Decentralizing Watchtowers for Payment Channels using IPFS

Hannes Bönisch, Matthias Grundmann
KASTEL Security Research Labs
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

***Abstract*—Payment channels have been proposed as a way to improve the scalability of blockchains such as Bitcoin. However, payment channel protocols require that participating parties watch the blockchain regularly for new transactions. If a party observes, in a given period of time, a fraudulent transaction that closes the payment channel in an outdated state, the fraudulent transaction can be revoked. Previous work has proposed to outsource this task to a third party, a so called watchtower. A user of a payment channel employs a dedicated watchtower and sends the data to the watchtower that the watchtower requires to revoke fraudulent transactions. In this paper, we replace the strict binding of a user to a watchtower by a decentralized approach for watchtowers that requires no direct interaction between a party of a payment channel and the watchtower. This decentralized approach uses IPFS to publicly store the information required by a watchtower. With this approach, anyone can detect and revoke a fraud by watching the blockchain and reading a file from IPFS that contains information for each outdated commitment transaction. A reward for successful revocations can be used as incentive.**

I. Introduction

Payment channels represent an approach to improve the limited transaction throughput of blockchains such as Bitcoin [1] and allow the sending of transactions between the parties of a channel without the need to publish each transaction on the blockchain. Because the distribution of the funds inside a channel is not stored on-chain, the funds are exposed to fraud: A dishonest party can publish an outdated channel state, which has the channel funds distributed in their favor. This behavior can be punished by the defrauded party by publishing a revocation transaction that punishes the dishonest party by sending all funds of the channel to the defrauded party. However, the defrauded party has only a limited period of time before the dishonest party can prevent the revocation, and, thus, the defrauded party needs to respond quickly.

To secure their funds while being offline for longer periods of time, parties of a payment channel can hire third party watchtowers to check that the channel is not being closed by the other party in an outdated state. In previous approaches for watchtowers (see Section II-B), a payment channel user makes an agreement with a specific watchtower and sends a revocation transaction to the watchtower for every update of the payment channel. The payment channel user needs to trust that the watchtower, who was employed, continuously watches the blockchain and reacts to frauds.

In this paper, we lift the strong binding between payment channel user and watchtower and introduce a novel approach for watchtowers without the requirement for an explicitly employed watchtower. The key idea is to make revocation transactions accessible to the public via IPFS [2]. Because the revocation transactions are publicly available, everybody can now run a watchtower to check the blockchain for outdated commitment transactions and secure the funds *of* the party being defrauded *for* the party being defrauded. The revocation transactions can include a reward for a successful revocation, thus, incentivizing participation in this 'blockchain neighborhood watch'.

This watchtower approach has two main distinctive features compared to previous approaches: (1) Separation of storage and watching: The main task of a watchtower is to watch the blockchain for commitment transactions while the storage of revocation transactions is performed by a storage host using IPFS. Therefore, the requirements and cost for running a watchtower are low and the number of watchtowers and storage hosts can be scaled independently. (2) No arrangement is required between watchtower and payment channel user: A payment channel user does not need to trust that a specific watchtower, with whom an agreement was made during channel establishment or during a state update, continuously runs and honors the agreement. Instead, watchtowers are incentivized to watch the blockchain because there is a reward for the watchtower included in each revocation transaction. No direct communication is required between the payment channel user and the watchtower. Therefore, a watchtower can revoke a fraudulent transaction even if the watchtower is not known to the payment channel user or if the watchtower did not yet exist when the channel was updated last.

In the following section, we explain in more detail how payment channels work and why watchtowers are required. While the decentralized approach for watchtowers can be used for different payment channel protocols, we put our focus on the Lightning Network [3] which is based on Bitcoin [1]; other approaches for payment channels have been proposed for example in [4]–[9]. We give an overview of other watchtower approaches and introduce IPFS. We present our approach for a decentralized watchtower based on IPFS in Section III. We discuss the properties of the approach and compare it to other approaches in Section IV and conclude in Section V.

This work was supported by funding from the topic Engineering Secure Systems of the Helmholtz Association (HGF) and by KASTEL Security Research Labs.

II. Background and Related Work

A. Payment Channels

Payment channels are used to perform transactions between two users without writing to the blockchain. Thereby, they do not only increase the transaction throughput of the underlying blockchain but also provide improved privacy properties. To create a payment channel, two parties lock funds via a funding transaction into a shared account. The funding transaction represents the initial state of how funds in the channel are distributed and is published to the blockchain. Transactions between the two parties of the channel are performed by updating the state to a new distribution of funds. To close the channel, the final distribution of the locked funds in the channel is published to the blockchain. Independent of the number of transactions performed inside the payment channel, only two transactions are published to the blockchain: one transaction to open the channel and one transaction to close the channel.

The security model of payment channels assumes that one of the parties in the channel is dishonest and might try to defraud the other party. A protocol for payment channels must ensure that an honest party finally receives the correct balance even if the other party publishes an outdated state or stops interacting. The Lightning Network's protocol uses the following construction to enforce this property: Each state is encoded in a commitment transaction. Both parties have the other party's signature to the commitment transaction so that they can close the payment channel in the latest state by publishing the latest commitment transaction on the blockchain. To disincentivize a dishonest party from publishing an outdated commitment transaction, a commitment transaction's funds cannot directly be spent but there is a time window during which the transaction can only be spent using a revocation transaction. A revocation transaction transfers all funds of the channel to the party who did not publish the outdated transaction. When a commitment transaction is superseded by a new commitment transaction during a state update, each party receives the revocation transaction for the outdated commitment transaction. This revocation mechanism requires a party to regularly watch the blockchain for outdated commitment transactions and react by publishing the respective revocation transaction.

B. Watchtowers

The requirement of liveness mentioned above, i.e., to regularly watch the blockchain for outdated commitment transactions, can be fulfilled by outsourcing the task of watching the blockchain to a third party, a watchtower. A watchtower watches the blockchain for outdated commitment transactions and publishes revocation transactions in place of the user who was defrauded. While a user of a payment channel typically only stores a set of secrets required to create revocation transactions, the user cannot simply share these secrets with a watchtower: the watchtower could use the revocation secrets to claim all funds in a payment channel for itself. Therefore, in most watchtower approaches, the user shares already signed revocation transactions with the watchtower and the watchtower publishes these transactions when needed. However, as we will see in the next paragraph, this sharing of signed revocation transactions creates some storage burden on the side of the watchtower.

1) Monitor: The approach of the Monitor watchtower [10] relies on a trusted third party that receives all the revocation transactions of a channel and publishes a revocation transaction in case an outdated commitment transaction is found on the blockchain. To prevent the watchtower from observing all transactions inside the payment channel, the revocation transactions are encrypted with the second half of the identifier (id) of their matching commitment transaction. The first half of the commitment transaction id is shared with the watchtower alongside the encrypted revocation transaction and is used to identify commitment transactions on the blockchain. If the watchtower finds a transaction on the blockchain whose first half of its id matches a stored value, the watchtower tries to decrypt the associated encrypted revocation transaction with the second half of the transaction id. If this succeeds, the watchtower publishes the revocation transaction.

The watchtower needs to store encrypted revocation transactions for all old states of all watched payment channels. Hence, the watchtower has high storage requirements. Another challenge is the handling of spam: As the watchtower only stores entries that contain half of a transaction id associated with encrypted data, the watchtower cannot distinguish authentic entries from random data.

2) Outpost: The Outpost watchtower [11] also relies on a trusted third party that provides the watchtower service. To reduce the storage requirements for the watchtower, the encrypted revocation transactions are stored in auxiliary transactions that are published with the commitment transaction. Therefore the watchtower only needs to store the decryption keys for the revocation transactions. To use an Outpost watchtower, a party of a payment channel has to share these encryption keys with the watchtower provider. While the approach of Outpost reduces the storage requirement for the watchtower, storing encrypted transactions on the blockchain increases the cost for users as they have to pay higher transaction fees.

3) TEE Guard: The TEE Guard watchtower [12] uses Trusted Execution Environments to provide a watchtower approach that only requires constant memory. The providers run watchtowers inside secure enclaves with access to the blockchain. Because the watchtower is run inside the enclave, the data and code inside is protected, even from the providers. Therefore a customer can share its revocation keys with the watchtower and benefit from the storage savings. The enclaves can also be interconnected to verify that a specific enclave was running at a specific time.

4) Other Watchtower Approaches: Other approaches for watchtowers have been proposed that require changes to the underlying Bitcoin protocol [13], [14] or are based on Ethereum [15], [16] which offers a richer feature set for smart contracts compared to Bitcoin. Another line of research

explored how to integrate watchtowers into the payment channel protocol [17]–[19]. Our approach presented in this paper is deployable for Bitcoin and can be used by the deployed Lightning Network with only a small set of changes. However, the proposed approach can also be transferred to other blockchains and can be combined with existing approaches for watchtowers.

C. IPFS

The InterPlanetary File System (IPFS) [2] is a decentralized file system and a peer-to-peer network. Files in IPFS are addressed by their content identifier (CID) which is a content based address. Therefore, a change in the content of a file means that the changed file will be addressed by a new CID. This change of CID is an issue in case an IPFS file is to be shared with other people via its CID, but its content should at the same time be modifiable. This issue can be addressed via the InterPlanetary Name System.

1) InterPlanetary Name System: The problem of addressing IPFS files that need to be changed with a single static identifier is addressed by the InterPlanetary Name System (IPNS) which provides dynamic IPNS addresses. The IPNS is based on asymmetric cryptography. To create a new IPNS address, a new key pair is created with the hash of the public key acting as the address. The private key is used to sign new address records that contain the information on where the IPNS address points to. These IPNS addresses can be used to reference CIDs of IPFS files.

Hence, if a file in IPFS needs to be modifiable, this file can be shared via an IPNS address. When the file is changed, the owner of the private key for this IPNS address updates the address to point to the new CID.

2) Persistence: In order for files to be accessible via IPFS, they have to be hosted by at least one node of the network. Hosting an IPFS file is also called pinning. Because we have to assume that the machine of a channel party is not permanently online, pinning files on a local machine is not sufficient. Therefore separate offsite machines like rented virtual machines would be necessary to make sure the pinned file stays available. Another option are third party pinning services, which provide the pinning of IPFS files for free or in exchange for payment. Examples of third party pinning services are Pinata[1] (paid) or nft.storage[2] (free).

III. Decentralized Watchtowers Approach

The essential concept of the decentralized watchtowers approach works as follows: A user who participates in a payment channel and wants to involve a watchtower, creates a Channel Justice File $F_{justice}$ and stores the file on IPFS. The Channel Justice File $F_{justice}$ contains a list of partial ids of outdated commitment transactions, encrypted revocation transactions, and encrypted reward keys that a watchtower can use to claim a reward. Each commitment transaction contains the IPNS address of the Channel Justice File $F_{justice}$. If a watchtower who watches the blockchain for commitment transactions, observes an outdated commitment transaction, the watchtower resolves the IPNS address, requests the Channel Justice File, and extracts and publishes the revocation transaction. The revocation transaction includes an additional reward output that can be spent using the private key which is shared alongside the revocation transactions in the Channel Justice File. This reward can be claimed by the watchtower if a fraud is prevented. In the following, we describe the Channel Justice File, the workflows for the payment channel users as well as for a watchtower, and reward offering in more detail.

A. Channel Justice File

All data required for a watchtower is stored in the Channel Justice File $F_{justice}$. For all but the latest commitment transactions in the payment channel, $F_{justice}$ contains the first half of the commitment transaction id ctx_{txid} and an encryption of the corresponding revocation transaction and a private key to spend the reward output in the revocation transaction.

The first half of the commitment transaction id ctx_{txid} is used as an identifier to find the row in the file for the corresponding commitment transaction. The second half of the ctx_{txid} is used as encryption key to encrypt the revocation transaction and the private key. This encryption is necessary because the Channel Justice File $F_{justice}$ is publicly available via IPFS. By encrypting the revocation transaction and the private key with the second half of the commitment transaction id ctx_{txid}, a third party can only decrypt them if the matching commitment transaction has already been published to the blockchain. Hence, a third party cannot learn information about how funds are distributed inside the payment channel. The idea of using the first half of the ctx_{txid} as an identifier and the second half as the encryption key was first introduced in the Monitor approach [10].

Format: Each row of the Channel Justice File contains the following space separated fields where $\texttt{ENC}(data, key)$ represents a symmetric encryption algorithm that encrypts $data$ with the provided key:

- First half of the commitment transaction id: $ctx_{txid}\texttt{[0:15]}$ (size: 16 bytes)
- Revocation transaction rtx that is encrypted with the second half of the commitment transaction id: $\texttt{ENC}(rtx, ctx_{txid}\texttt{[16:31]})$ (size[3]: $\sim$ 450 bytes)
- Private key rk for the reward output that is encrypted with the second half of the commitment transaction id: $\texttt{ENC}(rk, ctx_{txid}\texttt{[16:31]})$ (size: 32 bytes)

B. Workflow of Payment Channel User

1) Publishing $F_{justice}$: To be able to publish and update files in IPFS via a static identifier an IPNS address is required. Therefore the first step when creating a new payment channel is creating a new IPNS key pair which provides an IPNS address. When new transactions are performed, the IPNS

[1]https://www.pinata.cloud

[2]https://nft.storage

[3]The size of the revocation transaction is estimated based on the estimations by Khabbazian et al. [11] (300-350 bytes) with the addition of the reward output and the output that returns the IPNS address of $F_{justice}$.

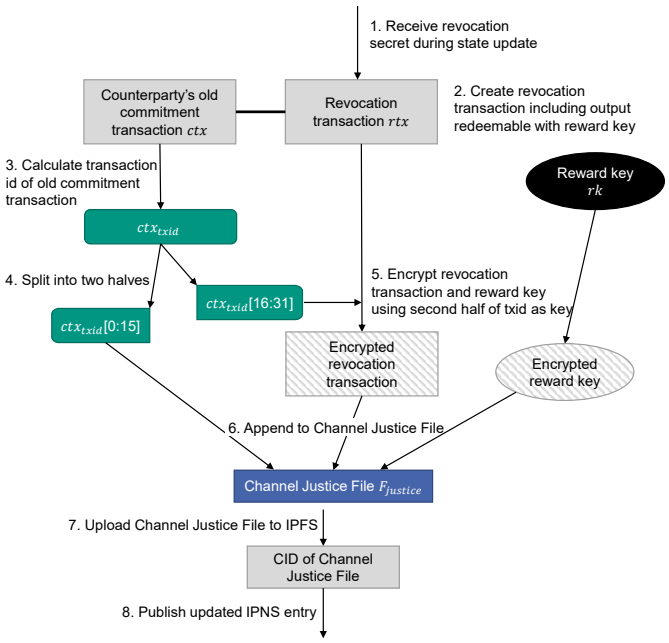

Fig. 1. Workflow of a user during a state update in the payment channel. The user creates the revocation transaction and a reward key, encrypts them with the second half of the old commitment transaction id and adds this data to the Channel Justice File.

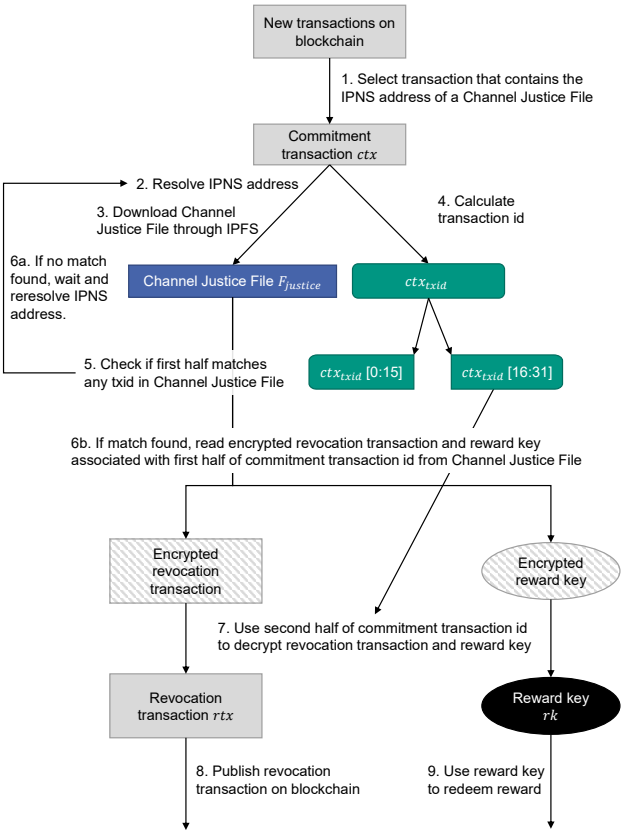

Fig. 2. A watchtower watches the blockchain for new transactions and processes commitment transactions transactions that contain the IPNS address of a Channel Justice File. If the observed commitment transaction was outdated, the watchtower decrypts and publishes the revocation transaction and is rewarded using the revocation transaction's reward output.

address of the previously created key pair is added to the commitment transactions of the opposing party. This is done with the with the `OP_RETURN` instruction. The `OP_RETURN` instruction stores arbitrary data like the IPNS address of $F_{justice}$ in a separate output. To make it easier to check if a transaction contains an IPFS address to $F_{justice}$ the '`CJF:`' prefix is added before the IPNS address.

The IPNS address is included in every commitment transaction but the IPNS address will be included at most once in the blockchain because at most one commitment transaction will be published.

2) Updating $F_{justice}$: With every new state in a payment channel, $F_{justice}$ needs to be extended (see Fig. 1). The new revocation transaction is built with the revocation secret received from the counterparty and contains the reward output for the watchtower. The private key of the Bitcoin address receiving the incentive output and the revocation transaction are encrypted with the second half of the commitment transaction id. Together with the first half of the commitment transaction id, both encryptions are appended to the Channel Justice File $F_{justice}$. After updating $F_{justice}$, it is shared via IPFS. Because the content of the file changed, it receives a new CID. Therefore, the CID that the IPNS address contained in the commitment transactions points to must be updated to point to the CID of the updated $F_{justice}$. After the IPNS address has been updated, watchtowers can read the Channel Justice File via the IPNS address in a published commitment transaction and revoke it if a matching revocation transaction exists.

C. Workflow of Watchtower

The main task of a watchtower is to check the blockchain for outdated commitment transactions (see Fig. 2). If a transaction containing an IPNS address to the Channel Justice File $F_{justice}$ is found, the watchtower resolves the IPNS address and reads $F_{justice}$. If $F_{justice}$ contains the revocation transaction for the commitment transaction, the watchtower decrypts the revocation transaction and publishes it on the blockchain. If $F_{justice}$ does not have a revocation transaction for the found commitment transaction, the IPNS address might not be updated yet or still needs some time to propagate. Therefore the watchtower will regularly try to resolve the IPNS address and check for a matching revocation transaction. If no revocation transaction was found within a certain amount of time, the watchtower will discard the IPNS address. In this case, the observed commitment transaction might be the latest commitment transaction and therefore no revocation is possible and needed.

D. Reward

Running a watchtower requires resources and therefore most people will not run a watchtower for free. To incentivize others

to run watchtowers, a reward is added that can be claimed by a watchtower that revokes fraudulent commitment transactions. This reward is offered by adding an additional output, the 'reward output', to the revocation transaction which can be spent using the reward key rk. This reward key is included in $F_{justice}$ and, thus, can be used by the watchtowers. Because this output can only be spent once, this reward creates a race for the watchtowers to publish the revocation transaction and be the first to spend the reward output.

IV. Evaluation & Discussion

A. Scalability of the Channel Justice File $F_{justice}$

The size of the Channel Justice File $F_{justice}$ of a payment channel grows with every transaction performed in the channel. For channels with a high number of transactions, the size of $F_{justice}$ increases considerably. A channel with one million transactions would have a Channel Justice File $F_{justice}$ with the size of 501 megabytes.[4] This can be problematic if the user that is sharing the Channel Justice File $F_{justice}$ has a slow upload speed: To be able to rely on the watchtower, the user only accepts new states of the payment channel after the previous $F_{justice}$ has been shared with a pinning service. Therefore the upload creates a limit for the transaction throughput of the channel. A simple fix for this problem would be to close a payment channel after a certain amount of transactions and open a new payment channel. Alternatively, the rate at which $F_{justice}$ grows can be reduced by storing the encrypted revocation transactions and the encrypted private keys of the reward output in separate files and not in the Channel Justice File $F_{justice}$ directly. The Channel Justice File $F_{justice}$ would then just be used as an index, containing references in the form of CIDs to the files containing the actual data. Depending on the hashing algorithm used, the size of the CID can vary. The current default hashing algorithm for CIDs in IPFS is SHA-256 [20] which creates a CID of 42 bytes. That means an hierarchical $F_{justice}$ of a channel with one million transactions would only have a size of 60 megabytes[5] instead of 501 megabytes.

B. Privacy

While the Channel Justice File $F_{justice}$ is publicly available over IPFS, it cannot be linked to a payment channel as long as the channel is not closed using a commitment transaction. After a commitment transaction has been published, the IPNS address contained in the commitment transaction can be used to link $F_{justice}$ to the payment channel. However, only the number of transactions that occurred in the payment channel is leaked because all relevant data contained in the Channel Justice File $F_{justice}$ is encrypted.

[4] 16 bytes for the first half of the commitment transaction id, 450 bytes for the revocation transaction, 32 bytes for the reward key rk and 3 bytes for spaces and the newline result in 501 bytes per line. 501 bytes * 1.000.000 $\approx$ 501 megabytes

[5] 16 bytes for the first half of the commitment transaction id, 42 bytes for the CID and 2 bytes for the space and the newline result in 60 bytes per line. 60 bytes * 1.000.000 $\approx$ 60 megabytes.

The decentralized watchtower approach has similar privacy properties as the Monitor and Outpost watchtowers. Even when a commitment transaction has been published, a third party can only gain knowledge about the number of transactions that occurred in a payment channel because the revocation transactions in $F_{justice}$ are encrypted. The distribution of the funds inside the payment channel is private and only the final distribution of funds is revealed once the channel is closed. The TEE Guard watchtower provides more privacy compared to the decentralized watchtower approach: If we assume that the secure enclaves are truly secure, then the provider has no access to the enclave's data and, therefore, gains no knowledge about the payment channel.

C. Cost

The watchtower needs access to the Bitcoin blockchain which is done by running a Bitcoin node. The watchtower also needs a way to access files from the IPFS network. This can be done by either using an IPFS gateway or running an IPFS node. Because the watchtower only has to check new transactions on the blockchain for an IPNS address, the required processing power is very low. In case a fraud happens, the downloading and reading of the Channel Justice File $F_{justice}$ requires additional bandwidth and processing power. Therefore running a watchtower is cheap and can even be profitable if some rewards are claimed successfully now and then.

However, the impact of the rewards is estimated to be rather low because they require a fraud to happen in the first place. Also, multiple watchtowers might engage in a race about the reward. In this case, a watchtower can improve its chance to claim the reward by increasing the transaction fee of the transaction that spends the reward output which at the same time reduces the watchtower's profit.

The cost on the user side depends on the cost of pinning the Channel Justice File $F_{justice}$ off site. This could be practically free if there are altruistic pinning services or if there exists a community that pins each other's files. However, pinning could be associated with costs if a more advanced strategy is chosen such as pinning the $F_{justice}$ on multiple servers in multiple locations. Most pinning services provide this functionality and also directly integrate with the IPFS client.

Comparing the cost of the decentralized watchtower approach to other watchtowers is not trivial because the cost of using a decentralized watchtower strongly depends on the way that is used to persist $F_{justice}$ in IPFS.

D. Reliability

The reliability of the watchtower depends mostly on the availability of the Channel Justice File $F_{justice}$. Ideally, the IPNS address of $F_{justice}$ is never revealed to the public. However, both parties in a payment channel know the address and if one party is dishonest, they might try to deplete the bandwidth of the hosts pinning the Channel Justice File $F_{justice}$ or use other ways to make the file unavailable (e.g., eclipse attacks on the storage hosts [21]). This could be

mitigated by using multiple servers with high bandwidth or even a DoS protection service.

Because users of payment channels do not interact directly with watchtowers, there are no contracts or guarantees in place that can be enforced. Therefore, a user of a payment channel who uses decentralized watchtowers has to trust that someone else runs a watchtower. This could be mitigated by having a trusted entity in the payment channel community that provides watchtowers. Our approach to incentivize running a watchtower is the reward output. However, this reward can only be claimed if a fraud happens and only one watchtower can receive the payout even if multiple watchtowers are active.

V. Conclusion

This work introduced a novel approach for watchtowers for payment channels. The main idea of the approach is to replace the dependence on a specific watchtower provider by relying on a storage provider and an arbitrary watchtower. This idea is implemented by sharing the data required by watchtowers via IPFS and linking the data to the payment channel through an IPNS address in each commitment transaction. Our decentralized approach still requires cooperation of other parties to make it possible for payment channel users to go offline for a certain amount of time without risking to loose their funds. However, by separating the tasks of storage and watching, specific incentive mechanisms can be used and, for example, approaches for incentivization found by the IPFS community can be used for the storage provider.

The decentralized approach, as presented here, leaves open questions regarding a precise analysis of privacy and scalability properties and regarding incentivization of watchtowers and storage hosts. In future work, we plan to approach these challenges, improve and extend the approach, and to implement a prototype.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," Tech. Rep., 2008.

[2] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," *arXiv:1407.3561 [cs]*, Jul. 2014, arXiv: 1407.3561. [Online]. Available: http://arxiv.org/abs/1407.3561

[3] J. Poon and T. Dryja, "The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments," Tech. Rep., 2016.

[4] C. Decker and R. Wattenhofer, "A Fast and Scalable Payment Network with Bitcoin Duplex Micropayment Channels," in *Stabilization, Safety, and Security of Distributed Systems*, A. Pelc and A. A. Schwarzmann, Eds. Cham: Springer International Publishing, 2015, vol. 9212, pp. 3–18. [Online]. Available: http://link.springer.com/10.1007/978-3-319-21741-3_1

[5] M. Green and I. Miers, "Bolt: Anonymous Payment Channels for Decentralized Currencies," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, Oct. 2017, pp. 473–489. [Online]. Available: https://doi.org/10.1145/3133956.3134093

[6] J. Lind, O. Naor, I. Eyal, F. Kelbert, E. G. Sirer, and P. Pietzuch, "Teechain: a secure payment network with asynchronous blockchain access," in *Proceedings of the 27th ACM Symposium on Operating Systems Principles - SOSP '19*. Huntsville, Ontario, Canada: ACM Press, 2019, pp. 63–79. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3341301.3359627

[7] S. Dziembowski, L. Eckey, S. Faust, and D. Malinowski, "Perun: Virtual Payment Hubs over Cryptocurrencies," in *2019 IEEE Symposium on Security and Privacy (SP)*. San Francisco, CA, USA: IEEE, May 2019, pp. 106–123. [Online]. Available: https://ieeexplore.ieee.org/document/8835315/

[8] M. Jourenko, M. Larangeira, and K. Tanaka, "Payment Trees: Low Collateral Payments for Payment Channel Networks," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, N. Borisov and C. Diaz, Eds. Berlin, Heidelberg: Springer, 2021, pp. 189–208.

[9] L. Aumayr, M. Maffei, O. Ersoy, A. Erwig, S. Faust, S. Riahi, K. Hostáková, and P. Moreno-Sanchez, "Bitcoin-Compatible Virtual Channels," in *2021 IEEE Symposium on Security and Privacy (SP)*, May 2021, pp. 901–918, iSSN: 2375-1207.

[10] T. Dryja, "Unlinkable Outsourced Channel Monitoring," Milan, Oct. 2016. [Online]. Available: https://scalingbitcoin.org/milan2016/presentations/D1%20-%208%20-%20Tadge%20Dryja.pdf

[11] M. Khabbazian, T. Nadahalli, and R. Wattenhofer, "Outpost: A Responsive Lightweight Watchtower," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, ser. AFT '19. New York, NY, USA: ACM, Oct. 2019, pp. 31–40. [Online]. Available: https://doi.org/10.1145/3318041.3355464

[12] M. Leinweber, M. Grundmann, L. Schönborn, and H. Hartenstein, "TEE-Based Distributed Watchtowers for Fraud Protection in the Lightning Network," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, ser. Lecture Notes in Computer Science, C. Pérez-Solà, G. Navarro-Arribas, A. Biryukov, and J. Garcia-Alfaro, Eds., vol. 11737. Springer International Publishing, Sep. 2019, pp. 177–194.

[13] O. Osuntokun, "Hardening Lightning," Stanford University, Jan. 2018. [Online]. Available: https://youtu.be/V3f4yYVCxpk

[14] G. Avarikioti, F. Laufenberg, J. Sliwinski, Y. Wang, and R. Wattenhofer, "Towards Secure and Efficient Payment Channels," *arXiv:1811.12740 [cs]*, Nov. 2018, arXiv: 1811.12740. [Online]. Available: http://arxiv.org/abs/1811.12740

[15] P. McCorry, S. Bakshi, I. Bentov, S. Meiklejohn, and A. Miller, "Pisa: Arbitration Outsourcing for State Channels," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, ser. AFT '19. New York, NY, USA: ACM, Oct. 2019, pp. 16–30. [Online]. Available: https://doi.org/10.1145/3318041.3355461

[16] B. Liu, P. Szalachowski, and S. Sun, "Fail-safe Watchtowers and Short-lived Assertions for Payment Channels," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 506–518. [Online]. Available: https://doi.org/10.1145/3320269.3384716

[17] Z. Avarikioti, E. Kokoris-Kogias, R. Wattenhofer, and D. Zindros, "Brick: Asynchronous Incentive-Compatible Payment Channels," in *Financial Cryptography and Data Security*, 2021, p. 29.

[18] Z. Avarikioti, O. S. Thyfronitis Litos, and R. Wattenhofer, "Cerberus Channels: Incentivizing Watchtowers for Bitcoin," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, J. Bonneau and N. Heninger, Eds. Cham: Springer International Publishing, Feb. 2020, pp. 346–366.

[19] A. Mirzaei, A. Sakzad, J. Yu, and R. Steinfeld, "FPPW: A Fair and Privacy Preserving Watchtower For Bitcoin," Tech. Rep. 117, 2021. [Online]. Available: https://eprint.iacr.org/2021/117

[20] IPFS. Content addressing and cids. [Online]. Available: https://docs.ipfs.io/concepts/content-addressing/

[21] B. Prünster, A. Marsalek, and T. Zefferer, "Total Eclipse of the Heart – Disrupting the InterPlanetary File System," *31st USENIX Security Symposium*, Sep. 2021. [Online]. Available: https://blog.ipfs.io/2020-10-30-dht-hardening/