i > c_{min}$. Next, we examine an intermediate state, $k(i < k < j)$ where $c_k - x_k \leq c_{min}$. Cardholder picks credit action according to σ_c. Here, the Dr action will get rejected by the agent in order to keep the minimum balance. Next, a possible option is to close collaboratively, resulting in a better performance than Cr (assuming the worst case, closing collaboratively after accepting the Cr). However, an extra cost may incur to reopen the channel on a chain larger than the top-up cost. Hence, if the cardholder continues, he can deviate to none other than Cr even though it is an additional cost of an off-chain fee to the gain. In the last state j, when the cardholder intends to close the channel, σ_c instructs to choose the collaborative close (C). Similar to the above, other methods of closing yield less than the collaborative close when the external gain is positive.

Then, we apply the one-shot deviation rule to the agent's response to the cardholder's request to close the channel collaboratively. According to σ_c, the agent gains ϕ_j^a by choosing to accept the proposal. The game reaches unilateral close subgames if she deviates to unilateral close action or to ignoring action. Regardless of the opponent's behavior, the agent's actions are subjected to one of the costs, on-chain fee, opportunity delay or payment amount (table in Figure 5(b)), which is in addition to ϕ_j^a. Similarly, dishonest close causes a loss of total assets. Therefore, acceptance can provide a better payoff than the rest.

Next, we consider the level starts from the subgame $\mathcal{S}_U$, with the given state $(b_i, b_j, 0)$. The subgame receives the proposal of changing the current state (c_i, a_i) to $(c_j, a_j)(j >= i)$ which is related to two scenarios, debit $(c_i < c_j, a_i > a_j)$ or credit $(c_i > c_j, a_i < a_j)$. To prove the SPNE, we apply the one-shot deviation principle for the given state, fixing the rest of the moves according to σ_c. According to the strategy profile σ_c, the agent chooses to accept the update A and gain ϕ_j^a. The agent can derive delayed payments from $\mathcal{S}_H$ (Lemma 4.10) and $\mathcal{S}_I$ (Lemma 4.12 with assumptions 2 and 3) and lose all assets from $\mathcal{S}_D$ (Lemma 4.6 with assumption 3). On the other hand, collaborative closing C, which is getting accepted by the cardholder according to σ_c, gives ϕ_i^a to the agent. Therefore, the agent can yield a maximum of ϕ_i^a if she deviates from A. Now the question is which action yields the highest; A, yielding ϕ_j^a or C, yielding ϕ_i^a. The difference can be deduced to $((n_j^+ - n_i^+) + (n_j^- - n_i^-))f_o$, which is greater than or equal to zero regardless of the scenario because $n_j^+ \geq n_i^+$ and $n_j^- \geq n_i^-$. Hence, choosing to accept has the highest payoff. Then, we apply the one-shot deviation rule to the smart card's response to the agent's request to close the channel collaboratively. According to σ_c, the cardholder gains ϕ_i^c by choosing to accept the proposal. If he deviates to any form of close or ignoring actions, the maximum he can secure is a delayed redemption. Both dishonest close and ignore to respond again yield a loss of the total balance. Therefore, the cardholder always gains less payoff if he deviates σ_c.

So far, we have proved that players have no incentive to deviate from the σ_c strategy given any state, including the subgames. Therefore, we conclude that the strategy profile σ_c is SPNE according to the one-deviation principle. □

Theorem C.12. *(Incentive Compatability of Uniliteral Actions) σ_u is NE in Γ_O if $t_0 \geq t_1$, the respective balances are worth releasing after $t_0 + t_1$ time, and $a, c > f_c$, but not SPNE.*

Proof. Let us assume (1) $t_0 \geq t_1$, (2) the respective balances are worth releasing after $t_0 + t_1$ time, and (3)$a, c > f_c$.

First, we prove that σ_u is NE in Γ_O, considering the overall strategy. The strategy is to close the channel unilaterally and ignore collaborative actions. If a player chooses to deviate to any collaborative action, the game is led to Ignore subgame as the opponent's response is to ignore such requests. If so, the player lands up choosing unilateral close (Lemma 4.12 with assumptions 2 and 3), which is the best choice but with a further delay to release the asset. Hence, the player could have chosen the unilateral close in the first place. Similarly, he cannot also gain a higher payoff by entering the dishonest close subgame (Lemma 4.6 with assumption 3). On the other hand, choosing to ignore does not yield a better gain for the player if $t_0 \geq t_1$. Both actions end up in a unilateral close subgame (Lemma 4.12 with assumptions 2 and 3), yielding the same utility with a different opportunity cost. Letting another player close the channel mandate no redeem delay on-chain. However, we assumed that the opponent waits for $t_0 \geq t_1$ to close unilaterally, which is a credible threat. Therefore, no player can increase utility by deviating σ_u.

Now, we apply the one-shot deviation rule to check the subgame perfectness of the strategy. If it is SPNE, no player gains a higher payoff by deviating. Therefore, let us consider an instance where the cardholder requests a debit action, initiating $\mathcal{S}_U^{\text{CA}}(b_i, b_j, 0)$ where the agent's move is to ignore the request (i.e., ..., Dr, I, H) according to σ_u. Instead, if the agent deviates to accepting the requests and, according to the one-shot deviation rule, we fix the rest of the actions as prescribed, similar to above, the game is led to the unilateral close game (i.e., ..., Dr, A, H). However, the outcome has different external gains, such that the agent is compensated with an extra fee when she accepts the debit request. Additionally, ignoring action causes an extra opportunity cost for the agent. Therefore, the agent deviating to accept action gains a higher payoff than ignoring. Hence, it does not satisfy the one-deviation rule, and it is sufficient to conclude that σ_u is not SPNE. □

D TRANSACTION SCRIPTS

```
P2WKH(<REMOTE_PAY_POINT>)
```

Listing (1) To Remote Script

```
OP_IF
  <rev_point>
OP_ELSE
  <CSV_DELAY_T1> OP_CHECKSEQUENCEVERIFY OP_DROP
  <LOCAL_PAY_POINT>
OP_ENDIF
OP_CHECKSIG
```

Listing (2) To Local Script

```
OP_DUP OP_HASH160
<RIPEMD160(SHA256(rev_point))> OP_EQUAL
OP_NOTIF
  OP_SIZE 32 OP_EQUAL
  OP_NOTIF
    <CSV_DELAY_1>
    OP_CHECKSEQUENCEVERIFY OP_DROP
    <LOCAL_PAY_POINT>
  OP_ELSE
    OP_HASH160
    <RIPEMD160(payment_hash)> OP_EQUALVERIFY
    <REMOTE_PAY_POINT>
  OP_ENDIF
OP_ENDIF
OP_CHECKSIG
```

Listing (3) SC HTLC Script

```
OP_DUP OP_HASH160
<RIPEMD160(SHA256(REV_POINT))> OP_EQUAL
OP_NOTIF
  OP_SIZE 32 OP_EQUAL
    <CSV_DELAY_1>
    OP_CHECKSEQUENCEVERIFY OP_DROP
    OP_HASH160 <RIPEMD160(payment_hash)>
    OP_EQUALVERIFY
    <LOCAL_PAY_POINT>
  OP_ELSE
    <CSV_DELAY_2> OP_CHECKSEQUENCEVERIFY
    OP_DROP
    <REMOTE_PAY_POINT>
  OP_ENDIF
OP_ENDIF
OP_CHECKSIG
```

Listing (4) CA HTLC Script

```
...
"vin": [
  {
  "txid": <FUNDING_TX_ID>,
  "vout": <FUNDING_TX_VOUT>,
  "scriptSig":
    0 <local_sig> <remote_sig>,
  }
],
"vout": [
  {
  "value": <local_amount>,
  "scriptPubKey":<local_script>
  },
  {
  "value": <remote_amount>,
  "scriptPubKey":<remote_script>
  },
  {
  "value": <htlc_amount>,
  "scriptPubKey":<htlc_script>
  }
]...
```

Listing (5) Transaction template with input and output segments. The hashes of relavant scripts and respective amounts will be placed in the corresponding places to build the transaction.

REFERENCES

[1] Hayder Albayati, Suk Kyoung Kim, and Jae Jeung Rho. 2020. Accepting financial transactions using blockchain technology and cryptocurrency: A customer perspective approach. *Technology in Society* 62, 101320 (2020), 101320. DOI: https://doi.org/10.1016/j.techsoc.2020.101320

[2] Mikael Asplund, Jakob Lovhall, and Simin Nadjm-Tehrani. 2018. In-store payments using bitcoin. In *Proceedings of the 2018 9th IFIP International Conference on New Technologies, Mobility and Security*. 1–6. DOI: https://doi.org/10.1109/NTMS.2018.8328738

[3] Aaron Baur, Julian Bühler, Markus Bick, and Charlotte Bonorden. 2015. Cryptocurrencies as a disruption? Empirical findings on user adoption and future potential of bitcoin and co. *Open and Big Data Management and Innovation: 14th IFIP WG 6.11 Conference on e-Business, e-Services, and e-Society, I3E 2015, Delft, The Netherlands, October 13-15, 2015, Proceedings 14*. 63–80. DOI: https://doi.org/10.1007/978-3-319-25013-7_6

[4] Dirk G. Baur, Kihoon Hong, and Adrian D. Lee. 2018. Bitcoin: Medium of exchange or speculative assets? *Journal of International Financial Markets, Institutions and Money* 54 (2018), 177–189.

[5] Sujit Chakravorti. 2010. Externalities in payment card networks: Theory and evidence. *Review of Network Economics* 9, 2, Article 3 (2010).

[6] Joseph Poon and Thaddeus Dryja. 2018. *eltoo: A Simple Layer2 Protocol for Bitcoin*. Lightning Network. Retrieved from https://blockstream.com/eltoo.pdf

[7] Joseph Poon and Thaddeus Dryja. 2016. *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*. Lightning Network. Retrieved from https://lightning.network/lightning-network-paper.pdf

[8] Matthew Green and Ian Miers. 2017. Bolt: Anonymous payment channels for decentralized currencies. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 473–489.

[9] Fumiko Hayashi and Ying Lei Toh. 2022. Assessing the case for retail CBDCs: Central banks' considerations. *Payments System Research Briefing* May 26, 2 (2022), 1–7.

[10] Nicole Jonker. 2019. What drives the adoption of crypto-payments by online retailers? *Electronic Commerce Research and Applications* 35, 100848 (2019), 100848. DOI: https://doi.org/10.1016/j.elerap.2019.100848

[11] George Kappos, Haaroon Yousaf, Ania Piotrowska, Sanket Kanjalkar, Sergi Delgado-Segura, Andrew Miller, and Sarah Meiklejohn. 2021. 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 25*. Springer, 167–186.

[12] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. 2019. Anonymous multi-hop locks for blockchain scalability and interoperability. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*, The Internet Society. Retrieved from https://www.ndss-symposium.org/ndss-paper/anonymous-multi-hop-locks-for-blockchain-scalability-and-interoperability/

[13] Subhra Mazumdar, Sushmita Ruj, Ram Govind Singh, and Arindam Pal. 2020. HushRelay: A privacy-preserving, efficient, and scalable routing algorithm for off-chain payments. In *Proceedings of the 2020 IEEE International Conference on Blockchain and Cryptocurrency*. IEEE, 1–5.

[14] Andrew Miller, Iddo Bentov, Surya Bakshi, Ranjit Kumaresan, and Patrick McCorry. 2019. Sprites and state channels: Payment networks that go faster than lightning. In *Financial Cryptography and Data Security: 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18–22, 2019, Revised Selected Papers*. Springer, 508–526.

[15] Susil Kumar Mohanty and Somanath Tripathy. 2021. n-htlc: Neo hashed time-lock commitment to defend against wormhole attack in payment channel networks. *Computers and Security* 106, 102291 (2021), 102291.

[16] Christopher R. Plouffe, Mark Vandenbosch, and John Hulland. 2000. Why smart cards have failed: Looking to consumer and merchant reactions to a new payment technology. *International Journal of Bank Marketing* 18, 3 (2000), 112–123.

[17] Slava Gomzin and Dan Itkis. 2018. *GRAFT: Decentralized, Real-Time Credit, Debit, and Crypto Payment Processing Blockchain*. Graft Network. Retrieved from https://www.graft.network/wp-content/uploads/2018/10/Graft_White_Paper_3.0_October_2018.pdf

[18] Sophie Rain, Georgia Avarikioti, Laura Kovács, and Matteo Maffei. 2023. Towards a game-theoretic security analysis of off-chain protocols. In *IEEE 36th Computer Security Foundations Symposium (CSF'23)*, 107–122. DOI: https://doi.org/10.1109/CSF57540.2023.00003

[19] Daniele Scafile and Denis Knaepen. An exploratory study investigating cryptocurrencies acceptance as a form of payment: The case of SMEs in italy and switzerland. (n.d.).

[20] Vibhaalakshmi Sivaraman, Shaileshh Bojja Venkatakrishnan, Kathleen Ruan, Parimarjan Negi, Lei Yang, Radhika Mittal, Giulia Fanti, and Mohammad Alizadeh. 2020. High throughput cryptocurrency routing in payment channel networks. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*.

[21] Fred Steinmetz, Marc Von Meduna, Lennart Ante, and Ingo Fiedler. 2021. Ownership, uses and perceptions of cryptocurrency: Results from a population survey. *Technological Forecasting and Social Change* 173, 121073 (2021), 121073.

[22] Sri Aravinda Krishnan Thyagarajan and Giulio Malavolta. 2021. Lockable signatures for blockchains: Scriptless scripts for all signatures. In *Proceedings of the 2021 IEEE Symposium on Security and Privacy*. IEEE, 937–954.

[23] Gregory E. Truman, Kent Sandoe, and Tasha Rifkin. 2003. An empirical study of smart card technology. *Information & Management* 40, 6 (2003), 591–606.
[24] Paolo Zappalà, Marianna Belotti, Maria Potop-Butucaru, and Stefano Secci. 2020. Game theoretical framework for analyzing blockchains robustness. Technical Report. https://eprint.iacr.org/2020/626.pdf. Accessed April 2024.

Received 29 April 2023; revised 1 February 2024; accepted 12 March 2024