Enabling Fast Settlement in Atomic Cross-Chain Swaps

Feng Zhuo[1,2], Zhaoxiong Song[1], Linpeng Jia[1], Hanwen Zhang[1,2], Zhongcheng Li[1,2], and Yi Sun[1,2(✉)]

[1] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{zhuofeng19s,songzhaoxiong,jialinpeng,hwzhang,zcli,sunyi}@ict.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China

Abstract. Cross-chain swaps have emerged as a critical and indispensable blockchain application, expanding the circulation scope of digital assets from a single chain to multiple chains, thus improving their flexibility and usability. *Atomic swap* acts as the key technology for achieving cross-chain swaps without the need for cross-chain infrastructures or trusted third parties. However, existing atomic swap solutions suffer from the *lockup griefing* problem, which prevents fast settlement of parties' assets when a swap is cancelled midway. While some research works provide solutions to compensate parties experiencing lockup griefing, there is currently no work focusing on preventing lockup griefing through mechanism design. To fill the gap, this paper introduces a novel atomic cross-chain swap protocol, called *Fast Settlement Swap (FS-Swap)*. By analyzing the fundamental causes of lockup griefing, FS-Swap provides a mechanism design that prevents the occurrence of lockup griefing in the first place. The experimental results demonstrate that, when a swap is cancelled midway, FS-Swap significantly reduces asset locking time by over 99% compared to existing atomic swap solutions, decreasing it from several hours to less than 2.1 min.

Keywords: Blockchain applications · Cross-chain swap · Atomic swap · Lockup griefing · HTLC

1 Introduction

Since its inception, blockchain technology has garnered widespread global attention due to its decentralized and tamper-resistant characteristics. Over the past decade, a multitude of diverse blockchain systems have emerged, with thousands of digital currencies being issued and circulated on those blockchains. Typically,

Supported by the National Key Research and Development Program of China (2021YFB2701103); the National Natural Science Foundation of China (U22B2032); ICT-SSC Blockchain Joint Lab; Hainan Zhongke Computing Blockchain Innovation Academy.

H. Duan et al. (Eds.): SecureComm 2023, LNICST 567, pp. 395–412, 2025.
https://doi.org/10.1007/978-3-031-64948-6_20

a digital asset can only circulate within a single chain, severely limiting the flexibility and usability of digital assets. To expand the circulation scope of digital assets, researchers have been studying the exchange of digital assets between different blockchains, known as *cross-chain swaps*.

Cross-chain swap refers to the exchange of assets between two untrusted parties on different blockchains. For example, when Alice wants to exchange her Bitcoin for Bob's Ether and Bob agrees to the swap, Alice needs to transfer her Bitcoin to Bob, while Bob needs to transfer his Ether to Alice. However, due to the lack of trust between Alice and Bob, neither party is willing to be the first to transfer their assets to the other, as there is a possibility that the other party reneges on the agreement and refuses to transfer their assets. Consequently, direct swap between the two parties becomes unattainable.

In the past, users had to entrust their assets to *centralized exchanges (CEX)* to facilitate cross-chain swaps. However, this exposed users to the risk of malicious CEXs stealing their assets, as evidenced by numerous cases [2,3,33]. Many cross-chain projects, such as Polkadot [13] and Cosmos [6], have developed cross-chain infrastructures like cross-chain bridges and relay chains to facilitate cross-chain swaps. However, these infrastructures not only come with high construction and maintenance costs but also present challenges in ensuring security [7,8]. With the introduction of *atomic swap* [1], users can securely exchange assets by entrusting their assets to publicly auditable and tamper-proof smart contracts, without the need for trusted third parties or cross-chain infrastructures. As atomic swap eliminates the risks associated with centralization and bypasses the high costs involved in building and maintaining cross-chain infrastructures, it has been widely implemented in *decentralized exchanges (DEX)* and cross-chain trading platforms [4,9,10,12].

However, existing atomic swap solutions suffer from the *lockup griefing* problem [18]. Atomic swap completes a swap by parties following a series of steps defined by the atomic swap protocol to publish transactions on the blockchain. However, due to factors such as asset price fluctuations, users often abandon the swap halfway, deviating from the protocol's steps, causing the swap can not be completed. In such cases, the protocol needs to refund the assets locked in smart contracts to the respective parties. Unfortunately, existing solutions require significant waiting times for refunds, resulting in a prolonged locking period for users' assets, known as *lockup griefing* or *sore loser attack* [35]. The duration of asset locking varies depending on different circumstances, typically ranging from several hours to several days [9,16,23,26]. Lockup griefing is common, as indicated by some research studies [21,24,30], which show that 20% to 70% of swaps between volatile cryptocurrencies are cancelled midway by users.

A user with ID "ZmnSCPxj" pointed out lockup griefing brings a risk of monetary loss for the parties with significant asset price fluctuations at the Lightning-dev mailing list [11]. Subsequently, numerous studies [17,18,24,25,27,35] have focused on lockup griefing. However, the idea behind these studies is to penalize users who abandon the swap and compensate those who experience prolonged locking after lockup griefing occurs, rather than preventing lockup griefing from occurring in the first place through mechanism design.

Lockup griefing should not be considered the norm, and the swap protocol should guarantee fast settlement, ensuring that users receive their rightful assets promptly, regardless of whether the swap is completed. Therefore, this paper proposes a novel atomic cross-chain swap protocol, *Fast Settlement Swap*, abbreviated as *FS-Swap*, which solves the problem of lockup griefing in existing solutions and achieves fast settlement. Compared to existing solutions, FS-Swap significantly reduces the locking time of users' assets.

By analyzing existing solutions, we find the underlying reason for lockup griefing is that these solutions rely on a pre-defined *timelock* for refunds, which often has a substantial value, requiring users to wait a long time until the expiration of this timelock to receive their refunds. Thus, FS-Swap introduces a rapid refund mechanism based on the *hashlock*. In cases where the swap cannot be completed, users can swiftly reclaim their assets by this mechanism.

The experimental results clearly illustrate the substantial reduction in asset locking time achieved by FS-Swap. When a swap is cancelled, compared to existing atomic swap solutions, FS-Swap achieves a remarkable reduction of over 99% in asset locking time. Specifically, the locking duration is reduced from the previous several hours to an impressive duration of less than 2.1 min.

2 Background

This section provides an introduction to the necessary background knowledge, including blockchain, smart contract, cross-chain swap, atomic swap, and lockup griefing.

2.1 Blockchain, Smart Contract and Cross-Chain Swap

Blockchain is a publicly readable and tamper-proof distributed ledger that records all valid transactions initiated by users within a blockchain network. Unlike traditional centralized databases, in a blockchain system, the verification and storage of transactions are collectively performed by all or multiple nodes in the network, ensuring transaction security and trustworthiness.

Smart contracts are programmable contracts based on blockchain technology. They are essentially pieces of code that run on the blockchain, defining certain operations and their triggering conditions. When the triggering conditions for a operation are satisfied, users can trigger execution of the operation by publishing a transaction. Once the operation is triggered, it will be executed automatically and uninterruptedly. Smart contracts enable reliable transactions and collaborations among users without relying on intermediaries. The programmability of smart contracts has opened up possibilities for complex blockchain applications, including cross-chain swap.

Cross-chain swap refers to the exchange of assets between different blockchain systems. In the blockchain ecosystem, different blockchain systems maintain their own independent ledgers, which restricts the free flow of digital assets across different chains. The goal of cross-chain swap is to solve this problem and enable safe

and unrestricted asset exchange between different blockchains. This exchange is important as it expands the circulation scope of digital assets, improving their flexibility and usability.

2.2 Atomic Swap

Atomic swap is a mainstream solution for achieving cross-chain swap. The goal of atomic swap is to enable two parties to exchange assets "atomically". *Atomicity* means that in a swap, either both parties receive each other's assets or both parties retain their original assets.

Existing atomic swap protocols ensure atomicity based on Two-Phase Commit [22] from the database field. In prepare phase, each party individually locks their assets in a separate smart contract, which is called *escrow contract* in some place. The two contracts both have a *transfer* operation and a *refund* operation. The *transfer* operation unlock the assets in contract and transfer them to the counterparty, for example, if a contract locks Alice's assets, the *transfer* operation of the contract will transfer the assets to Bob. On the contrary, the *refund* operation unlock the assets and return them to their original party. The *transfer* operations of the two contracts are triggered by the same conditions, and so do the their *refund* operations. In commit phase, either both *transfer* operations are triggered, then both contracts transfer the assets to the counterparty, or both *refund* operations are triggered, then both contracts return the assets to their original party.

The classic atomic swap protocol was proposed by Tier Norlan on the Bitcoin forum in 2013 [1]. Currently, this protocol is widely adopted by decentralized exchanges and cross-chain trading platforms as the primary atomic swap protocol. *Norlan's protocol* uses *Hashed Timelocks Contract (HTLC)* as the escrow contracts. HTLC sets the triggering condition for *transfer* operation using a *hashlock* and the triggering condition for *refund* operation using a *timelock*. The hashlock is essentially a hash value, and the timelock is a specific time point. Assume a HTLC locks Alice's assets, if Bob provides the correct pre-image of the hash value to the contract before the time, the hashlock will be unlocked, triggering the *transfer* operation. Otherwise, the timelock will be unlocked after the time, triggering the *refund* operation. Following Norlan's protocol, other atomic swap protocols also adopt the same triggering conditions.

2.3 Lockup Griefing

Lockup greifing in existing atomic swap protocols stems from using timelocks as the triggering conditions for *refund* operations. In this section, we take the Norlan' protocol as an example to depict the scenarios in which lockup griefing occurs and analyze the asset locking durations resulting from timelocks.

An HTLC is represented as $< h, T >$, where h denotes the hashlock and T denotes the timelock. Suppose Alice wants to exchange her x_1 $coin_1$ for Bob's x_2 $coin_2$, where $coin_1$ is a digital asset on blockchain BC_1, $coin_2$ is a digital asset on blockchain BC_2. Following Norlan's protocol, the swap is completed through

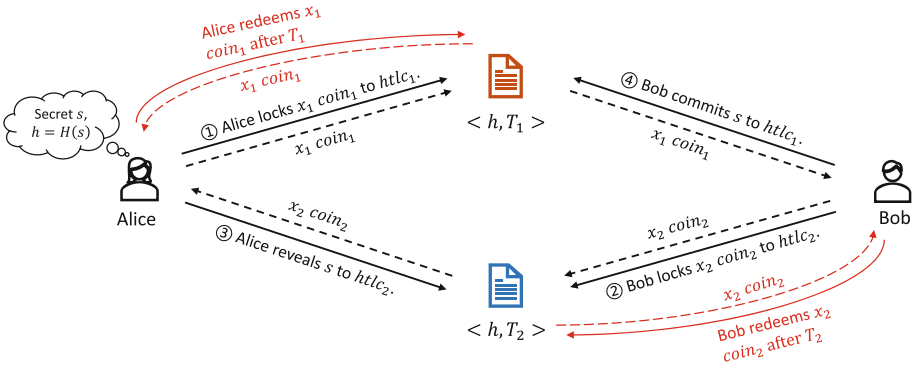

Fig. 1. The Steps of Norlan's Protocol. Solid arrows represent actions performed by parties, while dashed arrows indicate the direction of asset flow.

the following steps (as shown in Fig. 1): ① Alice generates a secret value s and calculates its hash value $h = Hash(s)$. She then locks x_1 $coin_1$ in contract $< h, T_1 >$ $(htlc_1)$ on BC_1. ② After Alice locks her assets, Bob locks x_2 $coin_2$ in contract $< h, T_2 >$ $(htlc_2)$ on BC_2, where $T_2 < T_1$. ③ After Bob locks his assets, Alice commits the secret value s to $htlc_2$ before T_2 to claim x_2 $coin_2$. ④ Once Bob sees that Alice has revealed s, he submits s to $htlc_1$ before time T_1 to claim x_1 $coin_1$, completing the swap.

If Bob abandons the swap after Alice has locked her assets, Alice will need to reclaim her assets after T_1. If Alice abandons the swap after Bob has locked his assets, Alice needs to reclaim her assets after T_1 and Bob needs to reclaim his assets after T_2.

Assume the time required for steps ①②③④ in Norlan's protocol is t_1, t_2, t_3, and t_4 respectively, and the swap starts at time T_0, then T_1 should satisfy $T_1 \geq T_0 + t_1 + t_2 + t_3 + t_4$, and T_2 should satisfy $T_2 \geq T_0 + t_1 + t_2 + t_3$ for the swap process to be completed successfully. Additionally, T_1 and T_2 should satisfy $T_1 - T_2 \geq t_4$ to ensure that Bob has enough time to claim Alice's assets after Alice has received Bob's assets.

The values of t_1, t_2, t_3, and t_4 are dynamic and, under normal circumstances, the time taken to execute a step is equivalent to the *block interval* of the blockchain on which the transaction occurs [29]. However, in special cases such as temporary user disconnections, blockchain congestion, or censorship attacks [26,31,34], the time taken to execute a step may be prolonged. T_1 and T_2 should be agreed upon by both parties before the swap begins. To ensure that the values of T_1 and T_2 always meet the requirements, they should be set based on the maximum possible values of t_1, t_2, t_3, and t_4.

Let Δ be enough time for one party to publish a transaction on a specific blockchain, and for the other party to detect the transaction [19]. We define Δ_1 for chain BC_1 and Δ_2 for chain BC_2, then Δ_1 is the maximum possible value of t_1 and t_3, and Δ_2 is the maximum possible value of t_2 and t_4. In practical

systems, Δ is usually ten times [9] or hundreds of times [16,23,26,28] of the block interval. Therefore, if both BC_1 and BC_2 are blockchains with second-level block intervals, T_1 and T_2 reach several hours. If BC_1 or BC_2 is blockchain like Bitcoin with minute-level block intervals, T_1 and T_2 may span several days.

Based on the above analysis, we can conclude that timelocks that require an extended waiting time are the fundamental cause of lockup griefing. If a mechanism could be established to offer an event-triggered refund method, allowing users to proactively trigger an event for rapid refunds when they discover that a swap cannot be completed, it would enable swift asset settlement. FS-Swap accomplishes this through its design.

3 Design

3.1 Assumptions

Blockchain. For the convenience of analysis, we assume that the participating blockchains in the swap support smart contracts, do not experience forks, and have a common supported hash function. They are able to synchronize their clocks with each other. These assumptions are same as the assumptions of the existing atomic swap solutions.

Participating Users. Each participant has accounts on two blockchains, one for receiving assets and one for sending assets. They can read the contents of these two chains and can publish transactions on them. These assumptions are same as the assumptions of the existing atomic swaps. We also assume that participants have a small amount of assets on the blockchain where they receive assets. Considering that participants have an account on that chain, we believe this assumption is not difficult.

Requirement Matching. The participating parties in the swap are aware of the values of hashlocks and timelocks. This is reasonable because there are already many third-party platforms [2,5] that collect and match users' swap requests, providing off-chain communication channels for matched users.

3.2 Overview

FS-Swap aims to solve the issue of lockup griefing and achieve fast settlement. To accomplish this, the core idea is to provide an event-triggered mechanism for quick refunds. In most cases where a swap cannot be completed, users can promptly unlock their assets through this mechanism, eliminating the need for waiting. Only in rare instances, when a swap is canceled due to a user being permanently offline, users would reclaim their assets by timelocks.

FS-Swap implements the mechanism based on the hashlock. FS-Swap uses a hashlock in the escrow contract, FS-HTLC, to set the triggering condition for fast refund to the original party of the assets. The pre-image of the hashlock is exclusively held by the other party who is not the original owner of the assets.

When the swap cannot be completed, one party disclose the pre-image to help the other party retrieve assets. This design prevents a party from arbitrarily retrieving their assets when the swap has been partially completed. For example, if Alice has already taken Bob's assets, Alice cannot refund her own assets because the needed pre-image for this refund is only known by Bob, who will not disclose it.

However, the design that the pre-image for the refund of one party's assets is only known by the other party poses a challenge: The other party has no incentives to disclose the pre-image to help the party refund quickly. To address this issue, FS-Swap requires each party to lock a small amount of assets as collateral in the contract where the other party locks their assets. If, when a swap cannot be completed, one party discloses the pre-image to facilitate a fast refund for the other party before the timelock expires, the party can retrieve their collateral. Otherwise, after the timelock expires, the collateral will be compensated to the other party.

With the introduction of the collateral, two requirements are imposed on the parties. The first requirement is that each party must lock their assets only after ensuring that the other party has locked their collateral. Failing to adhere to this requirement may expose a party to the risk of lockup griefing without compensation. The second requirement is that each party must promptly retrieve their collateral when they realize that the swap cannot be completed. Failure to comply with this requirement may result in the loss of their collateral. For instance, if Bob locks his collateral but Alice delays locking her assets for an extended period, Bob needs to promptly retrieve his collateral to abort the swap. Otherwise, if Alice locks her assets close to the expiration time T_2, Bob may not have enough time to trigger a fast refund of Alice's assets, resulting in the loss of his collateral. We will give the latest time for a party to retrieve their collateral in Sect. 3.3.

3.3 Detailed Design

This section begins by providing a detailed design of the FS-HTLC contract, followed by illustrating the specific workflow of FS-Swap using the example in Sect. 2.3, where Alice exchanges x_1 $coin_1$ with Bob for x_2 $coin_2$.

An FS-HTLC is represented as $< h, h', T >$, where the hashlock h is used to set the triggering condition for the *transfer* operation. Similar to traditional HTLC, this operation unlocks the assets and transfers them to the counterparty. The hashlock h' is used to set the triggering condition for the *fast refund* operation, which *immediately* unlocks the assets and returns them to the original party. Additionally, the timelock T is used to set the triggering condition for *slow refund* operation, which unlocks the assets and returns them to the original party after T.

In an FS-HTLC contract, the assets of one party, assuming Alice, and the collateral of the other party, assuming Bob, are unlocked by the methods as follows: If the hashlock h is unlocked, Alice's assets will transfer to Bob and Bob's collateral will return to Bob. If the hashlock h' is unlocked, Alice's assets will return to Alice and Bob's collateral will return to Bob. If the timelock T

is unlocked, Alice's assets will still return to Alice but Bob's collateral will be compensated to Alice for her lockup griefing.

To illustrate the detailed design of FS-Swap, we assume that the requirement matching platform has facilitated the swap and provided two parties with the necessary information, including two parties' account addresses on the two chains and public keys. Assuming that Alice and Bob have already agreed on the value of the hash lock h, and only Alice knows its pre-image s. Assuming that Alice and Bob have already agreed upon reasonable timelock values for T_1 and T_2. The analysis of appropriate values for T_1 and T_2 will be discussed in Sect. 3.4.

In the following, we will begin by presenting the execution steps for completing a swap (as shown in Fig. 2), followed by providing strategies for aborting a swap to handle various scenarios in which the swap cannot be completed.

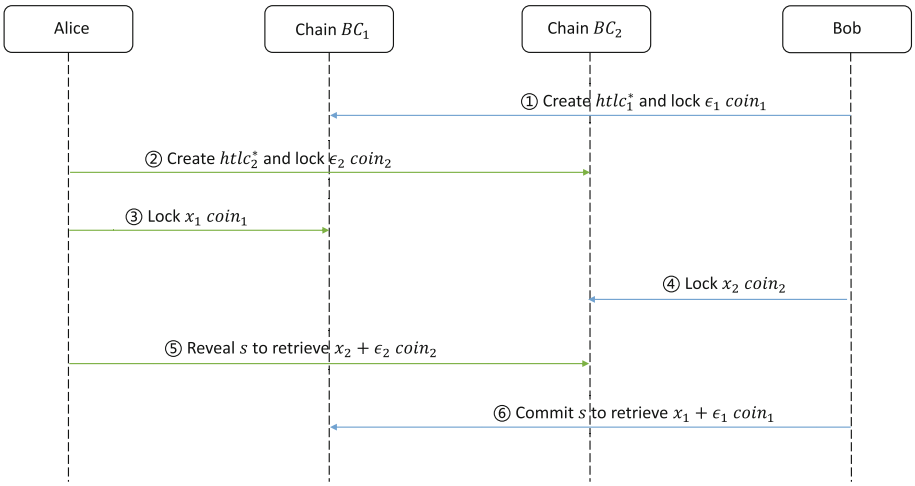

Fig. 2. FS-Swap's Execution Steps for Completing a Swap.

Steps for Completing a Swap. First, Alice and Bob create the escrow contracts and lock their collateral respectively as step ① and ②. It is important to note that the two steps are executed independently by Alice and Bob, thus they do not need to wait for each other.

① Bob generates a secret value s_1' and computes $h_1' = Hash(s_1')$. He then creates a contract $< h, h_1', T_1 >$ $(htlc_1^\star)$ on chain BC_1 and locks ϵ_1 $coin_1$ into $htlc_1^\star$ as his collateral.

② Alice generates a secret value s_2' and computes $h_2' = Hash(s_2')$. She then creates a contract $< h, h_2', T_2 >$ $(htlc_2^\star)$ on BC_2 and locks ϵ_2 $coin_2$ into $htlc_2^\star$ as her collateral.

Then, Alice and Bob lock their assets sequentially as step ③ and ④.

③ After Bob creates $htlc_1^\star$, Alice locks her assets of x_1 $coin_1$ into $htlc_1^\star$.

④ After Alice creates $htlc_2^\star$ and locks assets in $htlc_1^\star$, Bob locks the assets of x_2 $coin_2$ into $htlc_2^\star$.

Finally, Alice and Bob retrieve each other's assets sequentially as step ⑤ and ⑥.

⑤ After Bob locks his assets, Alice submits the secret value s to $htlc_2^\star$ to retrieve Bob's assets x_2 $coin_2$ and reclaim her collateral ϵ_2 $coin_2$.

⑥ Once Bob sees that Alice has reveaed s, he submits s to $htlc_1^\star$ to retrieve Alice's assets x_1 $coin_1$ and reclaim his collateral ϵ_1 $coin_1$, completing the swap.

Some steps have a specific deadline for execution, as shown in Fig. 3. If a step is not completed before its deadline, the swap cannot be successfully completed. The deadline for step ⑥ is T_1, and the deadline for step ⑤ is T_2, as imposed by the timelock restrictions. It is worth noting that the deadline for step ④ is $T_2 - \Delta_2$ (Δ_2 is the maximum possible time to execute a step on the blockchain BC_2, as defined in Sect. 2.3). This is because if Bob has not locked assets at time $T_2 - \Delta_2$, Alice will abort the swap by retrieving her collateral from $htlc_2^\star$. Since Alice needs to retrieve her collateral before T_2, she must allocate enough time (Δ_2) for the operation to assure the security of her collateral.

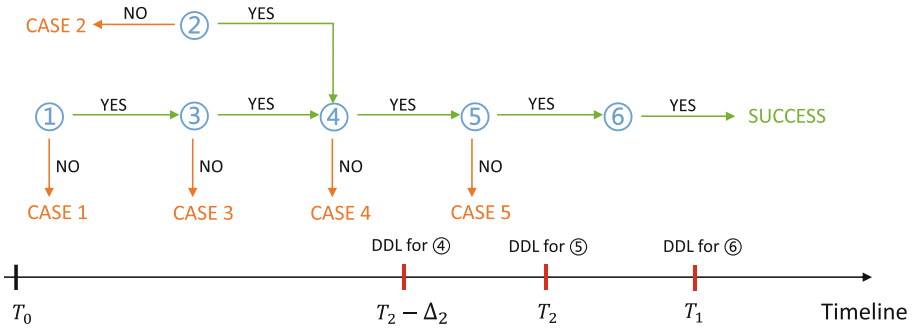

Fig. 3. FS-Swap's Process and Deadlines for Some Steps. "YES" means a step is completed, while "NO" means the step is not completed. "DDL" is an abbreviation for deadline.

Strategies for Aborting a Swap. Parties may abandon the swap before executing steps ①②③④⑤, as shown in Fig. 3. Once step ⑤ is completed and Alice has taken Bob's assets, Bob will not abandon the swap and definitely proceed with step ⑥. Below, we give the strategies for handling cases where parties exit before executing any of the steps ①②③④⑤.

CASE 1. If Bob exits before creating the contract in step ①, steps ③④⑤⑥ will not be executed. Consequently, Bob's collateral and assets, along with Alice's assets, will remain unlocked. If step ② has already been completed, then Alice's collateral has been locked, she needs to submit s_2' to $htlc_2^\star$ before T_2 to reclaim her collateral.

CASE 2. If Alice exits before creating the contract in step ②, steps ④⑤⑥ will not be executed. Consequently, Bob's assets and Alice's collateral will remain unlocked. If only step ① has been completed, then Bob's collateral has been locked, he needs to submit s_1' to $htlc_1^\star$ before T_1 to reclaim his collateral. If both step ① and step ③ have already been completed, then Bob's collateral and Alice's assets have been locked, Bob needs to submit s_1' to $htlc_1^\star$ before time T_1 to refund Alice's assets and reclaim his collateral. Otherwise, after time T_1, Alice can unconditionally retrieve her assets and receive Bob's collateral as compensation.

CASE 3. If Alice exits before locking her assets in step ③, steps ④⑤⑥ will not be executed. Consequently, Bob's assets and Alice's assets will remain unlocked. If step ① has been completed, similar to CASE 2, Bob needs to submit s_1' to $htlc_1^\star$ before T_1 to reclaim his collateral. If step ② has been completed, similar to CASE 1, Alice needs to submit s_2' to $htlc_2^\star$ before T_2 to reclaim her collateral.

CASE 4. If Bob exits before locking the assets in step ④, steps ⑤⑥ will not be executed. Consequently, Bob's assets will remain unlocked. If step ② has been completed, then Alice's collateral has been locked, she needs to retrieve her collateral before time T_2. For steps ① and ③, the handling strategy is the same as in CASE 2.

CASE 5. If Alice exits before revealing s in step ⑤, then Alice's assets and Bob's collateral have been locked in $htlc_1^\star$, and Bob's assets and Alice's collateral have been locked in $htlc_2^\star$. In this case, Alice needs to submit s_2' to $htlc_2^\star$ before time T_2 to refund Bob's assets and reclaim her collateral. Otherwise, after time T_2, Bob can unconditionally retrieve his assets and receive Alice's collateral as compensation. After receiving his assets, Bob needs to submit s_1' to $htlc_1^\star$ before time T_1 to refund Alice's assets and reclaim his collateral. Otherwise, after time T_1, Alice can unconditionally retrieve her assets and receive Bob's collateral as compensation.

3.4 Analysis of Key Parameters

In this section, we analyze the key parameters of FS-Swap and provide reasonable values for these parameters.

Timelocks. In FS-Swap, assuming the time required for steps ①②③④⑤ and ⑥ is t_1, t_2, t_3, t_4, t_5, and t_6, respectively, and since steps ② and ①③ can be executed concurrently (as shown in Fig. 3), the total time required for steps ①②③ is $max(t_1 + t_3, t_2)$. If the protocol starts at T_0, then T_1 must satisfy $T_1 \geq T_0 + max(t_1 + t_3, t_2) + t_4 + t_5 + t_6$, and T_2 must satisfy $T_2 \geq T_0 + max(t_1 + t_3, t_2) + t_4 + t_5$ for the swap process to be successfully completed. Furthermore, T_1 and T_2 should satisfy $T_1 - T_2 \geq t_6$ to ensure that Bob has enough time to retrieve Alice's assets after Alice has obtained Bob's assets.

Using the definition of Δ_1 and Δ_1 in Sect. 2.3, to ensure that the values of T_1 and T_2 always meet the requirements, the minimum value for T_1 is $T_0 +$

$max(2\Delta_1, \Delta_2)+\Delta_1+2\Delta_2$, and the minimum value for T_2 is $T_0+max(2\Delta_1, \Delta_2)+2\Delta_2$.

Collateral. If FS-Swap is only used to prevent lockup griefing, the collateral amounts, ϵ_1 and ϵ_2, only need to be greater than the transaction fee associated with triggering a fast refund. In this scenario, the purpose of the collateral is to incentivize a party to trigger a fast refund for their counterparty. As long as the collateral amount exceeds the transaction fee, it is always more profitable for a party to trigger a fast refund, reclaim their own collateral, rather than abstain and incur a loss of collateral. Therefore, it is enough for the collateral to exceed the transaction fee.

If FS-Swap is utilized to both prevent lockup griefing and provide compensation for parties who experience lockup griefing, the collateral amounts, ϵ_1 and ϵ_2, should be determined based on the locking time of the assets, T_1 and T_2. As $T_1 > T_2$, the collateral amount should satisfy $\epsilon_1 > \epsilon_2$. In this scenario, the collateral amounts can be determined by referencing from [17], utilizing existing option price models [14,15].

4 Security Analysis

In this section, we analyze the security of FS-Swap, demonstrating that parties' assets and collateral remain secure during the swap process. Asset security refers to the guarantee that each party either receives the counterparty's assets or retains their own assets. Collateral security ensures that as long as a party follows the protocol's instructions to execute or abort the swap, they will not lose their collateral. The possible attack actions that the counterparty can take are to delay or omit some steps.

Asset Security. For Alice, only if Bob locks his assets before $T_2 - \Delta_2$, she will continue the swap and reveal the preimage s. In this case, Alice has sufficient time ($\geq \Delta_2$) to unlock Bob's assets. Therefore, if Alice reveals s, she will definitely be able to obtain Bob's assets. If Alice does not reveal s, Bob cannot take away Alice's assets either. In this case, Alice's assets will either be refunded by Bob triggering a fast refund before T_1 or refunded after T_1, allowing Alice to retrieve her assets. In conclusion, Alice will either obtain Bob's assets or retrieve her own assets.

For Bob, he will lock his assets only after Alice has locked hers. If Bob locks his assets and Alice reveals s to claim Bob's assets, Bob will have already known s by time T_2. In this case, Bob has sufficient time ($\geq\Delta_1$) before T_1 to submit s and retrieve Alice's assets. If Alice does not disclose s, Bob's assets will either be swiftly refunded by Alice triggering a fast refund before time T_2, or he can retrieve them after T_2. In summary, Bob will either obtain Alice's assets or retrieve his own assets.

Collateral Security. For Alice, regardless of whether Bob abandons the swap or not, she has sufficient time ($\geq \Delta_2$) to retrieve her collateral. If Bob locks his assets before $T_2 - \Delta_2$, and Alice is willing to complete the swap, she has ample

time to reveal s before T_2 in order to reclaim her assets and collateral. In the event that Alice abandons the swap, she still has sufficient time to reveal s_2' to reclaim her collateral. Even if Bob has not locked assets at $T_2 - \Delta_2$, Alice still has enough time to reveal s_2' to reclaim her collateral.

For Bob, if Alice reveals s, Bob has learned s at T_2 definitely, so he can commit it to $htlc_1^\star$ before T_1 to reclaim his collateral. If the swap can not be completed, Bob must have know this at T_2, he has sufficient time to reveal s_1' to reclaim his collateral.

5 Evaluation

In this section, we performed simulated experiments to evaluate the *locking duration (LD)* of user assets when a swap cannot be completed in FS-Swap, as well as the *execution time (ET)* of the FS-Swap protocol when the swap is successfully completed. We compared these results with Norlan's Protocol [1], a classical atomic swap protocol, and Xue's Protocol [35], a classical atomic swap protocol with a premium.

5.1 Methodology

Simulation Methodology. We simulated the required blockchain systems for our experiments, along with client implementations for three distinct protocols using Python threads.

A blockchain system comprises of a transaction pool, miners, and user nodes. The transaction pool is simulated using a queue, and the mining process of miners and the random initiation of transactions by users are modeled using Poisson processes. The working process of the simulated blockchain system is as follows. User nodes publish transactions, which are then sent to the miner for validation. Upon validation, the transactions are placed in the transaction pool queue, where they await being packaged into a block by miners. When a miner obtains the right to create a block, they select a batch of transactions from the transaction pool, adhering to the first-in-first-out principle and respecting the maximum number of transactions a block can accommodate. Subsequently, the miner constructs a block containing these transactions and adds it to the ledger.

The protocol clients we simulated engage in swaps following the steps of their respective protocols. They periodically fetch on-chain transaction information at random intervals of 1–5 s. Subsequently, based on the protocol steps, the clients publish corresponding transactions according to whether they choose to abandon a swap or not.

Parameter. We simulated four blockchains with block intervals of approximately 25 s, 25 s, 6 s, and 47 s, respectively. On each chain, background transactions were sent at a frequency of approximately 67% of the maximum TPS of the chain. The maximum time limit Δ for each step execution was set to 144 times the block interval, referring to the Lightning Network project [16].

Comparative Solutions. Norlan's Protocol, as introduced in Sect. 2, serves as the basis for comparison. Xue's Protocol involves one or multiple rounds of locking premium, where increasing the number of rounds does not affect the locking time but does increase the execution time accordingly. In our experiments, we focused on the protocol with a single round of locking premium.

Experimental Setup. We set up three different chain arrangements: [25 s, 25 s], [6 s, 47 s], and [27 s, 6 s]. Under each arrangement, we conducted experimental comparisons of LD and ET for the three protocols.

Experiments on LD are conducted for FS-Swap under each chain arrangement. When a swap cannot be completed, the locking durations of user assets in Norlan's Protocol and Xue's Protocol are approximately equal to the values of the timelocks. Therefore, we only conducted experiments on LD for FS-Swap. Among the various situations where a FS-Swap swap cannot be completed, we can summarize the cases where parties' assets are locked into two categories: (1) Alice locks her assets, and then Bob exits, resulting in Alice's assets being locked; (2) Both Alice and Bob lock their assets, and then Alice exits, resulting in both Alice's and Bob's assets being locked. We conducted experiments to measure the locking duration for each party's assets under two different cases for each chain arrangement. The experimental results are presented in the "LD" row of Table 1.

Experiments on ET are performed for each of the three protocols under each chain arrangement. We conducted experiments to measure the execution time of the three protocols when the swap is successfully completed under the three chain arrangements. The experimental results are presented in the "ET" row of Table 1.

Table 1. Evaluation Result

			Norlan's	Xue's	**FS-Swap**
LD	[25s, 25s]	Alice-Alice[1]	4h	6h(+50%)	**94.8s(-99%)**
		Alice-Bob	3h	5h(+67%)	**21.7s(-99%)**
		Bob-Alice	4h	6h(+50%)	**23.5s(-99%)**
	[6s, 47s]	Alice-Alice	4.24h	6.36h(+50%)	**123.6s(-99%)**
		Alice-Bob	4h	6.12h(+53%)	**44.8s(-99%)**
		Bob-Alice	4.24h	6.36h(+50%)	**15.5s(-99%)**
	[47s, 6s]	Alice-Alice	4.24h	6.36h(+50%)	**58.2s(-99%)**
		Alice-Bob	2.36h	4.48h(+90%)	**9.6s(-99%)**
		Bob-Alice	4.24h	6.36h(+50%)	**39.7s(-99%)**
ET	[25s, 25s]		110.7s	195.6s(+77%)	**146.7s(+33%)**
	[6s, 47s]		111.3s	190.5s(+71%)	**163.7s(+47%)**
	[47s, 6s]		111.3s	188.2s(+69%)	**167.4s(+50%)**

[1] X-Y means the locking duration of Y's assets when X exits the swap.

5.2 Experimental Results

Locking Duration. Overall, in FS-Swap, the user locking duration ranges from 9.6 s to 2.1 min. When compared to Norlan's Protocol, there is a reduction of over 99% in the locking duration for each party in each case.

Specifically, when Alice exits, there is a difference in locking duration for each party's assets in FS-Swap depending on the chain arrangement. In comparison to the [25 s, 25 s] chain arrangement, the [6 s, 47 s] chain arrangement results in a longer locking duration for each party's assets, while the [47 s, 6 s] chain arrangement leads to a shorter locking duration. This suggests that the block interval of the second chain primarily influences the locking duration for both parties' assets when Alice exits.

Conversely, when Bob exits, in comparison to the [25 s, 25 s] chain arrangement, the [6 s, 47 s] chain arrangement results in a shorter locking duration for Alice's assets, while the [47 s, 6 s] chain arrangement leads to a longer locking duration. This suggests that the block interval of the first chain primarily influences the locking duration for Alice's assets when Alice exits.

Execution Time. When comparing Xue's Protocol and FS-Swap to Nolan's Protocol, both show an increase in protocol execution time, but the increase is smaller for FS-Swap. Overall, the protocol execution time only slightly increases from around 2 min to less than 3 min, which is perceived as minimal by the users.

Specifically, FS-Swap exhibits the shortest protocol execution time when the block intervals between the two chains are similar. When the block intervals differ, there is a varying degree of increase in execution time. This can be attributed to the fact that certain steps in FS-Swap are executed in parallel.

5.3 Summary

Compared to existing solutions, FS-Swap significantly reduces the asset locking time for parties in the event that the swap cannot be completed from 2–5 h to less than 2.1 min, greatly enhancing the user experience.

Regarding FS-Swap itself, it has the following characteristics: (1) When the sum of block intervals between the two chains remains constant, increasing the block interval of the first chain will decrease the asset locking time for two parties when Alice exits, but it will increase the asset locking time for Alice when Bob exits. (2) When the sum of block intervals between the two chains remains constant, increasing the block interval of the second chain will increase the asset locking time for two parties when Alice exits, but it will decrease the asset locking time for Alice when Bob exits. (3) When the block intervals between the two chains are similar, the protocol execution time is shorter upon successful swap. The difference in block intervals between the two chains will increase the protocol execution time.

6 Related Work

In 2013, a discussion about how to achieve the exchange of assets on different blockchains without relying on a trusted third party on the Bitcoin forum

attracts attention. In the disscusion, the concept of atomic swaps is introduced, and the typical atomic swap protocol is proposed by Tier Nolan [1]. Later, Herlihy et al. [19,20] extended it to support multi-party swaps. Xu et al. [21,24,30] conduct studies on the success rate of Nolan's protocol. Tsabary et al. [26,31,32,34] study on temporary censorship attack of HTLC, the underlying contract of atomic swap.

The lockup griefing problem arises from the fairness study of atomic swap. In 2018, A user with ID "ZmnSCPxj" points out that HTLC-based atomic swap protocols have optionality feature but do not charge fees for the "option" in Lightningdev mailing list [11]. Han et al. [17] argue that the optionality makes Nolan's protocol unfair and proposes Alice paying a premium to Bob to make it fair. Xue et al. [35] find that when lockup griefing (sore loser attack) is mitigated by paying a premium, premiums also affected by lockup griefing and are not compensated. They propose multiple rounds of premium locking to ensure that users will be compensated adequately if they have funds for lockup griefing, or that the amount of locked funds will be minimal. Arwen [18] proposes escrow fees to compensate decentralized exchanges for the loss of lockup griefing. [25,27] propose two separate methods to compensate parties for their premium (collateral)'s prelong locking.

7 Future Work

Extend FS-Swap to Diverse Cross-Chain Swap Types. FS-Swap is slated for expansion to support a broader spectrum of cross-chain swap scenarios, including multi-party swaps and acyclic swaps. Currently, FS-Swap's core design is primarily tailored for facilitating two-party cross-chain swaps. However, extending its capabilities to accommodate a more diverse range of cross-chain swap scenarios presents several challenges that require consideration. For example, in multi-party swaps, we need to address fairness issues related to compensating users experiencing lockup griefing, as a user's exit can trigger lockup griefing for multiple users, each with different durations of asset locking. In acyclic swaps, ensuring atomicity remains a primary challenge.

Remove Timelocks. FS-Swap will gradually transition away from the use of timelocks. In the realm of atomic swap solutions, the utilization of timelocks not only leads to issues like lockup griefing but also gives rise to security concerns, including temporary censorship attacks. Furthermore, it imposes limitations on the applicability of these solutions. While FS-Swap does reduce its reliance on timelocks through the implementation of hashlocks for refunds, it still relies on timelocks as the ultimate safeguard in rare refund scenarios. In the future, we plan to progressively eliminate the need for timelocks by incorporating mechanisms such as watchtowers to supplant their functionality.

8 Conclusion

In this paper, we introduced FS-Swap, the first atomic cross-chain swap protocol with *Fast Settlement* capability when a swap is canceled midway. FS-Swap achieves this capability by utilizing hashlocks to implement a rapid refund mechanism, further motivating users to utilize it through the introduction of collateral. In the majority of cases where a swap cannot be completed, users can swiftly reclaim their assets through this mechanism. We conducted a comprehensive analysis of FS-Swap, encompassing both assets and collateral. Our experimental evaluations involved comparing FS-Swap with two established mainstream solutions. The results demonstrated that FS-Swap significantly reduces the duration of asset locking in comparison to existing atomic swap solutions. In the future, our research will expand the applicability of FS-Swap to diverse cross-chain swap scenarios and seek ways to eliminate the use of timelocks.

References

1. Alt chains and atomic transfers. https://bitcointalk.org/index.php?topic=193281.0. Accessed 17 June 2023
2. Binance blockchain hit by $570 million hack - the New York times. https://www.nytimes.com/2022/10/07/business/binance-hack.html. Accessed 19 June 2023
3. Bitmart: Crypto-exchange loses $ 150 m to hackers - bbc news. https://www.bbc.com/news/technology-59549606. Accessed 19 June 2023
4. Block dx - decentralized exchange built by blocknet. https://blockdx.net/. Accessed 17 June 2023
5. Coinbase. https://en.wikipedia.org/wiki/Coinbase. Accessed 19 June 2023
6. Cosmos: The internet of blockchains. https://cosmos.network/. Accessed 12 Sep 2023
7. Explained: The multichain hack, July 2023. https://www.halborn.com/blog/post/explained-the-multichain-hack-july-2023. Accessed 12 Sep 2023
8. Explained: The poly network hack, July 2023. https://www.halborn.com/blog/post/explained-the-poly-network-hack-july-2023. Accessed 12 Sep 2023
9. Hashed timelocks contract. https://github.com/WeBankBlockchain/WeCross-Doc/blob/master/docs/routine/htlc.md
10. Komodo platform blockchain - home of atomicdex and kmd coin. https://komodoplatform.com/en/. Accessed 17 June 2023
11. [lightning-dev] an argument for single-asset lightning network. https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-December/001752.html. Accessed 17 June 2023
12. On-chain atomic swaps — decred blog. https://blog.decred.org/2017/09/20/On-Chain-Atomic-Swaps/, (Accessed on 06/17/2023)
13. Polkadot: Web3 interoperability — decentralized blockchain. https://www.polkadot.network/. Accessed 12 Sep 2023
14. Black, F., Scholes, M.: The pricing of options and corporate liabilities. J. Polit. Econ. **81**(3), 637–654 (1973)
15. Cox, J.C., Ross, S.A., Rubinstein, M.: Option pricing: a simplified approach. J. Financ. Econ. **7**(3), 229–263 (1979)

16. ElementsProject: Elementsproject/lightning: Core lightning - lightning network implementation focusing on spec compliance and performance. https://github.com/ElementsProject/lightning
17. Han, R., Lin, H., Yu, J.: On the optionality and fairness of atomic swaps. In: Proceedings of the 1st ACM Conference on Advances in Financial Technologies, pp. 62–75. AFT '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3318041.3355460
18. Heilman, E., Lipmann, S., Goldberg, S.: The Arwen trading protocols. In: Bonneau, J., Heninger, N. (eds.) FC 2020. LNCS, vol. 12059, pp. 156–173. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51280-4_10
19. Herlihy, M.: Atomic cross-chain swaps. In: Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, pp. 245–254. PODC '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3212734.3212736
20. Herlihy, M., Liskov, B., Shrira, L.: Cross-chain deals and adversarial commerce. Proc. VLDB Endow. **13**(2), 100–113 (2019). https://doi.org/10.14778/3364324.3364326
21. Ladóczki, B., Bíró, J., Tapolcai, J.: Stochastic analysis of the success rate in atomic swaps between blockchains. In: 2022 Fourth International Conference on Blockchain Computing and Applications (BCCA), pp. 41–46. IEEE (2022)
22. Lampson, B., Sturgis, H.: Crash recovery in a distributed data storage system. Unpublished Technical Report, Xerox Palo Alto Research Center, June 1979
23. Lightningnetwork: Lightningnetwork/lnd: Lightning network daemon. https://github.com/lightningnetwork/lnd
24. Manshaei, M.H., Jadliwala, M., Maiti, A., Fooladgar, M.: A game-theoretic analysis of shard-based permissionless blockchains. IEEE Access **6**, 78100–78112 (2018). https://doi.org/10.1109/ACCESS.2018.2884764
25. Mazumdar, S.: Towards faster settlement in HTLC-based cross-chain atomic swaps. arXiv preprint arXiv:2211.15804 (2022)
26. Nadahalli, T., Khabbazian, M., Wattenhofer, R.: Timelocked bribing. In: Borisov, N., Diaz, C. (eds.) FC 2021. LNCS, vol. 12674, pp. 53–72. Springer, Heidelberg (2021). https://doi.org/10.1007/978-3-662-64322-8_3
27. Nadahalli, T., Khabbazian, M., Wattenhofer, R.: Grief-free atomic swaps. In: 2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), pp. 1–9. IEEE (2022)
28. Poon, J., Dryja, T.: The bitcoin lightning network: scalable off-chain instant payments (2016)
29. Ricci, S., Ferreira, E., Menasche, D.S., Ziviani, A., Souza, J.E., Vieira, A.B.: Learning blockchain delays: a queueing theory approach. SIGMETRICS Perform. Eval. Rev. **46**(3), 122–125 (2019). https://doi.org/10.1145/3308897.3308952
30. Rueegger, J., MacHado, G.S.: Rational exchange: incentives in atomic cross chain swaps. In: 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), pp. 1–3. IEEE (2020)
31. Tsabary, I., Yechieli, M., Manuskin, A., Eyal, I.: MAD-HTLC: because HTLC is crazy-cheap to attack. In: 2021 IEEE Symposium on Security and Privacy (SP), pp. 1230–1248 (2021). https://doi.org/10.1109/SP40001.2021.00080
32. Wadhwa, S., Stoeter, J., Zhang, F., Nayak, K.: He-HTLC: revisiting incentives in HTLC. Cryptology ePrint Archive (2022)
33. Wikipedia contributors: Mt. gox — Wikipedia, the free encyclopedia (2023). https://en.wikipedia.org/w/index.php?title=Mt._Gox&oldid=1155286602. Accessed 17 June 2023

34. Winzer, F., Herd, B., Faust, S.: Temporary censorship attacks in the presence of rational miners. In: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 357–366 (2019). https://doi.org/10.1109/EuroSPW.2019.00046
35. Xue, Y., Herlihy, M.: Hedging against sore loser attacks in cross-chain transactions. In: Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing, pp. 155–164. PODC'21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3465084.3467904